THE SEVENTH WHITE HOUSE PAPERS

Graduate Research in the Cognitive
and Computing Sciences at Sussex

CSRP 350

editors:
Peter de Bourcier
Ronald Lemmen
Adrian Thompson

December 1994

Contents

Dedication

The editors would like to dedicate the Seventh White House Papers to Berry Harper for her great contributions to COGS. She will be greatly missed.

Preface

Each year since 1988, COGS graduate students have been meeting at Sussex University's conference centre, the White House, located at the Isle of Thorns, near Haywards Heath. Over several days, students are given the opportunity to give presentations on their work, exchange ideas, and most importantly, socialise. Out of this annual event arises a collection of short papers that have come to be known as the White House Papers.

This summer, all postgraduate students at COGS were invited to submit papers of around 2000 words for inclusion in the Seventh White House Papers. The resulting collection reflects work in many diverse areas of research, such as philosophy, behavioural ecology, computer vision, linguistics, medical informatics, HCI, software design, and artificial life.

Many people have sacrificed their time and effort to organise the Isle of Thorns Workshop and the White House Papers. The editors would like to thank the secretaries, postgraduate students, and members of the faculty for their help in making the workshop and papers a success. Thanks also to prof. Matthew Hennessy for financial support and special thanks to Jo Brook, for her continuous dedication to the organisation of the Isle of Thorns Workshop and White House Papers. Without her help, and extensive knowledge of the "art" of LaTeX, this collection would have appeared considerably later and, we fear, with hand-written page numbers. Finally, we would like to thank Berry Harper for her help in producing the White House Papers over many years, we dedicate this year's collection to her.

Peter de Bourcier
Ronald Lemmen
Adrian Thompson
December 1994

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

What is Synthetic Behavioural Ecology?

Michael Wheeler and Peter de Bourcier
michaelw@cogs.susx.ac.uk, peterdb@cogs.susx.ac.uk

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

Abstract Behavioural Ecology seeks functional explanations of ecologically embedded animal behaviour; i.e., it concentrates on the adaptive consequences of behaviour in relation to ecological context. We describe and justify a theoretical approach that we call 'Synthetic Behavioural Ecology.' This framework for adaptive behaviour research seeks functional explanations of the synthetically embedded behaviour of animats (artificial animals), i.e., it concentrates on the adaptive consequences of behaviour in relation to a synthetic ecological context. The goal of synthetic behavioural ecology is to complement the ongoing work in the biological sciences on the relationship between ecology and behaviour.

1 Introduction

Adaptive behaviour is behaviour which increases the chances that an autonomous agent can survive in a dynamic, uncertain, and (possibly) hostile environment. In the field of research known as *simulation of adaptive behaviour* (see, e.g., (Cliff, Husbands, Meyer, & Wilson, 1994)), the aim is to produce adaptive behaviour in autonomous robots or simulated agents, i.e, in what might be thought of as *artificial* animals — henceforth *animats* (Wilson, 1985). This paper introduces a specific approach within the general simulation of adaptive behaviour paradigm. The approach, which we call *Synthetic Behavioural Ecology* (henceforth 'SBE'), involves using existing work in the biological sciences to guide the construction of simple simulated eco-systems populated by animats. Experiments are then carried out in the synthetic eco-systems, with the aim of contributing to the scientific understanding of the relationship between ecology and behaviour.

As far as we are aware, the actual phrase 'synthetic behavioural ecology' had not been used prior to (de Bourcier & Wheeler, 1994). And, although important aspects of the SBE-framework will be familiar from other areas of simulation of adaptive behaviour research (e.g., the use of closed-environment simulations — see sections 3.2 and 3.3), we have deliberately brought together and made explicit certain theoretical and methodological commitments which, we believe, make SBE a distinctive approach. The aim of the current paper is to explain and to justify those commitments. The following paper in this collection describes some actual experiments using the SBE-framework.

2 Behavioural Ecology

2.1 Explanations in Biology

The distinguishing characteristic of behavioural ecology as a discipline in biology is that it concentrates on *functional* explanations of *ecologically-embedded* animal behaviour (Krebs & Davies, 1987). We can bring out what is meant by 'functional explanation in biology' by way of the Tinbergen questions. Tinbergen (1963) identified four different questions that could be asked about a given behaviour. These are questions of:

1. *Causation:* the mechanisms underlying that behaviour. These include the triggering environmental stimuli and the cognitive/neural/hormonal processes active in the animal.

2. *Development:* the ontogenetic sources of that behaviour. For example, birds often learn their mating songs from their parents.

3. *Evolutionary History:* the phylogenetic development of that behaviour. For instance, in principle, it should be possible to trace out the route by which initially incidental movements or responses, on the part of certain animals, were modified over evolutionary time to become ritualized, stereotypic signaling patterns.

4. *Function:* the adaptive consequences of that behaviour. That is, the behaviour is investigated in terms of the role it plays in contributing to the survival and reproductive prospects (Darwinian fitness) of an animal. For example, if we want to understand *why* the female of the scorpionfly, *Hylobittacus apicalis*, mates for longer with males who woo her with larger insects as courtship gifts, then we had better identify the contribution made by that behaviour to the female's adaptive success. In fact, it appears to be because the female's capacity to produce eggs is limited by the food she has available to invest in those eggs. By adopting a strategy such that the larger the pre-nuptial gift, the longer the male is allowed to copulate, the female encourages the male to arrive with larger insects (as he will end up fertilizing more eggs), and thereby ensures that she will have more food available to invest in her eggs (Thornhill, 1979).

So what about *ecological embeddedness*? The adaptive success of an animal depends on that animal's behaviour; and the adaptive success of any particular behavioural strategy is *directly dependent upon* the animal's ecology — including its food, mates, competitors, and predators. So the process of Darwinian natural selection will tend to produce animals that are well-adapted to their ecological niches. In other words, the creatures that survive to pass on their genes will tend to be the well-designed foragers, the well-designed mate-finders, the well-designed predator-detectors etc.. The clear implication is that the adaptive consequences of an individual animal's behaviour cannot be investigated in isolation from the specific features of that animal's ecological niche.

2.2 Optimality Models

One way to investigate the relationship between behaviour and ecology is to compare the behaviour patterns of different species. This comparative approach is particularly useful in cases of related species, as differences in behaviour may well be direct reflections of differences in ecology. A complementary approach is to focus on the behavioural strategies of individual animals, and to analyse those behaviour patterns in terms of the *economic costs and benefits* of adopting those particular strategies in the relevant ecological settings.

In generating an economic model, an assumption has to be made about the 'currency' of the model. For example, in generating foraging models, various currencies, such as rate of food intake, food-finding efficiency, and risk of starvation may be used (Krebs & Davies, 1987). Economic models also incorporate assumptions about the set of constraints (bodily and environmental) operating on the set of possible phenotypic behaviours. The idea then is to produce quantitative predictions of the costs and benefits of a behaviour pattern to an individual — in a particular ecological context — and to combine that cost-benefit model with the prediction that, because natural selection tends to produce efficient agents, an individual will act in a way that maximizes its own net benefit; i.e., its behaviour will tend to be optimal. Hence economic models become *optimality models*.

Where the relationship under investigation is between an individual animal and an environmental resource (e.g., in solitary food-foraging), optimality models often make the assumption that individual fitness is independent of phenotypic frequencies. In this case, individual fitness is calculated without reference to the behaviour of other individuals in the population. But, in many cases of adaptive behaviour,

what counts as the optimal individual strategy will be determined by the frequencies with which the various available strategies are adopted by the other members of the population; i.e., individual fitness is frequency-dependent. For these multi-agent, frequency-dependent situations, the appropriate language for generating optimality models is provided by the field of *game theory,* combined with the concept of an *evolutionarily stable strategy.* Game theory (von Neumann & Morgenstern, 1944) is a field in which the formal analysis of a competitive scenario proceeds by way of a pay-off matrix. This matrix defines how the various available strategies fare, when pitted against each other. An evolutionary stable strategy, or 'ESS' (Maynard Smith, 1982), is a strategy which, when adopted by most members of a population, means that that population cannot be invaded by a rare alternative strategy.

So behavioural ecology is a well-founded sub-discipline in the biological sciences, with a well-developed theoretical machinery. But there is an explanatory gap between the abstract mathematical models constructed using cost-benefit/ESS approaches, and the behaviour of animals in their natural environments. For example, in a recent discussion of ESS signaling theory (see the next paper in this collection), Grafen and Johnstone observe that "the biological content of the [ESS] models is very limited. ... The existing body of theory is probably too simple to be applied convincingly to any empirical example" (Grafen & Johnstone, 1993, p.249). This is where SBE comes in.

3 Synthetic Behavioural Ecology

3.1 General Principles

We offer the following set of principles as definitional of the SBE-approach.

1. Like its biological parent-discipline, SBE concentrates on asking, and (we hope) answering, functional questions.

2. Also in parallel with behavioural ecology, SBE concentrates on explaining behavioural strategies in the context of the ecological situations in which those behaviour-patterns occur.

3. In common with much adaptive behaviour research, the SBE-methodology is to construct synthetic ecological contexts in which the behaviour of animats can be observed. By using information from the biological sciences to guide this process, the synthetic environments employed should be simplified and idealized, but not trivial.

4. The primary aim of SBE is to contribute to ongoing work in the biological sciences, by providing a distinctive theoretical platform for testing pre-existing hypotheses, and, perhaps, for suggesting and formulating new hypotheses about animal behaviour.

When the field biologist is investigating behaviour in natural ecologies, her control over what she believes to be the key factors affecting that behaviour is, in the vast majority of cases, minimal or, at best, partial. By contrast, in SBE, the parameters underlying the nature of the 'physical' environment (e.g., the rate at which food is replenished) and those driving basic aspects of the animats' behaviour (e.g., the rate at which the animats' hypothetical metabolisms require food-intake) are under direct experimental control. So one potentially important way in which SBE may contribute to our understanding of the complex relationship between ecology and behaviour is through experimentation with the values of the parameters affecting key aspects of the (synthetic) ecological context. And, of course, a simulation can be run over and over again from similar initial conditions, in order to test the robustness of hypotheses. Natural ecologies are rarely so compliant (cf. MacLennan & Burghardt, 1994)).

There are already studies in the simulation of adaptive behaviour literature, which foreshadow our general approach. Here we shall mention just two examples. Koza, Rice, and Roughgarden (1992) base an investigation on field studies which show that the foraging behaviour of the Caribbean *Anolis* lizard is close-to-optimal. Genetic algorithms were then used to evolve foraging behaviour in a population of simulated lizards. The artificially evolved behaviour was shown to be similar to that of the real lizards. And

te Boekhorst and Hogeweg (1994) used a simulated eco-system, based on a natural habitat at Ketambe, to investigate the formation of travel parties in orang-utans. The results from the simulation suggested the hypothesis that these travel parties were emergent properties of simple foraging behaviours in interaction with the structure of the environment. Further observations in the natural ecology provided evidence in support of this hypothesis.

3.2 SBE and Computational Neuroethology

SBE can be distinguished quite sharply from another approach in adaptive behaviour research, namely that of *computational neuroethology* (Beer, 1990; Cliff, 1991). Computational neuroethology endeavours to explain the neural mechanisms underlying animal behaviour, by using computers as environments in which to model the complex neural phenomena. From this it is clear that the questions answered by computational neuroethology will concern the causal mechanisms underpinning behaviour, and not the adaptive consequences of behaviour.

Of course, achieving a more complete understanding of behaviour requires that we answer causal and functional questions, so synthetic behavioural ecology and computational neuroethology are not in competition. They are complementary approaches. Moreover, although it is essential to keep the different questions apart, the fact remains that the two types of constraint will sometimes interact. For example, Miller and Cliff highlight the assumption — usually made in game theory models — that the dynamics of decision making can be accurately characterized without reference to implementational time-lags or speed-accuracy trade-offs (Miller & Cliff, 1994, pp.415-16). Thus any assumption that we can investigate functional constraints in complete isolation from causal constraints — *or vice versa* — will often amount to a useful idealization.

3.3 SBE and Synthetic Ethology

In the synthetic ethology of MacLennan and Burghardt (1994), as in SBE, populations of synthetic organisms are allowed to behave and evolve in simulated worlds. However, in synthetic ethology, the intention is not to model (in however idealized or selective a form) a complex system present in the natural world. So while we actively seek to develop our models in ways that respect aspects of existing biological theory, MacLennan and Burghardt make a virtue of the fact that their approach eschews the use of such constraints. As they put it, "[the] design of a simulation is heavily theory-laden and necessarily highly-selective. This is true even for models based on current theoretical and empirical understanding of the phenomena being studied, for out of the multitude of features in the natural situation, only a small fraction can be selected for modeling. This is the Achilles heel of simulation: An inappropriate selection vitiates the relevance of the model" (MacLennan & Burghardt, 1994, p. 162). But how serious is this worry? All processes of scientific modeling make simplifying assumptions. Of course, an "inappropriate selection" of a feature may mean that the model turns out to be inapplicable; but, from the admitted existence of this risk, we see no reason to infer that all simulations are doomed to being either unhelpful or irrelevant. Even where further research or observations in natural habitats show the simulation to be inaccurate in predicting every aspect of some behaviour, there is the distinct possibility that it is only certain features or parameter-values of the model that need to be changed, or that some hitherto unaccounted-for additional constraints are required. Indeed, the later, more enlightening research may have been suggested, or made possible, only through the impetus provided by the previous model.

4 Conclusions

The goal of SBE is to contribute to ongoing work in the biological sciences, by providing a methodology which may help to bridge the explanatory gap between (i) the predictions of abstract mathematical models and (ii) data from highly complex natural environments. We must stress that there is no suggestion that SBE provides any easy answers to the difficult problems faced by biologists in studying the relationship

between ecology and behaviour. Our claim is merely that SBE provides a new way of asking old questions and, in time, has the potential to find some new questions to ask. The following paper in this collection describes an example of SBE at work.

References

Beer, R. (1990). *Intelligence as Adaptive Behavior: An Experiment in Computational Neuroethology*. Academic Press, San Diego, California.

Cliff, D. (1991). Computational neuroethology: a provisional manifesto. In Meyer, J.-A., & Wilson, S. W. (Eds.), *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, pp. 29–39 Cambridge, Massachusetts. M.I.T. Press / Bradford Books.

Cliff, D., Husbands, P., Meyer, J.-A., & Wilson, S. W. (Eds.). (1994). *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*, Cambridge, Massachusetts. M.I.T. Press / Bradford Books.

de Bourcier, P., & Wheeler, M. (1994). Signalling and territorial aggression: An investigation by means of synthetic behavioural ecology. In Cliff, D., Husbands, P., Meyer, J.-A., & Wilson, S. W. (Eds.), *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*, pp. 463–72 Cambridge, Massachusetts. M.I.T. Press / Bradford Books.

Grafen, A., & Johnstone, R. (1993). Why we need ESS signalling theory. *Philosophical Transactions of the Royal Society: Biological Sciences*, *340*, 245–250.

Koza, J. R., Rice, J. P., & Roughgarden, J. (1992). Evolution of food-foraging strategies for the caribbean *anolis* lizard using genetic programming. *Adaptive Behavior*, *1*, 171–200.

Krebs, J. R., & Davies, N. B. (1987). *An Introduction to Behavioural Ecology* (2nd edition). Blackwell Scientific, Oxford.

MacLennan, B., & Burghardt, G. (1994). Synthetic ethology and the evolution of cooperative communication. *Adaptive Behavior*, *2*(2), 161–188.

Maynard Smith, J. (1982). *Evolution and the Theory of Games*. Cambridge University Press, Cambridge.

Miller, G. F., & Cliff, D. (1994). Protean behavior in dynamic games: Arguments for the co-evolution of pursuit-evasion tactics. In Cliff, D., Husbands, P., Meyer, J.-A., & Wilson, S. W. (Eds.), *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*, pp. 411–20 Cambridge Massachusetts. M.I.T. Press / Bradford Books.

te Boekhorst, I., & Hogeweg, P. (1994). Effects of tree size on travelband formation in orang-utans: Data analysis suggested by a model study. In Brooks, R., & Maes, P. (Eds.), *Proceedings of Artificial Life IV*, pp. 119–129 Cambridge, Massachusetts. M.I.T. Press.

Thornhill, R. (1979). Adaptive female mimicking behaviour in a scorpionfly. *Science*, *205*, 412–14.

Tinbergen, N. (1963). On aims and methods of ethology. *Zeitschrift fur Tierpsychologie*, *20*, 410–33.

von Neumann, J., & Morgenstern, O. (1944). *Theory of Games and Economic Behavior*. Princeton University Press, New Jersey.

Wilson, S. W. (1985). Knowledge growth in an artificial animal. In Grefenstette, J. J. (Ed.), *Proceedings of an International Conference on Genetic Algorithms and their Applications*, pp. 16–23 Pittsburg, PA and Hillsdale, New Jersey. Lawrence Erlbaum Associates.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Aggressive Signaling

Peter de Bourcier and Michael Wheeler
peterdb@cogs.susx.ac.uk, michaelw@cogs.susx.ac.uk

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

Abstract The scientific account of intra-specific aggressive signaling is incomplete. In part, this is due to the fact it is hard to identify the consequences of the different ecological contexts in which signaling has developed. The goal of our investigation is to complement the ongoing work in the biological sciences on this issue. Using the theoretical framework of *synthetic behavioral ecology* described in the previous paper in this collection we perform a series of experiments involving populations of simulated animals (animats), whose simulated world is such that there is competition for food. Each individual displays aggressive intention signals in line with its bluffing strategy that is determined by a form of artificial evolution in which there is no explicit fitness function. By varying, in energy terms, the cost of producing aggressive signals, and by analysing the population dynamics at different costs of signaling, we are able to provide evidence that the handicap principle (according to which higher costs enforce honesty) can apply in multi-agent ecologies.

1 Introduction

Communication research concerned primarily with artificial autonomous systems has tended to focus on the emergence of complex *cooperative* behaviors from simple signaling interactions between relatively unsophisticated individuals. But, in the natural world, not all communication is in the interests of *explicit* cooperation; that is, not all communication concerns group-ventures such as collective nest-building or collective foraging. Many signaling-behaviors occur during *confrontations* between competing aggressors, where the selfish goal of each adversary is the personal control of some resource (e.g., a food supply or a mate). Of particular interest is the use of signals during aggressive encounters between members of the same species — cases of *intra-specific* aggressive signaling. When animals of the same species come into conflict, the incidence of unrestrained battles is relatively low. Instead of all-out fights, confrontations tend to revolve around signaling displays which, more often than not, allow the contestants to conclude matters without the need for potentially damaging physical combat. (see the coverage of red deer mating competitions by and Clutton-Brock and Albon (1979).)

Despite the existence of an enormous literature in the biological sciences on intra-specific aggressive signaling, this significant sub-set of signaling-behaviors has received little attention from the autonomous agent movement. But aggressive signaling is a genuinely adaptive behavior about which there are unanswered questions. We use existing work in the biological sciences to guide the construction of a simple simulated eco-system populated by artificial animals — henceforth *animats* (Wilson, 1985) — who are competing for limited supplies of food.

2 Biological Background

For an explanation of the biological background against which the experimental model was developed see de Bourcier and Wheeler (1994). In short we define aggression as the disposition to fight more intensely

or for longer (due to Maynard Smith and Harper, 1988) and signaling as a behavior performed by agent X to agent Y such that i) the evolutionary function of X's behavior is to change the behavior of Y, and ii) X's behavior is designed to affect Y's dispositions to behave via Y's sensory mechanisms, and not by physical force. (This second condition is intended to prevent the situation where one animal pushes another from counting as an example of signaling behavior.) We define a 'territory' as "a more or less exclusive area defended by an individual or group" after Davies and Houston (1984, p.148).

Zahavi's *handicap principle* (Zahavi, 1975), is that the reliability of intention-signals could be increased if the animal concerned had to invest, in some way, in those signals. The fundamental idea is illustrated by the fact that a signal which is, for example, wasteful of energy is, as a consequence of that wastefulness, reliably predictive of the possession of energy; hence honesty is enforced. Only cost-free intention-signals would be open to exploitation by bluffers. Reliable signals of aggressive intent must be more costly in fitness terms than they strictly need be merely to communicate unambiguously the information at issue. Moreover, the costs involved must be differential. A specific signal indicating a particular level of intended escalation must be proportionally more costly to a weak individual than to a strong individual.

3 Experimental Model

We now describe the animat-environment system employed in our experiments. Our animats 'perceive' and 'act' in a simulated world, which is two-dimensional and non-toroidal; i.e., it is a flat plane, the edges of which are barriers to movement. The dimensions of the world are 1000 by 1000 units, and each animat is round and 12 units in diameter. Space is continuous. This is in contrast to the cellular environments favored in many simulations in which an animat occupies one cell on a rectangular grid, and moves from cell to cell. Discrete cellular environments, while useful in certain contexts, introduce severe limitations into the dynamics of perception and action (Cliff and Bullock, 1993).

In the simple synthetic eco-system employed here, an animat's environment consists of food, plus other animats. Each animat may move in any one of 36 directions corresponding to a full circle in 10 degree divisions about its current position. Movement takes place in response to sensory information picked up from the environment via two idealized sensory modalities. Both senses are distal, but have different ranges. The first — which we think of as idealized vision — provides information about other animats. The second — which we think of as idealized olfaction — provides information about food.

Vision is simulated using two-dimensional projective geometry. The visual system is based on a 36-pixel eye providing information in a full 360 degree radius around the animat, with an arbitrarily imposed maximum range, such that a hard threshold (a sharp cut-off) occurs at a distance of 165 units from the animat. Each pixel is radially oriented, and returns a real number in the range 0 to 1, corresponding to the proportion of that pixel's receptive field containing other animats. The olfactory system employs principles similar to those used for vision, the only differences being that olfactory range is only 35 units, and that food particles are treated as point sources. The olfactory value returned for each of the 36 directions is thus proportional to the density of food present (within range) in that direction.

Food is present in the world in the form of randomly distributed food particles, each carrying an energy value. When an animat lands on a food particle, the energy value of the particle is transferred to the animat, thereby incrementing that animat's energy level. If an animat lands on more than one particle of food at any one time-step, its intake is restricted to one unit. In order to replenish the resource, new food particles are added to the world (again with random distribution) at each time-step. The number of new particles to be added at each step is set by the user, but the resource is 'capped' so that the food supply is never more plentiful than at the beginning of the run. Hence the food supply is limited; and the members of the population are in intra-specific competition for access to the available resources. The initial population of animats is created by placing a number of individuals at randomly chosen positions in the world.

Each individual begins life with the same energy level as its peers. But animats lose energy in a number of different ways, namely: *Being alive*, where a small existence-cost is deducted from the energy level

of each surviving animat; *Moving*, if an animat moves in any direction it incurs a movement cost; *Reproducing*, If an animat achieves a high energy level, then it will asexually reproduce. The offspring, an only child placed randomly in the world is given the same initial energy level as each member of the population had at the start of the run, and the corresponding amount of energy is deducted from the parent; And finally *Fighting*, Fights occur when animats touch. Such 'physical' combat results in a large reduction in the energy levels of the participants.

If an animat's energy level sinks to 0, then that individual is deemed to have died. Under these circumstances, it is removed from the world. So, as well as increasing through births, population size can decrease through deaths.

Due to the fact that animats lose energy on a regular basis, food-finding is an essential task. To encourage foraging behavior, each animat has a hunger level — its disposition to move towards food — which is updated at each time step. This figure is calculated by adding a constant to the difference between the maximum individual energy level possible (i.e., the value at which reproduction takes place) and an animat's current energy level. Thus an animat's hunger is inversely proportional to its energy level. The addition of the constant is to increase the overall likelihood that foraging behavior will occur.

In addition to the disposition to find food, each animat has a level of aggression — its general disposition to approach and to engage with other individuals in its visual field. An aggressive movement is defined as one in which the individual in question moves directly towards another animat. Each individual's disposition to behave aggressively varies dynamically with its recent behavioral history. When an animat moves in an aggressive manner, its level of aggression is increased by an amount proportional to the previous aggression level. Conversely, a non-aggressive movement results in a decrease in an animat's level of aggression, by an amount proportional to the previous aggression level.

At each time-step, the direction in which each animat will move is calculated using a probabilistic equation (for details see de Bourcier and Wheeler 1994). We devised the movement-equation to have the following basic effects: for some animat *F*:- If there are no other animats in F's visual field, and no food within F's olfactory range, then F will make a random movement. If F can sense food, and F is hungry, then the probability of F moving in the direction of that food is proportional to F's degree of hunger. If there are other animats in F's visual field, then the probability that F will move in the direction of another animat is proportional to F's aggression level. But F is also likely to move in the opposite direction (away from the other animat) with a probability proportional to the threat which F perceives from that animat. So, for example, if F is only slightly aggressive, and perceives another animat as being a significant threat, then F is more likely to retreat rather than to attack; while, if F is highly aggressive, and perceives the other animat as posing little threat, then F is more likely to attack than to retreat.

4 Experimental Conditions

In previous experiments we have shown that a simple form of territoriality emerges from the interactions of the animats simple behavioural repertoire with the dynamics of the simulated environment (de Bourcier and Wheeler 1994; Wheeler and de Bourcier, 1994). The experiments we discuss here are are based on what happens when animats are given the capacity to produce signals indicating aggressive intentions (i.e., an individual's apparent disposition towards making aggressive moves). An individual animat signals whenever at least one other animat is within visual range. To enable these signals to be communicated, the animats' visual systems are 'tuned' to pick up the values of the aggressive signals coming from each of the 36 directions.

Aggressive signals are displays for which a signaling animat has to pay a cost, via a deduction from its energy level. This cost increases as the level of aggression signaled increases, so that it costs more in energy to make a more aggressive signal. Thus the costs involved are differential, in the sense required by the handicap principle, because, given a specific signal made by a strong (high-energy) individual, it will cost a weak (low-energy) individual proportionally more to produce the same signal. As what is being signaled is supposed to be an indicator of 'current willingness to fight', it is appropriate that what is taxed

is an animat's energy level; in this model, there is a direct correlation between energy and the capacity to wage battle. The ways in which costs are paid in natural environments may be more complex, although some notion of an 'appropriate link' still seems to apply (Stamp Dawkins, 1993).

In order for us to investigate the operation of the handicap principle in this multi-agent context, it had to be possible for the various members of the population to adopt different *signaling strategies*, where by a 'signaling strategy' we mean the degree to which the signal produced by an individual accurately reflects that individual's true level of aggression. Aggressive signals are produced in accordance with the calculation $S = A + ((C/100).A)$ where S is the value of the signal made, A is that individual's current aggression, and C is an individual-specific constant, the value of which is an integer in the range 0-100. So, for example, individual A may always signal a value for apparent aggression which is equivalent to 20% more than its actual aggression level ($C = 20$), while individual B may always signal a value equivalent to 75% more than its actual aggression ($C = 75$). This provides the potential for the concurrent existence of a range of strategies, arranged on a continuum from truth-telling to extreme bluffing. (The consistent signaling of actual aggression ($C = 0$) is equivalent to complete honesty, and the consistent signaling of twice actual aggression ($C = 100$) is the maximum level of bluffing permitted in this system.) Notice that although an individual's signaling strategy remains constant throughout its lifetime, the actual values of the aggressive signals produced by any one animat will vary across time. This is because each individual's aggression level changes dynamically as a function of its activity.

The development of different signaling strategies was placed under evolutionary control. Each animat has associated with it a bit-string genotype, specifying that individual's particular signaling strategy (the value of C). At the beginning of a run, a random population of genotypes is created. Hence, the initial distribution of signaling strategies in the population is also random. But the ongoing distribution of the different signaling strategies is a matter decided by the evolutionary selection pressures imposed by the environment. Only those individuals who prosper in energy terms will become strong enough to reproduce. The reproductive process copies the parent's genotype (specifying its particular signaling strategy) to the offspring. However, to allow the possibility of replication with modification, there is a small probability that a genetic mutation will take place during copying. (Throughout the experiments reported here, an 8 bit genotype was used, and the mutation rate was set to be a 0.05 chance that a bit-flip mutation will occur as each bit is copied during the reproductive event.)

One of the key selection pressures in this environment is provided by the signaling strategies adopted by the other members of the population. Thus fitness is frequency dependent with respect to the distribution of phenotypes. Moreover, this distribution will change over time. Deaths will result in strategies becoming extinct; births without mutation will result in an increased number of individuals using a particular strategy; and mutations will result both in novel (with respect to a particular run) strategies coming into existence, and in strategies which had previously died out being revived. The other significant selection pressure, which interacts with phenotypic frequency to determine the overall fitness of any individual is, of course, the cost of signaling. This cost (imposed as units of energy deducted per unit of aggression signaled) is set by the human experimenter at the start of each run, and remains constant for the duration of that run. Over evolutionary time, the general trend is for only the well-adapted strategies to survive and prosper.

In contrast with most examples of the use of evolutionary techniques in simulation of adaptive behavior research, the artificial evolution present in this eco-system features *no explicit fitness function*. There is no ecologically-unrealistic, externally-imposed evaluation function, against which the performance of an individual is judged. The only arbiter concerning the fitness of a particular signaling strategy is the ecological situation itself, an ecological situation complicated by the presence of interacting multiple agents, many of whom are operating with vastly different social strategies.

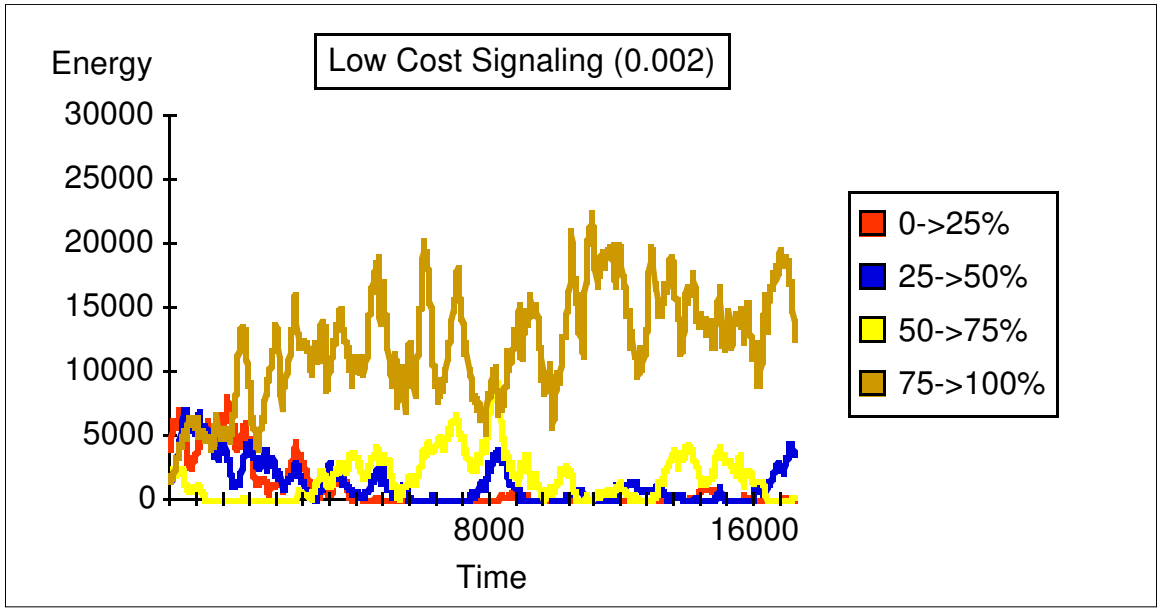

Figure 1: Low Cost Signaling — the total energy levels of the four sub-populations are plotted against time. See the text for a discussion of these results.

5 Results: Testing the Handicap Principle

In the experiments reported here, we did not assume that any members of the population were necessarily committed to what might be thought of as absolute honesty ($C = 0$). We merely let artificial evolution take its course, given some random point of departure. It was possible that the random initialization of the population of genotypes would result in one or more absolute truth-tellers, or that a genetic mutation would produce such an individual, but the probability was, of course, small. So the use of the continuum of strategies, as detailed above, meant that we needed to think in terms of relative honesty.

In order to analyse the effect of signaling-cost on relative honesty, we treated the total population as being made up of 4 sub-populations, each of which was identified according to a grouping in the signaling strategies deployed. The first group included all those individuals whose genetic specification determined that they produce signals indicating levels of aggression between 0% and 25% in excess of their actual aggression. So, in relative terms, this first group included the most 'honest' individuals. The second group included all those signaling 25% to 50% more than actual aggression, the third 50% to 75%, and the fourth 75% to 100%. Thus the fourth group were the most extreme bluffers. (There is, of course, something arbitrary in our choice of these particular divisions, but we felt that they would be sufficient to uncover the overall trend in the behavior.) A particular sub-population, at any one time, included all individuals adopting the appropriate strategy, including any offspring. We then needed a way to assess the success of the different strategies, at various values for the cost of signaling. To do this, we ran the simulation many times, setting various values for the relevant cost (units of energy deducted per unit of aggression signaled), and, during each run, we recorded the total energy present in each of the 4 sub-populations through time. Below we discuss two typical examples of the results obtained.

