The 11th White House Papers
Graduate Research in Cognitive and Computing Sciences at Sussex

Editor:
Fabrice P. Retkowsky

CSRP 495

October 1998

ISSN 1350-3162

UNIVERSITY OF
SUSSEX
AT BRIGHTON

Cognitive Science
Research Papers

THE ELEVENTH WHITE HOUSE PAPERS

CSRP 495

Graduate Research
in Cognitive and Computing Sciences
at Sussex

October 1998

Preface

At the Isle of Thorns (a small village situated not far from Haywards Heath), stand a few white buildings. These buildings, which sometimes act as Sussex University's conference centre, are most of the time used as a playground by rabbits. However, every year, they are disturbed by a congregation of COGS research students. In accordance with this time-honoured tradition, 1998 saw the 11th Isle of Thorns workshop, where COGS students gathered to present their work, share some ideas, and spend a lot of time socialising.

The White House Papers are a conclusion to this year's workshop. You will find here some articles, as well as some shorter papers, written by PhD students in the last few months. The aim is to show which domains we are interested in, and to give a rough idea to new students of what's to come.

We would like to thank Matthew Hennessy and the COGS Graduate Research Centre for funding the IoT workshop, as well as all the PhD students who contributed to these White House Papers, but particularly John Halloran for being the Post-Graduate Representative in Cogs, and organizing the Isle Of Thorns workshop.

Fabrice Retkowsky

To the 37, who Don't Do what they're Told

Contents

In: Retkowsky, F. P. (Ed.). (1998)
The 11th White House Papers: Cognitive Science Research Paper 495
School of Cognitive & Computing Sciences, University of Sussex

Odd bites into bananas don't make you blind
Learning about simplicity and attribute addition

Hilan Bensusan
hilanb@cogs.susx.ac.uk

School of Cognitive & Computing Sciences
University of Sussex
Brighton
BN1 9QH

Once upon a time there was a little girl named Emma. She had never eaten banana in all her life nor had she ever taken a journey on a train. On one occasion circumstances made necessary for her to journey from New York to Pittsburgh alone. To relieve Emma's anxiety her mother gave her a large bag of bananas to eat on her railway journey west. At Emma's first bite of her banana, the train plunged into a tunnel. At the second bite, the train broke into the daylight again. Emma, being a bright little girl, takes a third bite. Lo! Into a tunnel. A fourth bite and into the daylight again. And so on all the way to Pittsburgh (and all the way to the bottom of the bag of bananas). Is Emma justified in saying to the people who met her at the station, "Every odd bite of a banana makes you blind; every even bite puts things right again?" N. H. Hanson, p. 359

Abstract This paper shows how a meta-learning technique can be applied to decisions about pruning and representation adequacy. It describes a meta-learning technique that uses unpruned decision trees and information from decision tree construction to describe the learning tasks. Based on previous experience, a meta-learner decides which attribute addition strategy is the more appropriate for a given new learning problem. The technique is applied to simplicity and representation issues. In the simplicity camp, it is used to decide when to prune, how much pruning is appropriate and what the best pruning technique is for a given learning task. In constructive induction, it is used to decide between a pool of alternative new attribute constructors. Results suggest that induction on the connection between problems and simplicity and representation biases improves learning performance

This paper was published in the *Proceedings of the 10th ECML workshop on upgrading learning to the meta-level: model selection and data transformation*, Chemnitz, 1998.

Introduction

Wär nicht das Auge sonnenhaft, die Sonne koennt'es nie erblicken. Goethe, Zahme Xenien, Werke, Weimar 1887-1918, bk 3, 1805.

Human induction is not hopeless because it relies on the similarity between different inductive problems. We induce on data relying on our inductive hypotheses about inductive problems. Recent work on transfer of learning, ensembling and multi-strategy learning clearly indicate that using different biases to deal with different problems is the key to attain better learning performances. Meta-learning using a class of previous learning experiences relies on the conjecture that there are connections between biases and problems. Clearly enough, these connections are relative to the environment the problems come from. Within an environment, some bias are more appropriate than others for some problems and meta-learning holds that this can be learned.

The commonsense message from Emma's story above is that successful induction requires experience in an environment. Emma hasn't perform enough inductions to realise that in our environment odd and even bites of a fruit are not likely to affect anybody's sight. Emma, the bright little girl, conceives of a causal relation between odd and even bites into bananas and her ability to see. Anyone with more experience in our environment, even without having ever eaten banana or taken a journey on a train, would resist Emma's conclusion. Their learning biases is attuned to their world.

This paper explores the advantages of using a bias chosen by a meta-learner to take decisions about pruning and to improve the input representation. It describes how a meta-learning system (THE ENTRENCHER, see (Bensusan, 1998; Bensusan & Williams, 1997)) that selects a bias from a pool can be used to find an appropriate attribute to be added to the original attribute vector and can decide if, how and how much pruning is needed. Since no unique pruning strategy and no unique new attribute constructor can be assumed to be a universal bias, THE ENTRENCHER is invoked to inductively choose an appropriate bias.

Section 1 describes the components of THE ENTRENCHER. and the MONKSPACE, a class of problems similar to the original MONK problems (Thrun, Bala, Bloendorn, Bratko, & etc., 1991) and that is used in the experiments to be reported. Section 2 introduces the debates on simplicity and the different alternative pruning strategies. Section 3 reports experiments and results concerning simplicity choices. Section 4 introduces the issue of improving input representation and describes alternative new attribute constructors. Section 5 reports results about representation improvement and section 6 concludes the paper.

1 The system

THE ENTRENCHER system[1] learns to choose the best learning bias for a learning task among the ones provided in a bias pool. Therefore, it performs a kind of meta-learning. Chan & Stolfo (Chan & Stolfo, 1993) define meta-learning as "learning from information generated by learners". THE ENTRENCHER uses the decision tree generated from the training set and related information to describe a learning task. THE ENTRENCHER then performs a supervised meta-learning on a set of classified problems divided into training and test sets. The system acts as follows:

1. Applies all the learners in the bias pool to the training problems and tests their performance;
2. Classifies the problems in terms of the best performing bias;
3. Applies a meta-learning procedure to the classified training set of problems, generating a bias classifier;

The performance of the system is tested by assessing the average accuracy achieved by the bias classifier in a set of test problems [2]

THE ENTRENCHER performs a sort of hypothesis-driven meta-learning as it uses output hypotheses to describe the problems it aims to learn about. The system should apply a **baseline learner** to both training and test problems in order to get the values for the descriptor vector that describes the task. The baseline learner produces a compression of the training set. The system then comprises the following components:

A baseline learner generates hypotheses that are the working representations of the input problems.

[1] This is a development of the system described in (Bensusan & Williams, 1997).

[2] Notice that the meta-learner misclassifications might have different impact on the overall system performance.

A problem descriptor generates values for the descriptor vector that is partly based on the working representation of the problem and partly on the way the baseline learner generates the working representation.

A bias pool manager classifies the training problems and applies the learners in the bias pool to the test problems according to the meta-learning classifier. The training problems are classified in terms of the most accurate bias or, in case of more than one bias with the same accuracy, in terms of simplicity [3].

A meta-learner generates a bias classifier that selects a bias for each test problem among the ones in the bias pool.

THE ENTRENCHER uses a TDIDT (top-down induction of decision trees) procedure as baseline learner. In fact, this baseline learner is similar to a non-pruning C4.5 (Quinlan, 1993). The **baseline learner** provides a working representation of the problems as a decision tree which is encoded in terms of a descriptor vector that consists of the following real-valued descriptors:

Tree nodes per attribute: The proportion of tree nodes per attribute.

Tree nodes per instance: The proportion of tree nodes per training instance.

Average leaf corroboration: The average strength of support of each tree leaf. Support is measured by the number of training instances that correspond to the paths terminating in each leaf.

Average gain-ratio difference: The gain-ratio is an information-theoretic measure used in the C4.5 decision tree building. It measures the goodness-as-a-splitting-point of an attribute. This descriptor indicates the difference in goodness-as-a-splitting-point between the attributes at the first splitting point of the tree building process.

Maximum depth of the tree: This descriptor measures the size of the longest path from the root to a leaf. The depth of a path is calculated by measuring how many edges link the root to the leaf.

Number of repeated nodes: This descriptor measures how many repeated attributes appear in the tree.

Shape of the tree: The probability of arriving at each leaf given a randomly chosen path from the root to the leaf. This probability $p(N_i)$ of arriving at node N_i among the m sibling nodes from the ancestor N_A is given by:

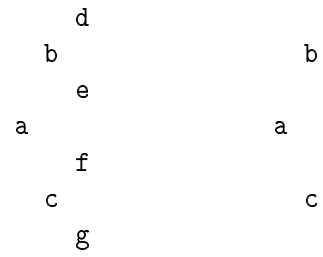
$$p(N_i) = p(N_A)/m$$

The shape is then measured from the probability of the leaves $p(L_j)$ in the following way, given a tree with x leaves:

$$-\sum_{j=0}^{x} p(L_j)\log_2(p(L_j))$$

[3] Notice that the meta-learning system itself assumes a simplicity bias. A meta-learner is a learner about learners and as such, it requires biases and, *a priori*, a simplicity bias is at least as good as any other.

The following two trees have shape of 2 and 1 respectively.



Homogeneity: The number of leaves divided by tree shape.

Balance of the tree: Given all the possible values V_i for $p(L_j)$, calculate $G(V_i)$ as

$$G(V_i) = n * V_i$$

where n is the number of times V_i occurs in the set of all the leaves of the tree. The balance is then measured by the following sum for all the x possible values for $p(L_j)$:

$$\sum_{j=0}^{x} G(V_j) \log_2(G(V_j)$$

The trees used to illustrate the shape descriptor both have balance 0.

Internal symmetry: How many subtrees with more than two (possibly internal) nodes repeated in the tree.

The meta-learner is then applied to the descriptor values for the training problems and a classifier is generated. When C4.5 is used as a meta-learner, the classifier is a decision tree. The following is a fragment of a meta-learning output classifier that decides between biases 1, 2 and 3 from the bias pool.

```
7.shape <= 3.91 :
|   7.shape <= 3.77 : 2 (11.0/1.0)
|   7.shape > 3.77 : 3 (2.0/1.0)
7.shape > 3.91 :
|   1.nodes per att <= 6.17 : 1 (2.0)
|   1.nodes per att > 6.17 :
|   |   1.nodes per att > 9.5 : 2 (11.0)
|   |   1.nodes per att <= 9.5 :
|   |   |   4.diff <= 0.01 : 1 (3.0)
|   |   |   4.diff > 0.01 : 2 (10.0/1.0)
```

The experiments reported in this paper involve learning tasks of different degrees of difficulty drawn from an artificial domain designed to test attribute addition algorithms. This artificial domain, that I shall refer hereafter as the MONKSPACE, consists of MONK-like problem types, including the 3 MONK

problems reported in (Thrun et al., 1991). Given the original 6 attributes of the MONK problems, there are 2^{432} possible classifications. For the current experiments, a number of classifications were chosen and 10 problems, composed by training and a test sets, were constructed for each classification. The meta-learning system was trained on an increasing number of problems and tested on different test sets of problems.

2 Simplicity biases

The Occam razor is a popular and widespread bias: whenever possible, prefer simplest hypotheses. In many contexts, both scientists and laymen would appeal to simplicity to decide between different alternatives. One can ground the use of the razor on some sort of analysis of induction by appealing to the greater plausibility (or prior probability) of simpler hypotheses (Laplace, 1952). However, measures of simplicity are bound to be representation relative and therefore simplicity becomes a form of soft bias (Utgoff, 1986; Dieterich & Kong, 1995).

In the context of decision tree induction, the Occam's principle is often mentioned to justify a preference for simpler trees. This has led to the idea that pruned decision trees are likely to exhibit better test set accuracy. However, recent experimental work has shown that larger trees sometimes perform better (Schaffer, 1993; Murphy & Pazzani, 1994; Spears & Gordon, 1994; Webb, 1996). The point was not only to show that the simplicity bias fails for some tasks but also to suggest that some of these tasks might belong to the so-called real world tasks class. For example, Webb (Webb, 1996, 1997) reports that his grafting procedures – inductive processes that add complexity to a decision tree by considering global rather than local information – enhance accuracy in the widely used UCI repository tasks.

If Occam razor is not a graal, we need to decide when to use it. We also need to establish, for each learning task, how and how much should we prune. There were various theoretical attempts to establish conditions under which simpler hypotheses are better, both in general (Blumer, Ehrenfeucht, Haussler, & Warmuth, 1987) and within the context of decision tree induction (Quinlan & Rivest, 1989; Fayyad & Irani, 1990; Gamberger & Lavrac, 1997). These are to be considered within their own established limits and as worst case results. Under these limitations, these results can seldom be applied to the practitioner's decisions about how simple should the conjectured decision tree be.

It seems natural, therefore, to invoke meta-learning to take these decisions. A meta-learning strategy is applied to learn pruning requirements for a class of tasks by examining various pruning strategy performances in similar tasks. For this purpose, the bias pool may be composed of the various pruning options available to decision tree classifiers. These include options as to whether or not to prune, how much pruning is appropriate and what of the available pruning strategy to apply.

Different pruning strategies have been proposed[4], with different advantages each. Quinlan's C4.5 system popularised error-based pruning, the weakness of its heuristic-based justification notwithstanding[5]. A major alternative to the error-based approach is the cost-complexity pruning approach that has been adopted by some TDIDT systems such as Quinlan's ID3 (Quinlan, 1986). It can be seen as a direct application of the minimal encoding principles, that involve the search for a balance between simplicity and consistency.

For the experiments reported here, Quinlan's error-based strategy and a cost-complexity pruning strategy are considered. Quinlan's strategy is based on viewing the classification errors in the training sets as events in the sample of training cases. The pruned tree is generated from the complete original tree, as opposed to the prune-while-you-build approach taken by learners such as ID3. Also, Quinlan's

[4]Refer to (Mingers, 1989) for an a slightly outdated overview and an experimental comparison.

[5]Recently an alternative pessimistic pruning was reported to have similar performance and to enjoy sounder justification (Mansour, 1997).

pruning requires no independent validation data set. Different levels of error-based pruning are given by different confidence levels in the reliability of the sample as an indicator of how many errors are to be expected by a given decision node (Quinlan, 1993). The other strategy considered is a cost-complexity technique that has the two features of Quinlan's strategy mentioned above: the pruned tree is generated from the complete original tree and no validation data set is required.

The cost-complexity pruning technique considered also admits different levels of pruning according to the different values of the acceptable cost threshold. Cost measures how much, in terms of training set coverage, is gained by each new split introduced. In addition, cost increases with the number of splits already made – with respect to how far down in the tree the subtree is. Pruned trees are generated by replacing subtrees by a leaf labelled by the majority class that covers all but exception ($E_{majority}$) cases of the subtree coverage ($X_{subtree}$), whenever the cost of the subtree is greater than the acceptable cost threshold. Let $D_{subtree}$ be the depth of a subtree, D_{tree} the depth of the whole tree and $S_{subtree}$ the size in terms of the number of nodes of the subtree. Cost is then defined as follows:

$$Cost_{subtree} = \frac{((D_{tree} - D_{subtree} + 1)^2) * S_{subtree}}{E_{majority}/X_{subtree}}$$

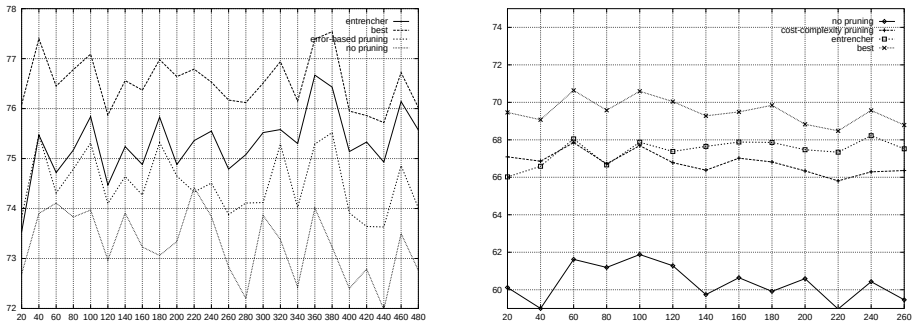
The quadratic penalty $(((D_{tree} - D_{subtree} + 1)^2))$ is designed to substantially increase the costs of the lowest subtrees since they are likely to concern details and therefore to overfit. Pruning proceeds from the leaves to the root of the tree.

3 Learning simplicity biases

In the experiments to be reported, 100 classifications from the MONKSPACE were randomly chosen and 10 problems, composed by training and a test sets of 50 instances, were constructed for each classification. The meta-learning system was trained on an increasing number of problems and then tested on different test sets of problems randomly drawn from the 1000 existing problems. The performances were compared to the best possible pruning option available for each problem and with the fixed different pruning options. In the graphs that report the experiments, the Y-axis is the accuracy in the test sets of 50 problems and the X-axis represents the training set sizes. Notice that the latter is relevant for the performance of THE ENTRENCHER system only since the fixed pruning options don't make any use of the training problems. Each point represents the average of 10 runs under a fixed training set size.

For each pruning strategy considered, a first question to ask is when pruning will enhance performance. In the first experiment, the meta-learning system is applied to decide between pruning with confidence level of 25%, C4.5's default value, and leaving the original C4.5 tree unpruned. In another experiment, the appropriateness of cost-complexity pruning has been considered. Here the meta-learner decides between pruning with a fixed acceptable cost threshold of 150 and leaving the original tree unpruned. Figure 1 shows the result. The two bottom lines represent pruning and no pruning. In the case of cost-complexity, unpruned trees have the worse performance all along whereas in the error based pruning case, the bottom line is the performance of pruned trees. In both cases the top line is the best pruning option for each problem and the second line approaching it is the performance of THE ENTRENCHER.

A second question in connection to simplicity is how much is good enough. Different degrees of pruning generate hypotheses with different degrees of simplicity. To learn the best simplicity bias for a problem amounts to find out how fine-grained the information contained in a training set is. Figure 2 plots the performance of the meta-learning system when selecting between different confidence levels for error-based pruning (0, 25, 50, 75). Figure 3 plots the curves for selecting the amount of cost-complexity pruning that the 50 test problems require by considering different acceptable cost thresholds (50, 150, 250).



(error-based) (cost-complexity)

Figure 1: Learning when to prune

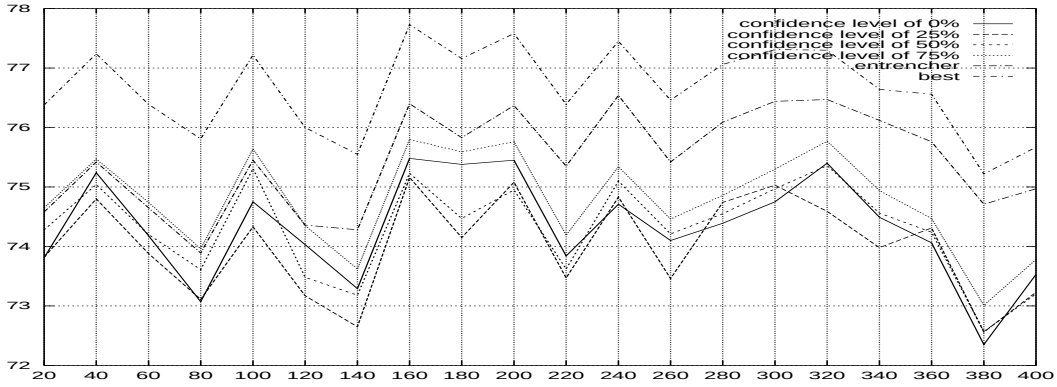


Figure 2: Learning how much error-based pruning is required

A last question concerning simplicity is related to the choice between different pruning strategies. Among the 1000 learning problems considered in these experiments, the average performance of the three main pruning options considered – C4.5 trees with no pruning, cost-complexity pruning with acceptable cost threshold of 150 and error-based pruning with confidence level of 25% – are roughly similar, as shown in table 1.

The similar average performance of the three options suggests that a fixed option for all the problems is not likely to be the best alternative. In fact, an informed choice between these options can be provided by the use of THE ENTRENCHER, as figure 4 reports. As the training set size increases THE ENTRENCHER's curve approaches "best" and when the training set size is greater than 160, the system consistently outperforms the fixed pruning alternatives.

	No pruning	Cost-complexity pruning	Error-based pruning
Average peformance	71.96	68.59	72.17
Standard deviation	8.49	8.29	8.50

Table 1: Performance of the pruning options in all the problems

4 Representation bias

The input representation of a problem essentially influences the overall learning performance (Craven & Shavlik, 1995). Since learning biases deal in representations, a good amount of effort has been put in finding appropriate strategies to improve the input description of a problem such that learning becomes easier for a given bias. This effort is often conceived as part of the learning process since a better input representation is expected to emerge from the interaction between the problem and the learner (Matheus, 1989; Rendell & Cho, 1990; Wnek & Michalski, 1994). Learners that have the ability to search for different input descriptions are often called constructive learners. Constructive induction is now a common strategy to improve learning performance as one can easily see that the ability to redescribe the problems often increase the VC dimension of a learning system [6].

In this paper, I shall consider different strategies for attribute addition, a form of constructive induction. The fundamental element of a system that re-represents by adding new attributes is a set of operators – often called constructors – to be applied to current attributes in order to produce the attributes to be added. Many systems have a fixed set of constructors (Pagallo & Haussler, 1990; Murphy & Pazzani, 1991). As a consequence, they have a fixed constructive bias and weaken the overall bias of the system always in the same manner.

THE ENTRENCHER can be invoked to choose between different constructors within a constructive bias pool. Therefore, the system consults its previous experiences on related tasks to decide which operator is the most appropriate one. Also, meta-learning helps detecting when the addition of new attributes is likely to ease learning. Detection of the need for new input representation and selection of an appropriate constructor is therefore performed by the meta-learner that classifies new tasks in terms of its input representation requirements and uses a training set of learning tasks as a source of knowledge.

For the experiments to be reported here, five constructors were considered. Each constructor is an application of a function that takes the attribute vector A of $|A|$ attributes understood as a function of a numbered example i that produces values $A(i)$ and a value v. The value for each n of the $|A|$ attributes in the vector in example i is referred as $A_n(i)$. Each constructor produces N new attributes where N is the number of possible values among all the attributes in A for constructors $C1$, $C2$, $C3$, $C4$. For $C5$, where two values v and v' are taken as inputs, $N = v * v'$. The functions are defined as follows.

$$C1(v,A,i) = \begin{cases} A_n(i) & \text{if } n < |A| \\ v & \text{otherwise} \end{cases}$$

$$C2(v,A,i) = \text{the number of times } v \text{ appears in } A(i)$$

$$C3(v,A,i) = \begin{cases} 1 & \text{if } v \text{ appears in } A(i) \text{ an odd number of times} \\ 0 & \text{otherwise} \end{cases}$$

$$C4(v,A,i) = \begin{cases} 1 & \text{if there is a pattern: } v, \text{ another value, } v \text{ in } A(i) \\ 0 & \text{otherwise} \end{cases}$$

$$C5(v,v',A,i) = \begin{cases} 1 & \text{if } v' \text{ appears as many times as } v \text{ in } A(i) \\ 0 & \text{otherwise} \end{cases}$$

[6] Refer to (Blumer, Ehrenfeucht, Haussler, & Warmuth, 1989) about VC dimension

THE ENTRENCHER will then have to choose between the 5 constructors and the simple baseline learner with no attribute added. If no new attribute is taken to be needed, the original representation of the problem is considered to be such that none of the existing constructors could improve it.

Now, I shall refer to a problem as falling into the domain of expertise of a given constructor whenever the new attributes built by the constructor promote a learning test accuracy of at least 95%. Of course, a problem can fall into the domain of expertise of more than one constructor.

5 Learning to choose constructors

For these experiments, 100 MONKSPACE classifications designed to generate problems that would fall into the domain of expertise of each constructor were generated. For each classification, 10 problems with 124 training instances were generated. Performance of the constructors were measured in a test set composed of the whole classified instance space (432 instances). The meta-learning system was trained on an increasing number of problems and then tested on non-overlapping test sets of problems. The performances was again compared to the best possible bias in the constructor pool. Here, as before, the Y-axis is the accuracy in the test sets and the X-axis represents the training set sizes. Once again, each point represents the average of 10 runs under a fixed training set size.

Now, since there are overlaps between domains of expertise, the difference of performance between the best and the second best constructor may be so small such that no clear signal exists for THE ENTRENCHER to learn. The experiments show that a minimal performance difference is required for a meta-learning with THE ENTRENCHER to take place. Figure 5 shows that, although the performance of the system is consistently better than the average performance of the six biases (five constructors and the baseline learner), no improvement of bias choice takes place with the increase of training size when the minimal performance difference between the two best learners is of 5%. When the minimal performance difference is set to 10% (figure 6), the system's performance clearly converges towards "best" as training size increases.

6 Conclusion

> *When a chemist announces the existence and properties of a newly discovered substance, if we confide in his accuracy, we feel assured that the conclusions he has arrived at will hold universally, though the induction be founded but on a single instance.* J. S. Mill (Mill, 1843), III, III, 3.205

THE ENTRENCHER successfully learns connections between tasks and biases. The results above show that the appeal to previously learned problems is an appropriate strategy to improve performance (or to decrease the number of instances required for induction). Tasks, as described by the system, appear to have properties on the basis of which it is possible to learn to choose an appropriate bias. In both cases of learning pruning strategies and performing constructive induction, the system, when sufficiently trained, performs better than a fixed bias. Future work includes comparing meta-learning with THE ENTRENCHER to results achieved by other multi-bias strategies, especially with ensembling (Dietterich, 1997). In any case, meta-learning seems to have a safe place among a repository of good mechanical induction procedures.

THE ENTRENCHER implements a decisive feature of human successful inductive hypotheses: their roots are in our inductive knowledge of the environment. Emma didn't lack anything but inductive experience. Experience enough to tune her biases to the problems that she's likely to find. The chemist referred by Mill is reliable because he had been trained to make use of inductive strategies that are appropriate for the problems that he's likely to find. Of course, if the bias is strong and appropriate, a

single instance might be enough. Meta-learning can therefore be seen as a force driving machine learning from Emma's naive conjecturing to the informed induction of Mill's chemist .

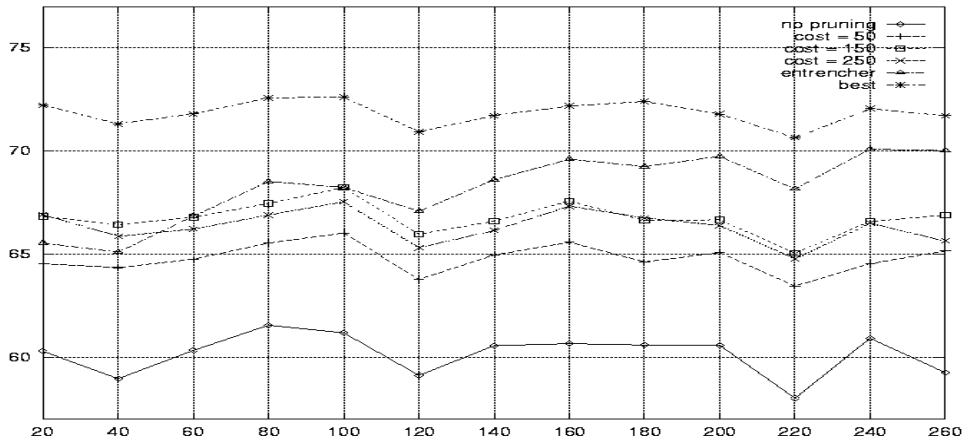


Figure 3: Learning how much cost-complexity pruning is required

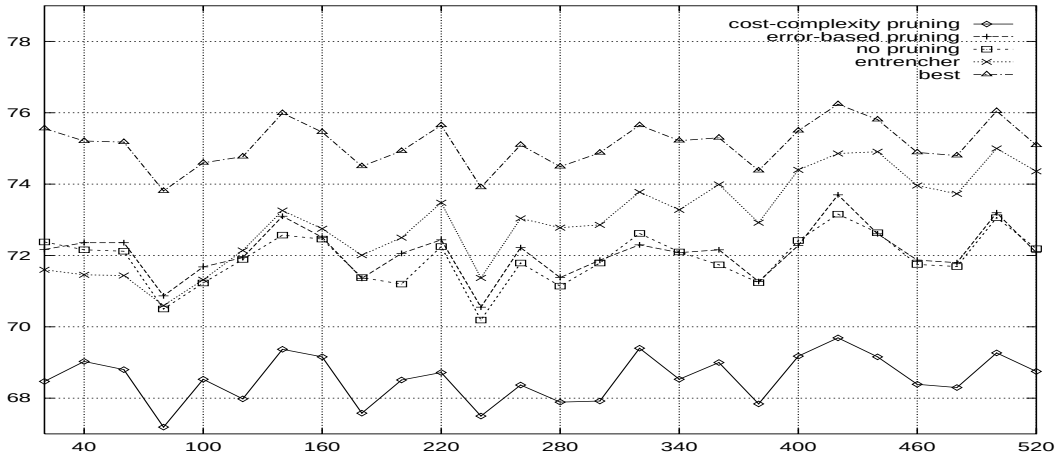


Figure 4: Learning how to prune

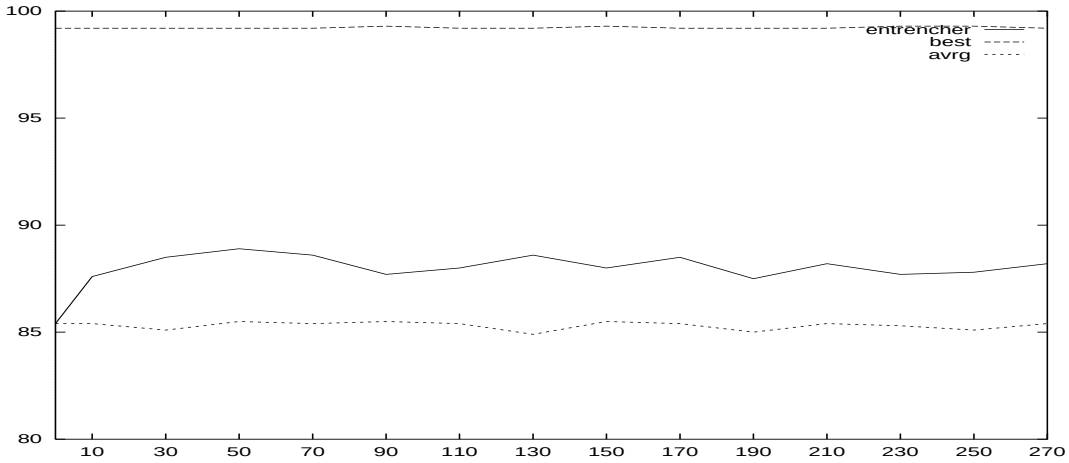


Figure 5: Learning constructors with minimal accuracy difference of 5%

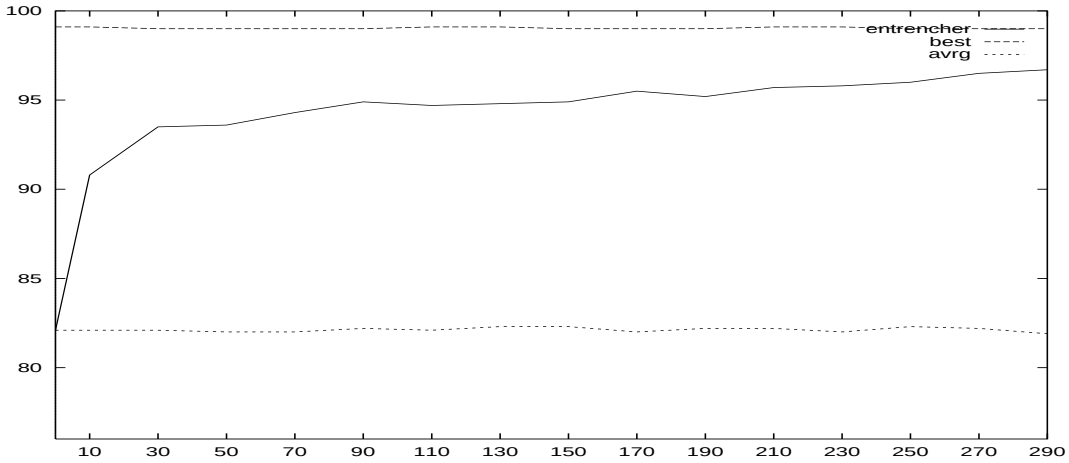


Figure 6: Learning constructors with minimal accuracy difference of 10%

References

Bensusan, H. (1998). God doesn't always shave with Occam's Razor – learning when and how to prune. In Nédellec, C., & Rouveirol, C. (Eds.), *Proceedigs of the 10th European Conference on Machine Learning*, pp. 119–124 Berlin. Springler.

Bensusan, H., & Williams, P. (1997). Learning to learn boolean tasks by decision tree descriptors. In Someren, M. V., & Widmer, G. (Eds.), *Poster Papers - 9th European Conference on Machine Learning*, pp. 1–11 Prague, Czech Republic.

Blumer, A., Ehrenfeucht, A., Haussler, D., & Warmuth, M. (1989). Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, *36*(4), 929–965.

Blumer, A., Ehrenfeucht, A., Haussler, D., & Warmuth, M. K. (1987). Occam's razor. *Information Processing Letters*, *24*(6), 377–380.

Chan, P., & Stolfo, S. (1993). Experiments on multistrategy learning by meta-learning. In *Proceedings of the second international conference on information and knowledge management*, pp. 314–323.

Craven, M., & Shavlik, J. (1995). Investigating the value of a good input representation. In Petsche, T., Judd, S., & Hanson, S. (Eds.), *Computational Learning Theory and Natural Learning Systems*, Vol. 3. MIT Press, Cambridge, MA, USA.

