Schemes for Learning and Behaviour: A New Expectancy Model

Ph.D. Thesis

Christopher Mark Witkowski

February 1997

Department of Computer Science
Queen Mary Westfield College

University of London

Abstract

This thesis presents a novel form of learning by reinforcement. Existing reinforcement learning algorithms rely on the provision of external reward signals to drive the learning algorithm. This new algorithm relies on reinforcing signals generated internally within the algorithm. The algorithm, SRS/E, described here generates expectancies (μ-hypotheses), each of which gives rise to a specific prediction when the conditions relevant to the expectancy are encountered (the μ-experiment). The algorithm subsequently tests these predictions against actual events and so generates reinforcement signals to corroborate or reject individual expectancies. This procedure allows for self-contained, completely unsupervised learning to an extent not possible with previous reinforcement procedures. The SRS/E algorithm is derived from a number of postulates that constitute a new Dynamic Expectancy Model developed in this thesis.

In contrast to the static policy map generated by existing *Q*-learning based reinforcement algorithms, which limit learning to one goal, the SRS/E algorithm generates a Dynamic Policy Map (DPM) from learned expectancies whenever a new goal is selected by the system. This new approach retains the advantages of reactivity to the environment inherent in existing reinforcement algorithms, while substantially increasing the system's flexibility in responding to varying circumstances and requirements. Also in contrast to previous reinforcement systems, goals may be selected arbitrarily and are not limited to those which were associated with reward during the learning steps. This new method allows multiple goals to be pursued either simultaneously or sequentially.

The single SRS/E implementation has been compared directly to the published results from of a family of reinforcement based algorithms, Dyna-PI, Dyna-Q and Dyna-Q+ (Sutton, 1990), themselves extensions to the groundbreaking *Q*-learning algorithm (Watkins, 1989). Under equivalent "ideal learning conditions" the SRS/E algorithm was found to outperform the equivalent Dyna reinforcement program to learn a simple maze task by a factor of some 40:1. The SRS/E learning algorithm was also found to be robust when tested under controlled "noise" conditions. SRS/E was also compared directly to Sutton's Dyna-Q+ algorithm on a range of alternative path and route blocking tasks and was found to offer a similar performance, but SRS/E employs a "biologically plausible" extinction mechanism, mirroring findings from animal behaviour research.

Finally SRS/E was tested with experimental designs for "latent learning" and "place learning", drawn directly from animal learning research. Both are regarded as presenting severe challenges to conventional reinforcement learning theories. SRS/E performs well on both tasks, and in a manner consistent with findings from animal experiments.

Table of Contents

1. The Behaviour of Animals and Animats 9
   1.1. Three Components of Natural Intelligence 10
   1.2. Reactive Models of Intelligence 11
   1.3. Action Selection Mechanisms 12
   1.4. Arriving at a Definition of Learning 18
      1.4.1. What is Not Learning 19
   1.5. A Caveat 20
   1.6. Thesis Outline 21
2. Theories of Learning 23
   2.1. Classical Conditioning and Associationism 24
   2.2. Reinforcement Learning 26
   2.3. Computer Models of Reinforcement Learning 29
      2.3.1. Markov Environments 31
   2.4. *Q*-learning 32
      2.4.1. *Q*-learning - Description of Process 33
      2.4.2. Some Limitations to *Q*-learning Strategies 34
   2.5. Classifier Systems 36
   2.6. Artificial Neural Networks 39
   2.7. Operant Conditioning 42
   2.8. Cognitive Models of Learning, Tolman and Expectancy Theory 44
   2.9. MacCorquodale and Meehl’s Expectancy Postulates 46
   2.10. Computational Models of Low-level Cognitive Theories 49
   2.11. Becker’s JCM Model 49
   2.12. Mott’s ALP Model 52
   2.13. Drescher’s Model 53
   2.14. Other Related Work 56
3. A New Dynamic Expectancy Model 57
   3.1. The Animat as Discovery Engine - The Thesis 58
   3.2. The Expectancy Unit as Hypothesis 59
      3.2.1. The Hypothesis Postulates 60
      3.2.2. Initial Conditions for the μ-Hypothesis Set 63
      3.2.3. Concluding Conditions for the μ-Hypothesis Set 63
      3.2.4. Hypothesis Based Models of Learning 65
      3.2.5. The Role of the Hypothesis in the Discovery Process 67
   3.3. Tokens, Signs and Symbols 69
      3.3.1. The Sign and Token Postulates 70
      3.3.2. Initial Conditions for the Token and Sign Sets 71
      3.3.3. Supporting Evidence for Signs and Tokens 71
   3.4. Actions and Reification 73
      3.4.1. The Action Postulates 73
      3.4.2. Initial Conditions for Actions 74
      3.4.3. Supporting Evidence for an Action Vocabulary 75
   3.5. Goal Definitions 76
      3.5.1. The Goal Postulates 76
      3.5.2. Goals, Starting Conditions and Discussion 77
   3.6. On Policies and Policy Maps 78

3.6.1. Policy Map Postulates 78
3.6.2. Evidence for Chaining 80
3.6.3. Evidence for Goal Suspension and Extinction 81
3.6.4. Comparison to *Q*-learning 83
3.7. Innate Behaviour Patterns 83
3.7.1. Balancing Innate and Learned Behaviour 85
3.8. Advances Introduced by the Dynamic Expectancy Model 86
4. The SRS/E Algorithm 89
4.1. Encoding the Ethogram: SRS/E List Structures 90
4.1.1. List Notation 91
4.1.2. Summary of Lists 93
4.2. Tokens and the Input Token List 94
4.2.1. Input Token List Values 95
4.3. Signs and the Sign List 96
4.3.1. Representing Signs 97
4.3.2. Other Sign List Values 99
4.4. Actions and the Response List 100
4.4.1. Response List Values 101
4.5. Innate Activity and the Behaviour List 102
4.5.1. Behaviour List Structure and Selection 102
4.5.2. Behaviour List Values 104
4.6. Goals and the Goal List 104
4.7. The Hypothesis List 105
4.7.1. Other Hypothesis List Values 107
4.8. Corroborating μ-Hypotheses, Predictions and the Prediction List 109
4.8.1. Prediction List Element Values 111
4.9. The Dynamic Policy Map (DPM) 112
4.9.1. Selecting actions from the DPM 115
4.9.2. Recomputing the DPM 116
4.9.3. The DPM, A Worked Example 117
4.9.4. Pursuing Alternative Goal Paths 121
4.9.5. Pursuing a Goal to Extinction 124
4.10. Creating New μ-Hypotheses 126
4.10.1. Maintaining the Hypothesis List 128
4.11. The SRS/E Execution Cycle 130
4.11.1. Summary of Execution Cycle Steps 131
4.12. The SRS/E Algorithm in Detail 134
4.12.1. Step 1: Processing Input Tokens and Signs 135
4.12.2. Step 2: Evaluating μ-Experiments on the Basis of Prior Prediction 136
4.12.3. Step 3: Selecting Innate Behaviours and Setting Goals 136
4.12.4. Step 4: Building the Dynamic Policy Map 138
4.12.5. Step 5: Selecting a Valenced Action 141
4.12.6. Step 6: Performing an Action 142
4.12.7. Step 7: Conducting μ-Experiments 142
4.12.8. Step 8: Hypothesis Creation and Management 143
4.13. Implementation 146
4.14. SRS/E - A Computer Based Expectancy Model 146
5. Experimental Design and Approach 148

5.1. Experimental Design ... 149
5.2. The User Interface ... 151
5.2.1. Controlling Execution Cycles ... 152
5.2.2. Displaying and Recording List Information ... 153
5.2.3. Managing Goals ... 153
5.2.4. Managing the Animat and Environment ... 153
5.2.5. Accessing Utilities ... 154
5.3. The System Execution Trace Log ... 155
5.3.1. Processing Log Results ... 155
5.4. Important Schedule Variables ... 156
5.4.1. Action Repetition Rate (Arep) ... 156
5.4.2. Action Dispersion Probability (Adisp) ... 157
5.4.3. Learning Probability Rate (Lprob) ... 158
5.5. Fixed Schedule Experiments ... 158
6. Investigations and Experimental Results ... 160
6.1. The Individual Experiments ... 162
6.2. Baseline Investigations ... 163
6.2.1. Description of Procedure ... 164
6.2.2. Results and Analysis of Baseline Experiment ... 165
6.2.3. Discussion ... 169
6.3. The Effects of Noise ... 171
6.3.1. Description of Procedure ... 171
6.3.2. Results and Analysis of Experiment ... 171
6.3.2.1. Tuning Parameters for Static Environments ... 172
6.3.2.2. The Effects of Noise: Learning or Behaviour? ... 175
6.3.3. Discussion ... 177
6.4. Alternative and Multiple Goals ... 178
6.4.1. Description of Procedure ... 178
6.4.2. Results and Analysis of Experiment ... 179
6.4.3. Discussion ... 183
6.5. Multiple-Path, Blocking, Shortcut and Extinction Investigations ... 184
6.5.1. Investigation One (Multiple-Path), Procedure ... 185
6.5.2. Investigation One, Results and Analysis ... 186
6.5.3. Investigation One, Discussion ... 188
6.5.4. Investigation Two (Goal Extinction), Procedure ... 188
6.5.5. Investigation Two, Analysis of Results ... 189
6.5.6. Investigation Two, Discussion ... 192
6.5.7. Investigation Three (Path Blocking), Procedure ... 193
6.5.8. Investigation Three, Results and Analysis ... 194
6.5.9. Investigation Three, Discussion ... 196
6.6. Latent Learning ... 197
6.6.1. Description of Procedure ... 198
6.6.2. Results and Analysis of Experiment ... 199
6.6.3. Discussion ... 200
6.7. Place Learning ... 201
6.7.1. Description of Procedure ... 202
6.7.2. Results and Analysis of Experiment ... 203
6.7.3. Discussion ... 204
6.8. Chapter Summary ... 205

7. Extensions to SRS/E and Further Work 208
7.1. An Association List 208
7.2. Seeking Multiple Goals Simultaneously 209
7.3. An Explicit Template List 211
7.4. Directing Learning Effort 212
7.5. Aversion 214
8. Discussion and Conclusions 216
8.1. Reactive or Cognitive? 216
8.2. Expectancy Model as “Missing Link” in Learning Theory 216
8.2.1. Types of Reinforcer 217
8.3. Relationship to Policy Maps and Universal Plans 218
8.4. One-Shot Learning Phenomena 221
8.5. Expectancy Theory and XBL - a Proposal 222
9. Appendix One 224
10. References 229
11. SUBJECT and AUTHOR INDEX 251

List of Figures and Tables

1-1: Tinbergen's Principle of Hierarchical Organisation 14
1-2: Brooks' Subsumption Architecture 15
1-3: Maes' Action Selection Architecture 17
2-1: A Classifier System 37
2-2: A Simple Neurone Model 40
2-3: A Multilayer Neural Network Model 41
2-4: A JCM Schema 50
2-5: A Schema in Drescher's Cognitive Model 53
2-6: A Composite Action 54
2-7: The Marginal Attribution Process 55
3-1: Extinction Curves Under Various Schedules 82
Table 4-1: SRS/E Internal Data Structures 92
4-1: Log Printout of a Valenced Path 115
4-2: Model DPM Generated from Sample Hypothesis List 118
4-3: Various Outcomes for Model DPM 119
4-4: Model Graph Recomputed for Goal 'S8' 121
4-5: A Sample Dynamic Policy Map 122
Table 4-2: Paths Through Figure 4-5 Graph 123
4-6: Summary of Steps in the SRS/E Execution Cycle 132
4-7: The SRS/E Algorithm 134
4-8: Step One, Token and Sign Processing 135
4-9: Step Two, Evaluation of μ-Experiments 136
4-10: Step Three: Select Innate Actions and Set Goals 137
4-11: Step Four, Construct Dynamic Policy Map 140
4-12: Step Five, Select Valenced Action 141
4-13: Step Six, Perform Action 142
4-14: Step Seven: Conduct μ-Experiments 142
4-15: Step Eight, Hypothesis Creation 144
4-16: Step Eight, Hypothesis Management - Specialisation 145
4-17: Step Eight, Hypothesis Management - Forgetting 146
5-1: Sutton's DynaWorld/Standard Environment 149
5-2: The SRS/E Experimenter Command Options 152
5-3: Effect of Arep on Random-Walk Path Length 157
6-1: Results from Sutton's Dyna-PI Experiments 164
6-2: Baseline Learning Curves (Lprob = 1.0, 0.25, 0.1 and 0.025) 166
6-3: Contribution of Individual Animats to Learning Curve 168
6-4: Baseline Learning with Noise (Adisp = 0.5, Lprob = 1.0, 0.25, 0.1 and 0.025) 172
6-5: Baseline with Noise (Adisp = 0.5, γ^1 = 1.0) 174
6-6: a) Path with Adisp = 0.5 (trial 101), b) Adisp = 1.0 (trial 102) 175
6-7: Policy Map at Conclusion of Trial 101 176
6-8: Planned Valenced Path (trial 101) 177
6-9: Simultaneous Goal Locations 179
Table 6-1: Results for Investigation One of Dual Goal Experiment 180

6-10: Animat Random and Valenced Paths (investigation 1, rseed = 80) 181
Table 6-2: Results for Investigation Two of Dual Goal Experiment 182
Table 6-3: Results for Investigation Three, Simultaneous Goals 183
6-11: Sample Simultaneous Goal Paths 183
6-12: Changing World Environments 185
6-13: Multiple Path Investigation, Individual Performance 186
6-14: Estimated Cost Profile (Path and H14) 187
6-15: Goal Extinction 190
6-16: Goal Extinction, Comparison of Cost Estimate to VBP 191
6-17: Goal Extinction (two path), Cost Estimate and VBP 192
6-18: Average Performance of Dyna Systems on a Blocking Task 194
6-19: Investigation Three, Individual 'Cumulative Reward' Curves 195
6-20: Investigation Three, Average 'Cumulative Reward' Curves 196
6-21: Tolman and Honzik's Latent Learning Results 198
6-22: The SRS/E Latent Learning Environment 199
6-23: Results of the SRS/E Latent Learning Experiment 200
6-24: Tolman and Honzik's 'Insight' Maze 202
6-25: Results from 'Insight' Experiment 204
7-1: Sign-Sign Associations (Secondary Cathexis) 209
7-2: Enhanced Goal Acquisition 210
7-3: The Effect of Valence Level Pre-Bias 213

Chapter 1

1. The Behaviour of Animals and Animats

Man has long sought to understand what constitutes life, and to understand the nature of living things. The new discipline of Artificial Life (Langton, 1989; Levy, 1992; Brooks and Maes, 1994) acts as a focus for research into a diverse set of topics relating to the modelling and understanding of life and the properties of living things. Artificial Life concerns itself with many aspects of those organisms we recognise as living entities. These aspects include evolution, morphology, swarming behaviours, behavioural models and learning, even the nature of life itself. The idea that "living" entities might yet be constructed artificially remains highly speculative and contentious, only in part due to the difficulties in agreeing a satisfactory definition of what does and what does not constitute the necessary properties of being alive. There is more general agreement that simulation can greatly add to our overall understanding of the nature of the structure and behaviour of living things. This work concerns itself with the behavioural properties of the individual. It will therefore touch upon the broader issues addressed by Artificial Life only in passing.

One question has engaged the minds of psychologists and those interested in a greater understanding of animal behaviour for decades. Is the behaviour of animals inherently driven by the current state of the world as perceived through the senses, or is it directed by goals, internally generated needs or requirements of the organism? Huge amounts of evidence supporting these two disparate viewpoints has been accumulated. It is an argument that is far from being resolved and one that has spilled over into the newer domains of Computer Science and Artificial Intelligence, where another generation of scientists is pondering the question and proposing new models of behaviour in an attempt to resolve the issue. The question was the subject of a meeting that invited this new generation of researchers to declare and defend their position - "models or behaviours" (Aylett, 1994). Paralleling this question is that of how learning is to be achieved in either of these possible situations. These problems have recently found renewed expression in an area of study broadly categorised as the "simulation of adaptive behaviour" (Meyer and Wilson, 1991; Meyer, Roitblat and Wilson, 1993; Cliff, Husbands, Meyer and Wilson, 1994; Maes, Mataric, Meyer, Pollack and Wilson, 1996). The debate is set to continue.

1.1. Three Components of Natural Intelligence

For the purposes of this thesis behaviour will be divided into three broad categories: (1) capabilities inherent to the individual from the moment it comes into being; (2) capabilities it may acquire as a result of interaction with its environment; and (3) capabilities acquired by processing or reformulating information or capabilities derived in any of the three categories. The first category will be referred to as "innate capabilities", the second as "learned capabilities", and the third will encompass a range of abilities broadly categorised as "problem solving", and "inductive" and "deductive inference". Some, possibly all, elements of the processes supporting categories (2) and (3) may also be an innate process inherent to the individual. Information from any category can potentially be utilised and exploited by any of the categories. Therefore the element of self and cross-reference of the categories is intentional. The "intelligence" of the individual will be based on some combination of these three basic activities (undoubtedly supported by many other activities of the individual and its structure). Intelligence will not be defined here by any specific ability, but rather by the degree or extent to which the individual can react and adapt to the circumstances that impinge upon it. One prevailing view holds that an individual can be considered intelligent solely on the basis of capabilities defined in the first category. Others argue that any useful degree of intelligence can only be displayed in individuals with significant capabilities in categories (2) and (3). This work will concentrate on the nature of intelligence as it arises from categories (1) and (2). This chapter and chapter two will consider the approaches adopted by others. Perhaps interestingly, these capabilities may arise either as a result of an evolutionary or a creational process, with little impact on the observable performance of the individual under study.

The term *animat* (Wilson, 1985, 1991) will be used throughout this work to indicate an artificial or simulated model of an animal. The term will also occasionally be used to denote properties shared by these simulated and natural animals. Specifically the term animat is used in preference to *agent*, which is used by various authors to refer variously to either an individual, or to component parts of an individual. The term animat is not intended to represent any specific organism or species type. The term *ethogram* will be used to represent a description, in operational form, of the behavioural capabilities of the animat in each of the three categories at the moment it becomes a free standing individual. The term "ethogram", after ethology[1], is apparently due to Kirsh (1991, p. 167).

1.2. Reactive Models of Intelligence

This section considers some of the issues relating to the first category of intelligent behaviour, variously named *behaviour based* (Maes, 1993), *reactive*, or *situated agent* models of behaviour (Agre, 1995). Brooks' (1991a) view of *intelligence without reason* and his (Brooks, 1991b) *intelligence without representation* arguments follow in a long tradition of stimulus-response (S-R) *behaviourism*. All argue that the majority of observed and apparently intelligent behaviour may be ascribed to innate, pre-programmed, processes available to the individual. This viewpoint is not without its critics, Kirsh (1991) for instance. Category (1), innate, capabilities of the individual derived from an evolutionary process are shared by all members of the same species (allowing for some variation between individuals). Individuals derived by a creational process acquire innate intelligence from their constructor. Similarly, we may be impressed by the advice from an expert system and yet be aware that the intelligence displayed is still derived from the knowledge of a human expert. In both cases the intelligence seems diluted. To a certain extent capabilities derived in this first category may be regarded as "intelligence without intelligence".

Innate intelligence is not, however, defined by degree. The behavioural repertoire of an insect may be completely mapped, and its ability or inability to react to any situation comprehensively modelled. At a distant end of this scale Pinker (1994)

[1](OED): ethology n. Science of character formation; science of animal behaviour

argues that human language ability, for all its complexity, is primarily innate. He cites much evidence that all undamaged humans develop language abilities to a largely uniform level of complexity by simply interacting with others, essentially regardless of (and possibly in spite of) any form of education or teaching. Specifics of vocabulary and grammar are environmentally determined, but vocabulary and grammar develop in all undamaged individuals as a matter of course during their infancy. Notwithstanding differences in their vocal tracts it is clear that, while non-human primates may be taught a limited vocabulary of symbols, attempts to teach or activate any significant tendency to structured grammar remain largely unsuccessful (Premack, 1976). Where significant progress has been reported this has lead to suggestions of observer bias.

The innate behavioural repertoire of many species has been extensively studied. Where this is done primarily by observation of the animal in its natural surroundings, the term *ethology* is often used. An alternative approach, adopted by behavioural scientists, places the subject animal in controlled experimental conditions to investigate the subject's reactions. Innate behaviour patterns are reasonably investigated by the former procedure, but aspects of learning and problem solving are often better researched by the latter method. This appears in part due to the wide range of innate activities a subject may perform, masking or hiding specific learning phenomenon under investigation.

1.3. Action Selection Mechanisms

Action Selection Mechanisms (ASM) attempt to provide a model to understand how behaviour is generated in response to the current requirements of the animal. These are specific implementations of category (1) notion of intelligence, that of unlearned or innate behaviour. They do so in a manner intended to illuminate the properties observed of living creatures. The systems discussed here tend toward the modelling of natural systems, but are not drawn exclusively from those that do so. For largely historical reasons these models concentrate on a variety of non-primate vertebrate species, including small mammals, birds and fish, whose behaviour may be closely observed and recorded. Tyrrell (1993, Ch. 8) provides a useful summary of a variety of action selection mechanisms drawn from both natural and artificial examples. Despite the huge body of observational evidence from the discipline of ethology and the subsequent introduction of computers allowing detailed simulation and testing of the various theories, there is still much controversy as to which of the many possible architectures represents the most appropriate description.

Tinbergen (1951, Ch. 5) devised a model for the organisation of behaviour based on observations by himself and others of a variety of species, including the digger wasp, the three-spined stickleback and the turkey. Tinbergen's model is a hierarchic control model of action selection. The creature is embodied with several central "instincts". Figure 1-1 models that of the reproductive instinct of the *three-spined stickleback*. Each central instinctive behaviour is inherently part of the creature, but it is not always manifest. Reproductive behaviour in the stickleback is a complex set of activities spread over a period of many weeks during the breeding season. Once initiated, say by the onset of warmer weather or lengthening hours of daylight in the spring, second level behaviours become active. In this model such behaviours are normally inhibited by a blocking mechanism. When circumstances appropriate to the conduct of some aspect of the innate behaviour are sensed an *innate releasing mechanism* (IRM) removes the block, so enabling behaviours at a lower level in the hierarchy. These sub-ordinate behaviours may then also be released by their IRMs, shown in figure 1-1 as grey coloured areas, when the conditions appropriate for their use are encountered. Lorenz had earlier proposed a simple hydromechanical analogy to illustrate the operation of the IRM (Lorenz, 1950).

Tinbergen distinguishes between *appetitive actions*, those which establish the conditions needed to continue or complete a sequence of behaviours and *consummatory actions*, which appear to "satisfy" the motivation for the action sequence and so complete it. Level 3 subordinate behaviours represent these appetitive and consummatory behaviours, and are observed and recorded by the ethologist. These behaviour units are considered to be *fixed action patterns* (FAP), groups of low level actions that may be initiated to complete some aspect of the overall instinct. Level 3 behaviour units may themselves be further sub-divided into the co-ordination of, for example, fin (level 4), and fin ray (level 5) movements, muscle activations (level 6) and so on.

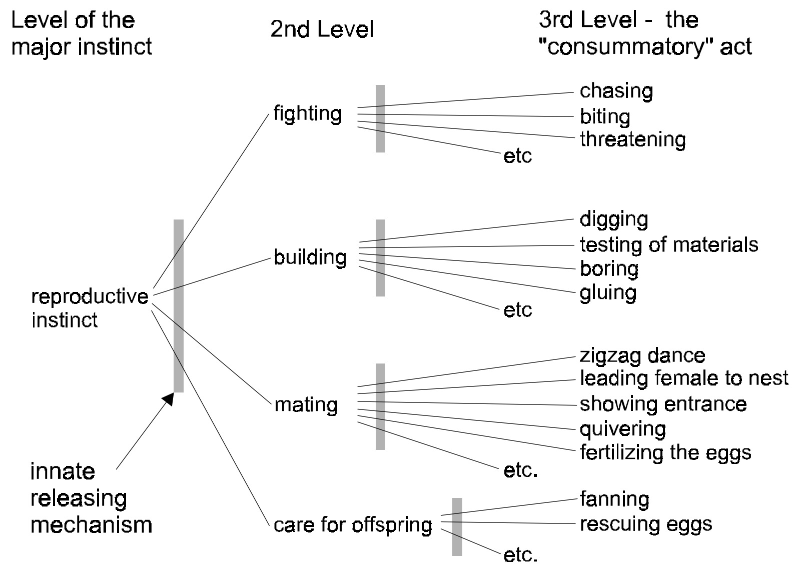


Figure 1-1: Tinbergen's Principle of Hierarchical Organisation

adapted from Tinbergen (1951), p. 104 & p. 124

Baerends (1976) presents a hierarchical model to account for the incubation behaviour of the herring gull. This model adds inhibition between superimposed control centres (level 2 behaviours), in which active centres suppress the effects of others. Friedman (1967) prepared a computer model and simulation of the concepts of innate behaviour. He retained the notion of an innate releaser mechanism, but argued that viewing level 3 behaviours as fixed action patterns was too simplistic. To counter this apparent oversimplification Friedman introduced *behavior units*, behaviour patterns controlled and maintained by feedback loops at level 3. His system was tested with a simulated artificial animal, *ADROIT*. Travers (1989) presents a computer simulation of the stickleback's innate reproductive behaviour; Hallam, Hallam and Halperin (1994) a simulation of aspects of behaviour in the *Siamese fighting fish*.

Rodney Brooks has described the *subsumption architecture* (Brooks, 1986). While not strictly an ethologically inspired model of behaviour it has proved influential in the design of subsequent reactive and behavioural models. Figure 1-2 illustrates some of the main features of the subsumption architecture. In a conventional model of robot task behaviour, Brooks argues, behaviour is decomposed into functional modules such as "perception", "modelling", "planning", and so on. Each module will be involved in the completion of many different task types. In a subsumption architecture the robot control system is decomposed into individual task-achieving modules, a "level of competence". Lower levels being responsible for simpler or more primitive activities. Each level is nevertheless responsible for a complete behaviour, having access to the sensory information it requires and the ability to send instructions to actuators. Examples of such behaviours include "obstacle avoidance" (level 0), "wandering behaviour" (level 1), "explorational and map building behaviour" (level 2), up to, say, the ability to reason about objects in the world and create plans.

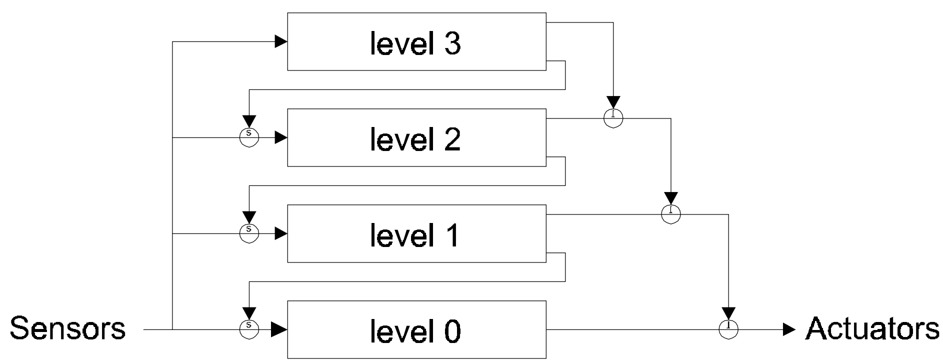


Figure 1-2: Brooks' Subsumption Architecture

adapted from Brooks (1986), p. 17 & p.18

In Brooks' model each level is created as a finite state machine. Every higher layer may subsume the behaviour of a lower layer, by modifying its input information (shown as a circled "S" on the input side of each layer in figure 1-2) and therefore adapt the lower level behaviour to its requirements. Alternatively the higher level may inhibit the output of lower layers to take control of the output behaviour (shown as a circled "I" on the output side of each layer in the figure). Brooks (1990) describes the *behavior language*, which allows behaviours defined in terms of the subsumption architecture to be complied into the native code for a variety of processor types including the Motorola 68000 and 68HC11, Hitachi 6301 and to Common Lisp.

Tyrrell (1993) argues that actions are not best selected on an all or nothing basis. Rather each module should contribute "evidence" for one or more of the possible actions available to the animat, with a “winner-take-all” strategy in place to select the final outcome to be sent to the actuators. His model is based on one devised by Rosenblatt and Payton to automatically control and navigate a mobile vehicle (Rosenblatt and Payton, 1989; Payton, Rosenblatt and Keirsey, 1990). Rosenblatt and Payton’s model overcame the potential loss of data in the subsumption architecture by allowing each behaviour module to feed (positive or negative) activations via weighted links to summation points for each action type.

Brooks’ subsumption architecture proposal is reminiscent of Paul Maclean’s *triune brain hypothesis* (Albus, 1981, p. 184). Each of three layers represents a stage in the evolution of the modern mammalian brain. All the layers have access to sensory mechanisms and motor outputs and are organised as a control hierarchy. The inner layer, layer one, is the primitive reptilian brain, equipped with reflexive and instinctive behaviours. Built over this primitive layer is the “old mammalian” brain, providing additional attributes, elements of planning, predictive abilities and some elements of memory. In turn the third layer, or “new mammalian” brain provides another set of capabilities including the sophisticated manipulation of arbitrary symbols and concepts, language and a distinct model of self. As in the subsumption architecture, each layer has access to information available to a lower layer but may also intercept and override the output of a lower layer.

Maes describes a bottom-up mechanism for action selection (Maes, 1989, 1991, 1993), which, while being primarily a computer based animat controller, addresses the problems of action selection from a broadly ethological viewpoint. Figure 1-3 illustrates the main points of her action selection model. The animat has a number of innate motivations (or, synonymously, goals), which are in turn connected to consummatory activities. Consummatory activities will, if performed, lead to a reduction or satisfaction of the attached motivation; eating assuages hunger, drinking slakes thirst and so on. Consummatory activities may in turn be linked to appetitive activities, ones that prepare the animat to complete the behaviour. Some appetitive activities lead directly to a consummatory activity; others are linked into chains of activities that lead the animat closer to the motivating goal. Thus eating food is preferable to moving towards food that can be seen, which in turn is preferable to moving to a location where food is remembered to be located, to having to explore for food.

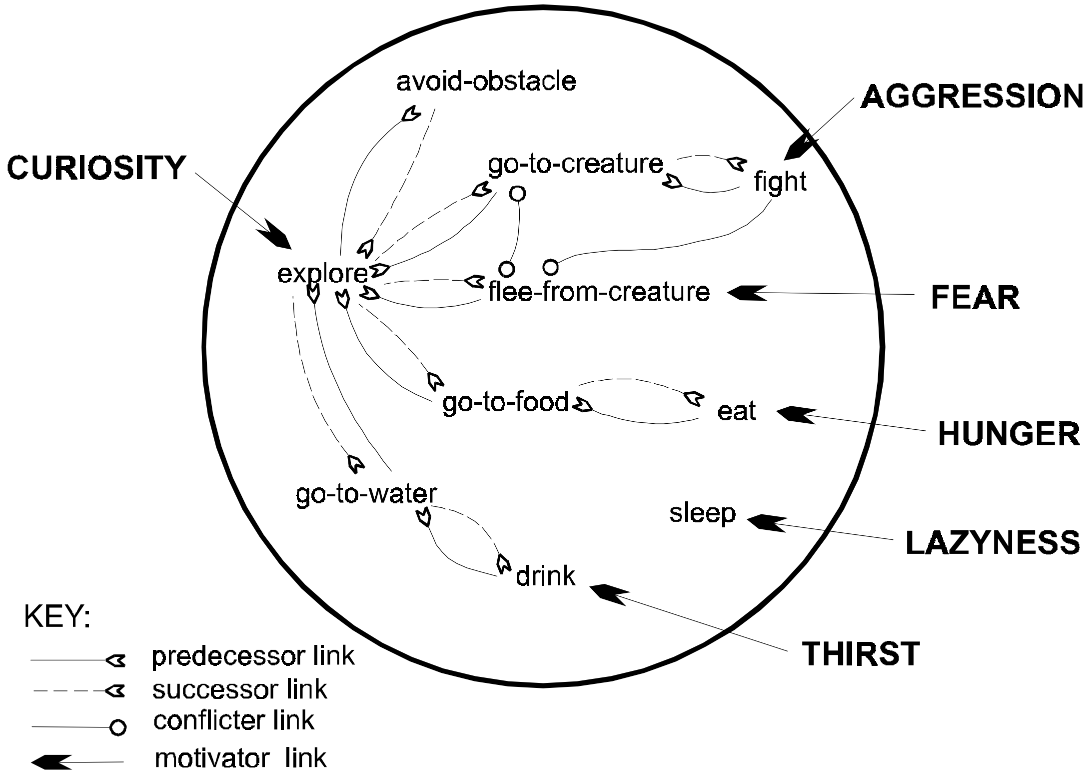


Figure 1-3: Maes' Action Selection Architecture

adapted from Maes (1991), p. 240 & p. 242

Activities are linked by a network of *predecessor links* (“——<”), a list of pre-conditions necessary to initiate an activity and by *successor links* (“- - -<”), add-list conditions arising as a consequence of performing the activity. Activities may also inhibit other activities with a *conflictor link* (“——o”). At any time each of the motivations will be characterised by a level of activation, a degree of “hunger”, “thirst”, “fear”, etc. Motivation activations spread throughout the network of activities through the predecessor links, the activation level being relative to the strength of the motivation and to the number and type of links between motivation and activity. At the same time appetitive and consummatory activities attain a level of activation based on the degree to which their preconditions are met, either by activation via their predecessor links, or directly from sensory conditions associated with the activity. *Activation* spreads in two directions, along both predecessor and activator links, inhibition via conflictor links. At any time, then, the animat may select an action based on both its current needs and the prevailing environmental circumstances in which it finds itself. Tyrrell (1994) implemented and tested Maes' action selection mechanism with a wide range of parameters and concluded that there were some significant drawbacks to the mechanism she had described.

Action selection mechanisms only address the first category of intelligence as described previously. They are an important part of the process, but insufficient to account for the range of phenomena observed. The next sections concentrate on the second category, that of learning and learned behaviour.

1.4. Arriving at a Definition of Learning

It has not proved easy to generate an all embracing definition of exactly what does, and what does not, constitute the process of learning. Learning is by no means synonymous with change; it is clearly a form of change, but one that makes "*useful changes in the workings of our minds*" (Minsky, 1985, p. 120). This definition is imprecise and incomplete. Simon (1983) extends the definition to "*learning denotes changes in the system that are adaptive in the sense they enable the system to do the same task or tasks drawn from the same population more efficiently the next time.*" Razran (1971, p17) suggests that a "*commonsense view of learning*" would be "*profit through experience,*" but immediately qualifies this to "*more or less permanent central modifications of a reaction or reactions through reacting and interacting of reacting.*" He then further excludes transient changes such as fatigue and sensory or effector adaptation. Razran and Simon have both identified a clear property of learning systems - they improve what they do by doing what they do.

Bower and Hilgard's (1981, p. 11) definition of learning develops the theme:

"*Learning refers to the change in a subject's behavior or behavior potential to a given situation brought about by the subject' s repeated experiences in that situation, provided that the behavior change cannot be explained on the basis of the subject's native response tendencies, maturation, or temporary states (such as fatigue, drunkenness, drives, and so on).*"

This definition amplifies the notion of change precipitated by experience and made manifest in behaviour. Thus category (2) intelligence is distinguished from category (3) intelligence in that the change is mediated by the receipt of external information, rather than a reprocessing of internally held knowledge. The distinction becomes increasingly blurred as previously learned information is itself reformulated. This last definition also introduces an element of permanence, or at least semi-permanent change, which does not readily revert to the previous condition without further experience within the environment. It is clear from these definitions that while learning is a change in behaviour, not all changes in behaviour can be regarded as learning. Chapter two reviews possible behavioural mechanisms that can be described as learning, the next sections consider some forms of behavioural change that are excluded by the definitions.

1.4.1. What is Not Learning

Bower and Hilgard's definition also excludes a number of other sources of change that should not be classified as learning. These sources of temporary change, such as fatigue or the influence of drugs, are essentially reversible and the animal will revert to its original behaviour once the effects of the influence abate[2]. Similarly, the effects of *habituation* and *sensitisation* are normally excluded from definitions of learning. There are many situations in which an organism will come to react less frequently or with less vigour to a particular sensation apparently only due to the frequency of presentation of that stimulus. The organism "habituates" with respect to the stimulus. An organism may also react more vigorously to a stimulus that has been withheld for an abnormal period. The organism is "sensitised" with respect to the stimulus. Both conditions are transitory and reaction reverts to normal levels once the stimulus regime is stabilised.

Maturation, on the other hand, does represent a permanent change, but one that also falls outside the definition of learning. Maturation represents behavioural changes in the organism that take place essentially independently of the individual

[2]Which is not to say that the organism will not modify its behaviour as a consequence of these influences. A drinker might subsequently imbibe more due to the pleasing effects of inebriation, or less due to the consequences of a hang-over. In either case the intoxicating effects of the alcohol ingested may be considered essentially transient.

organism's experience in its environment. Such behavioural changes mirror physical changes due to growth, and may be linked to or co-ordinated with the development of physical attributes. As an example of the maturation process Altman and Sudarshan (1975) investigated the development of reactions to different environmental situations in new-born rat pups, showing the appearance of successively more complex behaviour patterns during the first weeks of life. These changes are apparently pre-programmed to occur in the organism, in much the same manner that innate tendencies appear as pre-programmed reactions to specific stimuli.

Imprinting may be considered as a special case of maturation, in which the individual is pre-programmed to incorporate an external stimulus as releaser or trigger for some other pre-programmed behaviours. Only the stimulus adopted varies between individuals of the species, the mechanism to adopt some stimulus (often within recognisable limits), and the reactions it will subsequently elicit appear to be pre-programmed. Imprinting was first recognised by the ethologist *Konrad Lorenz* (1903-1989). He noticed that graylag goose chicks, which normally follow their mothers, would follow a human in preference to their mother if exposed to a human individual at a critical stage in their development. Imprinting is characterised by a typically rather narrow *sensitive period*, during which the effect develops easily. Ducklings (Hess, 1959) are most sensitive to the effect at between 13 to 16 hours after hatching. Attempts to imprint before 5 hours or after 21 hours from hatching invariably fail. The imprinting phenomenon has been widely researched and has been demonstrated in a variety of avian and mammalian species (Dewsbury, 1978, pp. 140-153).

1.5. A Caveat

This work strives to present a "biologically inspired" model of an animat controller; it is not intended as a specific model of any particular animal or species. Such models have been prepared, and often shed further light on the nature of the creature being emulated (Arbib and Cobas, 1991; Arbib and Lee, 1993; Hartley, 1993; Mura and Franceschini, 1994; and Webb, 1994, for instance). Cliff (1991) has promoted the term *computational neuroethology* for this type of study (Beer and Chiel, 1991), Sejnowski, Koch and Churchland (1988) the term *computational* *neuroscience*. Roitblat, Moore, Nachtigall and Penner (1991) propose *biomimetics* in relation to their neural network model of echolocation in dolphins.

In designing the animat controller some of the essentially engineering solutions that arise are resolved on the basis of *biological plausibility*. By adopting evidence drawn from many different species, under many different experimental regimes, general principles may be identified and integrated into a whole. However, it is unreasonable to assume that capabilities are evenly distributed across the animal kingdom. There is diversity at every point and at every level, so that a generalised model cannot be expected to account for detailed reactions in specific individuals.

1.6. Thesis Outline

This chapter has introduced the idea that animal intelligence is composed of three component parts, (1) innate behaviour, (2) learned behaviour and (3) behaviour directed towards inferring and deducing new knowledge from existing knowledge. As well as defining some terms, several models of innate behaviour were described and what does and does not constitute learning was also considered.

Chapter Two develops the theme of learning, concentrating on learning in reactive systems. A review of learning from a historical perspective introduces many important concepts and illustrates the spread of the problem being addressed. A review of recent and current research into computer models concentrates on work in reinforcement and *Q*-learning methods, classifier systems and artificial neural networks. The chapter also considers the evidence for a cognitive or goal driven view of learning and behaviour in animals. Existing models of intermediate level (sensory-motor) cognition are reviewed.

Chapter Three considers the role of hypothesis generation and verification by experiment at a behavioural level, consistent with reported observations of animal behaviour. A comprehensive set of postulates for a new *Dynamic Expectancy Model* is developed which combines the apparently disparate threads of reactive behaviour, perception and action, goal setting and pursuit, and learning.

Chapter Four develops a computer simulation algorithm (*SRS/E*) from the Dynamic Expectancy Model presented in chapter three. This chapter describes the data structures and processes required to implement the Dynamic Expectancy Model.

Chapter Five describes an experimental environment attached to the SRS/E program implementation and describes the facilities available to an investigator using the program.

Chapter Six reports a series of experiments with the SRS/E algorithm. These experiments are constructed to allow direct comparison with other published reinforcement learning algorithms, and to several well-established procedures from the behavioural sciences, which are adapted for use with the SRS/E program.

Chapter Seven describes some possible extensions to the Dynamic Expectancy Model to enhance the SRS/E algorithm.

Chapter Eight concludes by reviewing the relationship of reinforcement learning to cognitive structures and proposing *Expectation Based Learning* (*XBL*) as a fruitful line of research investigation for the future.

Appendix One gives a complete description of the execution cycle for the SRS/E algorithm, described in detail in chapter four.

A bibliography of references is attached, as is an index of topics and author citations by page number.

Chapter 2

2. Theories of Learning

Learning in animals and humans has been intensively studied in the scientific manner since the beginning of this century. Notwithstanding the quantity and quality of research undertaken during this period radically new theories describing the nature of the learning process in animals have appeared relatively infrequently. The first part of this chapter will concentrate on the major theoretical stances of the 20th century. In particular the *classical conditioning* paradigm developed by Russian academician Ivan P. Pavlov (1849-1936); reinforcement theories, initially postulated by Edward L. Thorndike (1874-1949); and the operant conditioning paradigm, established by B.F. Skinner (1904-1990). The second part of the chapter concentrates on the cognitive viewpoint originally developed by Edward C. Tolman (1886-1959). There are many comprehensive reviews of natural learning, Hall (1966), Bolles (1979), Bower and Hilgard (1981), Schwartz (1989), Lieberman (1990) and Hergenhahn and Olson (1993), to cite a selection. Bower and Hilgard's classic "Theories of Learning", now in its fifth edition since first publication in 1948, is used as a primary source for this work. Kearsley (1996) has prepared summaries of some 50 "learning theories", although many of these refer to specific learning phenomena in humans or to theories of education and instruction.