5.1 Experiment 1: Low Cost Signaling

In the first experiment described here, the cost of aggressive signaling was set to be low — 0.002 units of energy deducted per unit of aggression signaled. We allowed that there must be *some* cost in energy involved in transmitting a signal at all.

Figure 1 displays the total energy present in each of the 4 sub-populations over the first 16000 time steps of the low cost run. From these results it is clear that the group signaling between 75% and 100% in

excess of their actual aggression (the most extreme bluffers) are the most successful group; i.e., they are the most adaptively fit under these specific ecological conditions. In fact, after the initial settling down period (of approximately 1600 time-steps), the 75-100% group tend to dominate the world. Both the 0-25% group (the most honest contingent) and the 25-50% group struggle to survive in this eco-system. In particular, the 0-25% group quickly die out, and, despite their occasional reappearance due to fortuitous mutations, they never manage to maintain any foothold in the world. This is in spite of the fact that the 0-25% group was the best represented when the population was (randomly) created at the start of the run. There is one period (at around 8000 time steps) when the 50-75% group temporarily threatens the superiority of the most extreme bluffers. However, the 75-100% group soon recover, and regain the advantage. This challenge coincides with a temporary improvement in the fortunes of the 25-50% group, an improvement which is less substantial, and equally as ephemeral, as that experienced by the 50-75% group.

The most significant trend made evident through this analysis — that the 75-100% group (the extreme bluffers) are the most successful sub-population — has occurred in every one of the many runs at this low cost signaling that we have analysed so far. These results indicate that, in this simple eco-system, when there is a low cost to the signaling of aggressive intentions, the most successful (most adaptive) strategy is to produce aggressive signals that indicate levels of aggression well in excess of actual aggression. And, if we are allowed to hypothesize the presence of an ESS on the basis of empirical observation rather than formal mathematical analysis, then it appears that, for this ecological situation, extreme bluffing is an ESS. The failure of the low-bluffing strategies to re-establish themselves in the population, following their initial decline, implies that a population of high bluffers could not be invaded by an honest mutant. The results also suggest that honest signaling would not be an ESS, as a relatively honest population could be invaded by a higher-bluffing mutant.

5.2 Experiment 2: High Cost Signaling

In the second experiment described here, the cost of aggressive signaling was set to be much higher — 0.2 units of energy deducted per unit of aggression signaled. Figure 2 shows the results of one representative run at this cost. Again the graph shows the total energy present in each of the 4 sub-populations, during the first 16000 time steps.

In this particular high cost run, two of the four identified sub-populations co-exist alongside each other in the eco-system. These are the 0-25% group and the 25-50% group; i.e., the 2 groups who bluff at lower levels. For most of the run, the relative positions of the two relevant plots indicate that the 0-25% group is marginally more successful than the 25-50% group, although the 25-50% group enjoys occasional superiority (e.g., the period around time step 12000). Notice that the competitive nature of the eco-system (resulting from the capped resource level) means that when one of these two groups undergoes a period of unusual prosperity, its success tends to be at the expense of the other group. Neither of the high-bluff strategies is able to compete for any length of time. The 75-100% group is moderately successful for around the first 4000 time steps, but then its fortunes decline, and the group dies out at around time step 7000. The 50-75% group dies out very quickly. The two high-bluffing groups are occasionally resurrected through mutations, but neither experiences any period of sustained resurgence.

In all the many runs featuring this high cost of signaling, which we have analysed so far, the two groups adopting the lower-bluff strategies tended to be overwhelmingly dominant, when compared with the groups of higher-bluffers. And in general, the pattern of relative adaptive success between the two lower-bluff groups is also repeated; i.e., the most honest sub-population (the 0-25% group) tends to enjoy some degree of superiority — sometimes more so than in the run shown. So all the indications are that, in this simple eco-system, when there is a high cost to signaling, it is no longer beneficial to bluff excessively, because the energy cost incurred through such behavior is prohibitive. For this ecological situation, relative honesty appears to have the general character of an ESS; high-bluffing sub-populations cannot re-establish themselves when competing with populations made up of lower-bluffing individuals. Also, it appears that high-bluffing would not be an ESS, as a high-bluffing population could be invaded

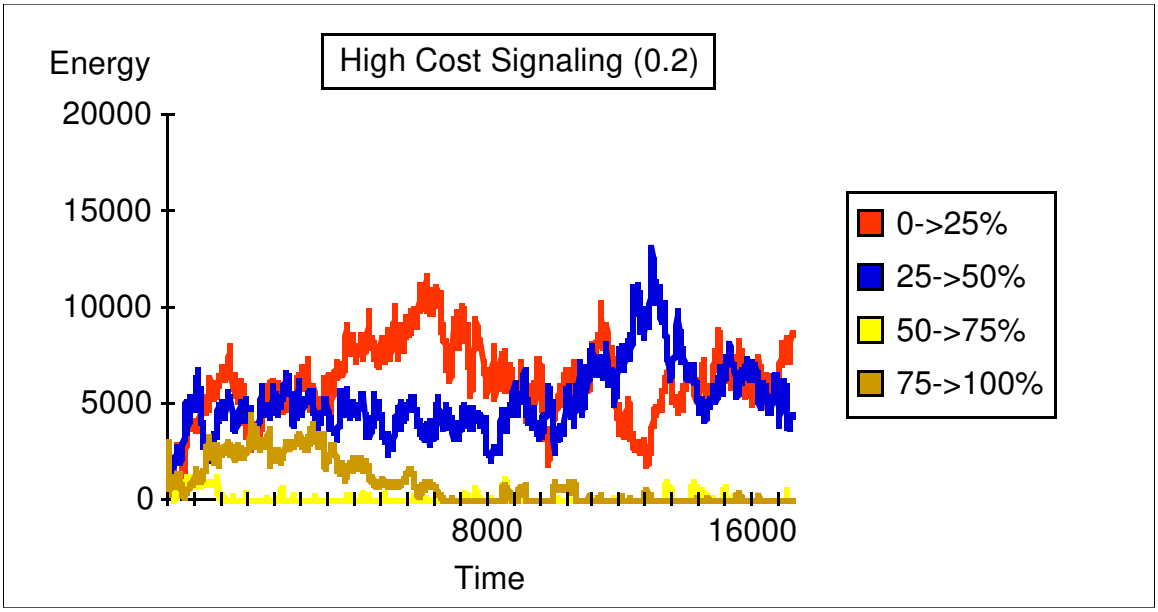

Figure 2: High Cost Signaling — the total energy levels of the four sub-populations are plotted against time. The general level of energy present in the various sub-populations is, of course, less than in the low cost example, precisely because the overall cost of signaling, in energy terms, is higher.

by a more honest mutant.

6 Conclusions

Thus, on the evidence from this synthetic ecology, the fundamental logic of the handicap principle does transfer to multi-agent signaling systems. When the cost of signaling is low, the extreme bluffers take over the world. When the cost of signaling is high, the more honest individuals are the dominant section of the population. So our results support the general hypothesis that a high cost to signaling will result in increased honesty. Moreover, the mechanisms though which relative honesty is enforced in our simulation are the same as those described by Zahavi.

References

Cliff, D., and Bullock, S. (1993). Adding 'foveal' vision to Wilson's animat. *Adaptive Behavior*, 2(1), 49-72.

Clutton-Brock, T., and Albon, S. (1979). The roaring of red deer and the evolution of honest advertisement. *Behaviour*, 69, 145-170.

Davies, N. B., and Houston, A. I. (1984). Territory economics. In Krebs, J. R., and Davies, N. B. (Eds.), *Behavioural Ecology - an Evolutionary Approach* (2nd edition)., chap. 6, pp.148-169. Blackwell Scientific, Oxford.

de Bourcier, P., and Wheeler, M. (1994). Signalling and territorial aggression: An investigation by means of synthetic behavioural ecology. In Cliff, D., Husbands, P., Meyer, J.-A., and Wilson, S. W. (Eds.), *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*, pp.463-472 Cambridge, Massachusetts. MIT Press / Bradford Books.

Maynard Smith, J., and Harper, D. G. C. (1988). The evolution of aggression. *Philosophical Transactions of the Royal Society: Biological Sciences*, 319, 557-570.

Stamp Dawkins, M. (1993). Are there general principles of signal design?. *Philosophical Transactions of the Royal Society: Biological Sciences*, 340, 251-255.

Wheeler, M., and de Bourcier, P. (1994). How not to murder your neighbor: Using synthetic behavioral ecology to study aggressive signaling. Submitted to the journal *Adaptive Behavior*.

Zahavi, A. (1975). Mate selection - a selection for a handicap. *Journal of Theoretical Biology*, 53, 205-214.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Dynamic Fitness Landscapes

Seth Bullock
sethb@cogs.sussex.ac.uk

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

Abstract : Genetic Algorithms (GAs) are typically thought to work on static fitness landscapes. In contrast, natural evolution works on fitness landscapes that change over evolutionary time as a result of co-evolution. Sexual selection and predator-prey evolution are examined as clear examples of phenomena that transform fitness landscapes. The concept of co-evolution is subsequently defined, before attempts to utilise co-evolution in the use of GAs as design tools are reviewed and speculations concerning future applications of automatic co-evolutionary techniques for design are considered.

1 Genetic Algorithms

Genetic Algorithms (GAs) are a design/optimisation technique inspired by natural evolution (Goldberg, 1989; Holland, 1975). The bare essentials of evolutionary theory (selection, reproduction, variation, fitness, etc.) are extracted and applied to artificial genetic material in an attempt to evolve solutions to problems.

A genetic algorithm works on a group of potential solutions to a problem, termed a population. Each solution takes the form of a string (chromosome) of letters (genes) from an alphabet (typically consisting of the binary digits $\{0,1\}$). The algorithm first assesses these 'genotypes' allocating a fitness value to each, dependent upon the degree to which its associated 'phenotype' solves the problem (e.g. how well the parameters specified by a chromosome characterise a useful bridge, or pack a lorry, or optimize a function, etc.).

Once this has been carried out, a new population of genotypes can be created by 'breeding' the current population. 'Parent' chromosomes are chosen with some bias towards those that are most fit. Both parents contribute some 'genetic material' to their 'offspring'. This is typically achieved through some form of cross-over operator which takes two chromosomes and produces two more by swapping a randomly chosen portion of the genome from each parent. During reproduction there is a small chance that mutation may occur resulting in a less than perfect copy of genetic material. The role of mutation in the reproduction process is to introduce some random variation to the population. The cycle then repeats as the offspring generation is assessed, and bred, etc.

Over evolutionary time populations will (hopefully) 'converge' on solutions to problems, many of which may be counter-intuitive to the designer. Convergence is said to have occurred in a population if the genes at each locus reach some criterion of uniformity across the population.

Just how the discovery of solutions is achieved is controversial, but essentially the search space (defined by the manner in which the problem is parameterized) is tested in a parallel manner which takes advantage of beneficial strings of consecutive genes (schemata) by tending to group them together to form potential solutions.

2 Landscapes

Optimisation techniques are often thought of as traversing landscapes. A potential solution is represented as a point in such a landscape, the height of which corresponds to its 'fitness' – the extent to which it solves the problem. Fitness can either increase with height or increase with depth. For the remainder of this paper I will adopt the former convention.

If an optimisation technique tends to move from one potential solution to another which is slightly higher then it is a good technique and will tend to reach maxima in the fitness landscape. These solutions correspond to the tops of hills, points at which any small change to the solution (changing the length of one of the struts in a bridge, rotating a box slightly in a lorry, increasing one coefficient in an equation by a small amount, etc.) will result in a poorer solution (somewhere on the hillside). For any landscape there may be a number of local maxima and will be at least one global maximum. Local maxima are hill tops that although better than their surroundings are not the highest hill tops in the landscape. They are good solutions but not the best. Global maxima are the highest ground in the landscape. They are the best possible solution to the problem.

As was mentioned in the previous section, genetic algorithms typically work with a population of solutions scattered across the fitness landscape which gradually converge on one of the maxima. The fitness landscape is fixed by the designer of the GA when she decides how she will assess the potential solutions in the population. For example, if evolving a bridge design, the GA designer may specify fitness as being a function of some measure of safety, a measure of traffic capacity, and a measure of cost. As such a function does not change throughout the evolution process, the fitness landscape can be regarded as static, i.e. a potential solution with fitness x will always have fitness x independent of its peers or the passing of time. Indeed almost all optimisation techniques traverse static fitness landscapes.

In contrast, natural evolution works on a dynamic fitness landscape. Over evolutionary time the fitness of a phenotype (solution) may change radically. What was a winning strategy (e.g. eating flora of type A) becomes 'out of date' as conspecifics, predators, resources, etc. change through their own evolution. The resulting *co-evolution*, the evolution of systems in response to each other, can be thought of as ensuring that an organism's evolutionary goal-posts (maxima in the fitness landscape) move. Rather than work towards the solution of some fixed problem, organisms are constantly adapting (over evolutionary time) to each other, their surroundings, etc. which are themselves adapting in response.

As an explicatory exercise, two examples of such co-evolution will be described before a more general characterisation of the phenomena at issue is attempted.

3 Sexual Selection and Predator-Prey Evolution

One of the oldest and most enduring problems in the study of animal behaviour is that of the genesis of the plethora of mating displays, colourful feathers, complex calls and songs, etc. that animals use to attract the opposite sex. With the advent of Darwin's *On the Origin of Species* the beginnings of a solution were formulated. Darwin proposed that life-forms *evolve* through a process involving reproduction, inheritance, variation and selection. He considered the struggle for survival to be the primary selective pressure. Organisms slightly better equipped to deal with the trials of life will leave a greater number of offspring than their less well equipped peers. Over many generations populations will come to be composed of individual organisms that are well adapted to the dangers and resources of their respective niches.

However, this 'natural' selection cannot account for the myriad of impressive and complex sexual displays that are possessed throughout the animal kingdom. The peacock's tail, the bower bird's love-nest, the frog's croak, and the stickleback's dance do not contribute to their respective chances of survival. If anything, such behaviours and ornaments actively detract from an organism's chances of survival through their costs in terms of energy and time (both of which could have been spent foraging or in some other useful activity), and in terms of becoming more likely to expire through predation (beautiful feathers and sonorous croaks not only attract mates but hungry predators too).

In order to explain the existence of such characteristics, Darwin proposed a separate selective pres-

sure. Sexual selection can be considered as a corollary of the fact that survival is not the only requisite of an evolutionarily successful creature. In order to survive over evolutionary time one must reproduce. The evolution of traits which, although not beneficial in terms of natural selection (they do not increases the organism's chances of survival), improve an organism's reproductive chances can be accounted for by appealing to the notion that organisms slightly better equipped to deal with the trials of mate choice and reproduction will, again, leave a greater number of offspring than their less well equipped peers.

Darwin divided the mechanism of sexual selection into two categories, *male contests* and *female choice*. The former category contains behaviour and morphological traits that enable males to beat other males at the mating game. For example, the antlers of red deer, and the tusks of the bull walrus have evolved to serve the function of enabling a male deer or walrus to beat competitors in fights which determine a sexual 'pecking order'. Many of these traits (e.g. size, strength, fighting prowess, etc.) contribute to a general survival ability as well as specifically to reproductive success. However, male-male competition can develop in areas that do not aid an individual creature's survival.

Sperm competition is one such example. In species utilising this practice (e.g. dragonflies) females typically store the sperm of their mates for a time before their eggs are fertilised. Copulating males will attempt to expel the sperm of their mate's previous suitors through a variety of penis specialisations. To combat such expulsion techniques, methods of ensuring the retention of sperm, including the use of genital cement, co-evolve.

Female choice may, at first glance, seem less bizarre, relying as it does on the understandable notion that it pays to be picky. In most species, females put more effort into producing young than their mates[1]. They provide the vast majority of the time and energy necessary to produce a viable offspring. Given that this is the case, females that mate with poor stock are wasting valuable resources. Females with an ability to distinguish between poor and high quality mates will enjoy an advantage over their less perceptive conspecifics in terms of the quality and number of offspring produced.

Although this principal seems sound, how can it result in the preferences that we see in the animal kingdom? Why should pea-hens prefer peacocks with stunning plumage displays? Why should female frogs fall for males with loud, deep croaks? Two attempts to answer this style of question will be detailed below.

Fisher (1915) considered the case in which an infrequently occurring trait is favoured by natural selection. Females mating with males who display this trait produce fitter offspring. Thus the females with preferences for the male trait will proliferate, as will the males which possess the trait. However, offspring inheriting such a trait are not merely better equipped for survival, but are also better equipped for the task of securing a mate. Fisher showed that under certain conditions a "runaway process" could result in the latter benefit overpowering the former, resulting in males with traits that actively detract from their survival chances, and females with preferences for such traits.

In contrast, Zahavi (1975) proposed that rather than the reduction in an organism's chances of survival being an unfortunate result of runaway sexual selection, females might actually benefit from mating with "handicapped" males. He claimed, and was subsequently supported by mathematical proofs (Grafen, 1990) that showed, that handicaps (e.g. the peacock's tail) which increase an organism's chances of perishing due to predation or starvation are honest signals of its quality.

His logic revolves around the observation that costly displays can only be made by those able to afford them. In this context, displays that are costly in terms of fitness (they reduce ones reproductive chances) can only be made by those organisms with a high enough fitness to afford them. Sexual selection will thus result in loud, energetic, opulent, costly, extravagance.

Space does not permit a fuller coverage of the issues raised by these theories and the interested reader is referred to Gould and Gould (1989) and Krebs and Davies (1993) for more thorough treatments of the topic. What remains pertinent to this essay is that, as in the male contests described above, co-evolution

[1] Sea-horses are a notable exception. The female deposits her eggs in her mate's pouch, the eggs are fertilised, and gestate within this sac until the male 'gives birth' to them. This role-reversal also affects their mating ritual. Male sea-horses are coy and shy, whilst females must actively win their affection by displaying.

drives the female's mate preferences and the associated male traits, and in many cases drives them in such a way as to create organisms which have deviated from their strictly survival oriented ancestors in an attempt to satisfy the constantly changing demands of sexual selection.

Sexual selection is by no means the only example of such co-evolution. Predator-prey evolutionary dynamics also exhibit what behavioural ecologists have termed 'evolutionary arms races'. The development of higher acuity in a predator may be countered by the evolution of camouflage in a prey, teeth and claws provoke carapaces and scales, toxins demand antidotes, etc., etc.

Such arms races result in highly developed behavioural skills and complicated morphology. Such complexity is the result of the increasing demands placed on organisms by their environment (including their conspecifics, predators, prey, etc.). The hunting skill and speed of the peregrine falcon, for example, could not have evolved without the concurrent evolution of the perceptual capacity and escape capabilities of its prey.

Does any kind of co-adaptation qualify as co-evolution? Daniel Janzen (1980) distinguishes between true co-evolution and what he terms "diffuse" co-evolution. He defines the former as specific, reciprocal, evolutionary change, i.e. continued evolutionary change in the trait of one population in response to the continued, reciprocal, evolutionary change of a trait possessed by another population. In contrast, diffuse co-evolution is non-specific, reciprocal, evolutionary change, in which the trait of one population changes over evolutionary time in response to a *group* of traits possessed by another population (which may contain several species).

For example, the evolution of egg-mimicry and egg-discrimination in species of bird that respectively perpetrate and suffer the depositing of eggs in foreign nests is an example of true co-evolution (which is common in brood-parasitism and host-parasitism) in that the traits have evolved specifically for the purpose of brood-parasitism and defeating brood-parasitism respectively. Conversely, the hard shells of many crustaceans have evolved in response to a general threat from predators with a variety of body crushing/piercing techniques and are thus examples of diffuse co-evolution (See Krebs & Davies, 1991, ch.6 for further examples).

In fact, under Janzen's definitions, many instances of co-adapted predator-prey traits *cannot* be classed as the product of co-evolution. For example, the ultrasonic sound detectors in lacewing moths, which are a specific counter-adaptation to the sonar hunting technique developed by their bat predators, may be the product of a one-way adaptation on the part of the moths with no reciprocal evolutionary change in the bat predation mechanism. If this is the case then the co-adapted traits cannot be termed co-evolved.

Armed with such notions of co-evolution we can proceed to examine the prospects of attempting to apply co-evolutionary techniques to the use of genetic algorithms as design tools.

4 Co-evolutionary Design

A fundamental problem for the designer of genetic algorithms is specifying the problem that is to be solved in a manner that allows incremental steps towards a solution to be rewarded. If a problem is not specified in such a manner the genetic algorithm will have no feedback with which to drive its search and will essentially perform randomly until it finds a solution that can be rewarded. Co-evolution circumvents such problems by automatically moving the GA's evolutionary 'goal-posts', gradually changing the problem as the population moves over a dynamically changing fitness landscape.

What are the prospects for such an automatic co-evolutionary approach? Initial work in this area is thin on the ground. Simulations of undirected co-evolution have been undertaken (e.g. Werner & Dyer, 1991), but have little relevance here as they typically seek neither to explicate co-evolution nor utilise co-evolution in the solution of some design task.

The incremental approach of the Evolutionary Robotics Research Group at the University of Sussex can be seen as a first attempt to use co-evolution in the design of autonomous agents (Harvey, Husbands, & Cliff, 1994). The agents involved initially face a simple sensory-motor problem, which is incrementally made more difficult in an effort to coax complex behaviour from systems which could not be evolved

from scratch. Such scaffolding techniques are reminiscent of the parent-child interactions which facilitate infant development (Rutkowska, 1994).

However, the hand-cranked nature of such scaffolding requires the presence of a human designer 'in the loop' and, potentially, the tasks of specifying the incremental goals that allow evolution to reach solutions to complex problems could itself become as problematic as designing the agents manually.

First attempts at utilising automatic co-evolutionary design includes work by David Hillis (1990) and Phil Robbins (1994), in which parasites are used to increase the performance of artificial agents, and Phil Husbands (1993) at the University of Sussex, in which the co-evolution of shop-floor schedules was explored. Such work, however, is in its infancy.

Before the full potential of co-evolutionary design techniques can be realised, the burgeoning body of work exploring artificial co-evolution must be consolidated. At Sussex, studies of predator-prey co-evolution (Miller & Cliff, 1994), sexual selection (Miller, 1994), and parental imprinting (Todd & Miller, 1993), have already been carried out and further research seems both worthwhile and inevitable. Open questions, such as the paucity of true co-evolution in natural predator-prey ecologies, in comparison to the relative abundance of such evolutionary dynamics in parasitic relationships, seem amenable to investigation through the artificial means employed within this style of research. The possibility of fruitful collaboration between the simulation of artificial co-evolution and the study of naturally occurring co-evolution seems to be a set of goal-posts worth shooting for, and one that will not be moving in the foreseeable future.

References

Fisher, R. A. (1915). The evolution of sexual preference. *Eugen. Rev.*, *7*(184 - 192).

Goldberg, D. E. (1989). *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley.

Gould, J. L., & Gould, C. G. (1989). *Sexual Selection.* Scientific American Library.

Grafen, A. (1990). Biological signals as handicaps. *Journal of Theoretical Biology*, *144*, 517 – 546.

Harvey, I., Husbands, P., & Cliff, D. (1994). Seeing the light: Artificial evolution, real vision. In Cliff, D., Husbands, P., Meyer, J.-A., & Wilson, S. W. (Eds.), *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behaviour*. A Bradford Book; MIT Press.

Hillis, D. (1990). Co-evolving parasites improve simulated evolution as an optimisation procedure. *Physica D*, *42*, 228 – 234.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. MIT Press.

Husbands, P. (1993). An ecosystems model for integrated production planning. *International Journal of Computer Integrated Manufacturing*, *6*(1 & 2), 74 – 86.

Janzen, D. H. (1980). When is it coevolution?. *Evolution*, *34*(3), 611 – 612.

Krebs, J., & Davies (1993). *An Introduction to Behavioural Ecology* (3rd edition). Blackwell Scientific.

Krebs, J. R., & Davies, N. B. (Eds.). (1991). *Behavioural Ecology - An Evolutionary Apporach* (3rd edition). Blackwell Scientific.

Miller, G. F. (1994). Exploiting mate choice in evolutionary computation: Sexual selection as a process of search, optimization, and diversification. In Fogarty, T. C. (Ed.), *Evolutionary Computing: Proceedings of the 1994 Artificial Intelligence and Simulation of Behaviour (AISB) Society Workshop*, pp. 65 – 79. Springer-Verlag.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Real Time for Real Power: Methods of evolving hardware to control autonomous mobile robots.

Adrian Thompson
adrianth@cogs.susx.ac.uk

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

Abstract It is possible to use artificial evolution automatically to "design" electronic circuits to control a mobile robot. *Evolvable hardware* allows each new variant circuit to be physically tried out as a real piece of hardware when its performance needs to be measured. This means that evolution can exploit many of the properties of the hardware which a human designer could not. There is also the potential to accelerate the evolutionary process by the use of a high-speed simulation of the hardware control system's environment. It is argued here that a circuit produced by an accelerated evolutionary process can not use the properties of the hardware as effectively as one evolved in real time. The methods of accelerated and real-time evolution will both have their place, but it is important to distinguish between the merits of each.

1 Introduction

In recent years, the case for evolvable hardware (electronic circuits which are directly manipulated by artificial evolution) has begun to be made [8, 2, 6, 14, 7]. This paper re-examines the methods of evolving hardware for the specific task of controlling an autonomous mobile robot. I reason that there are two distinct ways of using the power of evolvable hardware: either evolve real (not simulated) circuits at high speed in a simulated environment or evolve real circuits more slowly in the real world (which turns out to allow better exploitation of the hardware). In both cases, it is important to abandon design constraints which are merely for the benefit of a human designer, such as the enforcement of modularisation and synchronisation. This frees evolution to produce robot control structures which are both spatially (in terms of the topology and physical layout of the components and their interconnections) and temporally (in terms of the circuit's dynamics) too difficult for humans to design. Designing these control architectures might be difficult not only due to their complexity, but also because of their use of detailed characteristics of the hardware; characteristics which the designer either does not know about, or would be unable to put to use even if the information were available.

Firstly, though, what *is* evolvable hardware? When artificially evolving the control system of an autonomous mobile robot, each of a "population" of many different control systems is evaluated for its performance in mediating some desired robot behaviours to give a measure of "fitness." Control systems which cause poor behaviour are removed from the population, and replaced by the "offspring" of controllers which give better results. Fitter control systems are "bred" to produce more offspring than less fit controllers. The breeding process operates not on the controllers themselves, but on their *genotypes* — abstract descriptions of how the controllers are constructed. Usually, the genotypes of each of two parent control systems are combined in some way to form offspring genotypes which may have parts from each parent. Small stochastic changes (mutations) are also made to the genotypes during breeding. The control systems arising from the offspring genotypes are then evaluated to determine the desirability of the robot

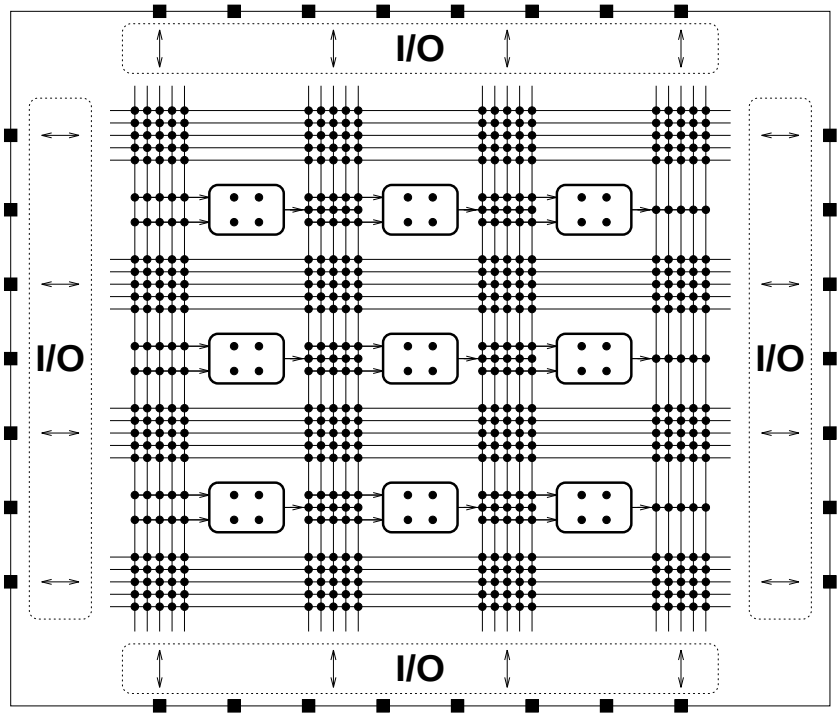

Figure 1: Part of an FPGA architecture.

behaviour they induce, and the process continues. The hope is that, starting from an initial population (often randomly generated), the evolving control systems cause the robot to get better at its predetermined job over time, finally reaching acceptable performance [9, 3, 1].

Evolvable Hardware (EHW) is a piece of electronics which can be automatically physically reconfigured to instantiate each individual control system in the changing population in turn. Hence, each controller can be evaluated as an electronic circuit in the real world, not as a simulation of some other type of system — not even as a simulation of an electronic circuit. Consequently, an implemented system is evolved directly; the constraints imposed by the hardware are satisfied automatically, and all of its characteristics can be brought to bear on the control problem. There is now the opportunity to evolve circuits as parallel distributed processing (PDP) systems in their own right, and which are tailored to exploit the exact internal physical properties of a chip[1] (which a designer could never use or even know about).

What I have just described is sometimes known as "intrinsic" EHW to distinguish it from hardware which can be reconfigured only a few times — "extrinsic" EHW.[2] In the extrinsic case, a software simulation of the hardware is used to evolve a configuration which, once providing satisfactory performance in the simulation, is applied to the hardware in a separate implementation step. Many of the advantages of EHW are lost by this approach, but it does have the benefit of being easy to achieve with 1994 off-the-shelf components. Very soon, however, the availability of devices suitable for intrinsic EHW will no longer be a problem, so this paper deals only with the the highly superior case of intrinsic EHW.

How can *hard*ware be reconfigurable at all? One answer is to have some aspects of the behaviour of the components, as well as the interconnections between them, determined by the values held in random-access memory (RAM) cells. The values held in these cells determine the configuration and hence the behaviour of the system, and can readily be changed under software or hardware control. A silicon chip which can be built along these lines is shown in Figure 1 — a Field Programmable Gate Array (FPGA).

[1]There is a danger that the circuit could come to rely on the properties of one particular device, and not work when placed on another which is nominally identical. It may be necessary to find ways of avoiding this; on the other hand, the effect could be useful in coping with defects on a particular chip or wafer.

[2]The terms "intrinsic" and "extrinsic" EHW are due to Hugo de Garis.

The diagram is of a simple imaginary architecture, but its structure is pertinent to real devices [17].

FPGAs typically consist of an array of many hundreds of reconfigurable blocks, shown here as boxes containing four dots. Each dot in this diagram represents one bit of RAM; for example, if these blocks are capable of computing an arbitrary boolean function of two inputs, then a block's behaviour can be determined by the contents of four bits of configuration RAM controlling it (there are 2^4 boolean functions of two inputs). Often, the blocks contain more functionality, such as multiple boolean functions and flip-flops (allowing clocked systems to be evolved). Inputs and outputs of the blocks can be connected to wires passing next to them; which wire is connected to which block is again controlled by RAM bits. The connections between wires are also controlled in this way (in fact, a hierarchy of wiring resources of different lengths is sometimes provided). The signals near the edges of the array are interfaced by special reconfigurable input/output (I/O) blocks to the pins of the chip.

Even from this simplified overview, we can see that the configuration RAM bits determine what circuit is physically present on the chip, because they control transistor switches which determine what is connected to what, and how blocks behave. If the content of the configuration RAM is placed under the control of an evolutionary algorithm, then the circuits determined by each new set of configuration bits it generates can be evaluated in the real world as real physical circuits on the chip. RAM-based digital FPGAs are not the only way that reconfigurable hardware can be built, but are probably going to be the dominant way over the next few years.

2 Accelerated Evolution

When artificially evolving robot controllers, the time taken before a satisfactory control system is obtained is usually dominated by the amount of time it takes to evaluate each individual's fitness. The time taken by the genetic operations (breeding) and the expression of the genotypes (to produce the actual control structures) is small in comparison.