Dieterich, T. G., & Kong, E. B. (1995). Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. Tech. rep., Department of Computer Science, Oregon State University, Corvallis, OR, USA.

Dietterich, T. G. (1997). Machine learning research: Four current directions. In press.

Fayyad, U. M., & Irani, K. B. (1990). What should be minimized in a decision tree?. In Dietterich, Tom; Swartout, W. (Ed.), *Proceedings of the 8th National Conference on Artificial Intelligence*, pp. 749–754 Cambridge, MA, USA. MIT Press.

Gamberger, D., & Lavrac, N. (1997). Conditions for Occam's Razor applicability and noise elimination. In van Someren, M., & Widmer, G. (Eds.), *Proceedings of the 9th European Conference on Machine Learning*, pp. 108–123. Springer.

Laplace, P. S. (1952). *A Philosophical Essay on Probabilities*. Dover, UK.

Mansour, Y. (1997). Pessimistic decision tree pruning based on tree size. In *Proc. 14th International Conference on Machine Learning*, pp. 195–201. Morgan Kaufmann.

Matheus, C. (1989). Feature construction: an analytic framework and an application to decision trees. Tech. rep. CSR-89-1559, Department of Computer Science, University of Illinois, Urbana, Urbana, IL, USA.

Mill, J. S. (1843). *A system of logic*. Longmans, London, UK.

Mingers, J. (1989). An empirical comparison of pruning methods for decision tree induction. *Machine Learning*, *4*, 227–243.

Murphy, P., & Pazzani, M. (1991). Id2of3: Constructive induction of m-of-n concepts for discriminators in decision trees. In *Proceedings of the Eight International Machine Learning Conference* San Mateo, CA, USA. Morgan Kaufmann.

Murphy, P., & Pazzani, M. (1994). Exploring the decision forest: An emprical imvestigation of occam's razor in decision tree induction. *Journal of Artificial Intelligence Research*, *1*, 257–275.

Pagallo, G., & Haussler, D. (1990). Boolean feature discovery in empirical learning. *Machine Learning*, *5*, 71–100.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, USA.

Quinlan, J. R., & Rivest, R. L. (1989). Inferring decision trees using the Minimum Description Length Principle. *Inform. Comput.*, *80*(3), 227–248. (An early version appeared as MIT LCS Technical report MIT/LCS/TM-339 (September 1987).).

Rendell, L., & Cho, H. (1990). Empirical learning as a function of concept character. *Machine Learning*, *5*, 267–298.

Schaffer, C. (1993). Overfitting avoidance as bias. *Machine Learning*, *10*, 113–152.

Spears, W. M., & Gordon, D. F. (1994). A simpler look at consistency. Tech. rep. AIC-94-018, Naval Research Laboratory, Navy Center for Applied Research on Artificial Intelligence, Washington, DC, USA.

Thrun, S. B., Bala, J., Bloendorn, E., Bratko, I., & etc., B. C. (1991). The monk's problems - a performance comparison of different learning algorithms.. Tech. rep. CMU-CS-91-197, School of Computer Science, Carnegie-Mellon University., Pittsburgh, PA, USA.

Utgoff, P. (1986). Shift of bias for inductive concept learning. In *Machine Learning: An Artificial Intelligence Approach, volume III*. Morgan Kaufmann, San Mateo, CA, USA.

Webb, G. (1996). Further experimental evidence against the utility of occam's razor. *Journal of Artificial Intelligence Research*, *4*, 397–417.

Webb, G. (1997). Decision tree grafting. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pp. 846–851 San Mateo, CA, USA. Morgan Kaufman.

Wnek, J., & Michalski, R. S. (1994). Hypothesis- driven constructive induction in AQ17: A method and experiments. *Machine Learning*, *14*, 139–169.

In: Retkowsky, F. P. (Ed.). (1998)
The 11th White House Papers: Cognitive Science Research Paper 495
School of Cognitive & Computing Sciences, University of Sussex

Configurational and featural processing: Two routes to the recognition of faces ?

Stephan Collishaw
stefanc@cogs.susx.ac.uk

School of Cognitive & Computing Sciences
University of Sussex
Brighton
BN1 9QH

1 Context and aims

The distinction between configurational and featural information is influential in most accounts of face processing (e.g.: Bruce, 1988; Tanaka & Farah, 1993; Bruce & Young, 1998). These two possible routes of processing have been studied by examining the effects of a number of global manipulations. Blurring appears to impair featural information whilst leaving configurational information relatively preserved (Costen et al., 1994), and scrambling and inversion show the opposite pattern of effects: they appear to impair configurational processing to a greater extent than featural processing (Yin, 1969; Rhodes et al., 1993).

The aim of experiment 1 was examine the impairments produced by combinations of three manipulations - blurring, scrambling, and inversion, on a face recognition task involving the identification of famous celebrities. It was hypothesised that: 1) Any one manipulation would lead to some impairment in recognition performance; 2) Manipulations would show interactive effects if applied together. Scrambling and inversion both affect configurational processing and were therefore expected to have relatively little (if any) additional impact over the effects of each of these manipulations alone. Blurring and scrambling, and blurring and inversion, were expected reduce recognition rates disproportionately, possibly to that expected by chance, as both configuration-based and feature-based processes are disrupted.

This experiment provides a long overdue test of a 'two-process' theory of face recognition where configurational and featural processing constitute relatively independent routes to the identification of a face. According to such a theory access to both routes is optimal, access to either route is sufficient, but at least one route must be available for recognition to occur.

Experiment 2 examines the effects of the same manipulations on the recognition of previously unfamiliar faces. Neuropsychological and experimental evidence suggests a possible dissociation between the processing of unfamiliar and familiar faces (Malone et al., 1982; Young, 1984). By comparing the effects of the manipulations employed in the two experiments this study addresses whether the recognition of unfamiliar faces (once seen) relies to a greater extent on either configurational or featural facial information than does the recognition of highly familiar faces.

2 Method

2.1 Sample

140 subjects (68 men and 72 women) were recruited for participation in experiments 1 and 2. Subjects ranged in age from 18 to 50 years (mean = 28.7 years).

2.2 Design

Both experiments consisted of seven conditions. These involved the recognition of faces which were:

- 1) Unmanipulated;
- 2) Blurred (Gaussian filter, r = 10 pixels);
- 3) Scrambled (re-arrangement of 5 horizontal face strips);
- 4) Inverted;
- 5) Scrambled and Inverted;
- 6) Blurred and Scrambled;
- 7) Blurred and Inverted.

A between subjects design was employed, and 20 subjects were randomly assigned to each condition. Subjects were assigned to the same condition for both experiments, and the order in which subjects completed the two tasks was counterbalanced.

2.3 Materials

For experiment 1, photographs of 22 celebrities, famous in the UK, served as targets. Distractor faces were individually matched to target faces on the basis of age, hair colour and length, and quality of image.

For experiment 2, two photographs were taken of 44 students at University College London. Half the faces were randomly designated targets, and one (unmanipulated) photo was used in a study phase, whilst the other (manipulated) was shown during the test phase. One photo of each of the remaining faces was matched to each target face, and these served as distractors in the test phase.

2.4 Procedure

Faces in experiment 1 were presented to subjects one at a time in a random order. Subjects had to decide whether each face was a celebrity or a non-entity. The accuracy of their choice as well as their reaction time was recorded. In experiment 2, subjects first received a study phase, in which they viewed unmanipulated versions of the target faces, one at a time for 3 seconds. The test phase of the experiment paralleled that used in experiment 1.

3 Results

3.1 Statistical Analysis

D-prime scores and RT data (for correct responses) were analysed using one-way analyses of variance and Tukey's HSD tests. Mixed-design analyses of variance were used to test for a possible interaction between the effects of condition and the effects of experimental task.

3.2 Experiment 1

The results provided strong support for the experimental hypotheses. Blurring, scrambling, and inversion each led to significant reductions in subjects' d-prime scores, when compared to subjects who received unmanipulated faces. However, in each condition performance was still significantly above chance,

indicating that each type of manipulation preserved information, capable of leading to the successful identification of target faces. Secondly, the three manipulations did not show additive effects when applied in combination. Instead, recognition accuracy was no lower for subjects in the scrambled and inversion condition than for subjects who received faces that had only been scrambled or only inverted. Blurred-Scrambled and Blurred-Inverted faces, on the other hand, were not recognised above chance level.

3.3 Experiment 2

The results of experiment 2 provided a partial replication of these findings. Blurring, scrambling, and inversion all led to some impairment of recognition performance. Scrambling and inversion together led to no impairment over and above that produced by scrambling or inversion alone, whilst recognition blurred-inverted faces was significantly lower, and did not differ from that expected by chance. In contrast to experiment 1, recognition accuracy in the blurred-scrambled condition was marginally better than that expected by chance.

Comparisons of performance in experiments 1 and 2 showed that while unfamiliar face recognition was harder in general, the relative disruption produced by all manipulations was greater for famous faces. However, there were only minor differences in the pattern of impairments produced by the manipulations employed. In particular, the effects of impairment to configurational information (through scrambling and/or inversion) relative to the impairment of featural information (through blurring) did not differ for famous and unfamiliar faces. Secondly, a comparison of subjects' response times showed no significant interaction between condition and task ($p > 0.4$). The pattern of response times over conditions 1 to 7 was very similar in experiments 1 and 2.

4 Conclusions

The results of these experiments provide strong evidence for the experimental hypotheses. Blurring on the one hand and scrambling and inversion on the other appeared to affect two rather different sources of information used by the face recognition system. It is argued that configurational information and featural information offer relatively independent alternative pathways to the identity of a face. Comparisons of the results of experiments 1 and 2 suggest that these findings are likely to be generalisable across the spectrum of familiarity (from once-seen to celebrity). However, further studies are called for, as experiment 2 provided only a partial replication of the results of experiment 1. No evidence was found for a dissociation between familiar and unfamiliar faces in terms of the relative importance of configurational and featural routes of processing.

In: Retkowsky, F. P. (Ed.). (1998)
The 11th White House Papers: Cognitive Science Research Paper 495
School of Cognitive & Computing Sciences, University of Sussex

From an Encyclopedia to a Teaching Space: Using the Web in Schools

Ann Light
annl@cogs.susx.ac.uk

School of Cognitive & Computing Sciences
University of Sussex
Brighton
BN1 9QH

If the perceptions of a society are to be changed, then schools are a good place to start the process. Every child passes through them and, in theory at least, every child has access to a similar range of experiences while passing through.

This short paper is a critique of how an interactive medium is being presented to schools - and, consequently, in schools, motivated by a wish to see a more hands-on and participatory approach taken up. This, I believe, is the key to greater understanding and control of the medium and, with it, greater potential for the equal society enshrined in the right of education for all.

It is only in the second half of the 20th century that people have been brought up on a diet of mass media such as radio, television and magazines in the home. Prior to that, information and entertainment channels were more diffuse and depended to a greater extent on local networks. The increasing centralisation and dissemination that accompanied the development of mass media has led to changes in audience behaviour. Consumption and interpretation of received material is now the dominant mode and very little generation of material for broadcast takes place locally. This shift has been recognised in education over the last 20 years with the introduction and increase in teaching of Media Studies, a curriculum area designed to equip students with an understanding of how messages are conveyed and manipulated by media producers, while giving them confidence that their own reponse and experience of the world is a valid one.

This work is important if consumption is to take place critically and a diversity of voices is to be encouraged. It becomes even more valuable as we move into an 'Information Society'. It is no longer enough to explain how goods and services are related to money in order to equip students with a basic understanding of citizenship and the political economy. Information has been identified as a new currency, a highly portable one which ignores conventional boundaries. Knowledge workers are taking over from the production line as the new wealth generators.

The change in importance of information and the media that carry it is not being ignored, far from it. In fact, an unprecedented campaign to provide access for schools to networked media was launched last year. At a national level, this took the shape of the Government's 'National Grid for Learning' initiative. At a local level, it is manifested by the appearance of terminals and modems in teacher centres and schools.

The question remains, though, as to whether the nation's schools are being 'wired' in order to teach knowledge using the information available online, to teach more about the nature of the information

economy or to teach information-handling skills. A responsible education should equip students with all three forms of learning. But there is little evidence that such distinctions are being made or planned for in the promotion of online resources.

Rather, the agenda seems confused. For instance, the Learning Technology Project (Ardern and Paley 1998), an educational think-tank with close links to the Government, express the gain to education in financial terms: the cost of educating the majority will reduce through pooling of information resources, with privately generated materials supplied online and a role for the state as regulator rather than provider.

The pilot information and communication technology (ICT) projects observed by the NCET had a different emphasis. These school and college based projects ranged from accessing remote resources to using electronic communication, with a few projects also creating resources for use by others. These projects involved, for instance, using interactive television, video-conferencing or taking digital photos to place into the school's web pages.

The projects' evaluators identified the following benefits as emerging from these activities: improved subject learning, development of network literacy, improved vocational training, improved motivation and attitudes to learning, development of independent learning and research skills, and social development. Of course, the motivational aspects may disappear as soon as using these technologies becomes commonplace, but these results demonstrate that enthusiastic teachers who are prepared to experiment with ICT can produce educationally rewarding results.

By contrast, Ardern and Paley express dismay at the Luddite tendencies of teachers and, it is true, that interest in resources such as the Internet is limited. But at the moment there is little education for teachers about what the technologies bring in the way of benefit and a lot of sensationalist talk in the media about some of the dangers. If the deployment of networked computers is left to teachers with enough motivation to find out for themselves what their benefits and limitations may be, then a majority of teachers and students will have been handled irresponsibly, despite the Government's flag-waving.

What are teachers' perceptions at the moment? Generally, there is a fear that education is being marginalised further - and, likewise, teachers' jobs - in a new emphasis on the technology of supplying it and training people to use this technology. This fear is also linked to uncertainty about the merit of using standardised materials, the likely consequence if lessons were beamed into schools from outside. The example of Microsoft's domination of supply in the States does nothing to reassure them.

More specifically, teachers and parents do not fully understand the potential of networked media. They have heard about the extensive collection of pornographic sites available at the click of a button and the bomb-making instructions that work. They appreciate the treacherous nature of the resource being offered them, but not how to turn it into a useful learning experience.

How might this be done? Many teachers have not used digital technology in the classroom and those that have are mostly familiar with CD-ROMs. This is a model of limited use in dealing with networks for a number of reasons.

CD-ROMs come pre-packaged, representing a unified vision and a single theme. The ones in use in schools have mostly been designed as an educational tool. The good ones offer a mechanism for structuring students' experience. Although some allow the addition of notes, no structural content may be added.

By contrast, information is placed on the Web by a huge number of different interests, many for promotional reasons. This makes it patchwork and patchy. No one monitors the content. It is full of lies, misrepresentations and misunderstandings. It is constantly expanding: anyone can put anything up

there. It appears infinite: nonetheless, vast tracts of useful information have not been placed there and may never be. It is difficult to search as it is comprises millions of pages, there is no systematic indexing system for sites, no agreed form of categorisation for search terms and no guarantee that what you search for exists.

Strangely, in view of this, one popular model of the Web being offered teachers is that of the encyclopedia - a model reinforcing notions of the Web's similarity with CD-ROMs in that information in encyclopedias is focused, reliable, easy to access and relevant. The Web is presented as an information resource from which students can find answers to the kind of questions that crop up in project work, an opportunity for 'adventurous individualised study' (NCET 1998). The metaphor is pervasive; it is even shown like this on television commercials for cars.

And the Web can be used like this, but only under tight teacher control. There are some educational sites which do offer many qualities in common with CD-ROMs (though they cannot support the same variety of multimedia experiences because of bandwidth limitations). Directing students to these sites and working with the information available on them is a practical use of the resource.

And there is help available for finding these sites. For instance, this need informs the design of the BBC Education website's resources for teachers. A database of educationally useful sites, like a catalogue, can be searched by subject, age group and level, providing a link, address and short description. The BBC offers its status as a responsible provider of educational materials to bring value to its endorsements. In designing its site, the BBC at least acknowledges the heterogeneity of the material available and the need to pick carefully; its model is closer to that of a library than an encyclopedia.

However, working with the Web like this is only the tip of the iceberg when it comes to using the Internet as a learning tool. Firstly, It is worth noting here that even though 'Internet' and 'Web' have become synonymous for many commentators, in fact email, mailing lists and electronic conferencing offer one-to-one and one-to-many communication mechanisms that support discussion, depend on participants for the content of the exchanges and do not rely on having a Web connection.

However, even leaving these other forms aside, If teachers accept models of the Web as encyclopedias and libraries, they are overlooking aspects of the Web that make it a complex and interesting object of study in its own right. They are not equipping students to use it for individual study, since they will be passing on a model of the Web with many misleading features. They are not exploiting the Web's strengths, such as the ease with which content can be created and displayed, either locally or to a broader audience. Anyone can build a site who has access to a computer; it can then be linked to the outside world by the addition of no more than a server on which to store it and a modem.

While clearly there are funding and training issues linked to using the Web as more than a means of gathering information, these issues must be regarded in the context of the extra power of the learning experience attained through the extra effort. There is also the matter of fitting a new kind of learning into timetables full with National Curriculum requirements. Nonetheless, classes of all ages will eventually have access to the technology and should, likewise, be given access to the best possible understanding of this technology. It should not be relegated to Media Studies, rather the study of media should become cross-curricular.

At the start of this paper, I referred to three forms of learning that networked media could facilitate: greater knowledge, an understanding of the new economy and better information-handling skills. I will approach these in the context of creating material for use on the Web.

The Web has large stores of information that can be accessed serendipitously and easily. If students are to work independently of adult guidance, in fact, if they are to develop good skills of discernment,

it is important that they are taught to judge the quality of the material they find, or at least to maintain a healthy scepticism. Sites can look professional without employing professional standards. Sites exist mainly for promotional purposes, be they political, personal or corporate, but this agenda is not always apparent from the design or content. One of the reasons for the Web's vastness, its patchwork nature and the presence of so many dubious voices is that it is a cheap and easy means of publishing.

This lesson can be taught, or it can emerge from building webpages in the classroom. Students are quickly able to produce good-looking pages which present fact, fiction or anything in between. The impact of seeing one's own work appear the equal of any other page that can be summoned is by far the most effective lesson in scepticism available.

This learning experience also forms a good basis from which to initiate discussions about ownership, control, authority and censorship, because these issues become concrete and relevant when the material being published is one's own. It can be a departure point for work on privacy and data protection, especially if students' pages are placed 'live' on the Web or a local network and a record of visitors is kept. In this way, something of the role and increasing importance of information can be conveyed, grounding these abstractions in the students' own experience.

Further, if information management is approached through site design, students are able to practice structuring their thoughts and ideas and to link them visually using hypertext. Many students have difficulty in applying a logical structure to extended pieces of work. The Web presents a couple of mechanisms for helping externalise the process of planning: hypertext offers a way of playing with alternative structures, while hierarchies and menus introduce a systematic approach to information design. These strategies can be explored on existing sites and employed using paper and pen, but without the realisation involved in actually producing the pages and links, many students will not be able to exploit the learning to the full.

A useful by-product of this is the generation of material that can be studied for insight into the challenges of categorising and indexing materials. This, in turn, helps students understand some of the difficulties in choosing search terms and may result in an improved search technique.

Apart from the vocational potential of developing these skills, there are two further advantageous by-products of working with students to build sites. The first is that this work redefines their relationship with media generally: if they can create good content and see it join that of other content providers on the Web, then media are no longer something to receive passively. The Web is potentially highly interactive and, by showing students how they can contribute, teachers blur the line between consumers and producers.

The second, and related, byproduct is the self-esteem that can be encouraged by the sensitve use of material that is created. In the same way that pictures are hung in corridors and poems appear in the school magazine, work for the Web can be made publicly available. Good work could be rewarded by inclusion on a public site, but because electronic storage is cheap, it would also be possible for every child to create a public 'homepage'. This could contain whatever they wanted and might be set as the page upon which the browser opened to access the Web when they logged on. In this way, even the most difficult or reticent pupil could be offered the chance to identify with the school, doing something 'cool' and that immediately looked good.

As I mentioned earlier, the issues of authority and censorship become concrete when one is dealing with actual work and the conflicting demands of students, teachers, parents and governors. Schools would have to develop policies on what might be made public: whether students would be given freedom to publish with an emphasis on personal responsibility or whether hierarchies would be established by

controlling scope: personal and group hypertexts, class networks, school intranet and, ultimately, a public site. Inevitably, it would depend on the prevailing ethos of school and the image that senior staff wanted to project.

So far, there has been little discussion of these issues, as there has been little mention of the benefits of a hands-on approach. When the TES did publish an article on how schools might use websites, it was entirely about promotion into the community (Flanagan 1998).

To conclude, there are many uses for the latest media in classrooms and, so far, this message has not been as widely disseminated as the media themselves. One of the most exciting aspects, the chance for students to participate in the creation of content has had the least attention of all. This paper presents an argument for why this needs addressing. The creation of Media Studies as a subject area was a slow, bottom-up response to the changing nature of society. Something more is needed this time.

In: Retkowsky, F. P. (Ed.). (1998)
The 11th White House Papers: Cognitive Science Research Paper 495
School of Cognitive & Computing Sciences, University of Sussex

3D Interactive Learning Environment

Nuno Otero
nunop@cogs.susx.ac.uk

School of Cognitive & Computing Sciences
University of Sussex
Brighton
BN1 9QH

1 Introduction

This text presents the proposal for a research programme to be developed on the application of virtual environments to learning. It will establish the main goals of the study, integrate them on existing work and propose the means to achieve such objectives.

In the second section we will present some of the most influential proposals on the classification of graphical simulations along with some discussion upon a typology for computer graphics systems and its usefulness. This topic is important since it will enable us to clarify the potential interactivity properties that these kind of systems have.

Then we will reflect about the existing conceptual frameworks to deal with the problem of learning in virtual environments. Little structured cognitive analysis has been carried out to explain the real advantages of applying VE's to learning. So we will have to look for general concepts and try to adapt and apply them to the specific problem of learning in virtual environments that we are interested in. We start by presenting, in the third section, a discussion of general cognitive issues related to the interaction in this kind of technologies. The fourth section presents the several types of learning that virtual environments can promote. We will focus on the conceptual learning, with a brief comment on what has been done in this field. Finally, the fifth section presents the general conceptual framework to be used.

Given the state of the art in this field, we aim at investigate about the benefits of VEs for learning along two main directions. The first one relates to the potential benefits of just exploring a simulated environment. The second one questions the benefits of representing information in an 3D interactive graphic format. To address this two issues we propose two experiences, the archaeological site exploration and the stereographic projection.

2 Virtual Environments and Related Definitions: trying to define concepts.

As Henry Fuchs refers in the foreword of the book "Virtual Reality Systems" (Earnshaw, Gigante and Jones, 1993), there is no consensus about the definition of virtual reality, even among experts. The debate around this definition is extensive and one should try to avoid unnecessary overlapping and misinterpretations.

2.1 The concept of Virtual Environment

According to Gigante (1993) VR is characterised by: "The illusion of participating in a synthetic environment rather than external observation of such an environment. VR relies on three-dimensional (3D),

stereoscopic, head-tracked displays, hand/body tracking and binaural sound. VR is an immersive, multi sensory experience." (pp. 3).

Kalawsky (1993) defines a virtual environment system as "...an artificial, fully immersive surrounding that allows the user to interact with computer generated objects. The interactive virtual environment can be a computer synthesised representation of a real world situation or an abstract form of a real world event." (pp. 77).

The list of different definitions continues. Another approach is to try to decompose the systems elements and clearly separate the features (the displayed images, sounds, haptics, etc) from the results to the user (for example the sense of presence, performance in tasks or other specific goals).

Wickens and Baker (1995) consider that VR systems can be decomposed into five distinct elements:

- three-dimensional (perspective and/or stereoscopic) viewing Vs two dimensional planar viewing;
- dynamic Vs static display;
- closed-loop (interactive or learner centred) Vs open-loop interaction;
- inside-out (ego-referenced) Vs outside-in (world referenced) frame-of-reference;
- multimodal interaction.

These elements are not independent but its implementation and analysis can be separated.

Ellis (1995) considers that computer generated virtual environments are a communication media where the used metaphor is an environment. The author refers that an environment should reflect three different components:

- A content, where we find objects and actors that are defined by vectors. The distinction between objects and actors relies on the ability of the latter to initiate interactions. Another important distinction is to consider the self that corresponds to the actor, that having a particular viewpoint establishes the frame of reference to the generation of images; all the elements exterior to the self are the field of action.
- Geometry, defined as the environmental description of the field of action and has dimensionality, metrics and extent.
- Dynamics, are basically the rules of interaction between the elements of the content. It is important to refer that Ellis (1995) proposes the concept of semantic or informational mechanics as a metaphor of a mechanics of content elements responses to the variations of the field.

Ellis (1995) also proposes the notion of virtualisation as "...the process by which a viewer interprets patterned sensory impressions to represent objects in an environment other than from which the impressions physically originate" (pp. 15).

Zeltzer (1992) classifies graphic simulation systems through a taxonomy that includes three components: autonomy, interaction and presence. The author considers that all graphical simulations have: "...(1) a set of computational models of objects and processes to be simulated, (2) some means of modifying the states of these models over time course of the simulation, and, finally, (3) communication channels that allow the participant to experience the simulated events and processes through one or more sensory modalities." (pp. 127).

From here, Zeltzer develops his concepts:

- Autonomy, refers to the reacting and acting capacity that the computational model have to events and stimulus. This means that we could distinguish, for example, between a system that limits itself to be read and another that alters his narration in accordance to user actions.
- Interaction, refers to "...the degree of access to model parameters at runtime." (pp. 127). Here Zeltzer (1992) points out that the degrees of freedom exhibited by the system to user interaction should be closely designed to meet task requirements.
- Presence, refers to a certain feeling emerging from our capacity to act in a world, which is closely related, in graphical simulations, to the fidelity of the sensory input and out put data.

These three properties form a cube in which the different systems can be compared. For example: a system with all levels high should be "...fully autonomous agents and objects that act and react according to the state of simulation, and that are equally responsive to the actions of the human participant." (pp. 129). Zeltzer (1992) admits that he still finds difficult to rigorously quantify his proposed properties. In fact, it seems that autonomy and presence are somewhat tricky concepts, the first because it seems difficult to clarify strict boundaries between systems and their autonomy, especially if they are not reacting to the same stimulus; the second because the implicit notion of presence does not, clearly, take into account the subjective dimension and possible different weightnings that different sensory cues have on the feeling of presence that, additionally, might also vary from task to task.

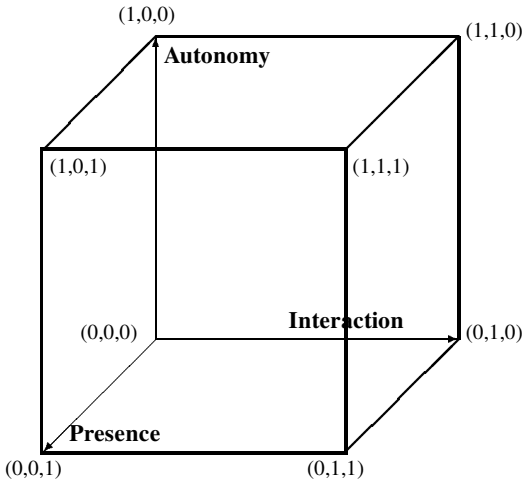


Figure 1: Adapted from Zeltzer, 1992, pp 129.

Although difficult, a typology of graphical computer systems and systematic study of the different interactivity properties and their relation to performed task seems essential, but a more fine grained framework seems to be required. Whitelock, Brna and Holland (1996) considered an extension of the Zeltzer model for educational purposes choosing three additional properties:

- Representational fidelity, that subdivides into:

  - technical fidelity, the degree of realistic rendering, colours, texture, motion etc, that raises the question of their relative importance to different tasks and activities;
  - representational familiarity, is the environment familiar?
  - representational reality, is the world possible?

- Immediacy of control, the medium that the system uses, where a set of natural behaviours for interface interaction corresponds to a more immediate system.

- Presence, our so many times referred, that the authors consider to require subjective and objective measures.

In a careful analysis one can find common issues in some of the frameworks revised and a different proposal should be towards integration. Kalawski (1993) refers the similarities between Ellis and Zeltzer's proposals. In fact, we can see that the differences established by Ellis about objects and actors resembles Zeltzer's concept of autonomy and that Zeltzer's interaction implies the same issues has Ellis's dynamics. The difference resides on the fact that Ellis expands his concepts considering that virtual environments are space metaphors and Zeltzer only pretends to classify graphical computer systems. About Whitelock et al. extension, a comment should be drawn. It is not completely clear, at least for us, how the proposed additional properties are integrated into the Zeltzer's previous model. One problem, for example, concerns the immediacy of control property defined by Whitelock et al. It seems to us that this property is already integrated on the Zeltzer's overall notion of presence, just that it is not independently specified.

In the present, I am working on the integration of the dimensions proposed by Zeltzer (1992), Ellis (1995), Wickens and Baker (1995) and Whitelock et al. (1996) towards the creation of dimensions that cover interactivity aspects of VEs important for a learning task. This integration will raise the possibility of constructing a general taxonomy of graphical simulations allowing a better understanding of the differences between systems. The chosen authors represent, in my view, the most influential proposals (Zeltzer's, Ellis and Wickens and Baker) and the one specific to the learning area (Whitelock et al.).

2.2 A brief comment about the concept of presence

Despite being out of the scope of this study, the concept of presence is a rather important one on the VE field. We shall, therefore, make a very brief overview on the main issues related with the notion and applicability of the sense of presence.

In fact, virtual environments are assumed to provide a different kind of experience, which is commonly reported as presence. It is symptomatic that presence is so often referred about this particular type of technology, although posterior analytical criticisms have considered the fragility of the notion in respect to its application and definition.

Presence is commonly defined as persons reported feeling of being in the place or present. Such feeling seems to be pursued by writers, graphic and theatre artists, film makers etc. It is interesting to question about the properties of virtual environments that enable an improved sense of presence, beyond the more traditional stimulus by authors and artists. What is there new, and how does this affect the sense of presence? At this stage, it is important to try to distinguish among different situations that can produce similar observable effects (see Barfield, Zeltzer, Sheridan and Slater, 1995). Reading a good book does not have the same sensorial cues than entering a virtual environment, so the cognitive processes involved are, at least to some extent, different.

Slater and Usoh (1993) refer that both external elements (related to technology) and internal elements (related to individuals) should be considered if one wants to understand how this sense of presence is

achieved. In particular, the sense of presence can be diminished if care is not addressed to appropriate interactive techniques (Usoh and Steed, 1994). Slater et al. (1994) go further on proposing that "body centred interaction" designs (dependent on the existence of consistent relationships, predictability and complitude available to the user, and on the usage of natural action schemas) promote higher levels of presence. Others have found that the degree of interactivity, or maybe in other words the realism of the interaction, appeared to be much more crucial than the degree of pictorial realism (Hendrix and Barfield 1996a and 1996b; Welch, Blackmon, Liu, Mellers and Stark, 1996), supporting the notion that the design of the interaction loop is a key factor.

The problem remains on how to assess the level of presence or, in another way, how to measure presence. It is consensual that the notion of presence should incorporate objective and subjective measures, the former including task demands, task results and correlated measures such as gross motor activity and psychophysiolical measures, and the latter including on line reports (verbal reports), post test evaluation with ratting scales and questionnaires and event explanations (Hendrix and Barfield, 1996a). One can see from the variety of assessment methods that there is still a lot to be investigated until the concept can have clear operational measures.

Another, more structural, question is to clearly understand the advantages of building a system for a particular task or goal that allows the emergence of the sense of presence. In another way, what does presence bring to the performance? Maybe this is task dependent. Intuitively, we can see that a more realistic graphical system is useful for training purposes, but when considering more abstract tasks the question rises many uncertainties.

3 Interacting with a Virtual Environment

This section discusses the way individuals interact in a virtual environment system. This will be important to describe and assess performances in such kind of environments in order to evaluate the adequability of the system to the proposed activities.

3.1 Brief discussion about interactivity in VEs

According to Carr (1995), virtual environments present a paradox, the one of pretending to simplify the comprehension of abstract information through egocentric ways of thinking. Here egocentric stands for the use of perceptual channels that invoke a more personal and subjective encoding to aprehend information, and the paradox relates to the fact that abstract information is the one represented in an "universal" way, that is, should be individual-independent. The use of sensory channels, or more natural ways of acting, could facilitate the comprehension of complex information taking advantage of, for example, body abilities of space apprehension to understand spatial relations. For the author, this has been one of the main issues driving the development of virtual reality, namely the challenge of making abstract concepts appear concrete and intuitive by matching a synthetic to a real perceptual experience. That is, Carr considers that one important feature of this kind of technology is that it enables the comprehension of abstract matters through perceptual knowledge. However, we believe that it is most important to understand how this process occurs simply because this may give some clues about the best way of constructing the system and display the information (as in Scaife and Rogers, 1996).

Actually, we ought to know how to present information through a certain sensory channel, allowing the user to "read" the representation conveyed by the system. This assumes that there are privileged sensory channels to the symbolisation of certain chunks of information in certain tasks with specific goals and that a key issue is the integration of different types of representations. To integrate different kinds of representing information we have to understand the dynamics between internal and external representations. It can be the case that the comprehension of an abstract concept is achieve through the translation

of information from one type of external representation to another, integrating and understanding the relationships between them. For example, Rogers and Scaife (1997) consider the interplay of concrete external representations and more abstract external representations in the comprehension of food webs. Considering that a certain way of presenting information allows a better comprehension is not enough (Scaife and Rogers, 1996).

Perhaps in a usability perspective, let us now look to the general tasks that a user can perform in a VE and their cognitive requirements. This perspective can aid us to have a more fine grained understanding of the activities that can be implemented in a VE and to realise what are the constraints.