Given the quantity of experimental data accumulated supporting each of the various approaches to learning it is well-nigh impossible to totally discount their relevance, yet each will effectively explain or predict only a limited range of experimentally obtained data. Indeed each position will have been modified, often several times, in the light of new results. In the context of the "biologically inspired" animat these existing theories and experimental studies provide the underlying concepts and results used to guide design decisions. Emphasis will be placed on determining the role played by any particular phenomenon in influencing or determining the overall behaviour of the animat - a “systems approach”, rather than a focus on exact duplication or representation of every phenomenon.

A parallel and more recent approach to the understanding of learning has arisen as “machine learning”, which attempts to synthesise, describe and analyse learning phenomena as a computational or algorithmic process (Carbonell, 1990; Langley, 1996, for reviews and summaries). There has been only limited cross-fertilisation of ideas and the two approaches, natural and artificial, have tended to remain largely distinct. Nevertheless the computer provides an effective platform on which to test ideas and theories related to natural learning.

This chapter will discuss computational models of learning germane to the development of a learning model later in this work. Each of the computational models in the first part of the chapter is broadly recognisable as having a “stimulus-response” or “behaviourist” format, models that select actions on the basis of prevailing input stimuli. The basis of future choices being mediated by a (typically externally) applied reward or error indication. Three main approaches will be considered in some detail, the “reinforcement learning” model, the “classifier system” model and the “connectionist” or “artificial neural network” (ANN) model. The computer models of learning described in the second part of the chapter clearly owe their origins to the cognitive standpoint.

2.1. Classical Conditioning and Associationism

Classical conditioning pairs an arbitrary sensory stimulus, such as the sound of a bell, to an existing reflex action inherent in the subject animal, such as the blink of an eyelid when a puff of air is directed into the eye. The phenomenon was first described by *Ivan Pavlov* during the 1920’s, and the experimental procedure is encapsulated by the earliest descriptions provided by Pavlov. Dogs salivate in response to the smell or taste of meat powder. Salivation is the *unconditioned reflex* (UR), instigated by appearance of the *unconditioned stimulus* (US), the meat powder. Normally the sound of a bell does not cause the animal to salivate. If a bell is sounded almost simultaneously with presentation of the meat powder over a number of trials, it is subsequently found that the sound of the bell alone will cause salivation. The sound has become a *conditioned stimulus* (CS).

Pavlov and his co-workers studied the phenomenon extensively. By surgically introducing a fistula into the dog's throat, saliva may be drained into a calibrated phial and production measured directly as an indication of response strength. Taking care to ensure that extraneous sensory signals are excluded, the strength of association adopts a distinctive curve. Initial association trials show little response, followed by a period during which the association gains effect rapidly, finally reaching an asymptotic level, possibly due to the production capacity of the gland. Each trial takes the form of one or more pairings of US and CS to establish the association, followed by one or more presentations of the CS alone to test the strength of the effect. Several additional features of the phenomena are noteworthy. If, subsequent to establishing an association, the CS is presented without further CS/US pairings the effect diminishes over following trials, a procedure known as *experimental extinction*.

The animal's response to the CS may be manipulated in a number of ways. The CR will typically be evoked to a CS similar, but not identical, to that used for the initial conditioning; for instance, tones of a similar but different frequency. This spread of CS stimuli may be refined by randomly presenting positive trials, CS+, where the association is present, and the CS tone is at the desired centre point frequency with unassociated CS- trials where the tone is not at the desired frequency. After a suitable number of trials the subject animal indeed responds to the CS+, but not the CS- stimuli. The procedure is known as *differentiation*, and has been used in various forms to determine the sensory acuity of various species. Similarly the spread may be broadened by a complementary process of *generalisation*. It has further been found that the speed and strength with which the conditioned association may be formed is critically dependant on the timing relationship between presentation of the CS and US. It is almost universally noted that the CS must precede the US for the conditioned association to develop. This time may be in the order of several hundred milliseconds, but the optimal interval depends on the nature of the association and the species under test. This observation has lead some observers to comment as to an anticipatory or predictive nature of the phenomenon (Barto and Sutton, 1982).

Classical conditioning has been extensively researched. Razran (1971) indicates that he has identified "*tens of thousands of ... published experiments and discussions of Pavlov launched research and thought,*" and provides a bibliography of some 1,500 titles of (primarily) Russian and American research. It is clear that the phenomenon is widespread and highly replicable. Bower and Hilgard (1981, p58) have commented "*almost anything that moves, squirts or wiggles could be conditioned if a response from it can be reliably and repeatably evoked by a controllable unconditioned stimulus.*" Rescorla (1988) argues that Pavlovian conditioning still has much to offer in our understanding of the learning of relationship between events, rather than as a simple connection to the unconditioned response. It is, however, clear that pure associationism of this form provides limited opportunity to explain the majority of animal learning phenomena.

Several effective models of classical conditioning have been produced. Grey Walter (Walter, 1953) constructed an electronic model (*machina docilis*) from thermionic valves that produced a quite reasonable simulation of the phenomenon. The unit was also designed to integrate with his ingenious free-roving, light-seeking automata *machina speculatrix*; also constructed from miniature values, relays and motors. Barto and Sutton (1982) and Klopf (1988) have produced computer simulations of single neurone models capable of simulating a wide range of experimentally observed conditioning effects. Scutt (1994) describes a simple adaptive light seeking vehicle based on a classical conditioning learning strategy.

2.2. Reinforcement Learning

Reinforcement learning stands as one of the most enduring models of the learning process. First described by Edward L. Thorndike (1874-1949) as the *law of effect*. This model of learning arose from Thorndike's observations of cat behaviour in its attempts to escape from a cage apparatus incorporating a lever the cat may operate to open an exit hatch. Cats react as if to escape on being enclosed in this manner. Thorndike noted that at first the cat would exhibit a wide range of behaviours including attempting to squeeze through any opening, clawing, biting and striking at anything loose or shaky[3]. Eventually one of these actions by the animal operates

[3]Paraphrased from Thorndike (1911)

the lever and it can escape. When placed in the apparatus on successive occasions the animal would typically escape sooner and eventually, after many trials, learn to operate the lever immediately.

These observations introduced several ideas. First was that of learning by *trial and error*; the subject makes actions essentially at random until some “satisfactory” outcome is encountered. Second was that learning appeared to be an incremental process; performance improves gradually with practice. Third was that of reinforcement, the probability that the animal will repeat some action is increased if it has in the past been following directly by a “reinforcing” or “rewarding” outcome. The more frequently the reinforcing outcome, the higher the probability, strength or frequency that the prior behaviour will be selected. It rapidly became apparent that some outcomes were inherently reinforcing, such as presenting food to a hungry animal, while others were not. Equally, the removal of an adverse condition (such as being trapped in a cage) might be as effective a reinforcer as was being presented with food when hungry. The presentation of a wholly adverse outcome (*aversion* or *punishment* schedules), such as the application of electric shock, leads to rather less predictable results. Reinforcement learning differs substantially from that of classical conditioning in that it is contingent upon the arrival of a reinforcing “reward”, whereas classical conditioning only depends on contiguity of stimuli. Reinforced behaviours may also be subject to differentiation and extinction under appropriate experimental conditions.

Such notions of reinforcement learning formed an ideal complement to the behaviourist school of psychology, established by *John B. Watson* (1878-1958) during the first decades of this century, and in particular the S-R (stimulus-response) school of behaviourists. In its most extreme form *S-R behaviourism* postulates that all behaviour can be explained in terms of actions selected on the basis of current stimuli impinging on the organism. Learning reduced to simple strengthening or weakening of connections between stimulus and response is therefore very attractive. S-R behaviourism, along with the necessary modifications, has been very influential throughout much of this century and finds current expression in the ideas of Rodney Brooks (intelligence without reason) and Philip Agre (reactive agents). Richard Sutton has been active in promoting computer models of reinforcement learning, of which more in the next section.

It soon became apparent that many factors affected the amount and rate of learning. *Clark L. Hull* (1884-1952) attempted to identify and subsequently quantify these factors and the effects they may have. Hull's work is extensively reviewed and analysed by Koch (1954), and summarised in Bower and Hilgard (1981, Ch. 5). Hull's model changed over time in response to new experimental observations. Equation 2-1 illustrates (and it is only illustrative) some of the major factors he identified and the manner in which they may be related.

$$ {}_{S}E_{R} = ({}_{S}H_{R} \times D) \times V \times {}_{S}O_{R} - ({}_{S}I_{R} + I_{R}) \qquad \text{(eqn. 2-1)} $$

In Hull's model *net response strength*, ${}_{S}E_{R}$, is primarily related to "habit", *H*, the connection established through reinforcement learning between stimulus ($_{S}$) and response ($_{R}$), and to *motivation* or *drive, D,* reflecting the current desirability of the reinforcement outcome. A satiated rat will not necessarily perform actions resulting in reinforcing food rewards. Habit connection strength is built up over many reinforcing trials, described by a negatively accelerating learning curve. *V* relates to the "goodness of fit" between the evoking and training stimuli. An *oscillatory factor*, ${}_{S}O_{R}$, provides temporary perturbations to response strength and is required to explain the natural variation of behaviour experimentally observed. Extinction phenomena are expressed as an inhibition factor, ${}_{S}I_{R}$, which counteracts the habit strength (I_{R} represents habituation due to response fatigue). Although Hull performed extensive series of experiments to establish exact parameters for each term the formulation fell into disuse. This was partly due to a reduction of interest in reinforcement learning, and partly because Hull was eventually obliged to postulate more than 15 separate terms. As a consequence this expression of reinforcement learning became too unwieldy for effective analysis.

The theories of Thorndike, Hull and the other S-R behaviourists were connectionist; a single link made between stimulus and response, strengthened and weakened over time according to some schedule of reinforcement. It has become clear that the development of the S-R link need neither be a smooth progression from weak to strong, nor develop at equal rates between individual animals used in a series of experimental trials. Generally, the smooth learning curve only becomes apparent once the results from several individuals are averaged. Each individual's activity shows marked variation in performance, though invariably the task can be completely learned. In some cases the animal attains apparently perfect task performance in a single trial, an effect referred to as *one-shot learning*. William Estes and his co-workers formulated a radically different approach, *stimulus sampling theory* (Bower and Hilgard, 1981, Ch. 8). Stimulus sampling theory provides a mechanism to account for one-shot learning observations and accounts for the appearance of the negatively accelerating curve when many individual learning trials are averaged. This approach subsequently developed into a more general *mathematical learning theory* approach.

In the stimulus sampling formulation all connections between stimulus and response were either absent or completely made. It also assumes that the individual was subject to many individual stimuli. At any time some sub-set of these stimuli would be active and so be subject to reinforcement. Therefore, at every reinforcing trial some subset would be active. Given a limited set of stimuli available to the animal, and a sampling regime that selected only a sub-set of the stimuli it is relatively straightforward to demonstrate that, on average, the selected sub-set will contain elements from the previously reinforced pairs with an increasing probability which accurately mimics the negatively accelerating learning curves already observed. This theory neatly explains the variability in performance between individual trials - chance determines whether the stimuli sub-set selected contains many or few previously reinforced pairings. If the initial set of reinforced parings exactly matches those intended by the experimenter, one-shot learning appears to take place. The formulation may also account for many of the other phenomena associated with the reinforcement learning paradigm.

2.3. Computer Models of Reinforcement Learning

Recent years have shown a considerable revival in research interest in reinforcement learning investigated as a form of machine learning (Sutton, 1992; Kaelbling, 1994, 1996). Two specific problems have been the focus of this renewed interest. First is the problem of delayed reward. This problem may be illustrated by considering a game playing task in which the players repeatedly play and have the task of improving their chances of winning. Reward is received at the conclusion of the game[4], credit for winning and debit for losing. During the game there is no indication of whether a move was good or bad. Yet during the game the player must make decisions about the move to be made on the basis of the current game situation. In an early paper Minsky (1963) referred to this as the *credit assignment problem*. If it is possible to accurately classify the current game situation, it should then be possible to assign a weight or desirability to this current situation that best categorises the move that should be made to optimise the player's overall chances of success in the game taken as a whole. The second problem attracting attention is how to react if the situation cannot be detected, fully recognised or accurately classified (Whitehead and Ballard, 1991; Chrisman, 1992; Lin and Mitchell, 1993; Whitehead and Lin, 1995; McCallum, 1995).

The solution to the former problem is critical if reinforcement learning is to adequately explain how an animat may give the appearance of goal directed behaviour in an ostensibly stimulus-response reinforcement paradigm. It is an interesting problem in that it appears to contradict the overwhelming body of experimental evidence from natural learning that indicates that reinforcement by reward (or aversion by punishment) is only effective if applied almost directly following the stimulus event. Sutton's (1988) reinforcement system, the *temporal differences method* (TD(λ)), exploits changes in successive predictions, rather than any overall error between an individual prediction and the outcome of a sequence of events to achieve the required disassociation of action now with later outcome. Computation of changes of individual decision weights following individual predictive steps followed a variant of the well-established Widrow-Hoff rule (Widrow and Hoff, 1960). Sutton (1991) identifies several additional well-established strategies by which reinforcement may be assigned to modify a behavioural policy, illustrated with examples drawn from machine learning algorithms dating back to the 1950's.

Reinforcement learning can be made more tractable if the overall animat task is split into a number of smaller tasks. Mahadevan and Connell (1991) describe a

[4] This is only to illustrate the problem, current game playing algorithms do not necessarily rely on the techniques of reinforcement learning.

robot controller based on reinforcement learning techniques, in which a simple[5] box pushing task is decomposed into three sub-tasks, "find", "push" and "unwedge", incorporated into a *subsumption priority architecture*. Learning in each sub-task is moderated by its own reward signal, "F-reward", "P-reward" and "U-reward". Millán and Torras (1991) describe an algorithm for learning to avoid obstacles in a simulated 2-D environment using a reinforcement learning method. Lin (1991) emphasises the role of a teacher in guiding reinforcement learning for a simulated mobile robot. As in the Mahadevan and Connell approach there are set reinforcement signals applied for completion of various sub-tasks, for instance, +1.0 if the robot successfully negotiates a doorway, +0.5 if it succeeds but also collides with the door-post, but -0.5 if collision alone occurs. The door passing task could be completed with or without a teacher, but a docking task required the teacher's intervention to be successfully learned. Lin's algorithm overcame the partitioning problem by recording past events in a trace, using a process of *experience replay*. Giszter (1994) describes an extension to Maes' action selection network to allow a form of reinforcement learning in a simulation of various frog spinal reflex behaviours. Maes and Brooks (1990) describe a learning algorithm applied to development of co-ordinated locomotion in the six-legged robot *Genghis*. Much recent attention in the field of reinforcement learning has focused on the *Q*-learning technique developed by Christopher Watkins, and has utilised the Markov environment as an experimental platform - these two topics are considered in some detail next.

2.3.1. Markov Environments

Markov environments (Puterman, 1994) represent a highly stylised description of an environment and are commonly employed in reinforcement learning research. A Markov environment is described in terms of four components:

- S - a state-space, described by some set of individual states, s
- A - the actions a possible in each state s
- T - a transition function describing the consequence of applying any action a in some state s

[5] "Simple?" It is this author's experience that the box pushing task with a robot of the form Mahadevan and Connell describe is far from straightforward.

R - "reward" r obtained by entering some state s

The *markov property* defines that transitions and outcomes depend only on the current state and the action; thus there is no need to know the system's history. This is a property of this particular model, not necessarily of any real process. A *policy* is a mapping of states and actions into rules for deciding which action to take in any of the states. A *stationary policy* indicates that the same action will result in the same transition between states on each application, thus: $T(x_t,a_t) \rightarrow y_{t+1}$. The transition defined by the action a in state x at time t always results in the state y at time t+1. It may be proved that an optimal strategy exists for the selection of actions in a stationary markov process (Ross, 1983). This set of conditions will be referred to later as a Finite Deterministic Markov State-Space Environment (FDMSSE). A *stochastic policy* indicates that a transition will transform between states on a probabilistic basis, thus: $P_{xy}(a) = \Pr(T(x,a) = y)$, which describes the probability that action a will transform the current state x to some other state y. This set of conditions will be referred to later as a Finite Stochastic Markov State-Space Environment (FSMSSE).

2.4. *Q*-learning

Watkins (1989) describes *Q-learning*, a novel incremental *dynamic programming* technique by a *Monte-Carlo method*, and applies this technique to the animat problem. Under well-defined conditions (the Markov assumptions) this method is shown to converge to an optimal stationary deterministic policy solution (Watkins and Dayan, 1992). The method concerns itself with determining a set of measures, Q, for each action, a, in each state, x. *Quality-values*, $Q(x,a)$, indicate the overall reward that might be expected for taking action a in state x. At the conclusion of the Q-learning procedure an animat may select an action a in any state x according to the set of Q values and be assured that the action represents a step on the (or an) optimal path to maximise reward.

2.4.1. *Q*-learning - Description of Process

For each step the animat takes some action a available to it in the current state x and may receive some reward r on completion of the step. The quality-value, $Q(x,a)$, can then be updated according to:

$$Q(x,a) \leftarrow (1 - \alpha)Q(x,a) + \alpha(r + \gamma \max_{b \in A} Q(y,b)) \qquad \text{eqn.(2-2)}$$

The learning rate (α, expressed as a fraction) determines the effect of the current experience relative to past experiences on the learning process. The discount factor (γ, also expressed as a fraction) determines the relative importance of immediately achievable rewards, as opposed to those which may be achieved at some point in the future. For this procedure to converge to an optimal set of values, $Q^*(x,a)$, each action a must be performed in every state x for which it is available an infinite number of times. Up to this point the selection criteria, $Q(x,a)$, allowing the selection of an appropriate action ($a = \max_{b \in A} Q(x,b)$) remains an estimate of the optimal strategy. To achieve convergence the learning rate α is successively reduced towards zero. Initial values of $Q(x,a)$ may be set arbitrarily, say at random.

Control must be maintained over the degree to which the animat has the opportunity to explore its environment against pursuing the optimal known reward path at any stage in the learning process. This is the *exploration-exploitation tradeoff*. If a partially computed policy is adopted prematurely, exploration is curtailed and learning is compromised. The animat pursues paths based on habit and the discovery of the optimal path delayed. To tradeoff exploration to exploitation Sutton has proposed the use of a Boltzmann distribution to increasingly bias the selection of actions on the basis of Q in preference to an exploratory strategy, say the selection of random actions. The probability of selecting the action a reflecting the current maximum $Q(x,a)$ as opposed to some other possible action is determined by the temperature coefficient T. As the "temperature" is lowered towards zero the animat more frequently selects the policy action. The *Boltzmann (soft max) distribution* employed is given by:

$$P_x(a) = \frac{e^{\frac{Q(x,a)}{T}}}{\sum_{b \in A} e^{\frac{Q(x,b)}{T}}} \qquad \text{eqn. (2-3)}$$

In a practical demonstration of *Q*-learning, Sutton (1990) defines the environment as a matrix of states *x* in which the animat may make the transition to adjacent states *y* by taking actions *a*. One state is defined as the goal *g*, and the animat will receive one unit of reward *r* each time it enters state *g*. There is no other source of reward. At the start of each trial the animat is placed at a starting state in the matrix. The trial is concluded once the animat enters the goal state and receives the reward. A new trial is begun with the animat again placed at the start. Learning performance is conveniently measured by the rate reward is accumulated over time. Initially, with a high value for *T*, the animat selects essentially random, exploratory, actions. As learning progresses the animat increasingly selects actions based on the learned policy it has created. Convergence is indicated when the animat always selects the path that maximises reward accumulated in the long term. Sutton's research and results are considered again in more detail later.

2.4.2. Some Limitations to *Q*-learning Strategies

One obvious limitation of the strategy is the large number of trials that must be performed before the effects of learning may propagate to states distant (in terms of intervening states) from the reward state. Sutton (1990) proposed an alternative algorithm, *Dyna-Q*, by which the animat records visits to states in a separate data structure, and uses this to "rehearse" (in a process Sutton refers to as "planning") actions to increase the apparent, or observed, speed of learning. Peng and Williams (1996) and Singh and Sutton (1996) both describe algorithms which record information about states visited in the recent past ("traces"), making them eligible for learning immediately whenever a reinforcing signal is encountered. Both algorithms combine aspects of *Q*-learning and reinforcement learning with *the temporal differences method* of Sutton (1988). Maclin and Shavlik (1996) have described a method by which advice from an external observer can be inserted directly into the *Q*-learner's utility function to reduce the number of training trials required and so speed learning.

Once created the policy map is essentially "static", changes to the shape of the underlying state-space diagram are not readily reflected in the *Q* values. Sutton (1990) describes the effects of an *exploration bonus*, which enables the animat to continue some level of exploration throughout its existence. The animat may then take advantage of shorter routes should they appear, or alternative paths should the existing one become blocked. Arbitrary exploration of this form must affect the optimality of the overall solution, and in turn compromise the ability of the algorithm to generate convergent solutions. Moore and Atkeson (1993) describe a similar mechanism, *prioritized sweeping*, which provides for an extra system parameter (r^{opt}) directing the system to explore areas of the environment that are currently underdeveloped - "*optimism in the face of uncertainty*." Novel transitions are selected in preference to well-tried ones in the hope that a large, but as yet undiscovered, reward state might be encountered. A separate system parameter (T_{bored}) quenches this optimism once the calculated confidence that the long term estimate of reward for the state reflects the true value. These modifications are reported to give significant performance gains over both the original one-step *Q*-learning algorithm and Sutton's Dyna modifications.

A further limitation is presented by the nature of the goal state and the reward it delivers. Several states may deliver reward and reward may be introduced at any step in the learning process. It may be that the animat might have many goals (as discussed earlier), the actions required to pursue each goal being different, and the nature of the reward received dependent on the desirability of the goal or goals active at the current time. Tenenberg, Karlsson and Whitehead (1993) describe a modular *Q*-learning architecture with many fixed size *Q*-learning modules each responsible for achieving a specific goal; the final action presented to the environment being selected by an arbiter module. Humphrys (1995) describes a system of many *Q*-learners, each acting as an independent agent, which must compete to provide the final output action for the animat. Competition between the individual internal agents is mediated by an additional algorithm (*W-learning*).

2.5. Classifier Systems

Classifier Systems (Booker, Goldberg and Holland, 1990) represent an elegant approach to the construction of stimulus-response artificial learning systems, which directly address the problems of delayed reward. Figure 2-1 shows the main component parts of a classifier system. The condition-action pairing in a classifier system is encapsulated into a list of *classifiers*. Classifiers test the status of messages recorded on a message list. Messages are all encoded as fixed length bit strings. Classifiers whose condition part exactly matches one of the messages on the message list may "post" their bit string message onto the message list. Some bit positions in the message string are reserved to indicate the status of various input sensors. Some positions will be written by the output messages of the classifiers. Some messages will act as output signals, to be directed to effectors. Each message has a tag, typically a short prefix bit code, which records the type of the message being encoded. These tags mean that certain message will only be considered by a sub-set of those classifiers that match that specific tag bit pattern. The condition bit string is composed of either 1's, or 0's or #'s. A '1' or a '0' in the condition part directly matches to a '1' or '0' in the message, a '#' may match either a '1' or a '0' - a *don't care* symbol. In this way a classifier condition may be required to match a message in the message list exactly (where it is composed of only '1's and '0's), or it may generalise over many possible messages in the message list (where the classifier condition contains '#'s).

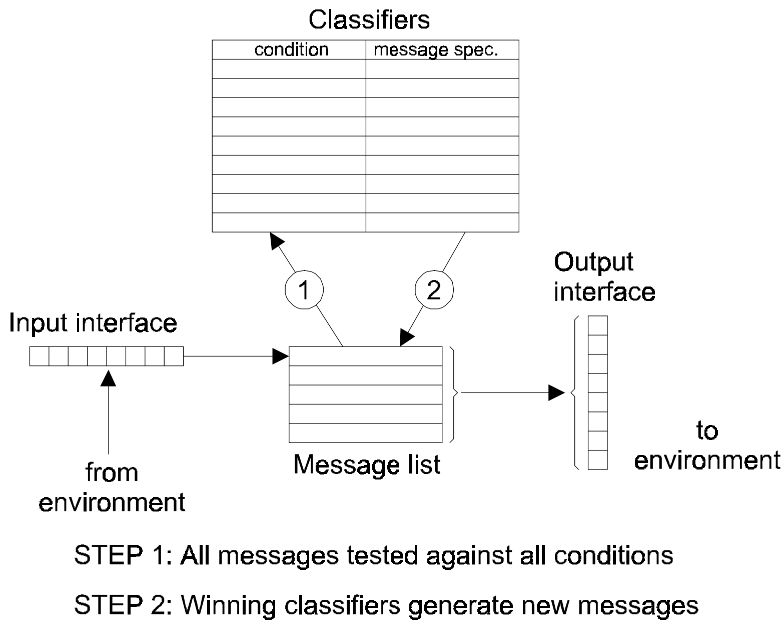


Figure 2-1: A Classifier System

after Booker *et al* (1990, p. 240)

Each classifier has associated with it a numeric quantity, the *strength value* of the rule, which reflects the classifier rule's "usefulness" to the system as a whole. In a system of any size the likelihood that a matching classifier' s message will be written to the message list is in proportion to its strength value. Strength values are updated by the reinforcement learning component of the system in proportion to the contribution the classifier rule made in garnering any reward. The algorithm for apportioning credit amongst the various classifier rules, even though reward events are sparse, is referred to as the *bucket-brigade algorithm*.

A classifier system operates with three basic sub-systems, a performance element, a credit assignment element and a discovery element. Heitkötter and Beasley (1995) provide a pseudo-code listing of the classifier system learning algorithm. The performance element is responsible for matching classifier conditions to the message list, maintaining the message list by adding new classifier message specifications and selecting external output actions. The strength of each classifier rule that successfully posts a message to the message list is reduced by a *bid amount*. This bid amount is calculated on the basis of the current strength value and the specificity of the rule (the number of “don’t cares” in the condition). The strength of any classifier which bids but fails to post its message is left unchanged. However, all the classifiers that previously posted messages used by the winning classifier subsequently receive an increase in strength based on the value of the successful bid.

Classifiers which bid and post messages just prior to external reward are credited with strength increases directly by the credit assignment element. Those which enable these classifiers receive a “share” of this reward - and so on throughout the system. The overall effect is to increase the strength of classifiers that are consistently implicated in successful or rewarding activities. In turn their greater strength increases the probability that they will be activated, and so receive reward. In this way the bucket-brigade algorithm orders the usefulness of all the classifiers in the system, and improves the external performance of the system. As with the *Q*-learning algorithm, classifiers distribute their success to those which contributed to it.

The discovery element allows for the creation of new classifier rules according to a *genetic algorithm* (Holland, 1975; Dawkins, 1986). This discovery component takes the best members of the population of classifiers and modifies or recombines them to create offspring classifiers that may be better fitted to the environment and task. The principal genetic method employed in classifier systems is that of the *genetic crossover*, which randomly exchanges selected segments between the pair of parent classifiers to create two new offspring classifiers. *Mutation*, in the form of random inversion of elements in the bit string, may also be employed. To maintain the size of the classifier list, the weakest classifiers may be discarded.

Wilson (1985), creator of the term “animat”, was the first to directly apply the techniques of classifier systems to the animat problem. Ball (1994) describes an animat control system combining a *Kohonen feature map* and conventional classifier system to create a “hybrid learning system” (HLS). The Kohonen map providing a self-organising element to pre-process sensory information into sub-symbolic features passed to the classifier component. Similar maps have been proposed as models of cerebral cortex function (as in Albus’ CMAC, q.v.) Dorigo and Colombetti (1994) decompose the animat task into several classifier systems in the *ALECSYS* algorithm to demonstrate learning and control in a small mobile robot. Venturini (1994) describes the *AGIL* system. AGIL incorporates modifications to the basic classifier system format that explicitly balance the effort the animat will expend in exploration of its environment to that of exploiting its learned knowledge. Riolo (1991) modifies the classifier system format to allow a form of lookahead planning. Dorigo and Bersini (1994) argue that classifier systems and *Q*-learning are essentially similar methods of reinforcement learning, separated more by a research tradition than essential technical differences. They demonstrate that a considerably simplified form of the classifier system may be treated as equivalent to a tabular form of *Q*-learning.

2.6. Artificial Neural Networks

Artificial Neural Networks (*connectionism*) represent a distinct approach to modelling and creating behaviour patterns. Much of the work in this area may be traced back to an abstract model of the neurone developed by McCulloch and Pitts (1943). The hope is that these units in some way provide a reasonable analogue of the internal function of the brain and nervous system of animals[6]. Figure 2-2 illustrates some of the features of this type of model. The central component of the model is a summation unit (Σ) that accepts signals from several sensory inputs (S_1 .. S_n) via weighted "synaptic" connections (W_1 .. W_n). Individual weights may be continuously adjusted between some negative value and some positive value. A *threshold unit* on the output side of the summation unit converts the output into a binary response from the simulated neurone.

[6] Leading to a early surge of optimism within the Machine Intelligence community that perhaps networks of simple units, initially connected at random and subsequently subjected to simple learning regimes would lead to complex self-organised behaviour. The idea is still seductive, but in the intervening half century has proved troublesome to attain in practice.

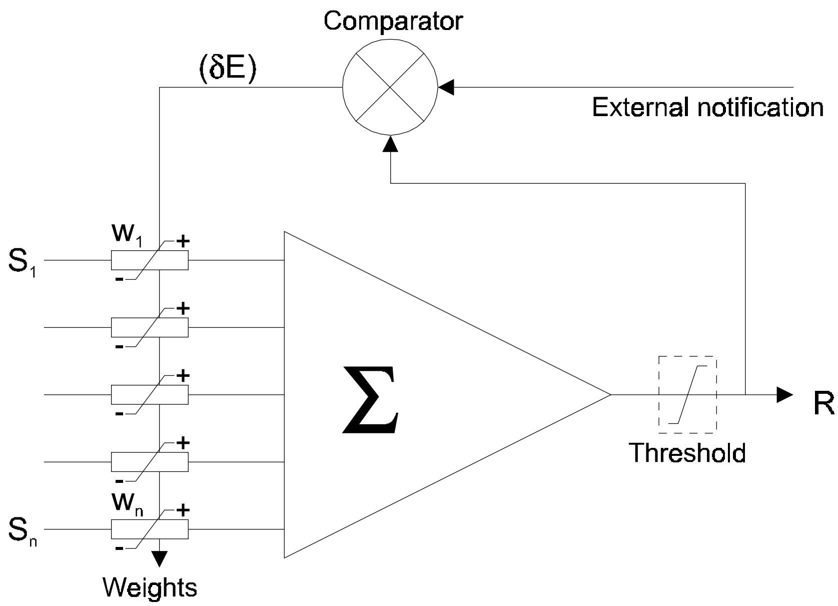


Figure 2-2: A Simple Neurone Model

An early implementation of the neural network approach as a simulation on a serial computer, the *Perceptron*, was provided by Rosenblatt (1962). Rosenblatt's Perceptron augmented the basic neurone model with an additional layer of *association units* that randomly connected each of the input points (S_1 .. S_n) to the sensory units via fixed positively (+1) or negatively (-1) weighted connections. Rosenblatt defined a procedure to update the weights when the output response of the unit differed from the desired one, as computed by an error comparator. The Perceptron learning procedure computed an adjustment to the set of weights implicated in an erroneous decision by an amount just sufficient to correct the decision. This method has subsequently been criticised for not stabilising if there is no set of weight values that correctly partitions the decision space. Several other procedures for learning by weight adjustment have been described (Nilsson, 1965; Hinton, 1990). More fundamental shortcomings of the connectionist approach were described by Minsky and Papert (1969), who argued that there were significant classes of recognition problems that this architecture could inherently not discriminate. Examples included the exclusive-OR function and various connected and disconnected figures. Research into Neural Networks went into decline for some years until revived by Geoffrey Hinton and others in the mid-1980's.

A neural network with multiple-layers of adjustably weighted "neurones" overcomes many of the criticisms levelled by Minsky and Papert, but introduces problems of how the various individual weights in the "hidden" layers might be adjusted. Figure 2-3 illustrates the architecture of a multi-layer artificial neural network. Rumelhart, Hinton and Williams (1986) describe the *backpropagation* algorithm, a method by which the effects of undesired classifications may be used to adjust weights distributed across many layers. The backpropagation algorithm is essentially a two-stage computation. In the first stage the activation of every unit in the network is calculated. In the second stage an error derivative (δE) is computed at the output layer and subsequently distributed to adjust the weights on intermediate hidden layers.

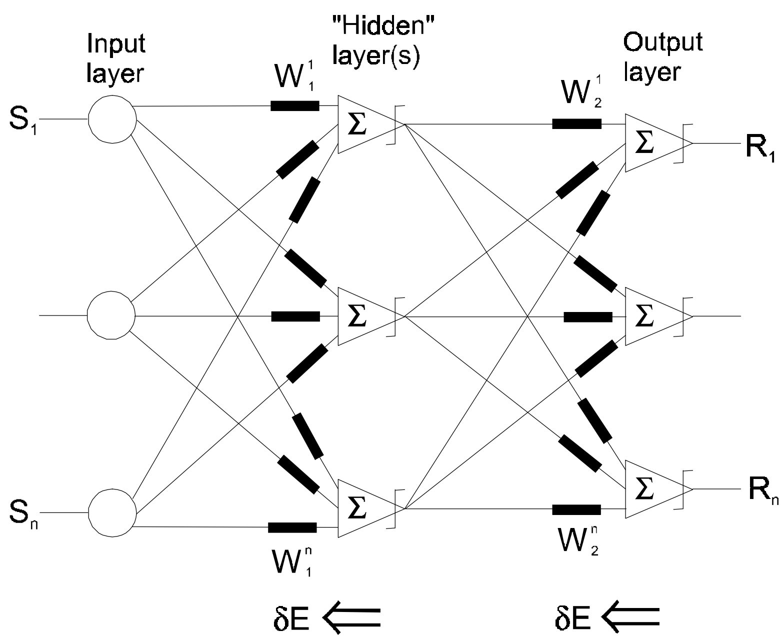


Figure 2-3: A Multilayer Neural Network Model

The backpropagation algorithm has been applied with some success to a range of tasks. Hinton (1986) describes a system for the discovery of "semantic features" in data and Sejnowski and Rosenberg (1987) a system for converting text into speech. Jochem, Pomerleau and Thorpe (1993) describe two systems *ALVINN* and *MANIAC*, multi-layer neural controllers for road following in a mobile vehicle. The ALVINN system comprised 960 input units (a 30 x 32 "retina"), 4 hidden units and 50 output units. The MANIAC system employed the same input and output arrangement but incorporated additional hidden units (a total of 16) in two layers, giving improved road following performance under a range of conditions. Pomerleau (1994) describes a neural network to control a walking robot. Chesters and Hayes (1994) describe experiments employing a connectionist model to investigate the effects of adding context memory signals to control a small mobile robot. Nehmzow and McGonigle (1994) describe their use of a supervised teaching procedure to train the Edinburgh *R2 robot* in a variety of wall following and obstacle avoidance tasks. Gaussier and Zrehen (1994) describe the use of *Khepera* mobile robots in research to investigate building a neural topological map.

Connectionism is evidently an S-R approach; a set of sensory data presented at the input units is translated into a set of output responses. It differs from the reinforcement approach in that an error signal is propagated to adjust many weights. In reinforcement learning a desired (or undesired) signal is typically used to adjust activity units specifically implicated in the behaviour choice. As a positive consequence of this, artificial neural networks are often considered to be robust in the face of a noisy or disrupted input data vector. Neural network models discussed thus far have all concentrated on supposed properties of collections of a simple and simplified neurone. Hinton (1990, p. 209) points out that the backpropagation algorithm is rather implausible as a biological model, as there is "*no evidence that synapses can be used in the reverse direction.*" Other writers have taken more care to link computer models of neural function to research findings in the areas of neuroanatomy and neurophysiology. Albus (1981), for instance, proposed a model based on the observed structure of the brain. Albus' Cerebellar Model Architecture Computer (*CMAC*) postulates a table driven look-up mechanism to map many sensory inputs to many motor outputs.

2.7. Operant Conditioning

The theories and models described so far are characterised by the stimulus-response (S-R) approach. An action is primarily selected on the basis of incoming sensory information. Once the strength value of a connection is computed, information about the circumstances leading to the reward or reinforcement on which the value is based is generally discarded. *B.F. (Burrhus Frederic) Skinner* (1904-1990) proposed a radically different mechanism, that of *instrumental* or *operant conditioning*. In the operant conditioning model responses are not "elicited" by sensory conditions, but "emitted" by the animal. Reinforcement is therefore between response and reward, not between sensory condition and reward. The action is described as the "operant" or "instrument" by which reward is obtained. Reward may only be forthcoming in some of the many situations in which the action can be taken. In this case it is referred to as a *discriminated operant*, the various circumstances being distinguished by sensory conditions.

Skinner and his followers adopted a purely behaviourist standpoint and have used their ideas to propose explanations for a wide range of human psychological concepts such as "*self, self-control, awareness, thinking, problem-solving, composing, will-power, … repression and rationalization*"[7] which might otherwise be addressed in a more nebulous "mentalistic" manner. Skinner did not reject respondent behaviour or classical conditioning as valid phenomena, just their central importance. Many largely retrospective and comprehensive reviews of Skinner's contribution are to be found, including Verplanck (1954), and Catania and Harnad (1988).

Skinner applied his ideas to a wide range of areas, such as education, behavioural and social control, and psychiatry. Of particular interest to the current work are the experimental techniques developed by Skinner to investigate operant conditioning. In an apparatus, now almost universally referred to as the *Skinner box*, certain learning phenomena in animals may be investigated under highly controlled and repeatable conditions. In a typical Skinner box apparatus the subject animal may operate a lever to obtain a reward, say a small food pellet. The equipment may be sound-proofed to exclude extraneous signals and different arrangements can be adopted to suit different species of subject animal.

Typically the subject will be prepared to operate the lever to obtain the reward before the start of an experiment. Once the subject is conditioned in this manner various regimes can be established to record effects such as stimulus differentiation, experimental extinction, the effects of adverse stimuli ("punishment schedules"), and the effects of different schedules of reinforcement. Progress of the

[7] Quoted from Bower and Hilgard (1981, p. 170)

learned response may be automatically recorded in a trace that shows the number (and/or strength) of the emitted response in relation to the frequency of reward. Figure 3-1 in the next chapter illustrates some results of this form and a number of the experimental designs used in chapter six are influenced by these procedures.

For all the experimental evidence accumulated and effort expended in attempting to apply their findings, Skinner and his followers did not place an over-emphasis on theorising about the mechanisms that might be involved. As a consequence, perhaps, few formal models of operant conditioning have been developed. One such model, the Associative Control Process (*ACP*) model (Baird and Klopf, 1993; Klopf, Morgan and Weaver, 1993) develops the two factor theorem of Mowrer (Mowrer, 1956). The ACP model reproduces a variety of animal learning results from both classical and operant conditioning. Schmajuk (1994) presents a two-part model incorporating both classical and operant conditioning modules emulating escape and avoidance learning behaviour.

2.8. Cognitive Models of Learning, Tolman and Expectancy Theory

The majority of models of learning discussed in this chapter so far - both natural and as computer models, follow the premise that observable behaviour, the "response" is primarily mediated by the appearance of stimuli. Learning is therefore reduced to strengthening or weakening the connection between possible stimulus sets paired to one of a number of available responses. Both the reinforcement and classifier system computer models described extend this concept to allow credit (or blame) associated with a reinforcement signal to be distributed to earlier events with the aim of optimising or maximising overall reward, as received reinforcement signal, which may be obtained. The associationism of classical conditioning is a clear exception, and operant conditioning also takes a distinct, alternative approach.

While forms of *stimulus-response (S-R) behaviourism* were highly influential for much of the first half of the twentieth century, it became clear that the predictions they made were inadequate to explain all of animal learning and much of human learning and behaviour. An alternative view, developed by Edward Chance Tolman (1886-1959) and others, was that behaviour was primarily mediated by the situation which was to be achieved, rather than the prevailing situation (as in S-R theory) or the action that would be taken (as postulated by operant conditioning studies). This was termed the *cognitive viewpoint*. Toates (1994) has pointed out that the term "cognitive" now encompasses a wide range of theories and approaches to Psychology and Artificial Intelligence. He notes that texts on "Cognitive Psychology" will often incorporate descriptions of the behaviourist standpoint with little comment as to the historical divisions once so strongly argued.

Tolman's keystone work "Purposive Behavior in Animals and Men" (Tolman, 1932) described a series of experimental observations and laid out the foundations of *expectancy theory*. Much of the experimental evidence presented was derived using rats in maze like experimental apparatus. It has been noted that while Tolman's theoretical position changed little over the years, his use of vocabulary to describe concepts and processes within the theory underwent a continuous series of changes and shifts. Tolman was a prolific author, with some 70 papers published during a distinguished career. Tolman's position is retrospectively described in an analysis by MacCorquodale and Meehl (1954) and again, in a more accessible form, by Bower and Hilgard (1981, Ch. 11).

One significant aspect of Tolman's theorising was to identify a number of situations that were, and continue to be, particularly difficult to satisfactorily explain in purely behaviourist-reinforcement terms. Bower and Hilgard (1981, pp. 330-342) review this evidence in some detail. Two particular phenomena, *latent learning* and *place learning*, illustrate these arguments. In latent learning Tolman argued that as reinforcement learning requires a reward at the conclusion of the behaviour sequence to establish its effectiveness, then, if learning could be demonstrated in the absence of reinforcement, behaviourist-reinforcement theories would be shown inadequate. Tolman convincingly demonstrated learning in rats in the absence of reinforcement. Consequently his expectancy theory, which can account for the phenomena, was supported.