The obvious way to evaluate a control system is to put it in the robot and see what the robot does in the real world, in real time. This is a slow process: for a basic task it might take two minutes of behaviour in the real world to evaluate a control system. With a population size of 30, then 30 generations would take more than a day, and this is for the very simplest of tasks — the problem will only get worse.[3] Methods of speeding up fitness evaluation (and thus the whole evolutionary process) are therefore of great importance.

In order to be evaluated in less than real time, the evolving hardware control system needs to be interfaced to a high-speed simulation of the robot and its interactions with the world. It is suggested by de Garis[2] that the environment simulation could be implemented in special purpose hardware situated next to the evolving hardware control system on a VLSI chip. Implementing the environmental simulator in hardware rather than software makes it faster, but does not solve the problem that it is extremely difficult adequately to simulate the interactions between a control system and its environment, such that a control system evolved for the simulated world behaves in a satisfactory way in the real world. This is especially the case when vision is involved [1]. Nevertheless, it is likely that environment simulation in special purpose hardware will be an important tool in evolutionary robotics as new simulation techniques are developed [10].

The point of this section is that, even if it is possible to evolve hardware in a high-speed simulation of the environment, there is a cost to be payed in terms of the efficiency of the evolved circuit. When a circuit which evolved for behaviour in the high-speed simulated world is ready for use in the *real* world, all of its dynamics must be slowed down by the same factor by which the real world is slower than the simulation. (Imagine a controller which was evolved for a high-speed simulated world and was then let loose in the real world without being slowed down. Everything in the environment would then be happening slower than it "expected," and the motor signals produced would tend to be too fast for the robot body and the

[3] Evolution times should be compared to the time for humans to design an equivalent system. Evolution times of a few years will be acceptable if the final product would take at least as long to be designed by humans, or if it could not be designed at all. Costs should also enter the comparison.

world. It would probably not work.) Hence, an EHW robot controller which was evolved for a high-speed simulated environment, and then slowed down to operate in the real world, will not be making maximal use of the available hardware. This is because it is capable of producing the same behaviour in a world which is running faster, and the resources needed to allow this could be being used for real-world performance.

In many cases, because of the huge power available from hardware, it will be sensible to trade off the efficiency of the circuit produced in favour of accelerated evolution, and the high-speed environment simulation approach may be chosen. However, it is the main point of this paper that there is another option: it is possible to increase the exploitation of the hardware's resources by evolving circuits in real time. That is the subject of the next section.

3 Real Time for Real Power

Work with the "gantry robot" at the University of Sussex [5] suggests that it may be feasible to carry out evolution in a real-world environment with fitness evaluations taking real time. This immediately avoids the problem identified in the previous section — no part of the hardware ever has to perform faster than it does when the circuit is controlling the real robot in the real world. *All* of the capabilities of the hardware can therefore be applied to the real-world control task (none need to be held in reserve to allow the circuit to be run faster to match a high-speed environment simulation).

In fact, the implications are greater than this. We have seen that for accelerated evolution, the dynamics of the evolving hardware robot control circuit must be speeded up and then slowed down again, but we did not say *how*. In practice, it will almost certainly be through clocking. If only clocked circuits can be developed by accelerated evolution, then this is a very tangible limitation to the technique. There is no such limitation on evolution carried out in real time.[4]

Why might unclocked circuits be better than clocked ones? Firstly, notice that a clocked digital system is a finite-state machine, whereas an unclocked asynchronous digital system is not. To describe the state of an unclocked circuit, the temporal relationships between its parts must be included. These are continuously variable analogue quantities, so the machine is not finite-state. This theoretical point gives a clue to a practical advantage: in an unclocked digital system, it is possible to perform analogue operations using the time dimension, even when the logic gates assume only binary values (see for example, the *pulse stream* technique [15, 16]).

For some types of system (perhaps including recurrent logic networks), clocking can be an unnecessary restraint on the circuit's behaviour, imposed to ease design by humans. In contrast, there are also situations in which clocking expands the useful dynamics of the system by providing easy control of the time-scales involved. I am not arguing that clocking is always bad, just that it is not always good.

In considering the design by humans of asynchronous circuits, Gopalakrishnan and Akella [4] note that asynchronous systems are best for certain kinds of applications: "Synchronous systems are simply too unnatural for such applications. Synchronous clocking works best when many decisions can be made statically (i.e. before execution time)." This condition is certainly not met when evolving the controller of a mobile robot. In addition, they note that, "Synchronous systems are hard to expand or modify incrementally. …In comparison, asynchronous circuits permit such incremental expansion." This suggests that asynchronous circuits may be *more easily evolvable* than clocked ones. They continue: "In asynchronous systems, the completion of an activity can immediately (i.e. without having to wait for the next clock 'tick' to arrive) initiate any activity that can logically follow. This can give very fast execution times in the average case." Perhaps, then, asynchronous circuits can make more efficient use of the hardware than clocked circuits.

It is also observed in [4] that asynchronous circuits minimise clock and power distribution problems and that: "Many people see an inherent robustness in asynchronous circuits. They have been investigating the widely believed 'self testing' nature of asynchronous circuits, or their low power consumption, or high

[4]For convenience, the environment could still be simulated, but it must be a real-time simulation.

immunity to voltage and temperature fluctuations." All of these points indicate that it is more appropriate to evolve asynchronous circuits for robot control than clocked ones. In fact, the reason that asynchronous circuits are little used is because of their difficulty of design — artificial evolution may be the answer not only in robotics, but in other areas of electronics too.

The evolution of circuits (either clocked or asynchronous) in real time allows maximal exploitation and accommodation of the natural temporal behaviour of the implementation (its physics). In addition, *asynchronous* circuits evolved in real time can put to use dynamics which would be considered as transients between clock-ticks in a clocked system. Circuits evolved in real time could be worth waiting for.

4 Conclusion

I have argued that when intrinsically evolving hardware to control a robot, the use of high-speed environment simulation to accelerate the evolutionary process imposes limitations on the nature of the circuit evolved. These limitations are not present when the circuits are allowed to behave in real time during their fitness evaluations, allowing superior circuits to be produced. Both of these methods will have their place in the growing field of evolvable hardware.

References

[1] Dave Cliff, Inman Harvey, and Phil Husbands. Explorations in evolutionary robotics. *Adaptive Behaviour*, 2(1):73–110, 1993.

[2] Hugo de Garis. Evolvable hardware: Genetic programming of a Darwin Machine. In C.R. Reeves R.F. Albrecht and N.C. Steele, editors, *Artificial Neural Nets and Genetic Algorithms - Proceedings of the International Conference in Innsbruck, Austria*, pages 441–449. Springer-Verlag, 1993.

[3] David E. Goldberg. *Genetic Algorithms in Search, Optimisation & Machine Learning*. Addison Wesley, 1989.

[4] Ganesh Gopalakrishnan and Venkatesh Akella. VLSI asynchronous systems: specification and synthesis. *Microprocessors and Microsystems*, 16(10):517–526, 1992.

[5] Inman Harvey, Phil Husbands, and Dave Cliff. Seeing the light : Artificial evolution, real vision. In Dave Cliff, Philip Husbands, Jean-Arcady Meyer, and Stewart W. Wilson, editors, *From animals to animats 3: Proceedings of the third international conference on simulation of adaptive behaviour*, pages 392–401. MIT Press, 1994.

[6] Hitoshi Hemmi, Jun'ichi Mizoguchi, and Katsunori Shimohara. Development and evolution of hardware behaviours. In Rodney Brooks and Pattie Maes, editors, *Artificial Life IV*, pages 317–376. MIT Press, 1994.

[7] Tetsuya Higuchi, Hitoshi Iba, and Bernard Manderick. *Massively Parallel Artificial Intelligence*, chapter "Evolvable Hardware", pages 195–217. MIT Press, 1994. Edited by Hiroaki Kitano.

[8] Tetsuya Higuchi, Tatsuya Niwa, Toshio Tanaka, Hitoshi Iba, Hugo de Garis, and Tatsumi Furuya. Evolving hardware with genetic learning: A first step towards building a Darwin Machine. In *Proceedings of the 2nd Int. Conf. on the Simulation of Adaptive Behaviour (SAB92)*. MIT Press, 1993.

[9] J. H. Holland. *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press, 1975.

[10] N. Jakobi. Evolving sensorimotor control architectures in simulation for a real robot. Master's thesis, School of Cognitive and Computing Sciences, University of Sussex, 1994.

[11] D. Mange. Wetware as a bridge between computer engineering and biology. In *Proceedings of the 2nd European Conference on Artificial Life (ECAL93)*, pages 658–667, Brussels, May 24-26 1993.

[12] Daniel Mange and André Stauffer. *Artificial Life and Virtual Reality*, chapter "Introduction to Embryonics: Towards new self-repairing and self-reproducing hardware based on biological-like properties", pages 61–72. John Wiley, Chichester, England, 1994.

[13] P. Marchal, C. Piguet, D. Mange, A. Stauffer, and S. Durand. Achieving von Neumann's dream: Artificial life on silicon. In *Proc. of the IEEE International Conference on Neural Networks, icNN'94*, volume IV, pages 2321–2326, 1994.

[14] Jun'ichi Mizoguchi, Hitoshi Hemmi, and Katsunori Shimohara. Production genetic algorithms for automated hardware design through an evolutionary process. In *IEEE Conference on Evolutionary Computation*, 1994.

[15] A. F. Murray et al. Pulsed silicon neural networks - following the biological leader. In Ramacher and Rückert, editors, *VLSI Design of Neural Networks*, pages 103–123. Kluwer Academic Publishers, 1991.

[16] Alan F. Murray. Analogue neural VLSI: Issues, trends and pulses. *Artificial Neural Networks*, (2):35–43, 1992.

[17] Trevor A. York. Survey of field programmable logic devices. *Microprocessors and Microsystems*, 17(7):371–381, 1993.

Miller, G. F., & Cliff, D. (1994). Protean behavior in dynamic games: Arguments for the co-evolution of pursuit-evasion tactics. In Cliff, D., Husbands, P., Meyer, J.-A., & Wilson, S. W. (Eds.), *From Animals to Animats 3: Proceedings of the Third International Conference on the Simulation of Adaptive Behaviour*, pp. 411 – 420. MIT Press.

Robbins, P. (1994). The effect of parasitism on the evolution of a communication protocol in an artificial life simulation. In Cliff, D., Husbands, P., Meyer, J.-A., & Wilson, S. W. (Eds.), *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behaviour*. A Bradford Book; MIT Press.

Rutkowska, J. (1994). Emergent functionality in human infants. In Cliff, D., Husbands, P., Meyer, J.-A., & Wilson, S. W. (Eds.), *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behaviour*. A Bradford Book; MIT Press.

Todd, P. M., & Miller, G. F. (1993). Parental guidance suggested: How parental imprinting evolves through sexual selection as an adaptive learning mechanism. *Adaptive Behavior*, *2*(1), 5 – 47.

Werner, G. M., & Dyer, M. G. (1991). Evolution of communication in artificial organisms. In Langotn, C. G., Taylor, C., Farmer, J. D., & Rasmussen, S. (Eds.), *Artificial Life II - SFI Studies in the Sciences of Complexity*, Vol. X, pp. 659 – 687 Redwood City, California. Addison-Wesley.

Zahavi, A. (1975). Mate selection – a selection for a handicap. *Journal of Theoretical Biology*, *53*, 205 – 214.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

An Overview of Motion Processing in Mammalian Visual Systems.

Stephen Eglen
stephene@cogs.susx.ac.uk

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

Abstract This paper summarises a talk given at the Isle of Thorns Conference in July 1994. The paper is broken down into three main areas. Section 1 describes the importance of motion processing for visual systems of varying complexities, from the fly to humans. Section 2 gives an outline of the monkey visual system, as an example of a mammalian visual system. This reveals the existence of a main pathway for motion processing. Finally, Section 3 considers two alternative ideas for how such a complex visual system could develop: genetic specification and activity driven self organisation. It is concluded that both processes are important for normal development of the visual system.

1 The importance of motion.

What is the starting point of vision? Light enters the eye and is transformed into a two dimensional neural image by the photoreceptors in the retina. In humans, this two dimensional retinal image is then sent via the optic nerve to the lateral geniculate nucleus (LGN) and then mainly onto the visual cortex. However, just thinking of the retina as producing two dimensional snapshots of the world and sending them to the cortex can be too restrictive. The retina is continuously processing light information both in space and time. Hence, the retina can be thought of as producing two dimensional spatial images over time, i.e. three dimensional spatiotemporal images. By detecting changes in both space and time, many important features about the visual image can be extracted.

J.J. Gibson was one of the first people to introduce the idea of looking at how the visual image changed over time (Bruce & Green, 1990, Chapter10). Gibson developed the notion of the *optic flow field*. Mathematically this is a vector field, with the vector at each point in the two dimensional field describing how that point is moving. Two example flow fields are given in Figure 1.

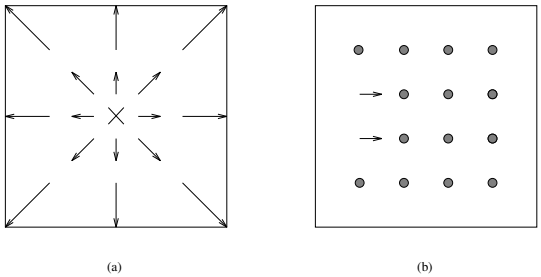

Figure 1: Two example flow fields. The direction of the arrow indicates the direction in which a point in the visual field is moving at a particular time. The size of the arrow indicates the speed at which that point is moving. In (b), the circles indicates that there is no movement at that point.

These two example flow fields are useful in that they demonstrate how analysis of optic flow can be used to determine aspects of self motion. In the first case, the global expansion pattern indicates that the

observer is heading towards the point marked X. Hence, global expansion of the optic flow field reveals the direction of heading. The second optic flow field in Figure 1 shows local movement within the optic flow field. This indicates that the observer is stationary and that something in the world is moving to the right. (There is an alternative interpretation: it could be that as the observer is moving, so is most of the world in the same direction, but this is less likely.)

But, how useful is this notion of the optic flow field? Gibson is often (rightly) criticised for not specifying how the neural mechanisms in the visual system can produce and use such a flow field. However, Lee (1980) was the first to show how simple it actually is to compute simple properties of the flow field, such as the well known time to contact measure, τ. This measure, τ, indicates how long it will take before the observer collides with the object which it is heading for, assuming the observer continues moving at constant velocity. Empirical evidence ranging from basketball players jumping to punch a ball through to gannets diving into water showed that the τ parameter was a good predictor of behaviour (Bruce & Green, 1990, p264).

The optic flow field is by no means the answer to all of the problems in vision. Computational methods of computing the optic flow field are not simple, typically requiring assumptions about the world which do not hold. For example, one popular method of computing the flow field was introduced by Horn and Schunck (1980). Here it is assumed that the overall intensity of the two dimensional image does not change over time. This method also relies on an iterative relaxation scheme to compute the optic flow field, which is computationally expensive.

Despite this computational problem, many natural visual systems seem capable of performing some motion processing. For example, elementary motion detectors exist very early in the fly visual pathway. These motion detectors then feed onto neurons in the lobula plate which are responsive to large field and small field motion (Egelhaaf & Borst, 1993, p666). Also, in the frog retina, there are the classic *bug detectors* — these are the ganglion cells in the retina that respond best "when a dark object, smaller than a receptive field, enters that field, stops, and moves about intermittently thereafter." (Letvin, Maturana, McCulloch, & Pitts, 1959, p1951). If the dark object is not moving, these ganglion cells do not respond.

Rapidly approaching objects are normally important for an animal to recognise, and so it is reasonable to expect to find 'looming detectors' in visual systems. As well as the previous examples, such detectors have been found in the nucleus rotundus of the pigeon brain. A sub population of neurons in this region are found to respond selectively to objects moving in the direction of the pigeon, responding over a range of stimuli sizes and velocities (Wang & Frost, 1992, p236). Finally, in monkeys, neurons in area MST (medial superior temporal) respond to global expansion and rotation patterns (Duffy & Wurtz, 1991; Orban, Lagae, Verri, Raiguel, Xiao, Maes, & Torre, 1992).

It is worth noting here that many visual systems do some form of motion processing and that the 'simpler' the creature, the earlier in the visual pathway it seems that the motion information is extracted. Motion processing is therefore important and maybe more so for simpler visual systems. When considering more complex visual systems, such as the human visual system, things are less clear, as the next section will show.

One last striking example of the importance of motion is shown by the biological motion displays, created by Johansson in the early 1970's (Bruce & Green, 1990, p327). Lights were attached to the joints of a person who were then filmed walking or running. The contrast of the camera was adjusted so that only the lights could be picked up by the camera, without any impression of the background or the person. If subjects viewed a single frame of this film, all they would see is a meaningless jumble of dots. As soon as they saw a few frames of the film, they instantly perceived the dots as belonging to a human body. This is a good example of a spatiotemporal image that is spatially not very informative being very informative when viewed temporally.

The aim of this section has been to show that motion processing is a very important part of visual processing. This does not imply that motion processing is the only process for all visual systems, although it does seem that simple visual systems rely quite heavily on it. The next section discusses in more detail how motion is processed in mammalian visual systems.

2 Processing of motion in Mammalian Visual Systems.

[Although this section refers to mammalian visual systems, there are differences between the cat, monkey and human visual system. For clarity, all specific details are taken from the macaque monkey.]

The visual cortex in mammals is located in the occipital lobe of the cortex, occupying about 55% of the total cortical area in the macaque monkey (Felleman & Van Essen, 1991). Retinal ganglion cells form the output from the retina, which passes to the LGN and onto area V1, the primary visual cortex. As well as area V1, there are many other (extra striate) visual areas that have been described over the last thirty years. These areas are normally defined according to various anatomical and physiological criteria, and some debate exists over how many there are. A few years ago, the number was thought to be between 15 and 20 (Van Essen & Maunsell, 1983, p374), but this number has steadily risen to about 30 (Felleman & Van Essen, 1991). Figure 2 shows the connectivity of some of the most commonly studied areas in the cortex.

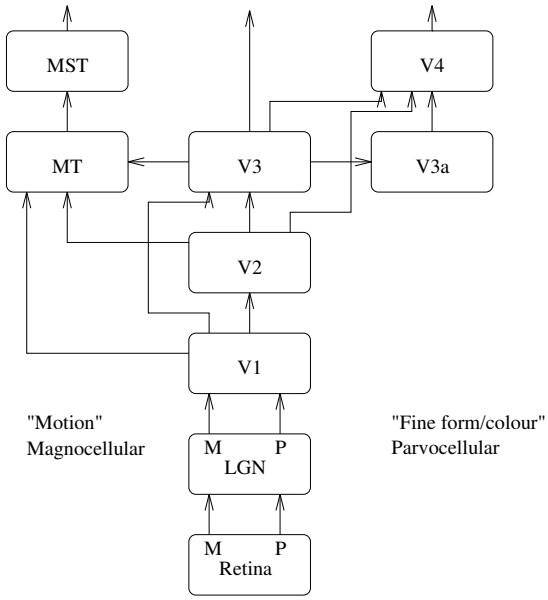

Figure 2: Outline of the main visual pathway found in the monkey visual system, from the retina to the main areas in the cortex. Note that only feedforward connections between areas are shown — extensive feedback connections do exist between areas but has not been shown for simplicity.

As can be seen from Figure 2, V1 is the first main visual cortical area. A vast amount of research has gone into trying to understand what is happening in V1, leading to a popular classification of neurons in V1 as either:

1. Simple cell — The cell responds optimally to a stimulus of a certain orientation at a certain spatial location. The cell's response is linear and can normally be predicted from the stimulus.

2. Complex cell — The cell responds to a stimulus of a certain orientation, but over a wider area than the simple cell. The cell's response is non-linear and cannot normally be predicted from the stimulus.

3. Hypercomplex cell — The cell responds optimally to a stimulus of a certain orientation moving in a direction which is usually perpendicular to the preferred orientation.

An example of the preferred stimulus for each of these cells is given in Figure 3. These cells are topographically arranged, so that neighbouring cells in area V1 are activated by neighbouring points in the retinal image. Secondly, these cells are arranged so that neighbouring cells respond to stimuli of similar orientations.

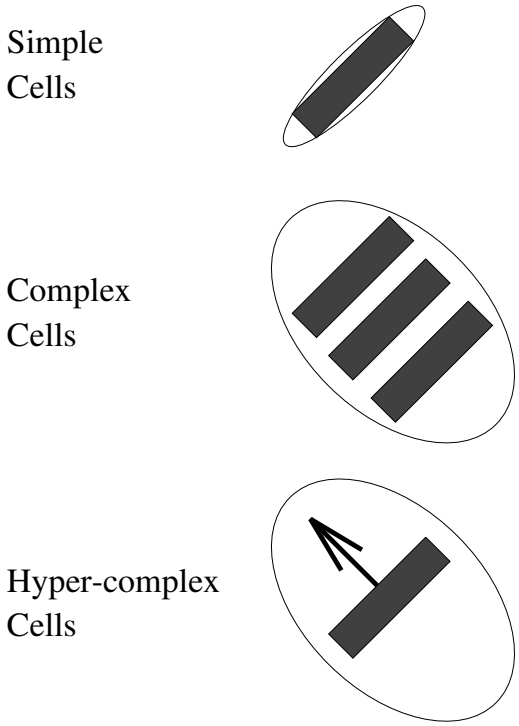

Figure 3: Popular classification of V1 neurons into simple, complex and hypercomplex cells. The grey bar indicates the optimal stimulus for the cell. The ellipse indicates the extent of the cell's receptive field. In the case of the complex cell, the bar can be in one of many positions – three possible positions are shown here. Finally, in the hypercomplex cell, the arrow indicates the preferred direction of stimulus movement.

Physiological	Pathway	
Criteria	Magnocellular	Parvocellular
Colour Selectivity	No	Yes
Contrast Sensitivity	High	Low
Temporal Resolution	Fast	Slow
Spatial Resolution	Low	High

Table 1: Characteristics of the Magnocellular and Parvocellular pathways. Data taken from Livingstone and Hubel (1988, p746).

The initial discovery of the simple,complex and hypercomplex cells led to a serial view of visual processing: retinal signals feed into the LGN and then into the simple cells. Simple cells feed into complex cells, which in turn feed into hypercomplex cells, and so on. However, this view did not last very long, as evidence accumulated for a more parallel, but not totally modular, approach (Merigan & Maunsell, 1993). As Figure 2 shows, the current view is that there are two main streams. The magnocellular pathway is responsible for motion analysis and broad spatial layout, whereas the parvocellular pathway is responsible for fine form and colour analysis (Livingstone & Hubel, 1988). It is interesting to note that even in the serial pathway from the retina to the LGN to V1, there is a subdivision into large Magno (M) cells and small Parvo (P) cells. These cells provide the major input to the Magno and Parvo channels. Table 2 gives an overview of the qualities of each of the pathways.

Returning to the problem of optic flow, neurons in areas MT (medial temporal) and MST are highly selective for moving stimuli. Neurons in area MT have a good selectivity for motion over a receptive field of up to 20×20 degrees. These neurons then project heavily to MST which are capable of detecting global expansion and contraction or rotational motion over a receptive field area of up to 100×100 de-

grees (Tanaka, Fukada, & Saito, 1988). Hence, a popular notion is to think of the magnocellular pathway as important for motion processing, e.g. (Livingstone & Hubel, 1988). However, it should not just be regarded as a motion pathway, as it ultimately feeds into the parietal area, which is needed in tasks such as finding the location of objects (Livingstone & Hubel, 1988, p744). This contrasts with the parvo pathway, which leads to the temporal cortex. This area is normally associated with deciding *what* the object in the visual field is. It is also important to note that the Magno and Parvo pathways should not be regarded as completely segregated. Cells in both pathways can typically fire to most stimuli, with the exception of colour, which is solely dealt with in the Parvo pathway.

3 Development of the Visual System.

The previous section outlined the organisation of mammalian visual systems. The initial discovery of the simple, complex and hypercomplex cells was made in the early 1960's by Hubel and Wiesel. There is such a high degree of organisation found in the visual system that led Hubel and Wiesel to believe that the visual cortex is genetically specified (Hubel, 1978 cited by Zeki 1993, p216). However, this is in contrast to the widely held belief that the cortex is very adaptive. In 1959, Penfield discovered that a lesion could destroy some aspects of language function, but if the patient was young enough, language function could be recovered. If the patient was an adult, the same lesion meant that language function could not be recovered (Zeki, 1993, p220). Could this plasticity be a general principle for the cortex, and not just for the language areas?

Evidence for the importance of environmental input affecting development of the visual system began accumulating in the 1960's. One of the first experiments was performed by Wiesel and Hubel (1963). The experiment involved covering one eye of a kitten from birth for the first ten weeks of life. The eye was then uncovered, and the kitten examined. The covered eye still functioned normally, but the normal pattern of ocular dominance (regular interdigitation of inputs from each eye across a patch of cortex) was not found in area 17[1]. Instead, most neurons respond only to inputs from the eye that was left open. None of the cells in area 17 that were tested could be influenced by the eye that was covered. Another early visual deprivation experiment was to raise kittens wearing goggles such that they could only see vertical lines. After the first few weeks of life, neurons in area 17 only responded to vertically oriented stimuli (Hirsh & Spinelli, 1970).

Many such neurophysiological experiments have since been performed, leading to the idea that activity of the visual system causes it to self organise. Consequently, non normal activity of the visual system produces abnormalities in the visual system. Faithful supporters of the idea that the visual system is genetically specified can always argue that the visual deprivation experiments are consistent with their ideas. It may be the case that although the visual system is genetically specified at birth, cells that are not active during the deprivation experiments die out, and so the mature animal loses some functions if they are not used during early life.

However, an important experiment against this argument was performed by Sur, Garraghty, and Roe (1988). The retinal inputs from the LGN (the normal innervation site) were redirected to the Medial Geniculate Nucleus (MGN)[2]. The ferrets were then reared to adulthood in a normal visual environment. It was found that many cells in the MGN and auditory cortex were visually driven, although receptive fields tended to be larger than normal, showing a preference for moving spots of bars. Additionally, a correct topographic map of the visual field was found in the MGN. Sur et al. concluded by saying:

> "Functional visual projections can also be routed into non visual structures in higher mammals, suggesting that the modality of a sensory thalamic nucleus or cortical area may be specified by its input during development." (Sur et al., 1988, p1437)

[1] Area 17 is the cat equivalent of monkey area V1

[2] This is the auditory equivalent of the LGN.

As well as the neurophysiological experiments, another approach to understanding the development of the visual system is by computer simulation. Typically, these simulations involve modelling neural networks to see if they can self organise to produce neurons with similar response properties as real visual neurons. The first main simulation of this type was produced by von der Malsburg (1973), initiating many other models over the following twenty years. Here at Sussex, many simulations, modelling simple cells (Barrow, 1987), complex cells (Barrow & Bray, 1992b), colour blobs (Barrow & Bray, 1992a) and ocular dominance stripes (Goodhill, 1992) have been performed.

Given the importance of motion processing that has been mentioned earlier, it is surprising that the modelling of motion cells has been quite rare, with the exception of (Wang, Mathur, & Koch, 1990) and (Nowlan & Sejnowski, 1993). (Specific models of MT and MST development have been produced, but these typically assume certain functions for the retina, V1 and V2, eg. (Sereno & Sereno, 1991; Tanaka & Shinbata, 1994).) This is now a topic of research here at Sussex.

4 Conclusion.

An overview of the importance of motion processing has been given, along with the architecture of a mammalian visual system. The existence of a magnocellular pathway that deals with (amongst other things) motion processing seems to reinforce the importance of motion. The final section outlined two main theories for the development of the visual system: (1) genetic specification or (2) activity driven self organisation. It is likely that a combination of both of these processes are needed to produce a normal visual system, by working on different aspects of the problem. First, genetic specification can initially describe the overall topography and connectivity of retinal inputs and cortical areas. Second, the activity driven processes can then fine tune the connectivity and response properties of individual neurons in accordance with the environmental input. Computer simulations of these activity driven processes can show how much each process contributes to the development of visual systems.

Acknowledgments

I would like to thank my supervisor, Harry Barrow, and Julian Budd for all the help they have given me during my first year at Sussex. I would also like to thank Geoff Goodhill for useful comments on this paper.

Reference

Barrow, H. (1987). Learning receptive fields. In *IEEE 1st International Conference on Neural Networks*.

Barrow, H. G., & Bray, A. J. (1992a). Activity-induced "colour blob" formation. In Aleksander, I., & Taylor, J. (Eds.), *Artificial Neural Networks, 2: Proceedings of the International Conference on Artificial Neural Networks*. North-Holland.

Barrow, H. G., & Bray, A. J. (1992b). A model of adaptive development of complex cortical cells. In Aleksander, I., & Taylor, J. (Eds.), *Artificial Neural Networks, 2: Proceedings of the International Conference on Artificial Neural Networks*. North-Holland.

Bruce, V., & Green, P. (1990). *Visual Perception: Phsyiology, Psychology and Ecology* (2 edition). LEA.

Duffy, C., & Wurtz, R. (1991). Sensitivity of MST neurons to optic flow stimuli .1. a continuum of response selectivity to large-field stimuli. *Journal of Neurophysiology*, *65*(6), 1329–1345.

Egelhaaf, M., & Borst, A. (1993). Motion computation and visual orientation in flies. *Comp. Biochem. Physiol*, *104A*(4), 659–673.

Felleman, D., & Van Essen, D. (1991). Distributed hierarchical processing in the primate cerebral cortex.. *Cerebral Cortex*, *1*, 1–47.

Goodhill, G. (1992). Correlations, competition and optimality: modelling the development of topography and ocular dominance. CSRP 226, School of Cognitive and Computing Sciences, Sussex University, UK.

Hirsh, H., & Spinelli, D. (1970). Visual experience modifies distribution of horizontally and vertically oriented receptive fields in cats. *Science*, *168*, 869–871.

Horn, B., & Schunck, B. (1980). Determining optical flow. Tech. rep. 572, AI LAB, MIT.

Lee, D. (1980). The optic flow field: The foundation of vision. *Phil. Trans. R. Soc. Lond. B*, *290*, 169–179.

Letvin, J., Maturana, H., McCulloch, W., & Pitts, W. (1959). What the frog's eye tells the frog's brain. *Proceedings of the IRE*, 1940–1951.

Livingstone, M., & Hubel, D. (1988). Segregation of Form, Colour, Movement, and Depth: Anatomy, Physiology, and Perception. *Science*, *240*, 740–749.

Merigan, W., & Maunsell, J. (1993). How parallel are the primate visual pathways?. *Annual Review Neuroscience*, *16*, 369–402.

Nowlan, S., & Sejnowski, T. (1993). Filter selection model for generating visual motion signals.. In Hanson, S., Cowan, J., & Giles, C. (Eds.), *Advances in neural information processing systems*, Vol. 5. Morgan Kaufmann.

Orban, G., Lagae, L., Verri, A., Raiguel, S., Xiao, D., Maes, H., & Torre, V. (1992). 1st-order analysis of optical-flow in monkey brain. *Proceedings of the National Academy Of Sciences of the United States of America*, *89*(7), 2595–2599.

Sereno, M., & Sereno, M. (1991). Learning to see rotation and dilation with a hebb rule. In Lippmann, R., Moody, J., & Touretzky, D. (Eds.), *Advances in Neural Information Processing Systems 3*, Vol. 3. Morgan Kaufmann.

Sur, M., Garraghty, P., & Roe, A. (1988). Experimentally Induced Visual Projections into Auditory Thalamus and Cortex. *Science*, *242*, 1437–1441.

Tanaka, K., Fukada, Y., & Saito, H. (1988). Underlying mechanisms of the response specificity of expansion/contraction and rotation cells in the dorsal part of the medial superior temporal area of the macaque monkey.. *Journal of Neurophysiology*, *62*(3), 642–656.

Tanaka, S., & Shinbata, H. (1994). Mathematical model for self-organization of direction columns in the primate middle temporal area.. *Biological Cybernetics*, *70*, 227–234.

Van Essen, D., & Maunsell, J. H. R. (1983). Hierarchical organization and functional streams in the visual cortex.. *Trends in Neuroscience*, *6*, 370–375.

von der Malsburg, C. (1973). Self-organisation of orientation sensitive cells in the striata cortex. *Kybernetik*, *14*, 85–100.

Wang, H., Mathur, B., & Koch, C. (1990). I thought I Saw It Move: Computing Optical Flow in the Primate Visual System. In Gluck, M., & Rumelhart, D. (Eds.), *Neuroscience and Connectionist Theory*, chap. 6, pp. 237–265. Lawrence Erlbaum Associates.

Wang, Y., & Frost, B. (1992). Time to collision is signalled by neurons in the nucleus rotundus of pigeons. *Nature*, *356*, 236–238.

Wiesel, T., & Hubel, D. (1963). Single cell responses in striate cortex of kittens deprived of vision in one eye.. *Journal of Neurophysiology*, *26*, 1003–1017.