3.2 General tasks performed in VEs and their cognitive requirements

Wickens and Baker (1995) present an analysis of the cognitive requirements of tasks that, in general, a user working in a virtual environment can perform. According to Wickens and Baker 1995), the tasks are:

- Search. Searching in a virtual space relates to the existence of a map that will allow the user to find objects. This map seems to need of two different kinds of views, one more general, with a broader perspective (exocentric view), and the other with egocentric point of view (egocentric view). Wickens and Baker (1995) point out that these electronic maps should: not contain unnecessary cluttering; have a flexible frame of reference permitting the use of canonical frames of reference or "track-up" ego-referenced frame; provide electronic ties between different objects in the abstract map coordinates and virtual space.

- Navigation. For the navigation task one should take into account that the mechanisms of interaction can be balanced in respect to two elements: velocity and flexibility on the one hand and situation awareness on the other. So, we can think of navigation systems where the user defines the location he wants to go and the system automatically places him there, or systems where the user walks all the way. It seems evident that between these two strategies there are many compromises and that the implementation of a certain system must be in accordance with the task that the user will perform. The authors also refer the importance of the visual depiction of surfaces and the stability of the perceptual motor loop:

  - There are some perceptual biases that should be considered especially when one has an interactive continuous navigation system (for example, perceptual biases of slant and distance, the stability of the perceptual motor loop, the influence of visual depiction of surfaces).
  - Another important element is the visual-motor coupling. Wickens and Baker (1995) consider five crucial issues:
    * Gain - is related to the setting of the gain for change in the display view.
    * Time delay - refers to the update rate of the visual world and has a big influence into interactivity. The more complex is the scenary the bigger the computing effort and therefore the possibility to affect time delay.
    * Control order - refers to the qualitative nature of the system state change as a result from a change in the control position.
    * Travel-view decoupling - concerns to the possibility of coupling or decoupling the direction of viewing and direction of moving.
    * Field of view - is the amount of the virtual environment that is viewable from a static point of view.

All these issues are important for a careful design and, naturaly, the most important is to know about the trade-offs related to task demands.

- Manipulation. When considering manipulation, Wickens and Baker (1995) suggest that the use of different styles of interaction can be advantageous to object manipulation. Different perceptual channels can be used, improving the information communication in different tasks. The type of visualisation is also relevant. This is because an object manipulation requires, most of the times, eye-hand coordination, making visualisation and manipulation difficult to clearly separate in this sense.

- Perception and inspection. Perception and inspection is related to the inspection and apprehension of the properties of the elements or objects of the virtual world. Light, shadows and motion parallax are the decisive issues to be considered.

The general tasks referred and their cognitive requirements are useful for starting to shape the implications of implementing a certain VE system for supporting the user to perform a certain activity.

4 Learning in Virtual Environments

Although a basic process, the definitions of learning are numerous and concurrent theories proliferate. Wild and Quinn (1998), in their discussion of educational theory and instructional multimedia, consider that in such context learning should be seen in terms of cognitive change. The type of knowledge to which they refer is the academic type or, in another way, others descriptions of the world, contrasting with experiential learning. But they do not exalt the discourse, recognising that the ability to internalise these descriptions and to transfer the knowledge to new problems comes only with the reconciliation of both types of learning. This comment can be helpful on discussing virtual environments if we believe on the ability of such systems to promote experiential learning or first person knowledge (Winn, 1993; Winn, 1997; Mikropoulos, 1997; Nikolou, Mikropoulos, Katsikis, 1997; Mikropoulos, Chalkidis, Kossivaki, 1997; Osberg, Winn, Hollander, Hoffman and Char, 1997; Winn, Hoffman, Hollander, Osberg Rose and Char, 1997). Perhaps one of the biggest advantages of virtual environments is its possibility to integrate different representations and ways of interacting with information. It is possible to consider that such strategy will enable the learner to build bridges between the two referred types of knowledge.

Tergan (1997), on reviewing the conceptual, theoretical and methodological problems that affect the design of hypertext and hypermedia systems in the context of learning, refers that the investigation efforts on design issues should be directed to a micro level. This means decomposing the problem into a cognitive analysis and considering the cognitive aspects of the learner as a core aspect, instead of using comparative studies to study the impact of a certain kind of medium. This issue is particularly important for the present research because it supports the study of a cognitive conceptual basis for enforming design.

The author also considers the importance of the external representations framework, streessing the necessity of a comprehensive theoretical framework that takes into account the interaction of codes, modes, and processing capabilities of external representaions, content aspects, instructional methods, learner variables, and contextual conditions. Furthermore, he refers the need to think about a fine grained task analysis of the activities that a learner has to perform when interacting with a external representation and, at the same time, consider the assembling of the several different elements/representations according to a rational that assures the proper matching of the task, the cognitive processing and learning.

In my perspective, Scaife and Rogers (1996) and Rogers and Scaife (1997) conceptual framework provide a good orientation for addressing the research issues of VEs and learning, following the need to

a more detailed cognitive analysis of graphical representations. The external cognition approach will be discussed further ahead.

4.1 Types of learning in VEs

Wickens and Baker (1995) consider four types of learning that can be promoted by virtual environments:

- Procedural learning. The goal is to teach a certain task or specific procedure relative to a well-defined situation. For the authors, the central characteristic of a system to this learning type is closed-loop interaction.

- Perceptual motor skill learning. Aims at teach some generic perceptual motor skill as, for instance, flying, training for particular medical procedures or for assembly or repair tasks. The authors refer that the subject should have an active role in the control loop and that the simulated task dynamics should be consonant with the real task. They also indicate that raising the sensorial feedback through more complex forms should be dependent on the type of task.

- Spatial learning. The objective is to create a virtual environment that can allow the subject to rehearse a certain specific situation. This task requires from the designer a detailed knowledge of the environment to be built. As examples, it can be referred the case of a surgeon practising to the performance of a particular operation or, in another level of this scale, flight pilots rehearsing manoeuvres that ask for terrain knowledge. The authors refer several studies that have investigated important aspects of possible properties that a virtual environment should possess to support certain tasks. For example, consider a flying task. In this case navigating through a virtual environment does not seem to have significative advantages if one compares with studying a 2D map: navigation performance was equally effective for both cases and the map study provided the subjects with a superior mental map of the environment. Another interesting result is the one from Wickens, Merwin and Lin (1994). The authors have investigated how subjects learned the shape of a 3D complex data structure varying the presentation form between 3D stereo and 2D mono, each corresponding, respectively, to high and low degree of realism. "While 3D perceptual augmentations assisted subjects in answering questions about the database as they were actually viewing the display, these enhancements provided only limited support for later memory and understanding of the surface shape." (Wickens and Baker, 1995, pp 530).

- Conceptual learning. Two essential points are refered: "First, as in other forms of learning, active closed loop exploration, rather than passive viewing, seems to be important (Gregory, 1991; Wickens, 1992a). Second, a key feature of any long term conceptual knowledge is the availability of alternative knowledge representation of the same material" (pp. 530). The authors reflect upon the importance of providing more abstract representations to the learner. These, functioning in conjunction with the virtual environment exploring experiences, could potentiate learning. It is evident that this integration must be carefully investigated because it is crucial that the learner understands how the different representations relate to each other.

4.2 Conceptual learning

The type of leaning we will be investigating is conceptual learning. Therefore, let us look closer to the use of VE's for it.

Whitelock et al. (1996) refer that, although VE's have showed their usefulness in training contexts, their utility in conceptual learning has not yet been established. In conceptual learning, the activities will have to require explanation and extrapolation (Whitelock et al. 1996).

Whitelock et al. (1996) consider that research on educational applications of VE's has been concerned with situation awareness or sensory-motor skills. There is no detailed research assessing the relationship between the structure and form of a VE and the nature of the conceptual learning that takes place. The authors also report Dede, Loftin, Salzman, Calhoun, Hoblit and Regian (1994) research on empirical evaluation of the effectiveness of VE for fixing misconceptions, and consider this work valuable in the sense that it raises questions about the design of appropriate modalities for instructing the domains or concepts.

There are several examples of VE systems that were built for conceptual learning, going from biology teaching (Nikolou, Mikropoulos and Katsikis, 1997), environmental education (Mikropoulos, Chalkidis, Kossivaki, 1997), physics (Brelsford, 1993; Dede, Loftin and Regian, 1994), science in a general sense (Winn et al., 1997) and the domains could go on. Firts questions are: Is there any specific domain candidate for implementation in a VE system? Does the domain really matter? Maybe the core issue is the possible types of representations that can be used to transmit the concepts and ways of interacting with them.

For an illustration we will refer the project developed in Human Interface Technology Laboratory's Learning Center, University of Washington, by Winn et al. The reason for this choice is that the authors conducted an evaluation program to understand the impact of their system in comparison with other presentation modes.

The Virtual Reality Roving Vehicle project give raise to several studies and systems and here we will make reference to the wetland cycles project. Assuming a constructivist approach, the authors intended to test the following hypothesis: "...that learning about a wetland cycle using constructivist principles (student-directed information retrieval and compilation, collaborative virtual environment design and construction) would yield greater comprehension of subject matter than learning about wetlands cycle through traditional means (teacher-directed classroom lecture, single textbook readings, worksheet completion). Our second hypothesis was that the traditional classroom approach would be more educationally efficacious than a no instruction control." (Osberg et al, 1997). The experiment involved 117 students assigned randomly to 4 groups: carbon, nitrogen, energy and water, which correspond to the world that they built. In fact, in each group the students experienced 3 conditions:

- study, design and build a virtual environment;
- study 2 cycles through traditional methodologies;
- not study one of the cycles.

The first condition (the constructivist treatment) included 4 phases: planning, building, programming and experiencing. The results show that the constructivist approach did not produced significative improving results comparing to the traditional approach. Furthermore, the authors refer that more important than experiencing was the building, this seems to mean that immersive VR alone did not produce positive results. But there is no clear description about the several variables that can have influenced the results, for example the authors consider that usability problems might have influenced the findings. As Rogers and Scaife (1997) refer there is a need for a different approach that would permit to distinguish the different effects that play when we are implementing a graphical representation through a computer system.

5 External Cognition

Scaife and Rogers (1996) refer that one can see many implicit assumptions about graphical representation that somehow do not address the real question whether or not there are cognitive gains from having

dynamic and interactive representations. There is a clear need for a conceptual framework or theory that judge the value of the different kinds of representations and cognitive processes that people use when interacting with them.

The conceptual framework that they created pretends to give a deeper understanding of how different representations are processed when people are engaged in different activities. The following clearly states the need of a more careful consideration of external representations and a more carefull analysis of the interaction between internal and external representations: "The value of this is to focus our attention more on the cognitive processing involved when interacting with graphical representations, the properties of the internal and external structures and the cognitive benefits of different graphical representations." (Scaife and Rogers, 1996, pp. 188).

For the authors there are 4 central cognitive properties according to which the external representations can be analysed over their ability to promote interaction (Scaife and Rogers, 1996 and Rogers and Scaife, 1997):

- Computational offloading, that relates to the possibility that different external representations can reduce the amount of cognitive effort required to solve informationally equivalent problems.
- Re-representation, that considers how different external representations sharing equal abstract structures can increase or decrease the difficulty of problem solving.
- Graphical constraining, that refers to the possibility that different external representations can show different degrees of constraining over the inferences that can be made about them.
- Temporal and spatial constraining, that considers the way that representations can promote the relevance of certain aspects of processes when framed through time and space dimensions.

Moreover, they propose some conceptual design key points concerning graphical representations (Scaife and Rogers, 1996):

- Explicitness and visibility, relates to the capacity of a graphical representation to direct the attention to the key elements that promote the learning through perceptual parsing and inferencing.
- Cognitive tracing and interactivity, refers to the full comprehension of different kinds of cognitive tracing and interactivity that the different types of graphical representation have.
- Ease of production, refers to the close link between producing and comprehending a diagram.
- Combining external representations, raises the problem of how to include different combinations of types of media.
- Distributed graphical representations, addresses the social aspect of learning.

In respect to the VR field, a bigger degree of abstraction of a representation should not mean less educational benefits. So, miming the world in perceptual fidelity is not, at least always, the best solution (Scaife and Rogers, 1996).

In their study, Rogers and Scaife (1997) chose a domain and a problem which allowed them to consider the effect of a more responsive media. They studied how to build different types of ERs that could make more explicit the relationships between more abstract and more concrete ways of presenting information, considering, also, that children would have less difficulties on understanding the more concrete ones. "By designing dynamic interlinked representations that co-vary in their abstraction and

interactivity it may be possible to provide children with a more effective way of understanding and reconstructing the formal notations used to describe the concepts." (Rogers and Scaife, 1997, pp.).

The dynamic linking is a concept that can have great relevance in the stereographic projection problem of my research, presented below. Such framework also states the importance of the choice and implementation of different interactivity types and external representations for supporting different learning stages. Hence, a crucial point is to know how to make the abstractions of a certain domain more explicit.

In the next section I will refer the design principles that Wickens and Baker (1995) propose for VEs. These design principles can match some of the considerations that Scaife and Rogeers (1996) refer and my goal is to see in what extend these two frameworks can shape my research design questions.

5.1 Design principles for learning virtual environments

Wickens and Baker (1995) state three design principles for learning virtual environments (in fact this design principles are also referred in Wickens (1992a and 1992b)):

- Consistency is one of the most well known and validated design principle. An example of inconsistency in VE's is the problem of using different navigation modes or frames of reference. At a first glance, one could think that the consistency principle would contradict the possibility of presenting multiple representations and interaction forms, and Wickens and Baker (1995) refer: "To be valuable in education, however, the novel perspective must be cognitively linked in the learner's mind, with a more abstract top-down, outside-in (global) perspective, so that the learner may see how the two relate." (pp. 532). This issue justifies the importance of the two next principles, redundancy of experience and the principle of visual momentum, that will permit an harmonisation of possible conflicts.

- Redundancy is another principle frequently found. The authors differentiate between redundancy and repetition. The former refers to the possibility of presenting identical or related information in different formats, and three advantages of this strategy, opposing to the repetition strategy, are considered: "...(1) it enables the information flow from computers to learners to be less vulnerable to learner shifts in attention (...) (2) For learning, it accomplishes the goal of long-term storage of material by achieving memory traces in different representational formats (...) (3) Again, for learning, redundancy allows different users (learners) to capitalise on the format of information display that is most consistent with their cognitive style (Yallow, 1980)." (Wickens and Baker, 1995, pp. 352).

- Visual momentum. This concept pretends to solve the difficulties of orientation that a subject, when using multiple displays, experiences. The authors refer three distinct points:

  - Use of consistent representations, that highlights the consistency principle safeguarding the possibility of, if necessary, using other representations of the same information as long as the relationships between them are clear to the user.
  - Use of graceful transitions, meaning that transitions from one representation to another should not be abrupt, allowing certain degree of navigation between states. Therefore it is preferable to use continuous navigation mechanisms.
  - Highlight anchors, where an anchor is a constant feature of the "world" that should be prominently highlighted on successive displays. This principle can be applied in spatial orientation, in a general sense, and as an aid to understand passages through different scales of visualisation.

As we will refer these design principles can be used in conjunction with the ones that Scaife and Rogers (1997) consider, at least, in the initial phases of the research. We think that these design principles can be further specified and this will be one of the general goals of my work.

6 Lines of research

The analytical framework that I am constructing for my research involves:

- find good dimensions for the characterization of graphical simulations in order to understand what are the key interactivity properties that these systems promote;
- consider the characteristics that Scaife and Rogers (1996) and Rogers and Scaife (1997) propose for external representations as main framework for the analysis of ERs;
- use the design dimensions referred by Scaife and Rogers (1996) and Wickens (1992) and Wickens and Baker (1995) for the raise of pragmatical design research;
- use the general tasks for VEs that Wickens and Baker (1995) propose as high level categories for an analysis of users' interactions with VEs; this analysis will be carried out using methodologies for data gathering that will be described further ahead.

6.1 The approach to the problem

In my opinion, investigating about the benefits of VEs for learning involve two core issues:

- Is it possible that simply exploring a certain simulated environment can promote an aditional learning gain through steering imagination or some other cognitive property?
- What are the benefits of presenting information in a 3D interactive graphic format, especially when considering a 3D problem?

The motivation for considering these two issues is the possiblity of understanding the benefits of certain interactivity properties of graphical simulations for learning certain types of content/domains. This can allow a deeper comprehension of the relations between content, learning tasks and external representations. To investigate about these two points, two experiences will be developed. The first one, more related to the first issue, uses an archaeological site and the user is required to navigate through it. The second one, designed to explore the second issue, is a system that enables the user to actually *do*a sterographic projection in a 3D context. For a more detailed discussion of both experiences, see below.

In the first problem the metaphor that best describes the question is the school field trip to a museum or real archaeological site. Is there any difference on being there and being shown pictures or movies of the place? Can we take advantage of the possibility that computer systems have to integrate different representations dynamically to provide a richer integrated learning environment? I would also like to state that I am considering a particular domain where I could avoid having to implement learning activities that would promote the comprehension of the domain itself. The activity will be a simple exploration of the virtual environment and the information that it contains. This strategy allows me to understand the effects of a VE by itself, avoiding the mix of effects that implementing a certain activity that would allow the understanding or solving a particular problem might have.

In some sense, it is the informational value of the actions that I am trying to control. For example, although learners can perform similar actions in the two VEs that I propose to use, the informational gains of each action can be quite different. In the sterographic projection case, the manipulation of the

object is central to the understanding of the problem, but manipulating a certain object in the archaeological site may not have any informational gain, unless the recognition of the object is dependant on its manipulation. It is even possible that for focusing the learner attention on the relevant information a good design decision, for the archaeological site, would be to limit the possibilities of manipulating the object, considering the alternative of presenting information by a simple click of an input device.

The second problem is more confined and less prone to multiple interpretations. It is a matter of understanding how can we make more explicit the construction of an abstract representation through the use of more explicit and perceptually driven representations. The main issue is to know how to build the concrete representations and how to link them with the abstract modes.

It is clear, however, that the comparison between the two problems is not a conclusive one since I can not guarantee the perfect correspondence of the two. This should be seen more as two case studies with semi controlled variables.

6.2 The archaeology site exploration

The problem that I want to investigate through the archaeology site exploration is the usefulness of more realistic simulations. More spefically, I want to understand the benefits of allowing a learner to explore simulation of a certain location or environment that otherwise he would not be able to experience. In some sense I am adressing the problem of what are the benefits of providing a concrete representation with high levels of interactivity in a domain that people can have difficulties to experience. This also considers the importance of providing levels of realism of the representation and realistic here must be separated in two different issues:

- the pictorial realism or technical fidelity according to Whitelock et al. (1996);
- the interactivity realism, which should include the notion of interaction of Zeltzer (1992) and immediacy of control of Whitelock et al. (1996).

The experiment would consist on comparing different systems: one using a VE and others using pictures or videos for the display of images about the content to be learned. The sites can also support other types of representations but those would be maintained equal through the systems to be compared. The comparison will consist on an analysis of users' interaction and answers to tests about the content (further details will be explained in the methodology section).

The general issues that I want to adress are:

- Are there any differences on the narratives that the learners produce using the different representations?
- What are the cognitive processes that lay behind different narratives? Maybe the subjects can produce a richer narrative of the interaction in the VE simulation condition due to the aditional interactivity that actively exploring the virtual world allow, providing a richer memory encoding and better remembering.
- If the learner assumes the viewpoint of one of the elements of the simulation, would he acquire additional valuable knowledge? What kind of relationship is or should be established between the learner and his "new" form to promote richer experiences?
- Can this narrative be used for enhancing the cognitive processing of abstract representations or more abstract explanations of the phenomena that the learner has to acquire?
- What types of representations work best together?

6.3 The sterographic projection

The problem of teaching the stereographic projection is a good example of a 3D problem. In fact, this involves the understanding of how to explain the building of a 2D diagram that represents the symmetry properties of a 3D object. Most of the times, the explanations that we can find in books combine the analytical way with graphics that try to make explicit what particular state of the problem is being addressed. A 3D interactive graphical representation can smooth the transitions of the several states allowing, at the same time, the integration of the corresponding analytical expressions. In some sense, this problem resembles the one that Rogers and Scaife (1997) addressed with dynamic interlinking between different representations.

For a start, we recall what Wickens refers as the anchors that each system requires and the subject ability to incorporate the information of the different representations. Probably we can find a threshold on the amount of information possible to be presented on one representation and the efficiency of using multiple representations. The anchors are the elements common to sucessive displays that promote the consistency of the overall representation. For the particular case of the sterographic projection, note that a large amount of information is required. The question we are willing to address relates to the way information should be presented on this kind of problem. Should we separate it through different 2D representations? Or should we try to use 3D representations?

A sequence of 2D representations is a possible solution to avoid cluttering and the perceptual difficulties that come with it, where cluttering is not only referred as visual cluttering but also as an overall perceptual loading if one considers using multimedia solutions. These issues are particularly relevant in the stereographic projection if the object to be studied has multiple faces with complex symmetry relations. On the other hand, with a 3D representation the different states of the problem can be showed using short transitions and displaying the steps clearly. The question we are addressinng is that of knowing if keeping track of different 2D representations across different screens, and, perhaps, sometimes having to use them in a sequential reading process, is better than having a single 3D enhaced representation. The interactivity that a 3D representation conveyes could allow a more rapid and efficient information processing using perceptual processes and avoiding memory load. In fact, keeping track of the relevant pieces of information through the display of different representations can require a big memory effort, and using the proximity in 3D for relating information could be a less demanding alternative than using 2D.

Another question that this experience may give some clues on concerns the mental representations or maps that allow for the search of information necessary to the resolution of tasks. In other words, we are questioning whether 2D and 3D representations give rise to completely different mental maps and whether there are efficiency diffrences between them. To show the importance of this, suppose that a certain learning step requires the user to frequently find specific pieces of information. In this case, which type of external representation builds the more efficient internal representation? It might be that a virtual environment needs just one cognitive map, while different 2D representations need several ones but less complex. If so, is it possible that the use of several cognitive maps, having, simultaneously, to understand the anchors from one to another and to look for particular elements in the representations, is worse (better) than the use of just one more complex cognitive map requiring less attention to the anchors?

In cognitive terms, and making a general comment about the discussed issues, the different systems should impose different cognitive and memory loads. An interesting question is to study how the dynamics between external representations and the subject occurs. The studies already carried out by Larkin and Simon (1987), Zhang and Norman (1994) and Zhang (1997) addressing the study of external representations are good examples of a formalisation of a problem that I consider to be useful for my research. But the definition and formalisation of the space problem is clearly a difficult issue to address

when one considers complex domains. The complexity of the representation itself can be another problem since complex representations involve different cognitive properties in complex dynamics. So, it can be the case that we can not avoid analysing complex perceptual problems, investigating, for example, if the subject recognises the particular affordances of the representational object or if the object promotes good or bad analogies.

6.4 The basic interactivity properties in the VEs

The interactivity properties being investigated are:

- Visualizing - the benefits of the possibility to display information using an aditional dimension, the 3D display. The use of 3D representations will have different goals in the problems. For the archaeological site will be a way to promote realism but for the stereographic site it will help the understanding of the problem states.
- closed loop interaction - here I would like to question which of the following topics is more relevant for each particular problem or how to combine them:
  - manipulation - the possibility to actively manipulate the representation seeing the results of each action in a direct way;
  - navigation - the benefits of the possibility to navigate through the virtual environment and see different perspectives of the representation and 3D objects;

7 References

Barfield, W., Sheridan, T., Zeltzer, D. and Slater, M. (1995).“Presence and performance within virtual environments”. In Barfield and Furness (eds), *Virtual Environments and Advanced Interface Design*, (pp. 473-513) New York: Oxford University Press.

Brelsford, J. (1993). “Physics education in a virtual environment”. *Proceedings of the Human Factors and Ergonomics Society 37th Annual Meeting*, (1286-1290).

Carr, K. (1995). “Introduction”. In Carr and England (eds) *Simulated and Virtual Realities*, (pp. 1-10). London: Taylor and Francis.

Dede, C.; Loftin, R.; Salzman, M.; Calhoun, C.; Hoblit, J. and Regian, J. (1994). “The design of artificial realities to improve learning newtonian mechanics”. In *Proccedings of the 1994 East-West International Conference on Multimedia, Hypermedia and Virtual Reality*, Moscow.

Ellis, S. (1995). In Carr and England (eds), *Simulated and Virtual Realities*, (pp. 11-52). London: Taylor and Francis.

Gigante, M. (1993). “Virtual reality: definitions, history and applications”. In Earnshaw, Gigante and Jones (eds), *Virtual Reality Systems*, pp. 3-14. Cambridge: Academic Press.

Hendrix, C. and Barfield, W. (1996a). “Presence within virtual environments as a function of visual display parameters”. *Presence: Teleoperators and Virtual Environments*, 5(2), 274-289.

Hendrix, C. and Barfield, W. (1996b). “The sense of presence within auditory virtual environments”. *Presence: Teleoperators and Virtual Environments*, 5(2), 290-301.

Kalawsky, R. (1993). *The science of virtual reality and virtual environments*, Reading, MA: Addison-Wesley.

Larkin, J.; Simon, H. (1987). “Why a diagram is (sometimes) worth ten thousand words.” *Cognitive Science* , 11, 65-99.

Loftin, R. and Kenney, P. (1994). “ The use of virtual environments for training the Hubble Space Telescops Flight Team”. Available at www.vetl.uh.edu/hubble/virtel.html.

Mikropoulos, T. (1997). "Virtual environments in science education". Available at www.uoi.gr/schools/edu/ptde/m

Mikropoulos, T.; Chalkidis, A.; and Kossivaki, P. (1997). "Virtual realities in environmental education: the project LAKE". Available at www.uoi.gr/schools/edu/ptde/mikropulos2.htm

Nikolou, E.; Mikropoulos; T. and Katsikis, A. (1997). "Virtual realities in biology teaching". VRET'97. Available at www.uoi.gr/schools/edu/ptde/nikolou1.htm.

Osberg, K.; Winn, W.; Rose, H.; Hollander, A.; Hoffman, H.; and Char, P. (1997). "The effect of having grade seven students construct virtual environments on their comprehension of science". Presented at the Annual Meeting of the American Educational Research Association, Chicago.

Rogers, Y. and Scaife, M. (1997). "How Can Interactive Multimedia Facilitate Learning" ...

Rose, H. (1995). "Assessing learning in VR: towards developing a pardigm Virtual Reality Roving Vehicles (VRRV) project" Internal report R-95-1.

Scaife, M. and Rogers, Y. (1996). "External cognition: how do graphical representations work?". *International Journal Human-Computer Studies*, 45, 185-213.

Slater, M., Usoh, M. (1993). "Representations Systems, Perceptual Position, and Presence in Immersive Virtual Environments". *Presence, Teleoperators and Virtual Environments*, 2 (3), 221-233.

Slater, M.; Usoh, M.; and Steed, A. (1994). "Steps and ladders in virtual reality". In Singh, Feiner and Thalmann (eds), *Virtual Reality Software and Technology - Proccedings of the VRST'94 Conference*, pp. 45-54, Singapore: World Scientific.

Tergan, S. (1997). "Conceptual and methodological shortcomings in hypertext/hypermedia design and research". *Journal Educational Computing Research*, 16 (3), 209-235.

Welch, R.; Blackmon, D.; Liu, A.; Mellers, B. and Stark, L. (1996). "The effects of pictorial realism, delay of visual feedback, and observer interactivity on the subjective sense of presence." *Presence, Teleoperators and Virtual Environments*, 5(3), 263-273.

Whitelock, D.; Brna, P.; and Holland, S. (1996). "What is the value of virtual reality for conceptual learning? Towards a theoretical framework". In Brna, Paiva and Self (eds), *Proceedings of EuroAIED*, (pp. 136-141). Lisbon: Fundacao Calouste Gulbenkian.

Wickens, C. (1992). *Enginnering Psychology and human performance*. New York: HarperCollins Publishers.

Wickens, C. and Baker, P. (1995). "Cognitive Issues in Virtual Reality". In Barfield and Furness (eds), *Virtual Environments and Advanced Interface Design*, (pp. 514-541) New York: Oxford University Press.

Wickens, Merwin and Lin (1994). "Implications of graphics enhancements for the visualisation of scientific data: dimensional integrality, stereopsis, motion and mesh". *Human Factors*, 36(1), pp 44-61.

Wild, M. and Quinn, C. (1998). "Implications of educational theory for the design of instructional multimedia". *British Journal of Educational Technology*, 29(1), 73-82.

Winn, W. (1993). "A conceptual basis for educational applications of virtual reality". Human Interface Technology Laboratory Report TR-93-9.

Winn, W. (1995). "The virtual reality roving vehicle project". *THE Journal*, feature 12/95.

Winn, W. (1997). "The impact of three-dimensional immersive virtual environments on modern pedagogy". Human Interface Technology Laboratory Report R-97-15.

Winn, W.; Hoffman, H.; Hollander, A.; Osberg, K.; Rose, H.; and Char, P. (1997). "The effect of student construction of virtual environments on the performance of high- and low- ability students". Presented at the Annual Meeting of the American Educational Research Association, Chicago.

Zeltzer, D. (1992). "Autonomy, Interaction and Presence". *Presence: Teleoperators and Virtual Environments*, 1(1), 127-132.

Zhang, J.; Norman, D. (1994). "Representations in distributed cognitive tasks". *Cognitive Science* , 18, 87-122.

Zhang, J. (1997). "The nature of external representations in problem solving." *Cognitive Science* , 21 (2), 179-217.

In: Retkowsky, F. P. (Ed.). (1998)
The 11th White House Papers: Cognitive Science Research Paper 495
School of Cognitive & Computing Sciences, University of Sussex

Developing an experiment workbench to study software reuse from a cognitive perspective

Fabrice Retkowsky
fabricer@cogs.susx.ac.uk

School of Cognitive & Computing Sciences
University of Sussex
Brighton
BN1 9QH

Abstract Software reuse, as a promising programming technique, has led to many technological developments. But it also involves programmers' cognition, and different theories compete as to how code reuse should be assisted by a computer tool. We decided to develop a reuse workbench, made of a reuse tool and an experiment toolkit, to simplify the evaluation of these various theories. We describe how we complemented the initial pen and paper design of the reuse tool by a simple experiment.

1 Introduction

Software reuse is an increasingly popular aspect of programming. In the face of the demand for an ever increasing volume of code, reusing code components to build new programs seems to be a good solution (McIlroy, 1968). Besides, reusing code brings the promise of improvement from a qualitative point of view. New technologies have been developed to support software reuse, such as object-oriented programming, database repositories, component distribution systems, etc.

Yet code reuse (like programming in general) raises another kind of issue. When a programmer wants to reuse an existing piece of code, he has to express his requirements, to identify a suitable component, and then to understand it, modify it, and integrate it in a new program. All these processes strongly involve the programmer's cognition: his memory, his understanding, his mental models, and so on.

Hence Burkhardt and Detienne (1994) consider that "the psycho-ergonomical approach can help supporting the design of help techniques for the search, selection and specialization of reusable assets", techniques which will be "as compatible as possible with the activity and the thought process of software engineers".

In this paper. we will explain why we decided to develop a workbench for conducting experiments on the cognitive aspects of software reuse, and how we supported its design with a simple experiment.

2 An experiment workbench for software reuse

2.1 Why a workbench ?

Many aspects of cognition are involved in software reuse, and many different theories or exist for each one of them. For example, data-flow charts, control-flow charts, and textual descriptions can be used to represent programs. But which one of them is the most suitable, the most efficient? And do their efficiencies vary depending on the type of software components they are applied to?

As a consequence, we are developing a tool which is aimed at running experiments in software reuse. This workbench will be made of two parts: a module-based software reuse tool, and an experimental tool-kit. In this paper we describe the development of the reuse tool.

Mulholland (1995) used a similar approach to evaluate four software visualisation tools. He embedded the four visualisation techniques into one single environment, the Prolog Program Visualisation Laboratory (PPVL), hence unifying the user interfaces, and then conducted some experiments to test the techniques' efficiency.

2.2 Initial design of the reuse tool

The idea behind the software reuse tool is to give access to a database of reusable Java components. To allow simple experimentation, it will be based on different sets of exchangeable modules. For example, the search technique, the database and the documentation techniques will all be different slots, into which alternative modules can be fitted.

The tool is based on a four-stage reuse model. First, the user will look for a component in a database by browsing or using a search engine. Then, he will seek more detailed information about the chosen component and try to understand it. Finally, he will try to specialize it, and then integrate it into his own program.

We decided to focus on the Java language for four reasons. First, Java databases of components are freely available on the Internet. Second, it is object-oriented. Third, it is very popular and there is a large pool of programmers as potential experiment subjects. Finally, the tool itself will be developed in Java.

2.3 Supporting the design by a simple experiment

To assist the design process, we decided to conduct a simple experiment that would highlight the most important issues that a reuse tool should tackle. This meant having programmers perform a simple reuse activity with a simplistic reuse system, observe their performance, and tell us what their problems were and what they thought of reuse tools. By using novice programmers, who did not have any real experience with reuse, we mainly looked at the initial problems that programmers encounter when trying to reuse.

This experiment was designed to demonstrate how well novice programmers could perform with a rudimentary reuse tool. It will serve later as a benchmark against which we can compare more advanced reuse tools. Using beginners in reuse, and doing later on the same kind of experiment with expert programmers, will also help us understand how programmers improve their reuse skills.

Sutcliffe and Maiden (1990) conducted a similar experiment, where novice systems analysts had to perform a requirements analysis task. The authors used protocol analysis, where oral information was used to understand the reasoning behaviour of the subjects. In this study we focus much more on the interaction between the subjects and the computer system, and try to see which factors can influence both this interaction and the performance of the subjects.