Similarly stimulus-response theory maintains that every response is triggered by some stimulus. Tolman argued that if the experimental animal could be placed in circumstances where different responses were appropriate in apparently identical stimulus conditions then stimulus-response theories would be again demonstrated inadequate. Tolman and others subsequently successfully demonstrated that animal subjects can indeed make different responses under apparently identical sensory conditions. Such conditions include manipulation of the motivational state of the animal (hunger, thirst, etc.); or by introducing obstructions into a specific maze apparatus, forcing the response at different route choice points. Several variants on the place learning experiments are described by Bower and Hilgard. All represent significant challenges to the behaviourist viewpoint. Sections 6.6 and 6.7 in chapter six replicate classic experimental procedures for latent learning and place learning respectively.

2.9. MacCorquodale and Meehl's Expectancy Postulates

For all the challenges that Tolman and expectancy theory present to the behaviourists it was not without problems. Perhaps the most persistent criticism of the approach was that the model was purely descriptive. The lack of formalised and explicit theoretical constructs heavily constrained the predictive power and hence usefulness of early expectancy models. Recognising this MacCorquodale and Meehl (1953) proposed a set of 12 *expectancy postulates* in an attempt to provide a testable and quantifiable basis for expectancy theory. MacCorquodale and Meehl redefined Tolman's notion of a Sign-Gestalt Expectancy (henceforth *expectancy*) as a three part "basic cognitive unit" of the form:

$$S_1 \rightarrow R_1 \rightarrow S_2 \qquad \text{(basic expectancy)}$$

The addition of an "S_2" component over a stimulus-response model provides for a form of instrumental or operant *modus ponens*; an implication of an outcome condition (S_2) caused by the action R_1 rather than purely indicated as desirable by the presence of the condition S_1. This is largely equivalent in structure to the notion of the *three-term contingency* "stimulus - response - consequence", used by Catania (1988) to express the fully discriminated Skinnerian operant class of discriminated stimulus, response and contingent outcome of reward or punishment. With the essential difference that it is the <u>identity</u> of the outcome that is recorded in expectancy theory, not just a measure of its <u>desirability</u> or quality as is recorded in the operant, or reinforcement learning approaches.

MacCorquodale and Meehl's twelve expectancy postulates refer to eight underlying processes, namely "mnemonization", "extinction", "generalization", "inference", "need", "cathexis", "valence" and "activation". Postulate 1, the *mnemonization* process, refers to an increment in "strength" of the expectancy where the component parts S_1, R_1 and S_2 are in close and ordered temporal contiguity. This increment is described by a negatively accelerating function, where the function acceleration rate is determined by the valence (*q.v.*, a measure of usefulness or desirability) of the S_2 component and the asymptote of the strength determined by the relative frequency or probability that S_2 follows the sequence $S_1 \rightarrow R_1$. Postulate 2, the *extinction* process, refers to a decrement in strength where the sequence $S_1 \rightarrow R_1$ is not terminated by the *expectandum*[8] S_2. It will be argued later that the relative frequency of contiguity, the function rate and the valence level are better considered as separate and distinct values and should not be convolved into a single "strength" parameter. MacCorquodale and Meehl did not propose an explicit or quantifiable mathematical formulation for either of these postulates.

Postulate 3, *primary generalization*, allows for sharing of expectancy strengths where two expectancies share R_1 and S_2 components and their S_1 components "resemble" one another. Postulates 4 and 5, *inference* and *generalized inference*, refer to processes by which temporal contiguity ($S_2S^\star$) between a known expectandum S_2 and another sign stimulus $S^\star$ increases or decreases the strength of the expectancies sharing elements, or in which elements are "similar", according to the degree of temporal adjacency and frequency of occurrence. A different approach to the evaluation of expectancies will be proposed later, which considerably diminishes the importance placed on these postulated mechanisms of generalisation and inference. As before MacCorquodale and Meehl did not proffer any suggestions as to the nature of "similarity" or "resemblance", or how they may be evaluated, between components in these shared expectancies.

[8] From the gerundive form "... to be expected"

Cathexis[9], postulate 11, refers to the strength of connection between a stimulus sign $S^\star$ and a drive, motivation or goal state. *Need strength*, postulate 10, describes the degree to which the subject is to be influenced by the cathectic situation. The *valence*, postulate 9, of a sign $S^\star$ is then defined by the product of the need (D) and cathexis ($C^\star$) attached to that sign ($D \times C^\star$). It is perhaps interesting to note, with hindsight, the pivotal role of innate mechanisms to control and balance motivation and behaviour (such as those being described by Tinbergen at about the same time) appears to have been largely unrecognised. MacCorquodale and Meehl were therefore unable to propose effective mechanisms for these postulated processes.

Secondary cathexis, postulate 6, allows for the induction of cathexis to an expectandum S_2, where a contiguity $S_2S^\star$ exists and $S^\star$ has valence. *Induced elicitor-cathexis,* postulate 7, allows cathexis to be induced to an S_1 component of an expectancy where its expectandum has already acquired valence, to an extent proportional to that acquired valence and the prevailing mnemonization strength of the expectancy. Tolman's (1932, p. 176) descriptions clearly indicate the notion of a *means-end-field* (later *cognitive map*, Tolman, 1948) by chaining expectancies in this manner[10]. Postulate 8, *confirmed elicitor-cathexis*, provides for additional strengthening of the expectancy where the sequence it describes is confirmed, and S_2 has valence.

Finally, in a process of *activation*, postulate 12, the action R_1 is evoked according to a *reaction potential* determined by a multiplicative function of expectancy strength and valence, when in the presence of the elicitor S_1. MacCorquodale and Meehl recognised that their postulate system for an expectancy theory was "*incomplete, tentative and certainly nonsufficient,*" but were able to present some hand-worked examples to illustrate their model.

---

[9](OED) Cathexis: n (Psych.) Concentration of mental energy in one channel, [f. Gk *kathexis* retention]

[10] The term "cognitive map" has more recently tended to be interpreted more literally, internal "maps" of spatial locations or terrain layout (Meyer and Guillot, 1991, for a compact review).

2.10. Computational Models of Low-level Cognitive Theories

Further development of expectancy theory, as with other psychological models, has depended on the use of computer based formalisations. Information processing models of cognitive processes impact theoretical development in several ways. Firstly, the model must be complete to the extent that an algorithmic process can be adequately defined for each essential element or component in the model. Secondly, each of these essential elements must be sufficiently defined to permit the creation of program code. Thirdly, they are testable and may be subjected to experimental regimes to determine their performance under controlled and repeatable conditions. In some instances their performance may subsequently be compared with results obtained by experiment with, and observation of, natural systems.

Three such models are presented in the next sections of this chapter, leading to the development of a novel Dynamic Expectancy Model. None of these models make direct reference to Tolman or expectancy theory, being described as "sensory-motor" or "intermediate-level" cognitive models, but the debt owed is nevertheless clear to see. Each model adopts a *schema representation*[11]. The three models are "JCM", described by Joseph Becker (Becker, 1970, 1973); "ALP", described by David Mott (Mott, 1981; Bond and Mott, 1981); and a model of the early stages of Piagetian development described by Gary Drescher (Drescher, 1987, 1991). Both Becker and Drescher elected to discuss or demonstrate their work using simulated environments, while Mott was able to demonstrate simple learning tasks utilising a real mobile robot.

2.11. Becker's JCM Model

Becker's *JCM* model of intermediate level sensory-motor cognitive behaviour adopted a "stimulus - action - stimulus" representation. Figure 2-4 illustrates the structure of the "schema", the primary form of information storage in the model. Many schemata are recorded by the system in a *Long Term Memory* (LTM). Sensory and input information enters the system via an "input register" into a limited capacity *Short Term Memory* (STM). STM acts as a *FIFO buffer*, and will

[11]Plural "schemata", "schema" or "schemas", following the preference of the original authors.

contain a small number, say six or so, items. As new items enter STM via the input register older items are lost, or they may be recycled. Individual elements of information, as entered into STM and recorded within schemata, are referred to as *kernels*. In Becker's representation each kernel takes the form of a predicate with arguments, for instance:

<colorchange right bottom black red>

The predicate in this case refers to a sensory effect (a colour change from black to red) in one of the sensory locations (right bottom cell in a simple 3 by 3 cell "eye" viewing a greatly simplified simulated blocksworld environment). Kernels may be defined as static sensory, indicating an absolute sensory value, differential sensory, indicating a change of sensor value, a motor or efferent command, or a request to interrogate a sensor.

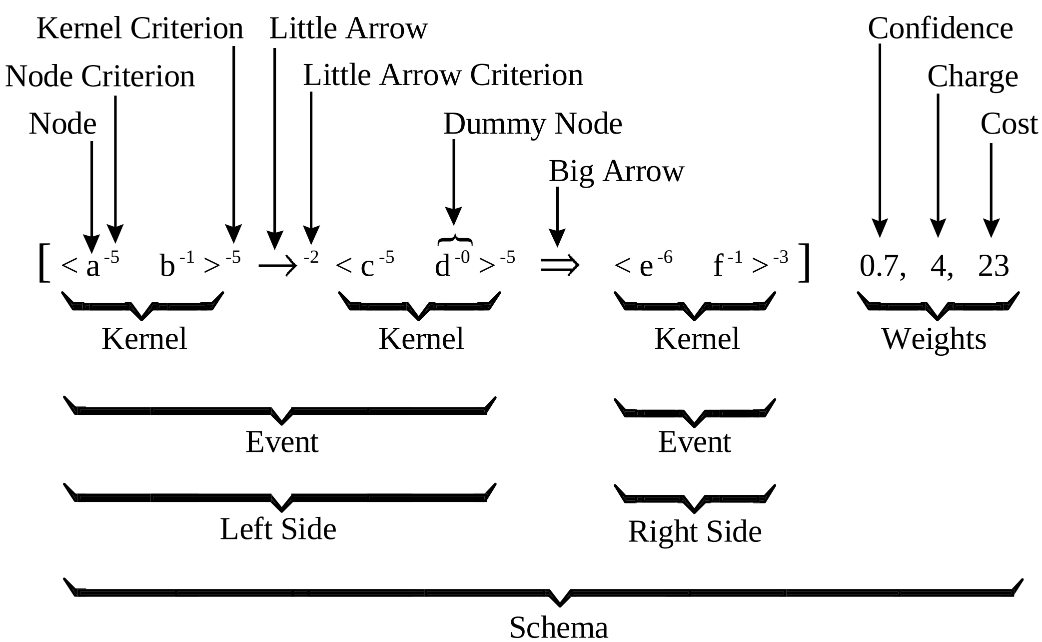


Figure 2-4: A JCM Schema

from Becker (1973), p. 410

Once created and retained in LTM individual schema left hand sides are matched to the current contents of the STM. Schemata with a high degree of match posit that the events defined on their right hand side will appear in STM at some point in the future. Schemata have a predictive role. The overall *schema confidence weight* is adjusted according to the validity of this prediction. Each kernel in a schema and each predicate and argument in each kernel has associated with it a criterion value. Criterion values indicate the relevance or importance of the component part to which they are attached.

Individual kernels are ordered, with the ordering indicated by the little arrow construct ("→"). The "little arrow criterion" records how significant the ordering indicated by the little arrow is to the overall success of schema application. The big arrow ("⇒") construct delimits the matching event to the predicted event. Becker describes *analogic-matching*, a complex algorithm by which individual criterion weights are adjusted according to the effectiveness of the schema in making successful predictions. The "charge" weight associated with each schema indicates the desirability of the right hand side as a system goal. The greater the charge value the greater the desirability of obtaining kernels into STM that will allow the complete matching of the schema. Kernels in partially matched schema may be established as sub-goals in Becker's method. Note that the cost weight associated with the schema refers primarily to the "cognitive cost", the computational effort required to make the match between LTM and STM, rather than a cost of performing the action embedded in the schema.

JCM was never implemented, partially, it might be suspected, as a result of the complexity inherent in the analogic-matching process and the consequential difficulties in devising stable algorithms to manage all the different criterion and schema weights. Nevertheless Becker's JCM design introduced a number of processes that were to be adopted later, notably in Mott's ALP system. Primary amongst these is the idea of schema creation by the process of *STM to LTM encoding*. A pattern of kernels being extracted from the input STM and reformulated as a LTM schema, which may in turn be verified by a predictive matching process.

Becker also promoted the idea of schema refinement through the processes of *differentiation* and *specialization*. In differentiation kernels are removed because their accumulated criterion values indicate they are irrelevant to the effect of the schema (as indicated by a small or zero criterion value). Negative criterion values indicate that the absence of the kernel is essential for the effective matching of the schema. Specialization is invoked to refine schemata where an intermediate confidence weight indicates an incomplete specification of the conditions for its application defined by the left hand side kernels. Specialization is achieved in JCM by the addition of further kernels on the left hand side of the schema.

2.12. Mott's ALP Model

Mott's *ALP* system considerably refined and implemented an intermediate level sensory-motor cognitive model and applied the result to developing behaviours in a small mobile robot. Mott retained the central representation of schema recorded in a long term memory, with a limited capacity STM. STM retains the input register, but each time slot may contain multiple kernels for matching into LTM schema. This modification overcame a dependence on a complex sensory attention mechanism to identify and select items for entry to STM. Critically, Mott reduced the complexity of the kernel, dispensing with the predicate and argument form. In ALP kernels are either derived directly from a sensor condition, the sensory kernel, or they represent an efferent action, the motor kernel. The little arrow notation, retained from JCM, now represented the passing of exactly one execution cycle, thereby reducing the "analogic-matching" process to manageable proportions. Mott overcame the problem of goal motivation inherent in JCM by introducing two new (sensory) *motivational kernels*, <HIGH>S and <LOW>S, respectively representing a condition that the robot should seek and a condition it should avoid. At a low level some conditions, such as "battery very low", are associated with motivational kernel (in this case <LOW>S).

ALP retained Becker's JCM mechanisms for creating new schema by STM to LTM encoding, triggered by the appearance in STM of *novel* kernels. Schema validation, differentiation and specialisation remain substantially as in JCM. Goal management is however substantially different. ALP is able to use schema to form chains of predictions about possible future events. When either a <LOW>S or <HIGH>S kernel is predicted this is treated as a goal definition, and a goal tree can be formulated to either avoid the undesirable predicted event, or to attain desirable ones. Paradoxically the system would not react to the direct appearance of a motivational kernel, only its predicted occurrence. Schema may be chained to form a goal solution, and actions selected to control the robot.

ALP was implemented in the *POP-2* programming language and ran on an ICL 1900 series mainframe in an interactive mode. ALP was heavily processor bound. The robot used was controlled by a local PDP-11 mini-computer, which packaged sensory information from the robot for onward transmission to the mainframe and interpreted commands sent from the mainframe. ALP was an essentially *ad-hoc* system that demonstrated the acquisition of some simple robot behaviours by the learning process. Its effectiveness as a behavioural system was severely restricted by the rapid loss of schema confidence in future events in the predictive chains and goal trees. Chains were limited to six goal cycles or three predictions. These restrictions in part arose due to the method of computing these possible outcomes, and in part to the uncertainty inherent in the experimental environment provided by the robot test-bed.

2.13. Drescher's Model

Drescher's model further simplified the notion of a schema. The context of a schema being reduced to a simple conjunction of sensory *primary items* (Drescher's term for a kernel), or their negation. All timing information was abandoned. Figure 2-5 illustrates the form of the schema. Drescher used a simplified simulated hand-eye co-ordination environment, similar in concept to that proposed by Becker, but with a larger number of states that may be visited. None of the tasks investigated required information about prior states and this limited form of context definition was adequate for the environment chosen. In these circumstances a Short Term Memory is redundant and was not used in the model.

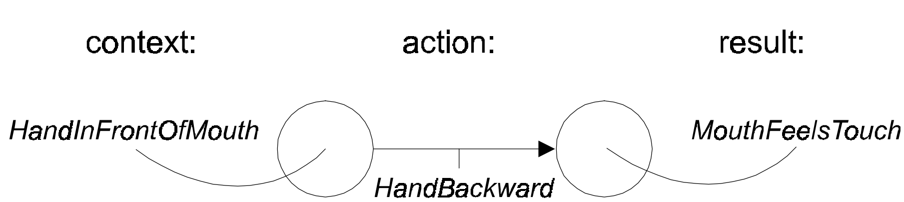


Figure 2-5: A Schema in Drescher's Cognitive Model

from Drescher (1991), p. 9

Drescher describes the *composite action*, chains of individual schema defined with respect to some goal state forming what is essentially a sub-routine that might substitute as the “action” of a single schema. Figure 2-6 illustrates the form of the composite action. Drescher also describes a process by which individual schema are considered as *synthetic items*, the whole schema being used as a record of a recent event in an attempt to simulate *Piaget’s* notion of *object permanence*.

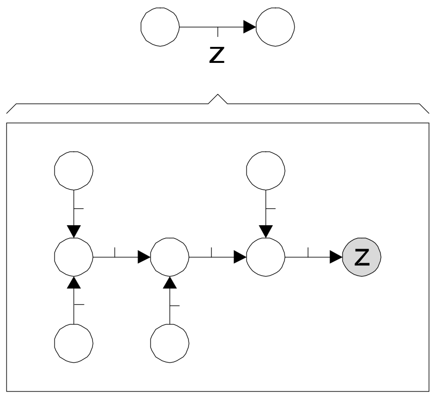


Figure 2-6: A Composite Action

from Drescher (1991), p. 91

Drescher employed a radically different approach to the generation of schema from the STM to LTM encoding used by JCM or ALP, the *marginal attribution* process. Figure 2-7 illustrates the stages in creating schemas of arbitrary complexity by this process. In step one “bare schema” are created, one for each of the primitive actions available to the system (notated by Drescher as “ /a/ ”). Bare schema have empty context and result slots. The system is then run for a period with actions being selected at random, a *trial and error* period. Exploration of a new environment by a naïve system is a feature of the JCM and ALP systems also. During this period of exploration each schema has associated with it an additional structure, the *extended result*, which accumulates outcomes applicable to the new schema.

At some point, after sufficient exploration has been completed, a set of new schemas are “spun off”. This is shown as step two. In this example the new schema “ \a\x” is created from the extended result. Many new schemas could be formed at this point. As each new schema has no context information it is considered to be “unreliable” and it is given an *extended context* structure, step three. This structure accumulates a record of items active as the new schema is used in a manner similar to the extended result. Following another suitable period of activity, items are selected from the extended context for inclusion into the new schema’s context, “p/a/x” in the example. This process may be repeated as often as required to further refine the context of the prototype schemas, step four.

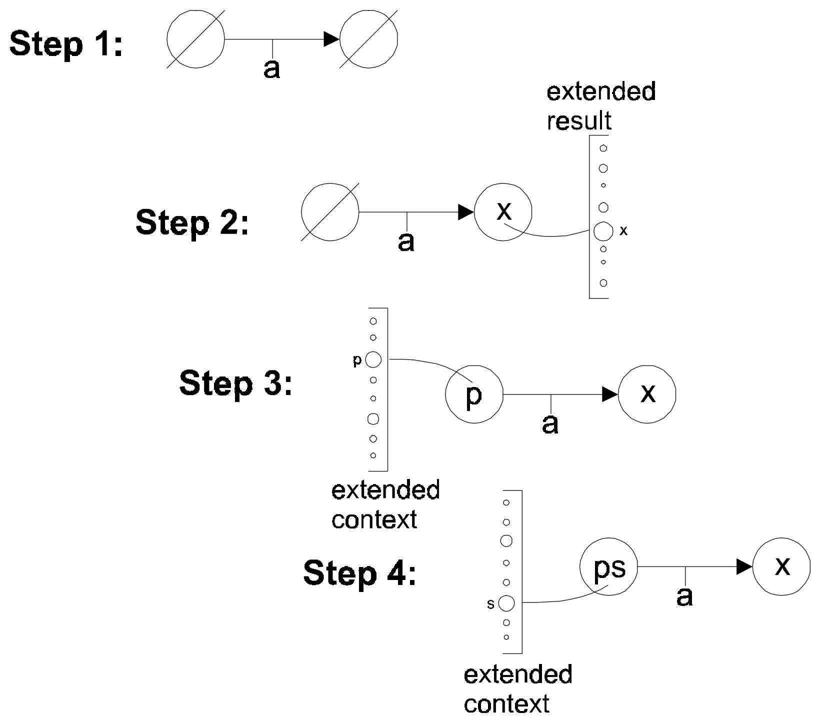


Figure 2-7: The Marginal Attribution Process

Prepared from a description in Drescher (1991)

This marginal attribution method for schema learning is inordinately inefficient, as evidenced by the extensive computational resources required to execute the procedure in the simulated environment described (Drescher, 1991, p. 141). Furthermore, Drescher provides little clue as to its effectiveness, beyond indicating the need to incorporate additional mechanisms to limit the creation of redundant schemas, the *redundant attribution* process.

2.14. Other Related Work

Jones (1971) describes a computer model of new-born infant suckling behaviour. Riolo (1991) presents a three term model (*CFSC2*) based on classifier systems concepts. An additional form of the classifier rule (the "e#/t#" rule type) allowed the system to describe transitions between either actual or hypothetical states. The system might therefore determine expected reward on the basis of look-ahead cycles. The CFSC2 model was used to demonstrate the *latent learning* phenomena. Bonarini (1994) describes a three part operator exploiting fuzzy logic. Shen (1993, 1994) describes the *LIVE system* that creates, utilises and refines new GPS style operators from successful and failed prediction sequences while performing problem solving tasks in its environment. LIVE models its environment using a set of prediction rules, triples in the form <condition action prediction>. Shen's system employs a number of heuristics in the creation of new prediction rules, and subsequently may revise them (through a process of "Complementary Discrimination Learning"). Prediction failures trigger the system to search for differences between the current failed instance, and stored instances of successful predictions using the same rule. The rule revision algorithm is noise intolerant, but has been demonstrated on a number of recognised tasks, including the *Towers of Hanoi*.

Chapter 3

3. A New Dynamic Expectancy Model

This chapter seeks to define and develop a new *Dynamic Expectancy Model*. This Dynamic Expectancy Model extends MacCorquodale and Meehl's original expectancy theory formulation to provide a workable and so testable implementation. It may be seen as part of the current trend to identifying existing "thought experiments" from the literature, reconstructing them as computer simulations and so re-evaluating and reviewing their premises and predictions by experiment and analysis in a manner that was previously impossible. The Dynamic Expectancy Model builds on the intermediate level cognitive models described by Becker (1973), Mott (1981) and Drescher (1991). It also draws on mechanisms and processes from a range of other sources, notably the accumulated work on innate behaviours and capabilities (Tinbergen, 1951; Brooks, 1986; and Maes, 1991, among others) and the notion of a *policy map* drawn from reinforcement learning methods (Sutton, 1990; Watkins, 1989).

The Dynamic Expectancy Model eschews mechanisms exclusively detected in human infant or adult subjects, but serves rather to address issues arising from work relating to the understanding and modelling of animal behaviour. In particular this new model identifies and addresses some of the limitations and shortcomings of behaviourist theories relating to learning and behaviour in lower animals, which were considered in previous chapters. The new model focuses on the idea that all animals (of whatever level of complexity) are essentially autonomous individuals, which may behave, learn and reason within the capabilities ultimately determined by their innate definition, the *ethogram*. This individuality does not imply that those individuals exist independently of other members of the same or other species. Many are dependent on parental care, naturally exist and co-operate in packs or communities composed of distinct individuals, exist in symbiotic or antagonistic relationships, or must attract a mate to reproduce.

The intermediate-level cognitive models of Becker, Mott and Drescher seek to emulate the developmental process of the human infant. Each was influenced to varying extents by the work of the Swiss child developmental psychologist *Jean Piaget* (1896-1980). Drescher (1991, Ch. 2) provides a description of the first six stages of infant development according to Piaget's observations. One fundamental problem with this approach is the rapidity with which normal human infant development proceeds. These intermediate-level cognitive models lack the power to account for the considerable increases in the child's performance and ability. Moreover, there is still little agreement as to whether some, most, or all of this observable improvement is primarily due to a learning or to a maturation process in which innate abilities are activated in an essentially constant order. These models may therefore be taken as simplifications of other cognitive-organisational theories of learning (Bower and Hilgard, 1981, Ch. 13) which are obliged to postulate a wide range of mechanisms to account for the diversity of human adult abilities. Tolman and expectancy theory takes a constructivist view, adopting mechanisms required to model and explain behaviour and ability of non-human animals, though he later attempted to expand the model to encompass many aspects of human behaviour.

3.1. The Animat as Discovery Engine - The Thesis

In the Dynamic Expectancy Model animats may be viewed as machines for devising hypotheses, conducting experiments and subsequently using the knowledge they have gained to perform useful behaviours. In this learning model the animat implements a low level version of a "scientific discovery process." A critical feature is the creation and verification of self-testing experiments, derived from simple hypotheses created directly from observations in the environment. Each hypothesis describes and encapsulates a simple experiment. Each experiment takes the form of an expectancy or prediction that is either fulfilled, so corroborating the effectiveness of the hypothesis, or is not fulfilled. From time to time goals, activities required of the animat, will arise. By constructing a graph like structure from the hypotheses it has discovered during its lifespan and then determining an intersection of this graph with current circumstances, the animat may determine appropriate actions to satisfy those goals. Part of the innate structure of the animat provides the rules by which this discovery process proceeds. Part imbues the animat with sufficient behaviour to set goals and to initiate and continue all these activities until learned behaviour may take over from the innate. Above all the animat must survive long enough to create hypotheses and conduct experiments.

Where Popper (1959, and see section 3.2.5 later in this chapter) describes a *Hypothetico-Deductive* approach, the Dynamic Expectancy Model adopts a *Hypothetico-Corroborative* stance. No mechanism for the construction of more complex models is incorporated into the Dynamic Expectancy Model. In order to distinguish hypotheses in the Dynamic Expectancy Model from those proposed by Popper, they will be referred to as *μ-hypotheses* ("micro-hypotheses"), similarly experiments as *μ-experiments* ("micro-experiments"). The construction and verification of low-level observation based μ-hypotheses would appear a useful pre-cursor to the independent development of any systematic theoretical model, whose structure is not wholly or primarily dependent on an *originator*[12].

3.2. The Expectancy Unit as Hypothesis

In the Dynamic Expectancy Model the expectancy, and so the basic unit of learning, takes the form of the predictive μ-hypothesis. This has critical implications. First and foremost of these implications is that each expectancy unit now contains the means to perform a self-contained test and so confirm or deny its own validity. In turn this implies the learning process is no longer dependant on external or reward signals to guide the process. Behaviour to seek goals is made independent of learning activity required to accumulate the knowledge, which may in turn be applied in performing goal directed behaviour. This section describes and discusses a number of "postulates" that define the operation of the expectancy unit as predictive hypothesis.

[12] Originator, the individual or process responsible for the creation of the animat and its ethogram.

3.2.1. The Hypothesis Postulates

Definition H0: The **μ-hypothesis**. Each μ-hypothesis records an assumed transition between two detectable sensory patterns (signs "s1" and "s2", q.v.) indicated or caused by an action ("r1") available to the animat system.

Postulate H1: **Prediction**. Prediction forms the basis of self-testability. Each μ-hypothesis encapsulates an expectation that predicts the occurrence (or appearance) of the consequent sign ("s2") at a specific time following the appearance (or occurrence) of the context sign ("s1") and the action ("r1").

Postulate H2: **μ-Experimentation**. μ-Experimentation is the mechanism by which predictive self-testability is achieved. Every μ-hypothesis is tested at every opportunity. A separate prediction relating to the consequent sign "s2" is created each and every instance where the context sign "s1" and response "r1" occur in the relationship defined in that μ-hypothesis. Each such prediction is termed a μ-experiment. The conduct of μ-experiments is insensitive as to why the triggering conditions "s1" and "r1" arose.

Postulate H3: **Corroboration**. Corroboration is one method by which the predictive ability of a μ-hypothesis is recorded. The quality of a μ-hypothesis is determined solely by its ability to accurately predict its consequent sign. The corroboration measure is defined as the ratio of the total number of predictions made by the μ-hypothesis to the number of correct predictions made, as verified *post-priori*. Any μ-hypothesis that has always given rise to a verified prediction will have a corroboration measure of 1.0. Any other μ-hypothesis will have a confidence or "corroboration" measure (Ch) of zero or greater, but less than one. Ch therefore reflects the probability of a valid prediction, thus:

$$Ch = p(s2 \,|^{t}\, s1+r1) \qquad \text{(eqn. 3-1)}$$

The use of the "*t*" symbol acts as a reminder of the temporal relationship that exists between the expectandum "s2" and the context. As this expression gives no indication of sample size, the corroboration measure is not in itself an indication of the usefulness, rarity or reliability of the prediction.

Postulate H4: **Reinforcement**. Reinforcement is a second method by which the predictive ability of a μ-hypothesis is recorded. In this context "reinforcement" substitutes for MacCorquodale and Meehl's use of the term *mnemonization*. In a measure related to corroboration, each successful verified prediction reinforces confidence in a μ-hypothesis. Conversely every unsuccessful prediction extinguishes confidence in that μ-hypothesis. The effect of each verification is discounted as further predictions are made. The *reinforcement measure* (Rh) is changed by the quantity:

$$\Delta Rh^{p+1} = \alpha(1 - Rh^{p}) \qquad \text{(eqn. 3-2)}$$

following each instance of a successful prediction (^{p}), and

$$\Delta Rh^{p+1} = - \beta(Rh^{p}) \qquad \text{(eqn. 3-3)}$$

following each unsuccessful prediction. Under constant conditions these relationships give rise to the widely observed "negatively accelerating" form of the learning curve. The two proper fractions the *reinforcement rate* (α) and the *extinction rate* (β) respectively define a "learning rate" for successful and unsuccessful prediction situations. They control the rate at which the influence of past predictions will be discounted. These parameters shall be normalised such that the Rh value of a μ-hypothesis that makes persistently successful predictions tends to 1.0, the Rh value of a μ-hypothesis that persistently makes unsuccessful predictions tends to 0.0. The positive reinforcement rate need not be equal to the negative extinction rate.

Mnemonization for expectancies in the MacCorquodale and Meehl postulates are fundamentally based on the notion of temporal adjacency and contiguity. This was inherited from decades of experimental observation that has repeatedly noted that learning phenomena are invariably stronger for events that are closely related in the temporal domain. This is entirely consistent with the provisions of the Dynamic Expectancy Model. Temporally adjacent predictions are tested first. The time-scale being extended only in circumstances where unsatisfactory predictive performance is determined over the shorter period.

Postulate H5: **Creation**. Creation is the method by which the animat extends the set of μ-hypotheses. μ-Hypotheses exist to predict future occurrences of signs; it is therefore reasonable to suppose that new μ-hypotheses might be created under two specific circumstances. First, every sign shall have at least one μ-hypothesis capable of predicting it. Novel signs (ones not previously recognised by the system) shall trigger a rule creation process, postulate H5-1, *novel event*. The consequence ("s2") for this new μ-hypothesis will be the novel sign. The context and action drawn from the set of recent signs and actions recorded by the system. By a process of *timebase shifting* the current, novel, sign will be shifted to be a future prediction, with a corresponding shift in the relative time relationship to the other components selected for the new μ-hypothesis.

In the second creation circumstance, known signs are detected without a corresponding prediction, postulate H5-2, *unexpected event*. A new μ-hypothesis may be created, using the same mechanism as for novel signs to cover the unexpected event. Shen (1994) and Riolo (1991) both describe broadly similar strategies for "rule" creation triggered by "surprise" events. Kamin (1969) has investigated the role of predictability and surprise in various classical conditioning procedures using rats.

Postulate H6: **Differentiation**. Differentiation is the mechanism by which the animat may refine its existing set of μ-hypotheses. Differentiation adds extra conditions to the context of an existing μ-hypothesis, reducing the range of circumstances under which that μ-hypothesis will be applicable. Differentiation may be appropriate to enhance μ-hypotheses that have stabilised, or stagnated, at some intermediate corroborative measure value. μ-Hypotheses should not be subject to differentiation until they have reached an appropriate level of testing (their "maturity"). Maturity is a measure of the degree of corroboration of a μ-hypothesis. It is otherwise independent of the age of a μ-hypothesis. It is expected that the differentiation process will create new, separate μ-hypotheses that are derived from the existing ones. Both old and new μ-hypotheses are retained and may then "compete" to determine which offers the best predictive ability.

Postulate H7: **Forgetting**. Forgetting is the mechanism by which the animat may discard μ-hypotheses found ineffective from the set of μ-hypotheses held. A μ-hypothesis might be deleted when it can be determined that it makes no significant contribution to the abilities of the animat. This point can be difficult to ascertain. Evidence from animal learning studies indicates that learned behaviours may be retained even after considerable periods of extinction. Experimental evidence from the implementation of the model described later will point to the value of not prematurely deleting μ-hypothesis, even though their corroborative measures fall to very low levels. Where a sign is predicted by many μ-hypotheses there may be good cause to remove the least effective. It is presumed that the last remaining μ-hypothesis relating to a specific consequent sign will not be removed; on the basis that some predictive ability, however poor, is better than none at all. Even if it was to be removed, a new μ-hypothesis would be created (by H5-2, unexpected event) on the first re-appearance of the consequent sign of the deleted μ-hypothesis. As no record is retained of the forgotten μ-hypothesis, any new μ-hypothesis created may be the same as one previously removed.

3.2.2. Initial Conditions for the μ-Hypothesis Set

The ethogram may be programmed to contain pre-determined μ-hypotheses, which will be used, corroborated, differentiated and forgotten as any other μ-hypothesis available to the animat. Equally the set of μ-hypotheses available to the animat may be empty at the time of *parturition*[13], the set being populated and maintained by actions defined by the various postulates described.

3.2.3. Concluding Conditions for the μ-Hypothesis Set

The animat is assumed to have a limited lifespan, but only by analogy with natural animals; there is no explicitly defined concluding or *terminating condition* defined in the Dynamic Expectancy Model. Learning by μ-hypothesis creation may slow and finally cease in the event that no new signs are encountered by the system, and when the existing signs are adequate to predict every appearance of each sign. These conditions may be encountered in the special environment defined by the

[13] Parturition, the moment the animat becomes a free-standing individual, dependent on the definition contained within the ethogram; analogous, perhaps, to the birth of an animal.

finite deterministic Markov state space environment (*FDMSSE*). Under these specific conditions, once every state has been visited at least once, then there will be no further μ-hypothesis creation on the basis of novelty (H5-1). Once every transition has been attempted in each state no new rules will be created on the basis of unpredicted appearance (H5-2). At this point there is a μ-hypothesis to accurately predict the next state, so that the conditions required to invoke μ-hypothesis differentiation (H6) and forgetting (H7) do not arise. Corroboration (H3/H4) does not cease under these conditions, neither does the option to recommence μ-hypothesis creation, differentiation or forgetting should the underlying structure of the environment change for any reason. It has been assumed that the animat has, inherent in its ethogram, some strategy that will eventually allow it to visit all states by all transition options. This may be by selecting actions at random.

A similar argument may be advanced in the case of the finite stochastic Markov state space environment (*FSMSSE*). As in the FDMSSE situation, learning by creation (H5-1) will cease once each state has been visited. Once each transition has been made, including all those derived from the additional probabilistic nature of the environment, creation by unpredicted event (H5-2) will cease. After an extended period of exploration in the environment the corroborative measure (H3) of each μ-hypothesis will tend to the true probability of the associated transition, although this will only ever be an estimate of the true probability. As before, should the structure of the state space change (new states or new transitions) new μ-hypotheses will be created to accommodate those changes.

Should the relative distribution of transition probabilities change, both the corroborative (H3) and reinforcement (H4) measures will change to reflect this as further exploration takes place. The corroborative measure reflects the overall “lifespan” situation. Under these circumstances the reinforcement measure has the potential to provide a better working estimate. Due to the probabilistic nature of the transitions none of the μ-hypotheses will achieve full corroboration. When the initial set of μ-hypotheses reaches the required level of maturity the differentiation process (H6) will become activated. New μ-hypotheses formed are subsequently tested in competition with their prototypes. Under the FSMSSE model conditions new context signs will be created by concatenation of additional states drawn from recorded past states (only one state is indicated at the current time). Given that the definition of the FSMSSE restricts the information bearing content for the choice to the current state, it may be taken that all such μ-hypotheses created by differentiation will, in the limit, be less effective than their parent prototypes. It is therefore an unfortunate consequence of the basic assumptions of the FSMSSE that differentiation will continue throughout the animat's lifecycle, without materially improving its behavioural performance. On the other hand its effect will not be catastrophic, the majority of the behaviour being mediated by the better corroborated initial set of μ-hypotheses.

Note that neither in the postulates, nor in either of these discussion cases (FDMSSE and FSMSSE) has any reference been made to the provision of an external source of reinforcement.

In general, the Markov state space environment may be considered a poor model of the natural environment. The fundamental assumption that the information required to select the best action to take is, or can be, described by the current sensory pattern remains, at best, contentious. Equally the idea that some combination of sensations will completely and uniquely describe a "state" that is constant over time and so may be returned to on numerous occasions fails to reflect our notion or experience of the natural world. Nevertheless, the FDMSSE and FSMSSE environments represent a well defined and extensively studied formalisation. They represent a convenient, repeatable and controlled test environment in which to conduct experiments to determine the properties and performance of a learning system. As these environments have been utilised by other authors, the Markov description represents a point of comparison between alternative theories of learning. Later sections in this work will return to the utility of the Markov environment as a test environment, and to comparisons with other research that has used these environments.

3.2.4. Hypothesis Based Models of Learning

An early suggestion that rats exploring maze test environments use a form of hypothesis was proposed by Krechevsky (1933). The term was later adopted briefly by Tolman (1938) as a description of his basic expectancy unit, although in his later writings the term “field-expectancy” is preferred. Restle (1962) provides a mathematical formalisation in which “hypotheses” (assumed or untested patterns of responses to cue stimuli) are sampled from a fixed size population by different means. In Restle’s model, hypotheses were either always correct (“C”), always wrong (“W”), or inconclusive (“I”), sometimes wrong, sometimes correct. Restle further proposed three selection strategies. Strategy (1) in which one hypothesis was selected and tested, then another, and so on (the *single-hypothesis assumption*). In strategy (2) all available hypotheses are selected for testing. In strategy (3) samples from the total population of available strategies are selected for testing (the *sub-set sampling assumption*). Restle was able to demonstrate that (under defined conditions) these three strategies are essentially equivalent - the “indifference to sample size” theorem.

Levine (1970) conducted a series of experiments with human subjects, designed to identify which strategy was used by the subjects. Subjects were asked to sort cards according to four easily discriminated elements (size, form, brightness and position). On some trials the subjects were given an indication, “right” or “wrong”, about their choice so that they may form one or more “hypotheses” about their selection choice (which may guide their future decisions). Interspersed with these indicated trials the subjects made unguided choices. Such *blind-trials* allow the experimenter to infer the hypotheses in use by the subject. These studies concluded that subjects repeated a hypothesis indicated as correct, and discarded a hypothesis indicated as incorrect. More significantly, many of the subjects appeared to be sampling several hypotheses at each stage, the sub-set sampling assumption, as indicated by the number of trials prior to perfect performance. In a related set of experiments the latency time for the choice was measured over successive trials. These experiments demonstrated a fall in decision time as possible, but ineffective, hypotheses were discarded. Decision latency time remained constant following the “solution trial”. More recent studies (Klahr, 1994) indicate that the hypothesis generation strategy used by human subjects is dependent on age and educational level. These results may call into question the appropriateness of applying data derived from human subjects directly to autonomous learning in animals or animats.

The emphasis of Kruchevsky's work was that rats explored their environments in a methodical, rather than random, trial-and-error, way. The basic assumption driving both Restle's and Levine's research was that hypotheses are selected and retained or rejected from a finite, known, set. In Levine's procedure subjects were apprised of the set size before the trials began. The Dynamic Expectancy Model makes no assumption about pre-existing sets of hypotheses. Hypotheses are generated and tested as the opportunity arises. In turn this gives rise to other possible μ-hypothesis creation (postulate H5) strategies. Implicit in the description so far is the idea that the animat initially creates a single, minimally simple hypothesis for each situation, tests that hypothesis for some while, and subsequently may need to refine or replace it. An alternative strategy might be to create a group of μ-hypotheses, utilising both the spatial and temporal aspects of the trace, and subsequently aggressively reject or delete all those from this sub-set that are not corroborated on subsequent trials, an "over-sampling" assumption. Under this assumption it may be appropriate that learned μ-hypotheses do not affect the behavioural repertoire until this initial selection phase is complete, leading to a flat section just prior to the main learning curve[14].

3.2.5. The Role of the Hypothesis in the Discovery Process

This thesis presents animal learning as a process of discovery. As part of the arguments leading to his development of the central thesis in his classic and seminal work into the nature of the scientific process, his "*Logic of Scientific Discovery*", the eminent Austrian born philosopher Sir Karl Popper (1902-1994) identified many essential properties of the hypothesis and its role in a self-sustaining discovery process encapsulated in a set of "methodological rules" (Popper, 1959). In this view of the discovery process "scientific truth" is determined by the creation of hypotheses, which are tested from the phenomena they predict. In turn experiments are devised to determine the validity of the prediction. This is a form of *modus tolens*[15], where theories from which hypotheses were properly derived are discarded when the hypotheses are falsified by experiment. While Popper

[14]Kleitman & Crisler (1927) present data showing a similar effect under classical conditioning conditions.

[15]If t, some theory, implies p, some conclusion (say a logically derived hypothesis), then the falsifying inference "$((t \rightarrow p).\neg p) \rightarrow \neg t$" requires us to reject t if we find p false.

decisively rejects inductive logic ("theory from examples"), he provides scant clue in these early writings as to how he considers theories themselves are to be formulated. Later authors active in the field of the philosophy of science have extended this model, and provided alternative views, of the scientific discovery process. Berkson and Wettersten (1984) have attempted to apply the principles of Popper's Logic of Discovery to the psychology of learning.

The "Logic of Scientific Discovery" (LSD) contains many insightful observations about the nature of the discovery process. A number of these observations, pertinent to expectancy theory and particularly relating to the nature of the hypothesis and experiments are considered now. Hypotheses that have more general applicability, those giving rise to a smaller range of derived "statements" and so have a higher "empirical content", have decreasing opportunity to escape *falsification* (LSD, s31). It is therefore incumbent on the discovery process to propose the simplest theories and hypotheses that are testable and so falsifiable, though simplicity itself is not a substitute for falsifiability. Hypotheses that are not testable ("undecidable" or "meta-physical") or those which are trivially true[16] ("tautologous") are to be discarded. Selection of the fittest systems of hypotheses should be as a result of the "*fiercest struggle for survival*" (LSD, s.6). Even if inadequate such systems of hypotheses should persist until falsified or replaced by one better able to be tested and found more fit.