Zeki, S. (1993). *A Vision of the Brain.* Blackwell, Oxford.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Macroscopic Explanations

Philip Jones
philipj@cogs.susx.ac.uk

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

Abstract There is a whole body of thinking in the social sciences which asks about the relationship of individuals to groups. Some of it sees group behaviour as merely epiphenomenal on the individual, but some makes claims that high level structures can exert a causal effect on the behaviours and perceptions of the agents who comprise them. As cognitive researchers, should we give these claims any serious consideration?

Introduction

> "The explanations of animal societies offered by biologists are essentially reductionist. That is, they attempt to explain the structure of societies as a consequence of the properties of the individuals which compose them. By no means all sociological or anthropological theories are of this kind. Theories in economics are reductionist… But many sociological theories are not reductionist in even this limited sense. The properties of the individual are seen as produced by society, and even as serving the purposes of that society, and not the other way around." (Maynard-Smith, 1982)

The quote comes from the introduction to a book on problems in sociobiology. And it neatly stakes out the territory that a science such as biology lays claim to when it attempts to explain social behaviour. The contrasting attitudes in Maynard-Smith's quote can also be seen as two views on agency. The reductionist sees agency as the emergent effect of multiple microscopic components. But Smith's sociologist sees the mind moulded by its environment and those macroscopic entities of which it is itself a sub-component. In fact, social scientists are aware of the tension between these two interpretations of the relationship between the individual's behaviour, and the social structure which he or she inhabits. They label these positions *structure* (or holistic or top-down) and *action* (or individualistic or bottom-up).

What is interesting about social science is that it looks very like cognitive science played in the mirror. Cognitive science concentrates on two levels: the first is the causally mechanistic, physical level of the brain; the second, the psychological, intentional level of the mind. The task facing engineers of artificial intelligence and cognitive philosophers is to make these two levels compatible; to explain the relationship between them in a way which fits the mind into natural science, but does not ride roughshod over our intuitive feelings about it. The mind is seen as emergent out of, or supervenient upon, or in some way above the mere mechanistic level.

By contrast, a social scientist, or at least one who falls into the holistic or structural camp, also sees two levels - one of which is mechanistically causal, the other of which is intentional. But here, it is the higher, macroscopic level which obeys causally mechanistic rules, and the low level which is intentional. Within the field, there are also fierce debates between those who see the mechanistic level as being causally responsible for the intentional, and those who wish to defend intentionality from macro-reduction to these principles. Emile Durkheim, for example, asks us to accept that

> "society is not a mere sum of individuals. Rather, the system formed by their association represents a specific reality which has it's own characteristics."

This superstructure so constrains the behavioural opportunities of the individual that it is seen as essentially guiding (or causing) that behaviour.

Within social sciences, there has been a robust defence against this way of thinking. From J.S. Mill in the 19th century to modern individualists, many are unhappy with the possibility of macroscopic structures being causal of microscopic properties. Two possible refutations seem to run as follows:

The first is to argue that these high level structures have no goals of their own. They merely inherit their seeming intentionality from the intentional humans who comprise them.

But against this, the holist believes that structures may be acting according to their own agendas. No one wants an economic recession or a bout of inflation; perhaps no one intends a moral norm to collapse. Yet these things happen. How is this to be squared with the idea that the intention to behave has come from the low level? One solution is the *great man* theory. That somewhere, there is someone who secretly did want the change and, this time, has got his way. Cognitive scientists may recognise this as a parallel to the idea of the grandmother sensor[1] or undischarged homunculus.

A second strategy individualists use against structuralists is to ask directly: how does it work? How can this high level system cause these low level people to do things? The answer demanded is a low level causal one: a story of who did what to whom. When presented with such, they can then argue that, this is obviously, only a story about individuals and their behaviour. In other words, this is a reductionist argument of the kind which those cognitive scientists who wish to preserve the mind from being mere brain behaviour, are continuously fighting against.

These are rough parallels, but nevertheless, our first glance into the distorting mirror of social science, has been enlightening. Because the mirror reverses the levels that we normally consider intentional and non-intentional, it might help us understand the opposing viewpoint to our usual one.

From our intuitions about folk psychology, we tend to reject the dogma of the holistic social scientist. But from our intuitions as cognitive researchers, about both homuncular decomposition, and reductive materialism, we are tempted to deny the arguments which the individualist social scientists use to deny high level structures. Any argument which seems to deny high level structure in favour of individual action, is dangerously close to one which would collapse the psychological into the neurochemical. By contrast, any argument which could pump enough hot air into the mind to keep it afloat above the brain, could probably launch a few leviathan like macroscopic entities.

There are, of course, further, arguments which separate out these two situations. Interlevel relationships come in many subtle flavours and it is possible to argue that the relation between neural and psychological is of a different kind than the relation between mental and structural.

But in this paper I intend to accept the possibility of these social structures and argue that not only can they be useful for cognitive scientists, but that we are already beginning to use them. First though, I will explore a little further what they are, and what claims they have to existing in the first place.

Living in Structure

Natural science provides views at multiple levels of abstraction upon the physical world. Quantum mechanics, classical atomic physics and chemistry are familiar examples. Each level provides a powerful, simple theory in terms of the relationships between different entities or solids. Of these, some are seen as candidates for being real, to use Dennett's terminology: *illata*; while others, although useful are considered mere abstractions or *abstracta*. (Dennett, 1987)

There are two fundamental philosophical questions which can be asked about these illata. The first is, "Do they really exist?" and the second is "How can we know them?". To introduce some terminology

[1] How did this macroscopic mind manage to think about grandmother when all it is is a load of neurons firing? Well, obviously, one of the neurons is the neuron which thinks about grandmother.

which will be familiar to some and mysterious to others, the first of these is a question of *ontology*, and the second *epistemology*. Martin Hollis (Hollis, 1994) thus summarizes the structural or holistic claim, attributed to Durkheim, like this:

- "an ontology of 'social facts', forming an order external to individual consciousness and not explicable by reference to human nature.
- a methodology wherein social facts are explained by their function 'in relation to some social end.'
- functional mechanisms working through the medium of the 'collective consciousness' and connecting social ends to the overall level of social integration needed if a society is to flourish.
- an epistemology, so far undisclosed, which warrants our subscribing to these components."

These "social facts" or illata, were traditionally seen as economic or political groupings such as class, or state. And they were believed to have particular interactions with each other which could be captured by *scientific* rules, regardless of the individuals who composed them. The relationship between high level structure and component may be subtle and convoluted. For example, at one point, Durkheim makes a claim which, though counterintuitive, illustrates the sophistication of the relationship he is promoting. The State is a powerful, self supporting, system[2] and criminals are not its enemies, but an essential component, which helps hold it together. (Presumably because common outrage at criminal acts encourages people to defend the state against them.) Were people to abstain from acts currently considered criminal, the State would begin to redefine other acts as taboo, to restore the balance within itself, just as living bodies balance the proportions of their components.

Another idea of this genre which deserves note, is that of *ideology* which is defined as false consciousness. For some holists the high economic level of description is seen as the true level of illata; and a viewpoint of an individual who sees the world only in terms of individuals and their personal beliefs and desires is considered misguided. Individual consciousness is the intersection of falsehoods invented by various high level illata such as Church and State as part of their homeostatic behaviour.

Here again, is a distorted inversion of a familiar idea. For some cognitive scientists, consciousness is very nearly treated as a falsehood; a narrative spun by the brain, possibly out of multiple contradictory threads of perception.

Some Recent Parallels

Not surprisingly, extreme holistic social science is often rejected, and many of the theories which were created in the nineteenth century and flourished in the early part of this century, are discredited. But has the idea of social structure really died out? One interesting recent area of study is *Game Theory*, which has been explored by von Neumann, and others over the last half century. Extraordinarily, game theory is held up as an individualist analytic tool and a rival to structure based explanations. It does start with the idea of an individual, who is *rational*. In other words, the individual will predictably chose to behave in a way which, as she perceives things, will maximise her gain in a particular situation. But then such situations are classified as being instances of one of several abstract games such as the *Prisoner's Dilemma* or *Chicken*. Thus the game provides a framework which is itself a kind of law.

If agent X is rational then agent X will perform action A.

Some social scientists see game theory as being in opposition to traditional holism, but the parallels seem much stronger. There is not such a great difference between saying that agent X is in a situation defined as a Prisoners Dilemma and should therefore do A; and saying that agent X is in a situation defined

[2]Today, we might say autopoeitic.

as a Class War and should therefore do A. Both allow an observer to tell the same kind of explanatory stories about why X performed A (because A was the rational choice) and both can make the same kind of predictions (assuming X is rational he will perform A). [3] Both are theories that there are situations or relationships which agents can find themselves in, whose very nature makes a particular actor's behaviour meaningful, or more predictable.

Thinking in terms of high-level structures can occur when borrowing explanations from biological sciences. Evolutionary theory is already shot through with intuitions that are top-down, and often criticised for being based on tautology. Concepts such as fitness, are only defined in terms of a circle of high level entities. No genotype is fit merely by virtue of its internal structure. Rather it is fit with relation to the phenotype, and other genotypes and environment. Co-evolution or the evolutionary arms race, is a high level structure introduced to provide explanations of current properties.

Evolutionary roboticists, who attempt to evolve solutions to problems, find themselves thinking in terms of the evolutionary "niche"; and trying to produce behaviour by designing the environment and fitness function within which a particular control system will evolve. Hence there is some acknowledgement of a world which is prior to, and causally responsible for, the behaviour of the individual. This is particularly evident when one hears calls for a metric that will compare and classify environments according to the behaviour they engender. As they attempt to *evolve* new agents, they are relying heavily on the perceived causal power of an illata like co-evolution. Even Durkheim's ideas about criminality have found a parallel in a discussion of the virtues of parasitism.

Outside the Head

For this final section, I will change the subject slightly in order to emphasize another use that we might make of macroscopic entities.

Consider Terrence Horgan's (Horgan, 1993) assertion, that:

> "Materialists who back away from type-type psychophysical identity claims, but who also seek to vindicate the causal/explanatory efficacy of mental properties, are already committed to some form of compatibilism on the issue of mental quasation. Since they are stuck with this compatibilist commitment anyway, *they should take seriously the possibility that the right kind of compatibility will vindicate the causal/explanatory efficacy of mental properties that do not supervene on the properties that physically realize them, and perhaps will also vindicate the causal/explanatory efficacy of mental properties that do not even supervene on what's in the head.*"

This is a quote with some questionable anomalies. But consider only the emphasized portion of the text. If, as Terrence Horgan seems to, we want to talk about mental entities as supervenient upon more than what's in the head, then we are going to need to consider physical contexts that have a far wider scope than individual brain structures. The brain is already too complicated an organ to really think about in detail. So considering brain and a context which could, itself, include many other intentional agents, will require a high degree of abstraction. Those abstractions will be entities within some macroscopic theory and thus be open to the sort of questioning that more traditional macro-entities receive. Macroscopic structure just *is* the vocabulary needed to describe and model the context within which an agent operates.

Conclusion : Taking Society Seriously

This paper is only one step in a project dedicated to taking social, or macroscopic structure seriously when thinking about minds. What, though, is the purpose of this attempt to resuscitate social or top-down ex-

[3] One difference may be that game theorists will deny that they are making strong ontological claims for Prisoner's Dilemma games. Prisoner's Dilemmas do not exist. They are merely abstracta, not illata and have no causal power over the players who find themselves in the relationship. By contrast, the holistic social scientist is making this claim about Class.

planations of agency when we seem to be making such progress through scientific and reductionist approaches? Isn't this just obfuscating the issue, or worse, trying to sneak some dodgy political or humanities talk into cognitive science?

First, there is an issue of general macro-scopic causation. Can there be a flow of explanatory responsibility back from the macroscopic entity to the individual? This relationship, sometimes known as quasation or quasi-causation is still controversial.

A nice toy example that is often used to make points about levels of description is Conway's Life Game. One can argue that the Life game may be viewed either on the level of individual cells or at a higher level of gliders and other abstractions. It is not that the Life game requires to be interpreted at either one or the other of these levels. Both are contemporaneously appropriate.[4]

Consider a particular Life universe containing one glider which at a particular time t has passed through cell C. At t we can say that cell C has become alive. The question "did C become alive because it became involved with the glider?" cuts right across carefully separated levels of description. In this case a macroscopic explanation is intuitively acceptable. It really does seem that the passing of the glider caused the behaviour of C. But Maynard-Smith's reductionist, while able to say that the glider passed through C at t because C came alive then, ought not to phrase it the other way around.

The second issue is the particular problem of social illata: higher level structures to which rational, intelligent agents belong. Does the fact that we are already intentional beings prevent these higher level structures from influencing or causing our behaviour? Or, as claimed by holists, do we derive intentionality from those structures which we make up?

There are several good reasons for studying social illata. The first, as pointed out in the beginning of the paper, is that social science provides a kind of distorted mirror image of cognitive science. Looking into it we see some of our intuitions turned upside down, and some familiar arguments stretched and squeezed into unfamiliar shapes. This may inspire us to new intuitions about the problems of relating levels.

Secondly, it appears that, through considerable interest in evolutionary theory, researchers, particularly in ALife, have already begun to call on some macro-structure to do explanatory work. The language used by ethologists when studying animal's group behaviour begins to blend into that of the sociologist.

Finally, macroscopic entities, can be seen as tools for talking about context and relationships within that context. In this case, telling explanatory stories about mental illata may require one to include macro-illata.

Reference

Dennett, D. (1987). *The Intentional Stance*. M.I.T.

Hollis, M. (1994). *The Philosophy of Social Science*. Cambridge University Press.

Horgan, T. (1993). From supervenience to superdupervenience. *Mind*, *102*(408), 554–586.

Maynard-Smith, J. (1982). Introduction. In Group, K. C. S. (Ed.), *Current Problems in Sociobiology*. Cambridge University Press.

[4] Although occasionally there are times during the evolution of the Life game, when there are NO patterns for which abstraction to an alternative levels would be appropriate. Imagine a large Life universe has just been filled randomly (a 0.5 probability that any cell will be alive. For a while, it is uncertain whether any stable structures will appear. Although the beauty of Life is that some nearly always do.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Creativity in Writing and Music

Rafael Perez y Perez
rafaelp@cogs.susyx.ac.uk

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

Abstract This paper gives a short overview of my approach to studying creativity. Specifically, it introduces the notions of experiential and reflective state and it tries to show why and how these will figure centrally in an account of creativity.

1 Introduction

The goal of my research is to study creativity in arts, placing special emphasis on the fields of music, and writing. The core of my investigation will concentrate on the relationship between reflective and experiential states and its role and importance for the creative process. *Experiential state* can be described as a state in which a flow of ideas is generated in order to build a piece of music, to write a story, etc. Such a flow of ideas is guided by tacit constrains like cultural background, the use of specific style or technique, etc. Although everybody can experience such states (like for example day-dreaming), in the case of artists some of the tacit constraints are the result of years of experience, the development of certain skills, and training in specific areas. These further constraints allow artists to produce works of art of which it is difficult to imagine that they could have been produced by the layman. *Reflective state* can be described as a state where the artist structures and/or evaluates his/her work. Again, during such state artists are helped by tools like music theory, writing techniques, etc. which they have learned during their training and experience as artists.

The main hypothesis of this research is that both states participate actively during the creative process, although during such process artists can have preferences for one of them. For example Chandler (1992) defines two types of writers, the Planners and the Discoverers:

> - *Planners* tend to think of writing primarily as a means of recording or communicating ideas which they already have clear in their minds;
> - *Discoverers* tend to experience writing primarily as a way of 'discovering' what they want to say.

Aarond Copland talks about different kinds of composers, "The type that has fired public imagination most is that of the spontaneously inspired composer–the Franz Schubert type, in other words... this type [of composer] is more spontaneously inspired. Music simply wells out of him..." (Copland 1955, p.22). The second type, called "The constructive type" by Copland, starts with a musical idea or theme, and then s/he analyses and plans how to work with it.

> In Beethoven's case there is no doubt about it, for we have the notebooks in which he put the themes down. We can see from his notebooks how he worked over these themes, how he would not let them be until they were as perfect as he could make them. Beethoven was not a spontaneously inspired composer in the Schubert sense at all. He was the type that begins with a theme; make it a germinal idea; and upon that constructs a musical work, day after day, in painstaking fashion (op.cit., p.22)

Terms referring to such preferences like Mozartians-Beethovians, The Makers-The Possessed, Executors-Discoverers, Doers-Thinkers, etc. are found in the literature (see Chandler 1992).

Thus, the research in the interaction of the Experiential-Reflective states can help to better understand the creative process.

2 Research Questions

Some of the questions I attempt to answer with my research are:

- Are Experiential-Reflective states important for creativity?
- What is the relation between them?
- Are these two states enough to provide bases for an explanation of creativity?
- Is it possible to develop a computational model of creativity which engages in Experiential-Reflective states?
- What is the importance of personal experiences and of cultural background for the Experiential-Reflective states?
- What is the relationship of feelings like ”lack”, ”incompleteness” or ”satisfaction” that artists experience towards their work to Experiential-Reflective states?

3 Methodology

The methodology I will follow during my research is based on the study and analysis of:

1. Artists' self-accounts.
2. Reports regarding to the creative process and products of other artists.
3. Examples of such creative processes and products.

There are many critics with respect to self-reports (see for example Weisberg 1992). Some of the disadvantages of this methodology are that it is not possible to select the people you are going to get the information from and that it is not possible to have any control on the accuracy of their accounts. Vicente Lenero (1983) has written that when journalists or researchers have asked him to talk about ”the intimacy” of his most famous book *Los Albaniles*, in order to defend himself from adverse comments or to capitalize in his favour positive appraisals, he has invented social worrying, narrative searches, etc. which never were really present during the process of writing. Thus, sometimes self-accounts are shaped for diferents kinds of pressures like expectations about how the creative process works, artist's intentions or messages expressed through his/her work, political positions, etc.

However, it does not follow that all self-accounts are misleading. In fact, this methodology offers impor tant advantages; the most significant of them is the opportunity to get information from an important number of well known creative artists with completely different background which would be impossible to get in a different way. And although sometimes there can be difficulties with the accuracy of self-accounts, they are invaluable sources of information and clues of the creative process. In order to minimize this problematicity, I will compare different artists' self-accounts, and researchers' and artists' opinion about the creative process in other artists with my own analysis and opinion. In this way I attempt to get a set of common characteristics between them, which will then form the framework of my model of creativity. Finally, I will attempt to design a computational system of such creative-process.

In the next paragraphs I will briefly give an interesting example of this kind of analysis to show how it can help in making a framework for the study of the creative process.

Mario Vargas Llosa (1966) describes how experiences in Victor Hugo's life influenced him to write *Les Miserables*. When Victor Hugo was young, he was impressed by a line of prisoners he saw on the street. He decided to write a short story about prisoners and prisons. When he was looking for information to write his book, he visited the prison and discovered that there were individuals serving life sentences because they had stolen a piece of bread. This situation made him feel angry and he came to realise how much social injustice there was. He tried to find a solution for this injustice: he talked in the Parliament, wrote articles, and at the same time tried to write a story about convicts' lifes. However, something was wrong with the story, it did not satisfy him, and he left it. When he was trying to improve prisoners' lifes and change the penal code, he heard about a wonderful, charitable and compassionate bishop who was living in a small town in France. Victor Hugo got excited with the story of this man and he decided to write a novel in which the main character was a similar bishop. Thus, he signed a contract with a publisher and began to write. However, when he finished the first version of the book he got the same feeling he had got when he wrote the book about the prisoners: something was lacking and he left the project again. He kept himself writing verses, novels, etc. for some years, until one day he got the idea to mix both themes and write a new novel. Nevertheless, this project failed as well; he had the feeling that the text was not real enough and he abandoned it when the French revolution started in 1848. Victor Hugo played an important role as mediator during the war; he visited all the barricades trying to achieve a truce while his house was invaded by the insurgents. However, in all those moments – writes Vargas Llosa – without being aware of it, he was accumulating that definitive third experience which would give to the convicted and bishop's stories that social and historical dimensions, that street's fervour which gives *Les Miserables* all its greatness. Victor Hugo wrote the last version of the novel many years later, when he was exiled on an island in the pacific.

The example of *Les Miserables* is very useful in many ways: First, it shows how certain events in the writer's life are decisive in his/her work (in this case, to come to recognise social injustice). Second, it shows how for some writers the development of a novel is based in a first image or idea (in this case the prisoners' line and the bishop). Third, it shows how important imagination is for writing, i.e., Les Miserables is not an historical book but a book which took real events, transformed them, and created a new way to see that reality. And forth, it shows how a worthwhile book requires a lot of hard work.

References

Chandler, D. (1992) The Phenomenology of Writing by Hand. In *Intelligent Tutoring Media* Vol. 3 No. 2/3, pp. 65-74.

Copland, A. (1955). *What to Listen for in Music*. New American Library, New York.

Lenero, V. (1983). Sobre *Los Albaniles*. In N. Klahn y W.H. Corral (eds.) *Los Novelistas como Criticos*, Tomo II, 1991. Mexico, D.F.: Fondo del Cultura Economica.

Smith, F. (1982) *Writing and the Writer*. London: LEA Publishers.

Vargas Llosa, M. (1966). La Novela. In N. Klahn y W.H. Corral (eds.) *Los Novelistas como Criticos*, Tomo II, 1991. Mexico, D.F.: Fondo del Cultura Economica.

Weisberg, R.W. (1992). *Creativity: Beyond the Myth of Genius*. W. H. Freeman and Company.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Cognitive Science as the Study of Consciousness

Ronald Lemmen
ronaldl@cogs.susx.ac.uk

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

Abstract Consciousness is to cognitive science what life is to biology. It is by having conscious experiences that cognizers have content. This content sits at the level of the whole cognizer, which is why I call it *organismal* content. Other notions of content depend on organismal content. I argue that from taking experiences seriously, it follows both that the notion of mental representations is incoherent and that the processes that *underly* organismal content aren't themselves contentful, at least not literally. Section 2 discusses some of the fallacies that often lead to the confused ideas that consciousness is epiphenomenal and that organismal content would have to be explained in terms of lower-level contents. Finally, section 3 looks at some of the consequences with respect to explanations in cognitive science.

1 A Science of the *Mind*

Once upon a time, cognitive science was supposed to be the study of symbol systems. Those days are behind us now, and now we are in the strange situation that it no longer seems clear what cognitive science is the study of. I am not asking for a definition of cognition, but for a concept which will hold the field together in the way "life" does with respect to biology. It may be proposed that cognitive science is the study of intelligent or adaptive behaviour. For reasons that I hope will become clear in this paper I reject such proposals. Instead, I want to suggest that cognitive science is–or at least should be–the study of consciousness.[1]

The concept of mind has always implied consciousness. I want to maintain that this is right, in opposition to the concerted effort in cognitive science to separate the mind from consciousness. As Jackendoff (1987) puts it, cognitive science makes a distinction between a computational mind and a phenomenal mind. Since all the action supposedly takes place in the computational mind, there is no role to play for the phenomenal mind: experiences *qua experiences* do not matter. Consciousness is considered to be an epiphenomenon, something without causal efficacy. Against this, I claim that epiphenomenonalism misconceives the mind. Epiphenomenalists–virtually all cognitive scientists–treat the (conscious) mind as something "over and above" the body, or as something "inner" to it. But, like life, consciousness is not something separate. Mind is an *aspect* of certain bodies, not a part. This is part of what it means to say minds are embodied. As a point of terminology: I use the terms *cognizer*, *mind-body*, and *experiencer* interchangeably, depending on the precise point I am trying to make.

Cognitive scientists either want to *explain* the way cognizers perceive, know, understand, desire, intend, communicate, etc., or they want to know how to *build* systems that can do all that. This makes it quite understandable that their focus tends to be on the mechanisms that give rise to intelligent behaviour and not on such a difficult phenomenon as consciousness. However, it does not follow that these are completely different issues, or that the notion of a computational mind is even coherent. The reason is that

[1] Note that I am using a very basic notion of consciousness. It does not imply self-awareness, it does not even necessarily imply "consciousness *of*" in a full sense. It is just the having of experiences, like pleasures and pains.

there has to be something that validates the use of *mental* terms. Usually this is not taken to be consciousness, but the fact that *content* (or meaning) is involved. We need to ask the question how something comes to have content.

When I spoke of content earlier, it was in the context of the consciousness of and understanding by a whole cognizer. This is usually referred to as *personal level content*, but because personhood is a much stronger notion than cognition, I prefer to use the neologism *organismal content* instead. Likewise I replace sub-personal content with organal content[2], hoping to get across that it is content carried by subsystems of a whole organism. Mental terms, including organismal content, are applicable to whole cognizers only. It is important to realise what this means: there is no cognition at organal levels. Parts of cognizers, even when they are functional parts, do not understand, intend, represent, etc., because they are not themselves experiencers. Whatever mechanisms and structures underly cognition, they are not cognizing. There is no unconscious cognition.[3] In another paper (Lemmen, 1994a) I argue from this that the notion of mental representations is incoherent. Part of that argument is also that if a representation is to be used *as* a representation, it has to be external to the representation user (i.e. cognizer).

Organismal content is "what is available in experience". It is for this reason it is *intrinsic* content. Experiences cannot be but meaningful to the experiencer, otherwise they would not count as experiences. It is because cognizers have experiences that things *matter* to them, that they have *reasons* to do one thing and not another. Without experiences one event is as meaningless as the next. Thus, content is essentially intertwined with consciousness. But since most approaches in cognitive science treat consciousness as epiphenomenal, they treat content as epiphenomenal too! Hence cognitive science appears to be neither about consciousness nor content. How can it claim to be about cognition?

In summary, a science of the mind should take experiences seriously (see also Lemmen, 1994b). Only experiencers engage in cognition. We should reject views that lead us to treat consciousness as epiphenomenal, like functionalism or the idea that content can be treated independently of consciousness.

2 Cognitive Subsystems and other Common Fallacies

The reader may think it problematic to organise a science around such a difficult concept as consciousness. Wouldn't we be better off studying systems that behave intelligently or adaptively? A first response to this is to point out that biology is not the study of things that behave *as if* they are alive. It is certainly true that an entity's behaviour is of vital importance with respect to whether or not we have reason to claim it is an experiencer (or alive). Its behaviour provides us with a way of finding out what it is. But behaviour isn't everything: if a robot behaves as if it knows what it is doing, but we find out it is nothing but a giant look-up table hooked up to some hardware, we will take back any mental ascriptions we may have made. More generally, as Searle (e.g., 1992) keeps pointing out, ontology ("what is it?") and epistemology ("how to find out about it?") are not the same thing.

A widespread source of the rejection of consciousness as the defining characteristic of cognition is the atomist conviction that wholes are made up out of parts that somehow *constitute* the wholes. This conviction leads cognitive scientists to try and decompose cognitive systems into *cognitive* subsystems. But since such systems could only be cognitive in virtue of being experiencers, a move like that is not warranted in the case of cognizers. In the words of McDowell (1994), the underlying processes *enable* cognition, but they do not *constitute* it. This also defies the popular philosophical project of "naturalizing content", which tries to show how things can *objectively* have content, independently of any *subjects*, i.e., experiencers. I will not discuss this project head-on, but will merely point out some of the mistaken ideas surrounding it and making it look plausible.

[2]These terms were originally coined by Ron Chrisley.

[3]I possibly need to be as explicit as possible about what I mean here. I am claiming that there is no unconscious understanding, desiring, intending or representing. I am not denying that there are internal processes which we do not experience but are nevertheless essential to our cognition. Such processes may be said to enable cognition and they are perfectly valid objects of cognitive scientific research. However, we should be somewhat wary of calling them cognitive, unless we would be prepared to also call metabolic processes cognitive.

First, the fallacy of taking metaphors literally. When we are trying to get an understanding of how some things work, it can be quite useful to say that certain events "mean" something. For example, we can say that smoke "means" fire or that the cigarettes in the ash-tray "represent" the fact that uncle Harry has just payed a visit. In a similar way we can talk about "meanings" and "representations" in car engines or control systems (like brains). But these are only metaphorical meanings and representations. We use these notions merely to indicate that we are aware of a reliable and causal correlation between two events.

Second, the fallacy of objectifying representational content. We often treat words as if they have objective meanings. However, what a word means depends on the ways it is *used* by the language community. In general, representational content is not intrinsic but dependent on the organismal contents of representation users. We sometimes forget this, because we take our own content for granted. The important implication for cognitive science is that, although it is often supposed to be self-evident, it is not true that content implies representation. Organismal content can't be explained in terms of representations, because we have to explain representational content in terms of organismal content.

Third, although it is relatively obvious that the consciousness of a cognizer is not to be explained in terms of lesser consciousnesses of its parts, this is not so obvious when the notion of intelligence is substituted for consciousness. In fact, it is a well-established strategy to try and show how the intelligence of the whole agent arises out of the interaction of the intelligences of sub-agents. The idea is that by descending the resulting hierarchy, the intelligence of the whole agent is gradually discharged. The process is supposed to bottom out at the level at which the tasks are so simple we can readily see how they can be performed by material entities, like neurons. However, as Searle (1992, pp. 55-7) points out, this strategy only looks plausible because we oscillate between two interpretations of what it means to be "intelligent". The strategy relies on a purely behaviouristic interpretation: being intelligent means behaving in an intelligent way. Under this interpretation we would have to say that neurons are a little bit intelligent. From a purely behaviourist-cum-functionalist perspective, neurons cannot be denied their fair share of intelligence; after all, they perform very complex "computations". We know this is preposterous. Neurons are not intelligent in the least, they have no idea what they're doing. So we have oscillated to the second interpretation of intelligence, the one that says that intelligence is a property of whole cognizers only. Hence, the initial decomposition of the whole agent in sub-agents was a mistake we shouldn't have made in the first place.

When we ascribe intelligence to another cognizer we do so because we have some appreciation of what it would take *us* to perform the way it does: a certain understanding and some mental skills. In other words, we implicitly suppose that the entity to which we ascribe intelligence, is the kind of creature we are. We assume we are dealing with *somebody*, i.e., we are again taking organismal content for granted. If it turns out not to be there, we transfer our admiration and intelligence-ascription from the original entity to its designer(s), even when it is a designer only in a metaphorical sense, like Mother Nature.

Fourth, and this may well be a major source of the previously discussed "evils", there is the common idea that what we are conscious of is just the tip of the cognitive iceberg. Since there is supposedly so much unconscious cognition, it has to be assumed that there is *content* that does not depend on consciousness for being content. It is the putative content at this lower level that many philosophers try to naturalise. I do not have the space here to comprehensively discuss the reasons for thinking mistakenly there must be unconscious cognition. But among them are the following ideas: the mind as an "inner" entity, memory as a store-house, content implies representation, and "if the input to the system is meaningful and the output is meaningful, then all the processes in between must be meaningful as well," even when we don't experience them (see also Searle, 1992, p. 246). These ideas should have lost much of their force, however, in these days of connectionism and dynamical systems approach. "If the brain can cross complex cognitive gaps without passing through intermediate representational states, then we are no longer compelled to posit unconscious representational processing to explain the data" (Lloyd, 1991, p. 453; "representation" is to be read metaphorically, of course).

3 Conclusion

Cognitive science is a science of mind, hence of consciousness. Since a cognizer is a whole–a mind-body, not an aggregate of a mind and a body–it is imperative that explanations of cognition should be neither in purely physical or behaviouristic terms nor in purely mentalistic terms. In the former case consciousness and content are treated as epiphenomenal, in the latter the mind is wrongly treated as disembodied. In order to give both experiences and the body their due, we need to reject all assumptions that lead to the view of the mind-as-inner or epiphenomenalism.

In ordinary language, representations are talked of as things *outside* of cognizers. However, when we talk of mental representations, such a reading leads to the conception of the conscious mind as an inner *part* of the cognizer. Usually even a part that makes no difference (epiphenomenalism). It is because of such dualistic implications that cognitive scientist have tried to establish a "respectable", technical reading instead, which says something like this: 'the contents that figure in experience are a selection of contents at an organal (i.e., sub-organismal) level. The vehicles which carry these organal contents are what we call mental representations'. However, such gerrymandering with language does not answer the philosophical question whether there *really* are mental representations. It merely leads us to describe the internal workings of a cognizer *as if* there were representations in there. But in the absence of proper representation-users in there, such talk is merely "as-if" talk (Lemmen, 1994a). But there is a deeper problem, which is the implicit assumption that organismal content is to be explained in terms of organal content. This is getting things exactly backward. It is organismal content that is intrinsic, organal content is metaphoric. Hence, a major consequence of the arguments in this paper is the rejection of mental representations. Similar views are held by Searle (1992) and McDowell (1994).