C. Hoadley and Clancy (1995) conducted two experiments about novice programmers and code reuse. Though the first one was focused on program understanding during reuse, the second studied how confidence in software reuse can influence the whole act of reusing some code. Here we are interested in how some technical and cognitive (rather than sociological) aspects influence code reuse.

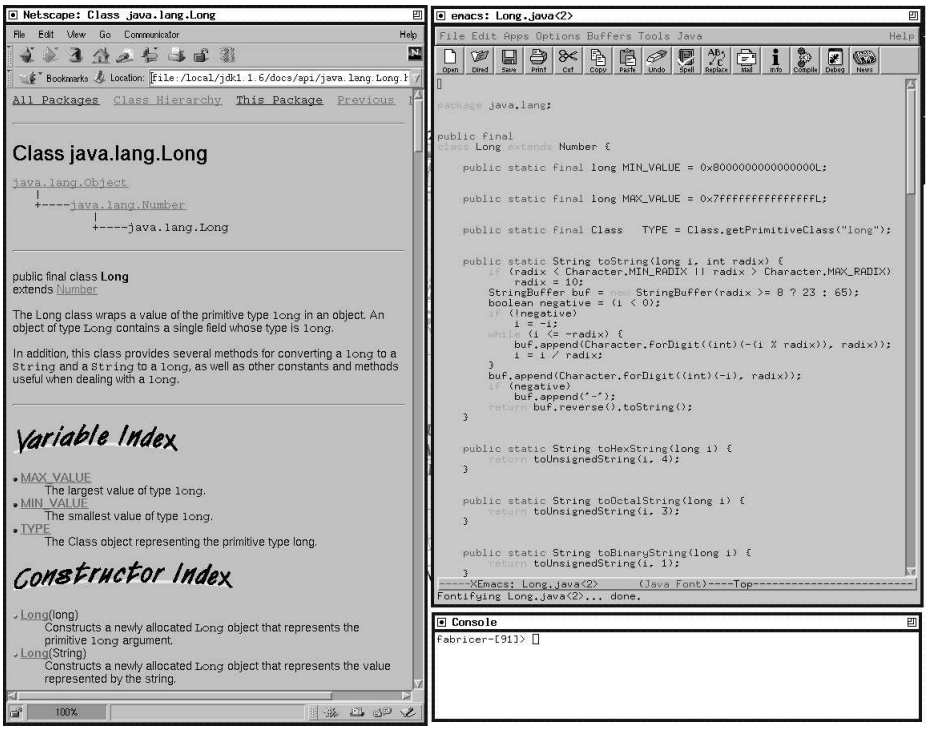


Figure 1: Experimental setup

3 The experiment

3.1 Design and materials

The experiment consisted in asking 12 Java beginners to program a simple class by reusing a class from the Java API packages.

The Java API packages are standard packages of classes, written by Sun (the creators of Java). Though they were particularly aimed at being reused by traditional object-oriented programming techniques (inheritance, class composition, method calls), the source code of these classes is available as well, so that code reuse is possible. We only had to remove from the classes the JavaDoc commands which were making the code longer and more cryptic. Furthermore, there is freely accessible documentation on the Web, in HTML. Hence, by supplying an Internet browser (Netscape Navigator in our case), the source code, and a text editor, we had a simple, rudimentary software reuse system (Figure 1).

Each subject had to perform two tasks amongst the 4 tasks available (Table 2). These four tasks are two different programming problems (programming a PhoneNumber class or a PhoneNumberFormat class). Each one of these problems (A or B) existed in two versions. In the first version (1), the problems were described without any kind of context, and reuse was, as much as possible, forced. In the second version (2), reuse was only suggested, and the class to program had a context: the subjects had to complete a large program in which a class was missing. Each subject either performed A1 and B2 or A2 and B1, in balanced order. The tasks descriptions can be found in Section 7.

The PhoneNumber (A) and PhoneNumberFormat (B) tasks were designed to be different from both a programming and a reuse point of view, so that the results would be less task-dependent. The Pho-

	PhoneNumber A	PhoneNumberFormat B
Forced reuse, without context	A1	B1
Suggested reuse, with context	A2	B2

Table 2: The four tasks

neNumber problem is a classic datatype problem, fairly easy to solve, and for which many reusable components exist in the API packages. By comparison, the PhoneNumberFormat class is a very short, but more original problem, probably more difficult to understand for beginners. There are some perfectly suitable components to reuse for this problem, but they are quite well 'hidden' in the API documentation, so that they are difficult to find out. Thus we predicted that A would be slightly longer to perform, though much simpler, and that it would lead to more and better reuse.

3.2 Subjects

The subjects were first-year undergraduates from the School of Cognitive and Computive Science (COGS) at Sussex University. They were either studying Computing Science or Artificial Intelligence and Computing Science, and had attended two courses of Java programming. We had 12 subjects, each one performing 2 tasks. Hence we had 24 tasks performed, i.e. 6 tasks of each type.

3.3 Protocol

First of all, the subjects were introduced to the idea of reuse and told what the experiment was about.

For each one of the two tasks, the subjects were given the requirements of the class they had to write. They had no more than 20 minutes to complete each task. They only had access to a text editor and to a local copy of the API on-line documentation. They didn't have to actually write a complete, working class: we didn't ask them to compile their programs. While the subjects were performing the tasks, the experimenter took notes on all their interactions with the API documentation, and on the subject's comments (they were asked to think aloud).

Once the two tasks were finished, they were asked three open-ended questions about software reuse and the rudimentary reuse system they had used. If the subject reused a component in their program, the suitability of this component was then evaluated, as well as the quantity of code that was actually reused (both by mark out of 5).

When all the tests had been done, the quality of the resulting pieces of code was evaluated, by a mark out of 5. For each task, we first gave to the six programs an initial mark which took into account (from the least to the most important): the syntax errors, the shortness/clarity of the code, the programming mistakes, and the conceptual errors. We then ordered the programs and checked that the marking was consistent for this task. Finally, we compared the consistency of the marking between the four different tasks, by verifying that the best and worst programs were of the same quality in each case.

4 Results

4.1 General remarks

The subjects all followed the same pattern of programming. First, they read and tried to understand the problem description (it took 1 minute on average). Then, they looked for a component to reuse (3 minutes on average, though some subjects didn't search at all), and finally did the programming. Only two tests (out of 24) required more than 20 minutes.

The 'Index of Classes' and 'Class Hierarchy' (cf 4.2) in the documentation pages were not used at all. There might be three possible explanations: either these pages are useless, or they are nearly 'hidden', or users need more experience to use them.

4.2 API use

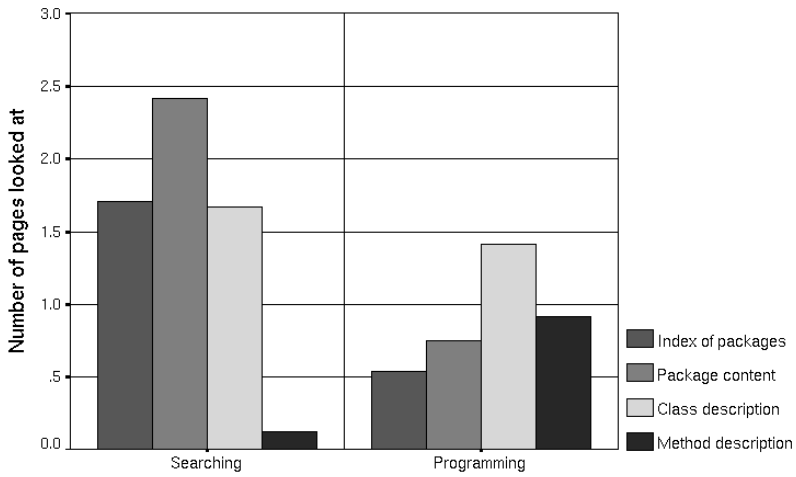


Figure 2: Use of the API documentation when searching and programming

The Java API documentation is based on a tree hierarchy. There are basically four levels of description: the Index of Packages, the List of Classes (for each package), the Class Description (for each class of each package) which includes a list of the class' methods, and the Method Description (for each method of each class).

When searching, subjects only used the first three levels of description (Figure 2). The Method level was hardly used. When programming, they used all the four levels, including the Method level. Comparing the searching and the programming stages, the first two levels were less used for programming ($t = -2.53, df = 23, p < 0.02$ and $t = -2.08, df = 23, p < 0.5$), the third one (Class level) was used the same, and the Method level was much more used for programming ($t = 2.94, df = 23, p < 0.01$). The statistical results mentioned here are by default 2-tail t-tests.

This suggests that the Method Description level is too precise for the Search stage. A proper search tool doesn't have to display such information: it will only be necessary for the Programming stage.

4.3 PhoneNumber - PhoneNumberFormat

The PhoneNumber problem (A) was predicted to be simpler, if longer, but proved to be slightly shorter to perform (no sig.). Subjects looked at the same number of API pages for searching, but they used

Variable	PhoneNumber	PhoneNumberFormat
Total time	15m 56s	16m 55s
No. of pages when searching	6.08 pages	6.00 pages
No. of pages when programming	4.33 pages	3.25 pages
Quality of the code	3.29 /5	3.29 /5
Percentage of reusers	42 %	42 %
Suitability of the components	2.80 /5	1.60 /5
Quantity of reuse	2.50 /5	0.90 /5

Table 3: Results for the PhoneNumber and the PhoneNumberFormat tasks

more API pages while programming (probably because the programming was longer). The quality of the resulting code is the same for both tasks, though the evaluation was suggestive.

We forecast that A would lead to more and better reuse. In fact as many subjects reused for both tasks. Yet, they indeed reused better for A: the components they reused were more suitable (not sig.), and they reused more lines of code ($t = 2.5, df = 8$, 1-tail p < 0.02).

As a whole we can say that there were enough differences between the two tasks from a reuse point of view to give more credibility to the other results.

4.4 Forced reuse, without context (1) - Suggested reuse, with context (2)

Variable	Forced/NoContext	Suggested/Context
Understanding time	1m 4s	1m 35s
Searching time	5m 32s	0m 53s
Programming time	11m 35s	12m 11s
No. of pages when searching	9.92 pages	2.17 pages
No. of pages when programming	3.17 pages	4.42 pages
Quality of the code	3.00 /5	3.58 /5
Percentage of reusers	67 %	17 %
Suitability of the components	2.50 /5	1.00 /5
Quantity of reuse	1.87 /5	1.00 /5

Table 4: Results for the 'Forced reuse/Without context' and the 'Suggested reuse/With context' situations

Logically, in the forced reuse/no context situation, subjects spent less time understanding and more time searching, and they looked at more API pages while searching. They spent the same amount of time on programming for both cases. This is because the programs were too small for reuse to have any effect on programming time. The subjects also looked at about the same number of pages while programming, though they looked at fewer methods and more classes for 1. The difference of one page (3.17 compared to 4.42) comes from a technical aspect of the API pages and from the fact that most of the subjects in the situation (2) skipped the Searching stage.

With forced reuse, as forecast, more subjects decided to reuse ($t = 2.76, df = 22$, 1-tail p < 0.006). They also reused better, largely because the only two subjects who reused in the suggested reused condition did reuse poorly.

Finally, the situation (2) produced better programs, mainly because (2) led to less reuse, and programs based on reuse were judged of lesser quality (see 4.5).

4.5 Expertise

Though the subjects were all beginners, there were initially two measurements of their expertise in Java: the number of programming languages they knew (including Java), and the amount of time they had spent programming (the sum for all languages, where one year was counted as three terms). Though, the quality of the resulting programs turned out to be constant. This suggests that past programming experience did not help in performing this task.

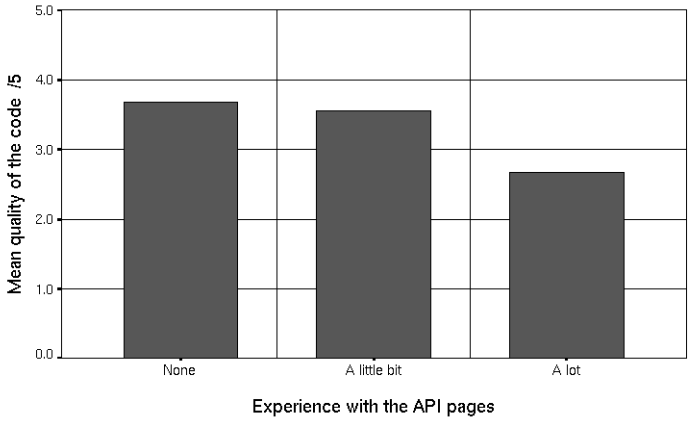


Figure 3: Expertise with the API pages has a negative effect

We also used as a third estimate wether the subjects had used the API pages while learning Java ('No', 'A little bit' or 'A lot'), which denotes wether they were serious and/or curious about learning Java. Surprisingly, the API pages experience seems to have a negative effect on performance (Figure 3). 'A lot' of experience led to lower quality code than 'A little bit' ($t = -2.11, df = 14, p < 0.06$) and than 'Not at all' ($t = -2.67, df = 14, p < 0.2$). There are two explanations for that:

- Firstly, subjects with 'A lot' of experience with the API pages reused more often: 65% of them reused, instead of 37% for 'No' experience and 25% for 'A little bit' (not sig.). Besides, programs based on reuse were evaluated as of lesser quality than normal programs (average of 2.80/5 compared to 3.64/5, $t = 2.44, df = 22, p < 0.03$). In fact, comparing the quality of reuse-based programs and normal programs, for these simple tasks, is fairly subjective. But looking at the reuse-based programs shows that reusing led to many minor errors such as forgetting to rename the class, including the new class in a package, inheriting from a class without implementing some abstract methods, and leaving 'as-is' many useless methods. These errors are characteristic of programmers who never did any code reuse before.

- Secondly, Figure 4 suggests that 'experts' are under-performing whether they reuse or not. This is even more surprising. It might be explained by 'experts' trying to perform very well under experimental conditions, thus focusing on secondary details and not solving the main problems first.

Finally, subjects who used the API pages 'A lot' in the past looked at less pages per unit of time while searching than others, and more pages per unit of time while programming. This means that they

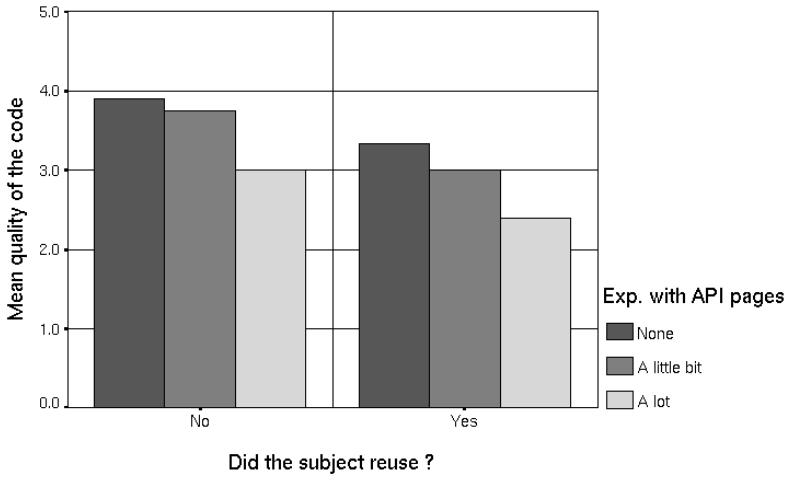


Figure 4: Experts under-perform whether they reuse or not

probably read the descriptions more thoroughly when searching (as opposed to 'beginners' who just browse), and that they knew how to use the API pages as a programming help.

5 Consequences for the Design

Once the subjects completed their two tasks, they were asked three open-ended questions about software reuse:

- What are, in your opinion, the good aspects of the API pages as a reuse tool?
- What are, in your opinion, the bad aspects of the API pages as a software reuse tool?
- What should a perfect reuse tool look like?

The answers we collected can be found in Section 8. From these answers, and from the points we made in the numerical analysis, we can draw some guidelines for the design of the reuse tool. Some of these guidelines are already met by the initial design, some led to a few modifications.

Component description As we saw before, the reuse tool will be based on a set of modules. One of the most important modules is the component description. What appeared from the experiment's feedback is that the API documentation style is a good basis to start with, since:

- it includes a list of all the methods;
- the class hierarchy at the top of each class is a good thing;
- it is well structured and uniform;
- there are lots of links between classes;
- it shows the correct syntax via an example.

Therefore we will base the first version of the component description module on the API documentation. Yet the subjects suggested a few modifications:

- the class and package names should also be self-explanatory for beginners;
- it should have less technical terminology;
- it should also describe the code itself, and make it easily accessible (or even include it in the description?), particularly for the 'Understanding' stage;
- it should include some examples;
- the packages should have a description as well;
- it should be less complicated, and shorter. This is easily feasible for the search stage: the experiment proved subjects don't use the 'Methods' level.

Navigation Since navigation is an important issue, we initially designed a complete and efficient set of navigation tools. The subjects reminded us that the navigation should be very simple (i.e. like the API, in HTML), and that:

- it should provide something so that users don't get lost;
- the Search stage should actually have a search tool;
- it should always suggest alternative possibilities, so that the user does not get trapped in one not-so-good solution;
- it should assist but not be intrusive.

As a consequence, it was decided to keep the navigation tools to a minimum, that is, a bar menu and a small wizard that allows quick navigation between the four stages of reuse.

Structure Finally, the system should include some editing tools (to specialize and integrate the components) and a built-in compiler (which was lacking from the experiment's rudimentary reuse setup). These were not planned at first, but will be included in the Specializaton and Integration stages.

6 The next steps in the development of the tool

Since this experiment was completed, we have developed a mock-up of the user interface, and have had it tested by a few possible users. The next step consists in designing the experiment toolkit, and then programming the whole reuse workbench for real.

Once the system is completed, we will perform the experiment described in this paper again, but using our tool instead of the rudimentary reuse system used here. This will have two aims:

- to test wether there are any major flaws in the design of the tool, or whether subjects have any problem in using this kind of integrated tool that guides and assists them,
- and to know wether, in a simple configuration based on the API documentation, our system already brings some kind of benefits.

Finally we will develop and compare some new sets of modules, for example to evaluate alternative search techniques or documentation styles.

7 Appendix 1: The four tasks

7.1 Task A1

Write a PhoneNumber (PN) class by reusing a Java API class.

- A PN object will contain a telephone number, such as 1273275779
- It will be initialized using a String parameter, i.e. ”1273275779”
- It will have a toString() method which will give back a String such as ”(1273) 275779”

You HAVE to reuse a Java API class to write this class.

7.2 Task A2

Here is the PhoneList class, which is used in an ‘Organizer’ program.

- It is basically a Vector of PhoneNumber objects
- The ‘Organizer’ program creates such PhoneLists, adds PhoneNumbers to them, remove PhoneNumbers, and print the PhoneList on screen
- PhoneNumbers are created using strings, such as ”1273275779”
- When the PhoneList is printed on screen, the PhoneNumbers should be displayed as ”(1273) 275779”

Write the PhoneNumber class.

You can reuse an existing class file from the Java API if you want.

```
public class PhoneList
{
    int MaxSize = 3;
    PhoneNumber[] PhoneArray = new PhoneNumber[MaxSize];
    int NbNumbers = 0;
// position 1 for PhoneArray[0]

    PhoneList()
    {
// creates two default numbers
        PhoneNumber OneNumber = new PhoneNumber("1111111111");
        PhoneNumber NineNumber = new PhoneNumber("9999999999");

        this.addNumber(OneNumber);
        this.addNumber(NineNumber);
        this.printNumbers();
    }

    public boolean addNumber(PhoneNumber aNumber)
    {
        if (NbNumbers == MaxSize)
    return false;
        PhoneArray[NbNumbers]  = aNumber;
        NbNumbers++;
        return true;
    }

    public boolean removeNumber(int position)
    {
        int i;

        if (position > NbNumbers)
    return false;
        if (position == NbNumbers)
    {
        PhoneArray[position] = null;
        NbNumbers--;
        return true;
    }
        for (i=position; i<NbNumbers; i++)
    PhoneArray[i-1] = PhoneArray[i];
        NbNumbers--;
        return true;
    }

    public void printNumbers()
    {
        int i;

        if (NbNumbers == 0)
    System.out.println("Empty List");
        else
    for (i=0; i<NbNumbers; i++)
        System.out.println("Phone n. "+i+": "+PhoneArray[i].toString());
    }
}
```

7.3 Task B1

Write a PhoneNumberFormat (PNF) class by reusing a Java API class.

- A PNF object will be able to format some strings
- It will for example format ”1273275779” into ”Brighton 275.779”

You HAVE to reuse a Java API class to write this class.

7.4 Task B2

```
import java.awt.*;
import java.applet.*;

public class PhoneWidget extends Applet
{

// The interface attributes
    TextField input = new TextField();
    Button OK = new Button("OK");
    Label output = new Label("");

// The format
    PhoneNumberFormat theformat = new PhoneNumberFormat();

// The result
    String result;

    public void init()
    {
        setLayout(new GridLayout(3,1));
        add(input);
        add(OK);
        add(output);
    }

    public boolean action(Event evt, Object arg)
    {
        if ("OK".equals(arg))
    {
        result = theformat.format(input.getText());
        input.setText("");
        output.setText(result);
    }
        return true;
    }
}
```

Here is a PhoneWidget applet, which will be used
in an 'Organizer' program.

- A PhoneWidget object allows the user to type in a telephone number, such as 1273275779
- This number is then formatted using a PhoneNumberFormat object
- The result, in this case "Brighton 275.779", is displayed on screen

Write the PhoneNumberFormat class.

You can reuse an existing class file from the Java API if you want.

8 Appendix 2: Open-ended questions on reuse

At the end of the experiment, we asked three open-ended questions about software reuse to the subjects. Here is a summary of their answers. Each piece of answer has been included, and answers that occurred twice or more are presented as such (x 2, x 3, etc.).

What are, in your opinion, the good aspects of the API pages as a reuse tool?

General remarks

- The source code is free, easily available and frequently updated.
- It's useful for reusing large quantity of code, it gets programmers interested in reusing code.
- It's a good reference book, but nothing more.

Remarks to take into account for the design of the tool

- It's easy to use (x 3), easy to navigate (x 3).
- It's a good documentation (x 3), definitive, comprehensive list, all the methods are in (x 2), it is well structured (x 2), the descriptions are uniform, i.e. they all have the same layout. The class hierarchy at the top of each class description is a good thing. There are lots of links between classes.
- lots of the names are self-explanatory. It shows correct syntax.

What are, in your opinion, the bad aspects of the API pages as a software reuse tool?

Education is necessary for good software reuse

- It's difficult to use for 1st time programmers or users (x 2). It lacks some explanation of how the whole documentation works.
- It's difficult to use general code for specific purposes. It's quicker to write short new classes than long general/abstract class where you have to delete huge parts. It's only available on the Internet.

Remarks to take into account for the design of the tool

- It's easy to get lost, and difficult to find something you don't know exist. It lacks of search facility (x 3).
- The documentation is too complicated sometimes, too long. There is some heavy, technical terminology (x 2). There's no description of what packages are associated with. It lacks examples (x 4). The descriptions don't say anything about code, it's only for functions calls.

What should a perfect reuse tool look like?

General remarks

- It should be easy to navigate through (like a browser, but not necessarily using HTML), and to find something that suits you (x 2).
- It should have a GUI for today's level of programming, icons, images you can relate to (x 2), similar to Win95, point and click.
- It should make some provisions for first time users, be off-line, and be free
- It should support a whole range of languages, not just Java, have some links to other software reuse sites, and be cross-platform (x 2). It should be possible to add new (documented) classes.
- It must create templates for the most frequently used code.

Remarks to take into account for the design

- It should have some easily understandable descriptions, with the complete list of all the methods and variables, and the methods should point at other methods which could be of use. The descriptions must use plain English, which is quicker to understand (*this has been proved wrong in some studies*). It should include the actual code as well as the description.
- There should be an (intelligent) search engine which would suggest components (x 4). Something like dBASE IV. It should always propose different solutions, so that the user does not get trapped in one not-so-good solution. It should assist but not be intrusive.
- It should include some editing tools (i.e. a simple way of taking the code of a component and modify it), and a built-in compiler.

References

Burkhardt, J. M., & Detienne, F. (1994). La reutilisation en genie logiciel: une definition d'un cadre de recherche en ergonomie cognitive..

C. Hoadley, M. Linn, L. M., & Clancy, M. (1995). When, why and how do novice programmers reuse code?. http://obelisk.berkeley.edu /tophe/ESP6/ESP95Final.html.

McIlroy, M. D. (1968). Mass produced software components. In Naur, P., & Randell, B. (Eds.), *Software Engineering: Report on a Conference by the NATO Science Committee (Garmisch, Germany, Oct.)*, pp. 138–150. NATO Scientific Affairs Division, Brussels, Belgium.

Mulholland, P. (1995). *A Framework for describing and evaluating software visualization systems: a case-study in prolog*. Ph.D. thesis, Open University.

Sutcliffe, A., & Maiden, N. (1990). Software reusability: Delivering production gains or short cuts. In D. Diaper, D. Gilmore, G. C., & Shackel, B. (Eds.), *Human-Computer Interaction - INTERACT'90* Amsterdam, Netherlands. Elsevier.

In: Retkowsky, F. P. (Ed.). (1998)
The 11th White House Papers: Cognitive Science Research Paper 495
School of Cognitive & Computing Sciences, University of Sussex

Structural Knowledge in Prolog

Pablo Romero*
juanr@cogs.susx.ac.uk

School of Cognitive & Computing Sciences
University of Sussex
Brighton
BN1 9QH

Abstract Knowledge about which parts of the program text are important to experienced programmers might give clues about the nature of their structural knowledge and also might be useful for the design of instructional systems for programming. There have been several studies that have suggested that programmers of procedural languages base the mental structures of the programs they understand in the idea of Programming Plans. Experienced programmers are said to posses a more advanced knowledge about these structures. There is also evidence that there are parts of these plans that are focal to programmers' structural knowledge. It is not clear that this is the case for Logic Programming, though. Prolog has some important differences when compared to procedural languages. It does not have obvious syntactic cues to mark blocks of code (begin/end, repeat/until, etc). Also, its powerful primitives (unification and backtracking) and the extensive use of recursion might influence the way programmers comprehend Prolog code in a significant way.

This paper reports an experiment that tried to characterise the nature of the structural knowledge that Prolog programmers build at early stages of the comprehension process by exploring which parts of the code are more important to them. The results suggest that Prolog Schemas, a construct related to data structure relationships, is relevant to Prolog programmers. These results contrast with those obtained for declarative languages, where the stress seem to be on functional information.

Keywords: empirical studies of programming, program comprehension, logic programming.

1 Introduction

Program comprehension is a skill that is central to several programming activities, so having a clear picture of comprehension as a cognitive process seems to be a prerequisite to building models of programming tasks such as debugging, modification, reuse, etc. Yet comprehension has been mainly studied for languages belonging to the structured programming paradigm.

Studying programming languages very different from the structured paradigm can offer interesting ways to test the findings of the area. The dominant model used to represent the structural knowledge that programmers build of the programs they understand is Programming Plans. However, this theory has yet to be tested for programming paradigms other than the structured one. A programming language significantly different from this main trend is Prolog. Prolog is in a class of its own because of its declarative nature and its powerful primitives but programmers often find it odd and difficult.

*Supported by a grant from the Mexican Council for Science and Technology, CONACYT

There are several models of structural knowledge proposed for Prolog, however, these models are not supported by psychological evidence. The experiment described in this paper tried to find out which of these models is more relevant by exploring which parts of the code are more important to Prolog programmers.

Characterising the mental model of experienced Prolog programmers can offer an interesting point of comparison and can also help to improve the way this language is taught.

This document is divided into three sections. The next section gives a brief account of psychological studies on program comprehension. The following part describes the experiment settings and results. The final section analyses the experiment results and compares them to those reported for other programming languages.

2 Studies on Program Comprehension

Program comprehension is an important skill for almost all programming tasks. One of the earliest models proposed for program comprehension was by Brooks (1983). Brooks proposed a theoretical framework to understand behavioural differences in program comprehension. He regards comprehension as a process of domain reconstruction. This reconstruction involves establishing mappings from the problem domain to program domain and some other intermediate domains. This process of establishing mappings consists of generating and refining hypotheses about the executing program and its relation to other domains. Hypothesis refinement is performed in a top-down fashion. This process begins with a primary, top-level hypothesis which is decomposed in several subsidiary, more specific hypotheses. The generation of hypotheses is performed by retrieving known structural units from the programmers knowledge. These structural units are used to generate more hypotheses or are matched against the program's code. According to Brooks, when comprehending, programmers look for specific patterns of code which can confirm the proposed hypothesis. Brooks mentions that these structural units are the same as those that are retrieved and used for code generation. However, in the case of comprehension, the process of assembling and modifying these structures is greatly reduced. Wiedenbeck (1986) gives empirical evidence that supports this theory of program comprehension. In her study, novices and experienced programmers tried to understand and memorise a short Pascal program. After this study period, they were asked to recall as much as they could of the program code. The results showed that experienced programmers, unlike novices, recalled beacon lines (relevant structural units) much better than other parts of the code. This fact seems to support the idea that certain key parts do exist in programs, and that these key parts would guide the comprehension process for experienced programmers.

There have been several studies that propose that these structural units of knowledge and the associated patterns of code are related to a concept known as 'programming plans'. These studies suggest that a programmer builds several mental models of the program, but among these different mental models the one that emphasises plan information seems to be very important. Plans are proposed as internal representations of the programmer's knowledge that consider the program as a hierarchy of meaningful chunks of code. These chunks are frames of stereotypical program fragments based on the program's semantic structures (also related to data-flow and functional relations). Plans can be seen as data structures that represent generic concepts stored in memory.

Table 5 gives an example of a plan. Pennington (1987), Gilmore and Green (1988) and Davies (1990) give empirical evidence of the existence of plans as mental constructs. Pennington (1987) shows that plan relations dominate programmer's mental representations of programs. Gilmore and Green (1988) give evidence that suggests that some programming languages might be better suited than others to reflect plan related issues. The study by Davies (1990) suggests that it might be the programmers' previous training rather than the programming language chosen which is the factor that determines a

Goal:	find an occurrence of ?x	
	CODE	PLAN TERMS
Plan:	?found := false	initialise to not found
	loop through category of ?x	
	if ?x then	
	?found := true	set it to true
	... := ?x	use it

Table 5: Plan description
From Gilmore and Green (1988)

```
p(X):-
    p(Y),
    g(Y,X).
```

Figure 1: The *after* technique

From Bowles and Brna (1993)

plan-like mental model preference.

The link between Brooks's theory of program comprehension and programming plans is given by Davies (1993, 1994) when he suggests that the comprehension beacons that Brooks mentions can be considered as the external analogue of the internally represented focal structures of the programmer's plan based structural knowledge.

The issue of generalisation of plans across different languages is central to this study. There have been several attempts to investigate the issue of plan-like constructs in Prolog. The studies by Brna, Bundy, Todd, Eisenstadt, Looi, and Pain (1991) and by Gegg-Harrison (1991) are related to this aspect.

Brna et al. propose that Prolog programmers use patterns of code in a systematic way when doing programming. These patterns of code are called "techniques" and their use is said to be one of the differences between experienced and novice Prolog programmers. According to Brna et al., experienced programmers have picked up a wide variety of these techniques and are able to recall and use them for solving new problems. However, they only give introspective evidence of the existence of techniques. The concept of techniques is further developed by Bowles and Brna (1993). Bowles and Brna describe a set of common programming techniques in detail. An example of a basic programming technique is given in figure 1. This technique is called the *after* technique because the value of *X* is constructed in the subgoal *g* from the value *Y* produced in the recursive call. Figure 2 illustrates an occurrence of this technique in the predicate *length/3*.

The concept of Prolog techniques has been used in a programming tutor built as a text editor. This tutoring system has been evaluated in a study by Ormerod and Ball (1996). This evaluation seems to indicate that this programming tutor promotes the learning of Prolog at early stages of instruction, but it did not seem to be of much help at intermediate levels.

The concept of programming techniques does not seem to be at the same level of programming plans. Techniques seem to be more general and basic, and plans more specific and sophisticated. It seems that programmers of other programming paradigms also need to know about basic programming knowledge.

```
length([],L,L).
length([H|T],L0,L):-
                    L1 is L0+1,
                    length(T,L1,L).
```

Figure 2: An occurrence of the *after* technique

From Bowles and Brna (1993)

```
schema_C([E|T],E,<< &1 >>).
schema_C([H|T],E,<< &2 >>:-
                          <not(EH)>,
                          <pre_pred(<< &3 >>,H,<< &4 >>)>,
                          schema_C(T,E,<< &5 >>),
                          <post_pred(<< &6 >>,H,<< &7 >>)>,
```

Figure 3: An example of a simple Prolog schema

From Gegg-Harrison (1991)

For example, they need to know how to build up the value of a variable as a function of other variables through calls to intermediate procedures. However, this programming knowledge is more basic, than, for example finding the occurrence of a variable inside a complex data structure. Another important difference is that while plans are related to the function of the program, techniques seem to comprise knowledge about data-flow relations.

A construct that seems to be more similar to programming plans is the programming schema proposed by Gegg-Harrison (1991). Gegg-Harrison describes a set of common programming schemas for list processing. A specific example from such set is given in figure 3. In this example, $<< \&n >>$ denotes any number of Prolog arguments, and clauses surrounded by $<>$ are optional. This example deals with the task of processing a list until the first occurrence of an element is found. The base case ensures that *E*, the element that is being searched for, is found. The second clause optionally checks that the list element being processed is not the one which is being looked for, performs an optional process, makes a recursive call trying to find the element in the tail of the list and calls a second optional process. Coincidentally, this schema is very similar to the example of a programming plan given in table 5. The difference with plans related to a goal oriented analysis of programs seem to be that schemas comprise knowledge about typical operation performed over familiar data structures and plans have to do with functional information.