Experiments are derived from, and test, hypotheses. Experiments must therefore encapsulate a complete description of the conditions under which the phenomena under test will be reproducible. Any conditions not included in the experimental procedure being considered irrelevant. In Popper's view a hypothesis may at best be corroborated, or otherwise falsified, and consequently the hypothesis and therefore the theory from which it was derived should be refined or refuted. In practice Popper recognises that there may be valid exceptions to the strict application of this approach, such as when the hypothesis fails due to incomplete specification, or where verifying observations have reached the limits of available experimental technique. In Popper's model of the scientific method hypotheses are

[16]It has subsequently become apparent that practical logic based systems which ignore the trivially true or apparently commonplace are prone to particularly gross omissions of reasoning (the "common-sense" component).

deduced from theories (the *Hypothetico-Deductive* approach). In the Dynamic Expectancy Model hypotheses are generated directly from observations and tested (the *Hypothetico-Corroborative* approach). In both schemes testing of hypotheses is a continuous process, the "*scientific game*" one without end. We may decide to suspend testing a hypothesis temporarily, but "*he who decides ... that scientific statements do not call for any further test, and that can be regarded as finally verified, retires from the game*" (LSD, s10).

Experiments are repeated so that we may "*convince ourselves that we are not dealing with a mere isolated coincidence*" (LSD, s.8). Popper refers to such coincidences as *occult occurrences*, repeated testing validates or rejects the phenomenon. A similar effect has been noted by experimental psychologists in animals, a behaviour based on a single rewarding circumstance, which persists even though the outcome is not repeated. This effect is usually referred to as *superstitious learning*, characterised as the elicitation of ritualistic or stereotyped behaviour under non-contingent "reward" schedules. Skinner (1948) describes an experimental schedule demonstrating the phenomenon in pigeons. Blackman (1974, Ch. 2) reviews "superstitious" behaviours in an operant conditioning context. This effect is apparently distinct from superstitious behaviour in humans, based on mystic or other beliefs (Jahoda, 1969).

3.3. Tokens, Signs and Symbols

Signs are specifically a combination of one or more elementary sensory units. They recognise a condition that may itself be composed of more than one sensory mode. In the Dynamic Expectancy Model these individual elements are referred to as *tokens*. Tokens perform the initial conversion of data from external transducers or sensors into symbolic form. Sensors abound in nature and it is not intended to further review the scope or extent of animal senses here. Similarly there have been significant advances in artificial transducers that may be incorporated into robotic devices. In the present model tokens will be represented as two-state symbols, indicating the presence or absence of the condition detected. This is a limitation that may need to be addressed in the future. The values of past tokens are recorded in an *activation trace*, specifically to allow *temporal discrimination*. By referring to elements in the activation trace behaviours may be related to past events, as well as those which are current.

3.3.1. The Sign and Token Postulates

Definition T0: **Token**. A token is a symbol relating to a basic unit of sensory input. A token indicates the instantaneous output from a detector. In the present model a token is either active or inactive, reflecting one of two possible detector states. Tokens are time tagged. They may represent the state of the detector at the current time or provide a record of the state of each detector at given times from the recent past (the "activation trace"). Older token records are discarded. Tokens may be attached to transducers to detect physical aspects relating to the animat and its environment. Tokens may also detect information processing activities within the animat.

Definition S0: **Sign**. A sign encapsulates a combination of conditions. These encapsulated conditions completely define the context ("s1") and the predicted outcome ("s2") for individual μ-experiments (postulate H2). A sign is a conjunction of tokens. Individual tokens may be negated (active to inactive, and vice-versa), providing an inhibitory connection. A token retains its time tag when incorporated into a sign.

Postulate T1: **Tokenisation**. Tokenisation is the process by which output from detectors is converted to an internal symbolic form. Such a token symbol may be considered as having a value associated with it that reflects the current (or past) output of the detector. The current token value changes according to the output of the detector.

Postulate S1: **Encapsulation**. Encapsulation is the process by which individual tokens are combined into a single sign. New signs are added to the system during μ-hypothesis differentiation (postulate H6).

Postulates T2 and S2: **Activation**. A token is considered "active" when the detector to which it relates is emitting the output relating to the tokenisation process. Similarly a sign is considered "active" when all its component tokens are (or were, in the case of time tagged tokens) active, taking into account any negations. Both tokens and signs may be considered as "tests" on the conditions they detect.

3.3.2. Initial Conditions for the Token and Sign Sets

The ethogram will define an initial set of tokens, and ensure they are attached to transducer and detector outputs. A single detector may be associated with several tokens, relating perhaps to different degrees or levels of output. The ethogram will also define any signal processing or transformations to be applied to detector output prior to tokenisation. The initial set of signs will contain one sign for each initial token, unnegated and reflecting the current value of the token. New tokens and signs may be added to the system during the lifespan. Tokens may be defined as active when the state of a transducer changes, either from off to on, or from on to off, or under both conditions. In the experimental conditions described in chapters five and six this effect is inherent in the nature of the environment and simulated transducers. Other environments, real or artificial, may call for specific signal processing to achieve these conditions.

3.3.3. Supporting Evidence for Signs and Tokens

There is a wide diversity of afferent and sensory mechanisms found in nature, and a substantial body of recent research into sensor and transducer systems for artificial animals and robots. This section addresses some of the issues, and presents a sample of sensory strategies to be found in nature. Above all it is clear that sensory sub-systems are far from amorphous, general purpose, elements. Nature abounds with well-documented examples of perceptual mechanisms tuned to the behavioural and learning requirements of their host animal. For instance, Tinbergen (1951, chap. 2) describes how the release strength of the food begging reaction varies in newly hatched herring gull chicks when presented a range of differently coloured model representations of the adult bill. Among many additional carefully observed and documented examples he also reports on the elicitation of the escape response in many species of bird when presented with silhouette profiles of predatory birds, while not reacting to silhouettes of other, non-predatory, species.

Arbib and his colleagues (Liaw and Arbib, 1993; Arbib and Cobas, 1990) have modelled the response of various frog and toad species to the threat posed by large looming objects as possible predators and the opportunity offered by small moving objects as potential prey. Additional neurological evidence that identifiable cells (or structures of cells) respond to external stimuli has been provided by the work of Hubel and Wiesel (1962), who reported that individual cells in the visual cortex become active when highly specific patterns are presented in the visual field of experimental animal subjects. Schölkopf and Mallot (1995) consider the experimental evidence for *place cells*, located in the rat *hypothalamus*, which fire (demonstrate significantly higher rates of electrical activity) when the rat is physically located in specific places.

Tokens, kernels (JCM and ALP) and primitive items (Drescher) are all abstractions from the totality of possible information that will be present at the time the token item is generated. The same is true in nature. The herring gull chick fails to note that the model bill is not a significant feature. The adult bird that the predator silhouette presents no threat - being made of wood and paint. On a different evolutionary path development of the innate releaser indicating this predator danger might be more specific, responding additionally to wing beat patterns, or hovering, swooping or other flight characteristics specific to the predator species. Foner and Maes (1994) point out that many current computer representations of input stimuli only take account of the current situation. This would also appear to be true for the majority of machine learning induction systems. Foner and Maes describe extensions to Drescher's original scheme to allow a one cycle record. This in turn allows extensions to the algorithm to focus attention on phenomena that change. Coincidentally there is also a significant body of evidence for single neurones that demonstrate firing activity specifically with respect to stimulus change.

The evidence for a *Short Term Memory* (STM) phenomena, employed in both JCM and ALP primarily rests with human nonsense syllable recall tasks. The evidence for an activation trace surmised from the apparent ability of various animal species to perform temporal stimulus differentiation. Recent reports implicate the *substantia nigra* brain area as a timing element capable of generating "metronome" like pulses in the millisecond to minute range to other parts of the brain (Highfield, 1996). This is a distinct phenomenon to the daily *circadian rhythm* (Lofts, 1970), which has been demonstrated to influence both physiological and behavioural aspects in a wide variety of species. There is extensive neurophysiological evidence that firing activation can continue after removal of a stimulus at the single neuronal level (an integration effect), though it is not obvious that these phenomena have significant or direct bearing on either the notion of STM or of the activation trace.

The encapsulation of multiple atomic conditions (the tokens) into the single symbolically identified ‘sign’ (the *sign-gestalt*) allows for an efficient and compact definition of the context-action-consequence triplet representation. Processing transducer and sensor data and hence the derivation of the input token is a critical issue for animat originators. Drescher’s *primary items* essentially unambiguously detect a state of the environment that is relevant to the algorithm; the position of the fovea, the location of the simulated hand and so on. By contrast the sensors on the robot used by Mott’s ALP system provided highly ambiguous and incomplete information. The same pattern of kernels was generated over a wide range of circumstances. The use of binary representations for light level, for instance, gave ALP little opportunity to determine the true consequences of its actions. In the experiments to be described in chapter six the creation of tokens is tightly coupled to the design of the environment.

3.4. Actions and Reification

The action and reification postulates define the efferent sub-system, which enables the animat to control *actuators* and so directly affect its environment. External actions, those which impinge on the environment, may be monitored by direct observation. Internal actions, such as those which affect the “physiology” of the animat, may only become apparent through measurement or by inference.

3.4.1. The Action Postulates

Definition A0: **Action**. An action is the basic unit of efferent event available to the animat. In the converse process to tokenisation, the animat may convert certain internal symbols into actions that directly impinge upon, and may change, the state of the animat or its environment. In keeping with tradition the terms “action” and “response” will be used essentially equivalently in this context throughout the thesis[17].

Postulate A1: **Reification**[18]. Reification is the process by which internal symbols are converted into detectable manifestations, for instance physical actions by the animat on the environment via its actuators. Such symbols may be delivered for reification by many routes within the model.

Postulate A2: **Action Cost**. The performance of any action by the animat will be presumed to consume resources otherwise available to the animat. Action costs may be measured in terms of energy expenditure, time taken to completion, or any other units that may be applied consistently within the confines of the ethogram, and which are appropriate to the physical and mechanical design of the animat and its actuators. Action costs are normalised to be 1.0 or greater, where 1.0 is taken as the minimum cost of any of the actions available to the animat.

Postulate A3: **Compound Actions**. Compound actions represent larger sequences of actions, which may be considered as a single tokenised item for reification. They are formed from simple actions (postulate R1) by concatenation. Compound actions formed in this way run to completion once initiated. The cost of a compound action will be taken as the sum of its individual component actions.

3.4.2. Initial Conditions for Actions

The list or vocabulary of actions initially available to the animat is defined in the ethogram. This vocabulary of actions will include all simple and compound actions and their associated costs. New actions may be added to the vocabulary during the lifespan of the animat.

---

[17] The action as “response” is a S-R behaviourist concept, it is therefore not entirely clear why the term should have been retained by those who did not necessarily regard “actions” as “responses”.

[18] (OED) reify v.t. Convert (person, abstract concept mentally) into thing, materialise; hence ~fication n. [f. L *res* thing + -I- + -FY]

3.4.3. Supporting Evidence for an Action Vocabulary

The ethogram may define actions over a wide range of complexity, from simple individual muscle or actuator motions ("molecular" in Tolman's vocabulary, or "characteristic" in McFarland and Sibly's, 1975) to increasingly complex combinations of actions which may be clearly recognised as a behavioural pattern ("molar" in Tolman's and "actions" or "activities" in McFarland and Sibly's). Each animal exhibits a vocabulary of "action patterns", apparently as characteristic of its species as is any physical attribute. The Dynamic Expectancy Model does not divide actions into "appetitive" and "consummatory", as in Tinbergen or Maes' models. In the Dynamic Expectancy Model actions may indeed lead to the satisfaction of a goal (q.v.), but goal satisfaction is rather a property of the goal description, not of any particular action that may precede the satisfying event.

Several detailed studies developing catalogues of essentially unitary behaviour "action patterns" in animals have been undertaken, for instance Shettleworth's work on the Golden Hamster (Shettleworth, 1975) or that of Reynolds' (1976) on the *Rhesus Monkey*. Shettleworth describes 24 mutually exclusive action patterns displayed by hamsters under laboratory conditions. Reynolds' work studied monkeys in a social setting, though in captivity, to prepare an extensive vocabulary of *action patterns*. Action patterns were described as either "postural" (68 distinct actions in 11 groups, including "attack", "threat", "dominance expressions", "submission", "grooming" and "sex") or "vocal", cataloguing the sounds made by his subjects. Reynolds provides comparisons with previous attempts at a terminology and discusses the difficulties in arriving at a uniform and agreed classification.

Mott's ALP used a list of five molecular actions ("<FORW>M", "<BACK>M", "<LEFT>M", "<RIGHT>M" and "<CRY>M"), corresponding to the translational and rotational movements available to the *QMC Mk. IV* robot. It is unclear what role the "<CRY>M" action played in the experimental set-up described. Drescher's system employed 10 molecular actions, four controlling foviation ("eyef", "eyeb", "eyel" and "eyer"), four controlling hand movements ("handf", "handb", "handl" and "handr"), and hand open and close ("grasp" and "ungrasp"). Many of the simulated and physical robot controllers based on classifier and reinforcement principles define action sets of similar size and complexity.

3.5. Goal Definitions

Goals represent the trigger or cue for the animat to engage in performing outcome directed behaviours.

3.5.1. The Goal Postulates

Definition G0: **Goals**. A goal establishes a condition within the animat causing the animat to select behaviours appropriate to the achievement or “satisfaction” of that goal. Goals are a special condition of a sign; goals are therefore always drawn from the set of available signs.

Postulate G1: **Goal Valence**. From time to time the animat may assert any of the signs available as a goal. Any sign asserted to act as a goal in this way is termed as having *valence* (or be valenced). None, one or many signs may be valenced at any one time. The converse condition, *aversion*, where the animat is required to avoid certain stimulus conditions is considered later (section 7.5).

Postulate G2: **Goal Priority**. Each valenced goal is assigned a positive, non-zero priority. This priority value indicates the relative importance to the animat of achieving this particular goal, in the prevailing context of other behaviours and goals. Goal priority is determined within the innate behavioural component of the ethogram. In the current model only one goal is pursued at any time - the *top-goal*, the goal with the highest priority.

Postulate G3: **Goal Satisfaction**. A valenced goal is deemed “satisfied” once the conditions defined by the goal are encountered, when the sign that defines the goal becomes activated (postulate S2). The priority of a satisfied goal is reduced to zero and it ceases to be valenced. Where goal seeking behaviour is to take the form of sustained maintenance of a goal state, the goal selection process must revalence the goal following each satisfaction event.

Postulate G4: **Goal Extinction**. In a situation where all possible paths to a goal are unavailable, continued attempts to satisfy that goal will eventually become a threat to the continued survival of the animat, by blocking out other behaviours and needlessly consuming resources. Such a goal must be forcibly abandoned. This is the *goal extinction point*. Goal extinction is closely related to the valence break-point postulate (P6).

Postulate G5: **Cathexis**. Cathexis associates a known goal sign with some other sign, following repeated simultaneous appearance. The association grows in magnitude with successive pairings and wanes to extinction should the pairing cease. This mechanism allows created signs to equivalence signs with innate goal properties.

3.5.2. Goals, Starting Conditions and Discussion

Goals are defined within the ethogram, and a mechanism must be defined to enable goals to be asserted whenever an appropriate circumstance arises. Current animat models, based on animal studies, might indicate the appropriateness of goals related to hunger, thirst, internal temperature control, external cleanliness, predator avoidance, location of shelter, mating, and so on (after Tyrrell, 1993). Goal setting and goal satisfaction need not be based on the same detectable phenomena. For instance, food seeking behaviour may be initiated by the detection of lowered blood sugar levels (or by changes in blood sugar controllers, such as insulin). However, due to the delay in the digestive process, were feeding to cease only when these levels were again elevated to a reasonable level the hapless creature would be gorged to bursting point. It has been demonstrated that many cues may be used to terminate feeding behaviour, the action of eating, the taste of sweet but non-nutritious saccharin solution, or by artificial distension of the stomach (by an inflated rubber balloon inserted into the gut). Clearly an overall balance must be achieved between long-term and short-term signals to ensure that behaviour and driving needs are matched.

Goals need not relate to physical requirements, and may be asserted by other mechanisms. Maes (1991) describes “curiosity” as a goal type, related to “exploratory” behaviours. Yet curiosity is rather the description of a process that involves exploratory or deliberate actions to elicit further information about goals. Such goals may be activated on an arbitrary basis, or specifically to provide additional maturity to a μ-hypothesis, to disambiguate between contradictory μ-hypotheses, or to engage in the process of *play*[19].

3.6. On Policies and Policy Maps

Whenever any goal is valenced (postulate G1) the Dynamic Expectancy Model calls for the animat to construct a Dynamic Policy Map (DPM). As with a *Q*-learning policy map, the DPM allows the animat to select an action based on an estimate of least cost path to the current goal. The DPM is constructed from all the μ-hypotheses available to the system at the time of its construction. Unlike the static policy map of *Q*-learning, commitment to any particular DPM structure and values is not made until the point a goal becomes valenced (G2).

3.6.1. Policy Map Postulates

Definition P0: **Dynamic Policy Map**. The Dynamic Policy Map temporarily assigns a measure of "effectiveness" to every sign known to the animat (the "policy value", q.v.) This effectiveness measure is an estimate of the effort that will need to be expended in traversing from any current situation (as defined and detected by a sign), to the goal sign with the highest given priority (postulate G2). The current DPM is discarded when its goal is satisfied (G3). A new DPM is reconstructed whenever a new top-goal is selected, or when either the set of μ-hypotheses (H5, H6 or H7), or their corroboration measures (H3 and H4) change significantly.

Postulate P1: **Induced Valence**. Any μ-hypothesis whose consequence sign ("s2") is identical to the top-goal sign, or to any sign with valence (postulate G1), induces valence into its context sign ("s1").

---

[19] Play (Dolhinow and Bishop, 1972; Hinde, 1970, pp. 356-359), has been widely observed in animal behaviour, in particular in primates and humans and other mammalian and avian species. Play is not observed in fish, amphibians and invertebrates. Play in animals is most often encountered as incomplete or stylised versions of recognisably adult behaviours, but it is not triggered by normal motivational cues and is without the expected consummatory component. There is a notable suppression of harmful aspects to the normal behaviour manifestation, such as biting. It is also easily interrupted by threat or hunger. Play is often associated with the individual's development in a social context, and as a way of gaining motor skills. It may also have an explicitly exploratory component.

Postulate P2: **Spreading Valence**. Any μ-hypothesis not already valenced, and whose consequence sign ("s2") matches a context sign of another μ-hypothesis that is valenced itself gains valence. Valence is induced (postulate P1) into the context sign, the context sign of the newly valenced μ-hypothesis may now act as a *sub-goal*. Valence may therefore spread throughout the set of μ-hypotheses and signs until all μ-hypotheses have acquired valence, or until no more μ-hypotheses can be reached by this process. The top-goal is defined as having a "valence level" of zero; each level of induced valence increases the valence level by one.

Postulate P3: **Cost Estimate**. The cost estimate for using any action associated with any μ-hypothesis shall be the action cost (postulate A2) divided by the corroboration measure (H3, eqn. 3-1). Thus if the μ-hypothesis has always successfully predicted the consequence its cost estimate (P3) will be equal to the action cost. Where the corroboration measure indicates a less successful rule, the cost estimate rises. Where the μ-hypothesis has always failed the cost estimate would tend to infinity. The reinforcement measure (H4) may be used equivalently in this calculation.

$$\text{cost estimate} \leftarrow \text{cost}(r1) \,/\, p(s2 \,|^{t}\, s1+r1) \qquad \text{(eqn. 3-4)}$$

Postulate P4: **Policy Value**. The *spreading valence* (postulate P2) process creates *policy chains*, indicating one or more paths or chains of actions (extracted from μ-hypotheses implicated in the valenced policy chain) extending between the goal and any sign involved in the DPM. The policy value for any sign that is not the goal and which is involved in the DPM is defined as the sum of individual cost estimates (P3) for each element in the policy chain. In practice the spreading valence method produces a graph or net like structure. Any policy chain shall be defined as comprising the transitions representing the policy cost of lowest overall value between pairs of sign nodes in that chain.

$$\text{Policy value}(s^{n}) \leftarrow \min\Big(\sum_{v=0}^{v=n-1} (\text{cost}(r1^{v+1}) \,/\, p(s2^{v} \,|^{t}\, s1^{v+1}+r1^{v+1}))\Big) \qquad \text{(eqn. 3-5)}$$

where *v* is the valence level of each link in the policy chain formed and *n* is the valence level of some sign “s”.

Postulate P5: **Action Selection**. Whenever there is a valenced top-goal (and so a DPM) an action may be selected for reification from the μ-hypothesis implicated in the DPM whose context sign is both active (postulate S2) and which has the lowest policy value (P4).

Postulate P6: **Valence Break Point**. Creating a DPM (postulate P2) and selecting an action (P5) establishes within the animat an expectation that the top-goal may be achieved at a certain cost (P4). The model defines a *valence break point* (VBP), typically some multiple of the *policy value* (policy value * n). When actions selected from the DPM fail the policy value rises. Should the policy value exceed that of the previously computed valence break point, goal directed behaviour is suspended, with the animat reverting to exploratory behaviours for a time. During this period the animat may create new μ-hypotheses if the opportunity arises, offering the possibility of a new path chain to the goal. Goal directed behaviour is reinstated with a less demanding valence break point (the policy value is now higher). Goal directed and exploratory behaviours alternate until either the goal is reached, or the goal is finally cancelled by the extinction process (G4). This process mirrors the experimental extinction phenomena repeatably observable in animal experiments (figure 3-1).

3.6.2. Evidence for Chaining

Evidence that animals may form explicit behaviour chains under controlled conditions is described by Blackman (1974). Such chains are created by the experimenter by manipulating the animal in an operant conditioning set-up to elicit some response, say Rx, to achieve a reinforcing reward under some discriminating stimulus situation, say Sx. Following this stage a response, say Ry, is conditioned to Sx, but only in the presence of another discriminating stimulus, Sy. Sx has no inherent reward characteristics, but acts as a *conditioned reinforcer*. Using this method chains of considerable length and complexity have been reported.

Sy → Ry → Sx → Rx → reward

An independent series of experiments on the *latent extinction* phenomena demonstrates that these behaviour chains may be disrupted, weakened or broken when individual elements of the chain are extinguished (Bower and Hilgard, 1981, describing the work of Stewart and Long, and others.) The ability to construct, and disrupt behaviour chains is not in itself direct confirmation of induced valence, but is important supporting evidence. Experience from animal training (Bower and Hilgard, 1981, p. 179) suggests that the chain need not be built up backwards from the primary source of reinforcement, but may also be built forwards, or by inserting operant elements into existing shorter chains.

3.6.3. Evidence for Goal Suspension and Extinction

Figure 3-1 shows stylised cumulative records (from Blackman, 1974, p.67, after Reynolds) derived from Skinner box experiments under various operant conditioning reinforcement schedules. In the *fixed ratio* (FR) schedule "reward" is delivered to the animal after a fixed number of "responses". In the *variable ratio* (VR) schedule "reward" is delivered after a random number of "responses". In the *variable interval* (VI) schedule "reward" is delivered at randomly varying intervals, independently of actions by the animal. Similarly, the *fixed interval* (FI) schedule delivers "reward" after a fixed interval of time, again independently of "responses" by the subject. All these schedules are applied to animals that have previously been conditioned to operate the Skinner box apparatus on a regular reward schedule.

The slope of the curve indicates the rate of the learned response (each response causes an upwards increment in the trace), downward "tick" marks indicate individual reinforcing reward events. Note the characteristic stepped form of the curve in the *extinction* phase of the experiments following the cessation of reward events. The stepped form reflects the changing relationship between two forms of activity during the extinction phase, shortening periods when responses are made, and lengthening periods when no responses are made. In time the learned response is apparently completely eradicated. This extinction process is a highly repeatable phenomenon, and has been widely reported under both classical and operant conditioning regimes. Experimental regimes also indicate a secondary process of *spontaneous recovery*, in which the previously extinguished effect re-appears, albeit in a weakened form, after a period of rest.

The Dynamic Expectancy Model emulates the shape of the extinction curve by the combined effects of the reinforcement (H4), valence break point (P6) and goal extinction (G4) postulates. Specific details of how these interact in the implemented model, and experimental analysis of the effects are described later. Extinction curves of the type shown in figure 3-1 indicate the manner in which an animat may abandon use of individual μ-hypotheses that prove ineffective. The reinforcement schedules themselves may yet reveal much about how μ-hypotheses may be created and managed in an animat designed with biological plausibility in mind.

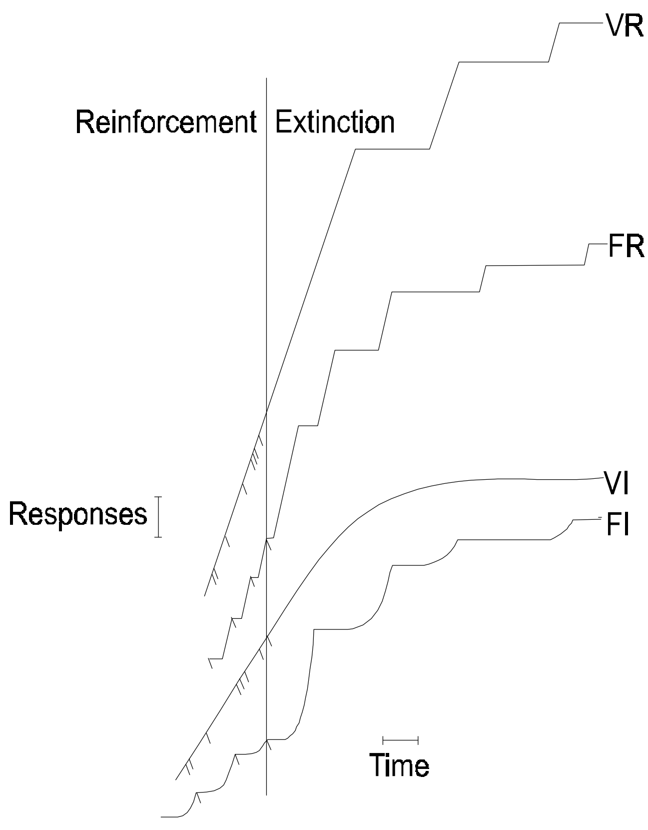


Figure 3-1: Extinction Curves Under Various Schedules

3.6.4. Comparison to *Q*-learning

The Dynamic Expectancy Model is based on a different set of fundamental premises to that of the reinforcement and *Q*-learning strategies of Sutton and Watkins. Watkins (1989, p.16) summarises the situation for *Q*-learning in three position statements: (1) that the capacity for maximally efficient performance is valuable; (2) that exploration is cheap; and (3) that the time taken to learn a behaviour is short compared to the period of time during which it will be used. Statement (1) is hardly in contention. Statements (2) and (3) indicate that the ultimate level of performance is inherently more important than the time taken to achieve it. “Optimality” is thus defined as maximising reward acquisition over an extended time period. Learning in the Dynamic Expectancy Model aims to provide the animat with the best path to achieve goal (reward) states as they become indicated, given the current level of knowledge. It may be that as the animat becomes more experienced the quality of that path might be expected to converge to some acceptable notion of “optimal”[20] behaviour. This would be the case, as discussed under the FDMSSE conditions considered earlier, except for the competing requirement that the animat continue to explore while any phenomena remain unpredicted, an innate drive to continuously augment and refine its state of knowledge.

3.7. Innate Behaviour Patterns

Innate behaviour patterns provide a grounding for intelligence. In the Dynamic Expectancy Model innate behaviours serve three distinct roles. First they provide the animat with sufficient behaviour to survive in its environment from *parturition*, before any learning. These behaviours imbue the animat with strategies to react to life threatening events, where learning would represent too high a risk for failure on the initial instances; predator avoidance for instance. Second to select and set goal priorities. Most goal directed behaviour serves basic physiological requirements. Innate behaviour detects conditions indicating those requirements and establishes them as goals. Third to provide a level of background behaviour to

---

[20]Optimality, like beauty, is in the eye of the beholder. The *Q*-learner may regard the shortest path between current state and reward state as the optimal path. A hungry predator waiting beside this path may agree.

ensure the animat is appropriately tasked whenever neither the primary nor secondary roles are activated. It may be appropriate that the animat enters a state of hibernation, torpor or sleep, a strategy that may conserve energy or serve other physiological functions. The animat may also use these periods to perform exploratory actions, thereby triggering μ-hypothesis creating postulates, and performing acts that corroborate existing μ-hypotheses. It is a consequence of the Dynamic Expectancy Model postulates that learning may take place in the absence of explicit reinforcement. Several strategies for this exploration may be applicable.

Definition B0: **Behaviours**. Behaviours are unlearned activities inherent within the system. Behaviours give rise to actions (postulate A0) in response to circumstances detectable by the animat. They are defined prior to parturition as part of the ethogram. There is no limit to the complexity (or simplicity) of innate behaviour. An animat might be solely dependent on innate behaviours, with no learning component.

Postulate B1: **Behaviour Priority**. Each behaviour within the animat is assigned a priority relative to all the other behaviours. This priority is defined by the ethogram. The action (postulate A0) associated with the behaviour of highest priority is selected for reification (A1).

Postulate B2: **Primary Behaviours**. Primary behaviours define the vocabulary of behaviour patterns available to the animat at parturition. These behaviours provide a repertoire of activities enabling the animat to survive in its environment until learning processes may provide more effective behaviours.

Postulate B3: **Goal Setting Behaviours**. The ethogram defines the conditions under which the animat will convert to goal seeking behaviour. Once a goal is set the animat is obliged to pursue that goal while there is no primary behaviour of higher priority. Where no behaviour can be selected from the DPM, the animat selects the behaviour of highest priority that is available. Behaviour selection and reification (A1) from the DPM resumes once there is any match between the set of active signs (S2) and the current DPM (P5).

Interruption of goal directed behaviour by a higher priority innate behaviour may draw the animat away from its top priority goal. For instance, goal directed nourishment seeking behaviour may be interrupted by high priority predator avoidance activity. Once the threat is passed goal directed behaviour will be resumed, although the animat's perceived "place" in the DPM graph will have shifted as a result of the intervening behaviour. The structure and corroboration of the DPM may have changed, and it must be re-evaluated as behaviour reverts to the goal directed form. Where goal seeking takes the form of a sustained maintenance of the selected goal state, the selection process must reassert the required goal each time it is satisfied.

Postulate B4: **Default (exploratory) Behaviours**. Default Behaviours provide a set of behaviours to be pursued by the animat whenever neither a primary nor goal setting behaviour is in force. Typically these default behaviours will take the form of exploratory actions. Exploratory actions may be either random (*trial and error*), or represent a specific exploration strategy. Selection of this strategy will impact the rate and order in which the μ-hypothesis creation processes occur (H5). Default behaviours have a priority lower than any of the primary (B2) or goal setting (B3) behaviours. The provision of default behaviours is mandatory within the ethogram.

3.7.1. Balancing Innate and Learned Behaviour

The balance between innate and learned behaviour varies widely throughout both nature and the study of artificial animats. Action selection models, such as those of Brooks, Chapman and Agre, Maes, and Tyrrell, place full emphasis on the provision of pre-programmed behavioural activity. Behaviours are selected to give the animat appropriate responses to its environment, and as a consequence animat behaviour may appear "intelligent" by virtue of this applicability. In this case the originator imbues the animat with a mechanism to determine which needs are required, and a mechanism to balance between them. Within its repertoire of innate behaviours a simulated animal may manage its requirements for nourishment and water, for warmth, for shelter, predator evasion and the need to procreate.

Similarly a robot may be programmed to partition its activities into different, and mostly mutually exclusive, behaviours - collecting soda-cans, environmental mapping, avoiding unexpected obstacles, seeking its recharging point and replenishing its batteries. Each robot may incorporate these, and other tasks, whose usefulness and complexity are limited primarily by the imagination, patience and programming skills of the robot designer. Recall that μ-hypotheses may themselves be defined in the ethogram, consequently the Dynamic Expectancy Model does not imply that all goal seeking behaviour must be learned.

At the other end of the scale many adaptive learning models adopt a *tabula rasa* approach. With little or no predefined coherent behaviour, they rely instead on a (pre-defined) learning mechanism to accumulate sufficient information about the environment to eventually create coherent and appropriate overt behaviour. Reinforcement and *Q*-learning schemes fall into this category, as does Drescher's schema system. Initially actions are selected at random, under a *trial and error* regimen and internal structures built or existing structures populated. With the application of sufficient trials purposive behaviour may be generated from the structures and information accumulated.

Mott's ALP was essentially initially a *tabula rasa* system, but a small number of low-level robot reflexes were provided. To prevent the robot becoming physically trapped into corners a reflexive backoff mechanism was pre-coded into the robot control-level controller. This is a recurring problem for mobile robot constructors, exacerbated in this instance due to the physical layout of the robot used, a square outline with differentially powered wheels forward of the centre-line. For this reason many mobile robots are designed with a circular, or at least rounded "floor-plan", with their drive wheels placed symmetrically about the centre-line. A second low-level innate reflex was found to be necessary to suppress the backoff reflex when the robot was at the charging point. This "discriminating push" reflex prevented contact with the charger being broken, ensuring that effective electrical contact was maintained between the robot's charger contact plates and the sprung base station charger contacts throughout the recharging period.

3.8. Advances Introduced by the Dynamic Expectancy Model

Cursory inspection of the *Dynamic Expectancy Model* postulates H3 and H4 might suggest that this is a conventional reinforcement model of learning. Procedures (encapsulated by equations 3-1, 3-2 and 3-3) by which reinforcing events strengthen or weaken disposition of the animat to adopt one behavioural option over another are similar to those of other well-established reinforcement methods. The source of the reinforcement is, however, radically different. In the Dynamic Expectancy Model the reinforcement signal is internally generated by the setting and subsequent verification of a prediction. In previous reinforcement systems the reward signal must be received from the external environment before any learning could occur. In the new model a valid reinforcement signal is generated whenever a behaviour choice is exercised and a μ-experiment activated, so that the processes of behaviour may now be largely disassociated from those of learning.

It will be demonstrated later that this new method allows for substantially improved learning rates over conventional reinforcement learning techniques (section 6.2). It is quite clear that learning triggered by external reinforcing reward is also a valid effect, and commonly observed in animals. While this thesis primarily explores the effects of internally generated reward, it will be demonstrated (section 7.4) that additional performance benefits may accrue to the animat when internal expectancy and external reward signals are combined.

The Dynamic Policy Map arises from the fundamental disassociation of the learning and (goal-seeking) behavioural processes. In the static policy map of, say, the *Q*-learning algorithm, each sensory state becomes increasingly permanently attached to a particular action relative to a fixed goal. While this may bring advantages in enhanced reaction times following the learning phase, it leads to an inflexible reaction to the changing needs of the animat with time and varying goals. The Dynamic Expectancy reinforcement method of learning allows the construction of a policy map only when it is required, and relative to the specific needs of the animat at the time of construction. μ-Hypotheses become "committed" to a particular goal only while that goal has the highest priority, and will be reallocated whenever the goals of the animat change. An example of this dynamic map construction will be given in section 4.9.3.

By generating the policy map dynamically in this way the advantage of the reactive response to active signs inherent in the static policy map is retained. By not committing any individual μ-hypothesis to any particular goal or reward during the learning process the Dynamic Policy Map may be reconstructed to provide a reactive policy relative to the current goal, even where the goal has not previously been implicated in the learning process.

By integrating expectancy learning with an action selection based model of behaviour a way of selecting goals is made possible. This combination of techniques also provides a way of defining innate, reactive stimulus-response behaviours. These innate behaviours provide the animat with a mechanism with which to react in a manner to allow survival while the individual learns the skills required to behave ever more appropriately in its environment.

Chapter 4

4. The SRS/E Algorithm

This chapter describes the *SRS/E* computer algorithm. SRS/E is derived directly from the *Dynamic Expectancy Model* postulates of learning and behaviour developed in the previous chapter. SRS/E follows in the tradition established by Becker's JCM, Mott's ALP and Drescher's systems by providing an intermediate level cognitive model based on the context-action-outcome triplet. As with these previous systems, SRS/E offers a sensory-motor view of learning. It is not, however, to be considered as a re-implementation of any of these existing systems. As with Mott's ALP and Drescher's algorithm, and indeed the majority of extant animat control algorithms, SRS/E is based on a repeating cycle of sensory acquisition from the environment, processing and taking overt actions into the environment.

Each model is a reflection of the times in which it was created. Becker's JCM proposal and Mott's ALP implementation adopt an associative net structure for schemata LTM; consistent with prevailing theories from psychology and cognitive science, for example, Norman (1969). Adopting a net structure served to contain the computational search and matching load inherent in these designs, bringing distinct practical advantages to Mott's implementation in the context of a time-sharing ICL mainframe. Drescher's later (1991) system adopted a "neural crossbar architecture", consistent with the revival of interest in connectionist thinking at that time. Availability of the massively parallel *Connection Machine* made the brute force approach of the marginal attribution algorithm feasible. In turn, SRS/E arises as a reaction to an upsurge of interest in reinforcement learning and related behaviourist concepts. SRS/E's name, an abbreviation of Stimulus-Response-Stimulus/Expectancy, pays passing tribute to the life's work of E.C. Tolman, and defines the positioning of the work. Various other items of terminology, notably the use of *Sign*, *Valence, Hypothesis* and (*Cognitive*) *Map*, are derived from the vocabulary developed by Tolman and his contemporaries.

In contrast to these other systems SRS/E is primarily an algorithm that manipulates lists of data. This chapter is divided into two main parts. In the first part the various types of data list are described. The second part presents the algorithm used to manipulate the lists, perform the learning tasks and generate overt behaviours, either from the animat's predefined ethogram, or as a consequence of learned information.

4.1. Encoding the Ethogram: SRS/E List Structures

SRS/E currently defines seven internal data structures. These data structures will be referred to as *lists*. Each list encapsulates an aspect of the animat's ethogram, and so record the instantaneous "state" of the animat. At defined points in its execution cycle the SRS/E algorithm will inspect the contents of these lists and generate behaviours based on the prevailing contents of those lists. Equally the SRS/E algorithm will add, modify or delete information stored on the lists by processes derived from the Dynamic Expectancy Model postulates described in chapter three. These processes will be defined later in this chapter. Each of the seven lists is composed of *list elements*. In turn each element of each list is itself composed of *list element values*, which record items of information relevant to each list element. So, for example, the Hypothesis List is composed of many individual μ-hypotheses, the elements of that list. Each μ-hypothesis has attached to it various hypothesis values, which are created and initialised at the same time as the individual μ-hypothesis, and may be updated each time the algorithm utilises the individual μ-hypothesis. All list element values (or "values") are updated by the SRS/E algorithm as a result of events impinging on the animat and actions the animat makes. The list structures, list elements and list element values are summarised in table 4-1, and described in the sub-sections that follow. List elements may be defined by the originator before the creation of an individual animat, as would be the case with the Response and Behaviour Lists. Otherwise, as would be typical for all the other lists, lists are empty at the point the animat becomes a free standing individual. In which case the SRS/E algorithm creates individual list element entries as the need arises.

4.1.1. List Notation

Throughout this chapter each of the seven lists will be represented by a single calligraphic character. Upper-case characters represent complete lists ($\mathcal{I}$, $\mathcal{S}$, $\mathcal{R}$, $\mathcal{B}$, $\mathcal{H}$, $\mathcal{P}$ and $\mathcal{G}$). Lower case characters represent individual elements in the respective list ($\mathscr{i}$, $\mathscr{s}$, $\mathscr{r}$, $\mathscr{b}$, $\mathscr{h}$, $\mathscr{p}$ and $\mathscr{g}$). Table 4-1 summarises this notation. A superscript notation will be adopted to indicate some property of a list or a list element. In particular the use of an asterisk will indicate "active" elements, those whose attributes match the prevailing circumstances on the current execution cycle. For instance $\mathcal{I}^*$ will refer to all those elements of $\mathcal{I}$ where the corresponding token has been detected in the sensory buffers $\mathcal{I}^* \subseteq \mathcal{I}$, therefore $\mathcal{I}$ - $\mathcal{I}^*$ will refer to all those elements of $\mathcal{I}$ where no corresponding input token has been detected. A number of additional superscripted forms will be introduced later; each will indicate some sub-set of a list, or specify some attribute of a list element. A notation in which the list element value name is used to refer to or access a list element or sub-list will also be employed.

As with JCM, ALP and Drescher's system every element in each SRS/E list has attached to it a number of numeric and other values. These values are updated as the algorithm executes and are in turn used by the algorithm in selecting overt behaviours and to guide the learning process. SRS/E is intended primarily as a platform for experimentation. List element values are therefore variously available for use in the algorithm as presented, and by reporting and analysis software created with the specific purpose of analysing and presenting experimental results. The list element values used by SRS/E are shown in table 4-1. Their functions and purposes are described following a detailed description of each list type. Such values will be shown in a different font "`thus`". List element value names shown in this different font are chosen to directly reflect the variable names employed in the current implementation of SRS/E used to conduct the experiments described in chapter six. The character in brackets associated with each value shown in table 4-1 indicates the data type selected for that value in the current implementation. A calligraphic character, "($\mathscr{s}$)" for example, indicates a pointer or reference to a list element of the indicated type; "(i)" indicates an integer type; "(t)" a "time" value, and "(b)" a bit-sequence. The types "(i)", "(t)" and "(b)" are all encoded conveniently as (long) integers. Time values are recorded as discrete intervals corresponding to execution cycles of the algorithm. ASCII encoded strings are indicated "(s)", real or floating point values as "(f)". The range of some floating point values will be restricted within the program.