The processes and structures that underly cognition are not themselves cognitive. Organismal content is not built out of organal contents, because organal contents are "contents" only through the metaphorical story of how one part of the brain "talks to" another part. This is an extremely useful metaphor, but it is a metaphor nonetheless. As McDowell puts it, "The 'as if' content that is usefully deployed at the lower level helps make intelligible the genuine content that appears at the higher level by way of 'enabling' explanations, not as somehow constituting that content" (McDowell, 1994, pp. 201-2). Real content is only to be found at the organismal level. This has important consequences for cognitive science. One is that it "radically alters the ontology of cognitive science explanation by *eliminating a whole level of deep unconscious psychological causes*." (Searle, 1992, p. 237, original emphasis). This will, albeit indirectly, have its consequences for those who study the enabling processes, because it changes the whole perception of what cognition is. Another positive consequence of the rejection of unconscious cognition, when combined with the taking seriously of the whole of our experiences, is that it "enables us to repossess the phenomenology of perception" (McDowell, 1994, p. 204). Furthermore, there is the surprising conclusion that maybe common sense was right all along: experiences really make a difference.

Reference

Jackendoff, R. (1987). *Consciousness and the Computational Mind.* MIT Press, Cambridge, MA.

Lemmen, R. (1994a). A dissolution of the representation debate. In Smithers, T., & Moreno, A. (Eds.), *On the Role of Dynamics and Representation in Adaptive Behaviour and Cognition*, III International Workshop on Artificial life and Artificial Intelligence, pp. 177–179 San Sebastian, Spain. Universidad del Pais Vasco.

Lemmen, R. (1994b). Taking experiences seriously. *Psycoloquy*, *5*(28). Psycoloquy is an electronic journal at ftp://princeton.edu/pub/harnad/.

Lloyd, D. (1991). Leaping to conclusions: connectionism, consciousness, and the computational mind. In Horgan, T., & Tienson, J. (Eds.), *Connectionism and the Philosophy of Mind*, pp. 444–459. Kluwer, Dordrecht.

McDowell, J. (1994). The content of perceptual experience. *The Philosophical Quarterly*, *44*, 190–205.

Searle, J. (1992). *The Rediscovery of the Mind*. MIT Press, Cambridge, MA.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Problems Caused by Ineffective Communication in Requirements Engineering

Amer Al-Rawas
ameral@cogs.susx.ac.uk

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

Abstract Large software projects involve many participants exchanging information through complex and recursive interactions. Effective communication is of vital importance throughout the project life cycle and particularly during the early phase of requirements specification. The various stakeholders must be able to communicate their requirements to the analysts, and the analysts need to be able to communicate the specifications they generate back to the stakeholders for validation. This paper describes some of the problems of communication between disparate communities involved in the specification activities.

1 Introduction

It is widely recognised that communication problems are a major factor in the delay and failure of software projects (e.g. see Curtis, Krasner, & Iscoe, 1988). This is especially true of "socio-technical" software systems, which must exist in a complex organisational setting. Communication is at its worst during the early phase of requirements specification where user and developers, in most cases, meet for the first time.

Requirements specifications for software systems are based on domain knowledge, be it technical, functional, administrative, or social. Ideally, the requirements team members are selectively recruited so that both the levels and distribution of knowledge within the team cover all aspects of the domain. However, this is seldom the case because of knowledge shortfalls such as the thin spread of application domain knowledge in most organisations (Curtis, Krasner, & Iscoe, 1988). In general, individual members do not have all the knowledge required for the project and must acquire additional information before accomplishing productive work (Walz, Elam, & Curtis, 1993). Knowledge acquisition and sharing can only be achieved through effective communication between the various stakeholders (e.g. sponsors, management, end-users, developers, etc.).

A great deal of information is exchanged amongst these communities throughout the project life cycle and in particular during the process of requirements specification. The interactions are based on communication channels provided by various software engineering methodologies. These channels provide a restricted one-way communication in the form of documents. This paper describes the communication difficulties and the problems caused by the inability to communicate effectively during the requirements specification phase of software projects.

2 Research Method

A combination of learning, data gathering and analysis techniques were applied to investigate the communication problems, their causes and consequences. The two main sources of information were the literature and an empirical study that was performed as an integral part of this research. The ever growing

literature on software engineering, computer supported co-operative work and related social science areas was surveyed to gather information about the software development problems, particularly those that occur in the early phases, and the sort of tools and techniques employed to overcome these problems.

The empirical work was carried out in two stages. The first consisted of informal interviews and observations to establish some knowledge about practices and methodologies of both developers and their customers. These interviews concentrated mainly on the communication channels between agents participating in any software development project, as well as on the problems that can be attributed to the ineffectiveness of those communication channels. Other management and technical issues were also discussed. Most of these interviews were taped for further analysis and reference. The second stage of our empirical work is based on two questionnaires; one for clients and one for developers. These questionnaires were designed to address some issues that were not addressed in the first stage.

To ensure representative coverage, our subjects included users and developers of various levels of experience, qualification and background. The users included some who had just had a software system installed and some with an ongoing project. Their work areas covered a combination of civil services and purely business environments. Their experience with computers ranged from absolute computer illiteracy to reasonably qualified expertise.

The developers were all involved in either developing a new software system or maintaining an existing one. Some were also involved in the provision of hardware systems. Their working area covered all aspects of software development from requirements gathering through to maintenance and project management.

3 One Way Communication Channels

In many ways, software engineering methodologies are communication methodologies. Much emphasis is placed on the notations used to convey information both within the development team and to the various stakeholders. Ideally, the channels of communication between these various communities would be perfect, so that all knowledge is shared. In practice, it is expensive and time-consuming to support extensive communication between the communities, and the channels are restricted to one way communication in the form of specification documents.

In most software development projects there is an implicit "over-the-wall" model: at each stage in the project, a specification is thrown over a wall to the next team who are waiting to proceed with the next phase. The metaphorical wall is sometimes encouraged by management practices, but more often is merely a result of the practicalities of co-ordinating a large team.

4 Informal Communication Between Stakeholders

Organisations are traditionally described in terms of an *organisational chart* which is often the first thing handed out to anyone inquiring about the structure of the organisation. However, many important power and communication relationships are not represented in the organisational chart. Henry Mintzberg (1979) makes an analogy between the organisational chart and a road map, where the map is invaluable for finding towns and their connecting roads, but it tells us nothing about the economic or social relationships between the regions.

Our empirical study showed that many difficulties are caused by unexpected interactions between elements of the system, be they software modules or humans. In spite of the time and effort spent on studying organisational structures and the flow of power and information through them, our subjects admit that they can never account for all possible interactions and often have had to backtrack as a result of discovering a new relation or line of communication that has to be incorporated into the system. These interactions are often too complex to be traced or regulated.

5 Unstated Assumptions

There is no trivial way to elicit background assumptions, for the very reason that they are background assumptions: they are often made unconsciously. One cannot just ask "what assumptions have been made?", as this will receive at best a vague and generalised answer. Many of the assumptions will be mundane or trivial, and will already be shared by all the communities. Capturing and representing these is not only wasted effort—it also reduces the effectiveness of documentation by unnecessarily increasing verbiage.

However, some assumptions can alter the interpretation of the problem and/or misguide the solution. Thus they must be recorded and communicated to the appropriate participants for acknowledgement and clarification.

Scenario

The following real world scenario took place during a software development project for a large civil service institution (C_1). The project is used as a part of the empirical work for our research into the nature, causes and consequences of communication problems.

In this scenario, the analyst asks a user for an identifier that can be used as a unique key for customer records. The user is not quite sure what the analyst means. The analyst explains that it is something that he can use to identify customers.

User: How about the customer's name?

Analyst: No, we can not use the name, because name duplicates may occur, unless we use a full name field (3 forenames followed by a surname) which will slow down the search operation. How about the passport number?

User: Yes, in fact we ask for the applicant's passport number in our current application form.

The analyst considers this problem solved and bases his following work on it. A few weeks later, the issue was raised again during a requirements review meeting with the head of *Data and Statistics* department (HDS) in the client organisation. In addition to his domain knowledge, HDS has a good background of data processing and management. He senses a problem with using the passport number as a unique primary key, hence he asks for time to investigate this issue personally.

HDS made the appropriate contact with people in the *Applications Section* where forms are prepared and filled by customers. He also made contacts with the passport and immigration authorities. The conclusion of his research was that a passport number can not be used as a unique key. During the next review meeting, HDS explains that they only used passport numbers as an additional identification key to prevent problems in the case of more than one customer have the same full name. He adds, passport numbers change when replaced by a new one which can happen for a number of reasons. At this stage, the analyst realises that a single identifier will not work and starts to re-investigate the possibility of using a combination of two or more keys to identify customer records uniquely.

The above scenario illustrates some of the issues discussed earlier in the section and shows the power of inter-personal communication in the form of review meetings. The delay in communicating the analyst's assumption to the more informed members of the users team costs the analyst purchase time in backtracking some of the decision that were based on that assumption.

6 Inconsistency

Standard software development cycles rely heavily on documentation as an exit condition in moving from stage to stage. Documents are also used for communicating ideas and exchanging information between various stakeholders. Consequently, a colossal amount of paper work is generated. These documents

often consist of sections written in different notations by different authors with different styles which make it almost impossible to maintain consistency.

For example, the recipient of a service from one of the projects examined in our empirical work was referred to as citizen, client, customer, applicant, candidate and property-owner by different participants who contributed to the production of the specification documents. All these names were used to describe the same entity. The names used reflected the concerns of each stakeholder.

7 The Notations War

Many clients do not understand computer terms. Analysts, software designers, and programmers do not understand the clients business terms. In general, the different communities involved in the specification process prefer different types of notation, and various people will be unfamiliar with various notations. For example, a user would not want to learn to read formal specification languages, but the programmer may require these to obtain an appropriate level of detail.

Most users express their requirements in natural language. Then it is the job of the analyst to translate requirements statements into some kind of representational objects in a domain model. However, in many cases, due to time constraints, analysts pass raw natural language requirements to programmers. A programmer who was interviewed in our empirical study complained that he often has to read large amount of text in order to understand a single requirement which could have been represented very concisely using a diagram or a formal notation. In one case he had to read over a page of text to understand the requirements for the database screen layout for a particular database form. This, he said, could have been represented more concisely by drawing a diagram which indicate the required dimensions of each section of the screen.

8 Poor Traceability for Requirements and Design Rationales

Requirements Traceability refers to the ability to describe and follow the life of a requirement, in both a forward and backward direction (Gotel & Finkelstein 1993). Requirements Traceability is vital for all phases of the software development cycle to aid reasoning about requirements and justify changes, thus improving the quality and cost effectiveness of software development and maintenance.

Despite growing numbers of specialised tools which support requirements traceability, their use is not widespread, and requirements traceability problems are still cited by practitioners who do not use them (Gotel & Finkelstein 1993). In fact, none of the practitioners we interviewed used requirements traceability specialised tools and those who used the more general CASE tools were not able to see any major improvements in requirements traceability. This is due to the constraints imposed by many of the CASE tools, the time and effort put into following their strict methodologies, and their limited support to the early stages of requirements specification.

Rationales of design decisions are rooted in the requirements specification. Many design decisions involve trade offs between competing requirements. The decision taken might not be the best solution, but one that is acceptable to all parties. Information about theses decisions and the rationales behind them is crucial for the later phases of the software development, particularly the maintenance phase. Because of the limitations of the conventional software engineering methodologies and notations, many of the design rationales go unrecorded.

9 Conclusions

There are a number of pitfalls in trying to make effective use of restricted communication channels. One of the dangers is that each community interprets things in the light of their own background assumptions. This is especially problematic with non-interactive communication, such as specification documents, where there is no opportunity to check that the reader has interpreted them as was intended.

The problems described in this paper provide the motivation for my PhD research. My research aim is to come up with an approach that will help in facilitating better communication without the need to introduce new methods or notations. It is hoped that such an approach will overcome some of the problems described in this paper.

References

Curtis, B., Krasner, H., & Iscoe, N. (1988). A Field Study of the Software Design Process for Large Systems. *Communications of the ACM*, 31(11).

Dasgupta, S. (1991). *Design Theory and Computer Science.* New York, Cambridge University Press.

Easterbrook, S. M. (1991). Resolving Conflicts Between Domain Descriptions with Computer-Supported Negotiation. *Knowledge Acquisition: An International Journal*, 3, 255-289.

Easterbrook, S. M. (1993). Domain Modelling with Hierarchies of Alternative Viewpoints. In *Proceedings, First IEEE International Symposium on Requirements Engineering*, San Diego, California, 4-6 January 1993:

Finkelstein, A. (1991). Reviewing and Correcting Specifications. In *The Fourth Annual Conference on Computers and the Writing Process*, University of Sussex, Brighton, U.K.

Finkelstein, A., Kramer, J., Nuseibeh, B., Finkelstein, L., & Goedicke, M. (1992). Viewpoints: a framework for integrating multiple perspectives in system development. *International Journal of Software Engineering and Knowledge Engineering*, 2(1), 31-58.

Finkelstein, A. C. W., Easterbrook, S. M., Kramer, J., & Nuseibeh, B. (1993). Requirements Engineering Through Viewpoints. In *Proceedings of the DRA Colloquium on Requirements Engineering*, Malvern, UK: Defence Research Agency.

Gotel, O. & Finkelstein, A. (1993). *An Analysis of the Requirements Traceability Problem.* Forthcoming.

Hymes, C. M. & Olson, G. M. (1992). Unblocking Brainstorming Through the Use of a Simple Group Editor. In *(CSCW '92) ACM 1992 Conference on Computer-Supported Cooperative Work.* "Sharing Prespectives", Toronto, Canada, ACM Press.

McDermid, J. A. (1993). Requirements Analysis: Orthodoxy, Fundementalism and Heresy. In M. Bickerton & M. Jirotka (Eds.), *Requirements Engineering*. London: Academic Press.

Mintzberg, H. (1979). *The Structuring of organisations.* Prentice-Hall.

Walz, D., Elam, J. and Curtis, B. (1993). Inside a software design team : Knowledge aquisition, sharing, and integration. *Communications of the ACM*. 36(10): 63-77.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Informal Interfaces: Informality in Human-Computer Interaction

Ian Cullimore
ianc@cogs.susx.ac.uk

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

Abstract This paper discusses the notion of informality in HCI, leading to the design of informal interfaces. Such an interface exhibits tolerance in its input and variance in its output. Informal interface representations are internally composed of informal objects that are a combination of a prototype, such as a straight line, and associated informal dimensions such as shakiness and thickness. In an informal interface it is the gist of human-computer interaction, instead of a higher level of formalism, which is paramount. Internal representations of informal objects can be decomposed, manipulated, and recomposed. An example is given of a software tool that has been developed to investigate the design of informal interfaces.

1 Introduction

The aim of this paper is to outline a framework for informal interaction between a computer and a human. By using the term informal in user interface design, we do not simply mean the converse of formal, nor sloppiness. Rather we are referring to interfaces that are tolerant of the user's input (the user has flexibility in choice of action) and that show variance in their output. More specifically, in an informal interaction there is no one-to-one mapping between an input event (e.g. a menu selection) and a state change in the notional machine, nor a simple mapping between the state of the notional machine and an output presentation. Consider the classic example of a pull-down menu system; here the user is constrained to a finite set of possibilities of function choices, each of which maps onto one state, and each such state is shown as a single presentation by the interface. Conversely, an informal interface may map a number of different input events onto a single state of the notional machine, with the states representing the gist of the interaction. The states of the notional machine may be presented in a variety of forms, governed by the constraints of the internal representation and the restrictions of the output device.

2 Why Informal Interfaces?

Informality suggests a lack of precision, an easing of social or linguistic conventions. The benefits of informality include being able to express a vague or partially-understood idea, and being able to explore the essence of a concept without being committed to its eventual form. Sketching, for example, has long been recognised, from the days of Leonardo da Vinci [5], as a powerful aid to allowing the mind to run freely and creatively. The sketch serves as a framework on which the mind can build. An informal interface is an analogy of the sketch in human-computer interaction, relaxing the conventional input/output constraints of current user interfaces, in order to offer the user a more evocative and richer environment in which to react and be creative.

3 Background

The study of any area of Human-Computer Interaction encompasses a necessarily wide range of disciplines, such as computer and cognitive sciences, psychology, formal methods, art and design, and philos-

ophy. Some work relates indirectly to informal interfaces, in that researchers have been experimenting with different concepts behind the human-computer interaction by using more intuitive graphical interfaces.

3.1 Sketching and Informal Representations

In art and design there has been some work in analysing the principles behind sketching [5]. The authors explain how Leonardo da Vinci advocated the use of "untidy indeterminacies" for working out compositions, because he believed that sketches stimulated visual invention. Research from cognitive psychology [1] suggests that this is the case, in the way that a mental-imagery model is used by the human brain. It is suggested [16] that the brain can create a mental image of a sketch, and then apply processes to alter or enhance that image to useful and creative effect. Negroponte [14] notes that "Sketch recognition is as much a metaphor as fact. It is illustrative of an interest in those areas of design marked by vagary, inconsistency and ambiguity. While these characteristics are the anathema of algorithms, they are the essence of design." Lohse [12] indicates how research into cognitive models for the perception and understanding of graphs can be applied to informalism; rough-sketch representations of graphs are inherently interesting as informal objects. One of the key concepts from informal interfaces is the relaxation of the invariance of output by computers, so it is revealing to study how people perceive and process meaning from graphical information. Lohse describes a computer program UCIE (Understanding Cognitive Information Engineering), which models the underlying perceptual and cognitive processes used by people to decode information from a graph, and considers results from the analyses of bar charts, line graphs and tables. Lansdown [10] points out that computer graphics designers tend to aim at photographic realism when "convincing naturalism" might be more appropriate.

Dix [4] explores the concept of non determinism and informal reasoning in user interfaces, and proposes that deliberately introducing non determinism can sometimes help in the system by actually reducing apparent non determinism for the user in a "limited non deterministic" system, i.e. one instance of non determinism can partially or fully cancel out another.

In his paper on Informalism in Interfaces [15], Reeker studies some examples of adaptive interfaces, and analyses concepts such as representations of visual knowledge, and projecting cognitive representational structures onto computational representations.

3.2 Cognitive Dimensions

Green proposes the notion of "cognitive dimensions" [9] as a descriptive vocabulary to more accurately describe relevant interface qualities in cognitive rather than computational terms. He introduces notions such as viscosity (resistance to change), role-expressiveness and premature commitment. These concepts are further explored in [7] and [8].

3.3 Constraints and Other Implementation Methods

Leler examines one central element of informal interface construction: the application of constraint satisfaction mechanisms [11]. This concept was used much earlier in Ivan Sutherland's seminal work on the constraint-based graphical interactive system Sketchpad [18] and in Alan Borning's ThingLab [2], as later expanded by the author [3] and others such as Stefik [17].

4 Informal Interface Structures

To put a conventional structure in place for informal interfaces, we can concentrate on the three key elements of the system — input, output and the internal representation. The example above assumed that the input device would be a stylus, drawing pen-ink on a screen (probably a hand-held LCD device), with output displayed directly on the LCD. Stylus input and tablet screen output is the closest technology at the moment to the natural and informal situation of drawing or sketching on a piece of paper.

4.1 The Input Mechanism

The input mechanism in the case of a stylus is as follows: a flow of pen-ink is input from the stylus position and both displayed on the screen and also stored in an internal pixel video buffer as a bit map or vector trace. Vector traces or bounding portions of the bit map buffer can them be passed to an informal object recogniser, which creates a set of frames [13], chosen from a database of prototype frames, by first extracting a prototype for each screen object. For instance, the object recogniser might first start to best-fit a straight line through the pixel group; if this was not successful it would try the next object in turn (a second order polynomial, perhaps) until it had exhausted its embedded list of possibilities or had found a match. An analysis would then be made of the residue, i.e. the difference between the actual pixel bit map and that forming the extracted prototype. This residue would be analysed, by working through a database of informal cognitive dimensions. For instance, the bit positions would be analysed for their perpendicular variance from the prototype to extract a measure of shakiness, and so forth for other required dimensions. Hence fillers for the slots of the particular frame are constructed.

4.2 The Output Mechanism

The output mechanism works by taking as input only the frame name (e.g. "straight line") and filler values (e.g. values for shakiness, thickness and so forth) as parameters; the image builder takes this primitive and creates its image on the screen with its informal object drawing engine. So the recreated image (e.g. of a roughly-drawn straight line) may not be an exact copy of the original, but it is perceptually equivalent in that the objects it comprises will be recognised as the same objects as in the original, with the same variance.

4.3 The Internal Representation

A key element to the underlying architecture of an informal interface is the structure of the internal core, constructed using frames (as also detailed in other literature [6]), bound together by a purpose-designed spatial constraint satisfaction mechanism [11]. An operator from the input mechanism creates a change in a frame; the constraint satisfaction mechanism then propagates the changes and arbitrates constraints between all frames in the model; the resultant is then sent (again as an operator) to the output mechanism for redrawing. For instance, rotating a roughly-drawn square 45 degrees will result in another, perceptually equivalent, roughly-drawn square, although again the rotated image will not actually be an exact bit-map copy.

5 Future Research Work

Current research is being focused on a number of areas, such as the construction of internal representational structures for informal objects, the decomposition and recomposition of such objects, and the application of functional operations (such as translation, rotation and addition) on informal objects. Software is being developed to experiment with these structures, and to allow for the examination of potential advantages and disadvantages for the users of such systems.

6 Conclusions

This paper has shown an outline of a new area of research in HCI, with the introduction of informality into interaction between human and computer. By coupling this notion of informality with internal frame and object representations and spatial constraint satisfaction mechanisms, we expect to be able to demonstrate novel computer interfaces which, while not necessarily appropriate for all interactions, may often allow for greater fluidity and expressiveness.

References

1. Boden, M. The Creative Mind; Myths and Mechanisms. Basic Books, 1990.

2. Borning, A. ThingLab — A Constraint-Oriented Simulation Laboratory. Xerox PARC paper and Stanford Computer Science Department Report STAN-CS-79-746, 1979.

3. Borning, A., & Duisberg, R. Constraint-Based Tools for Building User Interfaces. ACM Transactions on Graphics, Vol. 5, No. 4, October 1986. pp 345-374.

4. Dix, A. Non Determinism as a Paradigm for Understanding the User Interface. Cambridge University Press. Paper in Formal Methods in Human-Computer Interaction. Harrison & Thimbleby (eds.) 1990.

5. Fish, J. & Scrivener, S.A.R. Amplifying the Mind's Eye: Sketching and Visual Cognition. Leonardo, Vol 23, No. 1. pp 117-126, 1990.

6. Gonzalez, A.J. & Dankel, D.D. The Engineering of Knowledge-Based Systems — Theory and Practice. Prentice-Hall, 1993.

7. Green, T.R.G. The Cognitive Dimension of Viscosity: a Sticky Problem for HCI. In Human-Computer Interaction — INTERACT '90. Diaper, D., Gilmore, G., Cockton, G. & Shackel, B. (eds.). Elsevier.

8. Green, T.R.G. Describing Information Artifacts with Cognitive Dimensions and Structure Maps. MRC Applied Psychology Unit. Paper in HCI '91: Usability Now. Diaper, D. & Hammond, N.V. (eds.). Cambridge University Press.

9. Green, T.R.G. Comprehending and Manipulating Complex Information Structures. Queen Mary & Westfield College, London 1991. Summer School on 'Theory & Methodology of Cognitive Science Applied to HCI Problems'.

10. Lansdown, J. Not Only Computing — Also Art. Computer Bulletin, Series III, 1 (Part 2), pp 18-19, 1985.

11. Leler, W. Constraint Programming Languages: Their Specification and Generation. Addison-Wesley, 1988.

12. Lohse, J. A Cognitive Model for the Perception and Understanding of Graphs. CHI '91 Proceedings, ACM Press.

13. Minsky M. A Framework for Representing Knowledge. A.I. Memo 306, MIT Artificial Intelligence Laboratory, 1974.

14. Negroponte, N. On Being Creative with Computer Aided Design. Information Processing 77, I.F.I.P., Amsterdam, pp 695-704, 1977.

15. Reeker, L. Informalism in Interfaces. Paper presented at the Workshop on Informalism, Santa Cruz, California, May 28- 30 1991. Incremental Systems Corp.

16. Scrivener, S.A.R. The Interactive Manipulation of Unstructured Images. International Journal of Man-Machine Studies (1982) 16, pp 301-313. Received 23 May 1981, revised 23 September 1981.

17. Stefik, M. Planning with Constraints. Artificial Intelligence 16 (1981), pp 111-140.

18. Sutherland, I.E. Sketchpad: A Man-Machine Graphical Communication System. Ph.D. thesis, MIT, Cambridge, Mass, 1963.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

A Proposal for the Detection of Software Interactions

Joseph A. Wood*
joew@cogs.susx.ac.uk

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

Abstract This is a position paper, which presents my views on software interactions, why they cause complications and what can be done to overcome these obstacles. A notion of interaction is defined, and the concept of "misfocusing" between groups is introduced. Finally, a brief outline of how an automatic tool might tackle this problem is described.

1 Introduction

I have lost count of the projects I have seen where various modules work in isolation, but the integrated modules do not work as expected. All such systems were designed and reviewed; so what went wrong? This paper examines what I believe are some of the underlying reasons for this situation.

Before progressing further, I should explain that I am interested in what is called Programming-in-the-Large, (de Remer and Kron, 1976). I assume that all experienced software engineers can produce small programs correctly.

1.1 Paper's Structure

Section 1.2 presents an example of a software interaction from a large industrial project. Section 2 briefly explains my usage of the term software interaction. Section 3 looks at the kinds of interactions that cause problems in large software projects. Section 4 considers how these problems might be handled, and finally section 5 outlines my future research plans.

1.2 Software Interactions: an example

Before attempting to explain interactions, let me give a real example from one project. As part of a large multi-processor Ada[1] project, one group supplied the message handling (MH) capability for use by other ("user") groups. Being written in Ada the interface to MH had been made available and the meaning of each data-type had been defined. Message handling worked in isolation; the user modules compiled with the supplied interface and ran with stubs. When the first integration test was run, no messages where successfully exchanged—why? On creating a new message, MH had calculated the length of the message plus its workspace, when the user modules filled in the message, they also filled in the length of the message, (excluding MH's workspace). In short each group had a different view of the semantics[2] of message creation. This example is in essence trivial, but it is on such interactions that projects flounder. The situation becomes much worse with legacy systems (see for example (RE, 1994)); where the requirements may not exist and the original designers are no longer available.

*Supported by a CASE award from the Engineering and Physical Sciences Research Council in association with British Telecom Laboratories (ML 464531).

[1] Ada is a registered trademark of the US Department of Defense, Ada Joint Project Office.

[2] Capturing such semantics is a non-trivial problem.

2 What is a Software Interaction?

I have loosely used the term interaction above, without giving it a meaning. This section examines the nature of interactions in more detail.

An interaction occurs whenever a 'concept' passes across a 'boundary', i.e. a 'concept' is shared between parts of a system. The exact definitions of concept and boundary are domain dependent, but I shall give a couple of examples which should clarify their meaning.

In a language such as FORTRAN IV a boundary can reasonably be taken as a block and a concept as anything shared between blocks. For example, an integer array could be used to represent a date, with the caller believing that the date is expressed in British order[3] whilst the callee believes that the date is expressed in US order[4]. Of course, this particular problem is largely overcome in modern languages by shared data types.

In an object-oriented paradigm, objects respond to messages from other objects, so objects can be regarded as the parts of our system, and concepts generally map onto messages. Several exceptions to this mapping are possible, for example when an object uses knowledge of the internal details of an other object.

The essential property of all interactions being that some concept (or understanding) has to be shared across a boundary for the project to operate as intended. I believe that there are two basic kinds of interactions, namely:-

abstract The concept is intellectual, and there may be no visible connection between the parts.

material The concept has a material existence in the implementation. That is, we can point to the implementation and say that these two components are connected by sharing this 'resource'. Resources include such things as variables, types, messages, etc..

It is clear that a material interaction must have an underlying abstract interaction, but the converse is of course false. The parts that share a concept may be fully nested, overlap or wholly disjoint.

It could be argued that a monolithic system has no parts and therefore no interactions, as I have used the term. This might be technically true of the actual implementation, but it is not true of the underlying logical model, i.e. the system must have been conceived as interacting parts even if the expression of the system is monolithic. Further, to suggest using a single 'lump' of code as the system would be to lose all the advantages of information hiding, see (Parnas, 1972; Meyer, 1988).

3 What sorts of interactions cause problems?

It may be felt that this definition of an interaction is too broad. For example, if a routine calls a local routine some concept flows across the routine boundary, and yet I claimed earlier that software engineers could produce programs-in-the-small. Certainly, a routine call is an example of an interaction, but it constitutes what I consider to be a safe interaction because the concepts are used by only one engineer.

So, what sorts of interactions cause problems? Frankly, this appears to be an open question. Small programs do not appear to present significant problems whereas large programs do. Does this arise just because different components are produced by different people, i.e. this is a project management phenomenon, or is there another explanation? I do not believe that this is just an attribute of project management, not least because it happens on a wide spectrum of projects with diverse management styles. I attribute *the cause to a break down of the shared understanding*. The MH group in the example above had their own view of message creation, whereas the user groups had their own understanding of message creation which was slightly different from the MH group's.

The problem is not that different groups have a different understanding, it is rather that they *believe* that they have a *common understanding*, which is not true in all specifics. This explains why (for example)

[3] Standard British order being day-in-month/month/year.

[4] Standard US order being month/day-in-month/year.

shared maths libraries do not cause problems; all engineers have a shared (common) understanding of `sin` (say).

How does such a situation arise? All too easily, both parties think that the concept is obvious, and the user simply wants to use the provided service. I regard this as a type of misfocusing. By focusing I mean that an engineer's attention is focused on a particular activity, and other (non-central) issues are only peripheral. Hence as long as the periphery looks OK, no further notice is taken.

Many kinds of material interaction are now checked for in the later stages of a project, for example, type checking. However, the early stages of project development are less well supported, and in particular there are no checks for abstract interactions. How does this impact on software architecture?

Firstly, in the early stages of designing a system, i.e. when the architecture is being developed, attention focuses on the kinds and uses of services, not on the specifics of an interface. That is, designers are more interested in the broad nature of components rather than the exact details of how a service is provided. For example, message handling shall provide facilities for creating and sending messages to other parts of the system. Observed defects from this focusing are "we cannot provide this service because the information is not available". Hence, interfaces are broadened or shared data areas become a little more global. This I suspect is the cause of much software rust in legacy systems.

Secondly, it is during the architectural design phase of a project that concepts are introduced, and undergo rapid evolution as ideas are resolved and the overall structure of the system is resolved.

4 Preventing Interactions

Clearly, we cannot prevent interactions, because, if an entity has no interactions, it cannot form part of the solution. So how do we detect interactions? The easiest interactions to spot are when interface packages (say) are shared between (i.e. `with`ed) other packages, because this gives a concrete indication of a shared concept. Far harder, and more important, are when a concept is passed by consensus[5], such as when a requirement is shared between two components.

Material interaction is been tackled by such things as data types, information hiding and coupling. Far harder to tackle is the abstract interaction, or the 'unknown' shared concept, for which there appears to be no support.

4.1 Automatic detection of Interactions

A partial solution to interactions would be to keep a dictionary of shared concepts, much as a data dictionary is kept. However, maintaining and accessing such a dictionary would not be easy. Certainly a manual system seems doomed to be error prone, constantly out of date and bureaucratic.

5 Future Work

Work in the reverse engineering field has been done by Calliss (Calliss, 1989), taking monolithic code, and using the technique of program slicing (Weiser, 1984), to produce modularised code. I believe that the ideas of grouping functional entities together are strongly reminiscent of the process engineers go through during forward engineering high quality software. I propose therefore to tackle the interaction problem at the design stage, by capturing a design and examining it, with a view to reporting badly structured (i.e. heavily interacting entities) and possible alternatives. I think that in order to detect abstract interactions it will be necessary to capture requirements possibly using some form of requirement tracing.

I believe that this supports and enhances the normal design review activity, by providing feedback on potential problem areas.

[5] i.e. Unspoken agreement, perhaps even subconsciously.

5.1 How Does an Engineer Benefit?

Constructing a network of the system as an evolving whole encourages the view that a system is a set of interconnected pieces, highlighting the need to identify components and how they fit together. It also serves as the front-end to a central project database. This has implications for integration, since quite often components operate in isolation, but do not co-operate as a system.

An additional contribution of constructing a network is to identify missing pieces, or inappropriate parameters. A component as yet unresolved can be referenced (upon creation), have constraints, and notes recorded about it. So that when it needs to be referenced elsewhere, or the time comes to design it, the requirements and 'expectations' are already collected in one central point not just in disconnected documents, etc.. A component has to be designed to provide a service (Meyer, 1988), in response to its invocation. A significant part of designing a component is gaining an understanding of what the component is expected to provide and how other components expect to utilise its services. By capturing this information in a database, the system will be able to identify related components. For example, if a system is required to save and load data, the components that provide this facility must have a common understanding of data formats.