It can be noted that neither Gegg-Harrison nor Bowles and Brna give theoretical evidence of their proposed programming structures. Ormerod and Ball evaluate the programming environment built around the idea of techniques, but do not provide direct evidence of the existence of these constructs.

One of the few theoretical studies for Prolog is by Bergantz and Hassell (1991). In this study, Bergantz and Hassell analysed the verbal protocols of three expert programmers looking for a model

of program comprehension for Prolog that could characterise the flow of information and the temporal ordering of information relationships of this process. The information relationships that they considered were data-flow, control-flow, data structure and function. The results of this study seem to indicate that data structure relationships dominate the beginning of the comprehension process sessions while function relationships seem to dominate the final part of them. The authors suggest that data structure relationships play a dominant role at the beginning of the comprehension process because it is at this stage when programmers form a program model based on the detection of data structure information. A domain or real-world model of the program is constructed based on the detection of function relationships, and although the formation of this model occurs throughout the whole session, it seems to intensify at late stages of it.

The experiment described in the next section is not concerned with the whole comprehension session, but only with the beginning of it. It considers similar kinds of information but looks for evidence of them in the program code, rather than in the subjects' verbal protocols.

3 The experiment, method

This experiment was concerned with finding out which parts of the Prolog code were more relevant to programmers when performing program comprehension. Several segments of Prolog programs were related to different kinds of information relationships and the recollection account of the programmers was inspected to find out which of these code segments, and therefore information relationships, were more relevant under the conditions of the experiment.

This study is similar to the one by Wiedenbeck (1986), but there are several important differences. Similarly to Wiedenbeck's experiment, this study asked subjects to understand and memorise Prolog code, and then recall what they could of it. Several segments of this code were related to different models of structural knowledge of Prolog. These segments and models are related to functional, data-flow, data structure and control-flow relationships. The success of recollection of these segments was compared to establish which one of these proposed structural knowledge models was more important to Prolog programmers. However, there are important differences with Wiedenbeck's referred study. Probably the most important is that this is not a hypothesis testing experiment. There were no strong predictions about which structure seems to be the base of Prolog programmers' mental model. This study set out to explore which one of the considered models, if any, is more important for Prolog programmers. Another important difference is that Wiedenbeck used only one program in her experiment. Her results might have to do with a particular characteristic of that program rather than with the nature of the structural knowledge for comprehension in general. This experiment uses three programs, and although they share certain characteristics, if the results are similar for the three of them, there is more room for generalisations. Yet another difference is that this experiment considered a control group. This was due to the fact that there is a memorisation part in the experimental task. If the results are not as clear cut as Wiedenbeck's, then it could not be easy to determine whether these results are explained by memorisation or comprehension. A group that would only use memorisation (a group of non-programmers) will allow a comparison that might give an idea of the influence of such effect in the experiment.

3.1 Aims

The aim of this experiment is to find out which model of structural knowledge seems to be related to the mental model Prolog programmers build at the beginning of the comprehension process session. This experiment considered four different structural models of program comprehension knowledge and tried to identify, through a recalling task, which one of these proposed models is more important to Prolog programmers.

According to Brooks (1983) and Wiedenbeck (1986) programmers use beacons in the code to guide their comprehension process. Davies (1993, 1994) suggested that these beacons can be considered as the external analogue of the internally represented focal structures of the programmer's structural knowledge. This structural knowledge seems to be based in the idea of programming plans for the case of procedural languages, but this might not be the case for Prolog. It could be that either the idea of a plan has not counterpart in Prolog or the nature of plans is very different in this language.

3.2 Design

As mentioned before, this is not a hypothesis testing experiment, however, the experimental design of the main part of this study considered one independent variable, level of programming skill (expert, novice and non-programmer), and four dependent variables, the success of recollection of the four proposed mental model structures.

3.3 Subjects

There were 30 subjects in this experiment: 10 expert and 10 novice Prolog programmers and a group of non-programmers. The group of experts had at least three years of Prolog experience and each of the novices had taken a three month introductory course to Prolog. The group of non-programmers did not know anything about computer programming. It has to be noted that the novice population was inexperienced in Prolog, but not in programming in general. Most of them knew three or more programming languages apart from Prolog. Also, they seemed to have a much more recent contact with Prolog than the experts. Experts had not used Prolog for years, while for novices this gap was only of three or four months.

3.4 Materials

The experiment used three different programs. These programs were in average 23 lines long and every one of them accomplished a specific task. The first of these programs was a Prolog version of the 'rainfall' program. The second performed a binary to decimal conversion. The third program, the one in figure 4 implemented the bubble sort.

All of these programs were analysed in terms of focal structures of plans, techniques, schemas and focal control-flow points. In the case of the focal structures of plans a goal analysis was performed to identify the plans and the focal lines of these plans were selected as the beacons (as Davies suggests). For the techniques case, a techniques analysis was performed according to the definition by Bowles and Brna (1993). The segments of code that implemented the techniques were selected as the relevant parts of the program. For the case of schemas, Gegg-Harrison (1991) gives a definition in which compulsory and optional parts of these structures can be identified in Prolog programs. The compulsory parts were chosen as the important segments. Finally, the selection of beacons for the case of the control-flow relationships was straightforward. The critical control-flow points, the lines in which recursion was invoked or stopped, were considered as the important segments.

It is worthwhile noting that some of these structures' propose units other than lines as instances of relevant segments of code. Sometimes an instance of a structure can be scattered through several lines of the program.

These different structures seem to be related to different kinds of information in the program. Plans and focal structures derived from a goal analysis are related to function. Techniques, which focus on how instantiations of Prolog objects are linked through the program, seem to be based on data-flow information. The stress on well known data structures and the operations performed over them make

```
/* do_sort(-) */

do_sort(SortedList):-
        write('enter sorting data'),
        read(Key),
        next_value(Key,List),
        bubblesort(List,SortedList).

next_value(stop,[]).

next_value(Key,[Key|Rest]):-
        write('enter sorting data'),
        read(NewKey),
        next_value(NewKey,Rest).

bubblesort(SortedList,SortedList):-
        verify_sorted(SortedList).

bubblesort(List,SortedList):-
        swap(List,List1),
        bubblesort(List1,SortedList).

verify_sorted([]).

verify_sorted([X]).

verify_sorted([X,Y|Rest]):-
        X =< Y,
        verify_sorted([Y|Rest]).

swap([X,Y|Rest],[Y,X|Rest]):-
        X > Y.

swap([Z|Rest],[Z|Rest1]):-
        swap(Rest,Rest1).
```

Figure 4: A version of the bubble sort program.

schemas related to data structure information. Finally, control-flow relations seem to be highlighted by the points were recursion takes place.

Finally, as the experimental task includes the identification of the programs' functionality, 'disguised' versions of these programs have to be presented to the subjects. The criteria to 'disguise' these programs is similar to the one used by Wiedenbeck (1986).

3.5 Procedure

The programmer subjects of this experiment performed three similar sessions. In each session, they were given a hardcopy of the experimental program and were asked to study and memorise it. This study period lasted 3 minutes. After this, the subjects were given 5 minutes to recall and write down what they could remember of the program. Finally, these subjects used another period of 3 minutes to write down a short explanation of what, according to them, the program does. These estimated times were calculated following the same proportions as the times for the Wiedenbeck (1986) experiment.

The control group followed a slightly different procedure. They were not instructed to comprehend but only to memorise the programs. Also, they were not asked to write down an explanation of what the programs do.

As this was a pen and paper exercise, the collected data was the hand written account of both the recollection and the explanation of the functionality of the program by the subjects. The recollection account was analysed for each one of the proposed structures. For this analysis, the success rate of recollection of each one of the structure's instances was calculated for every subject. These instances were considered as correctly recalled if the subject wrote a verbatim copy of the program code for these segments; however, commas, dots, spaces and indentation were not considered in determining this success of recollection. Also, if the subject recalled a procedure or variable with a different name but was congruent through all the program with this modification, then the modification was not considered as an error.

As mentioned before, structure instances comprise several elements which could be scattered through several lines of the program. Only recollection of whole instances was considered as successful recollection. If, for example, 90% of a particular instance was correctly written down, this instance was not considered as correctly recalled. This strict criteria for the recollection of instances was considered as appropriate because this study is interested in which structures, as chunks, are relevant to programmers.

The programmers' account of the programs functionality was analysed in terms of its correctness. Each of the programs performs several functions. The *rainfall* program, for example, reads a list of values and obtains an average of these values and their maximum occurrence. The *conversion* program validates a binary number and converts it to its decimal equivalent and the *bubble sort* program (figure 4) reads a list of values and performs a sort procedure over them. The subjects' functionality description was required to mention these major functions in order to be considered correct. For example, for the *rainfall* program, statements equivalent to 'read a list of values', 'obtains an average' and 'obtains the maximum' were searched in the subjects' description. Only if the three statements were identified the account was considered correct. A similar criteria was used for the other two programs.

3.6 Results

The data of this experiment was analysed in three parts. The first part dealt with the success rate in identifying the function of the program by the two groups of programmers. The second part was concerned with comparing the success percentages of recollection for the four structures taken into account. The third part compared the percentage of recollection of each structure versus the percentage of recollection of the program's lines.

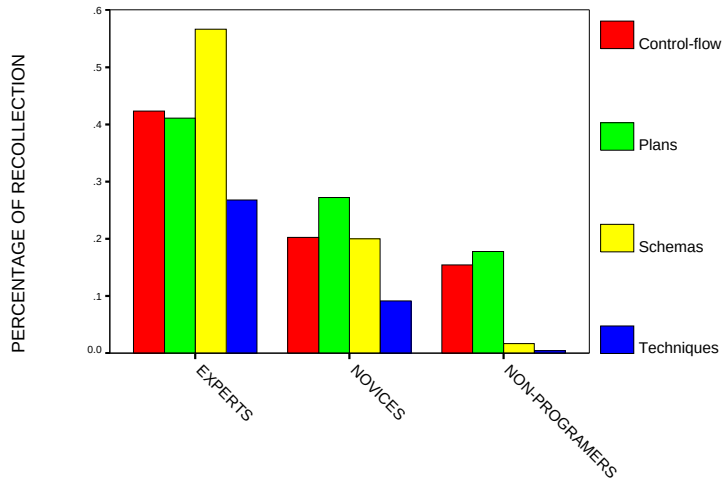


Figure 5: Percentage of recollection for structures

It was not surprising that the experts' group were more successful in identifying the function of the programs. Their percentage of correct identification was 70%, while for the novices it was 23.3%. A Chi-square test showed that this difference was highly significant ($F(1) = 15.15$, $p << .01$).

The results of the main part of this study are illustrated in figure 5. This figure shows the differences in success of recollection for the four structures considered and for the three experimental groups. As mentioned before, every instance of structure recalled by the subjects was considered correctly remembered if all of its elements were correctly recalled. It can be seen that schemas was the best remembered structure for experts. This result was similar for the three programs considered in this experiment.

As this part of the study considered one independent variable and four dependent variables, and these dependent variables were highly correlated, a MANOVA test was applied for this analysis. Only the data for the expert and novice groups fitted the requirements of the MANOVA test, so the statistical analysis took only these two groups into account. Also, as mentioned earlier, the experimental task comprised a memorisation component. This component seemed to influence the recollection task. The subjects' recollection of the code, for example, showed a tendency to remember better the beginning of the programs as well as certain short lines (read statements and some base cases of predicates). This effect seem to be related to memorisation rather than to comprehension of the code. The structures' instances were, in general, evenly distributed over the code, but the techniques structures, for example, seemed to concentrate on the bottom part of the programs, and therefore they were likely to be affected by this memorisation effect. The relative high percentage of recollection for focal structures of plans and for control-flow critical points for the non-programmers case (figure 5) seem to be explained by this effect. So, the statistical analysis included covariates in order to account for this memorisation effect.

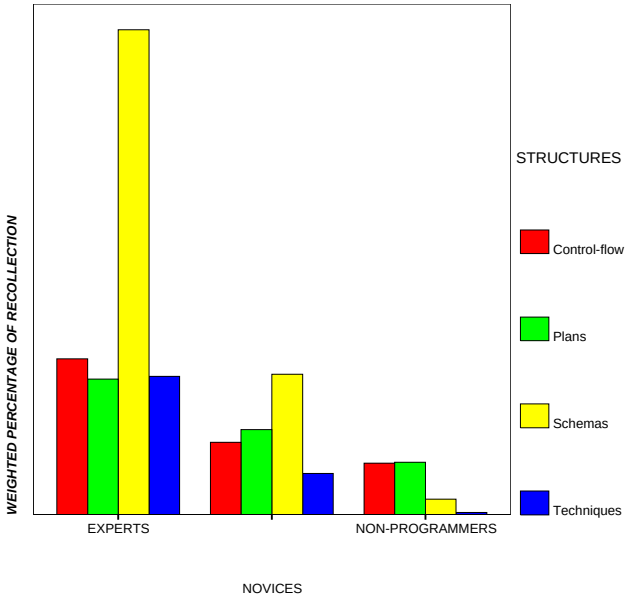


Figure 6: Weighted percentage of recollection for structures

The MANOVA analysis indicated that there were no global group differences. Univariate F-tests with their significance levels adjusted by the Bonferroni *t* procedure showed that schemas was the only structure for which there is a significant difference between experts and novices ($F(1,18) = 8.38$, $p < .05$). The regression component was not significant.

This analysis did not take into account the sizes of the structures. It was mentioned before that instances of structures normally comprised several elements. These elements were typically names of variables, names of procedures and list notation occurrences. The number of elements that the different structures comprised was similar to the average number of elements of the program lines for three of the structures (the average size of lines in number of elements is 3.7, of techniques is 4.8, of control-flow critical points is 3.9 and of plans focal structures is 3.7). However, the mean size of schemas was more than twice this average (8 elements per structure). It seems that remembering a structure with ten elements correctly is more difficult than remembering one with two, for example. However, this fact was not taken into account in the reported analysis mainly because the results would have been similar. The main effect would have been that the differences for the schemas case had been more significant. Figure 6 shows the same data as figure 5, only the percentages of recollection take into account how big was the structure instance.

Although a comparison between the four structures gave an idea about the relative importance of each one of them, some measure of their absolute importance seems to be needed. This measure might be provided by comparing the recollection rate of each structure with the recollection rate of lines outside structures. It was mentioned before that the average number of elements of lines and structures was similar (with the exception of schemas), therefore lines seem to be a neutral structure to which the proposed structures can be compared. For the third part of this study, each structure had a separate analysis. In this analysis, the percentage of recollection of each structure was compared with the percentage of recollection of lines that did not share elements with its instances. Figures 7 and 8, illustrate the results of

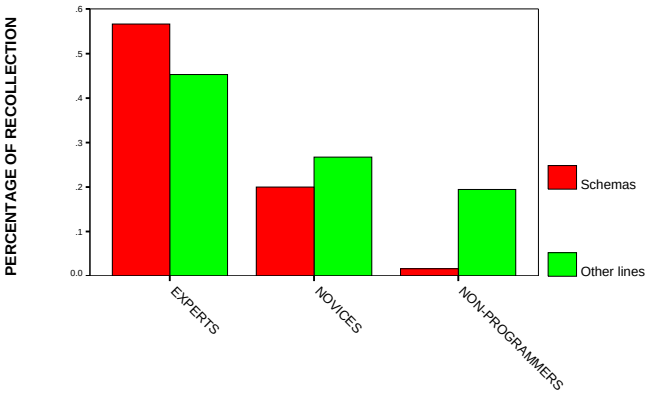


Figure 7: Percentage of recollection for schemas and lines outside them

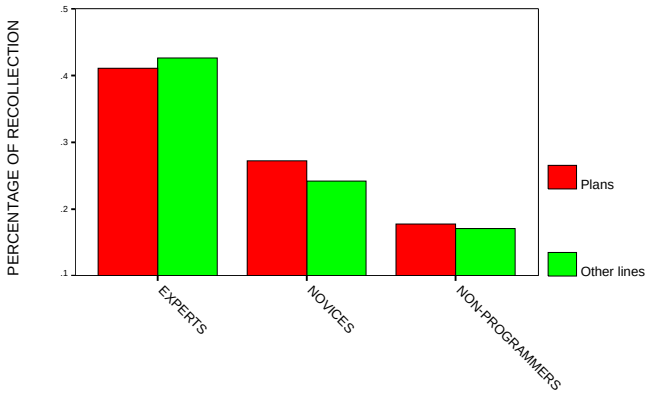


Figure 8: Percentage of recollection for focal structures of plans and lines outside them

these comparisons for schemas and plans. The results for techniques and control-flow look very similar to those for plans.

As direct comparisons between the groups of programmers had already been performed in the previous analysis, and also in order to avoid memorisation and size effects, the statistical analysis for this third part of the study focused in the rate of change of the difference between the recollection of structures and lines. For example, it can be seen that for the case of schemas (figure 7) this difference is negative for non-programmers and for novices, and positive for experts. It is therefore likely that the rate of change of this difference is significant. So the statistical analysis for each structure considered again one independent variable (level of skill), but this time only one dependent variable (structure-line percentage of recollection difference). For each one of the comparisons, a one-way ANOVA analysis was run after verifying that its assumptions had been met. The only case for which this rate of change among groups was significant was for schemas ($F(2,29) = 8.57$, $p < .05$).

4 Discussion

One implicit supposition of this study is that knowledge about different aspects of the program can be related to stereotypical constructions that can be identified in the code. In this aspect, it is different from studies on programming plans, because normally they only look for information related to functional aspects. This experiment took into account data-flow, control-flow, data structure as well as functional aspects and looked for evidence to find out which one of these seem to dominate the programmer's mental model at early stages of the comprehension process.

The results show that schemas, the stereotypical patterns of code related to data structure aspects, as whole structures or chunks seem to be important as a starting point in program comprehension sessions. Of the four structures considered, the performance of experts and novices in recalling code showed a significant difference only for schemas. Also, when comparing the difference between the percentage of recollection of each structure and of the program's lines outside these structures, schemas was the only structure for which this rate of change between groups was significant. It seems that these results show that schemas become more important for the comprehension process as Prolog programmers develop higher levels of skill.

Following Brooks's hypothesis it could be said that schemas seem to be the beacons that Prolog programmers use to guide their comprehension process. This contrasts with procedural languages, where plans and their focal structures seem to be important (Wiedenbeck, 1986; Davies, 1994).

It could be said that schemas, as structures which comprise knowledge about typical data structures and the operations performed over them, are a kind of data structure oriented plans. The main difference between the plans that seem to be relevant to programmers in this study and the ones identified as important for procedural languages seems to be that the former are based on data structure information while the latter are oriented towards functional relationships.

The finding that information related to data structures is important at early stages of the comprehension process is in agreement with the results of Bergantz and Hassell (1991). They found that 'data structure relationships play a dominant role at the beginning of the comprehension process' for the case of Prolog.

It is interesting to compare these results with those obtained by Wiedenbeck (1986) because the experiment reported in this paper is similar to hers. She found that functional information seem to dominate the construction of the programmer's mental model for the case of Pascal. Figure 8 can be used to make a closer comparison. With a graph similar to this one, Wiedenbeck shows how functional information is very important only for experienced programmers. These results could not be replicated in the present study. It could be argued that the experimental programs were different, but the results

when considering only the bubble sort program, which is similar to the sort program Wiedenbeck uses, are basically the same to those obtained when taking into account the three programs. So it seems that the key difference is the programming language considered.

It seems reasonable to think that in absence of any other information (neither internal nor external documentation, and with variable and procedure names that do not help much to grasp the meaning of the program) patterns of typical operations performed over familiar data structures can be very important to start making sense of the code. This lack of documentation and meaningful variable names seems to be an important issue for Prolog. Green, Bellamy, and Parker (1987) mention that Prolog, due to its poor 'role-expressiveness', is specially sensitive to naming style ('Salient variable names are almost the only method of making a Prolog program "role-expressive" and thereby revealing the plan structures' Green et al. (1987)). The obvious question is how naming style influences the program comprehension mental model, or in other words, which aspect of the program (data-flow, control-flow, data structure or function) would be relevant for programmers when meaningful variable names are considered.

The conclusion that data structure information seem to dominate the construction of the comprehension mental model does not mean that other kind of information is not important as well. In figure 5 it can be seen that techniques, for example, show a considerable difference between the novice and the expert population percentages of recollection. It is possible that at a later stage of the comprehension session, data-flow or functional relationships are of particular importance.

Another important point to take into account is that the nature of the experimental task was not very common for programmers. Normally a programmer does not try to understand a program just for the sake of it. Programmers might exhibit a different behaviour when asked to understand a program and then modify it, for example.

Although there were some memorisation effects, these do not seem to affect the results of the experiment. It seemed a good idea to control for this effect in some way other than a group of non-programmers. For example, by presenting the groups of programmers with non-sense as well as with 'normal' code and then comparing their performance between these two cases. However, it is difficult to reconcile the idea of non-sense code with a goal oriented analysis (or indeed any kind of analysis) of this kind of code. It has to be remembered that a goal oriented analysis, a schema type analysis, a control-flow and a techniques type analysis were a prerequisite to find out about the performance of the subjects.

The other kind of effects, those due to the size of structures, only seem to reinforce the importance of schemas as relevant structures for Prolog programmers. However, it seems that further experimentation is needed to confirm these results. Also, it is necessary to relate this findings to a more common measure of programming performance. As mentioned before, programmers do not normally engage in program memorisation tasks or in comprehension without a definite aim. It would be interesting to find out whether knowledge about schemas affects programming tasks such as debugging or program modification. If this were true, the next step would be to know how programmers use this knowledge.

5 Conclusions

This paper reported an experiment that explored which one of four programming knowledge structures seem to be important at early stages of the program comprehension process for Prolog programmers. These structures were related to functional, data-flow, data structure and control-flow information. It seems that information related to data structures is important in this case. The experiment involved a program comprehension and memorisation task followed by the recollection of the program. There were significant differences when comparing the performance of experts and novices programmers for the case of schemas, a structure that emphasises data structure relationships. These results suggest that data

structure relations are important at the beginning of the comprehension process for Prolog programmers.

The results of this experiment suggest that the mental model that Prolog programmers build at early stages of the comprehension process is different from the one that programmers of procedural languages construct. The former seems to be based on data structure relations while the latter, according to Wiedenbeck (1986) and Davies (1994), is related to functional information. This conclusion needs to be confirmed and its importance needs to be related to a more common programming task such as debugging or program modification. Only when it is established that these findings are important for actual programming tasks, this information could be applied to influence programming instruction and the design of programming tools.

References

Bergantz, D., & Hassell, J. (1991). Information relationships in PROLOG programs: how do programmers comprehend functionality?. *International Journal of Man-Machine Studies*, *35*, 313–328.

Bowles, A., & Brna, P. (1993). Programming plans and programming techniques. In *Proceedings of the AI-ED 93*.

Brna, P., Bundy, A., Todd, T., Eisenstadt, M., Looi, C. K., & Pain, H. (1991). Prolog programming techniques. *Intructional Science*, *20*(2), 111–133.

Brooks, R. (1983). Towards a theory of the comprehension of computer programs. *International Journal of Man-Machine Studies*, *18*, 543–554.

Davies, S. P. (1990). The nature and development of programming plans. *International Journal of Man-Machine Studies*, *40A*, 423–442.

Davies, S. P. (1993). Models and theories of programming strategy. *International Journal of Man-Machine Studies*, *39*, 237–267.

Davies, S. P. (1994). Knowledge restructuring and the acquisition of programming expertise. *International Journal of Human Computer Studies*, *40*, 703–726.

Gegg-Harrison, T. S. (1991). Learning Prolog in a schema-based environment. *Instructional Science*, *20*, 173–192.

Gilmore, D., & Green, T. (1988). Programming plans and programming expertise. *Quarterly Journal of experimental psychology*, *40a*, 423–442.

Green, T. R. G., Bellamy, R. K. E., & Parker, J. M. (1987). Parsing and Gnisrap: a model of device use. In Olson, G. M., Sheppard, S., & Soloway, E. (Eds.), *Empirical Studies of programmers, second workshop*, pp. 132–146 Norwood,NJ. Ablex.

Ormerod, T. C., & Ball, L. J. (1996). An empirical evaluation of TEd, a techniques editor for Prolog programming. In *Empirical Studies of programmers, sixth workshop*, pp. 147–162 Norwood,NJ. Ablex.

Pennington, N. (1987). Stimulus structures and mental representations in expert comprehension of computer programs. *Cognitive Psychology*, *19*, 295–341.

Wiedenbeck, S. (1986). Beacons in computer program comprehension. *International Journal of Man-Machine Studies*, *25*, 697–709.

In: Retkowsky, F. P. (Ed.). (1998)
The 11th White House Papers: Cognitive Science Research Paper 495
School of Cognitive & Computing Sciences, University of Sussex

Tangent Point Tracking for the Driving Task

Hanson Schmidt-Cornelius
hanson@cogs.susx.ac.uk

School of Cognitive & Computing Sciences
University of Sussex
Brighton
BN1 9QH

Abstract Driving around bends requires sensory-motor coordination to follow the curve and steer smoothly along the road. Prior research has established the need for an anticipatory system, based on the "tangent point" on the inside of road curves. In this paper, I investigate a computational mechanism which tracks tangent points in image sequences. The tracking process first detects low level image features, then a least squares algorithm is used to fit a polynomial to the extracted image clusters, finally the calculated tangent points are tracked using a history to filter out polynomials that contain noise or other scenery. This exploits the observation that the tangent point moves very slowly over the sequences of images. In conclusion, the mechanism has been shown to work well and future work will focus on more biologically inspired strategies for the tracking process.

1 Introduction

Although driving seems effortless for experienced drivers, it is clear from attempts to automate this process that the task requires sophisticated sensory-motor coordination to avoid crashes and steer smoothly along the road (Jochem, Pomerleau, Kumar, & Armstrong, 1994; Jochem, Pomerleau, & Thorpe, 1993; Pomerleau, 1995). Donges 1977 model (Donges, 1978) is still the basis for much research on driving tasks. In this model, there is both immediate visual feedback and anticipatory feed forward control of steering. The anticipatory signal refers to the curvature of the road ahead (Godthelp, 1986). The success of these models stems from the complementarity of the 2 components. The anticipatory mechanism measures and matches the steering response to the curvature of the upcoming bend but cannot guarantee that the car will keep centred in its lane. The feedback mechanism is then just required to make the small adjustments necessary from moment to moment to keep the car smoothly on track. This feedback can operate at low gain and thus avoid problems of instability inherent in using immediate feedback alone. Just how drivers calibrate or learn the relationship between road appearance and their steering response is still largely unknown. Indeed, Serafin (Serafin, 1993) concluded that there was little relationship between gaze direction and road curvature but more detailed studies (Land & Horwood, 1995, 1996; Land & Lee, 1994) have provided clear evidence that drivers do attend differentially to bends on the road. Drivers seem to rely particularly on the inside of corners (tangent points) and appear to gaze at these points, 1 to 2 sec. before entering a bend. Once in a bend, the driver's gaze returns to the tangent point regularly. In this paper I introduce a computational mechanism which tracks tangent points in image sequences, obtained from a car mounted camera. Section 2 gives a general introduction to the properties of tangent points and lists the three stages which constitute the detection and tracking process. The Subsections 2.1, 2.2 and 2.3 illustrate the tracking process in more depth. Section 3 demonstrates three road

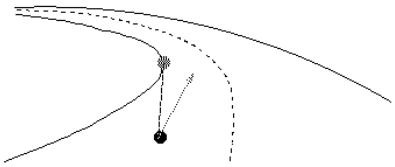


Figure 1: Tangent Point

scenarios to which the tracking algorithm is applied and analyses the effect of vibration to the tracking performance. Finally, Section 4 concludes the paper and reflects the main results.

2 Method

The tracking mechanism introduced in this paper makes use of well established and reliable image processing techniques and is designed to process image sequences with little scenery, such as shown in the images from Figure 6 and 7. Image sequences containing more scenery, such as shown in Figure 8 can also be processed and produces reasonable results.

A tangent point forms the location, where the driver's direction of gaze and the extreme inner point of a road bend touch, but do not intersect. The road schematic in Figure 1 illustrates this. The black dot represents an approaching car, and the grey dot represents the tangent point on the road bend. The dotted arrow indicates the car's current heading and the dashed line shows the driver's direction of gaze, which touches the tangent point.

As a car passes through a continuous bend, the tangent point appears to travel with the car. This property of the tangent point is crucial for the operation of the algorithm outlined here and will be explained in more depth in the following subsections.

The tangent point tracking algorithm is a sequential three stage process:

1. Process raw images with a mask and extract connected regions which have a strong contrast to the surrounding image.

2. Fit curves to the extracted regions and calculate potential tangent points.

3. Track tangent points over a history of images to determine their validity as road tangent points.

2.1 Low Level Processing

The first operation applied to the images is a combined mask for general smoothing and differencing along the horizontal axis. This emphasizes vertical features and reduces horizontal features at the same time. Figure 2 shows a road scene before and after this process. The road layout and especially the tangent point are emphasized in Figure 2b), allowing these features to be easily extracted in the following step.

Next, data is extracted from the processed images, to which curves (Fitzgibbon & Fisher, 1995; Lancaster & Salkauskas, 1986) are to be fitted. The greyscale images are first converted to binary images, which contain only black (image background) and white (image structures) regions. The white regions are clustered by an operator which detects eight-connected pixels. Noisy clusters which contain only a few pixels are removed and large clusters which lie adjacent but contain no common pixels, are connected. The connection is performed when most extreme pixels, along the horizontal or vertical axes

a) b)

Figure 2: Smoothing and X-Axis Differencing

of neighbouring clusters are located very close to each other. Joining merges clusters that may have been broken up by shadows from trees, lamp posts or noise in the raw images. An optional step in the low level processing phase can remove all clusters which are close to a larger region, by measuring the overlap of rectangular boxes around the image clusters. If the overlap exceeds a threshold, the smaller cluster is removed. This process can prevent close regions to be identified as separate tangent points, such as double lines, which sometimes separate single carriageways.

2.2 High Level Processing

The second phase estimates the location of tangent points in an image. First, a least squares algorithm (Lancaster & Salkauskas, 1986) is used to fit fifth order polynomials to extracted image clusters. The following assumptions are made at this point:

- The image array represents a coordinate system with the origin at the top left corner. The x-axis runs horizontal and the y-axis runs vertical along the image.
- The polynomials which are fitted to the image are of the form:

$$f(y) := \sum_{n=0}^{r} a_n y^n = x \neq 0$$

The applicability of the polynomial used to represent the curvature of the road is restricted, as it is only able to represent one y value for every x value, hence making it impossible to represent circular road layouts. This is however not relevant to the implementation here, as I am only interested in the tangent point just ahead of the car, which can be represented sufficiently with the type of polynomial used here.

The image clusters are probed at small and equal intervals, extracting only one y value for every x value. I represent the curves as fifth order polynomials, as trials have shown that this order produces optimal results for the approximation of road curvatures. Figure 3 shows a road scene and three graphs of corresponding polynomial fitting. In the raw image 3a), the road curvature and its tangent point are clearly visible. Figure 3b) shows a curve fitting approach with polynomials of second degree. The curvature of the polynomials are clearly not strong enough to approximate the tangent point accurately. Figure 3c) shows curve fitting with polynomials of fifth degree. The approximation of the tangent point and the curvature of the rest of the road is very accurate. Noise also appears to have little effect on the curvature of the polynomials. Figure 3d) shows a curve fitting approach with eighth order polynomials. The curvature approximations become inaccurate as the polynomials start to show an oscillating behaviour. The strong oscillation occurs when a polynomial tries to fit to image locations which are noisy and are not situated exactly on a smooth curve.

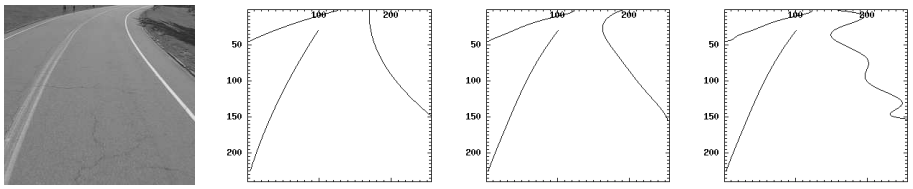


Figure 3: Degree of Polynomial

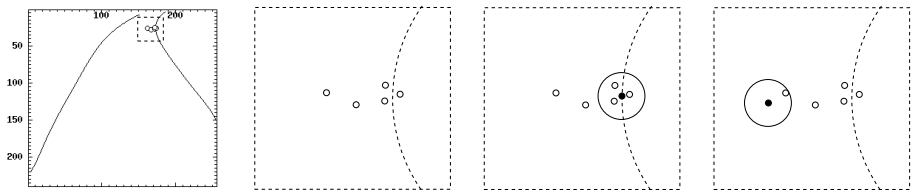


Figure 4: Tracking the Tangent Point

The extraction of potential tangent points from polynomials is a standard procedure and requires the first derivative (gradient). The tangent point, as shown in Figure 1, is the point in which the driver's direction of gaze and the extreme inner bend of the road, touch in one location, but do not intersect. It is mathematically defined (Reinhardt & Soeder, 1990) that the gradient of a polynomial in the tangent point is equal to the gradient of the tangent. Because the optic axis is close to the horizon, the required tangent point in the image occurs where the tangent of the curve is parallel to the y-axis. Mathematically, a vertical gradient can be tracked by setting the root of a polynomial's first derivative to 0. This produces the exact location of every tangent point with a vertical gradient. Some polynomials do however not have a gradient of 0 in the coordinate range of the image, they are subsequently removed from further processing.