List Symbol	List Description	List Element Symbol	List Element Values
$\mathcal{I}$	**Input Token List**. Binary, atomic input items from sensors. Associates input items to arbitrary internal symbols	i	`token_string` (s) `token_identifier` (i) `token_first_seen` (t) `token_last_seen` (t) `token_count` (i), `token_prob` (f) `token_activation_trace` (b)
$\mathcal{S}$	**Sign List**. Descriptions of an environmental "state", defined by a conjunction of tokens (i) and other internal symbols	s	`sign_conjunction` (see text) `sign_identifier` (i) `sign_first_seen` (t) `sign_last_seen` (t) `sign_count` (i), `sign_prob` (f) `sign_activation_trace` (b) `best_valence_level` (i)
$\mathcal{R}$	**Response List**. All available actions (simple and compound)	r	`response_string` (s) `response_identifier` (i) `response_cost` (f) `response_activation_trace` (b)
$\mathcal{B}$	**Behaviour List**. (condition,action) defined innate behaviour patterns (condition $\in \mathcal{S} \rightarrow$ action $\in \mathcal{R}$).	b	`condition` (s) `action` (r) `behaviour_priority` (f)
$\mathcal{G}$	**Goal List**. Actual or potential system goals, prioritised by $\mathcal{B}$.	g	`goal_sign` (s) `goal_priority` (f) `time_goal_set` (t)
$\mathcal{H}$	**Hypothesis List**. List of μ-hypotheses in the form (s1,r1,s2) s1 $\in \mathcal{S}$, r1 $\in \mathcal{R}$, s2 $\in \mathcal{S}$.	h	`s1` (s), `r1` (r), `s2` (s) `time_shift` (t) `hypo_identifier` (i) `hypo_first_seen` (t) `hypo_last_seen` (t) `hypo_activation_trace` (b) `recency` (i), `hypo_bpos` (f) `hypo_cpos` (f), `hypo_cneg` (f) `hypo_age` (t), `hypo_maturity` (i) `hypo_creator` (h) `valence_level` (i) `cost_estimate` (f) `policy_value` (f)
$\mathcal{P}$	**Prediction List**. List of predictions awaiting confirmation.	p	`predicting_hypo`(h) `predicted_sign`(s) `predicted_time` (t)

Table 4-1: SRS/E Internal Data Structures

4.1.2. Summary of Lists

The **Input Token List** records binary atomic input items from system sensors and assigns each one a unique, but arbitrary, internal symbol such that each subsequent appearance of the same input item will generate the same internal symbol. The Input Token List implements the "token" of definition T0.

The **Sign List** provides the system with partial or complete descriptions of the environmental "state". A sign is defined as a conjunction of input tokens and other internally generated symbols, and their negations, providing the structure to implement the sign of definition S0.

The **Response List** defines the set of all the actions available to the animat, to implement the action of definition A0. Simple actions are defined by the ethogram. Compound actions (postulate A3) may be formed by the concatenation of simple actions.

The **Behaviour List** explicitly defines the innate behaviour patterns for the animat as an integral part of the ethogram (definition B0). Fixed, pre-programmed, behaviour patterns (postulate B2) may subsequently be subsumed by learned, goal-seeking behaviour. For simple animat ethogram definitions the Behaviour List will also be responsible for setting goals (postulate B3) and so balancing the priorities between fixed and learned behaviour.

The **Goal List** records none, one or more possible goals being sought by the animat at any particular time (definition G0). The animat only pursues one goal at any one time, the *top-goal*.

The **Hypothesis List** records learned expectancies (μ-hypotheses) in the form "s1,r1,s2". Context "s1" and consequence "s2" are elements from the Sign List. Action "r1" is an element from the Response List. Each element of the Hypothesis List equates directly to a single μ-hypothesis, a small, isolatable fragment of knowledge about the animat's existence, well defined in terms of the other list types (definition H0). To be of value to the system each μ-hypothesis must make a clear and verifiable prediction. Corroborated μ-hypotheses are subsequently used by the animat to generate useful goal-seeking behaviours. The SRS/E algorithm provides the algorithmic resources to create, verify, modify, delete and use μ-hypotheses.

The **Prediction List** records expectations made by activated μ-hypotheses for confirmation or denial at a defined time. This structure retains time tagged predictions until they are verified (postulate H1).

4.2. Tokens and the Input Token List

SRS/E employs a grounded symbol approach to behaviour and learning and has much in common with the notion of *deictic representation*[21] (Agre and Chapman, 1987; Chapman, 1989; Whitehead and Ballard, 1991). *Deictic markers* point to aspects of the perceivable environment. Ideally each marker will point to only one object or event, or to one well-defined class of objects or events, in the environment. This allows the animat to respond appropriately to the presence of the object or occurrence of the event, or to learn the significance of the object or event with minimal ambiguity (the FDMSSE assumption).

Typically input tokens either directly reflect the value of some sensor, or are derived from sensor values to define a partially or wholly complete state descriptor. Thus SRS/E will equally accept ALP style kernels, such as "<LOW>S" or "<BRIGHT>S", derived directly from the transducer values from the robot, or Drescher's (1991, p117) *primitive items* "hp11", "vp11", or "fovf00-33" denoting partial state descriptors from the simulated environment. As with Mott and Drescher, SRS/E input tokens are binary in nature, present or absent. SRS/E does not employ the predicate and value representation described by Becker.

The SRS/E algorithm accepts sequences of tokens from the environment. During each execution cycle none, one or many tokens may be presented to the algorithm from a sensor sub-system integral with the animat. The first appearance of any token is registered into the Input Token List, $\mathcal{I}$, and the new token is assigned a unique internal code. This realises the *tokenisation* process, described in postulate

---

[21](OED) deictic: a & n, Pointing, demonstrative, [Gk: deiktikos]

T1. For every subsequent appearance of that token the unique code will be generated from the list. At each execution cycle the Input Token List $\mathcal{I}$ will be partitioned into those tokens that have appeared in the input stream on the current cycle and hence are active, and all the others that have not appeared and are not active. As indicated in section 4.1.1 the active partition is denoted $\mathcal{I}^*$.

Tokens may be registered into $\mathcal{I}$ by the originator as part of the initial ethogram definition and subsequently employed in generation of innate behaviour patterns. Apart from this, tokens have no inherent "meaning" to the system. Once registered into the Input Token List, token identities are permanently retained. SRS/E will accept new additional tokens at any point in the lifecycle of the animat. The appearance of novel tokens also drives the learning process. There is no generalisation over input tokens; non-identical input token strings are treated as wholly distinct.

The Input Token List is implemented as a *hash table* (Knuth, 1973), the internally generated token symbol value being set equal to the index position in the hash table. Initially the hash table is given a fixed size, but is grown automatically and the symbols re-hashed when the table is close to overflow. As part of this process all internal token symbol values are updated to reflect their new position in the table.

4.2.1. Input Token List Values

In addition to the `token_identifier`, the internal symbol, and the external representation of the token string `token_string`, the Input Token List maintains four additional numeric values for each Input Token List element. As an aid to the analysis of experimental data the input `token_string` is retained in the Input Token List and is shown in preference to the anonymous internal symbol in output trace and log files. The list element value `token_first_seen` records which execution cycle the token i was first detected. The value `token_last_seen` records the execution cycle when the token was most recently detected. The value `token_count` records the total number of cycles that the token i has occurred on $\mathcal{I}^*$. The raw probability of occurrence (`token_prob`) for any token may be derived according to the equation:

$$\texttt{token\_prob} \leftarrow \frac{\texttt{token\_count}}{\texttt{now - token\_first\_seen + 1}} \qquad \text{(eqn. 4-1)}$$

This raw token probability may be used as a measure to determine the degree to which the sensory sub-system is able to differentiate the phenomena indicated by the token from others. Generally, tokens with a relatively low raw probability measure facilitate the behavioural and learning process.

A record of recent past activations for each element ι is maintained in the variable `token_activation_trace` according to the assignments:

$$\texttt{token\_activation\_trace}^{t\text{-n-1}} \leftarrow \texttt{token\_activation\_trace}^{t\text{-n}} \qquad \text{(eqn. 4-2a)}$$

$$\texttt{token\_activation\_trace}^{\text{now}} \leftarrow \mathcal{I}^{\text{ activation\_state}} \qquad \text{(eqn. 4-2b)}$$

These trace values, and those for other list element types, are used in sign definitions to record past activations and provide a mechanism to implement *temporal discrimination*, an aspect of the μ-hypothesis differentiation process (postulate H6). The activation traces are of finite length, newer values entering the trace displace older values which are lost to the algorithm.

In the current implementation of SRS/E, n of equation 4-2a takes the values 1 to 32. The token activation trace is therefore conveniently represented as individual bit positions in a long integer. The operation described by equation 4-2a is achieved in the current SRS/E implementation as an arithmetic shift left by one bit position. The operation described by equation 4-2b by setting (or clearing) the lowest order bit of the integer recording the trace values according to the current activation value of the token.

4.3. Signs and the Sign List

Signs encapsulate one or more tokens into a single item (this is derived from postulate S1). They are identified within the system by a unique symbolic identifier.

The total Sign List is designated as $\mathcal{S}$. The subset of signs that are active at the current time are designated as $\mathcal{S}^*$. Sign activation was described by postulate S2.

4.3.1. Representing Signs

As with the schema representations of Mott and Drescher, SRS/E signs are a conjunction of primitive tokens, where the token must be present for the conjunction to be active, or negated tokens, where the token must not be present for the conjunction to be active. Drescher's representation is severely restricted with respect to Mott's in that the schema left hand side in ALP allowed inclusion of kernels from any position in *Short Term Memory* (STM), whereas Drescher's did not. Mott's use of the little arrow notation, with its strict time sequence information, imparts further contextual information to the schema left hand side. SRS/E also adopts an explicit time representation to tokens, so:

ALP: [<BRIGHT>S → <FRONT>S ¬ <CHARGE>S ....]

becomes:

SRS/E: (bright$^{@t\text{-}1}$ & front & ~charge ....)

In SRS/E all timings are considered to be relative to the current cycle (*t*=0 or, equivalently, *t*=now), negative from the past, positive into the future. Thus the notation "@*t*-1" is conveniently read as "at the current time minus one", or "on the cycle before the current one". *Token negation* is represented by the tilde character ("~"). The representation of past events in ALP is limited to the length of STM (typically six cycles), in SRS/E by the length of the activation trace (typically 32 cycles). Unlike Becker, but like SRS/E, Mott did not permit recycling of kernels from the end of STM into the input register as essential timing information is lost. Drescher offered no equivalent to a Short Term Memory in his system.

By convention an input token incorporated into a sign will be automatically dereferenced to its external form from the internally represented symbolic form whenever it is displayed or printed. Sign conjunctions may also incorporate other symbolic information contained within the SRS/E system. So a sign conjunction may include the symbolic name for another sign (from $\mathcal{S}$). Similarly actions (from $\mathcal{R}$) may be included. Thus past actions by the system are available for inclusion into the "s1" conjunction. μ-Hypothesis activation (from $\mathcal{H}$) may also be recorded in a sign, by including the symbolic name of the hypothesis (to be described in a later section). The inclusion of the hypothesis form into the sign conjunction may give the system limited access to its own operation and hence the possibility of predicting, seeking as a goal, and creating hypotheses about aspects of its own learning behaviour. The ramifications of this ability are beyond the experimental investigations of SRS/E presented here. This construct is broadly equivalent to Mott's proposal for an *internal kernel* and Drescher's notion of a *synthetic item*, but more concise and manageable than the latter as only the symbolic name is required. SRS/E does not, however, at present have any explicit support for the notion of *object permanence*.

The `sign_conjunction` may be more concisely defined as:

$$s \in \mathcal{S}^* \ \textbf{iff} \ conjunction_{n=1}^{k}(\chi_n^s) \qquad \text{(eqn. 4-3)}$$

where k gives the number of terms in the conjunction. Each of the items χ_n^s may substitute for one of four forms:

$$\chi_n^s \equiv \varkappa \in \mathcal{X}^* \qquad \text{form 1}$$

or

$$\sim \chi_n^s \equiv \varkappa \notin \mathcal{X}^* \qquad \text{form 2}$$

or

$$\chi_n^{s@-t} \equiv \varkappa \in \mathcal{X}^{*@\text{-}t} \qquad \text{form 3}$$

or

$$\sim \chi_n^{s@-t} \equiv \varkappa \notin \mathcal{X}^{*@\text{-}t} \qquad \text{form 4}$$

allowing for the presence of symbol of type $\varkappa$ (form 1), the absence of symbol of type $\varkappa$ (form 2), the recorded presence of symbol of type $\varkappa$ at time (now-t) in the past (form 3) and the recorded absence of symbol of type $\varkappa$ in the past (form 4). In these forms the symbol $\varkappa$ (and hence $\mathcal{X}$) may substitute for elements from any of the lists $\mathcal{I}$, $\mathcal{S}$, $\mathcal{R}$ or $\mathcal{H}$.

The sign definition adopted in SRS/E has no *don't care* ("#") representation of the form employed in classifier systems. If a symbol is not explicitly included its condition is taken as irrelevant. This is generally consistent with Popper's view that an 'experiment' should define all its relevant preconditions, but exclude all those inconsequential to its outcome. This representation is not as concise where small, bounded sets of features are to be considered, but offers significant advantages where small subsets of a very large feature set are to be represented and where past values of features are to be included. Many other representational schemes have been proposed to enable machine learning systems to represent left hand side preconditions completely or conveniently. In particular, Michalski (1980) describes a condition form for the VL_{21} logic system that includes enumeration, variabilisation and hierarchical descriptions; but not past events.

In the SRS/E implementation the Sign List is held as an indexed list of sign elements. The index is used to create the sign identifier (thus: "Snnnn", where nnnn is the index number). This designation for a sign symbol appears in the log and analysis information from the experimental runs of SRS/E. Individual conjuncts in a `sign_conjunction` definition are recorded as a triple: conjunct identifier, a negation flag, and time offset. In the current implementation they are recorded in a canonical form for efficient access. Also in the current implementation negation is indicated by recording the conjunct identifier (for instance `token_identifier`) with a negative value. Attempts to create a new sign that duplicates an existing sign are rejected by SRS/E.

4.3.2. Other Sign List Values

Each element of the Sign List is assigned a unique `sign_identifier`, as described, and each sign has associated with it `sign_first_seen`, `sign_last_seen`, `sign_count` and `sign_activation_trace` values. The derivation and use of each of these mirrors the derivation and use described for the equivalent Input Token values. Sign probability, `sign_prob`, is calculated in an analogous manner to `token_prob`:

$$\texttt{sign\_prob} \leftarrow \frac{\texttt{sign\_count}}{\texttt{now - sign\_first\_seen + 1}} \qquad \text{(eqn. 4-4)}$$

An additional measure, `raw_sign_prob`, may be derived from the individual probability (*p*) values of the component parts of the sign conjunction:

$$\texttt{raw\_sign\_prob} \leftarrow \prod_{n=1}^{k}(p(\chi_n^s)) \qquad \text{(eqn. 4-5)}$$

Where `sign_prob` >> `raw_sign_prob` the SRS/E algorithm may use this as an indication that the sign conjunction is a significant combination of component parts, and not just a combination of random or "occult" occurrences.

4.4. Actions and the Response List

The Response List, $\mathcal{R}$, records the basic actions available to the animat. For any SRS/E controlled animat, the originator "registers" a list of basic actions and their associated costs as part of the initial ethogram definition. Actions will be required to serve the needs of both the innate behavioural and the learning components of the SRS/E system, though the same actions may well be adequate for both purposes. In SRS/E the actions defined in $\mathcal{R}$ serve as instructions or commands to the actuation sub-system, whether physical or simulated. Selection and description of the actions in $\mathcal{R}$ are an integral part of any experimental run discussed in chapter six. SRS/E supports both simple (molecular) and compound (molar) actions. A compound action is one built from the concatenation of two or more simple actions, as described by postulate A3. Compound actions run to completion once initiated. This definition of compound action is therefore distinct from Drescher's definition of a *composite action*, which may be seen as an intermediate stage between the SRS/E compound action and the Dynamic Policy Map.

In the current implementation each action is held as an element in the indexed list $\mathcal{R}$. Individual actions are registered into the list before the start of each experimental run. Additional entries may be registered into the list at any time, to implement a *maturation* strategy, for instance. On each execution cycle SRS/E will select a single action from $\mathcal{R}$ to be reified (derived from postulate A1) and delivered to the actuator sub-system. The reified action is placed on the $\mathcal{R}^*$ list for the cycle in which it is active. Output actions take the form of an ASCII string (entered at the time of registration) to be interpreted by the actuator system as an instruction to perform some defined activity. Trace and log information arising from the use of SRS/E will automatically dereference the action index to this string for ease and clarity of analysis, as with Input Token List entries.

4.4.1. Response List Values

In addition to the anonymous internal symbolic value, `response_identifier` and the external string representation of the action, `response_string` stored with each action in $\mathcal{R}$, the SRS/E algorithm records `response_cost`, an estimate of the effort that will be expended whenever that action is taken (the action cost, from postulate A2). This is the estimate provided by the originator at the time the action is registered. It may reflect the energy required to perform the action, a notional amount of resource depleted by the action, or the time taken to complete the simple or compound action, or some combination of these and other attributes. This is broadly in keeping with Tolman's (Tolman, 1932, Ch. 7) observations that rats generally choose paths through experimental mazes that minimise delay or effort.

On a practical note this value also provides the Dynamic Policy Map generation algorithm a metric by which to evaluate the appropriateness of alternative paths through the map. The originator is required to specify `response_cost` values of unity or greater, and that these values be proportioned according to the relative effort across all actions in $\mathcal{R}$. The `response_activation_trace` maintains a transient record of past actions (a record of $\mathcal{R}^*$), computed as for `token_activation_trace` and `sign_activation_trace`.

4.5. Innate Activity and the Behaviour List

The *Behaviour List* $\mathcal{B}$ defines the innate behaviours for the animat. This definition is an essential part of the ethogram, and built into the animat at the time of its definition by the originator. Such behaviours will react to situations, events, and changes in the environment as prescribed by the originator. In the main these activities will be mediated and modified by internally generated and detected needs, drives or motivations differentially selecting or inhibiting aspects of *innate behaviour* patterns. Innate behaviours need not be fixed over the life-cycle of the animat and may vary according to a *maturation* schedule or *imprinting* regime. This section does not intend to revisit the mechanisms by which behaviours are formed and selected, nor to further consider the arguments over which of the many proposed strategies most effectively or closely model observed natural behaviours. It will, however, be primarily concerned with how the overt behaviour of the animat will be apportioned between the innate and learned parts of the mechanism.

4.5.1. Behaviour List Structure and Selection

The Behaviour List is a notional list of condition-action pairs (`condition` $\in \mathcal{S} \rightarrow$ `action` $\in \mathcal{R}$), fully in the tradition of the stimulus-response behaviourist camp. At each execution cycle every element b of $\mathcal{B}$ is evaluated against $\mathcal{S}^*$, and a list of applicable candidate actions, $\mathcal{B}^*$, formulated. The selection of behaviours on each cycle is thus made based on the evidence for their applicability. To achieve the required balance of innate and learned behaviours the Behaviour List will be considered to be in two parts. The first part, $\mathcal{B}^r$, lists condition-action pairs from which action candidates will be selected ($\mathcal{B}^{r*}$). This part of the list realises the *primary behaviours* of postulate B2. The second part, $\mathcal{B}^g$, lists condition-action pairs determining which, if any, goals the animat should pursue given the prevailing circumstances. This second part of the list realises the *goal setting behaviours* of postulate B3. During each execution cycle several possible actions, and several goals could be applicable. SRS/E makes its selection from $\mathcal{B}^{r*}$ and $\mathcal{B}^{g*}$ on a priority basis.

Each potential innate behaviour in the animat is assigned a priority by the originator, which is initially set within the ethogram according to its significance.

Thus in an animal simulation, predator avoidance might be assigned a high priority, and therefore be made manifest whenever the conditions that indicate the approach or presence of a predator. Other behaviours, those initiated by, say, the onset of hunger (detected, perhaps, by lowered "blood sugar levels") having a lower overall priority and so being interrupted by the avoidance behaviour. SRS/E must also adjudicate between innate and goal seeking behaviours, those derived from the Dynamic Policy Map. To achieve this, elements of $\mathcal{B}^g$ (and so $\mathcal{B}^{g*}$) are also assigned a priority in the ethogram. At each cycle SRS/E will either select the highest priority element from $\mathcal{B}^{r*}$, if this priority is higher than that for the highest priority element from $\mathcal{B}^{g*}$. Otherwise a Dynamic Policy Map will be created, or the existing one used, to generate a behaviour from stored μ-hypotheses.

Where none of the defined innate behaviours has an effective priority, it is inappropriate for the animat to pursue any of those behaviours. So, if it is not threatened, hungry, thirsty, tired or dirty, etc., then there is little to be gained by fleeing, eating, drinking, sleeping or preening, etc., just because one of these behaviours is slightly less irrelevant than the others. Therefore the SRS/E algorithm places a lower bound, the *basal level threshold* (ε), on behaviour activation, below which none of the behaviours defined in $\mathcal{B}$ will be selected. Yet the animat is expected to perform some activity on each cycle. Where no innate behaviour or goal behaviour is active the animat performs exploratory actions selected from $\mathcal{R}$. These implement the third, and mandatory class of innate behaviour pattern, the Default (exploratory) Behaviours (realising postulate B4). The learning mechanism is still actively monitoring the actions taken and their outcomes and learning continues during these periods of apparently undirected activity.

The Behaviour List as defined for the present version of SRS/E places restrictions on what may be effectively represented by the originator. It is adequate to generate the reflexive behaviours described for ALP. Any scheme by which behaviours are controlled through the presence of only binary releasers provides little useful analogue with the natural world, and gives rise to a range of difficulties in providing a useful simulation of innate behaviour. The default exploratory ("trial and error") behaviour is present in SRS/E as an inherent component of the system and requires no additional intervention by the originator. For the purposes of the experimental regimes to be described in chapter six the experimenter is able to activate goals externally.

4.5.2. Behaviour List Values

In addition to the `condition` and `action` values, each element of $\mathcal{B}$ has associated with it the value `behaviour_priority`, which defines the pre-assigned importance of the behavioural component. There is a fundamental difference between actions on the $\mathcal{B}^r$ and $\mathcal{B}^g$ parts of the Behaviour List. In the former case the action is selected from those available on the Response List. In the latter case the "action" taken is to place a sign onto the Goal List, or to manipulate the priority of the goal because circumstances have altered.

Potential exists to extend the $\mathcal{B}^r$ part of SRS/E to respond to a conventional external reward schedule. A separate reinforcement strategy may be put in place to re-prioritise elements of the Behaviour List relative to desirable outcomes, either employing a straightforward immediate reward mechanism or some variant of the *Q*-learning or *bucket-brigade algorithms*.

4.6. Goals and the Goal List

The *Goal List* is a sub-set of the Sign List ($\mathcal{G} \subseteq \mathcal{S}$). Any sign, whether created by the originator or formulated during the learning process, may be designated as a goal state (`goal_sign`). The structure of the SRS/E sign offers a single representational type which provides (1) a symbolic name, such that the goal can be conveniently identified internally within the system; (2) a description of what is relevant to the definition of the goal (and so what is not relevant); and (3) a test enabling the system to recognise when the goal has been achieved. Signs are attached to the Goal List under the control of the Innate Behaviour List ($\mathcal{B}^{g*}$), as previously described (postulate B3). The goal sign having the highest associated priority (`goal_priority`) is designated g^1 and so forms the seed to build the current Dynamic Policy Map. This is the *top-goal*. SRS/E supports many signs on the Goal List, after the top-goal these are designated g^2, g^3 and so on, ordered according to their given priority.

Goals are deemed *satisfied* when they appear on $\mathcal{G}^*$ (and so $\mathcal{S}^*$), realising postulate G3. The SRS/E algorithm automatically cancels satisfied goals by removing them from $\mathcal{G}$, and remaining goals on the Goal List are moved up the list automatically. As a consequence of this the Dynamic Policy Map is recomputed with the new seed and the observed behaviour of the animat changes accordingly. The change in behaviour is in effect instantaneous, and may lead to a completely different set of responses being employed by the animat in apparently identical circumstances. This is a significant departure from the reinforcement and *Q*-learning approach, where a single goal is repeatedly sought and a network of paths (a graph) constructed, dedicated to achieving the designated goal. When the Goal List becomes empty, use of the Dynamic Policy Map as a behaviour generator ceases. Until a new goal of sufficient priority is again placed on $\mathcal{G}$ observable behaviour reverts to innate actions drawn from the Innate Behaviour List $\mathcal{B}^{r*}$ or default behaviour mechanisms.

Under these circumstances the originator bears some responsibility for ensuring the stability of the Goal List ordering. SRS/E builds the DPM according to the top-goal g^1. It may be that $\mathcal{B}^{r*}$ gives rise to two goals of very similar priority, because, for instance, they are derived from sensors currently giving signals of equivalent significance. Under these circumstances the priority of the multiple goals may be unstable, swapping between the alternatives. The DPM is automatically recomputed at each priority swap causing changes or reversals of observed behaviour leading, in turn, to the inability of the animat to reach any of the enabled goal states. This is equivalent to the problem faced by any of the *Action Selection Mechanisms* (ASM) described earlier, where each must ensure that coherent patterns of behaviour are established to meet the needs of the animat.

4.7. The Hypothesis List

The Hypothesis List is the primary repository of learned knowledge within the SRS/E algorithm. Each element of the list, a *μ-hypothesis*, encapsulates a small, well-formulated, identifiable and verifiable fragment of information. A μ-hypotheses is not an unequivocal statement about the animat or its environment, but is an assertion about the nature of things - it may be true or it may be false. A μ-hypotheses may be partially complete and so true in some proportion of instances in which it is applicable. Every μ-hypothesis is an independent observation. SRS/E supports the notion of competing hypotheses, several hypotheses that share identical pre-conditions or which share identical conclusions. SRS/E accepts mutually inconsistent μ-hypotheses, to be resolved following corroboration[22]. SRS/E does not allow the installation of duplicate copies of identical μ-hypothesis.

The originator is, of course, at liberty to incorporate into the ethogram or controlling algorithm whatever consistency checking and verification mechanisms he or she considers appropriate. To do so takes the construction of the animat controller back to the realms increasingly referred to as traditional AI (Cliff, 1994) or *GOFAI* (Good Old Fashioned Artificial Intelligence, Boden, 1994). This is a valid approach, but not the one adopted here, and moves the animat definition towards the category (3) intelligence of chapter one. In SRS/E ambiguity is resolved by application and testing of the μ-hypotheses in the form of μ-experiments, which are conducted by the SRS/E system whenever the opportunity arises to do so. In turn, μ-experiments take the form of making verifiable predictions about the perceivable state of the animat or its environment at some defined time in the future.

All μ-hypotheses in SRS/E take the form of a triplet of component parts:

$$\text{Sign1} + \text{Response} \rightarrow \text{Sign2}^{@+t} \qquad \text{(eqn. 4-6)}$$

The first sign (Sign1 or just "s1") provides a context in which the performance of the action (Response or just "r1") is hypothesised to result in the appearance of the second sign (Sign2 or "s2") some specified time in the future (at '@' the predicted time, +*t* cycles in the future). The signs "s1" and "s2" are drawn from $\mathcal{S}$, the response "r1" from $\mathcal{R}$. Response "r1" is the action to be taken on this cycle, "s1" is the current value of the context sign. However "s1" may include token values drawn from the various activation traces, and so inherently defines a temporal as well as a spatial context. In Tolman's terms, "s2" is set as an expectancy whenever "s1" and "r1" are present. This expectancy relationship is the basis of the means-

[22]Or, if the animat is in a genuinely inconsistent environment, or in one which is unresolvably ambiguous, to remain inconsistent in perpetuity. Vershure and Pfeifer (1993) develop these issues further.

ends capability of SRS/E. If "s2" is an end, or goal, to be achieved, then “s1” and “r1” provide a means of achieving that end. In considering any μ-hypotheses with “s2” as its desired end, the corresponding “s1”, if it is not currently active and so available, may become an end, or sub-goal, in its own right. Developing a cognitive map of *means-ends-readiness* from many individual expectancies was a central component of Tolman’s expectancy theory. *Means-Ends Analysis* has developed into a cornerstone concept in traditional Artificial Intelligence from its introduction by Newell and Simon (1972) in the form of the *General Problem Solver* (GPS).

In a perfect μ-hypothesis “s1” defines exactly those conditions under which the response “r1” leads to the appearance of “s2” at the designated time. In an incompletely specified μ-hypotheses the relationship will hold on some occasions, but not others. A μ-hypothesis created as the result of an *occult occurrence* should hold very rarely (specifically, at a frequency of occurrence commensurate with the computed raw probability derived from its component parts). The evidence for *superstitious learning* was reviewed earlier. The conditions under which the μ-experiment may be performed occur whenever “s1” and “r1” are on their respective active lists ($\mathcal{S}^*$ and $\mathcal{R}^*$) at *t*=now, regardless of whether or not “r1” had been actively selected to achieve “s2”. Drescher (1991) refers to the latter case as *implicit activation*.

4.7.1. Other Hypothesis List Values

As with other list types, SRS/E μ-hypotheses have associated with them a number of values. These values record corroborative evidence about each μ-hypothesis and retain information used by the three main processes involved in the management of μ-hypotheses. These processes are: (1) μ-hypothesis corroboration and reinforcement (realising postulates H3 and H4); (2) building the Dynamic Policy Map (realising postulates P1 and P2); and (3) μ-hypothesis list maintenance (realising postulates H6 and H7). Some of the list element values associated with each μ-hypothesis are described next, and the three main processes and the μ-hypothesis values associated with them in the sections that follow. As each of the three processes are intimately interrelated, the order of these sections is somewhat arbitrary chosen.

Each μ-hypothesis on the Hypothesis List is assigned a unique `hypo_identifier`, created from the list index number. Index numbers are created in sequential order, and so indicate the relative age of the μ-hypothesis. The designation "Hnnnn" appears in the output log and analysis information, where nnnn is the list index number. The values `hypo_first_seen` and `hypo_last_seen` respectively record the cycle on which the μ-hypothesis was created and the most recent cycle on which the μ-hypothesis was active. A μ-hypothesis is defined as active when the following conditions are met on any given execution cycle:

$$\boldsymbol{h} \in \mathcal{H}^* \textbf{ iff } \mathrm{s1}(\boldsymbol{h}) \in \mathcal{S}^* \textbf{ AND } \mathrm{r1}(\boldsymbol{h}) \in \mathcal{R}^* \qquad \text{(eqn. 4-7)}$$

These conditions define when a μ-hypothesis will perform a μ-experiment by making a verifiable prediction. The value `hypo_activation_trace` records the most recent activations for the μ-hypothesis. The value `time_shift` records the number of cycles between an activation of a μ-hypothesis and the time that the "s2" sign is predicted to occur. The derived value `hypo_age` indicates the number of cycles elapsed since the μ-hypothesis was created. It is calculated from `hypo_first_seen` and the system variable "now".

The remaining values associated with each Hypothesis List entry may be characterised into serving one of three purposes. (1) Corroborative values recording the performance of the predictive ability of a μ-hypothesis. These values reflect the confidence the system may place in the effectiveness of the μ-hypothesis when building the Dynamic Policy Map, and in calculating when to modify or delete individual μ-hypotheses. These values broadly reflect the notion of *schema confidence weight* adopted by Becker and Mott. (2) Values computed, and re-computed, each time the Dynamic Policy Map is prepared. These values provide the action selection mechanism with the basis to determine which μ-hypothesis (and hence which action "r1") should be passed to the actuation sub-system during goal seeking behaviour. (3) Administrative values, recording information relevant to the creation and subsequent modification of individual μ-hypotheses. Major section headings will now be given over to the discussion of these values, reflecting their importance to the operation of the SRS/E algorithm.

4.8. Corroborating μ-Hypotheses, Predictions and the Prediction List

Every time a μ-hypotheses is activated it will perform a μ-experiment and so make a prediction, which will be verified on a later execution cycle. Each prediction is placed on the *Prediction List*, $\mathcal{P}$. As predictions are all of the form where a known sign is expected at a known time, the validation process is a straightforward matter of matching the elements of $\mathcal{P}$ which were predicted for the current execution cycle against the active Sign List $\mathcal{S}^*$. Alternative interpretations are available as to how "credit" for a correct or "debit" for an incorrect prediction should be assigned to the individual μ-hypotheses responsible for the prediction. These alternatives are reflected in the corroboration (H3) and reinforcement (H4) postulates. SRS/E maintains four values for each μ-hypotheses for this purpose.

Following Popper's notion that it is the absolute frequency of outcome that provides the appropriate measure of a hypothesis, the values `hypo_cpos` (cumulative positive, `cpos`) and `hypo_cneg` (cumulative negative, `cneg`) record the number of successful and unsuccessful predictions respectively. Specifically:

$$\texttt{cpos} \leftarrow \texttt{cpos} + 1 \textbf{ iff } \texttt{s2}(\boldsymbol{h})^{@t=\mathrm{pred}} \in \texttt{predicted\_sign}(\mathcal{P})^{@t=\mathrm{pred}} \qquad \text{(eqn. 4-8)}$$

$$\texttt{cneg} \leftarrow \texttt{cneg} + 1 \textbf{ iff } \texttt{s2}(\boldsymbol{h})^{@t=\mathrm{pred}} \notin \texttt{predicted\_sign}(\mathcal{P})^{@t=\mathrm{pred}} \qquad \text{(eqn. 4-9)}$$

These two equations compare predictions made at some point in the past (t=pred) to the appearance of actual signs at that predicted time. These two measures reflect the overall effectiveness of the μ-hypothesis over its span from the point of creation (the execution cycle recorded in `hypo_first_seen`), to the current execution cycle (less any predictions made, but not yet verified). The overall probability that the expectation defined by the μ-hypothesis will hold is therefore defined by:

$$\texttt{hypo\_prob} \leftarrow \frac{\texttt{cpos}}{\texttt{cpos} + \texttt{cneg}} \qquad \text{(eqn. 4-10)}$$

This is the corroboration measure (Ch of postulate H3). By definition every μ-hypothesis is assumed to represent a successful prediction at the time of its creation. This assumption is considered reasonable when using the pattern extraction creation process described later, even though the μ-hypothesis may subsequently be determined to denote an *occult occurrence*. This initial fillip to a new μ-hypothesis' confidence value will be referred to as the *creation bonus*.

In a changeable environment the validity of any given μ-hypothesis may also change with time. To reflect this the value hypo_bpos (bpos) is updated according to a discounting factor, thereby giving precedence to the effects of recent activations at the expense of those further in the past, specifically:

$$\texttt{bpos} \leftarrow \texttt{bpos} - \alpha(\texttt{bpos} - 1) \textbf{ iff } \texttt{s2}(\mathcal{h})^{@t=\text{pred}} \in \texttt{predicted\_sign}(\mathcal{P})^{@t=\text{pred}}$$

(eqn. 4-11)

or

$$\texttt{bpos} \leftarrow \texttt{bpos} - \beta(\texttt{bpos}) \textbf{ iff } \texttt{s2}(\mathcal{h})^{@t=\text{pred}} \notin \texttt{predicted\_sign}(\mathcal{P})^{@t=\text{pred}}$$

(eqn. 4-12)

otherwise

bpos unchanged

where:

α is the positive *reinforcement rate*, $(0 \leq \alpha \leq 1)$
and
β is the negative *extinction rate*, $(0 \leq \beta \leq 1)$

This implements the *reinforcement measure* (Rh of postulate H4). Long sequences of successful predictions for a single μ-hypothesis will asymptotically tend its bpos values to 1.0, long sequences of failed predictions will similarly tend bpos values towards 0.0. This notion of an asymptotic *negatively accelerating curve* is ubiquitous throughout the conditioning and behaviourist literature, and forms the basis of MacCorquodale and Meehl's (1954, p. 237) *strength of expectancy* measure. This procedure is similar to those used in most recent reinforcement and the *Q*-learning mechanisms.

The last value in this group is `recency`, which specifically records the outcome of the most recently completed prediction for each μ-hypothesis. The `recency` measure represents an alternative approach to Drescher's modelling of *object permanence*. The `recency` value remains asserted for any individual μ-hypothesis after a valid prediction about "s2" is detected. It is cleared when the prediction next fails. It acts as one form of event memory. Unlike Drescher's system SRS/E contains no inherent mechanism supporting the representation or manipulation of a "physical object".

The different measures `cpos`, `cneg`, `bpos` and `recency` serve different purposes in the generation of the Dynamic Policy Map (cost estimation) and in the management of the Hypothesis List (differentiation and deletion of ineffective μ-hypotheses). These differently computed values may reflect different views of the predictive effectiveness of μ-hypotheses. SRS/E may represent permanent (`hypo_prob`), semi-permanent or recurring (`bpos`), and transient (`recency`) phenomena. In this context the term "permanent" may equally be applied to an immutable physical law as to any phenomena that remains consistently predictable throughout the lifetime of the animat. For example, an animal, or animat learning to seek nourishment may locate a source that is habitually available, which may reliably be returned to. Equally a source of nourishment may be identified, which only comprises a finite quantity of sustenance. Finally the creature may happen across a single item of nourishment, which once consumed is finished. No second order effects are proposed for SRS/E to further classify individual μ-hypotheses into these various categories based on longevity of the phenomenon underlying the prediction. Such a strategy might properly be included in later implementations.

4.8.1. Prediction List Element Values

Each element of the list is created from the "s2" of any activated μ-hypothesis. Each element retains only three items, `predicting_hypo`, the identity of the μ-hypothesis responsible for the prediction, `predicted_sign` and `predicted_time`, the sign expected and the execution cycle on which it is predicted to occur. Elements of $\mathcal{P}$ are deleted as soon as the prediction they define has been verified against $\mathcal{S}^*$. As each prediction is held separately, any μ-hypothesis may have several predictions waiting for confirmation (as each μ-hypothesis may make at most one prediction on each execution cycle this is limited by the number of cycles between now and t^{pred}). There may equally be more than one prediction of a given sign for each future execution cycle, as many different μ-hypotheses may predict the same outcome.

4.9. The Dynamic Policy Map (DPM)

Whenever $\mathcal{B}^{g*}$ is not empty and the priority of the top-goal is greater than that for the highest priority candidate action from $\mathcal{B}^{r*}$ the SRS/E algorithm will attempt to construct a *Dynamic Policy Map* (DPM, after definition P0) from knowledge accumulated in the Hypothesis List. The effect of the Dynamic Policy Map is to categorise entries in the Sign and Hypothesis Lists according to an estimate of their effectiveness as being on a path of actions that will lead to the satisfaction of the top-goal. The SRS/E algorithm builds the Dynamic Policy Map by the process of *spreading activation*, based on repeated application of the *spreading valence* postulate (postulate P2). Individual μ-hypotheses, $\mathcal{h}$, which lead directly to the top-goal, g^1, are selected (where $s2(\mathcal{h}) = g^1$). This selection and binding process will be referred to as "valencing", following Tolman's use of the term. Context signs in these μ-hypotheses may then act as "sub-goals", allowing another sub-set of the Hypothesis List to be incorporated into the Dynamic Policy Map. The SRS/E algorithm stops building the DPM once all the entries in the Hypothesis List have been incorporated or there are no more μ-hypotheses that may be chained in this way. Signs and μ-hypotheses incorporated in the DPM are termed *sub-valenced*. The *valence level* of each μ-hypothesis incorporated into the DPM indicates the estimated minimum number of sub-goals that must be traversed to reach the designated goal sign.

The Dynamic Policy Map may be considered as a graph structure. Signs from the Sign List act as nodes, μ-hypotheses from the Hypothesis List the arcs. One special sign, the top-goal, acts as the seed or start point for the spreading activation process to create the graph. Development proceeds on a breadth-first basis, μ-hypotheses at each valence level are selected at the same step in the spreading activation process. This is implemented as a variant of the well-established *graph-search* procedure (Nilsson, 1980, Ch. 2).

Every arc has associated with it a cost estimate. An arc is traversed by selecting the action, “r1”, from the μ-hypothesis. The true cost of traversing the arc is given by the response_cost value assigned to each action (the *action cost* of postulate A2). This is simply the “effort” expended in taking the action, as provided in the Response List. The estimated cost of traversing the arc to a node at the next valence level takes into account the true cost of the action and the relative effectiveness of the μ-hypothesis in actually achieving its expected outcome, based on past experience. This cost_estimate for each μ-hypothesis is prepared from:

$$\texttt{cost\_estimate} \leftarrow \frac{\texttt{response\_cost}}{\texttt{hypothesis\_confidence}} \qquad \text{(eqn. 4-13)}$$

This realises the Cost Estimate postulate (P3). The hypothesis_confidence value is in turn prepared from:

$$\begin{aligned}\texttt{hypothesis\_confidence} \leftarrow\ & (\texttt{hypo\_prob} * \gamma^1) + \\ & (\texttt{hypo\_bpos} * \gamma^2) + \\ & (\texttt{recency} * \gamma^3) + \\ & (|\texttt{oscill}| * \gamma^4)\end{aligned} \qquad \text{(eqn. 4-14)}$$

where:

$$(\gamma^1 + \gamma^2 + \gamma^3 + \gamma^4) = 1$$

and

$$(0 \leq \texttt{oscill} \leq 1)$$

The hypo_prob, hypo_bpos and recency values are those previously described. The oscill component is an essentially random factor designed to perturb the path selection process. This has the dual effect of adding an element of uncertainty to encourage the use of other μ-hypotheses, and to allow the system to escape from potential behavioural loops. The effect of this parameter is intended to reflect the use that Hull describes for his *oscillatory* component, ${}_{S}O_{R}$, from which the current name is derived. In implementation the value of oscill is derived from the pseudo-random number sequence generator (and so is not really “oscillatory” at all). While superficially similar in effect to Sutton's (1991) *exploration bonus* in *Dyna-Q+*, the balance of goal-seeking behaviour to exploration is ultimately achieved in a quite dissimilar manner in SRS/E. This is considered in detail in chapter six.

The cost estimate for each arc, ignoring the `oscill` component, reflects the given action cost scaled by the recorded probability that the causal relationship described by the μ-hypothesis is indeed responsible for the transition. Assuming for the moment that the *selection factor* γ^1 has been set to one (and so γ^2, γ^3 and γ^4 are all zero[23]) the `cost_estimate` for the arc is equal to the true (given) cost of the action "r1" when `hypo_prob` is at its maximum value. This condition only holds when the μ-hypothesis has never failed. Where a μ-hypothesis has been created as result of an occult occurrence the value of `hypo_prob` will tend to zero, and so the value of cost_estimate will tend toward infinity. The `hypo_prob` value will never reach zero, due to the initial *creation bonus*. Increasing the relative contribution of γ^2 (at the expense of γ^1) biases cost estimates toward more recent experiences. Values for the factors γ^1, γ^2, γ^3 and γ^4 are set by the experimenter before each experiential run, and are fixed for the duration of that run in the current implementation.