As the design of a system continues a model of the component relationships must be built in an engineer's head. By positing alternative arrangements the tool will help the designers to achieve better modularity, with all its attendant benefits.

6 Acknowledgements

I am pleased to acknowledge the help and suggestions made by Betsy Cordingley and Dr. Jeremy Wilson. My special thanks go to my supervisor Dr. Rudi Lutz.

References

Calliss, F. W. (1989). *Inter-Module Code Analysis Techniques for Software Maintenance*. PhD thesis, School of Engineering and Applied Science (Computer Science), University of Durham.

de Remer, F. and Kron, H. H. (1976). Programming-in-the-large versus programming-in-the-small. *IEEE Transactions on Software Engineering*, 2(2):80–86.

Meyer, B. (1988). *Object-Oriented Software Construction*. Prentice-Hall International, Hemel Hempstead, Hertfordshire.

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058.

RE (1994). *Communications of the ACM*. volume 37, number 5, special issue on Reverse Engineering.

Weiser, M. (1984). Program slicing. *IEEE Transactions on Software Engineering*, 10(4):352–357.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Towards an Intelligent Debugging System for ML Programming Language

Changiz Delara
changiz@cogs.susx.ac.uk

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

Abstract We propose an intelligent debugging system with a knowledge representation that is independent from the programming language of a particular program to be debugged. We employ a knowledge representation technique called the Plan Calculus which has been developed by Rich [Rich, 1981b]. In order to debug a program we translate a given program to a surface plan representation, parse the surface plan to understand the overall function of the program, and use near-miss information to known plans to locate bugs and repair them.

Since our system will locate bugs by manipulating programming knowledge which is independent from any programming language, it can be used to debug programs written in any procedural language provided that the front end to the system is maintained. Our system is aimed at debugging student's ML programs and can be incorporated into an ITS (Intelligent Tutoring System) system as an Expert module or can be used as it stands.

1 Introduction

The aim of our system is not the general task of debugging which notoriously is beyond the state of the art, but the simple task of finding bugs and repairing them in student programs for known exercises. Our research proposal is implementing an intelligent debugging system for ML based on the plan calculus formalism. The plan calculus is a knowledge representation formalism for representing programs and programming knowledge such as algorithms and data structures in a procedural language. Our Bug Detection module is responsible for locating and detecting the logical errors of the given program and repairing them. In order to do that we translate (by way of a translation module) the student program into an internal representation called the *surface plan.* This is a representation of the program in terms of its control and data flow. Then we parse the surface plan by a chart parser (a plan recognition module, developed and implemented by [Lutz, 1989]) against a plan library to derive the highest level description of the semantics of the program. The plan library contains many common data structures and algorithms which are represented in the plan calculus formalism. The results of this understanding/description are stored in a *"chart"* in Lutz's [Lutz, 1992] term. In fact, there is one chart (database) for fully instantiated plans (complete chart) and another for partially instantiated plans (partial chart)[1]. There also is a reference library, which contains the high-level goal(s) for each programming task assigned to students.

2 The Bug Detection Module

In this section we shall give more explanation concerning the Bug Detection module which is the focus of our research. Suppose that a program is submitted to the Bug Detection module and has passed the translation phase and plan recognition phase. Having reached this stage, the program surface plan and the chart have been built and are ready to be used by the Bug Detection module. By also using the knowledge

[1]There are other databases for data plans and data overlays as well.

pre-stored in the plan and reference libraries, the Bug Detection module attempts to locate any logical errors in the student program. Subsequently it reports them (if any) either to the student so that he/she can correct his/her program, or to a tutoring module in a complete ITS system which tutors the student accordingly. In the debugging process for a given student program several cases can occur. The best case is that the student program achieves what it is supposed to do (i.e., it is correct). An alternative case would be that the student program achieves what is expected but there is some redundant code. Another case may be that the student program might achieve a similar task to the desired task. For instance, if the task is sorting a list of numbers in ascending order, but the student program sorts the list in descending order. Still another case that might occur is that the student program fails to achieve the highest-level goal but there is a partially instantiated plan for the goal in the chart. We consider this a near-miss situation. Yet another case might be that there is no fully or partially instantiated plan in the chart for the high-level goal. The worst case may be that none of the above case happens which implies that the knowledge of the system is not sufficient to understated the implemented program or the student program is so wrong that it does not fit in any of these categories. Here is the detailed explanation of the cases summarized above:

1. A highest-level plan (highest-level goal) of a given program referenced in the reference library exists in the complete chart and all of its sub-plans cover the student program in a one-to-one mapping.

2. This case is similar to case 1 except for the one-to-one correspondence. This means that the student program contains superfluous programming constructs which have produced extra plans at the surface plan level. In this case the Bug Detection module asserts that the given program is a correct implementation of the given programming task while reporting the redundancies.

3. The complete chart contains the student program high-level goal referenced in the reference library but differs from it. That is, there is a similar plan in the chart for the high-level goal. This difference might be wholly or partly with the input tie-points[2], output tie-points, controlling condition, or in the ordering of the input/output tie-points. In this case, the Bug Detection module will traverse the similar plan and examine its sub-plans to detect where the discrepancies have occurred. Then it deletes these sub-plans and creates new but correct corresponding sub-plans while making note of the modifications to be reported at the end of the debugging process and finally reconnects the new plans to the surrounding sub-graphs.

4. By examining the complete chart, the Bug Detection module can not find a plan corresponding to the highest-level goal. But the examination of the partial chart reveals the existence of several corresponding partial plans for the goal. Having multiple partial plans for a specific plan is due to the bottom-up parsing nature of the plan recognition module[3]. Now the module must select the "best" of those partial plans. To do this we have assigned quantitative values or scores to each plan of the plan library. For each surface-plan-level plan which we call primitive plan we assign a fixed score and store them in a database (actually in a hash table). Then the score of other plans such as temporal plans and overlays are computed out of the score of their primitive sub-plans. For the moment, the computation of score is just summing the scores of every sub-plan of a plan, but the process is recursive because a sub-plan of a plan may be a temporal plan - which it usually is. Therefore, the question which partial plan to choose is straightforwardly reduced to only selecting the partial plan with the maximum score. But the partial plan must have related tie-point(s) with the goal. Having chosen the compatible, related, and best partial plan, we choose each of the required un-instantiated sub-plan of the partial plan and debug the program for it.

5. By examining the complete chart and the partial chart, the Bug Detection module can not find a corresponding plan for the goal. In this case, it takes a corresponding rule [4] which can be used for

[2]In the plan calculus formalism we assign a number for each constant, variable, result of any primitive computation, etc. which is called a "tie-point" [Lutz, 1992].

[3]In circumstances where there is no high-level plan/goal to start with, the bottom-up approach is the most appropriate.

[4]We will use the expressions rule and plan interchangeably.

an instantiation of the goal and then takes the right-hand side of that rule as the current goal and continues the process recursively. If there is more than one rule which can implement the current goal then the Bug Detection module must select a "compatible" rule with the current goal. In this regard it applies another heuristic which says to take a compatible rule with minimum score in this category. A compatible rule is a rule which if it has an instantiated tie-point, then it must be the same as the corresponding tie-point in the current goal (if any). If the selected rule resulted in a contradiction with what has been done so far, it backtracks to select the next rule according to the above heuristic and the debugging continues.

6. If none of the cases occurred during the phases of the debugging then the Bug Detection module assumes that the student has come up with "bizarre code" [Johnson, 1986], which does not conform to any rational pattern, or he/she has introduced an algorithm which is beyond its knowledge. In this case it does not insult the student but just prints a message conveying the above conclusion.

3 An Example

For the sake of brevity we present a simple example, but the general idea is applicable to large programs as well. Suppose the following programming task is given to the students who are taking an elementary course in ML.

```
write a computer program to compute the sum of
elements in a list.
```

A typical correct solution to this problem is :

```
fun sum nil = 0 | sum (a::rest) = a + sum rest;
```

But the student has come up with the following incorrect solution:

```
fun sum nil = 0 | sum (a::rest) = sum rest;
```

In other words, the student for some reason has forgotten to accumulate the numbers generated in the recursion.[5] The surface plan for the student program which automatically is generated by the translation module is shown in figure 1. The highest-level plan (i.e., goal plan) which the Bug Detection module will look for in order to debug this program is depicted in figure 2. This goal, along with other goals, is pre-stored in the reference library. Before continuing with our explanation of the debugging process for this example, let us present plans which are involved in the goal for the student program (i.e., *sum*). This plan hierarchy is shown in figure 3. Note that the leaf nodes of the tree in figure 3 are primitive plans (i.e., surface-plan-level plans) and non-leaf nodes correspond to another plan in the plan library and the dangling arrows show the existence of other plans, but for clarity we did not include those. Surface-plan-level plans are those plans which are generated by the translation module. From the surface plan of the student program the plan recognition module found (instantiated) about forty complete plans from the plan library and stored them in the complete chart. Along with those plans several partial plans which failed to be completed are left in the partial chart. The most closely related plans for our example[6] are *cons2_+_@_ml_binrel_+_test*, *@_binrel*, *@_binrel_composite*, and *@_predicate*. There is no complete or partial plan for *sum*, *aggregate*, *reverse_iterative_aggregation*, or *@_aggregative* in the chart. There are several partial patches (plans) for *reverse_iterative_aggregation* in the partial chart.

Now we outline the debugging process of the student program. If the program had been implemented correctly, the module would have found the *sum* plan (shown in figure 2) in the complete chart. But this is not the case and therefore the Bug Detection module tries the second, third, and fourth cases as alluded

[5] For the moment we are not very concerned with students' misconceptions.

[6] There are many other plans in the chart but they are not important for the moment.

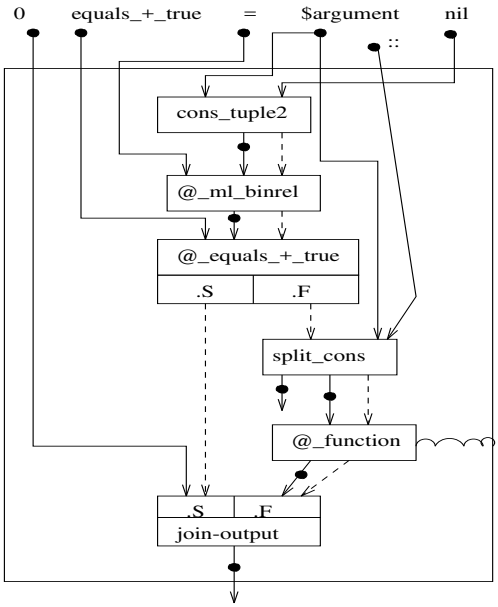

Figure 1: surface plan of the student buggy program

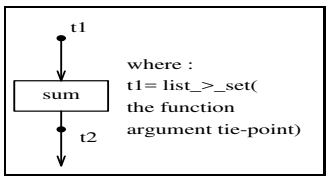

Figure 2: sum goal

to in section 2 and realizes that these are not the cases to proceed with. Then it moves to case 5 and figures out that this is the one. That is, the Bug Detection module could not find a complete or partial plan either in the complete chart or in the partial chart. As described earlier, it fetches the corresponding rules for this goal from the plan library and selects those rules which are compatible with the current goal.[7] Then it propagates any instantiated tie-points from the current goal to those rules and takes a rule. Now the right hand side of this rule becomes the current goal to proceed with. When a rule succeeds it discards the remaining rules for the current target, otherwise it tries the next rule in this category. For this example there is only one rule which is the *aggregate* rule (see figure 4) and the module sets it as the current goal and attempts to debug the given program for that goal.

Likewise, by examining the complete and partial chart for the current goal, it realizes that there is no corresponding plan for it. Consequently, it fetches the corresponding rules from the plan library. In this case there are two rules namely, *reverse_iterative_aggregation* and *iterative_aggregation*. In situations such as this–where there is no preference between the rules to be chosen–the Bug Detection module applies a heuristic which says to first try the rule which recurses in reverse manner. Therefore, it chooses *reverse_iterative_aggregation* rule[8] first and if it fails to account for the implementation of the student program and repair it then attempts the *iterative_aggregation* rule. Having done that, the current goal becomes *reverse_iterative_aggregation* (see figure 5). Any instantiated tie-points from the goal hierarchy are propagated to this current goal.

[7]Some rules have pre-assigned fixed tie-points which may not be compatible with the current goal.

[8]Because the natural way of implementing recursion in ML is not tail recursion.

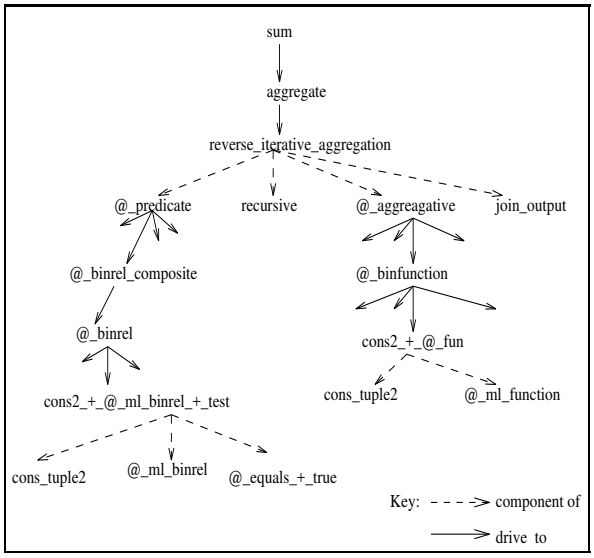

Figure 3: Plan Hierarchy for the student *sum* function

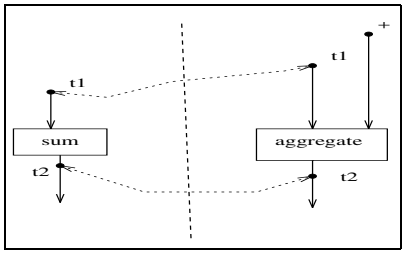

Figure 4: sum_as_aggregate overlay

By consulting the plan library again, it realizes that there is no rule derivable from this rule (see figure 3 for plans hierarchy).[9] Now the whole debugging process culminates in debugging the student program for the *reverse_iterative_aggregation* plan and debugging this case falls into case 5. For this goal, the module finds several partial plans in the partial chart and it selects the most suitable one. In order to do so, it applies a heuristic which says to choose those partial plans whose score so far is greater than or equal to half of their score when they were complete. This implies choosing those partial plans which have more instantiated sub-plans. Furthermore, between those plans choose plans that have any tie-point in common with the current goal. Still between them choose partial plans of which if *join_output* is a sub-plan then this sub-plan must not be an overlay of another *join_output*. Finally, from the recognized partial plans choose the ones whose sub-plans have no output tie-point of a global constant. Note that these factors help the Bug Detection module to reason in uncertain circumstances.

Now in our example, it chooses one of these partial plans, examines it, and realizes that it has two un-instantiated sub-plans (out of four) namely, *@_aggregative* and *recursive*. In these situations, it must debug the given program for each of the sub-plans involved. Which sub-plan to choose first is the next question to be answered. In this regard, the Bug Detection module applies another heuristic which states that it should concentrate on non-recursive sub-plans first[10] and among them choose the one with the maximum score. Note that for each plan in the plan library, we have maintained a property list in the

[9]Note that we are traversing the plan hierarchy in a top-down manner.

[10]Without debugging the non-recursive sub-plan debugging recursive sub-plan becomes very hard. Since non-recursive sub-plans affect the input/output tie-points of a recursive sub-plan.

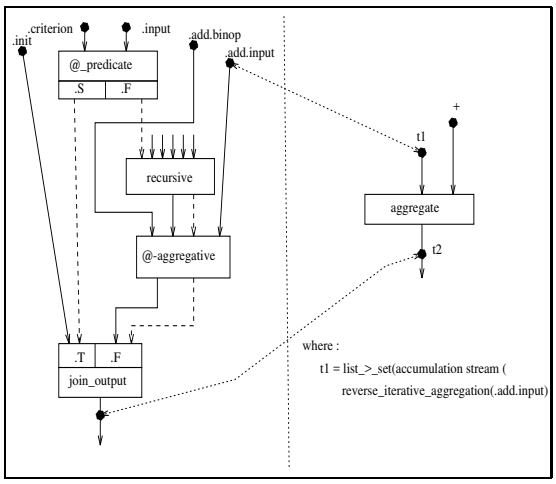

Figure 5: aggregate_as_reverse_iterative_aggregation overlay

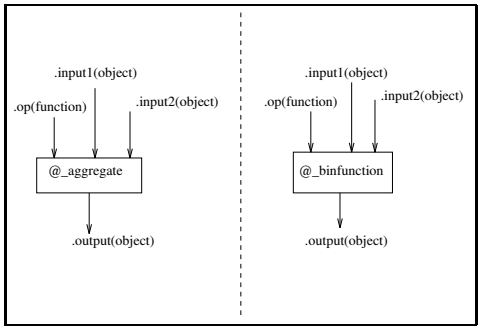

Figure 6: @_aggregative_as_@_binfunction overlay

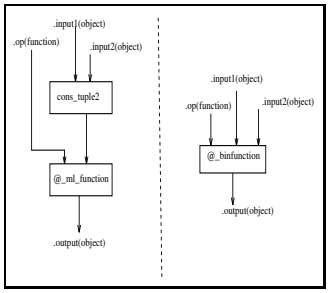

Figure 7: cons2_+_@_fun_as_@_binfunction plan

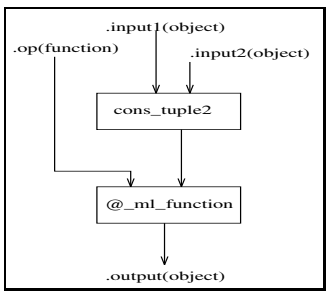

Figure 8: cons2_+_@_fun plan

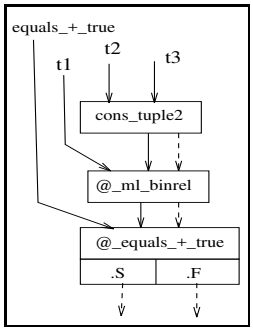

Figure 9: cons2_+_@_ml_binrel_+_test plan

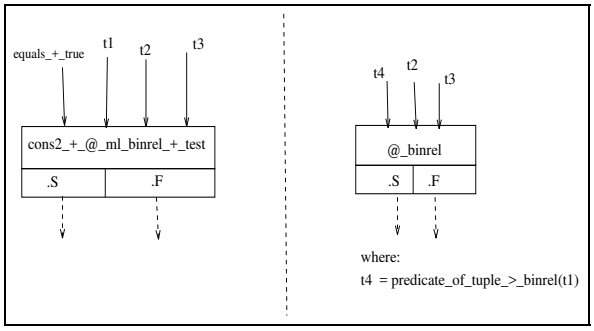

Figure 10: cons2_+_@_ml_binrel_+_test_as_@_binrel overlay

form of a hash table which contains the plan type and its score. In this case it chooses the @_*aggregative* sub-plan to proceed with. Now the current goal for the module is the @_*aggregative* plan. Debugging this falls under case 5 (because there is no complete or partial plan in the chart) which in turn directs the debugging the program for @_*binfunction*. Figure 6 shows the corresponding overlay. There are four rules corresponding to this plan in the plan library and the Bug Detection module chooses a rule from the extracted set which is related to the current target. If this choice led to a discrepancy with what has been asserted, then the module backtracks the process to select the next rule according to its heuristic again. This process continues until it succeeds or it abandons the whole process because of contradictions.

At this stage, the module selects *cons2_+_@_fun* plan (see figure 8) as the current goal and debugs the program for it. This falls into case 5 again, but this time all of its sub-plans are primitive plans. Therefore, it asserts that the student program has failed to produce such plans and it creates a corresponding plan for each of the sub-plans involved and passes them to the plan recognition module. This invocation entails the addition of the sub-plans to the complete chart as well as the instantiation of other pertinent plans such as *cons2_+_@_fun*, @_*binfunction*, and @_*aggregative*.

Now it is the turn of the *recursive* sub-plan to be debugged. This case is more tricky because it requires the deletion of an existing rule for the recursion and the creation of a modified version of the rule.[11] This involves including new tie-points which are introduced in newly created plans while omitting old tie-points (in the case of the removal of any old plan).

In order to activate the new modified recursion rule which is added to the plan library we create a primitive plan (i.e., a nape) of the type of @_*function* with the *operation* tie-point assigned to the student function name in the translation phase. Its argument tie-point is the one which was in the old @_*function* and its output tie-point is the one coming out of the *recursive* sub-plan. This activation causes the *reverse_iterative_aggregation* plan, *aggregate* plan, and *sum* plan to become fully instantiated and added to

[11] The translation module generates a new rule for each student function it processes.

the complete chart.

Since the whole debugging process is recursive, at this stage of the analysis, the debugging process completes and recursion unwinds. Since the highest-level goal (i.e., *sum*) plan is found in the complete chart, the whole debugging process is terminated. This means that the Bug Detection module found the highest-level goal but only by repairing the student program. In the final stage, the Bug Detection module reports on what has been done, for example: stating that @_*binfunction* plan was missing and has been created and added to the chart, implying that the student had missed a binary operation (i.e., +) to sum up the 'head' of the list in the way back of the recursion.

This example shows how the Bug Detection module locates and repairs the bug. For the interest of the reader we have included the graphical representation of the plans and overlays referred in figure 3. We omitted their logical definition for the sake of clarity. Interested readers concerned with the logical foundation of the plans and overlays in the plan calculus are referred to [Rich, 1981b] and [Rich, 1981a].

4 Conclusion

In this short paper we delineated the overall strategies used by the Bug Detection module. We showed how the module locates and identify the bugs and repair them. Finally we closed our report by presenting a simple example (but comprehensive in content) to give some flavour of how a bug is located and repaired by our proposed tutoring system.

Acknowledgement

I am grateful to my academic supervisor Dr. Rudi Lutz for his comments and willingness to discuss the material presented here. This research has been fully supported by the Ministry of Culture and Higher Education of the Government of Islamic Republic of IRAN.

References

[Johnson, 1986] W. L. Johnson. *Intension-Based Diagnosis of Novice Programming Errors*. Morgan Kaufmann Pub. Inc., USA, 1986.

[Lutz, 1989] R. Lutz. Chart parsing of flowgraphs. In *In proceedings of 11th Int. Joint Conf. Artificial Intelligence*, pages 116–121, Detroit , Michigan, 1989.

[Lutz, 1992] R. Lutz. *Towards an Intelligent Debugging System for Pascal Programs: On the Theory and Algorithms of Plan Recognition in Rich's Plan Calculus*. technical report, The Open University, Milton Keynes , UK, 1992.

[Rich, 1981a] C. Rich. A formal representation for plans in the programmer's apprentice. In *Proc. of the 7th Int. Joint Conf. on Artificial Intelligence, ICJAI*, pages 1044–1052, August 1981.

[Rich, 1981b] C. Rich. *Inspection Methods in Programming*. technical report no. 604, Artificial Intelligence Laboratory, MIT, June 1981.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Goal Formulation as an AI Research Issue

Remedios de Dios Bulos
remedios@cogs.susx.ac.uk

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

Abstract Although research in AI Planning systems has made considerable progress from its humble beginnings, much is still to be desired. A lot of research work remains to be explored to address several unresolved issues and problems. An important research issue in AI Planning systems that needs to be addressed, investigated, and solved concerns the problem of endowing an artificial resource-bounded rational agent with the ability to formulate its own goals, as it navigates a world that is characterizable as complex, dynamic and uncertain. This paper cites and discusses the main reasons and issues as to why goal formulation is an important research topic to be tackled in Artificial Intelligence. First, a definition of goals is given. Next, several reasons are identified as to why goal formulation is a research issue in AI. A summary and analysis of related work on goal formulation is then presented. Lastly, a list of research questions that need to be resolved is enumerated.

1 Introduction

Research work in AI Planning systems has progressively evolved from the simple operation of identifying and generating possible action sequences to achieve a set of specified goals (classical AI Planning) to the more sophisticated process of integrating the different functions of planning, execution, monitoring/control and learning. Moreover, the characteristics and environment of the domain of experimentation and application have gradually metamorphosed from simple, static and predictable to complex, dynamic and uncertain.

However, although research in AI Planning systems has made considerable progress from its humble beginnings, much is still to be desired. An important research issue in AI Planning systems that needs to be addressed, investigated, and solved concerns the problem of endowing an artificial resource-bounded rational agent with the ability to formulate its own goals, as it navigates a world that is characterizable as complex, dynamic and uncertain.

1.1 What are Goals?

In order to fully understand and address the various issues involved in the formulation of goals, a clear definition of what goals are is first needed. As used throughout this paper, goals shall refer to the end results that an agent would like to achieve or avoid. They are primarily acquired through association with other agents, objects, events, symbols and behavior. They are often multiple, occur simultaneously and may interact with each other. They are stored and organized according to a dynamic hierarchy. They are activated by the stimuli generated by the environment, and are chosen depending on the relative position they occupy in the hierarchy and their potential for realization. [Ford, 1992; Pervin, 1991; Read & Miller,1989; Koontz & Weihrich, 1988]

1.2 What is Goal Formulation?

As agents operate in complicated, dynamic and unpredictable worlds, many promising goals of varying degrees of importance and potential of realization could be acquired and detected. However, due to their resource limitations, agents need to decide wisely on what goals to respond to and when. As Pollack has said "intelligent behavior depends not just on being able to decide how to achieve one's goals, but also being able to decide which goals to pursue in the first place, and when to abandon or suspend the pursuit of an existing goal" [Pollack, 1992].

In this paper, Goal Formulation shall refer to the intelligent behavior that an agent exhibits when reasoning (and deciding) what goals to pursue and when to pursue them. It may be described as the integration of several processes, namely: detecting which goal is to be accomplished (Goal Detection), assessing the feasibility of accomplishing the goal (Goal Assessment), determining the relative position of goals in the hierarchy (Goal Prioritization), evaluating whether a newly detected goal is to be accepted, rejected, or modified and whether a currently active goal is to be pursued (continued), terminated, suspended, or modified (Goal Evaluation) and determining ways of modifying goals (newly detected or active) to suit present circumstances (Goal Modification).

2 Goal Formulation as a Research Issue

In the artificial intelligence field, most of the AI planning systems that have been designed and developed so far operate on the basis of an already existing, predetermined goal or set of goals. That is, they start with a set of goals that have been previously identified and assigned by the user.

A truly intelligent agent, whether a person or an artificial system, must be able to make the most out of any given situation, whether the situation proves to be adverse or pleasant. As the situation permits, he should either maximize his gains, or satisfice whatever he could accomplish and should be able to explore other possible goals that could be achievable in the light of present circumstances. As a rational agent and a survivor, an intelligent individual does not wind up in oblivion or in the state of inactivity should the situation become most uncooperative in achieving his current goals. He either postpones or terminates the achievement of current goals, but at the same time looks for possible opportunities and then formulates new goals to take advantage of the prevailing conditions. Should he possess prioritized multiple goals, he diverts his attention to accomplishing goals of lesser priorities.

Most AI planning systems are not capable of exhibiting a "genuine" goal-oriented or goal-directed behavior. They are not built with the capability to monitor and analyze the environment and to formulate their own goals. They lack the ability to take advantage of opportunities being presented by changing environmental conditions. They do not have the flexibility of behavior to change their focus of attention or direction when needed most. They are bereft of the ability to make a rational choice in the light of present circumstances. [Georgeff, 1990]

However, goal formulation is not merely the detection or inferring of one's goals. More than this, it involves reasoning and decision-making on which of the newly detected goals are to be pursued and which of the currently pursued goals should be continued, postponed or abandoned. To be able to reason and decide regarding the validity of its goals, the agent should possess the knowledge and behavior needed to assess the feasibility of achieving goals, to detect and analyze the consequences of both positive and negative interaction of goals, to resolve conflicts arising from negative goal interactions, and take full advantage of the promising benefits created by positive goal interactions.

In real world domains, planning agents are resource-bounded. Although they may have unlimited wishes, they have limited supply, access to, and use of resources. A planning agent endowed with the capability to formulate its own goals can enumerate or detect an infinite number of goals (achievable and non-achievable alike). However, constraints as well as limitations force it to reason, decide, and eventually choose the set of goals it considers possible and practical to achieve. It also has to consider the relationship of its goals among each other because goals can compete for the limited resources or share in the use of resources. In the light of limited resources, it also has to prioritize its goals according to their

importance, i.e. which goal should be accomplished first.

However, due to the dynamism and uncertainty of the environment, decisions that were made previously (in terms of plans and goals) may be affected and become inapplicable later. During such trying and unexpected situations, the agent must be able to decide quickly, react and respond appropriately, be able to cope up with the new constraints and conditions, regain full control of the situation, and resume its normal functions.

When present circumstances become too difficult for plans to succeed, the agent must be able to decide whether it has to continue achieving the goal (that is, by adopting another alternative plan of lesser expected value or replanning), postpone the pursuit of a goal, terminate the pursuit of a goal completely, or come up with a modified goal that is tailored to the prevailing circumstances. It should also be able to decide when to suspend the attainment of the goal temporarily, to redirect or refocus its attention to goals that must be desperately accomplished during that time.

However, a dynamic and uncertain environment does not always spell trouble for the agent. At times, various opportunities that were not present before or not predicted to come may emerge. In these situations, an agent should also have the capability to decide whether to take advantage of these opportunities or not. It has to make a good judgement of whether another goal should be formulated in order to capitalize on the merits of the situation or to just pursue the status quo.

3 Addressing the Goal Formulation Problem through AI

This section aims to present the most relevant and related literature to date that deals with the goal formulation research issues discussed in the preceding section. It discusses how some AI researchers have touched or tackled some issues concerning goal formulation.

When speaking of goal formulation, the first important question that comes to mind or needs to be determined is where do goals come from or how are they acquired? In the AI research community, this question was addressed by Schank & Abelson in their research on story understanding [Schank & Abelson, 1977]. In their book ”Scripts, Plans, Goals and Understanding”, Schank and Abelson emphasized the importance of knowing the different kinds of goals plus their interaction among each other to formulate expectations. They also stressed the need to recognize the existence of a goal for an actor in order to be able to predict his future actions. They postulated a GOAL MONITOR which recognizes the triggering of goals, interprets the nature of goals, keeps track of the fate of goals, and makes predictions about goal-related events. [Schank & Abelson, 1977]

Wilensky also recognized the need for an AI planning system to formulate its own goals. He has stressed the significance for an autonomous planner to be equipped with the capabilities of inferring ”its own goals based upon its overall mission together with the situation in which it finds itself”. [Wilensky, 1990]

In the development of the Berkeley Unix Consultant Project, Wilensky together with his co- researchers has identified the UCego as one of the components of the Unix Consultant (UC). The UCego, which was developed by Chin, determines UC’s own goals and attempts to achieve those goals. [Wilensky, Chin, Luria, Martin, Mayfield & Wu, 1988]

Georgeff, Lansky & Ingrand, in their work on the Procedural Reasoning System (PRS) have stressed the need for a planning system to have its own set of beliefs, desires and intentions. Although the PRS does not have the ability to formulate its own goals, it has been endowed with the capability to reason about its own internal state. It reflects upon its own beliefs, desires, and intentions and modifies them when the appropriate situation arises. [Georgeff & Ingrand, 1989; Georgeff & Lansky, 1990]

Lizotte and Moulin, in their model SAIRVO, also tackled some aspects of goal formulation. SAIRVO possesses components that could verify and recognize goals. The process VERIFY A GOAL ”examines the contents of the accumulations facts and facts to be inferred to find new facts (positive interactions) that will enable some of its goals” [Lizotte & Moulin, 1990]. On the other hand, the process RECOGNIZE A GOAL chooses the goals of the planner. The motivation rules which contain the goals or conclusions to

be detected given a set of premises are applied on all information. [Lizotte & Moulin, 1990]

4 Unresolved Issues on Goal Formulation

Analysis of the above cited works indicates that by and large, research on goal formulation has concentrated on the goal detection aspects. Goal detection is the process of signalling or reminding the system (planner) that it has a goal(s). The goal is usually detected when a situation or condition that gives rise to the goal is sensed by the planner. Although such a method is valid and effective, it is most probably not the only means to detect the occurrences of goals. Also, further elaboration is needed on the types of situations that emerge and how such situations give rise to goals.

Another important issue that needs further study is the reasoning and decision-making process that is undertaken when evaluating whether a newly detected goal should be accepted, rejected, or possibly modified. It is essential for the agent to detect changes that affect the acceptance or rejection of the goal during deliberation to prevent or minimize the waste of further effort and resources. It is equally significant to determine when the evaluations should take place.