2.3 Keeping Track of Tangent Points

By now, most of the irrelevant image data is removed, and the remaining polynomials contain tangent points in the region of the image coordinate system. Even in images which are cluttered with noise, there are seldom more than ten polynomials that contain tangent points in the image region. This last phase keeps track of polynomials with potential road tangent points and filters out polynomials that contain noise and scenery information. When following a road tangent point, one predominant feature is very soon apparent: The tangent point hardly moves its location over a sequence of images. This feature makes tracking the tangent point relatively easy, as potential tangent points must be extremely close to existing tangent points from previous images. When the program is initiated, a history of five images is generated, storing information on where tangent points were previously encountered. Until this history is generated, all potential tangent points are displayed. Once the history is available, the program only displays tangent points which are located close to tangent points in the history.

The example in Figure 4 illustrates the process of tangent point tracking. Figure 4a) shows a road scene which consists of fitted polynomials and a five frame history of a tracked tangent point. Figure 4b) shows an enlarged region of the tracked tangent point. In 4c) and 4d), new tangent points are added to the history. However the point in 4d) is not a road tangent point. A valid tangent point has to be in

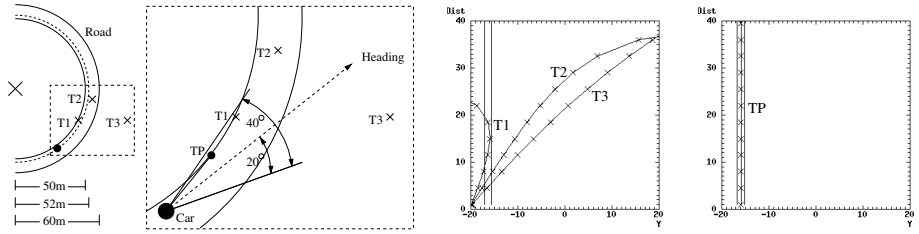


Figure 5: The Tangent Point and Noise

a neighbouring range of at least two other tangent points in the history. A "neighbouring range", in the context of this program, means that at least two tangent points from the history have to be within a close radius of the new tangent point. In 4c), three previous tangent points are within the radius of the new tangent point, allowing it to be considered as a road tangent point. In 4d), only one previously tracked tangent point is within the the radius of the new tangent point.

Figure 5 demonstrates the plausibility of this tracking mechanism:

The 180° curve segment on the left hand side of 5a) has a radius of 50m to the inner curb, 60m to the outer curb and a 52m radius by which a car is driving through the curve. While driving through the curve, four regions appear to contain potential road tangent points. T1, T2 and T3 are some environment feature, and TP is the road tangent point. The enlarged section on the right hand side of 5a) shows the location of the four features T1, T2, T3, TP, the heading of the car and the 40° horizontal field of view. As the car moves through the curve, the detected features move into the field of view. The diagrams in 5b) and 5c) show where the features appear in the driver's field of view. The horizontal axes show the driver's horizontal range of view from $-20°$ to $20°$. The vertical axes range from 0m to 40m and indicate how far the car has travelled. Every 3.5m (frame rate: frame/3.5m) the location of all four features are tracked and marked with a cross. Vertical bars indicate locations in which road tangent points are assumed at some point. For the purpose of illustration, the location in which T1, T2 and T3 first appear, is set to the travelled distance of 0m (bottom left hand corner). The TP is tracked anywhere within the range of the curve.

Figure 5b) shows how the location of the three features T1, T2 and T3 move along the horizontal axis, as the car drives on. A clustering of tangent points occurs when several crosses from one feature occupy a neighbouring range in the horizontal field of view. If more than two crosses are within the neighbouring range, a road tangent point is assumed. In frame 6, after 17.5m the program assumes T1 to be a road tangent point. At frame 7 and after 21m, T1 is out of the neighbouring range and is not interpreted as a tangent point anymore. By reducing the frame rate, such misinterpreted tangent points can be prevented. Features T2 and T3 are not misinterpreted as road tangent points, as the gap between each cross exceeds the neighbouring range.

Figure 5c) shows the location of the tracked road tangent point. It does not deviate along the horizontal range of view, allowing the tangent point to be tracked around the entire bend, regardless of the frame rate.

The discussion of the tracking mechanism is modelled on a purely idealistic world; allowances have to be made for deviations in the curvature and deviation from the drive path.

a) b) c)

d) e) f)

Figure 6: Tracking only Visible Tangent Points

3 Results

The results shown here were produced with the help of image sequences from the Vision and Autonomous Systems Centre Image Database at Carnegie Mellon University and image sequences from the image database at Istituto Elettrotecnico Nazionale, Italy.

The following selection of road sequences show tracked tangent points, produced by the algorithm outlined in section 2. The tracked tangent points are highlighted by white dots, surrounded with a black ring.

3.1 Tracking the Correct Point

In Figure 6 I can observe the algorithm's tracking performance, when a tangent point is suddenly obscured by an obstacle. The history of the last tangent points suggest a tangent point in 6c), but it is intentionally not approximated. Visual road curvature information is not available at the location of the tangent point in 6c). As soon as road curvature information becomes available again, the tangent point is tracked, as can be seen in 6d).

3.2 A Track History

In Figure 7 a history of tangent points is plotted on the images 7a) to 7i), this illustrates the migration of the tangent point over time. The vehicle is entering and following a lefthand bend from a straight road. Surrounding image regions, such as the lamp posts and the trees are moving past the vehicle at a steady rate. The tracked tangent point however appear to cluster in one region of the images. This observation can be made throughout the whole bend, and indeed this clustering is a main characteristic of tangent points.

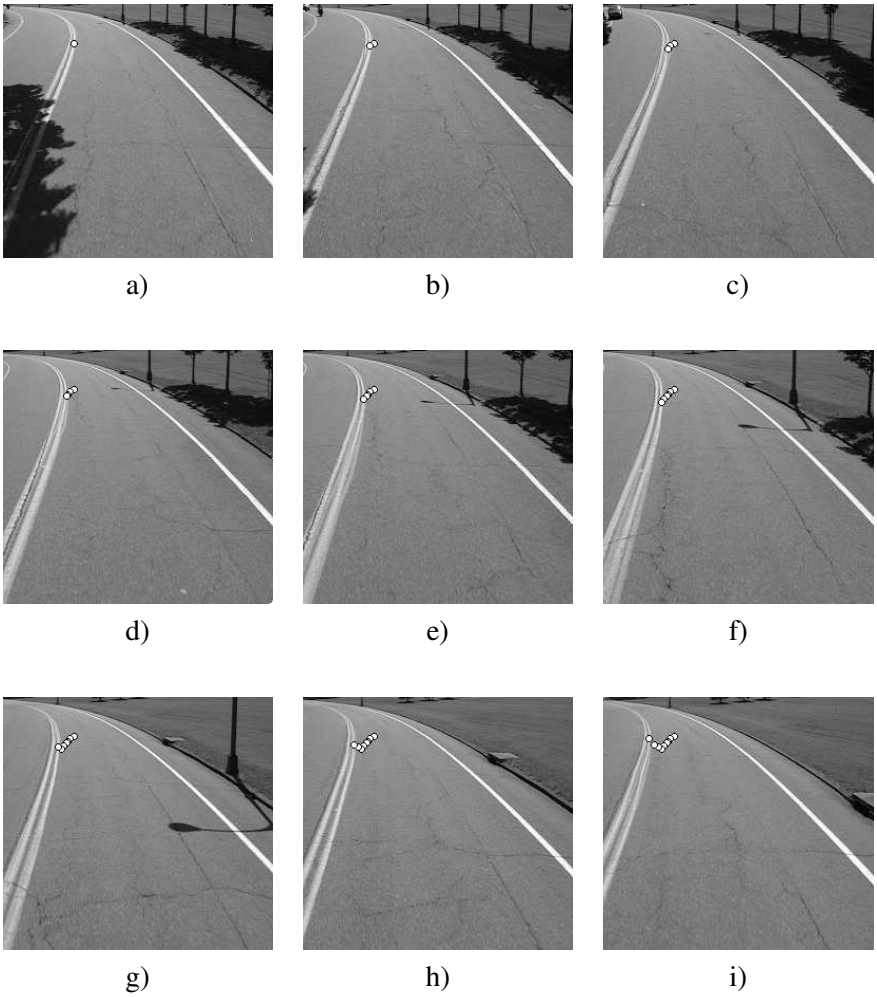


Figure 7: Clustering of Tangent Points

3.3 Several Tangent Points

The sequence in Figure 8 shows that the tracking algorithm is not restricted to a set number of tangent points. The road shows two tangent points which could be followed by the driver, the curve of the road and the actual road markings. Both tangent points are tracked accurately during the sequence. A further point to note, is that the images in Figure 8 have a higher frame rate and contain very much more scenery than the other sequences shown in this paper, the program is still able to track the tangent points accurately.

3.4 Added Camera Vibration

On roads with a good surface, as can be seen in the examples from subsections 3.1 to 3.3, the tracking algorithm performs fairly well. However not all roads have such a good surface. Roads with cracks and potholes would increase the horizontal camera vibration. The capability of tangent point tracking under increasing vibration is hence an issue which has to be addressed. The graph in Figure 9 demonstrates

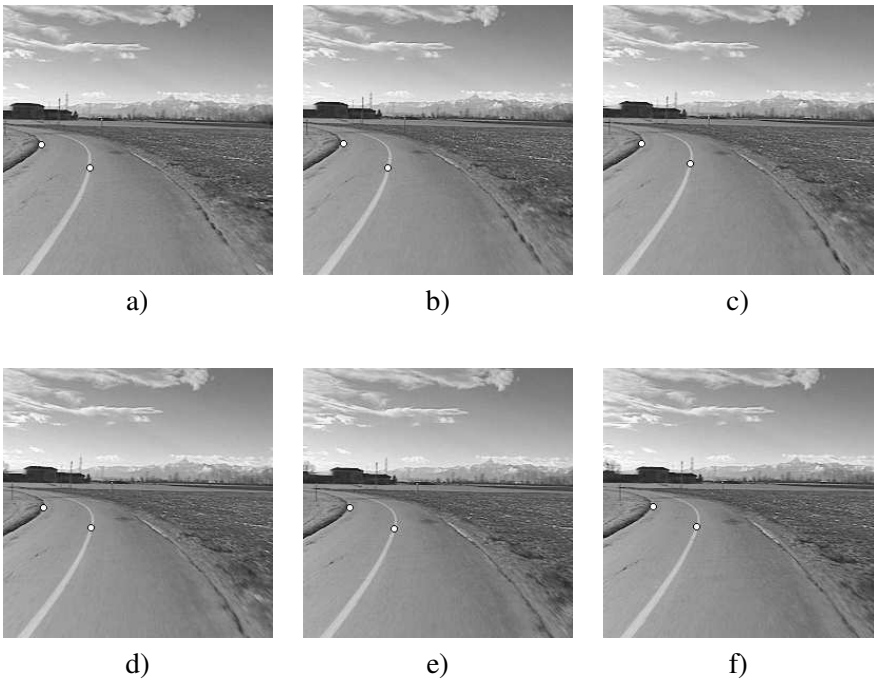


Figure 8: Several Potential Tangent Points

the performance of tangent point tracking ability, under increasing camera vibration[1].

The sequence consists of 158 continuous road images in which the program can detect 54 tangent points correctly and 1 tangent point incorrectly, with no added camera vibration. The graph's vertical axis shows the amount of detected image regions, and the horizontal axis shows the amount of randomly added horizontal camera vibration in units of pixel. The solid black dots indicate the amount of correctly identified tangent points and the black circles indicate the amount of incorrectly identified tangent points.

It can be seen that the amount of incorrectly identified tangent points stays consistently low throughout the whole trial period. The detection of correct tangent points does however degrade gracefully with an increase in pixel vibration.

4 Conclusion

The tangent point tracker, outlined in this paper, is designed specifically for the tracking of tangent points in road bends and is one of possibly many implementations. It was my objective to show that it is possible to track road tangent points with fairly simple and well established image processing techniques. The tracking algorithm has been shown to work on image sequences from differing sources and works without modifications. The results in Section 3 also show that the tangent point tracking algorithm does perform well on roads with a smooth surface. Based on the promising results from this study and the evidence that tangent points influence the human driving process, work will continue in this area. I will be looking at methods to improve the tangent point tracking performance under added camera vibration by investigating methods which do not refer to specific coordinates in the road image. Further more I will be looking at more biologically inspired methods of tangent point tracking and eye movement simulation of human beings. I expect that these future investigations will lead me to the development

[1] The camera vibration was randomly added to the sequence of images.

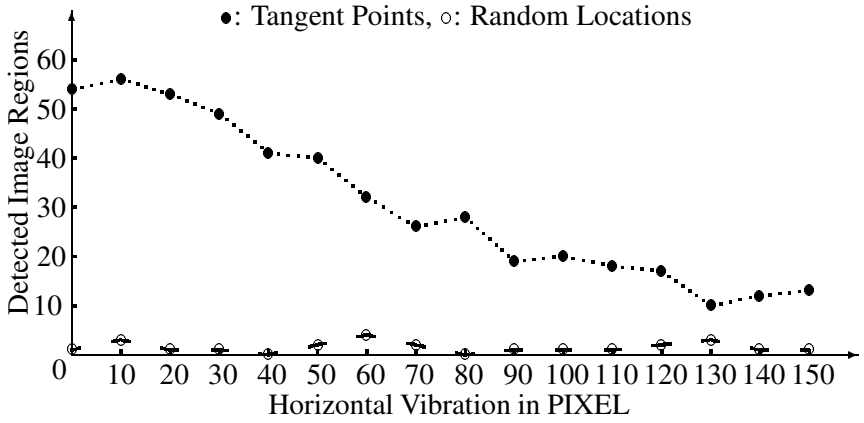


Figure 9: TP tracking performance under added horizontal camera vibration

of algorithms which may control active cameras. Further it is expected that the investigations will also provide a deeper understanding of the visually guided control which governs human driving.

5 Acknowledgements

I would like to thank the Vision and Autonomous Systems Centre at Carnegie Mellon University and the Istituto Elettrotecnico Nazionale, who provided me with image sequences for the system trials; Mike Land for his very helpful comments and finally my supervisors: Hilary Buxton and David Young who provided valuable suggestions and were of great help and support to this paper.

References

Donges, E. (1978). A two-level model of driver steering behaviour. *Human Factors*, *20(6)*, 691–707.

Fitzgibbon, A. W., & Fisher, R. B. (1995). A buyer's guide to conic fitting. Tech. rep., Department of Artificial Intelligence, University of Edinburgh, 5 Forrest Hill, Edinburgh, EH1 2QL, Scotland.

Godthelp, H. (1986). Vehicle control during curve driving. *Human Factors*, *28(2)*.

Jochem, T. M., Pomerleau, D. A., Kumar, B., & Armstrong, J. (1994). Pans: A portable navigation platform. Tech. rep., The Robotics Institute, Carnegie Mellon University, Pittsburgh PA 15213.

Jochem, T. M., Pomerleau, D. A., & Thorpe, C. E. (1993). Maniac: A next generation neurally based autonomous road follower. Tech. rep., The Robotics Institute, Carnegie Mellon University, Pittsburgh PA 15213.

Lancaster, P., & Salkauskas, K. (1986). *Curve and Surface Fitting An Introduction*. Academic Press.

Land, M. F., & Horwood, J. (1995). Which parts of the road guide steering?. *Nature*, *377*, 339–340.

Land, M. F., & Horwood, J. (1996). The relations between head and eye movements during driving. *VISION IN VEHICLES - V*, 153–150.

Land, M. F., & Lee, D. N. (1994). Where we look when we steer. *Nature*, *369*, 742–744.

Pomerleau, D. (1995). Ralph: Rapidly adapting lateral position handler. Tech. rep., The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA.

Reinhardt, F., & Soeder, H. (1990). *dtv-Atlas zur Mathematik Tafeln und Texte* (7 edition)., Vol. 2. Deutscher Taschenbuch Verlag.

Serafin, C. (1993). Preliminary examination of driver eye fixations on rural roads: Insight into safe driving. Tech. rep., University of Michigan Transport Research Institute. UMTRI-93-29.

In: Retkowsky, F. P. (Ed.). (1998)
The 11th White House Papers: Cognitive Science Research Paper 495
School of Cognitive & Computing Sciences, University of Sussex

Autopoiesis and Search

Oliver Sharpe
olivers@cogs.susx.ac.uk

School of Cognitive & Computing Sciences
University of Sussex
Brighton
BN1 9QH

1 Introduction

This paper is a position paper rather than a results based paper. It looks at the nature of search and in particular the paper answers the question: 'How should autopoiesis be modelled ?' by suggesting that a search algorithm will form the basis of the first kind of models of autopoiesis. The argument for this position is based around the observation that the kinds of dynamics that appear to be necessary for autopoiesis either require exponential growth of the number of particles in the system or require a search algorithm as an approximation of such growth.

It will be assumed that the reader already knows a fair bit about autopoiesis (Maturana & Varela, 1980), search algorithms (Davis, 1987; Goldberg, 1989) and dynamical systems (Kauffman, 1993). Autopoiesis and life are used interchangeably in the paper.

2 What is Search ?

2.1 Preliminaries to a definition

In this section I want to address the question of how can we recognise a search process by looking at the dynamics of a given system. In particular I want to argue that a search dynamic should be defined independently of the intentions of the user of the dynamic, and it should be independent of the amount of knowledge we have about the dynamic.

So to give an example, say we have an optimisation task such as a travelling salesman problem (TSP), where we want to find the best solution to a given problem, in the case of TSP the shortest route. We can define the search dynamic to be the likely 'movement' of the search algorithm through the search space of solutions. The expression of this dynamic will be in some formal notion that is independent of any real knowledge of the problem at hand let alone the reasons why the user is trying to find a solution to the problem. Hence, the formal description of the search dynamic will be independent of the intentions of the user. The formal expression of how to solve the TSP problem does not include the reason why the user wants to solve the problem. Now we can invent a totally abstract search space, one for which we have no real interest in finding the best solution as it doesn't represent any real problem that we wish to solve. Then, if the formal descriptions of the two search dynamics are the same, I think it is reasonable to say that the two are the same search process. It just so happens that in one case the search process is being used to solve a real problem but in the other case it is just being considered as an abstract dynamic.

Taking the same TSP example again, if we use brute force to evaluate every possible point in the search space (ignoring the time that this would take), then we would no longer be interested in using a search algorithm to find a good solution to the problem. The nature of the whole search space would be

known, including which solution is the best solution, and therefore if we are using the search space to find a solution to a real problem it is no longer useful to search the space as we already know which solution is the best. However, if we still ran the same search algorithm on this same problem, the dynamics of the search algorithm would be the same. It has no way of knowing that we already know the best solution. Therefore, if we want to make a definition of search processes that is based on the dynamics of the process it must be independent of how much knowledge we have about the system.

So I have argued that we cannot define search as being the process of 'finding a solution to an unknown problem'. This kind of definition relies too much on the intentions and ignorance of the user and is not based on the dynamics of the system. I want to find a definition of search that is based purely on the dynamics, and this way prevent the absurd situation where a process can at one time be a search process but at another time not be a search process - simply because of a change in the intentions or knowledge of the user.

Another issue that I want to be aware of when arriving at a definition is the fact that search algorithms and search spaces do not have dynamics by themselves. It is only the combination of a particular search algorithm with a particular search space that will give rise to a particular search dynamic. Therefore when I refer to a particular search process I mean the dynamics of a particular search algorithm on a particular search space. Hence two search processes may have the same search dynamic even if they are constructed out of different search spaces and/or search algorithms. Here the search space includes the representation, the fitness function and the details of the basic mutation and cross-over operators.

So I am interested in defining which kinds of processes are search processes and which are not. In particular, given a set of data from the dynamics of a process, I would like a definition of search that gives us a way of determining whether the data comes from a search or a non-search process. First let's look at what kinds of data structures search processes generate.

2.2 The Generic Framework for Search Algorithms

Before trying to look at the dynamics of systems that *might* be search, let's look at the types of dynamic systems that we know are search. From this we can at least decide which types of system we will even consider as being possible candidates. To do this, let us look at a generic framework for search algorithms, from which we can define the structure of the generic trace of a search process. The framework is based around evaluation space.

Evaluation space is the space in which all search algorithms operate. Every search space has an associated evaluation space. For a given search space, each point in its associated evaluation space represents a *set* of points in the search space. So the evaluation space is the set of all subsets of the search space, the power set. There are points in evaluation space that represent each of the singleton sets, representing when just one point has been evaluated so far. There's the null set of course, when no evaluations have occurred and all the way through to the single point that represents the set of all points in the search space, the time when all points in the search space have been evaluated. All search algorithms can be thought of as methods of moving about evaluation space.

The size of a point in evaluation space is the size of the set that it represents. The size measure is thus a measure of 'work' as evaluations are considered to be the unit of work in the search process. To reach a set with n members you have to have done n evaluations which will have taken n units of work. As all search algorithms can only do one unit of work at a time, when we look at the history of any algorithm's movement in evaluation space it can only move one step at a time, from a point of size n to a point of size $n+1$.

This movement of an algorithm through evaluation space is called its *history trace*. The history trace captures the order in which the search algorithm evaluated the points that it visited during its search. So

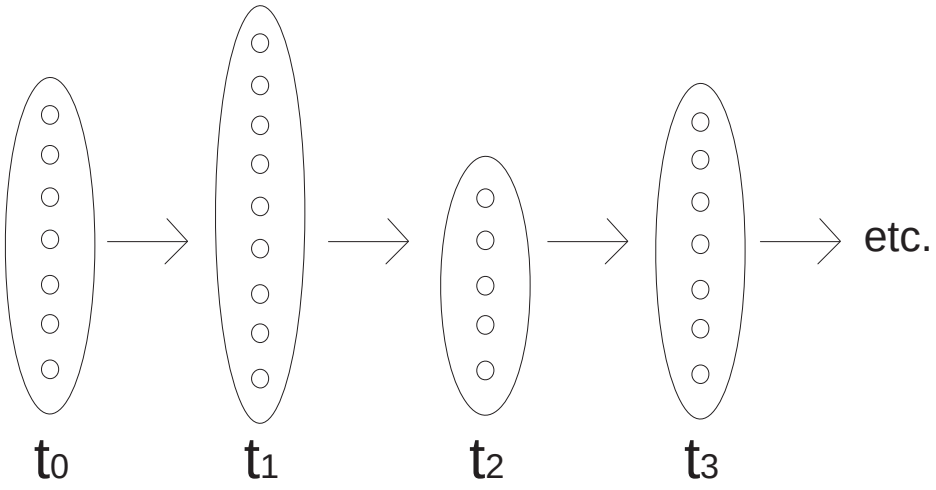


Figure 1: The structure of an active set trace.

the structure of a history trace is a list, or time series, of objects. The history trace also shows us the order in which the search process performed work, as each evaluation is considered to be a unit of work. Hence the history trace shows us the work dynamics of a search process.

A significant aspect of the search process that the history trace ignores is the set of evaluated points from which the next candidate solution will be generated. Most search algorithms use hints gained from previously evaluated points to help direct the future search direction. However, most do not remember the whole of the history trace. Instead algorithms keep a subset of the history trace as a set from which to generate the new candidate solutions. In a GA this subset of the history trace is called the population. In the general case, I will call this set the *active set*. A search process as well as generating a history trace also generates an active set trace.

Within this framework all search algorithms are constructed out of two biases, the first is the way that the algorithm constructs its active set from the history trace, and the second is how it generates new candidate solutions from the active set. As all search algorithms can be expressed in this framework, hence the dynamics generated by this framework, in particular the active set trace, will form the basic data structure of all possible search processes. The data structure consists of a 'time' series of sets, possibly of varying size. In looking for a definition of search, I am looking for a way of judging a given time series of sets as either being or not being generated by a search process.

We can now compare two search processes and say whether or not they are the same basic search process or not. First we want the representations (or objects over which the search is occurring) to be uniquely indexable and then we need to have a one-to-one mapping between the indexes of the two different search process. With this mapping we want to see if the dynamics of the two search processes are isomorphic or not. As most search processes are probabilistic in their nature, and each data point in the time series can be considered as a state, such processes can be modelled as Markov chains (Syski, 1992). So with the mappings between two search processes and the Markovian models of the processes we can now make definitions of two levels of equality relationships between two search processes.

- Two search processes are said to be identical if they have isomorphic active set dynamics. (ie they have the same Markovian model of the active set trace)
- Two search processes are said to be essentially the same if they have isomorphic work dynamics (ie they have the same Markovian model of the history trace)

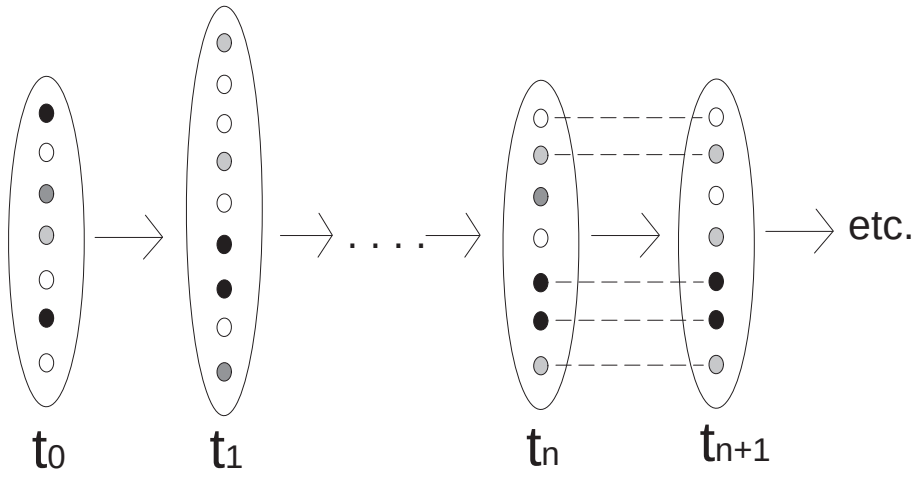


Figure 2: The degree of persistence increases within an active set trace of a search process

We are also now in the position of having the generic data structure of all search processes. Any dynamical system that does not generate such a structure will not be considered to be a search process. Our definition of search will only apply to active set traces. Let's look at this definition.

2.3 A Definition of Search

So then, what is and what is not search. There is of course a danger of ending up with such a wide definition that it is unuseful. The real question that I want to be able to answer is: given a supposed active set trace how can one judge whether or not it was generated by a search process ?

This is where the notion of *persistence* must come into the picture. With an active set trace from a search process some persistence measure should surely be increasing with time. Fitter individuals will last longer in the trace than un-fit individuals, similarly fitter settings of the loci in the genotype will persist for longer in the trace than unfit settings. If the search process is a hill-climber on a Mt.Fuji optimisation problem then, in the end, the active set trace should settle down to containing only the optimum solutions of the problem.

Although the increase in persistence may not be a monotonic one, it should be a fairly noticeable trend. In the case of optimisation, say a hill-climber on a Mt.Fuji landscape, then this trend should continue until the search process has reached the optimum[1]. At the end of an optimisation search process in general, the active set should remain relatively similar - at worst moving around a neutral network of equally good points in the space - it should remain in the maximally persistent state or group of states. Hence persistence (arguably 'order') should generally increase over time within the active set trace of a search process.

So my personal definition of search is the following :

Search is the accumulation of persistence.

Under this definition it is clear that many dynamical systems are considered to be search processes. Most systems with a clear set of attractors will fall into this definition. But I am quite happy with that.

[1] Arguably the dynamics of a hill-climber fixed at the top of a Mt.Fuji is not the dynamics of a search process any more - it's *too* static

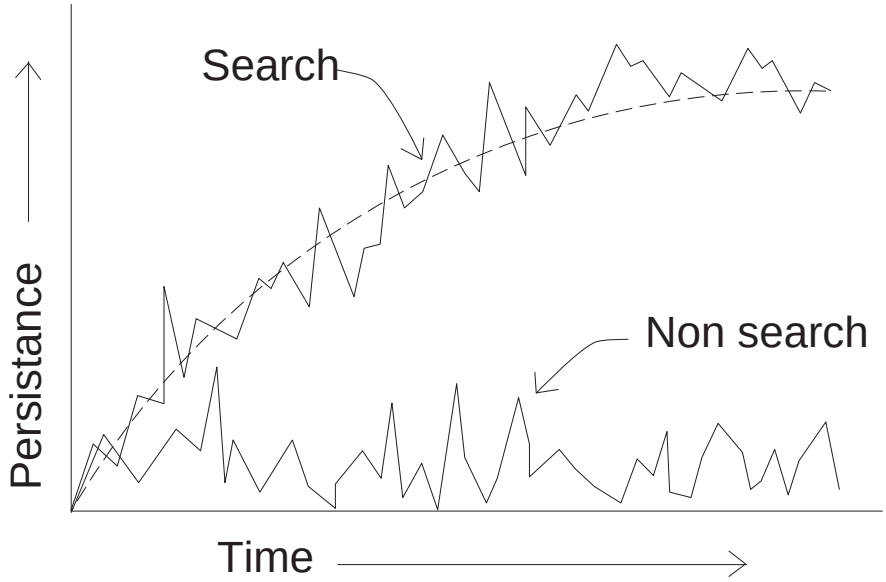


Figure 3: Distinguishing between search and non-search processes

Most important is that a random process will not fall into this definition, and neither will the process of a hill-climber on a needle in a haystack landscape (or indeed flat landscape) and I am very happy with that too. The dynamics of a hill-climber on a needle in a haystack landscape is what one might call a termination dynamic where the active set trace is an apparently random sequence of objects with no accumulation of persistence terminating abruptly with a totally persistent object - the needle. This type of trace is one that many many Turing machines will produce: an apparently random sequence of states followed by a terminating state - ie a state that repeats for ever or that signals the halting of the machine. I am quite happy for these behaviours, these dynamics, to not be labeled as search. In that way the definition of search is not too broad as to include all computations.

So if we plot a graph of the amount of the state that is exhibiting persistence either static or dynamic then we should be able to spot a search process from a non-search process. A closeup of the graph of a search process at the end will look like a non-search process, except for the fact that it will have a high average amount of active persistence. By active I mean that the persistence particles are still being acted upon, they are just being selected to remain the same. It is the choice that makes them persist not an absence of attention - and it is this choice of certain things over others that characterises search and will lead to a general trend of an increase in persistence.

One important thing to note here is that a hill-climber climbing up a Mt.Fuji may at first sight look like it does not fit into this definition. As the hill-climber moves up the hill it is constantly changing its state until it reaches the top where it terminates - hence on first thought it looks as if this is one of those non-search terminating dynamics. However, if we consider the sub-states of the state of the climber then hill-climbing is precisely the accumulation of good mutations, for example good settings of the bits in a bit string, so that over time more and more of the bits will not vary much. More and more of the bits in the string will be set at their 'correct' value and therefore we can define a persistence metric on the sub-states of the climber and see that a hill-climber on a Mt.Fuji landscape does indeed accumulate persistence within its sub-states.

Another point worth mentioning is that a measure of persistence will only be dependant on the state of the system over time and will have no connection with the intentions or knowledge of the user - hence

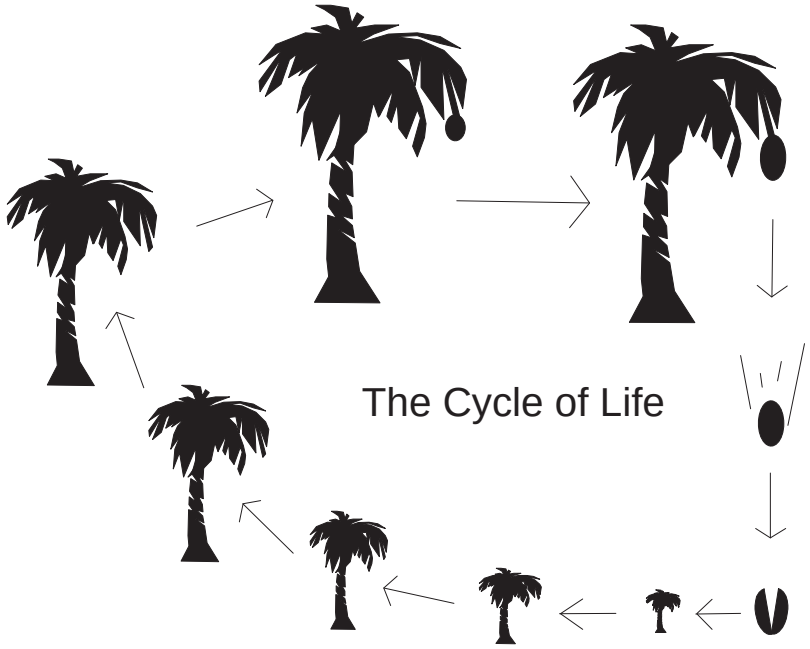


Figure 4: The cycle of life as a cyclically persisting relational object

this is the kind of definition of search that I have been looking for. It does not matter whether the system is optimising an explicit fitness function or running with an asynchronous birth and death process; if it is search there should be an increase in persistence.

What also becomes apparent here is that in order to classify a dynamical process as being or not being search one needs to have a measure of persistence over the states that are being searched. So what we need to do now is to look more closely at what I mean by persistence.

3 Persistent Objects

What I am slowly getting towards is the idea that life is dynamically persistent in the sense that mouse objects maintain the existence of very similar mouse objects for millions of years in a dynamic way whereas a rock may persist for millions of years but only in a static way. Over time the rock will slowly disintegrate whereas the mouse objects will proliferate. Life appears to increase the amount of persistence. But before looking deeper at the dynamics of life, we need to develop further the notion of persistence.

3.1 Formal definitions

Before getting into the definitions, I want to be clear about some terminology that will be used in these definitions. We are looking at active set traces, trying to determine the degree of persistence in these traces over time. Each active set represents the *state* of the search process at a given time. Each sub-state of a given active set I will refer to as a *state holding particle*, so each particle is capable of holding its own state. The dynamics of a search process is then made out of a time series of states where each state consists of a set of state holding particles.