No account in the computation of the cost estimate is taken of the experience of the μ-hypothesis, as recorded in the `hypo_age` and `hypo_maturity` measures, in the current implementation. For the experiments described later the creation bonus serves to increase the likelihood that a new (and therefore inexperienced) μ-hypothesis will be selected and so appears to provide an adequate balance of new and old knowledge. A more sophisticated strategy may bias the estimate to more experienced μ-hypotheses where the importance or priority of the goal is high. Conversely newer, less experienced, μ-hypotheses may be favoured in *play* situations, where (apparently unimportant) goals are set for the explicit purpose of gaining experience and knowledge. Such considerations are left for future investigations.

[23] Note that these superscripts indicate the first γ, the second γ and so on; similarly g^1, g^2, etc.

4.9.1. Selecting actions from the DPM

Every μ-hypothesis implicated in the DPM is assigned a `policy_value`, the minimum sum of individual `cost_estimate` elements across all the arcs from the sign node associated with "s1" to the goal sign node g[1]. This is a realisation of Postulate P4. During the graph building process the `policy_value` associated with each node is updated if a lower cost route to that node is discovered. Figure 4-1 shows a printout from an experimental log showing a *valenced path*, the lowest (estimated) cost path from the current situation to the desired goal. It records the individual μ-hypotheses (e.g. "H119") selected from the graph, the individual cost contributions from `cost_estimate` ("cost") and the cumulative `policy_value` ("total") values as the valence levels are traversed. It starts with a node ("X2Y0", the printout has automatically dereferenced signs to external names) that is currently on the active Sign List $\mathcal{S}^*$, and so defines the μ-hypothesis ("H126") which will contribute the reified action ("U") in the current execution cycle.

```
H126 predicts X2Y1 from X2Y0 (active) after U (cost = 1.818182, total = 15.006273)
H117 predicts X3Y1 from X2Y1 after R (cost = 1.290323, total = 13.188091)
H119 predicts X4Y1 from X3Y1 after R (cost = 1.059603, total = 11.897769)
H120 predicts X5Y1 from X4Y1 after R (cost = 1.290323, total = 10.838166)
H4 predicts X6Y1 from X5Y1 after R (cost = 1.290323, total = 9.547844)
H5 predicts X7Y1 from X6Y1 after R (cost = 1.290323, total = 8.257522)
H6 predicts X8Y1 from X7Y1 after R (cost = 1.290323, total = 6.967199)
H8 predicts X8Y2 from X8Y1 after U (cost = 1.126761, total = 5.676877)
H9 predicts X8Y3 from X8Y2 after U (cost = 1.078894, total = 4.550116)
H10 predicts X8Y4 from X8Y3 after U (cost = 2.351558, total = 3.471222)
H11 predicts X8Y5 (goal) from X8Y4 after U (cost = 1.119664, total = 1.119664)
Valenced path in 11 steps, estimated cost 15.006273
```

Figure 4-1: Log Printout of a Valenced Path

It is important to note that the valence path printout is not a set of prescribed actions to be performed to reach to goal state, as would be the case in *STRIPS* (Fikes and Nilsson, 1971), but rather a sub-set of the total DPM. It is presented to provide the experimenter with information about the current state of the animat under investigation. The action selected may, or may not, lead to the expected sign at the lower valence level on the valence path. On the next execution cycle a new assessment of the environment is made, as indicated by a new $\mathcal{S}^*$.

The next action is selected from the DPM on the basis of the new S^*. It may be that the next action on the existing valence path is selected. However the new S^* may indicate that a shorter route has, through fortuitous circumstance, become available; equally only longer routes may now be available. In each eventuality the DPM acts essentially equivalently to the *policy map* in *reinforcement* and *Q-learning* algorithms, recommending the best course of action relative to the current circumstances and the goal sought.

There is a pathological case where no intersection between S^* and the DPM exists and so no action can be selected from the DPM. Under this circumstance the current algorithm selects an exploratory trial and error action at random. A more sophisticated variant of the algorithm might balance the return to exploratory activity with a “faith” that the action was perhaps successful, but that the expected outcome had not been properly detected. In this way the animat may continue along a previously computed valence path and avoid the potential disruption caused by deflecting to exploratory actions.

4.9.2. Recomputing the DPM

There are several circumstances where the SRS/E algorithm must recompute the Dynamic Policy Map. When the top-goal, g^1, is satisfied, the next highest priority goal becomes the top-goal, and a new DPM must be computed before another action may be selected. Similarly innate behaviours from the Behaviour List may alter the priorities of the Goal List (realising postulate B3), also precipitating a recalculation of the DPM. At each execution cycle many μ-hypotheses may have their values updated, reflecting predictions they made in the past. At any cycle new μ-hypotheses may be added to the Hypothesis List, or existing ones deleted from the list. Any of these changes can have profound effects on the best paths through the graph. On the other hand, recomputing the DPM is a cost overhead not to be ignored. The SRS/E algorithm must recompute the DPM if the goal changes, but the experimenter may control the sensitivity of SRS/E to changes in the Hypothesis List.

The system variable `rebuildpolicynet` is cleared each time the DPM is rebuilt. It is incremented by some quantity Δ each time the Hypothesis List changes, and by some (typically smaller) amount δ every time a μ-experiment prediction fails. Before each use of the DPM `rebuildpolicynet` is compared to the system constant `REBUILDPOLICYTRIP`, the DPM being recreated once this trip value is reached or exceeded by `rebuildpolicynet`. Apart from the effect these values have on the balance of resource utilisation by SRS/E on policy construction and other computational activities, they also have a profound effect on aspects of the animats observable behaviour. This effect is particularly apparent in the *dual path blocking* experiments described later. In the current implementation Δ and δ are selected such that the DPM is rebuilt following any change.

4.9.3. The DPM, A Worked Example

Figure 4-2 shows a graph generated from the model Hypothesis List shown embedded in the figure. For the purposes of this example a DPM comprising eight signs and 12 μ-hypotheses is created. In this instance the top-goal, g^1, is equated to sign number “S16”. Only three actions are available on the Response List, “A1”, “A2” and “A3” all with an actual cost of one. The third column shows some possible “cost estimate” values for the various μ-hypotheses following a period of behaviour. At each valence level in the graph the policy cost associated with each sign is the cumulative policy value of the lowest cost path through the graph to the chosen goal. Each arc is labelled with the μ-hypothesis responsible for the transition, with its action and associated cost estimate.

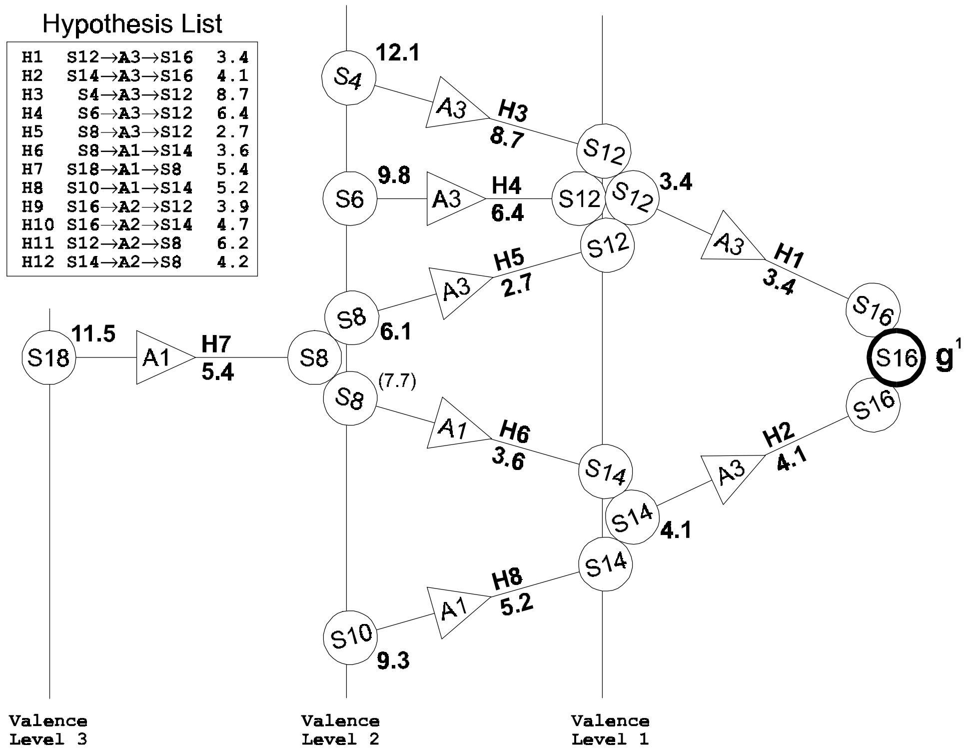


Figure 4-2: Model DPM Generated from Sample Hypothesis List

It may be that on the current execution cycle signs “S4” and “S18” are active and so on S^* (figure 4-3a). Policy cost for “S18” is lower than “S4”, so SRS/E selects action “A1”. The expectation is that “S8” will appear on S^* on the next execution cycle, and so action “A3” from μ-hypothesis “H5” would be selected. As a consequence these circumstances the `hypothesis_confidence` value of the successful μ-hypothesis “H7” would be strengthened, and that for the unsuccessful μ-hypothesis “H3” would be diminished (figure 4-3b). With “S8” on the active Sign List, SRS/E will choose the path described by “H5”, performing action “A3”, expecting sign “S12”. If this expectation is met, “H5” is strengthened, and action “A3” (from “H1”) will be selected on the next execution cycle; leading to goal satisfaction if that subsequent expectation is also satisfied.

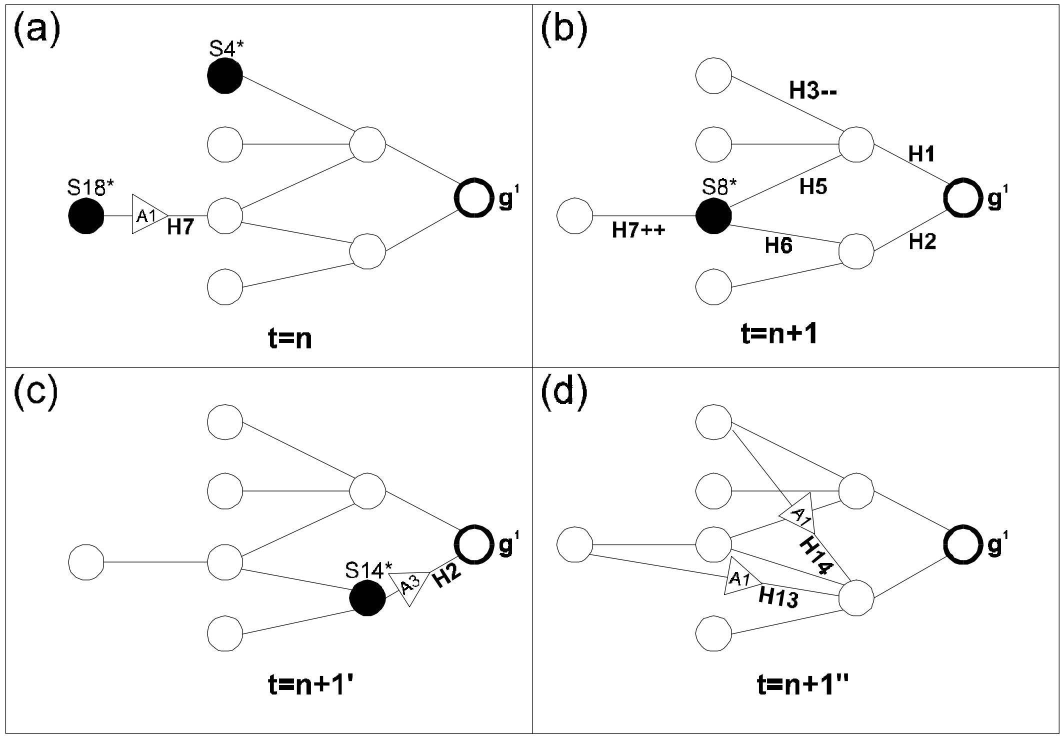


Figure 4-3: Various Outcomes for Model DPM

If, at the step indicated by figure 4-3b, the action “A3” did not lead to the expected sign “S12”, but instead “S8” remained on $\mathcal{S}^*$ then confidence in “H5” would be weakened. Eventually the cumulative cost of the path “H5”-“H1” would exceed that for “H6”-“H2”, at which point SRS/E would attempt action “A1” (from “H6”). Note that the confidence in “H6” was unaltered during the time “A3” actions were attempted, because it was not placed on $\mathcal{H}^*$ as its “r1” precondition was not matched and so it was not eligible to issue a prediction. The rate at which the estimated cost of any path rises under these circumstances is primarily controlled by the β extinction rate factor; though changes in estimated cost will not take effect until increments to δ (and Δ) cause the DPM to be recomputed.

What SRS/E hypothesises about the consequences of its actions in the environment, and what actually occurs may not hold true in practice. Considering again the situation described by figure 4-3a, it may be that rather than the expected activation of “S8”, sign “S14” is activated (figure 4-3c), either through some previously unknown path, or by a previously undetected event. On this execution cycle SRS/E would select the action “A3” associated with μ-hypothesis “H2”. If this expectation subsequently holds the top-goal would be achieved, and so removed from the Goal List. As a side effect of this unexpected transition SRS/E may create the new μ-hypotheses “H16:(S4→A1→S14)” and “H17:(S18→A1→S14)” (figure 4-3d), employing the mechanism of postulate H5-2.

Under the initial conditions described by figure 4-3a, the new paths of lower estimated cost offered by “H16” and “H17” may be considered in future instances in preference to either “H3” or “H7” originally available. Where they are due to a genuinely repeatable phenomenon the confidences of these new μ-hypotheses will be strengthened, leading to the adoption of the lower cost estimate path. Where the μ-hypotheses were created due to occult or unrepeatable circumstances the use of the new, apparently preferable, path will fall into disuse following a number of unsuccessful applications. The experimental procedure adopted in chapter six can give rise to this phenomenon (for instance, the effect shown in figure 6-10c), and it will be considered further.

The effects of recomputing the Dynamic Policy Map can completely alter the response of SRS/E to incoming tokens. Figure 4-4 shows an alternative computation of the DPM graph using the same Hypothesis List as Figure 4-2, but where the goal definition has changed from “S16” to “S8”. Note in particular that, although none of the cost estimates for the μ-hypotheses have changed, the response of the system to signs “S14” and “S12” is now completely different. This feature differentiates the behaviour SRS/E from the reaction of reinforcement and *Q*-learning systems in the manner highly reminiscent of Tolman’s arguments in favour of *expectancy theory* over stimulus-response theorising.

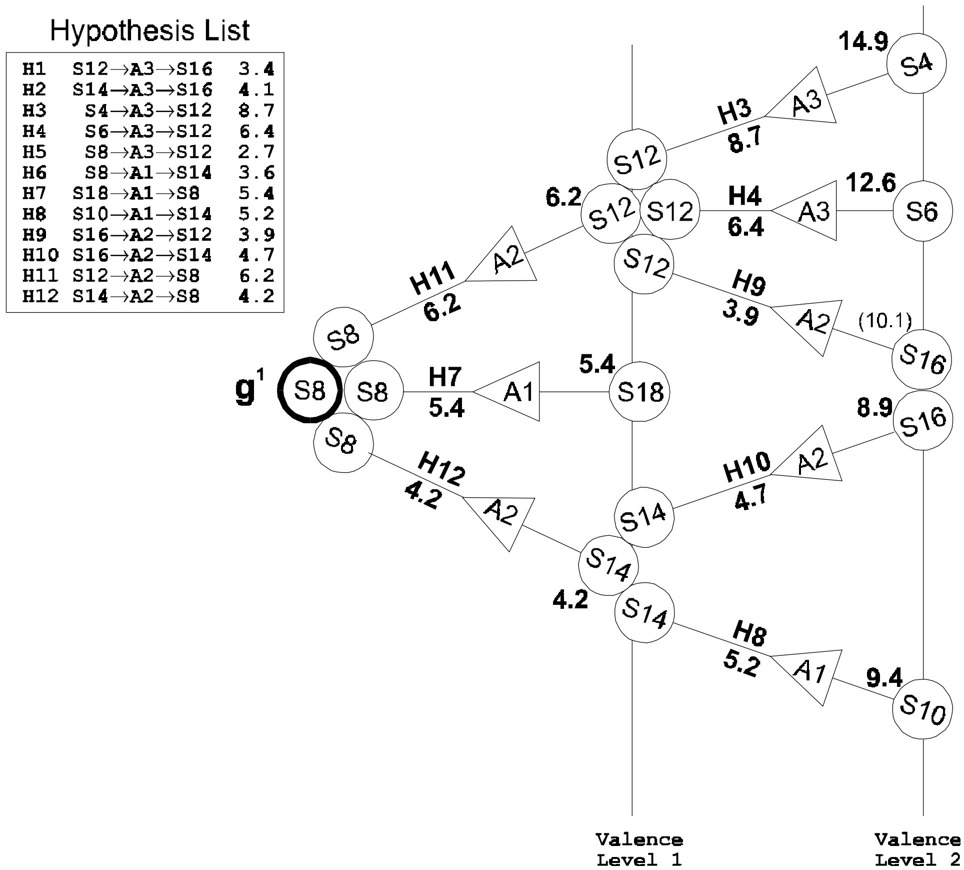


Figure 4-4: Model Graph Recomputed for Goal "S8"

4.9.4. Pursuing Alternative Goal Paths

The Dynamic Policy Map indicates the path with the currently most favourable estimated cost from an active sign state to the highest priority top-goal state. Actions are selected on the basis of this estimate. Consider the DPM graph shown in figure 4-5.

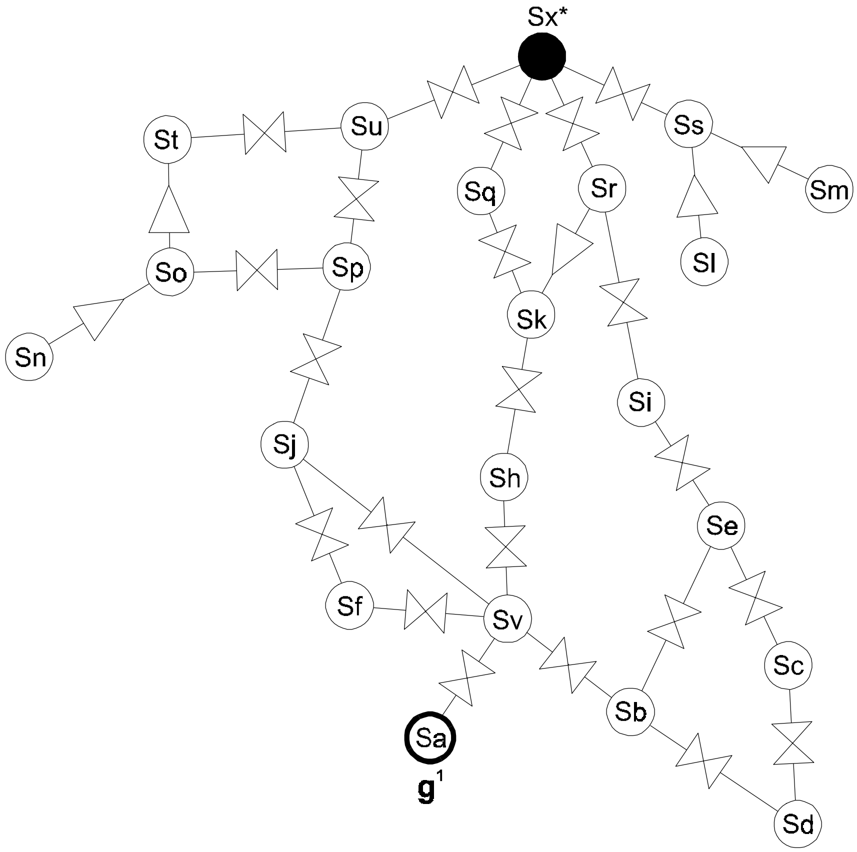


Figure 4-5: A Sample Dynamic Policy Map

The top-goal (g^1) is equated to "Sa", and the only active sign is "Sx*". Six distinct paths are available to the animat. These are summarised in table 4-2, with illustrative cost estimates. Individual signs are shown with letters, not sequence numbers, purely as a shorthand notation. The double arrow on an arc indicates a pair of μ-hypotheses, for instance a path is known both between "Sx" and "Su" and between "Su" and "Sx". No path is available through the loop formed by "Su"-"St"-"So"-"Sp" as no μ-hypothesis exists for the transition "St"-"So", as indicated by the unidirectional arrowhead.

Path	“Estimated Cost”
(1) Sx-Sq-Sk-Sh-Sv-Sa	18.4
(2) Sx-Sr-Sk-Sh-Sv-Sa	20.8
(3) Sx-Su-Sp-Sj-Sv-Sa	38.5
(4) Sx-Su-Sp-Sj-Sf-Sv-Sa	45.7
(5) Sx-Sr-Si-Se-Sb-Sv-Sa	67.9
(6) Sx-Sr-Si-Se-Sc-Sd-Sb-Sv-Sa	158.1

Table 4-2: Paths Through Figure 4-5 Graph

On the basis of the cost estimates shown, the animat will select the action “r1” associated with μ-hypothesis “Hxq” (indicating the transition from “Sx*” to “Sq”). If this expectation is met, the animat selects “Hqk”, and so on. Should this path succeed, then sign “Sa” will be removed from the Goal List, and path (1) will be strengthened. If, while at node “Sq”, the expectation described by “Hqk” failed, the cost of the remaining path “Sq*”-“Sk”-“Sh”-“Sv”-“Sa” would rise, due entirely to the increased estimate for “Hqk”. In practice under these circumstances, the increase in cost for a single expectation failure is relatively small and it may be that the estimated cost of the remaining path is still below that for any alternative, so that “r1” from “Hqk” will be tried again. Even if the remaining path would have a greater cost, if the effect of δ (the expectation failure policy rebuild increment) is small the DPM may not be rebuilt, and the policy decision will remain unaltered.

At some point, the cost estimate would come to exceed that for the next lower estimated cost path, “Sq*”-“Sx”-“Sr”-“Sk”-“Sh”-“Sv”-“Sa” in a recomputed DPM, and the action associated with “Hqx” would be selected. If this is also blocked at some point, the next lowest cost estimate path would be attempted, starting from the currently active node. Each time the cost estimates indicate a new path, following a DPM recomputation, a new solution path is tried. The frequency with which the DPM is recomputed determines how persistent the animat will appear to be in pursuing a blocked course of action.

Individuals with values of Δ and δ that are small relative to `REBUILDPOLICYTRIP` will persist with one course of action longer than individuals where these values are correspondingly larger. *Persistence of behaviour* may be an appropriate course of action. In the environment he describes, the probability of Mott's robot reaching the charger under the influence of the schema "<BRIGHT>S → <FORW>M ⇒ <ON-CHARGE>S" is very low. It is nevertheless the best option available, and a persistent individual animat that did not swap between other alternatives frequently would be advantaged. In other circumstances the ability to change to a potentially better solution path may be advantageous, where there is serious competition from other individuals for limited resources, for instance. No second order learning phenomena are currently implemented in the SRS/E algorithm to determine an appropriate balance between persistence and fickleness in selecting a solution path.

4.9.5. Pursuing a Goal to Extinction

In the situation where all possible paths to a top-goal are unobtainable, continued attempts at the goal become a threat to the animat's survival by locking out other behaviours. The goal must be forcibly abandoned, this is the *goal extinction point* (postulate G4). Goal extinction is achieved in the SRS/E algorithm by removing the unsatisfied top-goal, g^1, from $\mathcal{G}$. The animat would then be free to pursue the next highest priority goal as top-goal, or other behaviours if there are no further elements on $\mathcal{G}$. Extinction of behaviours has been widely observed experimentally (section 3.6.3). Extinction does not, however, appear as an abrupt abandonment of the behaviour. Instead the behaviour persists for a time (the "on-period"), then suspended briefly (the "off-period") before being resumed for another on-period. This alternation of apparently goal directed behaviour with periods of some other activity persists for a time, until the goal directed behaviour finally appears to be completely suppressed. The relative lengths of the "on" and "off-periods" change in a characteristic manner, the periods "on" shortening and the periods "off" lengthening.

During goal directed behaviour SRS/E always takes the best possible estimated path, there is no explicit exploration during this type of behaviour. SRS/E does not attempt to locate new paths, but instead applies its resources to achieving the goal using the best known path. At the end of the first "on" period behaviour reverts to default *trial and error* actions. This period has the effect of exploring for new paths through the graph. If the animat "stumbles" upon the solution and arrives at the goal it is satisfied in the normal way, and a new path is known for future use. Lengthening periods of exploration have the effect of widening the area of search in the graph space, increasing the likelihood of happening on a previously unknown path through the cognitive map and thereby reaching the top-goal[24]. The duration of the first “on-period” is determined from the initial cost-estimate of the best path in the graph. The *valence break point* (VBP, described by postulate P6), is set to some multiple of the initial lowest policy value cost estimate (`bestcost`) computed by the algorithm. This multiple is defined by the system constant `VALENCE_BREAK_POINT_FACTOR`, currently set to 10.

$$\texttt{VBP} \leftarrow \texttt{bestcost} * \texttt{VALENCE\_BREAK\_POINT\_FACTOR} \qquad \text{(eqn. 4-15)}$$

Thus in the example given by figure 4-5 (table 4-2), goal directed behaviour would continue until the estimated cost of the best available path exceeds a value of 184.0. The multiplier value is selected to give the animat ample opportunity to achieve the goal by direct use of the DPM, allowing a generous margin for failed expectations.

Once the *policy value* of the best path reaches the VBP value the goal is temporarily suppressed, and VBP is again multiplied by the valence break point factor (to 1840.0). On reaching each break point behaviour reverts to exploratory actions for a period determined by a `goal_recovery_rate` parameter, the *goal recovery mechanism*. Actions taken during this period are referred to as *unvalenced actions*, to distinguish them from purely trial-and-error exploratory activities. On the first suppression the goal recovery rate is high, and behaviour reverts to goal directed quickly after only a few unvalenced actions.

On reaching each subsequent valence break point the goal recovery rate is reduced (in the current implementation by a factor of two) and so the number of unvalenced actions during the off-period increases. Each time the blocked μ-hypothesis fails the estimated cost of the step increases at an exponential rate, and the time taken to

---

[24]Panic reactions may be an extreme form of this phenomena, wild or exaggerated actions being performed, possibly beyond the normal limits to physical well-being, in a final attempt to escape some intolerable condition. Indistinguishable behaviours may equally be part of the innate behavioural repertoire, unrelated to goal seeking.

reach the next VBP level decreases as a consequence. At some point the estimated cost of the path exceeds the *goal cancellation level*, Ω, and the unachievable top-goal is automatically deleted from the Goal List.

The extinction process will be demonstrated experimentally in chapter six, but it is most clearly shown when only a single path exists through the DPM to the goal. Such is in effect the case in *Skinner box* experiments. Only pressing the bar delivers the reward. Similarly the only known route to the goal definition sign “Sa” in figure 4-5 is via the path “Sv”-“Sa” (μ-hypothesis “Hva”). If the experimenter denies the animat access to “Sa”, then “Hva” will be tried on every attempt to reach the goal (since there is no other known option), and the estimated cost of this step will rise until Ω is reached. On the other hand if there is some other, as yet unknown, route, then the periods of exploration give the animat the possibility of discovering it by growing the cognitive map. These effects are investigated in the *path blocking* and *alternative path experiments* of chapter six.

4.10. Creating New μ-Hypotheses

New μ-hypotheses are created under two specific circumstances, (1) the appearance of a completely novel sign, postulate H5-1 (*novel event*); and (2) the appearance of a sign that is known, but which was not predicted, postulate H5-2 (*unexpected event*). SRS/E may therefore operate under the *tabula rasa* conditions discussed previously. It is also a strong example of an *unsupervised learning* procedure, no intervention is required from the originator or experimenter to cause or guide the learning process. The originator may, of course, build behavioural patterns into the Behaviour List intended to advantage or bias the animat’s learning process. The experimenter may equally establish situations that trigger or exploit the animat’s innate learning ability to train or teach the animat. In the experiments to be described no such behaviour patterns are used. Conditions under which the experimenter intervenes are described were appropriate.

SRS/E uses a *pattern extraction* method for creating new μ-hypotheses. The detection of a novel or unpredicted sign, notated for the moment “s2”, causes SRS/E to extract a recent action, “r1”, from $\mathcal{R}$, as recorded in the response_activation_trace values, and to extract a sign, “s1” from $\mathcal{S}$, as recorded in the sign_activation_trace values. The new μ-hypothesis:

$$h \leftarrow \mathcal{H}(\text{s1,r1,s2}^{@+t}) \qquad \text{(eqn. 4-16)}$$

is created from the components extracted from the various traces. Note the use of the notation “$x \leftarrow X(y)$” to denote the creation of a list element of type x from some (appropriately typed) element or elements “y”. Note also that the action selected is to be drawn from at least one execution cycle in the past, and that the context sign “s1” shall be contemporary with the action “r1”. As a convention, where “s2” follows “s1” and “r1” by exactly one execution cycle the use of the *“@” (at) notation* will normally be dispensed with, as this is the default relationship. Where all the component token parts for “s1” are drawn from their respective activation traces, then action selection and prediction by the μ-hypothesis will not depend on the current state of the system, only on the recorded past states.

In keeping with Popper’s observation that the simplest means possible should be employed to describe the phenomena (*occam’s razor*), the current implementation of SRS/E initially creates new μ-hypotheses to this notion, concurrent sign “s1” and action “r1” predicting the target sign “s2” on the next execution cycle. The exact combination of elements for the new μ-hypothesis are specified by a *hypothesis template*, which in the current implementation is coded into the structure of the SRS/E algorithm. As the size of $\mathcal{S}^*$ increases, the number of possible options for inclusion in the new μ-hypothesis will increase. Currently, SRS/E may limit the number of μ-hypotheses created for each novel or unpredicted sign appearance. This, in effect, creates a *sampling strategy* for the learning process. The mechanism for an explicit sampling strategy implemented in SRS/E is described later.

This is a form of *instrumental learning*, predicated on a fundamental notion of *causality* between the context in which the animat makes actions, the specific actions made by the animat and the consequences to the animat and its environment of those actions. It is an *animat-centric view*, but there may be other active agents in the environment causing changes. These are only recorded by the animat in so far as they affect the animat's ability to manipulate its circumstances.

Shettleworth (1975) provides evidence that animals may be predisposed to utilise features from the environment selectively. With or without this innate bias it would be a reasonable alternative strategy to create many μ-hypotheses in an attempt to explain the occurrence of the novel phenomena, and allow the subsequent corroboration process to select useful μ-hypotheses and discard the remainder, a sub-set sampling assumption. In the absence of any underlying "theory" about the environment, which is the default assumption, each μ-hypothesis forming "guess" is as good as another[25].

4.10.1. Maintaining the Hypothesis List

Given the use of the *pattern extraction* (token selection from the various lists $\mathcal{I}$, $\mathcal{S}$, $\mathcal{R}$ and $\mathcal{H}$) method for creating new μ-hypotheses one of four outcomes will emerge following a period of corroboration. First, an individual μ-hypothesis may accurately predict its outcome. Second, a μ-hypothesis may accurately predict its outcome only in a fraction of the instances in which it is activated. Third, a μ-hypothesis may never, or very rarely predict correctly. Fourth, a μ-hypothesis may not be activated again, and so will make no predictions that may be corroborated.

The first of these outcomes needs no immediate action. The second outcome may indicate that the μ-hypothesis be a candidate for *specialisation*, one form of differentiation (postulate H6). By this process extra tokens are added to the context sign "s1", on the assumption that the μ-hypothesis is underspecified in its application. JCM and ALP both propose a specialisation mechanism. In the current definition, the Dynamic Expectancy Model isolates candidate μ-hypotheses which have intermediate corroboration values, and which have a maturity (`hypo_maturity`) value greater than the system defined *maturity threshold* level (Ψ). The use of the maturity criteria ensures that candidate μ-hypotheses have undergone a sufficient number of activations and hence corroborative predictions. Maturity is not equivalent to age.

[25]This cluster of hastily formed guesses contingent on a new phenomena may be related to the "first appearances" effect, widely, but often apocryphally, described. For instance King (1987).

For any of these candidates, which are currently on the active list $\mathcal{H}^*$, and where the confidence measure falls between a system defined *lower confidence bound* (θ) and *upper confidence bound* (Θ), an additional token term is added to the existing context sign "s1". In the current scheme this token is drawn from the record of token activations recorded in the respective activation traces. It is in essence another "guess", but (as with μ-hypothesis creation) one drawn from the population of extant observations. The original μ-hypothesis is retained, and a new one appended to the Hypothesis List. Duplicate μ-hypotheses are not installed by SRS/E. By appending the new, modified, sign "s1" to the Sign List a stream of novel signs is created to further activate the μ-hypothesis creation process.

The experiments described later make extensive use of the μ-hypothesis creation steps, but do not necessitate the use of this specialisation step. It is therefore largely speculative. However the intention is to create a population of μ-hypotheses, which attempts to improve its performance based on predictive ability within the lifespan of the animat. Where the initial μ-hypotheses were created from the simplest combination of parts, new μ-hypotheses will only be created when these minimalist interpretations of the environment are demonstrated inadequate through the corroboration process. Among other candidate approaches to this step in the SRS/E algorithm are the use of the cross-over and mutation techniques employed by *Genetic Algorithms* (GA), and the techniques used by the *machine learning by induction* schools of thought.

Both Becker and Mott also discuss *generalisation*, the converse operation to specialisation. In generalisation terms are removed from the context of ineffective schema on the premise that they contain irrelevant additional kernels which over specify and hence reduce the effectiveness of the μ-hypothesis. The Dynamic Expectancy Model does not provide any explicit mechanism for generalisation. It instead relies on the notion that less effective μ-hypotheses will be removed, after a suitable period of corroboration, by the deletion/forgetting process described below.

The third outcome indicates a candidate for deletion, as it apparently fails in its task as a hypothesis about the environment. The current definition for SRS/E selects a candidate set of μ-hypotheses for deletion on the basis of their maturity (compared to the *maturity threshold*, **Ψ**) and confidence values from a sub-set of the population sharing a common consequence sign "s2". A reasonable minimum value for the lower confidence bound (θ, also the minimum bound for specialisation) would be one based on *joint probabilities* (Harrison, 1983):

$$\texttt{joint\_prob} = p(\text{"s1"}) * p(\text{"r1"}) * p(\text{"s2"}) \qquad \text{(eqn. 4-17)}$$

The joint probability value would be that value approximated by a μ-hypothesis created following a true chance or *occult occurrence*. The algorithm's readiness to delete μ-hypotheses must also be related to the number available for predicting "s2". Where only one, or a very limited number of μ-hypotheses are available it appears inappropriate to expunge this knowledge, even where it is demonstrated to be of restricted value. Experimental evidence from *Skinner box* experiments would appear to indicate that experimental animals do not erase operant behaviours even after full extinction, as evidenced by the spontaneous recovery of the extinguished behaviour after a period of rest. It may also be noted that where only a single action elicits reward its use may be particularly persistent during the extinction process.

The fourth outcome offers no information on which to base a decision, and so a pragmatic approach is indicated. In principle an old, untested, μ-hypothesis has no more nor less potential as a valuable item of knowledge than a more recently created one, which has yet to be tested. Where nothing else is known about the outcome there is a clear reason to retain the uncorroborated μ-hypotheses. Where other alternatives already exist, and space is becoming at a premium, a Hypothesis List element falling into this category is a clear candidate for deletion - but as a purely housekeeping consideration.

4.11. The SRS/E Execution Cycle

In the second main part of this chapter the *SRS/E* algorithm is considered in some detail as a series of interrelated computational processes. SRS/E must explicitly balance the demands placed upon it by definitions of innate behaviours provided in the animat's ethogram, goal-initiated behaviours, and by the requirement to generate new behaviours. Goal-setting, goal-seeking and the learning processes are all defined or controlled as part of the total ethogram. The extent to which the animat can create learned behaviours and the degree to which it can override innate behaviours with learned ones are also defined in the original ethogram. In this way SRS/E can truly be described as implementing a “scheme for learning and behaviour”.

4.11.1. Summary of Execution Cycle Steps

Whereas the first part of this chapter described the definition of the various list types and discussed much of the rationale behind various design choices in the construction of the current implementation of SRS/E, this part describes the algorithm primarily from the viewpoint of the manipulations performed on those lists during an individual execution cycle. Figure 4-6 summarises the main steps in each SRS/E cycle. Sub-sections summarise these list manipulations with a degree of formality, utilising the notation developed earlier. The intention of this algorithm is to create a situation where each of the lists is sustained on a continuing basis.

In **step one** the algorithm accepts tokens derived from the animat’s sensors and transducers. These are converted to the internal symbol form using information recorded in the Input Token List, and used to evaluate the activation state of all Sign List elements.

In **step two** the Prediction List is inspected for any predictions made in the past which fall due on the current cycle. These predictions are compared with the active Sign List, and the hypotheses making the predictions are updated, for both successful and failed expectations. This is the corroboration and reinforcement of existing μ-hypotheses (from postulates H3 and H4).

In **step three** the algorithm evaluates the Behaviour List to prepare a candidate action and to determine which, if any, innate behaviours or goals are appropriate in the prevailing circumstances. The SRS/E algorithm requires that the Behaviour List provide a priority associated with each candidate activity or goal. When the highest priority activity is greater than the highest priority goal, no goal seeking behaviour is considered and the algorithm skips immediately to step 6 to perform the chosen action. Whenever step three does not actively select any purposive behaviour or assert a goal a default, exploratory, action will be selected.

Step 1a) *Gather Input Tokens to form* $\mathcal{I}^*$

1b) *Update* $\mathcal{S}^*$

1c) *Cancel satisfied goals from* $\mathcal{G}$

Step 2) *Evaluate past* μ*-experiments from* $\mathcal{P}^t$

Step 3a) *Select default action candidate from* $\mathcal{R}$

3b) *Select innate action and priority from* $\mathcal{B}^{r*}$

3c) *Set goals* $\mathcal{G}$ *and priorities from* $\mathcal{B}^{g*}$

3d) *Innate priority > goal priority?* $\rightarrow$ *to step 6*

Step 4) *Build Dynamic Policy Map (*DPM*) relative to* g^1

Step 5) *Select valenced action from (*DPM $\cup$ $\mathcal{S}^*$*)*

Step 6) *Perform selected candidate action*

Step 7) *Perform* μ*-experiments from* $\mathcal{H}^*$*, update* $\mathcal{P}$

Step 8a) *Novel occurrence?* $\rightarrow$ *create hypothesis on* $\mathcal{H}$

8b) *Unexpected occurrence?* $\rightarrow$ *create hypothesis on* $\mathcal{H}$

8c) *Partially effective hypothesis?* $\rightarrow$ *differentiate to* $\mathcal{H}$

8d) *Ineffective hypothesis?* $\rightarrow$ *delete from* $\mathcal{H}$

Step 9) *To step 1*

Figure 4-6: Summary of Steps in the SRS/E Execution Cycle

In **step four** the algorithm builds (if required) a Dynamic Policy Map. This is performed as a spreading activation graph building algorithm. μ-Hypotheses that are known to lead directly to the top-goal are considered to have a valence level of one, and so define a set of sub-goals (their “s1” component), which in turn act as sub-goals at valence level two, and so on.

In **step five** the algorithm matches the current perceived situation, as expressed by the active Sign List from step one, with the Dynamic Policy Map generated in step four, to select a candidate action to be performed in step six. Step five must also cater for situations where there is no intersection between the current policy map and any active signs, and for circumstances where the policy map proves ineffective at providing a goal path.

Having defined an action to take, either as a high-priority innate action, a goal directed action selected from the Dynamic Policy Map or a default action, this action is passed to the animat actuators in **step six**.

Once an action is selected, and given the active Sign List from step one, a sub-set of the Hypothesis List will be active, able to make a prediction. Active μ-hypotheses take part in μ-experiments. **Step seven** selects all the active μ-hypotheses and causes them to append their prediction about "s2" onto the Prediction List. A μ-hypotheses does not have to have contributed to the action selected in step six to be considered active (*implicit activation*).

Step eight concerns itself with the management of the Hypothesis List. In keeping with the principles defined in the previous chapter. μ-Hypotheses may be *created*, varied or removed within this step.

Having concluded one cycle (**step nine**), the algorithm returns to step one and begins the next. It might again be noted that SRS/E does not provide for any *terminating condition*, there is nothing inherent in the basic algorithm that concludes the continued execution of cycles.

The base SRS/E algorithm, coupled to any behavioural definitions provided by the originator in the ethogram, is expected to imbue the animat with an appropriate degree of *behavioural autonomy*. The new-born animal or human child may require protection and nurturing, the child may be tutored and educated, but these things do not compromise our notion that they are autonomous and so ultimately self-sufficient. Should the undamaged individual require continued nurture, not achieve a normal degree of self-sufficiency, or be unable to learn without continued tuition, then it might reasonably be concluded that an adequate level of autonomy had not been achieved within the ethogram definition. Similarly the ethogram design may call for a protected maturational period, and as an essentially autonomous learning system the animat may be teachable, but these do not undermine the defining behavioural autonomy properties for the ethogram or animat.

4.12. The SRS/E Algorithm in Detail

Figure 4-7 illustrates the major steps in the SRS/E algorithm, the most significant data pathways and their relationships to the various list structures. Individual steps in the algorithm are described in greater detail in the sections that follow. Steps which read from the list structures are indicated with a solid line termination (“•→”), those which add to a list structure by a “+” indicator (“⊕←”), and those which remove elements from a list by a “-” termination (“⊖←”). Each of the *subsumption points* (SP1 and SP2) indicates a stage in the algorithm where a previously selected candidate action may be replaced (subsumed) by an action of higher priority.