Assessing the feasibility of a goal could be a long, tedious and delicate process. Given preliminary knowledge and estimates of the minimum constraints to achieve a goal, some goals may be obviously classified as ”non-achievable”. Possible problems that may arise during feasibility assessment of the goal should be investigated.

Currently held goals should also be regularly evaluated. Some goals may need to be abandoned and others be temporarily suspended. This is brought about by the changing conditions in the environment as well as the introduction of new goals into the agent's overall repertoire and hierarchy of goals.

Dealing with goal interactions is a very important aspect in goal formulation. The goal formulator must know whether negative goal interactions can be resolved and whether newly detected goals have positive interactions with other currently held goals. Although research on goal interactions has generated the interest of some researchers, the bulk of the research work is mainly concentrated on the development of mechanisms for resolving goal and plan conflicts through the application of critics [Sussman, 1975; Sacerdoti, 1977], constraints [Stefik, 1990], and meta-planning [Stefik, 1981; Wilen sky, 1983]. However research on what should be done should unresolvable goal conflicts occur is wanting.

The prioritization of goals is another issue of goal formulation that needs to be investigated. According to Schank and Abelson there is no known calculus on how to prioritize goals [Schank & Abelson, 1977]. It is therefore necessary that a study concerning the reasoning and decision-making utilized by the agent when prioritizing goals should be tackled. Vague and immeasurable terms such as ”importance” of a goal need be defined. Modifying rejected or unsuccessful goals is another issue that needs to be studied. An agent should find ways of redefining its goals if previously evaluated to be unachievable or proven to be unsuccessful.

Goal Formulation is not a lone and isolated process. It is a part of the management process system and integrated with the other processes that comprise the system. Goal formulation interacts with the process of analyzing the environment to enable the detection of goals. It interrelates with the planning process in assessing the feasibility of achieving the goals detected. It interacts with the monitoring and control and execution processes to regularly evaluate the status of the goals, that is, whether a goal should be continued to be pursued, suspended, or terminated. To ensure the proper and timely formulation of goals, the relationships with other components and the conditions that would call for the interactions should be identified and defined.

5 Conclusion

Most of the existing AI Planning systems were developed based on the working assumption that a goal is supplied to the agent (planner). However, to be truly called ”intelligent” an agent must not only be capable of knowing how to achieve its given goals. It must also have the capability to formulate its own

goals. It must reason and decide what goals to achieve and when to achieve them. It must be able to detect its own goals, assess their feasibility, prioritize them, evaluate their validity (continuation, termination, suspension, modification) and modify them in the light of present circumstances. An intelligent agent's success in pursuing its goal-directed activities will largely depend on the behavior it exhibits during the formulation of goals. These above-cited reasons justify the research issue that an agent should be endowed with the capability of formulating its own goals.

References

Allen, J. F. (1979), *A Plan-Based Approach to Speech Act Recognition*, Doctoral dissertation, Department of Computer Science, University of Toronto. Also available as Technical Report No. 131, University of Toronto.

Chin, D. N. (1988), *Intelligent Agents as a Basis for Natural Language*, UCB:CSD-88- 396, Division of Computer Science, (EECS), University of California, Berkeley, CA.

Faletti, J. (1982), PANDORA – A program for Doing Common Sense Planning in Complex Situations, in *Proceedings of the Second Annual National Conference on Artificial Intelligence*, pp. 185-188, August, PA.

Ford, M.E. (1992), *Motivating Humans Goals, Emotions, and Personal Agency Beliefs*, USA: Sage Publications, Inc.

Georgeff, M.P. (1990), Planning, in Allen, J., Hendler, J. & Tate, A. (eds.) *Readings in Planning*, San Mateo California, USA: Morgan Kaufmann Publishers, Inc.

Georgeff, M.P. & Ingrand, F.F. (1989), Decision-making in an Embedded Reasoning System, in *Proceedings Eleventh International Joint Conference on Artificial Intelligence*, pp. 972-978, Detroit, Michigan.

Georgeff, M.P. & Lansky, A.L. (1990), Reactive Reasoning and Planning in Allen, J., Hendler, J. & Tate, A. (eds.) *Readings in Planning*, San Mateo California, USA: Morgan Kaufmann Publishers, Inc.

Koontz, H. & Weihrich, H. (1988), *Management* 9th Edition, New York: McGraw- Hill.

Lizotte, M. & Moulin, B. (1990), A Temporal Planner for Modelling Autonomous Agents, in Demazeau, Y. & Muller, JP. (eds.) *Decentralized A.I.*, pp. 121-136 Amsterdam, Netherlands: Elsevier Science Publishers.

Pervin, L.A. (1991), Self-Regulation and the Problem of Volition, in Maehr, M.L. & Pintrich, P.R. (eds.) *Advances in Motivation and Achievement* Vol. 7, Greenwich, Connecticut: JAI Press Inc.

Pollack, M.E. (1992), The Uses of Plans, *Artificial Intelligence*, 57, pp. 43-68.

Read, S.J. & Miller, L.C. (1989), Inter-Personalism: Toward a Goal-Based Theory of Persons in Relationships, in Pervin, L.A. (ed.) *Goal Concepts in Personality and Social Psychology*, Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Schank, R.C. & Abelson, R.P. (1977), *Scripts, Plans, Goals and Understanding*, New York, USA: Lawrence Erlbaum Associates.

Stefik, M. (1990), Planning with Constraints (MOLGEN: Part 1), in Allen, J., Hendler, J. & Tate, A. (eds.) *Readings in Planning*, San Mateo California, USA: Morgan Kaufmann Publishers, Inc.

Stefik, M. (1981), Planning and Metaplanning (MOLGEN: Part 2), *Artificial Intelligence*, 16, pp. 141-169.

Sussman, G. J. (1975), *A Computer Model of Skill Acquisition*, New York: American Elsevier.

Wilensky, R. (1990), A Model for Planning in Complex Situations, in Allen, J., Hendler, J. & Tate, A. (eds.) *Readings in Planning*, San Mateo California, USA: Morgan Kaufmann Publishers, Inc.

Wilensky, R. (1983), *Planning and Understanding A Computational Approach to Human Reasoning*, Reading, Mass.: Addison-Wesley Publishing Company

Wilensky, R., Chin, D.W., Luria, M., Martin, J., Mayfield, J., & Wu, D. (1988), The BERKELEY UNIX Consultant Project, *Computational Linguistics*, 14, No.4.

Wood, S. (1990), *Planning in a Rapidly Changing Environment*, DPhil Thesis, School Of Cognitive and Computing Sciences, University of Sussex, Brighton, UK.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Structural Extensions for the ELKA Model

Ricardo Garza M. *
ricardom@cogs.susx.ac.uk

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

Abstract In this paper the forked link is used in the construction of extended relationships for the design of ELKA database conceptual model. The properties of such relationships are defined as structural extensions to the semantics of the model. The introduction of these extended relationships allow the capture more meaning during the conceptual database design process. The specification of generalization/specialization hierarchies may be perceived as a particular case of the extended relationships.

1 Introduction

This paper presents the main characteristics of the *forked link*, as an extension to the semantic modelling power of the ELKA (**E**ntity, **L**ink, **K**ey, **A**ttribute [1]) model approach for the design of conceptual database models, and some properties of the extended relationships that can be modelled with them. The main contribution of the forked link consist in allowing the specification and representation of objects with complex structures that can not be adequately modeled in conventional modelling techniques such as the Entity Relationship (ER) or the ELKA models.

In the model a relationship between entities is is establish with the *link* as a reference made by an entity to another using a key of the referred-to entity. A *link class* is a set of links which are referenced from all entities in one entity class to entities in another entity class. There are three types of link classes in the model, the 1-to-1, the weak-n-to-1 and the strong-n-to-1 link classes described by functions from an entity class A to an entity class B with the followings characteristics: In a **1-to-1 link class** for every entity 'a' in A there exist exactly one entity 'b' in B, and for every entity 'b' in B there exist *zero or one* entity 'a' in A; in a **weak-n-to-1 link class** for every entity 'a' in A there exist exactly one entity 'b' in B, and for every entity 'b' in B there exists *zero, one or more* entities 'a' in A; and, in a **strong-n-to-1 link class** for every entity 'a' in A there exists exactly one entity 'b' in B, and for every entity 'b' in B there exists *one or more* entities 'a' in A.

The graphical representation of the ELKA model uses lines to represent link classes. The 1-to-1 link class uses an arrowhead pointing to the front of the link, indicating the direction of the function. The weak and strong n-to-1 link classes use diamonds in the back of the link indicating the 'n' side of the link. A white diamond is used for the weak link and a black for the strong link.

We conceive the forked links as a special link constructed by merging the front side of two or more link classes into a common front side while its branches are the back sides of the merged links. The *grade* of the fork is given by the total number of link classes merged together to form the forked link class.

The entity class at the front of the fork is called the *superentity class* while every entity class at the back of the fork is called *subentity class*. The *forked link class* is defined as a set of forked links which are referenced from entities at the back of the entity classes to entities at the front entity class.

*Supported by Consejo Nacional de Ciencia y Tecnología and Instituto de Investigaciones Eléctricas, México.

2 Properties of the Forked Links

We can conceive three independent properties for these newly created forked link classes. The first one is a dichotomous property related to its possibility of sharing, i.e. a forked link may be overlapping or disjoint; the second one distinguishes also two possibilities for the existence dependence, i.e. a forked link may be strong or weak; and the last one, the multiplicity, is a three-valued property derived from the semantics of the ELKA links merged to form the forks, i.e. a forked link may be single, multiple or mixed. The characteristics and description of these three independent properties are as follows:

2.1 Overlapping vs Disjoint Forks

This property denotes whether the forked link can or cannot allow the migration of the key attribute values from the superentity class through one or more of its back sides. In an overlapping forked link, the link allows the migration of its key attribute values from the superentity class through zero, one or more of its n ($n \geq 2$) back sides. It is disjoint if instead it causes the migration of any key value from the front side of the fork to at most one of its back sides.

Graphically, an overlapping forked link is shown with the use of a diamond at the merging opposite direction at the merging point.

2.2 Strong vs Weak Forks

This property denotes if each key attribute value at the superentity must or must not migrate through the fork to at least one of its back sides. If the fork is strong it requires the migration of every particular key value from the superentity through the forked link class to at least one of its back sides. On the other hand, if the fork is weak, the migration of every key attribute value from the superentity class is not required.

Graphically, a strong forked link is show with a black diamond/triangles at the merging point, while the weak fork is represented with a white diamond/triangle, in direct analogy with the ELKA graphical representation for the strong and weak n-to-1 link classes [1]

2.3 Single, Multiple and Mixed Forks

This property reflects the semantics of the basic ELKA link classes that have been merged to construct the forked link classes. A forked link is said to be single if all merged links were 1-to-1 link classes; the fork is multiple if all merged links were weak-n-to-1 link classes; and finally, the forked link is said to be mixed if it has been constructed by merging both 1-to-1 and weak-n-to-1 link classes.

Graphically, a single forked link is represented using an arrowhead pointing to the superentity at each of its branches; a multiple forked link is depicted using white diamonds at its back sides, and a mixed forked link uses both arrowheads and diamonds at their back sides, i.e. white diamonds at the back o the weak-n-to-1 links merged and arrow heads at the back of the 1-to-1 link merged.

2.4 Forked Link Synthesis

From the combination of the three independent properties it is possible to synthesize twelve forked links, whose the graphical representation is shown in Figure 1. In the figure every forked link is depicted showing the superentity class. The Fork Cardinality is defined as the minumum and maximum number of subentities that may be selected to transmit the front entity key value at any particular time.

3 ELKA Extended Relationships with Two or More Subclasses

With the introduction of the forked link, the model is now fitted with the constructs required to represent two main sets of extended ELKA relationships: one composed of relationships at the front of only one subentity class, and the second constituted by relationships at the front of two or more subentity classes.

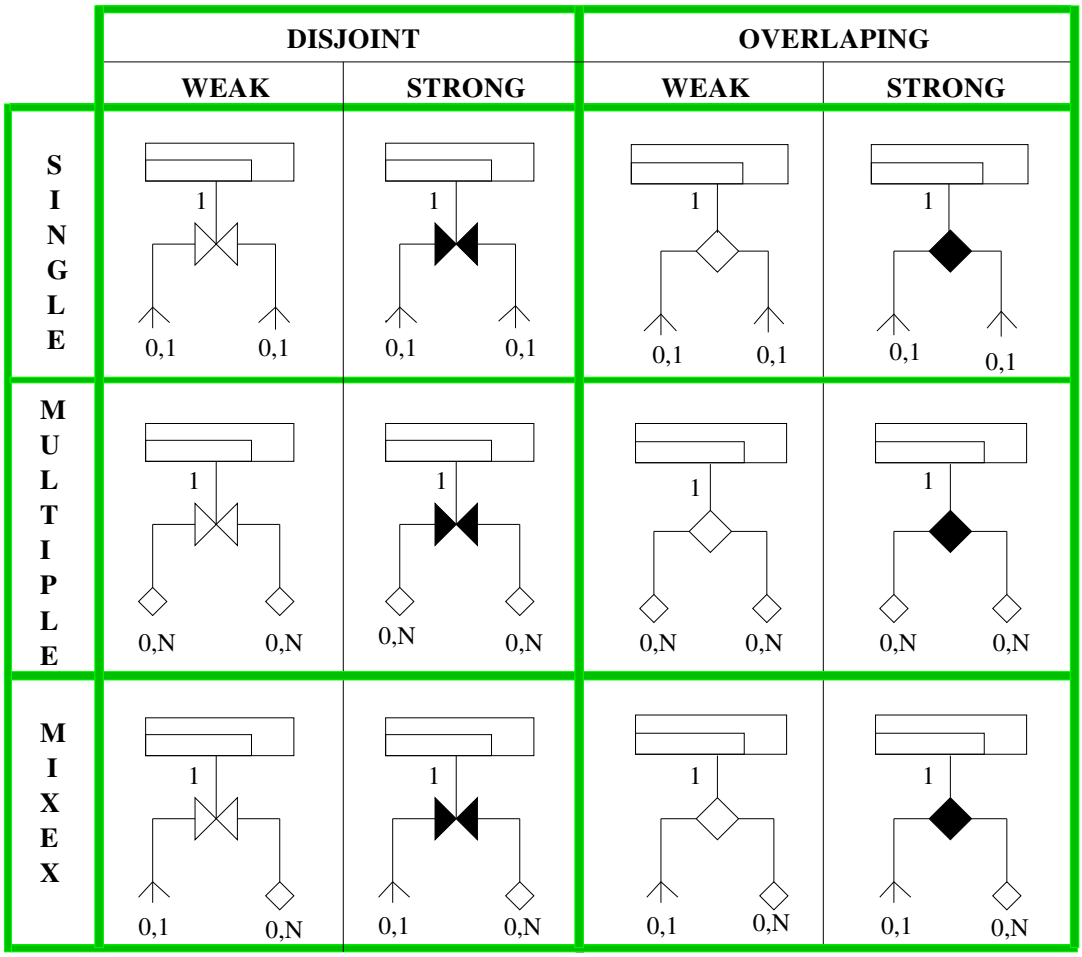

Figure 1: Forked Link Used to Construct Extended ELKA Relationships.

In this paper we are going to concentrate on the latter case.

These extended relationships have four main components: a forked link, the superentity class, the subentity classes, and a *subtype defining entity class*. The superentity stores the common attributes of the individual entities and includes a *discriminatory attribute* whose value is used to determine in which subentity, if any, its specific attributes are to be stored. Each one of the subentity classes includes the particular attributes that describe in detail the specific characteristics of the subentity classes.

The subtype defining entity class has as entity members the subtype defining rules with the criteria for the population of the subentity classes. These rules are similar to the ones used in [4] for their specialization hierarchies, but in our approach every subentity can have various subtype rules suited to represent different cardinality constraints.

3.1 Disjoint Extended Relationships

The six disjoint forked links in Figure 1 are used to define disjoint extended relationships. In these relationships, the forked link class causes the migration of the superentity key values to zero or one of its N ($N \geq 2$) mutually disjoint subentity classes, while for every entity a_j in a subentity class A_i there is *exactly one* entity b in the superentity class. The six disjoint relationships, that may be described by functions from its subentity classes to the superentity class, are described at continuation:

3.1.1 Weak-1 Disjoint Relationship

We use the single disjoint-weak forked link to represent a weak-1 disjoint relationship. Here for every entity a_j in a subentity A_i there exists exactly one entity b in the superentity, and for every b in the superentity there exist *zero or one* subentities A_i, $i = 1, ..., n$, that inherit *once* the value of b.

3.1.2 Strong-1 Disjoint Relationship

We use a single disjoint-strong forked link to represent the strong-1 disjoint relationship. Here for every entity a_j in a subentity A_i there is *exactly one* entity b in the superentity, and for every entity b in the superentity there exists *one* entity a_j in *exactly one* subentity A_i, $i = 1, ..., n$, that inherits *once* the key value of b.

3.1.3 Weak-N Disjoint Relationship

We use a multiple disjoint-weak forked link to represent the weak-N disjoint extended relationship. Here for every entity a_j in a subentity A_i there is *exactly one* entity b in the superentity, and for every b in the superentity there are *zero or one* subentities A_i, $i = 1, ..., n$, that inherit *one or more* times the value of b.

3.1.4 Strong-N Disjoint Relationship

We use a multiple disjoint-strong forked link to represent the strong-N disjoint extended relationship. Here for every entity a_j in a subentity A_i there exists exactly one entity b in the superentity, and for every entity in the superentity there exists *exactly one* subentity A_i, $i = 1, ..., n$, that inherits *one or more* times the value of b.

3.1.5 Weak-Mixed Disjoint Relationship

We use a mixed disjoint-weak forked link to represent the weak-Mixed extended relationship. Here for every b in the superentity there are: (a) No subentities that inherits the value of b, or (b) one subentity A_i, $i = 1, ..., n$, at the back of a 1-to-1 branch that inherits *once* the value of b, or (c) one subentity A_i, $i = 1, ..., n$, at the back of a weak-n-to-1 side that inherits *one or more* times the value of b.

3.1.6 Strong-Mixed Disjoint Relationship

We use a mixed disjoint-strong forked link to represent the strong-mixed extended relationship. Here for every entity in the superentity there either is: (a) one subentity A_i, $i = 1, ..., n$, that inherits *once* the value of b, or (b) one subentity A_i, $i = 1, ..., n$, that inherits *one or more* times the value of b.

The six overlapping forked links of Figure 1 are used to assemble extended overlapping relationship structures with characteristics described by functions from its $n \geq 2$ subentities to the superentity. These six overlapping relationships are the Weak-1, Strong-1, Weak-N, Strong-N, Weak-Mixed and Strong-Mixed overlapping extended relationship.

In these relationships, the forked link class is capable of causing the migration of the superentity key values to zero, one or more of its n ($n \geq 2$) subentity classes, while for every entity a_j in a subentity class A_i there is *exactly one* entity b in the superentity class. These six overlapping relationships are the Weak-1, Strong-1, Weak-N, Strong-N, Weak-Mixed and Strong-Mixed overlapping extended relationship.

3.1.7 Weak-1 Overlapping relationship

We use a single overlapping-weak forked link to represent the weak-1 overlapping relationship. Here for every entity b in the superentity there are *zero, one or more* subentity classes A_i, $i = 1, 2, ..., n$, that inherit *once* the key value of b.

3.1.8 Strong-1 Overlapping relationship

We use a single overlapping-strong forked link to represent the strong-1 overlapping relationship. Here for every entity b in the superentity there are *one or more* subentity classes A_i, $i = 1, 2, ..., n$, that inherit *once* the key value of b.

3.1.9 Weak-N Overlapping relationship

We use a multiple overlapping-weak forked link to represent the weak-1 overlapping relationship. Here for every entity b in the superentity there are *zero, one or more* subentity classes A_i, $i = 1, 2, ..., n$, that

inherit *one or more* times the key value of *b*.

3.1.10 Strong-N Overlapping relationship

We use a multiple overlapping-strong forked link to represent the strong-1 overlapping relationship. Here for every entity *b* in the superentity there are *one or more* subentity classes A_i, $i = 1, 2, \ldots, n$, that inherit *one or more* times the key value of *b*.

3.1.11 Weak-Mixed Overlapping relationship

We use a mixed overlapping-weak forked link to represent the weak-mixed overlapping relationship. Here for every entity *b* in the superentity there are: (a) *zero, one or more* subentity classes A_i, $i = 1,2,...,n$, that inherit *once* the key value of *b* and (b) *zero, one or more* subentity classes A_i, $i = 1,2,...,n$, that inherit *one or more* times the key value of *b*.

3.1.12 Strong-Mixed Overlapping relationship

We use a mixed overlapping-strong forked link to represent the strong-mixed overlapping relationship. Here one or both of the following conditions must be satisfied: for every entity *b* in the superentity there are (a) *one or more* subentity classes A_i, $i = 1,2,...,n$, that inherit *once* the key value of *b*, and/or (b) *one or more* subentity classes A_i, $i = 1,2,...,n$, that inherit *one or more* times the key value of *b*.

4 Conclusions

The extended relationships described give more semantic power to the database modelling design process and have been constructed and described using ELKA-like descriptors. The semantics for both complex objects and generalization/specialization hierarchies found in [5, 6, 7, 8, 9] can be derived from the semantics of the single forked links. The use of the multiplicity property of the forked links in constructing extended relationships have not been found in the literature. The new relationships facilitate the application of the ELKA methodology to the analysis and design of object oriented applications in wich complex structures frequently occur.

References

[1] G. Rodriguez, The ELKA Model Approach to the Design of Database Conceptual Models, PhD Dissertation, University of California, Los Angeles, 1981.

[2] P.P. Chen. The entity-relationship model: toward a unified view of data, ACM Trans. Database Sys. 1(1), 1976.

[3] R. Garza, Extensions and Dynamic Properties for the ELKA Model, in Lissoni, Richardson, Miles, Wood-Harper & Jayaratna (eds.), Inf. Syst. Methodologies Conference, BCS-ISM 94, Edinburgh.

[4] ter Hofster, van der Weide, Expressiveness in conceptual data modelling, Data & Knowledge Engineering, 10, 1993.

[5] P. Beynon-Davies, Entity Models to Object Models: Object Oriented analysis and database design, in Information and Software Technology, Vol 34, No. 4, 1992.

[6] C. Batini, S. Ceri, S.B. Navathe, Conceptual Database Design An Entity-Relationship Approach, The Benjamin Cummings, 1992.

[7] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen, Object-Oriented Modelling and Design, Prentice-Hall International, Inc. 1991.

[8] R. P. Whittington, Database Systems Engineering, Clarendon Press, Oxford, 1988.

[9] J. Iivari, Relationships, Aggregations and Complex Objects, Information Modelling and Knowledge Bases III, S. Ohsuga et al eds, IOS Press, 1992.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Feature Extraction Using Wavelets for the Classification of Human *in Vivo* NMR Spectra

Rosemary Tate
rosemary@cogs.susx.ac.uk

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

Abstract This paper reports the use of the discrete wavelet transform to extract features for classification from *in vivo* Magnetic Resonance ^{13}C spectra. Normally peak areas or heights are used to analyse MR spectra, but these can be difficult to quantify and the use of wavelet coefficients, which implicitly represent the shapes of the peaks,was investigated as an alternative. The spectra were obtained at Hammersmith Hospital for a study into the effects of diet on subcutaneous fats, using volunteers from three dietary groups: vegans, vegetarians and omnivores. These spectra had already been successfully classified according to dietary group, using linear discriminant analysis, with peak heights as the variables. The use of selected wavelet coefficients was equally as successful and eliminated the need for direct quantification of the peaks.

Introduction

Magnetic resonance spectroscopy provides information on the chemical composition of certain substances. The peaks in the spectra represent nuclei in different molecular sites, resonating at slightly different frequencies. The quantities of a particular substance are normally calculated by measuring the area or the height of each peak.

MR spectra obtained *in vitro* generally have sharp peaks which can often be easily quantified. MR spectra obtained *in vivo*, however, are far less easy to analyse, particularly those obtained using the comparatively low magnetic fields which must be used when obtaining data from human subjects. Furthermore, the many technical problems involved in obtaining signals from living tissue may result in a low signal-to-noise ratio and various artefacts, such as those caused by motion, make identification and quantification of peaks in individual spectra extremely difficult. The aim of this work is to investigate ways of analysing *in vivo* MR spectra using techniques which do not necessarily rely on such quantification but instead look for general patterns in the whole data set.

A pattern recognition approach requires the extraction of features which can be used to describe and classify the data. While quantification of peaks may not be directly possible, any features would need to be implicitly related to their relative widths and heights. Wavelets are good for modelling spectra since they are localised in space (as are peaks) and are also localised in frequency which characterises spatial scale. Wavelet coefficients were therefore calculated in an attempt to provide features for classification. In order to carry out this investigation we used a set of high quality *in vivo* spectra which had already been successfully classified using automatically extracted peak heights as the variables. We found that the use of selected wavelet coefficients as features provided equally good classification. The following sections describe the results.

Data

The subject of this investigation was a set of ^{13}C spectra of subcutaneous fat in the thigh, obtained from a group of 75 healthy volunteers. Figure 1 shows a typical spectrum. The area under each peak represents the level of Carbon nuclei resonating at a particular frequency. The frequency at which a certain nucleus (in this case the ^{13}C nucleus) resonates will vary according to its molecular site. Thus each peak gives a measure of a different chemical compound or bond, and thus a measure of the different types of fats.

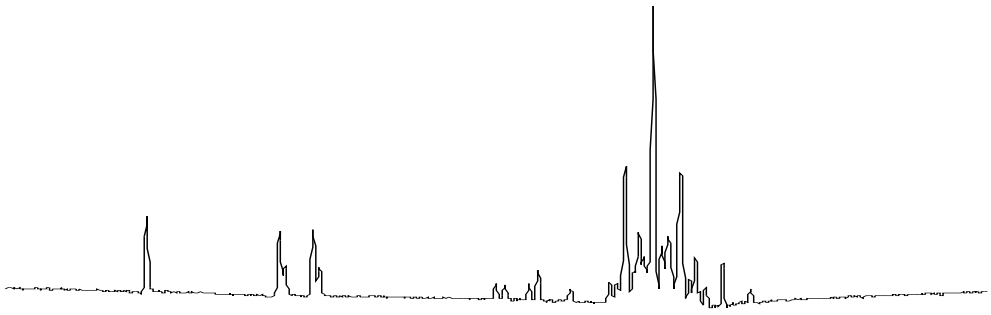

Figure 1: A typical ^{13}C spectrum

The spectra were obtained as part of a study carried out at Hammersmith Hospital to investigate how the type of fat that people eat affects that stored in the body. The volunteers were classified as being either vegan (class 1, n=33), vegetarian (class 2, n=8) or meat-eaters (class 3, n=34). The spectra were of a high quality and most of those from the two main groups could be classified reasonably easily by eye. They were therefore ideal for investigation in this study.

Methodology

The Discrete Wavelet Transform

The Discrete Wavelet Transform (DWT) transforms a data vector of length n (where n is a power of two) into another vector of n ”wavelet coefficients”. The transformation is carried out using a set of basis functions called ”wavelets”, which are dilations and translations of a single function called the ”mother wavelet”. Wavelets are localised in both space and in characteristic scale, and thus can be used to model spiky data such as spectra. If the basis functions approximate the shape of the peaks in the spectra closely enough, only a small number of wavelet coefficients will be needed to represent the original data vector, and these can be used as features for classification. Figure 2 shows one of the set of basis functions used in this study. Note the similarity in shape of the function to the shape of the peaks in the typical spectrum.

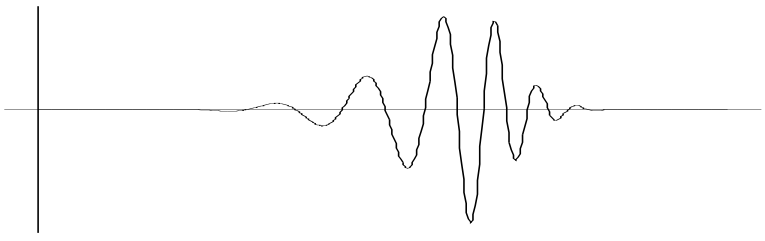

Figure 2: A typical wavelet basis function

Processing the Spectra

Each spectrum was represented by a data vector of length 1024. Pre-processing was carried out to make the spectra compatible. The peaks, which had slightly different positions in each vector due to instrumentation variables which changed between data acquisitions, were aligned by centering each spectrum on the largest peak, simultaneously reducing the vector to the 512 central points which included all the peaks except one (see figure 3). The resulting vector was normalised to length 1 to compensate for arbitrary scaling differences. Each vector was transformed into a set of 512 wavelet coefficients using a Discrete Wavelet Transform (DWT), with the Daubechies 20 coefficients to form the mother wavelet [1]. Figure 2 shows a spectrum and its wavelet coefficients. The boxed area of the spectrum shows the points of the spectrum that were transformed, the boxed wavelet coefficients were those used for classification.

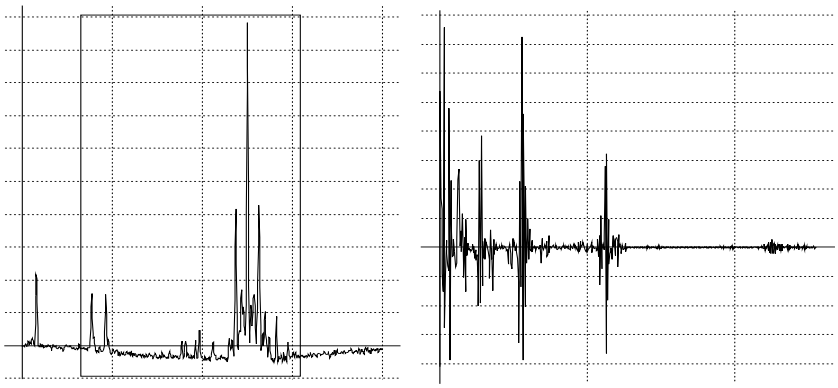

Figure 3: A spectrum and its wavelet transform

The resulting 512 wavelet coefficients, together with the class of each individual were then entered as variables into the SPSS package for statistical analysis.

Calculation of correlation coefficients showed significant correlations between some of the wavelet coefficients and class. The greatest correlations were shown by coefficients 35 (-.68), 37 (-.66), 38 (-.64) and 58 (.6) which suggested that the region including the first four peaks would be most significant in determining class. This was as expected since the first four peaks represent unsaturated fat, which show distinctly higher levels in the vegans. The magnitudes of all other correlations were less than 0.6 and no significant correlations were found for the last 361 wavelet coefficients. Linear Discriminant Analysis had proved successful in distinguishing between vegans and meat-eaters when peak heights had been used as the variables and it was also the method of choice for this study using wavelet coefficients. It was first necessary to select the variables for the discriminant function. Since many of the wavelet coefficients were highly correlated with one another it was decided to carry out Principal Component Analysis of the wavelet coefficients and to see whether a few of the PC's might be used as the variables. Discriminant Analysis was carried out on all three groups and then on the two main groups (the meat-eaters and vegans). A test set was produced from 15 randomly selected cases. The remainder were used to produce the discriminant functions.

Results

Only 60 wavelet coefficients (numbers 5–64) were necessary to produce results comparable with those of the previous study using peak heights as the variables for the discriminant function. Although the first two PC's accounted for only 35% of the variance in the data, the second PC was highly correlated with class (.69). The first PC had the only other significant correlation (-.34). When the first two PC's were entered as the variables for DA, and the vegetarians were excluded from the analysis, 93% of the training set and all except one of the test set were classified correctly. The same success rate was achieved when

another random sample was selected. When the vegetarians were included, the results were not so good: 67% and 71% of the two training sets, and 73% of both test sets, which each included two vegetarians, were classified correctly. This poorer result was probably due to two factors: firstly the number of vegetarians may be too small for them to be included in such an analysis, and secondly because the differences between the diets of vegetarians and the other two groups is not nearly as great. This was shown by the fact that almost all of the misclassifications were omnivores being classified as vegetarians or vice versa. These results were very similar to those obtained when peak variables were used.

Conclusions

This study demonstrates an application of wavelets for feature selection in the classification of data. While these particular spectra could be classified equally successfully by other means, use of the wavelet transform removed the need for selecting and quantifying peaks, thus reducing the amount of pre-processing needed. This could be very useful when automated processing is required. The localised property of the wavelet transform, apart from its advantages in in modelling the peaks, also allows us to identify the most important contributory features to classification. These spectra were unusual for *in vivo* human spectra in that the quantification of the peaks was relatively easy. This is not generally the case. The results from this study indicate that wavelets might provide a useful tool in analysing more problematic and complex spectra. One major problem with localised spectra is known as the rolling baseline, resulting in a fairly arbitrary mean level of the spectrum, which contributes to the difficulties of estimating the heights of the peaks. A feature of the DWT that may prove useful is that the mean level is represented by the first four wavelet coefficients. While these first four coefficients contain large-scale information about the spectra, they may not be necessary for classification, as was indicated in this study.