To give some examples of this type of dynamical system we may have a system where each state of the system consists of a collection of atoms (the particles) each with their own internal state. Similarly

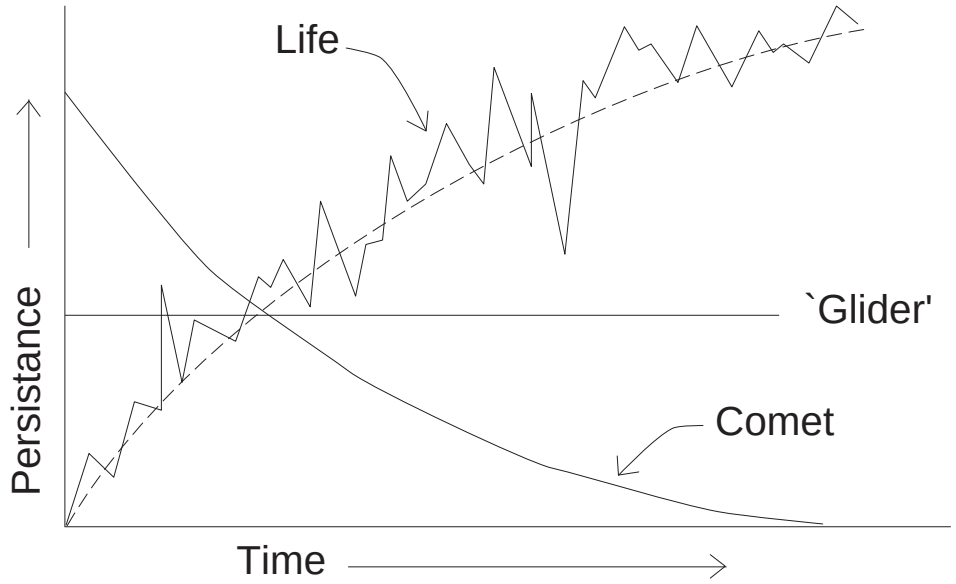


Figure 5: The different styles of persistence of a comet, a ‘Glider’ and life

the states of the system may be cells with their own internal state. There are many such dynamical systems used as models of interesting phenomena.

So what is a persistent object ? Well the simplest form of persistence would be a static object, such as a rock, which persists in its form for a long period of time before being destroyed. If we take, say a comet in space, and measure say how many particles of the comet are remaining in the same relationship with each other over time then we will see a gradual decrease in the number of particles of the system that are remaining in this relationship. The level of persistence will slowly be decreasing. Another type of persistence is that of the glider structure in the Game of Life cellular automaton which persists for ever if undisturbed. The most interesting form of persistence, as mentioned earlier, is life. Objects that are alive not only increase the number of them that are persisting through reproduction but also increase their ability at persisting through evolution. Arguably the human objects are the objects that are most talented at persisting that have ever existed in this neck of the universe. I guess only time will tell if this is true or not.

How well defined is this notion of persistence ? Let’s look at tree objects. If we start with the seed of a tree, we could quite easy define spatial relationship between atoms that must hold true if we wish to call an object a seed. Similarly we could construct a spatial relationship that must hold true for an object if we want to call it a fully grown tree. The most difficult thing is to provide a definition of tree that is able to tell us that a seed object remains the same object as it grows into a tree. This type of definition is precisely what we need if we want to be able to measure the persistence of tree objects. To achieve this we can formally define a type of object called a cyclically persistent relational object (CPRO). The key to this type of definition is the idea of linking the object’s various spatial relationships together using temporal relationships. Thus the object is tied together into a cyclic chain of relationships. The temporal relationships are constructed out of particles that do not change their relationship between two of the spatial relationships.

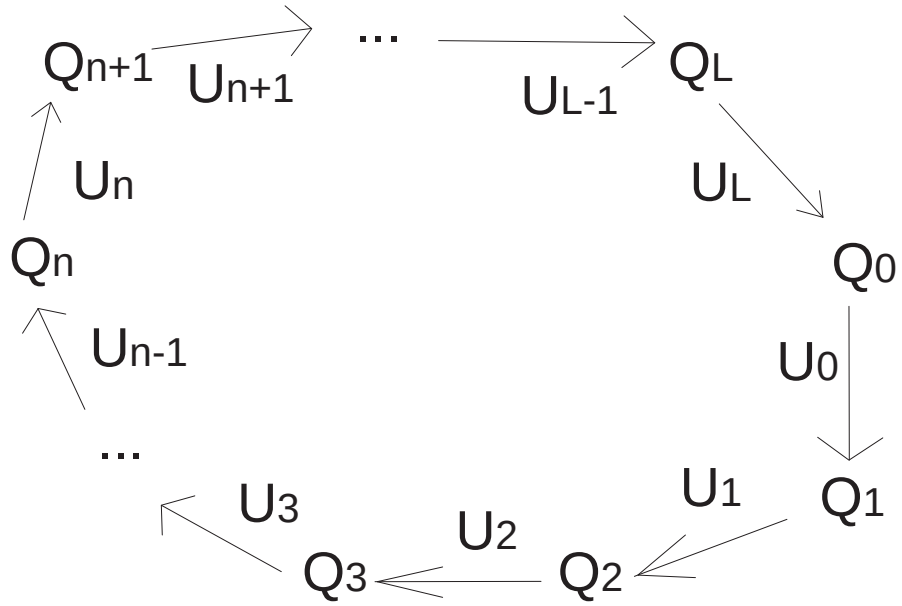


Figure 6: The formal structure of a Cyclically Persisting Relational Object (CPRO)

The model is as follows:

A CPRO which cycles after L time steps is constructed out of two sets of relationships:

- The spatial relationships Q_0 to Q_L
- The temporal relationships U_0 to U_L

such that the following conditions hold true:

- All the Q and U relationships are boolean relationships between the states of particles that either holds true or false for a given set of particles.
- $\forall i (U_i \subset Q_i)$ and $(U_i \subset Q_{i+1})$
- $\forall i$ there must exist at least one particle p_i such that $p \in U_i$ and $p \in U_{i+1}$

This last condition ensures that the CPRO chain is unbroken. This is a necessary condition to ensure that factories making kettles do not become part of a CPRO that turns static kettle objects into CPRO kettle objects 'through' the factory. It does not mean to say that over many cycles of the CPRO persisting the particles satisfying this condition have to be the same ones, only that there must exist at least one particle that can fulfill this condition on each step of the cycle.

One important point to note about this definition is that it does not stick the object to a particular group of particles. The Q and U relationships can continue to hold true over a changing set of particles, and as long as the stated conditions are maintained we can safely say that the object remains the same CPRO over time even though it is constituted from different particles. This is a very useful property.

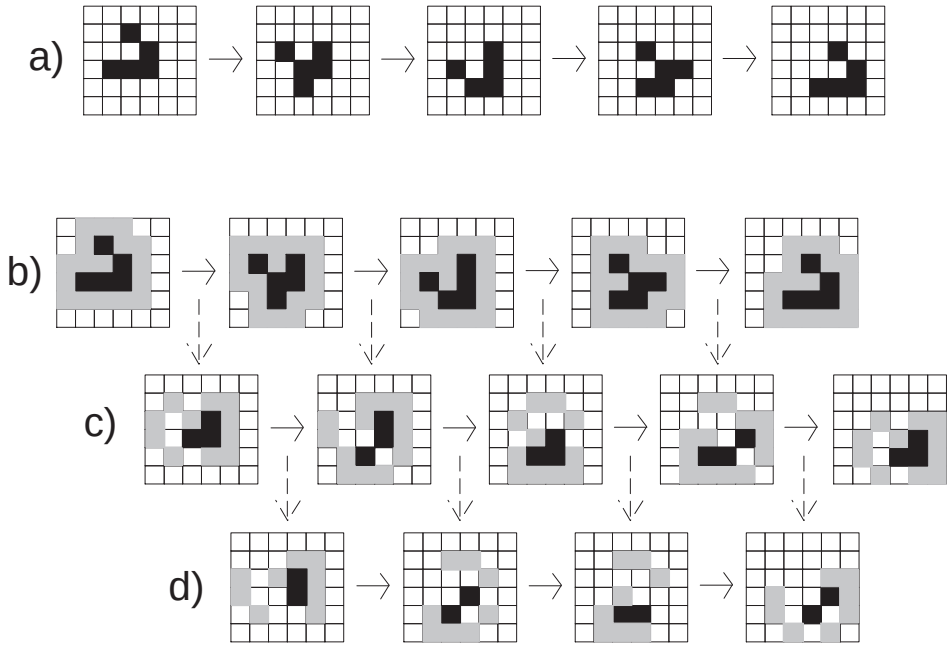


Figure 7: A glider from the Game of Life CA as an example of a cyclically persistence relational object (CPRO)

3.2 Gliders as an example

Let's look at an example CPRO to make things a little clearer. Gliders are a type of object that occurs within the dynamical system of the Game of Life cellular automaton. Figure 7a) shows the cyclic progression of the glider over time. Figure 7b) shows the various Q relations. Figure 7c) shows the temporal U relations of particles that remain the same between two Q relations. Finally figure 7d) shows the potential candidates for the p particle at each step of the cycle. So these are the diagrammatic versions of the spatial and temporal relationships that hold true in the case of a glider, showing that we can be confident that a glider is indeed a single CPRO object that moves over the particles that 'hold' it.

Now that we have defined a notion of persistence the next task is to see whether or not we can measure this persistence. If we cannot then it is a useless notion. If we can measure persistence then we can use it to draw a graph of persistence versus time for a given active set trace, and thereby see if it has the characteristics of a search process: does persistence generally increase with time ?

3.3 Measuring Static Persistence

So let's first get a look at a persistence measure that would be ignorant of cyclic persistence of any kind. So we are looking to measure the amount of static persistence within a trace. So this is surely the degree to which the state of the particles within the whole state remain fixed. We are not yet even talking about relationships between particles. Hence the degree of persistence is linearly dependent on the degree of persistence of each individual particle's state. So, how can we measure the persistence of a particle's state, especially as a function of time. As fair a method as any would be to take the percentage of the whole time for which the particle has been in its current state. So this is a measure that is knowledgeable of the whole trace to date rather than the trace up to the point being calculated. So for example:

1.0	1.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0
2.0	2.0	1.0	2.0	2.0	5.0	5.0	5.0	5.0	5.0
0.2	0.2	0.1	0.2	0.2	0.5	0.5	0.5	0.5	0.5

So the first column shows the state of a particular particle over time, its trace. The second shows the length of the persistent trace that each state of the particle belongs to and the third shows the measure of static persistence over time.

We can define the algorithm for measuring the static persistence of a trace as follows:

- Give the algorithm a trace, T, of length L time steps with dimension N. (by dimension I mean that there are N particles within the state)
- Associate with each particle the % time for which its current state persisted. (this will be order 2L for each particle so O(N*L) in total)
- Add all of the persistences of each particle at each time point to get the static persistence of the state at each time point: (this will be order N for each time point L so again O(N*L) in total)

So measuring static persistence of an N particle system over L time steps is O(N*L). This demonstrates that the concept of static persistence at least is well defined in the sense that we can actually measure it and therefore use it.

3.4 Measuring Cyclic Persistence

Although the measure of static persistence gives us useful insights into the nature of search processes, it by no means gives us a complete picture. The measure of static persistence will probably only allow optimisation search processes to show up clearly. The measure cannot 'see' the patterns of cyclic persistence and therefore cannot pick them up. Ultimately the most interesting search processes are those that give rise to cyclic persistence, the type of persistence that is capable of supporting autopoiesis. So we must look beyond static measures and try to develop a measure of cyclic persistence within traces. Only with such a measure could we start to probe the nature of search processes that give rise to autopoiesis.

In section 3.1 we introduced a formal definition of a cyclically persisting relational object (CPRO). Now we need to produce a measure that tells us how much of a trace is constructed out of such objects and therefore how much of the state is persisting in the form of such objects. It is only if we can produce such an algorithm that the concept of CPROs can be of any use to us. The proposed method is to first of all develop an algorithm that can test if a given particle is a part of a CPRO or not, and thereby find the CPROs in the trace. Once we've found them we can count them and count how many of the particles are involved with them, thereby calculating the degree of persistence within the system. So to show that it is possible to calculate the degree of cyclical persistence we only have to show that we can algorithmically determine whether or not a given particle is a member of a CPRO. We don't care at this stage about efficiency, we just want to check that it is algorithmically possible, and thereby that the notion of CPROs is useful.

3.4.1 An algorithm for finding CPROs

So here is a description of an algorithm that can answer the question: is this particle part of a CPRO or not ? If the answer is yes then the algorithm can return the CPRO as well. The only constraint on the existence of this algorithm for a given system (and it is quite a serious constraint) is that there must be a function *G* that takes a particle and returns a spatial relationship of particles that could be in the same object as the particle in a given time slice. The function *G* does not have to be perfect as its guesses can

be refined later, however it should if anything always slightly overestimate rather than underestimate the size of the object that a given particle is related to. For the moment let's assume that such a function is possible within the system that we are examining. Then the algorithm goes like this:

- We are given a particle p.
- From p we use G to generate the first spatial relation Q_0
- From Q_0 at time t_0 we make U_0 as the maximum part of the relationship Q_0 that still holds true at t_1.
- From U_0 we can now use the function G to generate Q_1 from the particles of U_0.
- We repeat this process of using Q_i to generate U_i and then using G to generate Q_{i+1} from U_i. On every generation of U_i we check to see that there exists at least one particle p such that $p \in U_i$ and $p \in U_{i-1}$ for the process to continue iterating. This ensures that the chain being created is an unbroken chain and hence is definitely from one object. Note that this iteration that generates the chain of the cycle will also be broken if no U_i can be generated at all.
- We continue this process until we reach a Q_L such that $Q_0 \subset Q_L$, the chain is broken or until we run out of time looking.

Assuming that we were successful in finding a Q_L such that $Q_0 \subset Q_L$ the we have our first approximation of the Q and U relationships that define the CPRO. This first approximation can be refined by following the cycle through the trace a number of times, trimming each of the Q and U relationships of each step of the cycle to ensure that they are the same on each occasion. The hope is that this trimming process will soon converge on a well defined CPRO. In the case of the Glider it is clear that G would be easy to define and that the CPRO relationships for a Glider would be easy to find using this kind of algorithm.

However it is also worth noting that in more complex, more interesting cases the cyclically persisting relational objects may be able to persist through various paths of transformations, hence finding them would be harder. One thing that I am sure must remain for the object to be considered one object that is persisting in a CPRO like fashion is that even if there are many paths down which the state of the object can go, there must be some equivalent set of spatial and temporal relationships to those I've defined for CPROs.

One other issue that I have not really delved into yet, but I must do at some time, is that of possibly having to add to the CPRO structure a set of enabling conditions for each step. This is to ensure that the description of the CPRO includes the knowledge of what kind of environmental disturbances will destroy the CPRO, and I guess more importantly which will not. This leads to the idea of Adaptive Pros that can withstand many types of disturbances and still persist. Such ACPROs would probably be multi-pathed as described above.

To find the enabling set and the extra paths of an ACPRO would require following the paths of a CPRO discovered through the algorithm above and seeing under what conditions it is destroyed and seeing which conditions make it follow another path of persistence. First its status as a CPRO must be established through one loop that gets back to where it started and at least starts to repeat the same cycle, then these extra questions can be asked about the loop. I am working on the assumption that for an object to be an ACPRO it must somewhere behave like a CPRO for at least one loop of the cycle. Hence it should be possible to find them using the above algorithm.

The algorithm can only tell if a particle is part of a CPRO that is still 'healthy'. An example of this is that it is very difficult to see that a particle from a leaf on the ground was once part of a CPRO, as now

it appears to be a slowly deteriorating static object. Having made this assessment of it it would be wrong to classify all other leaves in the system as being deteriorating static objects. So, what this algorithm cannot say is whether or not the particle *has* been part of a CPRO that is now disintegrating. However, it should always be able to answer whether or not the particle is part of a CPRO that is still 'healthy' as it were. So it can give a decisive yes/no as to the current membership of a CPRO that still has at least one cycle to go.

4 The Dynamics of Autopoiesis

This is really where all of my thinking is heading towards. Trying to understand in which types of dynamic systems it is likely that autopoiesis can be investigated. This 'virus of matter' that is life seams to be slowly increasing and increasing the number of atoms of this planet that are involved with the process of life. It has now got to the stage that with man slowly learning how to leave the planet it won't be long before life starts recruiting even more atoms into its dynamic persistence. So life is not the increase in general persistence, but rather the increase in dynamic persistence within the universe. So to study life we need to be looking at dynamical systems in which the amount of dynamical persistence is not only free to increase, but is likely to increase over time. Let's first look at which types of deterministic systems could be capable of supporting such increasing levels of persistence. At the very least that means being able to support cyclically persistent relational objects (CPROs).

So let's have a look at the stability of CPROs within deterministic systems. One thing I want to note is that a system capable of supporting autopoiesis should not be very sensitive to the starting state of the system. In the next couple of sections I make the argument that to get increasing levels of persistence you need to either have the potential for exponential growth in the number of particles involved with persistence or use search as an approximation to this.

4.1 Markov Chain models

Given a CPRO, when some environmental effect on the CPRO occurs, let's call it a mutation to the CPRO, then this alteration can either make the CPRO more stable, equally stable or less stable. Let's say that 51% of the mutations are deleterious and 49% are advantageous, then a simple random walk Markov chain model (Syski, 1992) can predict the expected extinction time for such a CPRO. Once the CPRO has reached the point of having no persistence then it will disintegrate and will not exist any more. The Markov chain model predicts that given a set of such CPROs, there is an expected finite time after which all of them will have disintegrated assuming that the chances of mutations occurring to them does not change. As the majority of deterministic systems will have a much larger percent of mutations being deleterious to CPROs this shows the precarious position of CPROs in purely deterministic systems. Irrespective of how the CPROs come in to being, they are inherently unstable on average in the long run.

So in such a deterministic system, to get a CPRO to be stable in the long run requires considerable thought on the part of the designer of the system. In the Game of Life, CPROs are generally stable so long as they are not interfered with. Usually when they are interfered with they either break up into smaller CPROs or they completely disintegrate. To ensure that a deterministic system has an increase in persistence requires that the system be set up in such a way that given the particular starting state the CPROs that occur are likely to have mutations to them that will result in better persistence. This is a tall order and it suggests a set of rules for the system that are designed with the different levels of persistence in mind. I cannot help but imagine that in such systems persistence will increase, but only up to the point that the designer has built into the system. To get ever more complex forms of persistence will require the designer to put ever more complex rules into the system to ensure that that the likely mutations to CPROs will on average increase and not decrease their persistence.

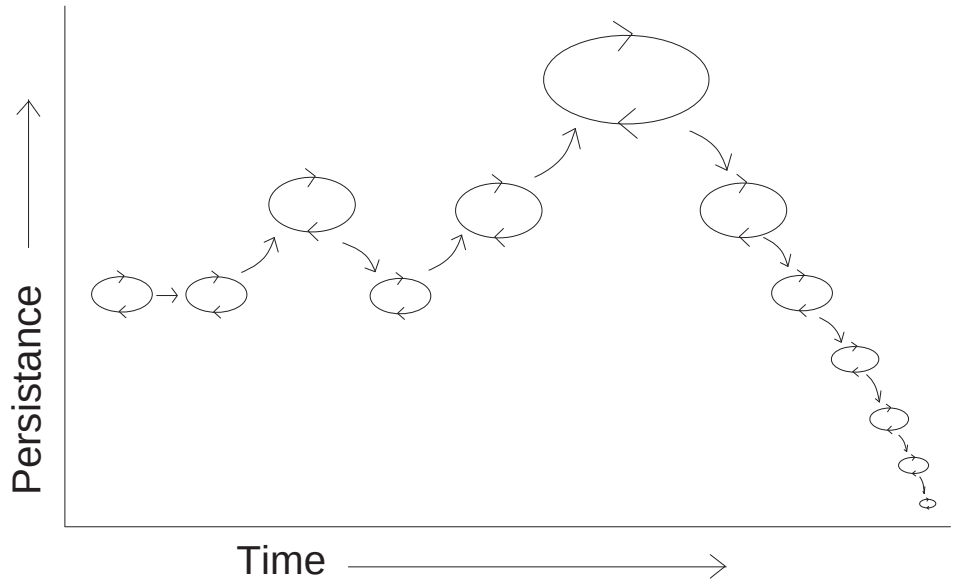


Figure 8: A Markov Chain model of persistence as a random walk will predict a finite time before extinction if the deleterious mutation rate is greater than the advantageous mutation rate

4.2 Replication as a Persistence Ratchet

What is really needed to enable a general increase in persistence to occur is some equivalent of replication. Replication is at the heart of ensuring that persistence at least does not decrease, and therefore has a chance that it will occasionally increase. Replication acts like a ratchet on persistence. Once we have replicating cyclically persistent relational objects (RCPROs) then we are in business. The only other thing that we need is to have the rate of replication being larger than the rate of mutation. So now we are in the position where given a simple RCPRO, its numbers will grow exponentially. As we know that the rate of mutation is less than the rate of replication we can guarantee that even if all of the mutations that occur are deleterious this type of RCPRO will still persist. Hence if nothing else, the level of persistence in the system will not go down. Replication has given us a ratchet on persistence. Now if an advantageous mutation occurs to the RCPRO, however unlikely, once it occurs it too can take hold in such a way. The only requirement for the guarantee is that replication occurs faster than deleterious mutations.

With this persistence ratchet in place, persistence in the system will generally increase with time. So any system that is capable of supporting such a persistence ratchet and therefore capable of supporting autopoiesis, will be exhibiting the dynamics of a search process.

A weaker form of the persistence ratchet guarantee is that the number of deleterious mutations that are likely to occur is less than the number of replications that are likely to occur in a given time. If most of the mutations that are likely to occur will be neutral then it is relatively easy to have this weaker guarantee holding true. So with RCPROs it is possible to have ever increasing levels of persistence even if the chance of a given RCPRO mutating into a more persistence RCPRO is small. The only cost is that the dynamical system must potentially support exponential growth in the number of instances of RCPROs. Long periods with mainly neutral mutations will result in exponential growth in the numbers of RCPROs. Although finite resources in the system could potentially give rise to a form of co-evolution between the RCPROs, there must be sufficient room in the system to ensure that the replication ratchet is definitely in place. As the number of deleterious mutations in most systems is likely to be large compared

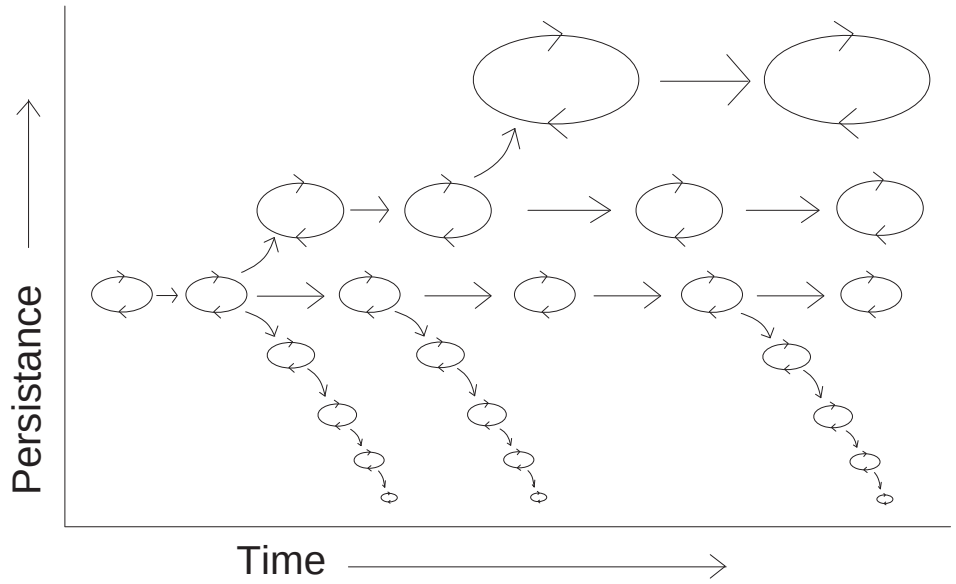


Figure 9: Replication is at the heart of a *persistence ratchet*. This useful dynamic comes at the cost of exponential growth in the number of objects

to even the neutral mutations, therefore the growth rate must be similarly large. So, to get autopoietic systems to emerge in a dynamical system not only must the system support RCPROs but also it must be able to support them growing at a rate that is faster than the rate at which deleterious mutations occur. The danger here is that this means that the number of particles in the system will be growing, potentially in an exponential fashion. Hence, even if we are generous with the algorithm complexity and say that it is O(n) where n is the number of particles, then as the number of particles increases over time we will have a decrease in the performance of the algorithm, potentially it will slow down at an exponential rate. This is not good news. It renders this kind of model of autopoiesis effectively intractable.

If we want to study autopoiesis we have to find tools with which the study is a tractable proposition. In the next section I make the argument that traditional generate and select search *algorithms* can be used as an approximate model of these search *processes* that occurs in dynamical systems that can support persistence ratchets. They make the modelling tractable by removing the necessity of exponential growth to maintain the persistence ratchet.

4.3 Search as an approximate model of Exponential Growth

The only way out of this exponential slow down is to somehow make an approximation of such a system by trimming the exponential growth to reduce the number of calculations that are necessary, thereby enabling the simulation to be feasible. The most important thing that must be preserved in this approximation is the persistence ratchet as this is the key to why the system was interesting in the first place. The best way to achieve both of these goals is to remove from the system RCPROs that have just had deleterious mutations, and thereby remove all of the calculations required in modelling their decreased persistence. In this way all of the attention of the model can focus on RCPROs that maintain the same level of persistence or higher levels of persistence, hence preserving the persistence ratchet. This requires some measure that can be applied to RCPROs to see which are the ones that have a lower than average ability at persisting - a lower fitness. Those with lower fitness are removed from the simulation

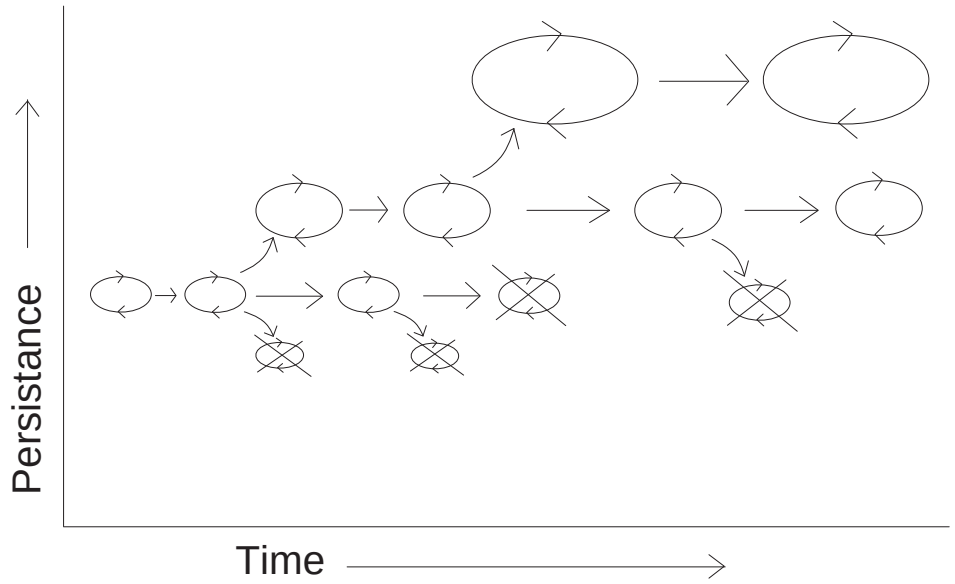


Figure 10: Using a search *algorithm* can enable a tractable approximate model of the search *processes* that occurs in dynamical systems that can support persistence ratchets.

as they are unlikely to lead to higher levels of persistence and are therefore considered to be a waste of CPU cycles. Hence we have an approximation of a system with exponential growth that works by concentrating the attention of the simulation on only those RCPROs with higher than average fitness - a search algorithm.

As with any approximate method, such a search processes cannot capture all of the subtleties of the system that it models. The quality of the approximation will depend heavily on the quality of the fitness function that assesses the ability of RCPROs at persisting. If this fitness function is very accurate then the approximation can be effective. This comes from the fact that selecting and modelling only the most stable RCPROs should be equivalent in the long run to generating all possibilities but then only the most stable surviving, as either way in the end you are left with only those that were most stable. A search algorithm, through actual selection, simply speeds up the process of removing unstable RCPROs from the system, thereby removing ultimately unfruitful work from the simulation.

So this is my argument for why a search algorithm is likely to be the first type of engine for a dynamical system used to model autopoiesis. This is not because I think it would be the best way to model autopoiesis, nor is it because autopoiesis will require a search process and therefore why not use a search algorithm. The argument is simply saying that for the time being, a search algorithm will be the only way of modelling a search process (persistence ratchet) that can support autopoiesis in a tractable way. The alternative would require a potentially exponential growth in the number of particles involved and therefore a potentially exponential slow down in the speed of the simulation. This is unacceptable and the use of a search algorithm can offer an approximation of the persistence ratchet necessary for autopoiesis.

4.4 Finite resources - natural search

One issue that has to be looked at is the effect of finite resources. Whilst exponential growth of life on planet earth is limited by finite resources, early life was so small compared to the size of the planet

that it was 'playing' in an approximation of infinite resources. Once life had filled the space available, it will have been growing its ability at persisting and increasing number of different types of RCPROs persisting in the soup. In this new situation of limited resources the persistence ratchet is maintained in nature by survival of the fittest - the natural search algorithm of evolution. Those individuals that were less capable of persisting are the ones that are most likely to be eaten first. The persistence ratchet is maintained, although it is altered in its character as now there is competition between the RCPROs for the limited resources, and hence a selection pressure that will 'push' for ever more complex abilities at persisting in the presence of other RCPROs. The traditional evolutionary story continues from here.

On the opposite side of the coin from this idea is the notion that death, or rather extinction, is NOT a necessary part of life. If there were infinite resources, the persistence ratchet of reproducing at a faster rate than deleterious mutations would ensure that persistence would still increase over time even though there would be no 'selection pressure' as used in the theory of evolution. Although in such a system you may find that one type of organism object would start feeding on another and therefore increase the preys mutation rate to the point where it becomes extinct, there is no *need* for such a competition to occur. Depending on chance, and the possibilities laid available by physics, in a world of infinite resources, life could in theory become more complex and more capable at persisting *without* the need for survival of the fittest. I argue that it is the presence of a persistence ratchet 'mechanism' that enables life to become more capable at persisting, more complex with time. It is only the fact of finite resources that leads to survival of the fittest in the traditional evolutionary sense.

4.5 Layers of Complexity

Another issue that I want to briefly mention is how do layers of complexity arise. Why do cells form into organisms that form into social units, etc .. ?

As the RCPROs slowly climb the scale of ability at persisting, they have to maintain the ratchet mechanism if they are to remain at the new levels of persistence indefinitely. The ratchet is what ensures that a particular RCPRO will not become extinct. If it is not holding true for a given RCPRO then there is an expected time for that RCPRO to become totally extinct. After that time it has to be rediscovered if it is to re-emerge. Hence only those RCPROs with the ratchet relationship holding true will survive.

As this ratchet is preserved when the rate of replication is greater than the rate of deleterious mutations there are two ways of maintaining the ratchet. One is to increase the rate of replication, the other is to decrease the rate of deleterious mutations. Nature appears to have toyed with both of these approaches, but most of all it has tried to reduce the number of mutations that will be deleterious. Note that in this context I am not talking about mutations to a particular gene mechanism, rather mutations to part of a RCPRO which 'push' is outside of its normal cycle and are therefore a potential threat to the integrity of the RCPRO.

In this way RCPROs are likely to develop more complex abilities at persisting with time as they become more adaptive and thereby less susceptible to deleterious mutations. I argue that as they become more adaptive, so they are likely to become multi-pathed RCPROs rather than single pathed RCPROs, as discussed earlier, and thereby can become holders of state. These new holders of state can then form new 'particle' objects at a higher level of complexity, a higher level of abstraction. This, I believe is what drives the move in life towards ever increasing layers of complexity.

4.6 The quantum conjecture

The final point that I want to mention is the one that I am least qualified to talk about, but appears to me to be quite an essential ingredient of the whole argument. So far the argument has been about why levels of persistence should be capable of increasing with time in dynamical systems. To explain this I have

introduced the persistence ratchet. The persistence ratchet acts as an explanatory tool as to why the next level of persistence has a fair chance of coming into existence given the current level of persistence. It is in effect an induction argument. All induction arguments need to have a starting point. So the glaring hole at the moment is this: how does *any* level of persistence with a persistence ratchet come into being in the first place. How do we explain the existence of the first RCPRO ?

The conjecture is that the whole ball game of persistence ratchets through exponential growth or search gets going at the quantum level. In a very simple reading quantum mechanics appears to unfold in two distinct phases. First a superposition is generated and then, according to the stability probabilities of each of the possible states, an actual state is 'chosen' when the superposition collapses. It is a generate and select process, a search process. It is a process that stochastically selects for stable states - persistence will generally increase with time. As quantum mechanics appears to offer us a form of increasing persistence at the smallest level of our universe, this for me is good enough to explain how the first persistence ratchet comes into being, and therefore how this sort of induction over persistence ratchets explains the existence and growing complexity of life.

5 Summary and Conclusion

This paper has looked at the nature of dynamical systems, first in the context of which types of systems can be classified as being search process, and second by looking at which types of system should be capable of supporting autopoiesis. In conclusion I am arguing that autopoiesis is most likely to occur within a search process that supports the existence of *persistence ratchets*. I have used the idea of persistence ratchets to form a kind of inductive argument for why ever increasing abilities at persisting - ever more complex forms of life - should naturally arise in such systems. I have then made the conjecture that it is the nature of quantum mechanics that makes the initiation of this induction a likely occurance in our universe as quantum mechanics itself appears to be like a search process.