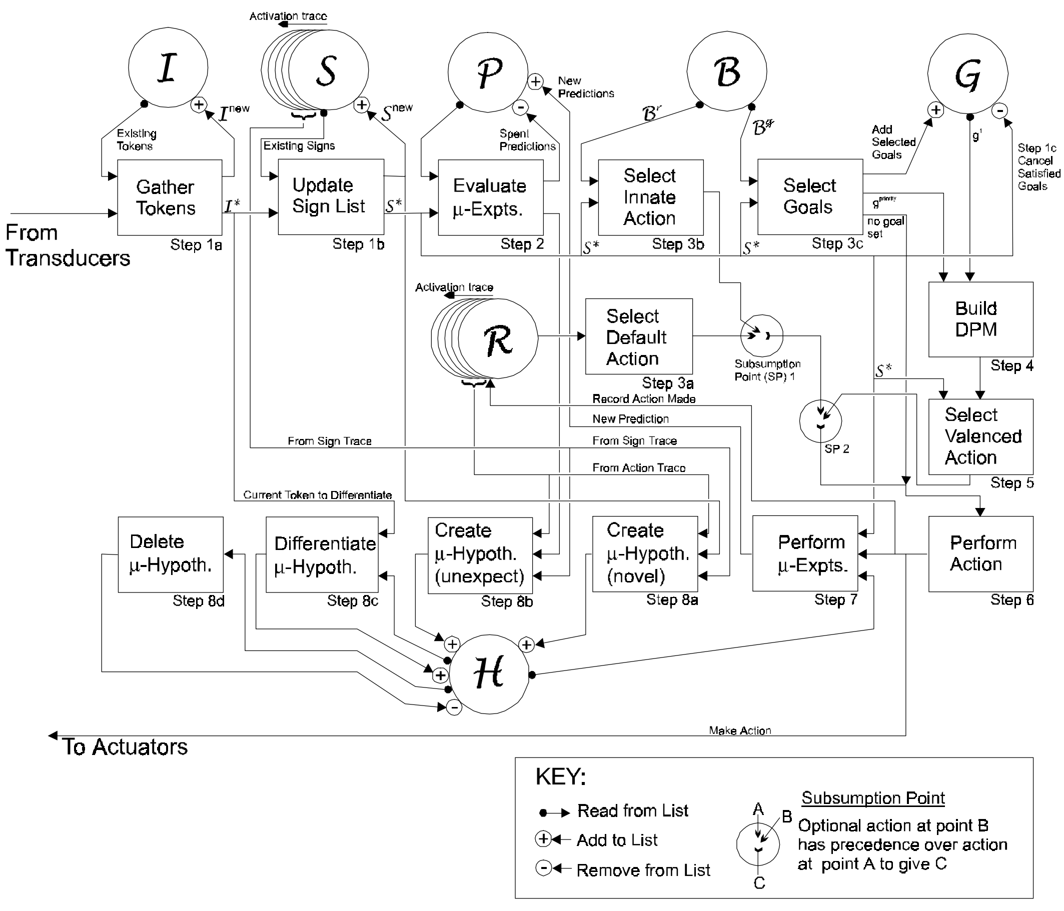


Figure 4-7: The SRS/E Algorithm

4.12.1. Step 1: Processing Input Tokens and Signs

Figure 4-8 shows the list management activities undertaken during **step 1.0** of the SRS/E cycle. In **step 1.1.1**, input token strings are accepted from the input buffer and converted into the internal token form ($\mathcal{i}$). **Steps 1.1.2** perform additional processing on input tokens previously unknown to the system (i.e., any not already on $\mathcal{I}$). The novel token is appended to $\mathcal{I}$ (**step 1.1.2.1**). Additionally a new sign is created from each novel token (**step 1.1.2.2**) and appended to the temporary list $\mathcal{S}^{new}$. Tokens present in the input buffer on the current cycle are assigned to the active Token List, $\mathcal{I}^*$, (**step 1.1.3**). New signs created in step 1.1.2.2 are added to the Sign List (**step 1.2**). The temporary list $\mathcal{S}^{new}$ will be used to drive the learning process of step 8.1. Once all input tokens have been processed, each sign is evaluated according to the criteria laid down in equation 4-3, forms 1 through 4. Every sign meeting the criteria defined for activation are placed on the active Sign List $\mathcal{S}^*$ (**step 1.3**). **Step 1.4** matches elements on the Goal List ($\mathcal{G}$) to any active signs ($\mathcal{S}^*$), and automatically cancels satisfied goals.

Initialise $\mathcal{S}^{new} \leftarrow \{\}$; $\mathcal{I}^* \leftarrow \{\}$; $\mathcal{S}^* \leftarrow \{\}$;

1.1 Accept tokens into buffer, for each `token_string` do

    1.1.1 $\mathcal{i} \leftarrow \mathcal{I}$(`token_string`)   [convert input string]

        [note: $\mathcal{X}(\mathcal{y})$ convert element of type $\mathcal{y}$ to element of type $\mathcal{X}$]

    1.1.2 if $\mathcal{i} \notin \mathcal{I}$ [a token previously unknown to the system]

        1.1.2.1 $\mathcal{I} \leftarrow \mathcal{I} + \mathcal{i}$ [append $\mathcal{i}$ to $\mathcal{I}$]

        1.1.2.2 $\mathcal{S}^{new} \leftarrow \mathcal{S}^{new} + \mathcal{S}(\mathcal{i})$ [create a sign containing $\mathcal{i}$]

    1.1.3 $\mathcal{I}^* \leftarrow \mathcal{I}^* + \mathcal{i}$

1.2 $\mathcal{S} \leftarrow \mathcal{S} + \mathcal{S}^{new}$

1.3 For each $\mathcal{s}$ where $\mathcal{s} \in \mathcal{S}$

    1.3.1 if (`EvalSignConjunction`($\mathcal{s}$))

        $\mathcal{S}^* \leftarrow \mathcal{S}^* + \mathcal{s}$ [eqn. 4-3]

1.4 $\mathcal{G} \leftarrow \mathcal{G} - (\mathcal{S}^* \cap \mathcal{G})$ [cancel satisfied goals]

Figure 4-8: Step One, Token and Sign Processing

4.12.2. Step 2: Evaluating μ-Experiments on the Basis of Prior Prediction

Once active signs have been determined the algorithm may assess the accuracy of past predictions falling due on the current execution cycle and so update the individual μ-hypotheses responsible for those predictions (figure 4-9). **Steps 2.1** process each element of $\mathcal{P}$ where the `predicted_time` is equal to `now`. Where the `predicted_sign` is on $\mathcal{S}^*$ the μ-hypothesis identified by the Prediction List element `predicting_hypo` is updated according to equations 4-8 and 4-11 (**step 2.1.1.1**). The temporary list $\mathcal{S}^{\text{pred}}$ records each sign that was correctly predicted (**step 2.1.1.2**). Similarly **step 2.1.2.1** updates each μ-hypothesis responsible for an incorrect prediction falling due at the current time, according to equations 4-9 and 4-12. For each failed prediction the system variable `rebuildpolicynet` is increased by the amount δ (**step 2.1.2.2**). Spent predictions are removed from $\mathcal{P}$ (**step 2.1.3**). The temporary list $\mathcal{S}^{\text{unexpected}}$ records all active signs that were not predicted by any μ-hypothesis (**step 2.3**), these will be used to drive the learning process of step 8.2.

Initialise $\mathcal{S}^{\text{pred}} \leftarrow \{\}$;

2.1 for every p ($p \in \mathcal{P}$), such that `predicted_time`(p) = `now`, do

  2.1.1 if `predicted_sign`(p) $\in \mathcal{S}^*$ [prediction succeeds]

    2.1.1.1 Update `predicting_hypo`(p) [according to α, eqn. 4-11]

    2.1.1.2 $\mathcal{S}^{\text{pred}} \leftarrow \mathcal{S}^{\text{pred}}$ + `predicted_sign`(p)

  2.1.2 if `predicted_sign`(p) $\notin \mathcal{S}^*$ [prediction fails]

    2.1.2.1 Update `predicting_hypo`(p) [according to β , eqn. 4-12]

    2.1.2.2 `rebuildpolicynet` ← `rebuildpolicynet` + δ

  2.1.3 $\mathcal{P} \leftarrow \mathcal{P} - p$ [remove spent prediction]

2.2 $\mathcal{S}^{\text{unexpected}} \leftarrow \mathcal{S}^* - \mathcal{S}^{\text{pred}}$ [record unpredicted signs]

Figure 4-9: Step Two, Evaluation of μ-Experiments

4.12.3. Step 3: Selecting Innate Behaviours and Setting Goals

The availability of $\mathcal{S}^*$ also allows the Behaviour List, $\mathcal{B}$, to be evaluated (figure 4-10). The default candidate action, `candidate_action`, for this cycle is selected from $\mathcal{R}$ in **step 3.1**. In the present scheme the default candidate action is selected at random from those available. This forms the trial and error (or other default) action if no other candidate is selected during the current cycle. A list of active behaviours, $\mathcal{B}^{r*}$, is selected from the *primary behaviours* part ($\mathcal{B}^{r}$) of the Behaviour List on the basis of a match between the condition part and the active Sign List $\mathcal{S}^{*}$ (**step 3.2**). The action with the highest priority is selected from the active primary behaviours ($\mathcal{B}^{r*}$) and assigned to `innate_action` according to the stored `behaviour_priority` values (**step 3.3**). The actual priority of that behaviour is recorded in the variable `innate_priority` (**step 3.4**). If `innate_action` has a higher priority than the *basal level threshold* (ε) it is adopted as the candidate action, `candidate_action`, for the current cycle in preference to the one selected in step 3.1 (**step 3.7**). The Goal List is built from the *goal setting behaviours* part of $\mathcal{B}$ ($\mathcal{B}^{g}$) in **step 3.5**, and the Goal List priority ordered (according to `goal_priority`) in **step 3.6**. SRS/E selects between innate and goal seeking behaviours on each cycle according to the priority of the top-goal, g^1, and the value recorded in `innate_priority` (**step 3.8**). Where an innate behaviour is selected the algorithm skips directly to perform the candidate action in step 6 (**step 3.8.1**).

Initialise $\mathcal{B}^{*} \leftarrow \{\}$;

3.1 `candidate_action` ← `SelectRandomAction`($\mathcal{R}$)

3.2 for each b where `action`(b) ∈ $\mathcal{B}^{r}$ **AND** `condition`(b) ∈ $\mathcal{S}^{*}$

    3.2.1 $\mathcal{B}^{r*} \leftarrow \mathcal{B}^{r*} + b$;

3.3 `innate_action` ← `action`(max(`behaviour_priority`($\mathcal{B}^{r*}$))) [innate action]

3.4 `innate_priority` ← max(`behaviour_priority`($\mathcal{B}^{r*}$))

3.5 for each b where `action`(b) ∈ $\mathcal{B}^{g}$ **AND** `condition`(b) ∈ $\mathcal{S}^{*}$

    3.5.1 $\mathcal{G} \leftarrow \mathcal{G} + b$ [build Goal List]

3.6 $\mathcal{G}$ ← order(`goal_priority`($\mathcal{G}$)) [order Goal List by priorities]

3.7 if(`innate_priority` > ε) [above basal threshold?]

    3.7.1 `candidate_action` ← `innate_action`

3.8 if(`goal_priority`(g^1) < `innate_priority`) [select goal or innate]

    3.8.1 skip to step 6.0

Figure 4-10: Step Three: Select Innate Actions and Set Goals

4.12.4. Step 4: Building the Dynamic Policy Map

Steps 4.1 determine whether the *Dynamic Policy Map* is to be constructed on this execution cycle. If the goal g^1 is already satisfied, the goal is cancelled (**step 4.1.1**), and the next lower priority goal selected (**step 4.1.2**). If no goal remains on the Goal List control passes directly to step 6.0 (**step 4.2**). If the top-goal is unchanged since the last cycle and the `rebuildpolicynet` value has not exceeded `REBUILDPOLICYTRIP` no change is required and the algorithm skips directly to valenced action selection in step 5.0 (**step 4.3**).

Steps 4.4 (stage 1 of the construction) build the first valence level in the DPM. For all elements ($\mathcal{h}$) of the Hypothesis List where the consequence "s2" is equivalent to g^1 the steps 4.4.n are taken. The estimated cost for the transition is obtained (equation 4-13) and held in $\mathcal{h}^{£}$, the cost estimate value for μ-hypothesis $\mathcal{h}$ (**step 4.4.1**). The temporary list $\mathcal{S}^{v=2}$ is built from the context signs "s1" for μ-hypotheses selected (**step 4.4.2**), these form the sub-goals at the next valence level. The temporary list $\mathcal{H}^{£}$ records the estimated policy cost for the μ-hypothesis $\mathcal{h}$ as $\mathcal{h}^{£}$ (**step 4.4.3**). Similarly the temporary list $\mathcal{S}^{£}$ records the lowest cost solution found so far for each sign implicated in the construction of the DPM (**step 4.4.4**). If the context sign "s1" for any instance of $\mathcal{h}$ is already on the active Sign List $\mathcal{S}^{*}$, then a path from the current situation to the goal has been found (**step 4.4.5**) and the flag `pathavailable` is set **`TRUE`**. The lowest cost path estimate `bestcost` is updated if the estimated cost of this new path is lower than any previously found solution path from this sign to the top-goal (**step 4.4.6**). Once `pathavailable` is asserted the algorithm might to skip to step 5.0 (i.e., perform the action associated with the element $\mathcal{h}$ with the active context sign), or it may continue to build the DPM to discover possible lower cost paths. Were the animat to be constrained to perform an action within a given time this flag is an important indicator that a path exists. The current implementation places no such time constraint on the algorithm.

Steps 4.5-4.8 (stage 2 of the construction) continue the spreading activation process for successive valence levels, `vn+1` (**step 4.5**), until there are no further nodes to expand (**step 4.6**) which terminates the DPM construction. Each node identified as a sub-goal at the previous valence level is expanded (**steps 4.7**) in the manner described for steps 4.4. The temporary list $\mathcal{H}^{£}$ records the policy value for each μ-hypothesis by adding the new cost estimate value for the transition to the previously computed lowest policy value for the sub-goal "s2" (**step 4.7.1**). The temporary list $\mathcal{H}^{£}$ is updated to reflect new policy values (**step 4.7.2**). Whenever a new sign node or a lower estimated policy cost to a sign node is discovered (**step 4.7.3**), the sign is established at the new valence level (**step 4.7.3.1**) and the new or lower cost is recorded (**step 4.7.3.2**). The net effect of this process is to categorise every μ-hypothesis, and so each sign "s1", which is implicated in the DPM by its lowest estimated policy cost to the top-goal. The flag `pathavailable` may be set at any valence level (**step 4.7.4**). The variable `bestcost` is updated whenever a new lowest estimated cost is encountered (**step 4.7.5**). If there is no intersection of sub-goal node and $\mathcal{S}^*$, `pathavailable` remains **`FALSE`** and `bestcost` remains undefined.

Initialise $\mathcal{H}^{£} \leftarrow \{\}$; $\mathcal{S}^{v} \leftarrow \{\}$; $\mathcal{S}^{£} \leftarrow \{\}$;
`rebuildpolicynet` $\leftarrow 0$; `pathavailable` $\leftarrow$ **FALSE**;
`bestcost` $\leftarrow$ **MAXVALUE**; `vn` $\leftarrow 1$ [valence level one]

Rebuild map if goal changed or 'rebuild' greater than threshold
4.1 while ($g^1 \in \mathcal{S}^*$) [top-goal already satisfied]
4.1.1 $\mathcal{G} \leftarrow \mathcal{G} - g^1$ [so remove]
4.1.2 $g^1 \leftarrow$ max(`goal_priority`($\mathcal{G}$)) [and select next highest]
4.2 if($\mathcal{G} = \{\}$) skip to step 6.0 [no goals on Goal List]
4.3 (if $g^1 = g^{1@t-1}$ **AND** `rebuildpolicynet` < `REBUILDPOLICYTRIP`)
skip to step 5.0 [no need to rebuild DPM]

Stage 1 - create first valence level
4.4 for each $\mathcal{h}$ such that `s2`($\mathcal{h}$) = g^1
4.4.1 $\mathcal{h}^{£} \leftarrow$ `GetCostEstimate`($\mathcal{h}$) [eqn. 4-13]
4.4.2. $\mathcal{S}^{v+1} \leftarrow \mathcal{S}^{v+1}$ + `s1`($\mathcal{h}$) [record valenced sub-goals]
4.4.3 $\mathcal{H}^{£} \leftarrow \mathcal{H}^{£} + \mathcal{h}^{£}$ [cost of transition s1 to goal]
4.4.4 $\mathcal{S}^{£} \leftarrow$ s1($\mathcal{h}^{£}$) [record sign cost]
4.4.5 if(`s1`($\mathcal{h}$) $\in \mathcal{S}^*$)
`pathavailable` $\leftarrow$ **TRUE** [path solution found]
4.4.6 if(`bestcost` > $\mathcal{h}^{£}$) `bestcost` $\leftarrow \mathcal{h}^{£}$

Stage 2 - continue spreading activation until done
4.5 `vn` $\leftarrow$ `vn` + 1
4.6 if($\mathcal{S}^{v} = \{\}$) skip to step 5.0 [expansion complete]
4.7 for each $\mathcal{h}$ such that `s2`($\mathcal{h}$) $\in \mathcal{S}^{v=vn}$ [expand each sub-goal]
4.7.1 $\mathcal{h}^{£} \leftarrow$ s2($\mathcal{S}^{£}$) + `GetCostEstimate`($\mathcal{h}$) [eqn. 4-13]
4.7.2 $\mathcal{H}^{£} \leftarrow \mathcal{H}^{£} + \mathcal{h}^{£}$ [record total cost of path]
4.7.3 if(`s1`($\mathcal{h}$) $\notin \mathcal{S}^{v}$ **OR** s1($\mathcal{h}^{£}$) > s1($\mathcal{S}^{£}$)) [new or better path]
4.7.3.1 $\mathcal{S}^{v+1} \leftarrow \mathcal{S}^{v+1}$ + `s1`($\mathcal{h}$) [new sub-goals]
4.7.3.2 $\mathcal{S}^{£} \leftarrow \mathcal{S}^{£}$ + s1($\mathcal{h}^{£}$) [record lower sign cost]
4.7.4 if(`s1`($\mathcal{h}$) $\in \mathcal{S}^*$)
`pathavailable` $\leftarrow$ **TRUE** [solution path found]
4.7.5 if(`bestcost` > $\mathcal{h}^{£}$) `bestcost` $\leftarrow \mathcal{h}^{£}$
4.8 return to step 4.5 [expand next valence level]

Figure 4-11: Step Four, Construct Dynamic Policy Map

4.12.5. Step 5: Selecting a Valenced Action

Steps 5 (figure 4-12) determine whether a valenced action is appropriate, and if so select the action. These steps also manage the *goal extinction* process. A value for the *valence break point* is determined first. If VBP is already set, this value is used (**step 5.1**). Where this is the first instance of a DPM, or the previous valence break point has been exceeded, a new value for VBP is computed according to equation 4-15 (**step 5.3**). The valence break point is cleared if no path is found (**step 5.2**). A temporary list of μ-hypotheses, $\mathcal{H}^{\#\pounds}$, is formed from the intersection of those μ-hypotheses with valence (recorded on $\mathcal{H}^{\pounds}$) and whose condition part "s1" is on the active Sign List $\mathcal{S}^*$ (**step 5.4**). The candidate valenced action, `valenced_action`, is extracted from the element of $\mathcal{H}^{\#\pounds}$ with the lowest estimated policy cost to the goal (**step 5.5**). If the estimated cost of this proposed action is still less than VBP, this valenced action is selected as the overall candidate action, `candidate_action`, for the execution cycle (**step 5.7**). Where there is no intersection of valenced μ-hypotheses and the active Sign List, the candidate action selected in step 3 will be used. This summary of the algorithm does not detail the sub-steps for the *goal recovery mechanism* previously described. **Step 5.8** determines if the total estimated cost of the path has exceeded the *goal cancellation level*, Ω, and if so removes the current top-goal from $\mathcal{G}$.

5.1 VBP $\leftarrow$ `GetValenceBreakPoint()`	[establish VBP]
5.2 if (`pathavailable` = **FALSE**) VBP $\leftarrow 0$	[no path to goal]
5.3 else if (VBP ≤ 0 **OR** VBP > `bestcost`)	[compute VBP]
VBP $\leftarrow$ `bestcost` * VALENCEBREAKPOINTFACTOR	
5.4 $\mathcal{H}^{\#\pounds} \leftarrow \mathcal{H}^{\pounds} \cap (\mathrm{s1}(\boldsymbol{h}) \in \mathcal{S}^*)$	[candidate active signs]
5.5 $\boldsymbol{h} \leftarrow \min(\mathcal{H}^{\#\pounds})$	[select least policy cost]
5.6 `valenced_action` $\leftarrow$ r1($\boldsymbol{h}$)	
5.7 if(`policy_value`($\boldsymbol{h}$) $\leq$ VBP)	[break-point reached?]
`candidate_action` $\leftarrow$ `valenced_action`	[no, use valenced action]
5.8 if(`policy_value`($\boldsymbol{h}$) > Ω)	[goal cancellation level?]
5.8.1 $\mathcal{G} \leftarrow \mathcal{G} - g^1$	[so cancel top-goal]

Figure 4-12: Step Five, Select Valenced Action

4.12.6. Step 6: Performing an Action

Figure 4-13 describes the action reification process. The action `candidate_action`, selected either as an innate response from the Behaviour List $\mathcal{B}^{g}$ (step 3.3), from the Dynamic Policy Map as a valenced action (step 5.7), or as a default trial and error action (step 3.1) is sent to the animat's effectors to be performed on the current cycle (**step 6.1**). The element of the Response List finally selected is recorded on the active Response List $\mathcal{R}^*$ for the current execution cycle (**step 6.2**).

6.1 `DoAction(candidate_action)`	[reify candidate action]
6.2 $\mathcal{R}^* \leftarrow$ `candidate_action`	[record in trace]

Figure 4-13: Step Six, Perform Action

4.12.7. Step 7: Conducting μ-Experiments

Figure 4-14 describes the steps taken to create the predictive expectations. The active Hypothesis List $\mathcal{H}^*$ is constructed from every μ-hypothesis where the context sign "s1" appears on the active Sign List $\mathcal{S}^*$ and the action "r1" appears on the active Response List $\mathcal{R}^*$ (**step 7.1.1**). SRS/E does not distinguish between actions made as part of the goal seeking process and those made due to innate behaviour definitions or for any other reason. As a consequence SRS/E corroborates μ-hypotheses whenever they establish an expectation. Such expectations are added to the Prediction List as a triple recording the μ-hypothesis responsible for the prediction, the predicted sign, the time at which that sign is predicted (**step 7.1.2**). The value t is recovered from the `time_shift` value associated with the μ-hypothesis. These predictions will be corroborated in step 2 of later execution cycles.

initialise $\mathcal{H}^* \leftarrow \{\}$;

7.1 for all h, such that `s1`(h) $\in \mathcal{S}^*$ **AND** `r1`(h) $\in \mathcal{R}^*$

7.1.1 $\mathcal{H}^* \leftarrow \mathcal{H}^* + h$	[record activation]
7.1.2 $\mathcal{P} \leftarrow \mathcal{P} + \mathcal{P}(h,$ `s2`(h), `now` $+ t)$	[make prediction]

Figure 4-14: Step Seven: Conduct μ-Experiments

4.12.8. Step 8: Hypothesis Creation and Management

Steps 8.1 (figure 4-15) are concerned with the creation of new μ-hypotheses when a *novel event* is detected. These steps are triggered when the temporary list S^{new} is not empty. Elements were placed on S^{new} in step 1.2. A new μ-hypothesis is created from a context sign ("s1") selected from the Sign List activation trace record (**step 8.1.2**), from an action ("r1") selected from the Response List activation trace (**step 8.1.3**), and the novel sign extracted from S^{new} ("s2"), (**step 8.1.4**). The newly formulated μ-hypothesis is added to the Hypothesis List (**step 8.1.5**) and its values set to reflect the *creation bonus* previously described. As the μ-hypothesis is created from a novel sign, there is no possibility that it will duplicate an existing μ-hypothesis. The *timebase shift* is achieved by predicting the occurrence of "s2" n cycles in the future, where the "s1" and "r1" values were previously extracted from the respective activation traces n cycles in the past. The relative time shift, +*t*, is recorded in the μ-hypothesis `time_shift` value.

The creation of a new μ-hypothesis may affect the structure of the DPM, and so the system value `rebuildpolicynet` is incremented by Δ to hasten or trigger a DPM rebuild (**step 8.1.6**). The novel sign is removed from S^{new} (**step 8.1.7**), and steps 8.1 repeated until this list is empty. An explicit sampling learning strategy is implemented by omitting steps 8.1.2 to 8.1.6 for one or more of the signs on S^{new} according to a frequency set by the *learning probability rate*. The learning probability rate will also be referred to by the abbreviation *Lprob* and by the symbol (λ). When the learning probability rate is 1.0 every opportunity to create a μ-hypothesis will be used, if it were set to 0.0 no μ-hypothesis creation would occur. In electing to implement a *sampling strategy* at this point any sign passed over will only seed a new μ-hypothesis as a result of the process described in steps 8.2, as it will not reappear on S^{new}.

Steps 8.2 create new μ-hypotheses when unexpected signs are detected. Elements were added to the temporary list $S^{unexpected}$ in step 2.2. The basic mechanism for μ-hypothesis creation is identical to that described in steps 8.1. In a sampling strategy

($\lambda < 1.0$) passed over signs can reappear on $\mathcal{S}^{\text{unexpected}}$ again (as they may remain unpredicted), and so be the subject of this process on a subsequent execution cycle.

Creation on the basis of novelty

8.1 for each $\mathscr{s}^{\text{new}}$ such that ($\mathcal{S}^{\text{new}} \neq \{\}$ **AND** $\mathscr{s}^{\text{new}} \in \mathcal{S}^{\text{new}}$)

8.1.1 if (rand(0.0 .. 1.0) > λ) skip to step 8.1.7

8.1.2 `s1` $\leftarrow$ `Select(`$\mathscr{s}^{x} \in \mathcal{S}^{*@-t}$)

8.1.3 `r1` $\leftarrow$ `Select(`$\mathscr{r}^{x} \in \mathcal{R}^{*@-t}$)

8.1.4 `s2` $\leftarrow \mathscr{s}^{\text{new}}$

8.1.5 $\mathcal{H} \leftarrow \mathcal{H} + \mathcal{H}$(`s1`,`r1`,`s2`$^{@+t}$), where `s1` $\neq$ `s2`

8.1.6 `rebuildpolicynet` $\leftarrow$ `rebuildpolicynet` + Δ

8.1.7 $\mathcal{S}^{\text{new}} \leftarrow \mathcal{S}^{\text{new}} - \mathscr{s}^{\text{new}}$

Creation on the basis of unpredicted event

8.2 for each $\mathscr{s}^{\text{unexpected}}$ such that ($\mathcal{S}^{\text{unexpected}} \neq \{\}$ **AND** $\mathscr{s}^{\text{unexpected}} \in \mathcal{S}^{\text{unexpected}}$)

8.2.1 if (rand(0.0 .. 1.0) > λ) skip to step 8.2.7

8.2.2 `s1` $\leftarrow$ `Select(`$\mathscr{s}^{x} \in \mathcal{S}^{*@-t}$)

8.2.3 `r1` $\leftarrow$ `Select(`$\mathscr{r}^{x} \in \mathcal{R}^{*@-t}$)

8.2.4 `s2` $\leftarrow \mathscr{s}^{\text{unexpected}}$

8.2.5 $\mathcal{H} \leftarrow \mathcal{H} + \mathcal{H}$(`s1`,`r1`,`s2`$^{@+t}$), where `s1` $\neq$ `s2`

8.2.6 `rebuildpolicynet` $\leftarrow$ `rebuildpolicynet` + Δ

8.2.7 $\mathcal{S}^{\text{unexpected}} \leftarrow \mathcal{S}^{\text{unexpected}} - \mathscr{s}^{\text{unexpected}}$

Figure 4-15: Step Eight, Hypothesis Creation

Steps 8.3 (figure 4-16) describe the *specialisation* process by which individual μ-hypotheses are made more specific in their application. Extra specificity is achieved by adding discriminant terms to the context sign conjunction ("s1"). The current definition selects μ-hypotheses that are: (1) active; (2) exceed the *maturity threshold* (Ψ), in that they have been tested many times; and (3) have an indeterminate confidence probability values (`hypo_prob`, or `bpos`) falling between the *upper* (Θ) and *lower* (θ) *confidence bounds*. A selected μ-hypothesis must be active to ensure that the additional elements added to the conjunction are drawn from the set of extant events at the time of modification (i.e., those falling within the range defined by the respective activation traces).

A new context sign is created by adding an additional term to the existing context sign conjunction (**step 8.3.1**). This new term may be drawn from the Input Token List, the Sign List, the Response List or the Hypothesis List. It may take any of the four forms described in equation 4-3. Action (**step 8.3.2**) and consequence (**step 8.3.3**) parts are copied from the existing μ-hypothesis. The new μ-hypothesis is appended to the Hypothesis List (**step 8.3.4**). The original μ-hypothesis is not removed, and will compete with the new one. The new sign created in step 8.3.3 is appended to the Sign List (**step 8.3.5**). On its first subsequent activation the new sign will appear as a candidate on $\mathcal{S}^{\text{unexpected}}$, as there is no μ-hypothesis to predict it. This mechanism therefore provides a continuing source of new signs to drive the learning process indefinitely.

Specialisation (differentiation)

8.3 for all $\boldsymbol{h}$, such that $\boldsymbol{h} \in \mathcal{H}^*$ **AND** `hypo_maturity`($\boldsymbol{h}$) > Ψ
AND `hypo_prob`($\boldsymbol{h}$) > θ **AND** `hypo_prob`($\boldsymbol{h}$) < Θ

8.3.1 `s1` $\leftarrow \mathcal{S}($`s1`$(\boldsymbol{h}) + \mathcal{X}^{@-t})$	[differentiate s1]
8.3.2 `r1` $\leftarrow$ `r1`($\boldsymbol{h}$)	[copy action]
8.3.3 `s2` $\leftarrow$ `s2`($\boldsymbol{h}$)	[copy s2]
8.3.4 $\mathcal{H} \leftarrow \mathcal{H} + \mathcal{H}($`s1,r1,s2`$^{@+t})$	[install new μ-hypothesis]
8.3.5 $\mathcal{S} \leftarrow \mathcal{S}$ + `s1`	[install new sign in $\mathcal{S}$]
8.3.6 `rebuildpolicynet` $\leftarrow$ `rebuildpolicynet` + Δ	

Figure 4-16: Step Eight, Hypothesis Management - Specialisation

Step 8.4 (figure 4-17) defines the criteria for μ-hypothesis deletion. μ-Hypotheses that persistently fail to make effective predictions may be removed. The degree of maturity should be high and the corroboration measures should indicate that the μ-hypothesis has little or no predictive value. μ-Hypotheses are deleted by simply removing them from the Hypothesis List (**Step 8.6**).

Deletion (forgetting) under competition

initialise $\mathcal{H}^{\#} \leftarrow \{\}$;

8.4 for all h, such that $h \in \mathcal{H}^*$ **AND** `hypo_maturity`(h) > Ψ

AND `hypo_prob`(h) < Θ

8.4.1 $\mathcal{H}^{\#} \leftarrow \mathcal{H}^{\#} + h$ [build candidate list]

8.5 $h^{\text{delete}} \leftarrow$ min(`hypo_prob`($\mathcal{H}^{\#}$)) [select a deletion candidate]

8.6 $\mathcal{H} \leftarrow \mathcal{H} - h^{\text{delete}}$ [update Hypothesis List]

8.7 `rebuildpolicynet` ← `rebuildpolicynet` + Δ

Figure 4-17: Step Eight, Hypothesis Management - Forgetting

4.13. Implementation

The SRS/E algorithm is implemented in Microsoft *Visual C++* and runs as a text only Window under *Microsoft Windows* v3.1 or Windows 95. Each of the major lists and their associated functions are defined as object classes. The use of the term "list" here does not imply the use of a list processing language such as *LISP*. Elements of these Lists are allocated and reallocated dynamically, typically stored and indexed as array members. In the interests of efficiency this implementation eschews conventional object oriented message passing in favour of cross-class access functions.

4.14. SRS/E - A Computer Based Expectancy Model

In this chapter the *Dynamic Expectancy Model* developed in chapter three of this thesis has been translated into a single algorithm, SRS/E. MacCorquodale and Meehl (1953) recognised that their expectancy theory postulates were "*incomplete, tentative and nonsufficient*". Becker's JCM was only presented as a proposal for implementation. Mott achieved a substantive implementation of ALP, but was heavily constrained by the timesharing technology available at the time, and by the generally impoverished nature of the robot interface he employed. Drescher provides scant indication of the results for his claimed implementation, beyond an indication of the extensive computational resources required to sustain the marginal attribution process.

The SRS/E algorithm, and its implementation, stands as a "proof by existence", a working model created from the postulates presented in chapter three. The SRS/E algorithm claims to be "sufficient" in this respect, and as an implementation at least one step beyond "tentative". Each of the postulates contributes a small component part of the whole. The processes are less tightly coupled than, say, Watkins' *Q*-learning; a repetitive application of a simple reinforcement transfer rule. More tightly coupled than, say, the idea implicit in Minsky's (1985) notion of a "society of mind". The relatively large number of Dynamic Expectancy Model postulates, and so algorithmic steps, reflects the apparent need to construct a balanced and functional mechanism; in much the same manner as an automobile design requires many coupled systems to achieve an acceptable level of usability, safety, reliability, maintainability and performance. It may be that further work will demonstrate that the system is still overspecified, and elements may be deleted without affecting overall functionality.

Yet SRS/E does not claim completeness. There is still a substantial "back-catalogue" of published research describing a huge range of phenomena that must eventually be explained or incorporated into a larger single model of the animat. In keeping with an idea that evolution adds capabilities to the best of previous generations and proto-typical species it seems inevitable that extra postulates, rather than simplification, will be found necessary.

The next chapter describes an experimental environment to investigate the properties of the SRS/E algorithm as implemented.

Chapter 5

5. Experimental Design and Approach

The implementation of the SRS/E program may be considered to be in two separate parts. The first part encodes the behavioural and learning activities of the algorithm discussed in the previous chapter. The second part provides an emulation of an experimental environment that may be used to investigate the properties of the algorithm. This chapter considers the nature of this simulated "world" and describes some of the facilities available in the SRS/E program to assist the experimenter investigating the algorithm as implemented.

Communication between these two parts of the program is primarily via a single sub-routine call, "`DoWorldAction`()". This is a manifestation of the abstract activity described by the "`DoAction`()" construct of step six of the SRS/E algorithm (figure 4-13). The "`DoWorldAction`()" call takes two important parameters. The first parameter passes an action from the animat to the environment parts. The action takes the form of a `response_string`, an ASCII string extracted from an element of $\mathcal{R}$ representing the action to be taken by the animat on the current execution cycle. The second parameter returns a sequence of tokens representing sensory events detected by the animat following completion of the action supplied in the first parameter. Tokens are returned from the "environment part" to the "animat part" of the SRS/E program via an input buffer and recorded in the *Input Token List*, $\mathcal{I}$. Each token is defined as a sequence of characters from the ASCII set. Each token is separated from others in the input stream by a delineation character. Tokens have no embedded meaning to SRS/E, but the naming policy that has been adopted here is convenient for experimenter analysis of the generated trace and log files. Certain of the user interface utilities exploit this specific token format, and the adoption of an arbitrarily named but otherwise equivalent token would disrupt their operation.

5.1. Experimental Design

The experimental design used here follows that devised by Sutton (1990) to investigate the performance of the *Dyna-PI*, *Dyna-Q* and *Dyna-Q+* algorithms. Environments for his series of experiments took the form of simple grid mazes, through which the animat may progress from some starting point to some other finish or goal point. The animat may not leave the boundaries of the environment (there is no wrap around), and obstacles may be placed into the maze, making certain locations unattainable. Blockages may be added at any point throughout the experimental procedure to test the response of the animat to changing circumstances. In Sutton's design the animat may make one of four basic actions, each translating the animat into an adjacent cell in the grid. These actions, registered into the *Response List* $\mathcal{R}$ as "N", "S", "W" and "E", are equivalent to the actions "UP", "DOWN", "LEFT" and "RIGHT" defined by Sutton. Each action is assigned an *action cost* value of 1.0.

The simplest of these environments is shown in figure 5-1. It will be referred to as *DynaWorld/Standard* in the current work. Several other researchers have used this environment. Booker (1990) and Riolo (1990) have both described extensions to classifier systems tested with this environment. Peng and Williams (1992) describe extensions to the original Dyna framework. Littman (1994) investigated "memoryless" policies, where actions are based solely on current sensation - "traceless" in SRS/E terms. Each investigator is at liberty to adapt or create their own new or variant environment, and there are consequently a wide variety of designs in use.

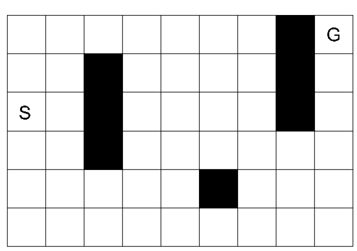


Figure 5-1: Sutton's DynaWorld/Standard Environment

DynaWorld represents a constrained and restricted experimental environment. While not appearing overly demanding as a learning task the maze environment follows in a long tradition of utilising highly controlled experimental environments. They have been espoused, in particular, by the behaviourist and instrumental conditioning schools of research. The latter group in particular place experimental animals in repeatable situations (as typified by the *Skinner box*) with the specific aim of investigating learning phenomena in isolation from other aspects of the subjects naturally occurring behavioural repertoire. The choice of a maze environment is also particularly resonant of the research methods employed by Tolman, of which a number of emulations follow in later sections. All the investigations performed here use simulated environments.

Several other pre-defined environments are available in the currently implemented SRS/E program. At the beginning of each experimental run the experimenter may select from a number of predefined maze patterns. Besides the DynaWorld/Standard environment Sutton (1990) defined an environment of the same size, but which is divided into two parts by a row of obstacles. This "Changing-World" environment is shown in figure 6-12. By selectively removing or adding blocks the behaviour of the animat may be investigated under several conditions where previously known paths disappear, and where new paths become available. A separate maze environment, not due to Sutton, is used in the latent and place learning experiments described later. This environment is shown in figure 6-22. It provides the animat with three distinct paths from the start to goal points, each of different length. The experimenter may, optionally, define environments of arbitrary size, and add or remove blocked cells as required. The experimenter may also elect to allow the animat to translate in all eight directions, "N", "NE", "E", "SE", "S", "SW", "W" and "NW". Diagonal actions attract an action cost of 1.414 (i.e., $\sqrt{2}$). An action that would cause the animat to leave the boundaries of the environment, or to enter a blocked cell, leaves the animat position unaltered, but incurs the cost associated with the action. Following Sutton's definition every cell in the maze is uniquely and reliably identified. In this implementation cells are identified by a single token of the form "XnYn", where "n" is substituted with the cell's X (or Y) co-ordinate. Cell co-ordinates are measured from (0,0), the bottom left hand corner.

Animals are notoriously variable in their performance, even in the most controlled of experimental conditions. The simulated environment holds a number of significant advantages over experimentation with live animals. First among these is the ability to maintain a high degree of control over the environment and the animat. Various aspects of the behavioural repertoire can be suppressed where they would interfere with the progress of the experiment. Motivation, in terms of goal setting, can be controlled independently of the underlying requirement (for instance hunger initiating food seeking activity) by manipulating the equivalent drive. The simulated animat also demonstrates a considerable degree of variability, arising from the nature of the randomising conditions used. Fortunately a ready supply of subject animats may be created to achieve a significant or reliable demonstration of highly variable performance phenomena at little or no cost and far less inconvenience than their naturally occurring counterparts.

The SRS/E program offers a repeatable experimental situation. From identical starting conditions the program will run identically over successive trials. Once any condition varies the course of an experiment will diverge. This facility may be used in several ways. First animats may be effectively “cloned”, taken to some fixed point in the procedure, which is then modified according to the experimental schedule to investigate the specific effects of each variation. Second the procedures may be used in bulk, without constant monitoring and interesting instances identified from the logged record and replicated for further, more detailed, investigation. Finally the SRS/E program offers a high degree of visibility. The use of on-screen information display, in conjunction with the recorded logged information allows the experimenter access to a record of the internal processes that gave rise to specific behaviours and actions. The type and quantity of information displayed and recorded has been refined over a period of time to best reflect what is required for a full analysis. Examples exploiting these facilities occur throughout the next chapter.

5.2. The User Interface

With an environment defined and major parameters selected, the investigator may intervene during each experimental run to control the conditions required by the schedule. At the conclusion of each execution cycle the investigator may utilise the command interface to make changes, to request diagnostic or analytical information, or continue the experiment. This interface is presented on a command line basis, and the options available are shown in the printout of figure 5-2. The interventions available to the investigator fall into five categories: (1) controlling execution cycles; (2) displaying and recording list information; (3) managing goals; (4) managing the animat and environment; and (5) accessing SRS/E program utilities.

```
Command: ?
        <enter>: run for one cycle
        <number>: run until cycle <number>
        @<number>: run for <number> cycles
        ! <x> <y>: break when animat reaches <x> <y>
        t: show Token List
        s: show Sign list
        s<token_id>: show signs using <token_id>
        h[<number>]: show Hypothesis <number> or List
        e[filename]: Export hypothesis List
        p: show current Prediction list
        g: show goal list
        g <sign_number>: set goal (G: save temp tally)
        g-<sign_number>: clear goal
        M: show policy Map (m: valence level map)
        w: show World (W[-]: temp tally [and clear]; WT: world tally)
        = <x> <y>: Move Animat to X,Y
        r: move animat to random starting location
        + <x> <y>: Set obstacle at X,Y
        - <x> <y>: Clear obstacle at X,Y
        u: update system settings
        ; (or *): record comment in trace file
        f: - not available (no trace file)
        #: Show partial path
        ?: this Help
        q: quit

Command:
```

Figure 5-2: The SRS/E Experimenter Command Options

5.2.1. Controlling Execution Cycles

Many experimental schedules to be described call for periods where the animat is free to roam the environment, alternating with goal directed activities. The investigator may single step (“<enter>”) through the experiment, giving time to absorb the information about the previous cycle’s activity, or may allow the experiment to run up to a specified cycle (“<number>”), or for a specific number of execution cycles (“@<number>”). Certain experimental regimes call for the animat to be allowed to locate the goal by random walk exploration, prior to detailed investigation. SRS/E allows the investigator to specify an interrupt condition (“!<x> <y>”) which returns the program to manual control once the named location given by the co-ordinates “<x>” and “<y>” has been visited by the animat.