Acknowledgements

I should like to acknowledge the support given by D. J. Bryant, E. L. Thomas and J. D. Bell of the Robert Steiner NMR Unit at the Hammersmith Hospital and by P. M. Williams of the School of Cognitive & Computing Sciences, University of Sussex. I should also like to acknowledge the financial support from the Science and Engineering Research Council.

References

[1] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, second edition, 1992.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

MML, a Modelling Language with Dynamic Selection of Methods

Vicente Guerrero-Rojo*
vicenter@rsuna.cogs.susx.ac.uk

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

Abstract Second generation knowledge based systems often incorporate multiple problem solving methods. Up to day, there is a need for modelling languages capable of handling, invoking, evaluating and choosing multiple methods at run-time. There are several modelling languages with such capabilities. With them it is possible to develop robust, more flexible and less brittle systems. Unfortunately, those languages are not flexible enough to cope with the behaviour of the systems when more methods are incorporated. In this paper we propose a new modelling language which overcomes these shortcomings.

1 Introduction

Second generation knowledge based systems (systems) often incorporate multiple problem solving methods. The advantages of having multiple methods in a system have become apparent: robustness [Simmons 93], flexibility [Vanwekenhuysen, Rademakers 90], broader kind of reasoning [Delouis 93], less brittleness, reusability [Punch, Chandrasekaran 93].

The decision about which method to use is very much an open problem [David et al. 93]. Nowadays there is a need for modelling languages capable of handling, invoking, evaluating and choosing multiple methods at run-time [Chandrasekaran, Johnson 93].

There are several modelling languages with such capabilities. With them it is possible to develop robust, more flexible and less brittle systems. These capabilities are mainly derived from the addition into the systems of more than one method and specialized activities such as selection, ordering and evaluation of those methods. The success of those languages so far is due to the use of general methods. In other words, no interaction at all between methods exist. However, problems may appear when: the methods involved are sub-methods (part of other methods), in particular when some of those combinations are invalid, effective for specific problems or inefficient; when the user of the systems have preferences for specific combinations rather than for single methods; when no method can be chosen because of a lack of information about the context of the problem; when some of the methods require a particular handling that differs from the others. Thus, it can be said that those modelling languages are not flexible enough to cope with the behaviour of the systems when more methods are incorporated.

As an initiative to overcome those shortcomings in the next section a new modelling language is proposed. The objective of that section is to describe briefly MML (Multiple Method Language), a flexible modelling language capable of overcoming the above mentioned problems. The last section provides some conclusion. Throughout the text the ideas are clarified by examples from the problem known as Sisyphus-92 (from now on the Sisyphus problem) [Linster 91, Linster 92].

*Sponsored by CONACyT, MEXICO.

2 MML - Multiple Method Language

The MML is a task-independent modelling language for the explicit representation of systems with flexible control strategies which allow dynamic selection of methods. MML is being designed as an initiative to overcome some of the shortcomings that current modelling languages present [Guerrero 94], as well as to easy the representation and acquisition of the knowledge, activities and control strategies involved in such systems. It can be said that MML is a generalization of a number of modelling languages such as LISA [Delouis 93] and TIPS [Punch, Chandrasekaran 93]. The underlying ideas of this approach are:

- A modelling language should be a reflective language in which the following basic components can be defined explicitly [Guerrero 94]: an object-level system, a meta-level system, meta-level objects and their features (properties and abstract structures), meta-level activities (*what* can be done at this level), and control strategies (*when* it is done).

- At the same time, the underlying architecture should be open ended in order to allow the addition of new components, components' instances or instances with extended or modified descriptions. This facility represents a generalization from current approaches.

- The objects in the system should be described not just in terms of their features (properties and abstract structures) but also in terms of how those properties are used: *what* and *when* (activities and control strategies).

- A general modelling language should be meta-task specific. It means that it should provide some primitives meta-level activities for specific groups (i.e. method selection, explanation, monitoring). For example it might have method related meta-level activities primitives.

- Brittleness might be reduced not just by providing multiple methods and the capability for their dynamic selection but also, incorporating a number of other method related activities. For example, monitoring the development of the execution of the method, fault diagnosis and repair, analysis of results (e.g., quality, quantity). At the same time it should provide a minimum set of those elements for easy of system development. This idea represents an improvement in current approaches.

- Flexibility might be increased providing facilities for defining not just one general control strategy but rather, providing the facilities for defining specific control strategies for each node in the control structure, in particular for tasks. Control strategies can be shared or assigned by default to several nodes in the control structure. This facility represents a generalization from current approaches.

- The problems, the methods defined for solving those problems, the activities and the control strategies that are applied to those methods are modelled explicitly at the knowledge-level and clearly separated from each other. The language is represented in a task-method control structure in which two types of control knowledge can be identified and clearly separated: knowledge for controlling the decomposition (sequencing) of subtasks or sub-methods, and knowledge for controlling meta-level activities. Both might be represented in procedural or heuristic terms depending on the specific application.

In the proposed approach there are two levels: meta and object. Both levels are controlled by a single interpreter. At the meta-level there are basically two objects: tasks and methods. The underlying architecture resembles the so-called subtask-management [Harmelen 91]. The system follows an object level plan (control structure) and hands over control to the meta-level in specific situations such as task activation.

The following subsections provide an outline of MML.

```
[define                 propose
properties:
   [type                task]
   [goal 'The goal  of this  task is the  allocation (solution)  of ...]
   [input               components resources]
   [output              allocations]
   [control-terms       NameOfMethod]
abstract structures:
   [associated-methods  decomposition random-init sequential]
   [satisfaction-crit   allocations  not = empty ]
   [preferences         [random-init all] [sequential decomposition]]
   [code
            collect-methods(associated-methods) -> SetOfMethods;
            while SetOfMethods /= [] do
                select-a-method(SetOfMethods, appropriateness-crit)
                                    ->   NameOfMethod;
                applicable(NameOfMethod) -> Applicable;
                if Applicable = true then
                    apply-method(NameOfMethod);
                    test-satisfaction(satisfaction-crit) -> TaskSuccess;
                    if TaskSuccess = true then
                        return;
                    endif;
                endif;
            endwhile;
    ]
activities:
   [applicable((NameOfMethod) -> boolean   using same in NameOfMethod]
   [apply-method((NameOfMethod)            using same in NameOfMethod]
   [collect-methods(associated-methods) -> SetOfMethods using getprop]
   [select-a-method( SetOfMethod) -> NameOfMethod
                                    using select-by-appropriateness]
   [test-satisfaction( satisfaction-crit) -> boolean]
control strategies:
   [apply-task( code)       using code-interpreter]
]
```

Figure 1: The propose task in the Sisyphus problem

2.1 Meta-level objects

MML has basically two meta-level objects: tasks and methods. Tasks represent the problems to be solved while methods the different ways in which problems can be solved. Both tasks and methods are related by means of a control structure. In order to describe those objects, MML relies in the distinction of four features: properties, abstract structures, meta-level activities, and control strategies. Every feature consists of an attribute and a value. The value might be implicit or explicit. Implicit values refer to basic inferences, methods, or tasks, that once activated, can generate the expected (explicit) value or the desired behavior.

2.2 Properties

Properties are a collection of basic features of objects. An object might have any number of properties and new ones can be defined. Properties might be further classified in at least two sets: static and dynamic. The former are defined by the knowledge engineer and the later by the interpreter. For example, static properties are: a goal, the type of object, the type of control knowledge. The status of an object is an example of dynamic properties.

2.3 Abstract structures

They are structures that categorize and represent the knowledge about the relations and constraints between the objects (at both levels) in the system. The different categories in this knowledge helps in its acquisition. Several abstract structures have been identified. They might be included as primitives in a modelling language. For example: satisfaction criterion, failure criterion, associated methods, preferences, code.

2.4 Meta-level Activities

These activities are processes that run at a meta-level. They are the components that interpret and manipulate abstract structures. For example, activate a task, select a method, evaluate the results of a method, select a domain model. In other words, they are the semantics attached to abstract structures. Activities in MML might be specified as basic inferences, meta-methods or tasks, some of which might be primitive. For example, the activities might be simple as basic inference that evaluates a structure to true or false (e.g. test-satisfaction see figure 1), or so complex that a method or a task is required (e.g. select-a-method see figure 1). The decision of how to specify activities represents a departure from the current languages since MML has great flexibility in this respect. So, depending on the complexity of the activity, its representation varies.

2.5 Tasks

Tasks characterize the set of problems to be solved. A task has a goal which is a specification of what needs to be achieved. A task by itself does not include, as part of its description, any specification of how it will be accomplished. At most it just describes the strategies that apply to the methods that are known to satisfy its goal. In this framework a task is decomposed into methods and not into subtasks. Tasks may or may not have associated (on *a priori* basis) methods that are known to satisfy its goal. Among the properties that a task might have, the following are the most common: goal, input, output. For example, the *propose* task in the Sisyphus problem can be described as in figure 1. This description can be interpreted as: the task will succeed if there is an non empty allocation (components into resources). The methods are collected and then applied until task satisfaction or no more methods remain. They are selected by a primitive activity called select-by-appropriateness. The activities *applicable* and *apply-method* are defined as external activities through the control term NameOfMethod since they are activities that depend on the method selected. The way in which the task is applied is defined in a control strategy named *apply-task.*

2.6 Methods

Methods characterize the set of mechanisms: algorithms, plans of actions, sets of heuristics, that are available for the satisfaction of tasks. Methods, as tasks, are represented by the four components mentioned above. Methods do not have also, a fixed set of features. For example, the *decomposition* method in the Sisyphus problem can be described as in figure 2. This description can be interpreted as: the *decomposition* method is a non-terminal, non-backtracking method with two sub-methods (i.e. assemble-plan, assign-resources). Its control knowledge is procedural. It has two activities, one determines if the method is applicable and the other how to apply its control knowledge. This method has no associated control strategies.

A method can be decomposed into subtasks or sub-methods. Both the decomposition and the sequencing knowledge needed to control such decomposition are stated explicitly. Among the properties that a method might have, the following are the most common: goal, input, output, type (terminal, non-terminal, basic inference), structure (method decomposition), ck-type (formalism involved in the description of its sequencing knowledge: procedural, heuristic), backtracks (capability of backtracking). MML recognizes three different types of methods:

```
[define             decomposition
properties:
   [type            method]
   [goal            'To allocate components into resources using ...']
   [input           components resources]
   [output          allocations]
   [m-type          non-terminal]
   [ck-type         procedural]
   [backtracks      false]
   [structure       assemble-plan assign-resources]
   [control-terms   NameOfMethod]
abstract structures:
   [applicable-crit
                    components-value       exist and
                    resources-value        exist ]
   [appropriateness-crit
            time =       bound    and
            a-plan       exist    and
            problem-type in [sisyphus researchers]
   [code            ...
                    debug(m, 'Executing decomposition');
                    getdomain( components, value) -> Components;
                    getdomain( resources, value) -> Resources;
                    for Res in Resources do
                        putdomain(store, allocations, Res, []);
                    endfor;
                    ...]
activities:
   [applicable( applicable-crit) -> boolean]
   [apply-method( code)   using code-interpreter]
control strategies:
]
```

Figure 2: Decomposition method in the Sisyphus problem

- Terminal. This method is a method whose internal structure is not known or is not worth worrying about (named black box). For example, statistical routines. These methods can only be applied rather than monitored for example. There is a distinguished set of terminal methods, namely basic inferences. These are methods which cannot be split into subtasks any more although they do not represent black boxes. For example, the method assign-resources or the user of the system who may also participate as a terminal method.

- Meta-method. This method is used to store some of the control strategies and activities. They are methods at the meta-level which have access to data not available at the object-level. Besides this fact, meta-methods are similar to methods. For example, the meta-method single-method (see figure 3) that can be used as a default strategy in the Sisyphus problem. This description says that the method single-method consists of a single activity, how to apply its abstract structure *code*. It has no associated control strategies.

- Non-terminal. This method proposes the sequence of subtasks or sub-methods that carry out a task. The sequencing knowledge in these methods is either deterministic (procedural control) or non-deterministic (heuristic control). This approach does not worry about the control knowledge involved in the terminal methods and the basic inferences.

A number method related control activities have been identified in the dynamic selection of methods (e.g., collection, ordering, selection, evaluation of methods) and might be included as primitive activities

```
[define                    single-method
properties:
   [goal                   'call an object-level method']
   [type                   meta-method]
   [ck-type                procedural]
   [m-type                 meta-method]
   [structure  take-method applicable
               apply-method test-satisfaction]
   [control-terms       NameOfMethod]
abstract structures:
   [code
    vars NameOfMethod, Applicable, TaskSuccess;

     take-method(associated-methods) -> NameOfMethod;
     applicable(NameOfMethod) -> Applicable;
     if Applicable then
         apply-method(NameOfMethod);
         test-satisfaction(satisfaction-crit) -> TaskSuccess;
         if TaskSuccess = true then
            return;
         endif;
     endif;
    ]
activities:
   [apply-m-method( code) using code-interpreter]
control strategies:
].
```

Figure 3: Single-method meta-method in the Sisyphus problem

in a modelling language (e.g. select-by-appropriateness see figure 1).

2.7 Control strategies

They are a collection of (control) statements that prescribe the order in which meta-level activities are applied. They not only incorporate the control knowledge associated with the activities but also the control knowledge specified by the methods to which those activities are related to. For controlling meta-level activities MML uses high level control statements and built-in functions (i.e. conditional, loops). This represents another difference with current environments. The control in some of them is full of symbol-level constructs. In MML the need for using symbol-level constructs is minimized since some constructs and built-in functions are predefined and a library of them is included in the language. Control strategies are represented as any other activity (e.g. apply-task see figure 1).

3 Conclusions

This paper has proposed MML, a new modelling language which incorporates the following features:

- MML is an open ended reflective language.
- MML describes objects in terms of both their properties and how those properties are used.
- The language facilitates dynamic selection of multiple methods along with other method-related activities such as diagnosis and repair.
- Control strategies are represented at each node in the control structure.

- The language distinguishes between control knowledge for decomposition of methods and control knowledge for method-related activities.

MML has been designed to capture some of the more desirable features required for solving problems with multiple methods. Initial experiments with the Sisyphus problem indicates that MML is not only a more flexible system but also it has led to an improvement in the specification of the knowledge about methods.

Further work will involve extending the range of methods defined in MML and evaluating its modelling capabilities on a variety of selected problems.

Acknowledgments

The author expresses his appreciation to Sharon Wood and Steve Easterbrook for their support and contributions. This work has been supported by a scholarship of the Consejo Nacional de Ciencia y Tecnología, México.

References

[Chandrasekaran, Johnson 93] Generic Tasks and Task Structures: History, Critique and New Directions, in: David J.M., Krivine J.P., Simmons R. (editors), Second Generation Expert Systems: a step forward in knowledge engineering, Springer, 1993.

[Delouis 93] Delouis Isabelle, LISA : un langage réflexif pour la modélisation du contrôle dans les systèmes à bases de connaissances: application a la planification des réseaux électriques, DPhil Thesis, June 1993.

[David et al. 93] David J.M., Krivine J.P., Simmons R. (editors), Second Generation Expert Systems: a step forward in knowledge engineering, Springer, 1993.

[Guerrero 94] Guerrero Rojo Vicente, MML, a Modelling Language with Dynamic Selection of Methods, Research Paper CSRP 344, School of Cognitive & Computing Sciences, University of Sussex at Brighton, UK, 1994.

[Harmelen 91] van Harmelen Frank, Meta-Level Inference Systems, Pitman, London, 1991.

[Linster 92] Linster Marc, Sisyphus'92 Models of Problem Solving, Arbeitspapiere Der GMD 630, Gesellschaft Fur Mathematik, Und Datenverarbitung MBH, March 1992.

[Linster 91] Linster Marc, Sisyphus'91 Models of Problem Solving, Arbeitspapiere Der GMD 663, Gesellschaft Fur Mathematik, Und Datenverarbitung MBH, July 1992.

[Karbach, Voβ 92] Karbach W., Voβ A. Reflecting about expert systems in MODEL-K in: Chapter 8, Harmelen van F. (editor), Knowledge-level Reflection: Specifications and Architectures ESPRIT Project P3178 KADS-II Doc. RFL/UvA/III/2, 1992.

[Punch, Chandrasekaran 93] Punch W.F., Chandrasekaran B. An Investigation of the Roles of Problem-Solving Methods in Diagnosis, in: David J.M., Krivine J.P., Simmons R. (editors), Second Generation Expert Systems: a step forward in knowledge engineering, Springer, 1993.

[Simmons 93] Simmons Reid, Generate, Test and Debug: A Paradigm for Combining Associational and Causal Reasoning, in: David J.M., Krivine J.P., Simmons R. (editors), Second Generation Expert Systems: a step forward in knowledge engineering, Springer, 1993.

[Vanwekenhuysen, Rademakers 90] Vanwekenhuysen J., Rademakers P. Mapping a Knowledge Level Analysis onto a Computational Framework, in: Aiello Loigia C. (Editor), Proceedings of the 9th European Conference on AI, 1990.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Anaphora Processing: A Cross-Linguistic Discussion

Marco Rocha*
marco@cogs.susx.ac.uk

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

Abstract This paper raises questions about anaphora processing from both a psycholinguistic and a computational point of view. The cross-linguistic aspect involved derives from the observation of anaphoric relations in English and Brazilian Portuguese, focusing on spoken language. The occurrences are extracted from two corpora of dialogues: the London-Lund Corpus of Spoken English for the English examples; and the NURC (Norma Urbana Culta) for the Brazilian Portuguese examples. The final purpose of the research is to create a corpus annotation that successfully incorporates all the relevant elements of anaphora processing. The possibility of automatically annotating any similar corpus is an important concern of the research, although it will not be discussed in this paper.

1 Anaphora Resolution

As it is well established in the literature about anaphora, the term in fact encompasses a variety of cohesion phenomena. There is quite a bit of controversy on how wide the range of phenomena included should be. Nevertheless, the main focus of investigation is on anaphora resolution, that is, how the processor[1] finds the antecedent for an anaphoric term. One significant share of antecedents for anaphors can be found using a 'naive' algorithm such as the one spelled out in Hobbs (1986). The algorithm is said to be 'naive' because it does not involve any semantic processing. Antecedents are found through a relatively simple mechanism using recency and syntactic information like the notion of c-command (Reinhart 1983) within a parse tree. Thus, the antecedent for the pronoun *it* in the example below could be found without resorting to semantic processing:

```
(1)
A:  how's the thesis going
B:  uh I'm typing it up now
B:  typing up the final copy
A:  hm
```

Hobbs concentrates on pronoun resolution, and therefore nonpronominal anaphoric noun phrases - like *final copy* in the example above - would have to be resolved in some other way, probably involving world knowledge, in a complete processor. However, even if only pronouns are taken into consideration, an algorithm dealing exclusively with syntactic information would not be able to find the antecedent in all

*This research is funded by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) under grant no.200608-924

[1]The word *Processor* is used here with the meaning widely assigned to it in psycholinguistic work, that is, anything able to process, whether human or machine.

cases. Hobbs acknowledges this, but the percentage of correct antecedents found (83%) is still encouraging, although the figure may be quite a bit lower for spoken language. Example (2) below reproduces the continuation of the dialogue which began in (1).

(2)
A: uh when are you submitting it ,
B: erh - well it it would have been
A: next term ,
B: this - autumn -

The pronoun *it* in the second sentence can only be interpreted if the processor makes use of lexical information contained in lemmas to infer the antecedent noun phrase *submission* out of the verb in the preceding sentence. The following utterances bring a number of new demands on the anaphora interpreter.[2]

(3)
B: but er - I had to go to work -
B: this winter -
B: and that really <3 sylls> ,
A: but if you're typing it up now er <why can't 1 syll> yes ,
B: it's going so slowly though you know
B: it's this it's these awful these awful symbols .
A: mm ,
B: you know it's a combination of of the phonetic alphabet
A: mm
B: plus the reformed spelling

The demonstrative in the third utterance refers to the fact that B had to go to work. This antecedent cannot be determined using only syntactical information. The following utterance by A contains a pronoun *it* whose antecedent is again B's thesis, which cannot be retrieved without discourse knowledge involving topicality. The next *it* in the fifth utterance may refer to *typing*, and thus involve lemma processing, or to *thesis*, creating a chain with the distant anaphora in the previous utterance. However, it is not essential for overall understanding whether the chosen antecedent is one or the other (see Sampson 1988 for a discussion of antecedent indeterminacy).

On the sixth utterance the antecedent for *it* seems to be some kind of discourse element like 'the problem' or 'the point'. Discourse understanding might be hampered if the pronoun were considered simply as nonreferential, although the antecedent is not a specific object. The next occurrence of *it* seems to refer to the set of symbols mentioned before. Some kind of adjustment - at least to deal with the conflicting agreement - must be carried out for the correct antecedent to be recognised. In a relatively short fragment, as shown, many problems might arise for a purely syntactic algorithm.

2 Evidence from Portuguese

Anaphoric relations in Portuguese may involve the use of very different knowledge. To begin with, there is no equivalent to the pronoun *it* nor is it required by the grammar that all independent sentences have a

[2]Speech between angle brackets, often with the number of syllables therein contained, contain a word or words that could not be precisely identified for transcription.

phonetically expressed subject. No simple introduction of a prodrop notion will solve the problem, as the insertion of dummies which are nonreferential or have indeterminate antecedents seems strongly counter-intuitive and not at all helpful from a processing point of view. One example[3]:

```
(4)
justamente é isso...é liberdade...você pode ser uma artista
exactly is that(it) is freedom you can be an artist
famosa...  passa pela rua tranqüilamente...
famous pass3rsg by the street peacefully
```

The first utterance presents a few problems with anaphors that cannot be handled easily by adding a trace index to mark pronoun dropping. In fact, such index would create new problems which may lead to unnecessary processing complications. For instance, the phrase which would translate *é isso* in this context into English would be *that's it*. Assigning separate antecedents for *that* and it in such phrases seems not only difficult but pointless. People most probably process the phrase as a collocation which stands on its own, referring to the point in process of being made throughout the discourse fragment.

In the second phrase *é liberdade*, a possible 'dropped' pronoun to be reintroduced would be nonreferential. It is hard to understand why a pronoun should be introduced and then dismissed as nonreferential. It is anyway very unlikely that native speakers of Portuguese understand anaphoric references using this sort of strategy. It does not seem to be a good option for anaphora resolution in a NLP system either. Even when the antecedent is explicit and a specific object represented by a noun phrase in the discourse, it still does not seem necessary or advantageous to introduce elements that are not phonetically realised.

```
(5)
você disse que ( ) que as
you said3sg that that DefArt
pessoas são todas iguais né...uma que não são...
persons are all equal, isn't it?  one that not are
```

At the end of the utterance, the inflected form of the verb *ser* (to be) seems to be quite enough for the processing to locate directly the antecedent *pessoas* without resorting to any equivalent for *they*. Although the pronoun is a necessary feature of English syntax and thus must be interpreted by a processor, the same does not hold for Portuguese. The observation of data in these two corpora seems to point toward distinct referencing processes for each of these two languages. The higher amount of information provided by morphological features in Portuguese makes it possible for a verb form or adjective to refer directly to an antecedent. There seems to be no need to postulate hidden elements, such as trace indexes to mark dropped pronouns. A processor would spot the verbs without a subject or noun phrases without a head noun and search for antecedents on the basis of morphological data. The crucial information needed would then be how to use syntactic, lexical - including collocations - and discourse knowledge dealing with topicality to find the correct antecedent.

3 The Approach Chosen

The main tool for the investigation of anaphoric relations in both corpora is an annotation created specifically for the purpose of this research. The analyst then searches the dialogues to spot anaphors and subsequently their antecedents in order to enter the annotation manually. As topicality is an essential aspect in the approach, the dialogues are previously segmented according to topic continuity. There are three distinct levels in the segmentation: **discourse fragments**, with the same overall discourse topic; **discourse**

[3]Glosses for the Portuguese examples have been provided. They are intended on a word-by-word basis and are not full translations. For instance, the Portuguese word order is not changed into English word order. The conventions used are *3sg*, meaning third person singular, and *DefArt*, meaning definite article.

segments, with the same local topic; and **subsegments**, which contain subtopics related to the segment topics.

The annotation is made up of eight **slots**. The three first slots refer to the segment or subsegment. The first one specifies whether the unit is a segment or subsegment, together with a number that fits the unit into a sequence within a fragment - in the case of segments - or within a segment - in the case of subsegments. The second slot specifies a discourse function from a set which is kept as small as possible. It is not possible to discuss every option for each slot in this paper for reasons of space. The third slot specifies a topic for the segment or subsegment.

The next two slots refer to each anaphor token. The first one specifies the anaphoric word or phrase, a nontrivial task in some cases of Portuguese.[4] The second slot codifies the anaphor grammatically - like for instance, demonstrative pronoun (DPRO). It has been necessary to create subdivisions specifically for the purposes of the research, as *one-anaphora with modifier (one_anaph_modif)*.

The other three slots concern the antecedent. The first one specifies whether it is implicit or explicit, where some special or difficult cases appear. The second relates the antecedent to topicality, specifying if it is for instance the segment topic, the discourse topic, the subsegment topic or a thematic element related to it, to mention the main options. The last slot attempts to define the kind of knowledge predominantly involved in the processing. This is of course a very troublesome decision, but important conclusions may be drawn as a result of this analytical effort.

Once the annotation of a sizable amount of text is completed, an attempt will be made to establish statistically significant relationships between the different elements in each slot of the annotation. The final aim is to produce an antecedent-likelihood theory relating each kind of anaphor to a possible kind of antecedent, as well as to a form of knowledge prevailing in the processing. Possible applications will then be discussed, the most immediately visible ones being automatic annotation of corpora to show anaphoric relations and support for machine translation from English into Portuguese and vice-versa. The approach might possibly be extended to other languages.

References

Hobbs, J.R. (1986) Resolving pronoun references. In B.J. Grosz, K. Sparck-Jones, B.L. Webber & C. Sidner (eds.) *Readings in Natural Language Processing*. Los Altos: Morgan Kaufmann

Reinhart, T. (1983) *Anaphora and Semantic Interpretation*. London: Croom Helm

Sampson, G. Machine Translation: A Non-Conformist View. In King, M., ed. (1987) *Machine translation today: the state of the art*. Edinburgh: Edinburgh University Press.

[4] Some anaphors are verb *ser* copulas that occur without a subject, making it hard to determine precisely which words concur for the finding of an antecedent, because the second element in the copula becomes essential and may involve more than one constituent.

In: de Bourcier, Lemmen & Thompson eds., 1994 *The Seventh White House Papers: Graduate Research in the Cognitive & Computing Sciences at Sussex.* University of Sussex, School of Cognitive & Computing Sciences, Brighton, UK. Research Paper CSRP 350.

Doing a PhD with Hindsight

Julian M. L. Budd and Eevi E. Beck
julianb@cogs and Eevi.Beck@nr.no

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

Every PhD is different; there are no general rules to help you. Our experiences differ, as you see below in the slightly different emphasis we place on what we'd like to 'pass on' to others. We hope, however, that some of what we say below (which is based on our experiences of going through this process) will be useful to you. In hindsight, these are issues which we wished we'd paid more attention to ourselves.

Don't Let It Take Over Your Life

The most important thing to remember is that the PhD is not your life. It can easily take over if you let it. Make sure that there is time to relax and do other things, otherwise you can get too wrapped up in your work so that every bad day is a disaster and criticism is taken as a death-blow to your research. You should, perhaps, treat it like a normal job. Work for a set number of hours, and then try to forget about it.

Julian [1]:

It's your thesis

Another thing to remember is that it's your thesis not your supervisor's. Of course you should be ready to listen to advice, but *you* should decide what to do. Equally, don't become too dependent upon your supervisor for help. Get regular feedback on your work and discuss your problems with other people whose opinion you respect. Don't assume your supervisor is an all-knowing, infallible *God*: supervisors are human too (it's a fact).

Get advice regularly

Try to get the advice you want from your supervisor. For example, make them be explicit about what they are suggesting you should do. Don't just accept vague generalities, make sure you get practical advice about how to implement their suggestion. This way it should reduce both misunderstandings and the need to go back and get further information.

Don't let yourself get stuck

It's very easy to get yourself stuck without seeking advice. You think you should be able to solve the problem on your own and you don't want to bother your supervisor or any one else. You don't want anyone to think that you're stupid. But it's likely that you're not stupid, it's just that the problem is hard. It is very important to keep the "momentum" in your research or in writing up your thesis. When a problem arises it's all to easy to spend too long trying to solve it on your own and getting depressed about it. After a while, you may start to avoid the work/writing problem and as time passes you become frustrated with your inability to solve the problem. And this just gets worse, until finally you don't seem to be making any progress at all, and it seems impossible to get started again. That's why "momentum" is important.

[1] 10 months since submission; 3 months since viva (successful).

Make friends

Lastly, don't become isolated from other research students. It's possible they have already encountered your problem and know how to solve it. Moreover they can give you support when things are not going well and you need cheering up.

Eevi [2]:

Go to conferences

If you can, go to conferences. Discuss your work and your ideas with people there. If you don't know anyone before you go, it'll be a good opportunity to meet people in your field. Chances are that you'll find that some people are really interested in what you're trying to do, which can be a great boost to your confidence! Also, seeing that there's nothing unusual in lots of people not being interested can be good: it's not you or your work there's anything wrong with, it's just the reaction that everyone gets. Some are going to be interested, many not.

Be critical of criticism

If someone – your supervisor, another student, a prestiguous name, or anyone else – criticises your work, try not to take it personally. Evaluate it calmly, if you can: does it make sense to you? Has she (or he) misunderstood what you're trying to do? Are there aspects you need to clarify, like what limitations you have set on your work? If you think they're misguided, might there still be something to their criticism; an angle you hadn't thought about, or a pitfall you're in danger of falling into?

If it makes sense to you, follow advice. If it doesn't, try to work out why it doesn't, and defend it (in later conversations, in your thesis, or wherever). In an existence where feedback on your work is a scarce resource, try to learn something from it either way. At the same time, if you feel radical, *be* radical. It's *your* thesis. In fact, many established researchers expect PhD students to be the ones to rock the boat of established truths in the discipline. If you are convinced you have grounds to do that (or will have grounds), then go for it. Follow your convictions!

Enjoy life!

To stay healthy, physically and emotionally, you have to make sure you don't just think thesis (yours and others') all the time. Do your favourite sport or start another one; spend time with friends; set up your life so you're regularly and frequently reminded that there are other things which are more important than your thesis. If you have children, you're probably getting reminded of this already, but many a single preson has become a social hermit, and many a relationship has felt the strain after a while of one or both doing a PhD. Try to make your weekly routine include things you really enjoy which have nothing to do with academic work.

Keep your distance

The interest of people who're not doing a PhD and never have done one is not going to last if you're constantly talking thesis. That's *why* I think it's really important to spend time with others – you're forced to talk and therefore think about other things.

Keeping a distance from your thesis makes it less likely that you'll turn into a social hermit. What it also can do is help you work better when you are working: occasionally distancing yourself from your work can help you see its weaknesses and strengths, where you have to put in more effort and where you don't.

I got into seriously unhealthy ways towards the end of writing up, thinking about little else than the thesis, not eating properly and constantly tired but often not able to sleep even so. I started noticing how

[2] 2 weeks since submission; not yet examined.

the work I did at times of prolonged immersion was... well, I might have interesting ideas, but it was often *not* what I needed to get on with to meet the deadlines I'd set. Coming back to it after a day out walking (or whatever), I'd often only then realise how I was taking myself off on a tangent which I could ill afford.

Don't kill yourself

This one is no joke. People *have* committed suicide where lack of progress in a PhD has seemed to be a big part of the depression. Don't let the next one be you. If you feel miserable, stop. If a day is not enough, stop for a week, a month, a year, or forever. Within the university there are structures for taking a formal break in your PhD. Use them. I know several people who have taken a break to reassess whether they really want to be doing what they are doing, and have decided that although they felt competent to finish the PhD, that wasn't what they really *wanted* after all. They stopped, and were probably happier for it. Others, like myself, have taken breaks completely away from the thesis to deal with other things in life, and have later chosen to go back to it and continue. The important thing is that you know that you don't *have* to finish just because you started, and that it doesn't affect who you are as a person if you choose not to. In many ways that can be a sign of greater strength than carrying on doing what everyone expects of you!