There is still however, one very large part of the story that I believe needs further explanation: what kinds of search *spaces* will be capable of supporting autopoiesis. I hope through the above arguments I have convinced the reader that it is in search *processes* that autopoiesis is possible, and that I have weakened the requirements on the search spaces of such a system, as they no longer need to 'lead' the dynamics to increasing levels of complexity. As long as the increasing levels of complexity are possible, the persistence ratchet (through exponential growth or a search algorithm) and enough time should enable 'life' to find the extra abilities at persisting. However, the search space must surely still have some constraints on it and this subject I have not touched upon here.

Implicit in this argument comes a personal definition of life as the following:

Life is that which increases its ability at persisting

I have also given a more concrete definition of persisting objects to show that the concept is a useful one. What I have not been able to address yet is whether or not such a measure of persistence will always exist ? Neither have I answered the question: is persistence a property of a system that is largely independent of the measurement method used, or can one system both appear to have and to not have persistence depending on how you look ? These are thorny issues that have to be addressed in future work.

Also further work needs to be done to show actual graphs of persistence measures over time for known search processes. In fact there are of course various details of all of this paper that need further development, but the general thread of the whole argument is presented here, and future work will fill in the gaps.

The ultimate take home message of the paper is that if you want to study autopoiesis, you have to be using a dynamical system that can support search processes, either by allowing exponential growth or by using a search algorithm.

References

Davis, L. (Ed.). (1987). *Genetic Algorithms and Simulated Annealing*. Morgan Kaufmann.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley.

Kauffman, S. (1993). *Origins of Order*. Oxford University Press, New York.

Maturana, H. R., & Varela, F. J. (1980). *Autopoiesis and Cognition : the Realization of the Living*. Reidel Dordrecht.

Syski, R. (1992). *Passage Times for Markov Chains*. IOS Press.

In: Retkowsky, F. P. (Ed.). (1998)
The 11th White House Papers: Cognitive Science Research Paper 495
School of Cognitive & Computing Sciences, University of Sussex

The Effect of Mood on the Accessibility of Reasons Why Positive or Negative Future Events Might Happen: An Application of Availability Heuristics to Worry-based Pessimism

Helen M. Startup & Graham C.L. Davey †
helenst@cogs.susx.ac.uk and grahamda@cogs.susx.ac.uk

School of Cognitive & Computing Sciences
University of Sussex
Brighton
BN1 9QH

Abstract Objectives
Because pathological worriers have unrealistically high expectations of negative events happening, the present study investigated the effect of mood on the generation of reasons why future events might happen, and on judgements about the probability of such events happening.

Design
The study used a between-subjects design in which different groups of nonselected subjects were given either negative, positive or neutral mood inductions.

Methods
After separate mood inductions, subjects were asked to give as many reasons as they could think of for and against either a hypothetical negative future event happening or a hypothetical positive future event happening. Finally, all subjects were asked to rate the likelihood of these future events happening to them.

Results
Inducing negative mood resulted in subjects providing significantly more reasons why negative future events might happen, and inducing a positive mood resulted in subjects giving significantly more reasons why positive future events might happen. Furthermore, subjects in the negative mood induction judged the negative future event as more likely to happen than did subjects in either the positive or neutral groups. Subjects in the positive mood condition judged the positive future event as more likely to happen than did subjects in either the negative or neutral conditions.

Conclusions
This suggests that the unrealistically high estimates of the likelihood of negative future events made by chronic and pathological worriers may depend on the accessibility of reasons why the negative event might happen, and that accessibility of these reasons can be influenced by current mood state.

†This research was supported by ESRC Research Studentship No.R00429734640 to Helen Startup

1 Introduction

Chronic or pathological worrying is a central feature of depression and many of the anxiety disorders . It is a cardinal diagnostic feature of generalized anxiety disorder (GAD) (Barlow, Blanchard, Vermilyea, Vermilyea & DiNardo, 1986) and is present to some extent in most other anxiety disorders (Barlow, 1988). Brown, Antony & Barlow (1992) have found that scores on the Penn State Worry Questionnaire (PSWQ) distinguish GAD subjects from all other anxiety disorder patients, but that patients diagnosed as suffering a variety of other anxiety disorders (such as panic disorder, panic disorder with agoraphobia, social phobia, simple phobia, obsessive-compulsive disorder) also score higher on worry than normal subjects. In addition, pathological worrying is a condition associated with extreme emotional discomfort, and measures of worrying are highly correlated with measures of trait anxiety (e.g. Davey, Hampton, Farrell & Davidson, 1992; Meyer, Miller, Metzger & Borkovec, 1990), state anxiety (e.g. Metzger, Miller, Cohen, Sofka & Borkovec, 1990; Wisocki, Handen & Morse, 1986), and depression (e.g. Meyer et al., 1990; Metzger et al., 1990).

One of the central features of worrying is that it is primarily future-oriented, and tends to represent anxious concern over threatening or problematic future events. In a survey of worries in a non-clinical population, Davey, Tallis & Capuzzo (1994) found that over 78% of worries concerned immediate or medium-term future threats or dangers. In addition to this future-oriented characteristic of worry, worriers tend to label more future events as threatening than nonworriers (Davey et al., 1992), and also show a greater degree of belief in these future threatening events happening than nonworriers (MacLeod, Williams & Bekerian, 1991; MacLeod, 1994). Chronic worriers give higher probability estimates of future negative events happening than matched control subjects, and similar findings have been found with GAD patients, depressed subjects (Butler & Mathews, 1983) and nonclinical subject populations with high trait anxiety scores (Butler & Mathews, 1987).

MacLeod (1994) has attempted to explain this bias towards worriers giving higher probability estimates that future negative events will happen by alluding to the seminal work on judgemental heuristics pioneered by Tversky & Kahneman (1973). Tversky & Kahneman (1973) argued that frequency or probability judgements are determined by the ease with which either instances of the event can be retrieved from memory (availability heuristic), or the ease with which causal explanations for the event can be constructed or retrieved from memory (simulation heuristic). In the case of chronic or pathological worriers, MacLeod (1994) outlines four possible ways in which judgemental heuristics might lead to worriers believing that future negative events are more likely to happen than do nonworriers. First, because of the considerable amount of time that worriers spend ruminating about a problem, they may have elaborated many more reasons why a future negative event may happen than would a nonworrier; the sheer volume of these reasons and their ease of accessing will determine judgements about the probability of the event happening. Second, it may be the case that worriers have actually experienced in the past more examples of the forthcoming negative events than nonworriers (cf. Borkovec, 1994). Thus worriers may base their probability judgements on the ease of recall of similar events from long-term memory. Third, worriers may already have thought quite extensively about the probability of the future negative event happening, and simply retrieve pre-formed judgements about the event. The more the worrier thinks about this judgement, the more available the judgement will be and the more entrenched it is likely to become. Fourth, since worrying is commonly associated with negative mood in the form of heightened levels of anxiety and depression, worriers may use their mood state as a direct source of information in making a judgement, as with 'I feel anxious/sad there must be something bad about to happen' (cf. Schwarz & Clore, 1987) or alternatively, negative mood may instigate the construction of scenarios about whether the future negative event can be prevented by them or not. For example, their negative mood may allow them to construct more scenarios related to how the negative event cannot be

prevented than how it can be prevented.

In a test of the application of the availability heuristic to explaining worry-related pessimism about future negative events, Macleod, Williams & Bekerian (1991) found that severe worriers tended to generate more reasons than nonworriers why putative future negative events might happen and fewer reasons why they might not. In addition, when severe worriers were able to think of more reasons why the negative event might happen and fewer reasons why it might not, this was associated with perceiving that the event was more likely to happen. MacLeod et al. (1991) hypothesised that this effect was the result of worriers spending more time elaborating reasons why negative events might happen than nonworriers, and it is the number and accessibility of these reasons that determines the pessimistic probability judgement. The effects found in this study could not be explained by differential moods influencing judgements, because self-reported mood measures did not differ significantly between worriers and nonworriers throughout the experiment.

Although this study does support the hypothesis that worriers judge future negative events as more probable because they have access to a larger number of reasons why the event might happen than do nonworriers (the simulation heuristic), it still begs the question of whether this is a dispositional feature of worriers or whether there are individual variables that can influence pessimistic judgements in the same way in nonworriers. For example, while catastrophic worrying is a central feature of pathological worrying, this can be facilitated in nonpathological worriers by manipulating negative mood (Johnston & Davey, 1997) and poor problem-solving confidence (Davey, Jubb & Cameron, 1996).

While MacLeod et al. (1991) found that in their study differential mood states between worriers and nonworriers could not account for the results, it is possible that negative mood might influence pessimistic judgements in a way predicted by the simulation heuristic even in individuals who are not characterised as severe or pathological worriers. This is because negative mood may facilitate access to negative information in a mood-congruent fashion (Bower, 1981; Teasdale, 1983): when in a negative mood individuals may be able to access more readily information about why future negative events might happen than either why negative events will not happen or positive events might happen.

The present study uses a mood induction procedure to investigate the effects of negative and positive mood states on the generation of reasons why negative or positive future events might happen, and relates these effects to the generation of probability judgements about the events' occurrence.

2 Method

2.1 Subjects

The subjects were 60 undergraduate students from the University of Sussex. They comprised 29 females and 31 males. Their ages ranged from 17 yr to 42 yr with a mean age of 22 yr. All subjects were volunteers of which 30 received a small fee for their participation and the remainder were not paid.

2.2 Procedure

Subjects were assigned randomly to one of three groups. These groups were labelled Negative (N=20), Positive (N=20) and Neutral (N=20) depending on the valency of the mood induction they were to experience. Subjects were tested individually in a small room containing an audio-cassette player with headphones and an angle-poise lamp. Brightness levels in the room could be adjusted by opening or closing the window blinds and using the angle-poise lamp or main fluorescent lamp for lighting. Subjects were informed that the purpose of the study was to examine their ability to generate reasons both for and against future hypothetical scenarios, and that for the purpose of the present study two such scenarios had been chosen. Subjects were also informed that they would be required to fill in some questionnaires

and complete some tasks. Finally, subjects were told that in order to alleviate boredom, a short break would occur in the proceedings in which they would be asked to relax and listen to a short extract of music.

Stage 1 All subjects were asked to complete the Penn State Worry Questionnaire (Meyer, Miller, Metzger & Borkovec, 1990) as a measure of trait worrying, and then asked to rate their current levels of anxiety, sadness and happiness on separate visual-analogue 100-point scales (where 0= not at all anxious/sad/happy, and 100= extremely anxious/sad/happy).

Stage 2 (Mood Induction) Subjects were told to take a break from writing and to place the headphones on their head so that they could listen to a short piece of music. Subjects were told just to listen to the music and not write during this period. Subjects were then asked to listen to 8 min. of audiotaped music. Group Negative listened to music which a pilot study had indicated induced negative mood (Gyorgy Ligeti Lux Aeterna). To facilitate this negative mood, lighting in the experimental room was subdued by drawing the window blinds and using only the angle-poise lamp for illumination. Group Positive listened to 8 min. of music designed to induce a positive mood (Vivaldi, The Four Seasons) during which the blinds in the room were raised to allow full daylight, and the central light and angle-poise lamp were switched on. Group Neutral listened to a tape containing music which a pilot study had indicated produced no significant changes in mood measures (Chopin, Waltz). During the playing of this tape the window blinds were raised and the angle-poise and central room lights were off. During the pilot study, and in addition to rating their mood both before and after listening to the extracts, subjects were asked to think up adjectives that appropriately described the music for them. The negative music elicited adjectives such as 'eerie', 'mysterious', 'spiritual', 'relaxing' and 'morbid'. The positive music elicited adjectives such as 'exciting, 'jolly', dynamic', and 'racy'. The neutral music evoked adjectives such as 'calming' and 'monotonous'. The between-subjects design of the study ensured that subjects only listened to one type of music and so could not use the other types of music for comparative purposes.

At the end of the mood induction period, subjects were once again asked to complete the 100-point visual-analogue mood scales.

Stage 3 (Reasons Task) During this stage subjects were asked to generate reasons why they thought either a negative or a positive future event might or might not happen. The negative scenario was to "imagine that in the next 12 months you might break your leg", and the positive scenario was to "imagine that in the next 12 months you will make a new close friend". Each scenario was presented separately to the subject for a maximum of 8 min. (order of positive and negative scenario was counterbalanced across subjects). Subjects were given a single sheet of paper with a line ruled down the middle. The heading on the left side required subjects to write down reasons why the event might happen and on the right side reasons why the event might not happen. Subjects were instructed not to feel compelled to write their reasons in any specific order or quantity, but just to write them as they came to mind in the appropriate column. Subjects were further instructed to number their responses in the order in which they wrote them. When the time for the reasons task was completed, subjects were then asked to indicate a percentage probability that the event would happen to them in the next 12 months. When the second of the reasons tasks was completed, subjects were debriefed and told that the purpose of the music was to induce a temporary mood state. Subjects' responses clearly indicated a general unawareness of the purpose of the music during the study. All subjects were given the opportunity of listening to a relaxation tape for 5 min. before leaving the experimental room.

3 Results

3.1 Penn State Worry Questionnaire (PSWQ)

Mean score on the PSWQ for all 60 subjects was 48.9 (sd 12.80). This compares with a mean score of 47.6 found in general unselected samples of American subjects (N=1323) (Molina & Borkovec, 1994), and a mean score of 63.24 found in analogue clinical samples diagnosed as GAD by GAD-Q screening (N=324) (Molina & Borkovec, 1994). There was no significant difference in PSWQ across the three mood induction groups [F(2,59)=.15, p=.85]. Mean PSWQ scores for the three groups were 50.25 (sd 14.61), 48.00 (sd 11.65) and 48.70 (sd 12.53) for Negative, Positive and Neutral Groups respectively.

3.2 Mood measures

Table 6 shows the mean anxiety, sadness and happiness measures for each Group both before and after the mood induction. These were subjected to a group (negative vs. positive vs. neutral) x time (pre-induction vs. post-induction) analysis of variance. Anxiety ratings exhibited a significant group x time interaction [F(2,57)=7.56, p¡.001]. Although there was no significant difference between groups on the pre-induction measure [F=2.25, LSD all ps¿.05], there was a significant difference between groups on the post-induction anxiety measure [F(2,57)=7.34, p¡.001]. The Negative Group reported significantly higher post-induction anxiety ratings than both the Positive and the Neutral Groups [LSD, both ps ¡.05]. There was no significant difference in post-induction anxiety ratings between groups Positive and Neutral. These data were also subjected to an anxious (time 1) x anxious (time 2) within-subjects analysis of variance. This exhibited a significant anxiety x time interaction [F(2,57)=5.00, p¡.005], reflected as a significant increase in anxiousness between time 1 and time 2 in the Negative Group [mean diff=12.95, p¡.03] compared with a decrease in the Positive and Neutral Groups.

Mood	Anxiety		Sadness		Happiness	
	Pre-induction	Post-induction	Pre-induction	Post-induction	Pre-induction	Post-induction
Negative	20.5 (20.8)	26.5 (19.4)	14.0 (22.6)	27.5 (21.4)	63.5 (14.5)	44.5 (20.4)
Positive	21.3 (20.6)	9.05 (15.9)	11.3 (13.6)	9.9 (13.8)	67.9 (14.3)	75.2 (15.1)
Neutral	16.6 (17.2)	9.7 (13.1)	17.8 (21.0)	15.9 (22.7)	59.7 (12.4)	63.9 (13.9)

Table 6: Mean mood ratings taken before and after the mood induction (with standard deviations)

Sadness ratings also demonstrated a significant group x rating interaction [F(2,57)=5.97, p¡.005]. Again, there were no significant differences between groups on pre-induction ratings [F=1.62, LSD all ps¿.05], but a significant effect of group on the post-induction ratings [F(2,57)=4.09, p¡.05]. At post-induction, the mean sadness score was significantly higher for the Negative Group than for the Positive Group [LSD, p¡.05], but not significantly different to the Neutral Group [LSD, p¿.05]. Sadness ratings (as recorded at time 2) broke the assumption of homogeneity of variance and were therefore subjected to a square root transformation which successfully stabilised variances (p¡.05). The within-subject main effect of time was nonsignificant [F(1,57)=2.18, p¿.1]. However, the group x time interaction was significant [F(1,57)=8.52, p¡.001], reflecting the fact that sadness ratings increased significantly from time 1 to time 2 in the Negative Group but not in the Positive or Neutral Groups [mean diff.=15.35, p¡.02].

Happiness ratings also exhibited a significant group x rating interaction [F(2,57)=19.35, p¡.001]. Pre-induction happiness ratings did not differ significantly across groups (LSD, all ps ¿.05), but did differ significantly between groups at the post-induction stage [F(2,57)=17.22, p¡.001]. Happiness ratings for the Negative Group were significantly lower at post-induction than ratings for both the Positive and Neutral Groups, and the Positive Group showed higher post-induction happiness ratings than the Neutral Group [LSD, all ps¡.05]. When subjected to a within-subject analysis of variance, the happiness (time 1 vs. time 2) x group interaction was significant [F(1,57)=19.35, p¡.001], reflecting the fact that happiness ratings go down in the Negative Group [mean diff.=-4.78, p¡.001] but up in the Positive and Neutral Groups.

3.3 Reasons Task

Figure 1 (not included) shows the 'response profiles' for each mood condition - the mean number of reasons for and against the occurrence of the positive and negative scenarios for the three mood conditions. These data were subjected to a group (negative vs. positive vs. neutral) x scenario (good vs. bad) x reasons (for vs. against) analysis of variance, with simple bonferroni contrasts on the mood variable. The three way interaction was found to be highly significant [F(2,57)=18.70, p¡.001]. Contrasts (bonferroni comparing group negative to neutral and group positive to neutral) showed that the 'response profile' of Group Negative (the difference between reasons for and against good and bad scenarios) was not significantly different from that of Group Neutral [t=1.72, p=.09], however, this result was approaching significance. The 'response profile' of the Positive Group was, however, highly significantly different from that of the Neutral Group [t=-4.22, p¡0.00].

In the case of the bad scenario (breaking a leg), a group (positive vs. negative vs. neutral) x reasons (for vs. against) mixed design analysis of variance with simple bonferroni contasts on the mood variable was performed. The two way interaction was found to be highly significant [F=(2,57), p¿.000]. Contrasts (bonferroni, comparing group negative to neutral and positive to neutral) suggested that the difference between the number of reasons given for the event occurring (reasons for) and the number of reasons against the event occurring (reasons against) in the next twelve months was significantly greater for the Negative Group than for the Neutral Group [t=-2.21, p¡.02]. The difference between reasons for and against the bad event occurring in the next twelve months was, however, significantly smaller in the case of the Positive Group as compared with that of the Neutral group [t=2.4, p¡.01]. In the case of the bad scenario (breaking a leg), the number of reasons given against the event occurring (reasons against) showed no significant effect of group [F(2,57)=0.51, p¿.1]. However, the number of reasons given for the event occurring in the next twelve months (reasons for) was significantly affected by group [F(2,57)=8.65, p¡.001]. More specifically, the Negative Group gave significantly more reasons for the bad scenario occurring than did either the Positive Group or the Neutral Group [LSD, both ps¡.05]. There was no significant difference in the number of reasons given for the bad scenario occurring between the Positive and Neutral Groups [LSD, p¿.05].

In the case of the good scenario (making a new close friend), a group (positive vs. negative vs. neutral) x reasons (for vs. against) mixed design analysis of variance with simple bonferroni contrasts on the mood variable was performed. The two way interaction was found to be highly significant [F(2,57)=10.37, p¡.000]. Contrasts (bonferroni comparing group negative to neutral and positive to neutral) suggested that the difference between the number of reasons given for (reasons for) the good event and the number of reasons against (reasons against) the event occurring in the next twelve months was not significantly different in the case of the Negative and Neutral mood groups [t=.36, p¿.5]. However, the difference between the number of reasons for the event (reasons for) and the number of reasons against (reasons against) the event occurring in the next twelve months was significantly greater in the

case of the Positive Group than in the case of the Neutral Group [t=-3.75, p¡.000]. Morover, the number of reasons given for this event not occurring in the next twelve months (reasons against) showed no significant effect of group [F(2,57)=0.97, p¿.1]. However, the number of reasons for the good scenario occurring was significantly influenced by the mood condition [F(2,57)=5.43, p¡.01]. Specifically, the Positive Group gave significantly more reasons for the good scenario happening than did either the Negative Group or the Neutral group [LSD, both ps¡.05]. There was no significant difference between Negative and Neutral groups on this measure [LSD, p¿.05].

Finally, analysis of the reasons task data for the Neutral Group exhibited a significant effect of reasons (for vs. against) [F(1,19)=11.85, p¡.005] but not of scenario (good vs. bad) [F(1,19)=0.10, p¿.1], suggesting that when subjects are unaffected by a mood induction, generating reasons for an event occurring predominates - regardless of whether the event is a positive or negative one.

3.4 Event Probability Judgements

Figure 2 (not included) shows event probability judgements for the good and bad scenarios for the three experimental groups. In the case of the bad scenario, there was a marginally significant effect of group [F(2,57)=2.67, p=.077], in which the Negative Group rated the bad event as more likely to occur than did the Positive or Neutral Groups. In the case of the good scenario, there was a significant effect of group [F(2,57)=3.33, p¡.05], in which the Positive Group rated the good event as significantly more likely to occur than did either the Negative or Neutral Groups [LSD, both ps¡.05].

4 Discussion

The results suggested that manipulating mood does significantly affect the generation of reasons why either a negative or positive future event might happen (pro reasons). In particular, (1) inducing a negative mood resulted in significantly more reasons given for why a negative future event might happen (compared with subjects who had either a positive or neutral mood induced), and (2) inducing a positive mood resulted in subjects giving significantly more reasons why a positive future event might happen. Although the mood induction had no effect on number of reasons why either the positive or negative future event might not happen (con reasons), 'response profiles' (difference between reasons for and against good and bad events) differed significantly between groups positive and neutral but non significantly (marginally) between groups negative and neutral. More specifically, the contrast statistics suggets that 1) 'response profiles' differed significantly between groups negative and neutral in the context of the bad scenario but non significantly in the context of the good scenario. 2) 'response profiles' differed significantly between groups positive and neutral in the context of the good scenario, and interestingly also (negatively) in the context of the bad scenario. Secondly, these differences resulting from mood induction condition were also reflected in event probability judgements. Subjects in the positive mood induction judged the positive future event as significantly more likely to occur than did subjects in either the negative or neutral mood inductions. Similarly, there was a marginally significant effect for probability judgements concerning the negative future event in which subjects in the negative mood induction judged this event as more likely to occur than did either the positive or neutral groups. Notably, however, the overwhelming observation from the event probability data is that people gave higher probability values for the positive (new close friend) than for the negative (breaking leg) event, regardless of the mood induced. Perhaps, reality based probabilites are exerting an influence here, such that (for instance) in the case of the scenarios chosen it seems plausible that making a new close friend is indeed objectively and overwhelmingly more likely than breaking one's leg in the next twelve months (especially considering the subjects were undergraduate students). This may account for the imbalance in judgements. Nevertheless, this methodological isssue only adds weight to the marginally significant tendency of subjects in

a negative mood to judge negative events as more likely than individuals in induced positive or neutral moods.

MacLeod et al. (1991) have used this relationship between the accessibility of reasons why events might happen and judgements about the probability of the event occurring to explain why severe worriers have significantly higher estimates of bad future events happening to them than do nonworriers (see also Vasey & Borkovec, 1992). However, their explanation is based on worriers having already elaborated these reasons and them being more readily accessible than reasons why negative future events might not happen. The present results suggest that this relationship between number of articulated reasons and probability judgements can also be found in an unselected population of subjects when mood is manipulated. This indicates that prior elaboration of reasons through chronic worrying is not a necessary condition for the simulation heuristic to account for event probability estimates, but that mood also appears able to influence the accessibility of reasons why events might occur.

The main effect of mood on the generation of reasons why future events might or might not happen appears to be a reciprocal one in which negative mood influences the number of reasons why a bad future event might happen, and positive mood influences number of reasons why a positive future event might happen. Neither mood state influences reasons why events might not happen. The influence of mood on reason generation for incongruent scenarios may, however, be varied. Whereas the 'response profile'of negative mood subjects in the context of a bad scenario clearly differentiates from that of subjects in a neutral mood, there is no significance difference between the 'profile' of these groups in the context of the good scenario. This, suggetsts that a negative mood solely exerts its influence on reasons for an event occurring. In contrast the pattern is more complex in the case of a positive mood. The 'response profile'of subjects in a positive mood in the context of a good scenario differed significantly from those in an induced neutral mood, interestingly, however, there was also a significant (negative) difference in response profile for these groups in the context of the bad scenario. This finding suggests that perhaps a positive mood has an inhibitory effect on incongruent scenarios unlike a negative mood that merely exerts its influence on 'pro' reasons. The findings from the negative and neutral mood conditions are consistent with the fact that people find it easier to retrieve reasons why an event would happen rather than retrieve reasons why an event would not happen (Dunning & Parpal, 1989). Indeed, data from the neutral mood induction group demonstrate that, in the absence of a mood manipulation, pro reasons are significantly more readily generated than con reasons - regardless of whether the event being considered is a positive or negative one. Furthermore, if a mood is congruent with the valency of the event for which reasons are being sought (e.g. thinking of reasons why a bad event will happen while in a negative mood), such reasons are likely to be more readily retrieved than if the mood and valency of the event are incongruent (Blaney, 1986; Bower, 1981; Teasdale, 1983), and when pro reasons are more accessible this appears actively to inhibit con reasons (Tversky & Kahnemann, 1973; Hoch, 1984). These processes may well contribute to the effect of negative and neutral mood solely on the generation of pro reasons and not con reasons. Moreoever, given that reasons for a good event in a positive mood were offered in most abundance by the response group and also considering that individuals are prone to persist at processing which maintains a positive but not a negative mood (Sinclair and Mark, 1992; Martin and Stoner, 1996) one might indeed expect an inhibitory effect of incongruent scenarios in a positive but not a negative mood.

Processes which facilitate the elaboration or accessibility of reasons why bad or negative future events might happen will provide some explanation of why chronic worriers have such unrealistically high judgements about the likelihood of such events happening (Vasey & Borkovec, 1992; MacLeod, 1994). For example, the catastrophising process characteristic in pathological worriers provides a suitable mechanism for the generation and maintenance of reasons why bad future events might happen. Catastrophising is a persistent iterative process in which the worrier continues to generate worsening

scenarios about the worry topic (Davey & Levy, 1997; Vasey & Borkovec, 1992), and this iterative process allows for both the generation and elaboration of reasons why bad future events might happen. Similarly, the dysphoric and negative mood frequently associated with pathological worrying (Davey et al., 1992; Meyer et al., 1990) provides a congruent context for facilitating the accessibility of reasons why bad future events might happen. Thus, both catastrophising and negative mood provide the worrier with processes which facilitate the elaboration and accessibility of reasons why bad future events might happen. These reasons in turn provide a basis for the elevated probability judgements that worriers make about such events.

Finally, there are two possible limitations to the interpretations of the present results. Firstly, the recording of the pro/con reasons was done on a single answer sheet with a ruled line down the middle. This leaves open the possibility of contamination of the con reasons with the pro reasons, as when subjects simply generate opposite reasons, using their initial response as a cue. Subjects were, however, asked to number their responses in the order in which they entered them to check the above possibility. Indeed post hoc examination of the response sheets revealed that (for instance) subjects did not (for instance) simply generate a con reason by giving the opposite of a pro reason and vice versa. Indeed the nature of initial response (pro vs. con) and the patterning of subsequent responses apperared varied across subjects. Moroever, in the main, within subject pro and con reasons appeared to be qualitatively different. Secondly, because subjects make the event probability judgement after generating reasons why these events might happen, they may interpret the task as one of having to make the probability judgement congruent with the reasons they have generated. However, there are good reasons for believing that accessibility of reasons about why an event will or will not happen does have a causal influence on probability judgement regardless of whether demand effects are influencing probability judgements. For example, experimental manipulation of the generation of reasons why events might or might not happen affects probability judgements in a predictable way (e.g. Sherman, Zehner, Johnson & Hirt, 1983). This suggests that event probability judgements are related to the quantity and accessibility of reasons regardless of whether demand effects might be influencing the data in individual studies. In addition to this, arguments for a demand interpretation of the present results was not supported in post-experimental discussions with the subjects - none of whom gave any indication of believing there to be a link between the reasons generated and the event probability judgement, and many even believed that the judgement task was a distractor unrelated to the reason generating task.

5 References

Barlow D.H. (1988) Anxiety and its disorders. New York: Guilford Press.

Barlow D.H., Blanchard E.B., Vermilyea J.A., Vermilyea B.B. & Di Nardo P.A. (1986) Generalized anxiety and generalized anxiety disorder: Description and reconceptualization. American Journal of Psychiatry, 143, 40-44.

Blaney P.H. (1986) Affect and memory: A review. Psychological Bulletin, 99, 229-246.

Borkovec T.D. (1994) The nature, functions, and origins of worry. In G.C.L. Davey & F. Tallis (Eds) Worrying: Perspectives on theory, assessment and treatment. Chichester: Wiley.

Bower G.H. (1981) Mood and memory. American Psychologist, 36, 129-148.

Brown T.A., Antony M.M. & Barlow D.H. (1992) Psychometric properties of the Penn State Worry Questionnaire in a clinical anxiety disorders sample. Behaviour Research & Therapy, 30, 33-37.

Butler G. & Mathews A. (1983) Cognitive processes in anxiety. Advances in Behaviour Research & Therapy, 5, 51-62.

Butler G. & Mathews A.(1987) Anticipatory anxiety and risk perception. Cognitive Therapy & Research, 11, 551-565.

Davey G.C.L. & Levy S. (1997) Catastrophic worrying: personal inadequacy and a perseverative iterative style as features of the catastrophising process. Submitted.

Davey G.C.L., Hampton J., Farrell J.J. & Davidson S. (1992) Some characteristics of worry: Evidence for worrying and anxiety as separate constructs. Personality & Individual Differences, 13, 133-147.

Davey G.C.L., Jubb M. & Cameron C. (1996) Catastrophic worrying as a function of changes in problem-solving confidence. Cognitive Therapy & Research, 20, 333-344.

Davey G.C.L., Tallis F. & Capuzzo N. (1994) The phenomenology of non-pathological worry: A preliminary investigation. In G.C.L. Davey & F. Tallis (Eds) Worrying: Perspectives on theory, assessment and treatment. Chichester: Wiley.

Dunning D. & Parpal M. (1989) Mental addition versus mental subtraction in counterfactual reasoning: On assessing the impact of personal actions and life events. Journal of Personality & Social Psychology, 57, 5-15.

Hoch J.S. (1984) Availability and interference in predictive judgement. Journal of Experimental Psychology: Learning, Memory & Cognition, 10, 649-662.

Johnston W.M. & Davey G.C.L. (1997) The psychological impact of negative TV news bulletins: The catastrophising of personal worries. British Journal of Psychology, 88, 85-91.

MacLeod A.K. (1994) Worry and explanation-based pessimism. In G.C.L. Davey & F. Tallis (Eds) Worry: Perspectives on theory, assessment and treatment. Chichester: Wiley.

MacLeod A.K., Williams J.M.G. & Bekerian D.A. (1991) Worry is reasonable: The role of explanations in pessimism about future personal events. Journal of Abnormal Psychology, 100, 478-486.

Martin L.L and Stoner P. (1996). Mood as input: What we think about how we feel determines how we think. In L. Martin and H. Tesser. Striving and Feeling - Interactions among goals, affect and self regulation. Lawrence Erlbaum Associates. Mahwah, New Jersey.

Metzger R.L., Miller M.L., Cohen M., Sofka M. & Borkovec T.D. (1990) Worry changes decision making: The effect of negative thoughts on cognitive processing. Journal of Clinical Psychology, 48, 76-88.

Meyer T.J., Miller M.L., Metzger R.L. & Borkovec T.D. (1990) Development and validation of the Penn State Worry Questionnaire. Behaviour Research & Therapy, 28, 487-495.

Molina S. & Borkovec T.D. (1994) The Penn State Worry Questionnaire: psychometric properties and associated characteristics. In G.C.L. Davey & F. Tallis (Eds) Worrying: Perspectives on theory, assessment and treatment. Chichester: Wiley.

Schwarz N. & Clore G.L. (1987) How do I feel about it?: The informative function of affective states. In K. Fiedler & J. Forgas (Eds) Affect, Cognition and Social Behavior. Toronto: Hogrefe International.

Sherman S.J., Zehner K.S., Johnson J. & Hirt E.R. (1983) Social explanation: The role of timing, set and recall on subjective likelihood estimates. Journal of Personality & Social Psychology, 44, 1127-1143.

Sinclair R.C and Mark M.M.. (1992). The influence of mood state on judgment and action: Effects on persuasion, categorization, social justice, person perception and judgemental accuracy. In L. Martin and Tesser (Eds.). The Construction of social Judgments (pp.165-193). Hillsdale, NJ: Lawrence Erlbaum.

Teasdale J.D. (1983) Negative thinking in depression: Cause, effect or reciprocal relationship? Advances in Behaviour Research & Therapy, 5, 3-25.

Tversky A. & Kahnemann D. (1973) Availability: A heuristic for judging frequency and probability. Cognitive Psychology, 5, 207-232.

Vasey M. & Borkovec T.D. (1992) A catastrophising assessment of worrisome thoughts. Cognitive Therapy & Research, 16, 505-520.

Wisocki P.A., Handen B. & Morse C.K. (1986) The Worry Scale as a measure of anxiety among homebound and community elderly. Behavior Therapy, 5, 91-95.