In the current program this is tied to the animat entering a specific named cell (defined by its co-ordinates). Future versions could, more generally, interrupt on the detection of a specific token, sign, or some combination of these types.

5.2.2. Displaying and Recording List Information

At any command cycle the investigator may display the contents of the Token List ("t"), the Sign List ("s"), the Hypothesis List ("h"), the Prediction List ("p"), or the Goal List ("g"). Additionally the investigator may inspect individual signs associated with a specific token ("s<token_id>"), and individual μ-hypotheses ("h<number>"). The complete Hypothesis List may be exported at any command cycle in a form suited to later importation to a standard spreadsheet utility ("e[filename]").

5.2.3. Managing Goals

In addition to viewing the goal list, the investigator may, at any command cycle, assert ("g<sign_number>") or clear ("g-<sign_number>") any goal. Goals may also be asserted from behaviours coded into the Behaviour List, and are automatically cleared when the goal is satisfied, or when extinguished by the extinction process. When asserting a goal the investigator is also prompted to supply a goal priority for that goal. Whenever a goal is asserted the temporary *world tally* is cleared. The world tally records the number of times each cell has been visited since last reset. The use of the command "G" in place of "g" to assert a goal leaves the tally unchanged to accumulate values.

5.2.4. Managing the Animat and Environment

The shape and size of the experimental environment is fixed at the start of each experiment, however the investigator may add ("+<X> <Y>") or remove ("-<X> <Y>") obstructions in the environment. The animat may be moved to a named location ("=<X> <Y>"), or moved at random to a new starting location ("r"). The animat may not be placed on a blocked location. When using the random relocation command the investigator must be careful not to create any unintentional enclosed pockets of cells into which the animat might become inappropriately trapped.

At any command cycle the investigator may display the "World Tally", showing the number of visits to each cell either since the experiment started ("WT"), or the temporary tally ("W"), which records visits to each cell since the goal was last asserted. The temporary world tally is automatically initialised when a goal is asserted, or it may be explicitly initialised with the "W-" command. Figure 6-10 demonstrates the use of these commands to record the general movements of the animat between stages in a single experiment. The "w" command shows the shape of the environment, currently obstructed cells and the animat position to confirm the investigator has performed the required steps in the experimental schedule correctly.

A representation of the current Dynamic Policy Map may be obtained with the command "M". An example of this data is shown in figure 6-7. Information about the μ-hypotheses with the best estimated cost path to the top-goal for each cell in the maze is mapped onto environment co-ordinates. There may be many μ-hypotheses associated with each of the cells that are not represented. The information presented in the first line of each cell shows the individual μ-hypothesis name and the valence level at which it appears in the DPM. The second line shows the response action associated with the μ-hypothesis. The third shows the estimated cost for the action according to the prevailing evaluation function. The last line in each cell the total estimated cost of the valenced path to the goal.

Each cell represents the "s1" component of the selected μ-hypothesis. Any cell that has not been visited (through, for instance, insufficient exploration), or which is blocked is shown blanked. The DPM displayed is that resulting from the most recent build. If no goal has yet been activated during the current experiment, no DPM is available and none can be shown. A short form display of the current DPM is also obtainable with the "m" command, which displays only the μ-hypothesis identity and valence level.

5.2.5. Accessing Utilities

The investigator may change the values of the important system settings at any point during an experiment using the "u" command. Comments may be recorded to the trace file ("*" or ";"). The trace file may be temporarily suspended, and subsequently reactivated if required (“f”). An experiment is concluded with the quit (“q”) command.

5.3. The System Execution Trace Log

Each time the SRS/E program is run the experimenter has the option to create a record of all significant activities that occur during that run (the *log file*), which may be inspected and analysed in detail after the experiment is concluded. The log file records the following information. (1) The creation and modification of all μ-hypotheses. (2) All predictions made, at the time of their corroboration. The resultant `cpos`, `bpos`, `recency` and other significant values for the predicting μ-hypothesis are recorded. (3) Periodic summaries of numbers of μ-hypotheses created and modified. (4) A copy of the *valenced path* (as figure 4-1) each time the DPM is recomputed. Trial and error actions are not recorded, but valenced (and unvalenced) actions are. (5) The experimenter may request at any time a log record of the complete (or selected elements of) the Token, Sign, Hypothesis, Prediction or Goal Lists. (6) The system automatically logs important activities, such as goal activations, satisfactions and extinctions, changes in goal priority, and actions by the experimenter, that change the environment or animat. The user may also write “freeform” textual comments to the log at any point.

At the conclusion of the experiment the complete final Sign and Hypothesis Lists are logged. Log files are automatically “watermarked” with the start and finish times of the experiment. The current SRS/E program has been augmented with several routines to display information about the DPM in a manner that relates the internal representations to the layout of the simulated environment. Where such representations are recorded in the log they are specific to the simulated environments, not a general feature of the SRS/E system. They will be introduced as appropriate when experimental run results are considered.

5.3.1. Processing Log Results

A utility program, *filter.exe*, has been prepared to extract relevant information from SRS/E log files to facilitate their analysis. Log files (as they are in human readable form) can grow to an unwieldy size. “filter.exe” contains options to prepare a more concise format for review, as well as to extract data in a form suited to graphing and tabulating utilities.

5.4. Important Schedule Variables

At the start of each experimental run the investigator is able to define a number of parameters in addition to specifying the form of the environment. The three most significant of these variables are: (1) the *action repetition rate*, abbreviated to Arep; (2) the *action dispersion probability*, abbreviated to Adisp; and (3) the *learning probability rate* (Lprob, λ). At the start of each experimental run the investigator will also be required to select a seed (*rseed*) for the pseudo-random number generator[26] used. The selection of the same seed allows an experimental run to be replicated while all other conditions remain equal.

5.4.1. Action Repetition Rate (Arep)

Many of the experiments to be described call for actions to be selected at random during an initial period of *trial and error* exploratory behaviour. Sutton adopts the term *random walk* to describe this activity. A true random walk can lead to the animat doubling back on itself to such an extent that exploration of a maze of any size may take an excessive number of execution cycles, with specific areas becoming "over explored". Some of the experiments to be described require that the animat has the opportunity to partially explore most of the environment. The action repetition rate parameter increases the probability that the animat will select a new action at each cycle. With Arep set to 0.0 a new action is selected every cycle, with Arep set to 1.0 the system would always use the same action. An Arep value of 0.5 indicates that the same action as the previous one will be selected with a probability of 0.5 and a new one with a probability of 0.5. Higher values of the action repetition rate increase the tendency for longer sequences of the same action.

[26]The random number generator ("rand()") supplied with the compiler has been used.

Figure 5-3 summarises the effect of the action repetition rate on random walk length over 2,500 trials for each of four settings, where the animat must traverse the maze (of figure 6-1) from start to goal in each trial. The figure shows the number of individual trials (a single traversal) falling into "buckets" of 100 steps. The minimum possible path length is 14 steps. The distribution is skewed, but it may be seen that an Arep value of 0.0 (new action always) leads to a higher average path length (841.9 steps), and a considerable number of instances where the path length reaches a large value than when higher values of Arep are selected. The average path length for Arep = 0.25 was 589.1, for Arep = 0.5 was 419.5 and for Arep = 0.75 was 343.4. The minimum random walk length achieved in the 10,000 trials was 19 (Arep = 0.75). Any advantages gained by increased exploration rates are somewhat offset with higher values of Arep by a tendency for the animat to become trapped at edges and corners, an effect that has detrimental effects in some experimental situations.

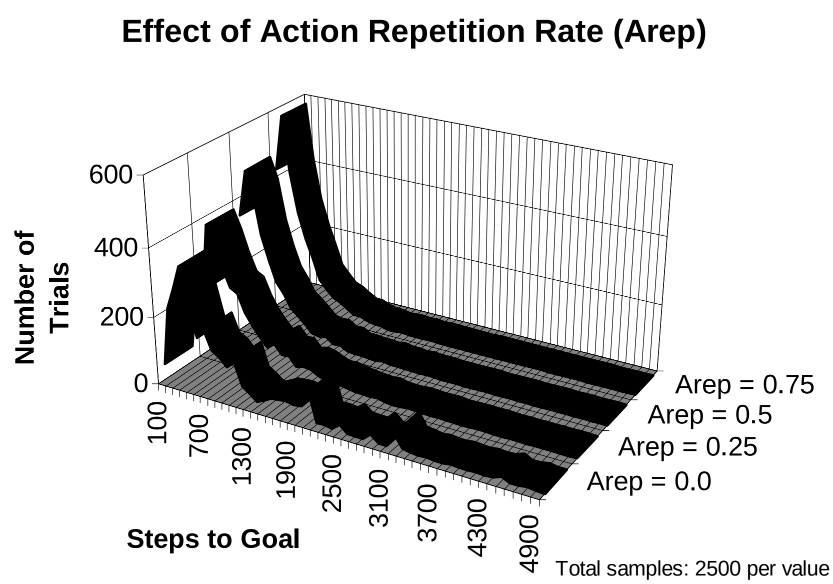


Figure 5-3: Effect of Arep on Random-Walk Path Length

5.4.2. Action Dispersion Probability (Adisp)

Sutton defines a class of noise for the Dyna environments in which actions made by the animat are translated into another action (at the interface between animat and environment) with a given probability p. Actions are translated into either the action one segment clockwise or the one segment anti-clockwise. So, for example, “UP” will be converted to “LEFT” with a frequency defined by (1-p)/2, or to “RIGHT” with a frequency defined by (1-p)/2, or left unchanged with a frequency defined by (p), similarly for the other actions available. The probability with which this translation occurs is controlled in the SRS/E program by the Adisp parameter. When Adisp is set to 1.0 all actions are unmodified. With Adisp set to 0.5, 50% of actions would be unmodified, 25% converted to the action viewed clockwise, 25% to the action viewed anti-clockwise. The source and destination states are still recognised correctly in Sutton’s definition. Other forms of “noise” might also be defined.

5.4.3. Learning Probability Rate (Lprob)

This schedule variable equates directly with the *learning probability rate* (Lprob, λ) described previously. The implementation and properties of the learning probability rate were described in chapter four (section 4.12.8).

5.5. Fixed Schedule Experiments

Several of the experimental procedures to be described call for an intricate or highly repetitive sequence of steps to be performed so as to appropriately demonstrate the properties of the system. Where this is the case the SRS/E suite incorporates program code to set up each trial within the overall experiment and to record the results obtained for subsequent analysis. Typically, the investigator will be required to establish basic parameters for the experiment, but will not be required to directly monitor or intervene in its progress.

Three such *fixed schedule experiments* have been used in obtaining the results described later. The first schedule sets the animat to a defined starting position and counts the number of steps (execution cycles) required for the animat to reach a defined goal location. Having reached the goal the animat is returned to the start location and the run restarted. This may be repeated as many times as required. This fixed schedule is used to provide the comparative results relative to Sutton’s Dyna algorithms (section 6.2, next chapter), and to investigate the effects of noise (section 6.3). These procedures were used to determine the results presented in figure 6-3, and to generate control data for a range of subsequent experiments. The second fixed experimental schedule automates the path-blocking experiments, presenting results in the style of a cumulative reinforcement curve to allow easy comparison with Sutton's results (section 6.5.7). The third fixed schedule automates the latent learning task (section 6.6), and accumulates results such that they may be presented in a manner facilitating comparison with those of Tolman and Honzik (1930). As every fixed schedule experiment starts from known parameters, it is possible to replicate any particular experimental trial up to a known point before transferring to manual control. In this way a particular outcome may be investigated in greater detail or pursued for additional steps if required. The full trace file may be disabled during the fixed schedule phase to avoid recording unnecessary detail and subsequently re-enabled during the manual phase to monitor results in detail.

The next chapter describes and discusses a number of experiments performed with the SRS/E program.

Chapter 6

6. Investigations and Experimental Results

This chapter describes a series of experiments with the SRS/E program. The approach has been to investigate the properties of the algorithm under highly controlled conditions, allowing a clear view of the algorithm's behaviour and performance. Some of the investigations mirror those used to investigate reinforcement learning systems from the modern machine learning paradigm, but some revive and repeat historical investigations used to disambiguate between competing theories of natural learning. It is interesting to note that these issues are still debated as actively as ever after decades of research. There are significant differences in the constitution of animals and animats, and some of the procedures must be modified to reflect these. Nevertheless it is hoped that the spirit of the original experiments is faithfully captured, and some of the lessons and challenges revealed will make a substantive contribution to this ongoing debate.

The previous chapter described the provisions that have made to enable the investigator to design and conduct experiments with the SRS/E program and to analyse and present the results obtained. Section 6.2 of this chapter describes a series of "baseline" experiments in which the performance of the SRS/E algorithm is compared directly to the performance of the Dyna-PI algorithm described by Sutton (1990). The SRS/E algorithm performs the task described by Sutton more efficiently by a factor of some 40 times. Additional investigations in this section clearly demonstrate the development of the classical negatively accelerating learning curve from the widely varying performance of many individual animats, in a manner predicted by the *stimulus sampling theories* previously mentioned.

Experiments described in section 6.3 determine the effects of "noise" on the performance of the SRS/E algorithm. These experiments adopt a definition of noise provided by Sutton, and clearly indicate that the SRS/E algorithm will learn effective solutions even when presented with high levels of disruptive noise. These experiments also distinguish between the effects of noise on the learning process and on animat behaviour. Direct comparison with the Dyna-PI algorithm was not possible as Sutton did not report results with his algorithms.

The experiments described in section 6.4 investigate how the SRS/E algorithm responds to multiple and alternative goals. A number of experimental situations are explored which demonstrate the flexibility provided by the Dynamic Policy Map approach adopted by the SRS/E algorithm. In the alternative goal experiments the animat is required to traverse between a known start and goal situation, which is then reversed (such that the start becomes the goal and *vice versa*). In the multiple goal experiments the animat must visit several, arbitrarily selected goals. These tasks are not achievable with an unmodified *Q*-learning algorithm or any of Sutton's Dyna algorithms, as they all use a static policy map, and so no comparison of performance can be possible. These experiments therefore highlight a radical improvement between existing external reward and the Dynamic Expectancy based methods of reinforcement learning introduced by this thesis.

The investigations described in section 6.5 replicate experimental conditions used by Sutton to determine the effects of blocking known solution paths and opening new solution paths during individual trials of his Dyna-Q+ algorithm. Dyna-Q+ is a specifically modified variant of the Dyna-PI algorithm to address these tasks. The SRS/E algorithm matched the published performance in all the tasks described, although the method employed by the two algorithms is substantially different. SRS/E incorporates an extinction mechanism, not present in *Q*-learning or the Dyna algorithms, which allows the animat to abandon unachievable goal directed tasks and thus escape from potentially "life" threatening situations. The extinction mechanism is developed on biologically plausible grounds.

The experiments of section 6.6 replicate classic "latent learning" procedures. The latent learning experiments were the first to demonstrate conclusively that learning in animals could take place in the absence of external reward or reinforcement. Latent learning may be easily demonstrated with the SRS/E algorithm, and this chapter replicates the procedures adopted to show the effects in animal experiments. Demonstration of latent learning by a reinforcement algorithm employing the *Q*-learning or Dyna methods would appear to be highly problematic, and remains a challenge to those espousing that school of thought. Similarly section 6.7 describes a replication of the “place learning” experiments, in which the animat must make different responses when placed in apparently identical stimulus situations from trial to trial. While the SRS/E algorithm responds to the place learning challenge in a similar manner to experimental animals, it remains unclear how a conventional reinforcement algorithm based on a static policy map could achieve this.

It might be noted that Sutton was obliged to employ a family of algorithms, Dyna-PI, Dyna-Q and Dyna-Q+, to demonstrate the experimental procedures described in this chapter. A single program implementing the SRS/E algorithm has been used for the experiments to be described.

6.1. The Individual Experiments

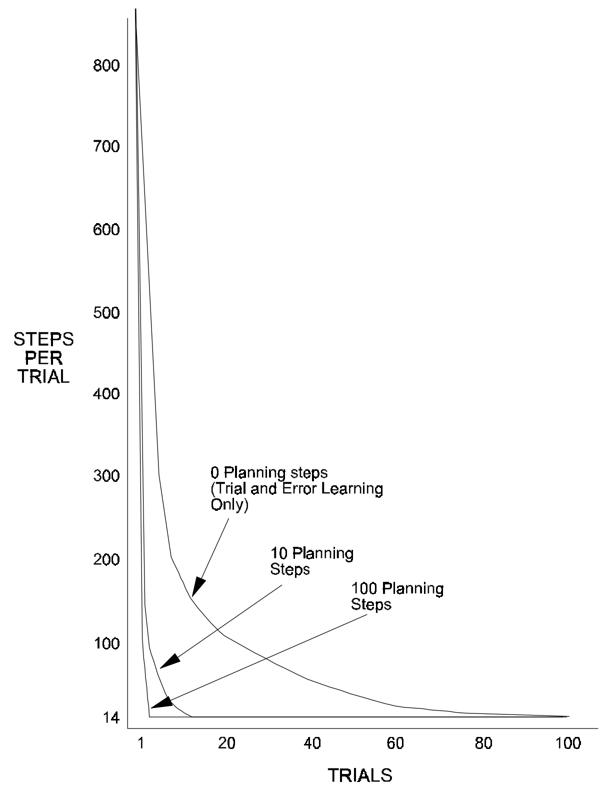
The sections that follow describe a series of individual experiments that attempt to characterise the performance of the SRS/E algorithm in well defined and controlled environments with particular reference to its learning capabilities. Each section is divided into three major parts. Part one will consider the rationale for the experimental schedule and describes the method and experimental procedures adopted for the experiment. As these may be derived from two separate methodologies, natural learning and machine learning, some care will be taken to ensure the data is extracted appropriately to identify and accommodate cross-domain issues. Part two will present the results from specific experiments. Wherever possible this presentation of results will take graphical or tabular form to provide for easy assimilation of the main points being investigated. Where a comparative investigation is being performed (one which replicates or substantially adapts part or all of an established procedure) an attempt will be made to present the SRS/E results in a form reflecting that of the original or source work, where this does not unduly impact or compromise the current experiments. Part three discusses the results of the experiment.

6.2. Baseline Investigations

These initial experiments attempt to characterise the SRS/E algorithm under highly controlled conditions, and to compare its performance to a well-established example of reinforcement learning. Sutton (1990) has extensively investigated a family of algorithms related to the idea of dynamic programming. To establish a performance baseline SRS/E is tested under conditions functionally identical to the descriptions given for Dyna-PI and “learning curves” (indicating improvement in performance following practice) obtained. Dyna-PI is presented by Sutton as showing substantial performance improvements over previous reinforcement learning methods.

Dyna-PI alternates “actual” movements in its simulated environment with “hypothetical experiences” derived from a world model created from data gathered during the actual exploration phases. Sutton refers to these periods of hypothetical activity as “planning”; a more apposite term might be “rehearsal”. The three curves of figure 6-1 indicate the effect of increasing the ratio of “hypothetical experience” relative to “actual experience”. The outer curve, labelled “0 planning steps” is equivalent to the performance of the underlying learning algorithm, converging with the optimal performance line (14 steps/trial) after about 90 trials. Where the animat is permitted 10 “planning” steps interspersed with each actual trial the curve reaches the optimal value after some 12 trials. As the ratio increases, the performance improvement becomes ever more apparent. In effect an equivalent amount of computation has been performed, although observable activity is substantially reduced.

SRS/E retains no additional internal world model. To obtain baseline learning curves SRS/E will be successively handicapped by artificially limiting the frequency with which it can exploit a recognised learning (by creation) opportunity. This is achieved by manipulating the learning probability rate (Lprob), while leaving other experimental conditions unchanged. Varying the learning probability rate introduces sampled learning, partially emulating the effects of spurious or irrelevant signs being incorporated into μ-hypotheses.



\monolith\mazes\graphic 5.4

Figure 6-1: Results from Sutton's Dyna-PI Experiments
(from Sutton, 1991, p. 219)

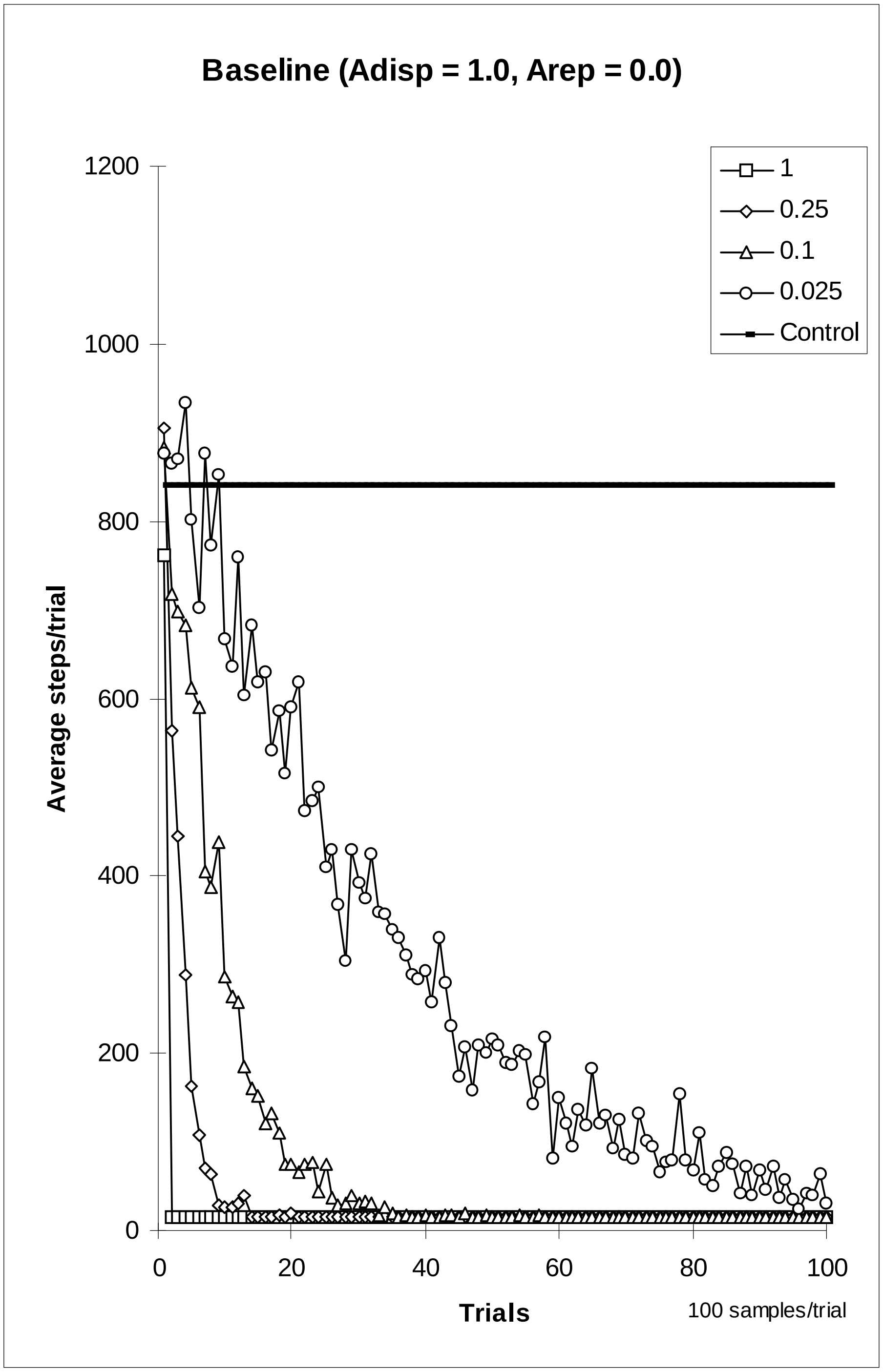
6.2.1. Description of Procedure

To perform the baseline experiments the first fixed schedule is used, which automatically selects and initialises the DynaWorld/Standard environment. Four separate learning curves are created with four different values of the *learning probability rate*, 1.0 (all learning opportunities taken), 0.25 (25% of opportunities taken), 0.1 (10% of opportunities) and 0.025 (2.5% of opportunities). The other factors are held constant for the duration of the experiment. In addition a control baseline is established indicating the animats' performance without valenced behaviour. Each curve is the average of 100 separate experimental runs, each of 100 trials. For each run a new animat (based on a new random starting seed) is placed at the starting point (located at X = 0, Y = 3) and allowed to run the maze. The number of steps taken to reach the goal (at X = 8, Y = 5) are recorded for each trial.

At the conclusion of each trial the animat is returned to the starting point, the goal reasserted (with a priority of 1.0) and the animat released to traverse the maze following whatever valenced path is available. In Sutton's experimental paradigm reward is assigned and the animat is returned to the starting location when the goal is reached. As corroborative learning does not take place in SRS/E until predictions are verified, the animat is allowed to remain undisturbed in the experimental maze for an additional 16 execution cycles after the goal is reached before the trial ends. Each curve is therefore composed of 10,000 visits to the goal location (100 runs of 100 trials). The control line is determined from 2,500 random walks from start to finish. The complete experiment comprises 42,500 visits to the goal location. This is comparable to Sutton's experimental design. The remaining system and animat parameters were held constant throughout the procedure (Arep = 0.0, Adisp = 1.0, α = 0.5, β = 0.2, γ^1 = 0.0, γ^2 = 0.9, γ^3 = 0.1, γ^4 = 0.0).

6.2.2. Results and Analysis of Baseline Experiment

Figure 6-2 summarises the results of the baseline learning experiments. With learning probability rate = 1.0 every opportunity to learn by creation is taken. As the exploration by random walk is protracted due to the selection of a new random action at each cycle most of the possible μ-hypotheses have been created by the first time the goal location is reached. The random walk length for the first trial is highly variable (average of the 100 runs 743.25, best 24 steps, longest 4,380). On being returned to the starting point for a valenced trial to the goal location there is consequently a good chance that an optimal (there are many such paths), or nearly optimal path will be created. The average path length for this second trial is 15.32 (best is 14 steps). Of the 100 runs, 53% of the second trial achieved the optimal 14 step path, 34% the 16 step path, 8% the 18 step path, 4% the 20 step path and one path of 22 steps. By trial 100 the average valenced path length had fallen to 14.96, still above the achievable best.



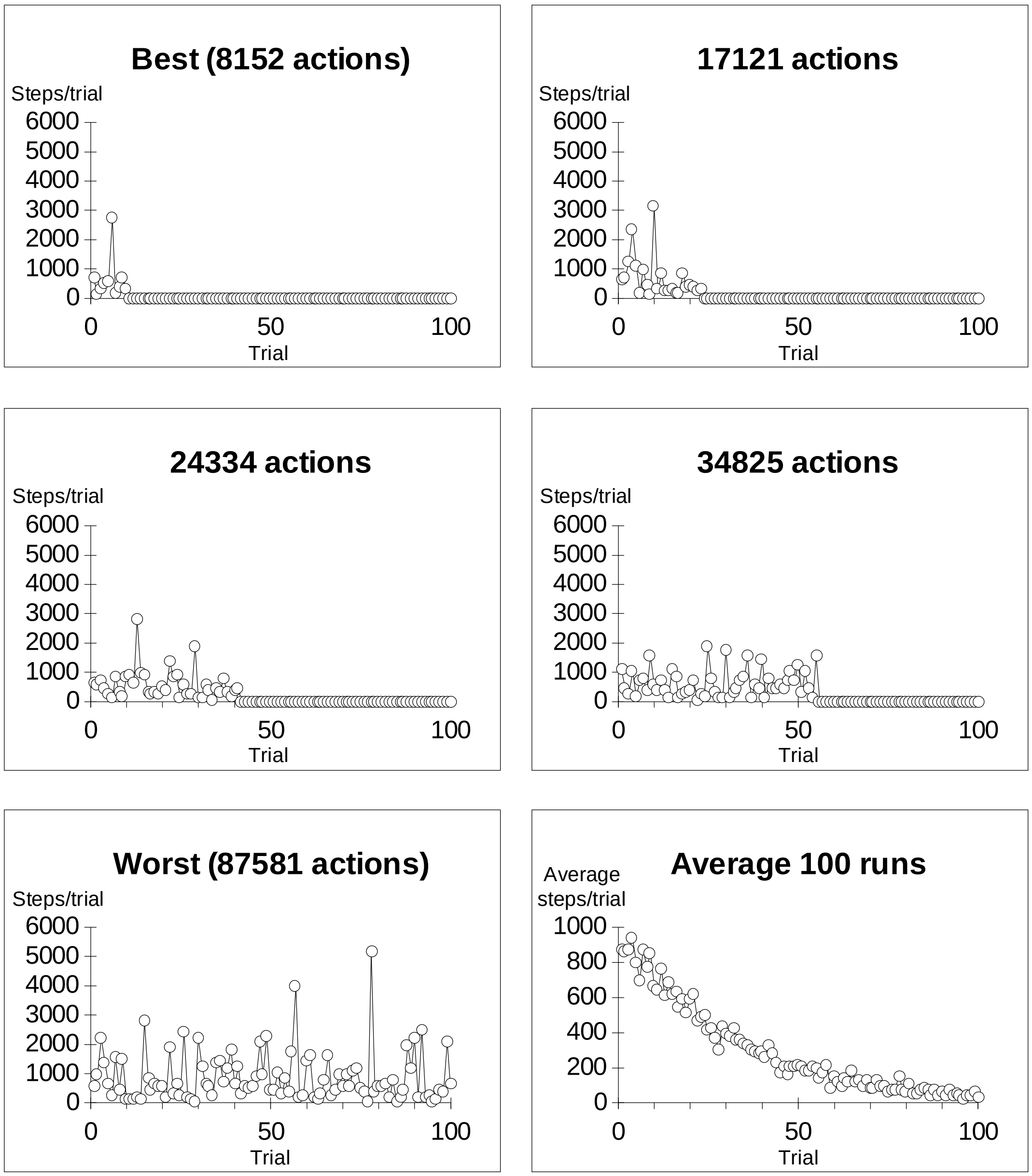
monolith\results\bse1all\base1.xls

Figure 6-2: Baseline Learning Curves (Lprob = 1.0, 0.25, 0.1 and 0.025)

With values of Lprob less than unity, the learning curves take on a more traditional appearance. Discovery of the optimal (or near optimal) path is delayed. The effect of decreasing the probability that a learning by create event will occur has a quite distinctive effect on the rate at which performance improves (as indicated by falling steps/trial), and on the point at which performance stabilises at its minimal level. The last animat to find its stable valenced path for Lprob = 0.25 (diamond graph markers) is at trial 26, the last one for Lprob = 0.1 at trial 56 (triangle markers). The penultimate animat for Lprob = 0.1 stabilised at trial 40. This point of stability has not been reached for the Lprob = 0.025 curve after 100 trials, four individuals from the initial 100 animats still not having found a complete valenced path. An individual animat is defined here as an animat assigned a specific value to the pseudo-random number generator seed (*rseed*) at *parturition*. This value will remain unchanged for the individual for the duration of the experiment.

Figure 6-3 details the performance of a selection of individual animats from the Lprob = 0.025 curve. The five individuals are selected on the basis of the total number of actions they took during the experimental run. Individuals were ordered according to the total number of steps taken in the 100 trials, the sub-figures indicate the "best" (fewest steps), the "worst" (most steps) and the quartile individuals. The "best", individual 84, (rseed = 840) made a total of 8,152 actions (minimum possible is 1,400, figures exclude the run-on period), stabilising by trial 11. Individual 69 (rseed = 690) had stabilised by trial 10, but the preceding random walks had taken more steps. The individual ranked 25th in the population (individual 68) stabilised on trial 24, 50th (individual 78) at step 42, 75th (individual 9) on trial 56 and the "worst" (individual 99) finally stabilised on trial 116. The net effect is shown in the lower right sub-figure.

For each trial, where Lprob ≠ 1.0, the transition from a poor solution path to the near optimal, stable, one is in most cases quite distinctive and often abrupt - as though "the penny dropped". Inspection of the trace information confirms that the effect is primarily due to the probability with which μ-hypotheses at low valence levels leading to the goal sign are formed. Until these particular μ-hypotheses have been created the formation of an effective Dynamic Policy Map is not possible, and so the majority of actions remain unvalenced. Even though this final step is not in place the learning of other μ-hypotheses is still taking place. Once the near goal connections are made, with a probability regulated by Lprob, sufficient μ-hypotheses are invariably available to create an effective DPM from start to goal. A less common effect where a short "stub" DPM builds out from the goal, which subsequently connects to the main body of knowledge is also observed. The overall observable effect on measured path lengths of this stub phenomenon in relation to random walk length is small. This interpretation of the probabilistic nature of the learning process has much in common with the *stimulus sampling theories* promoted by *William Estes* and others (Bower and Hilgard, 1981, Ch. 8 for summary of this position).



monolith\figures.ppt:slide 3

Figure 6-3: Contribution of Individual Animats to Learning Curve

6.2.3. Discussion

Under the learning conditions defined by the learning curve where Lprob = 1 the performance comparison with Sutton's Dyna-PI system is clear. Where Dyna-PI takes approximately 90 trials to reach a stable minimum path solution, SRS/E does so in a single trial across all individuals in the test population. Dyna's poor performance in these circumstances arises from two properties. First, reinforcement is only made at the point the animat reaches the goal, and second, the effects of that reinforcement only propagate back towards the start state labelled "S" one level at a time. At a very minimum then the influence of the reinforcing goal state cannot reach the starting point until the animat has made many "forward" transitions. It might be conjectured that there is a form of "two-steps-back/one-step-forward" strategy that would optimally spread the goal's influence, but this would be a highly artificed strategy. In practice sufficient numbers of propagating transitions are not made until a large number of trials have been completed. Protracted learning rates are recognised as a limitation of this class of reinforcement learning algorithm (e.g., Wyatt, 1995). The protracted learning rate of this class of reinforcement algorithm provides an advantage in terms of noise immunity. The lack of immediate commitment allowing an accurate model of the variability to be constructed. SRS/E will be tested in a later experiment to determine the degree to which learning rate and task performance degrade under the noise conditions defined by Sutton.

Is it not the case then that all SRS/E is doing is recording every transition, building a simple graph and so easily traversing it? For Lprob = 1.0 the conditions for learning are indeed ideal under these experimental conditions. Each state is recognised by a unique and reliable identifier, every action reliably transitions between two such states, the μ-hypothesis creation mechanism explores exactly this relationship first, and the animat is permitted to learn *ad libitum*. Why should learning be anything other than *one-shot* when conditions are ideal? As these conditions move toward more realistic circumstances the expected, and observed, learning performance falls away from this ideal case. In doing so they repeatably demonstrate the forms of the learning curve so ubiquitously observed in experiments with animals.

Several reinforcement algorithms claim to achieve optimal performance over a fixed task of this nature[27], yet SRS/E does not demonstrate perfect performance even after 100 trials under the optimal conditions (Lprob = 1.0, figure 6-2). Recall that the average path length was 15.32 on the second trial, and improved only marginally to 14.96 after all 100 trials. Why should this be? SRS/E and reinforcement learning algorithms make fundamentally different assumptions. Dyna-PI is set a repetitive task and builds a static policy map. For every condition an optimal policy action is ultimately made available. By successively reducing the learning rates and action selection variability (by reducing the *Boltzmann distribution* "temperature") the policy map stabilises. Under these conditions it may be more germane to enquire how the performance of SRS/E improves at all while the goal is continually reasserted. The answer lies in the 16 run-on cycles following the animat's arrival at the goal location. Learning occurs independently of valenced behaviour and new μ-hypotheses can be created during this brief period.

SRS/E is specifically an algorithm for learning and behaviour. Goals arise, are satisfied (or not) and the animat moves on to some different activity. Once a goal is asserted the algorithm pursues it via the best path without additional exploration, using whatever information is available at the time. The experimental circumstances described here exclude any variability due to noise, so that when the goal is continually reasserted without interruption, the animat pursues the path without variation. Where an optimal path is located first, then all subsequent paths are also optimal, where a sub-optimal path is located, all subsequent paths will be sub-optimal. Under normal conditions the animat would pursue other activities, allowing new μ-hypotheses to be created, and so overall improvement in goal acquisition would occur over time. There is a detectable correlation between the amount of exploration during the random walk exploratory phase and the resulting average path length under valenced test conditions. Enabling the oscill (γ^4) component would explicitly add the dimension of exploratory behaviour, but would always tend to detract from the performance of optimal solutions.

[27] Notably those which reduce to an established *dynamic programming* technique and are thus able to exploit the existence of optimal solution proofs (Ross, 1983).

6.3. The Effects of Noise

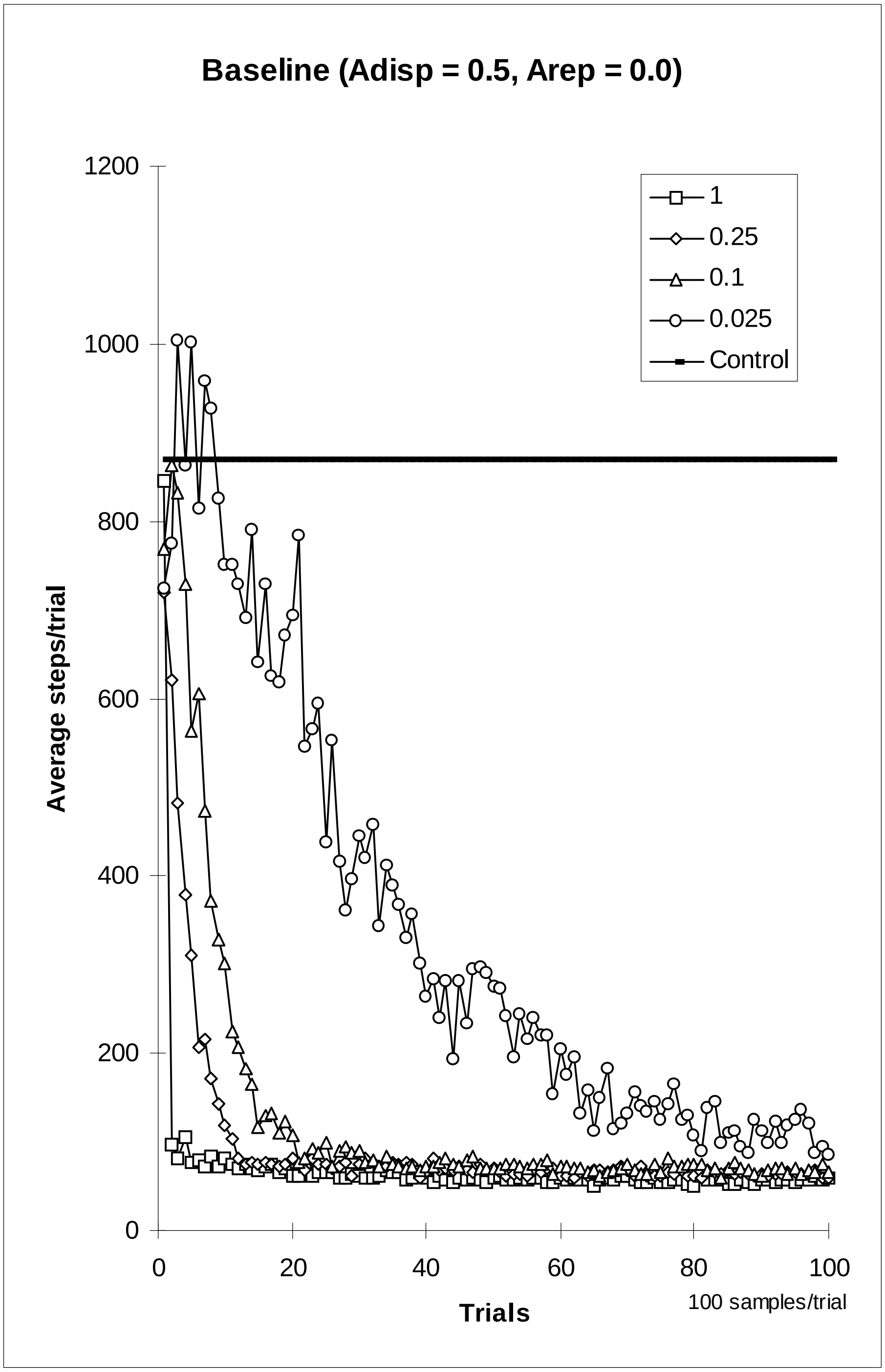
Sutton (1990) defines a test procedure for determining the effects of noise on the Dyna family of reinforcement algorithms. Noise, by Sutton's definition, perturbs the proper action of the animat by altering the effect of its actions, effectively after the animat has issued them, and so is completely outside the control of the animat. Provision for adding this form of noise is made within the SRS/E system. It is controlled by the action dispersion probability (Adisp) parameter. Adisp is selected by the investigator at the start of each experimental run. Its use and effects were described earlier in chapter five. This series of experiments is designed to evaluate the effects on both learning and valenced behaviour in SRS/E. Sutton did not publish noise results for the Dyna algorithms.

6.3.1. Description of Procedure

The experimental procedure described for the baseline experiments was repeated, with the exception that Adisp was set to 0.5 (50% of actions changed, 50% unchanged). The data from the total of 42,500 trials was recorded and plotted as before. A separate control line was determined for these experiments. The complete experimental procedure was then repeated with Adisp set to 0.75 (75% of actions unchanged, 25% changed).

6.3.2. Results and Analysis of Experiment

Figure 6-4 summarises the results from this investigation for Adisp = 0.5. Two points are of note. First is that the slope of the learning curve is not noticeably different for the results obtained in the noise free situation. Second the average valenced path length following stabilisation (as measured by the mean of the last 25 trials for Lprob = 1.0, 0.25 and 0.1, total of 7,500 individual trials) is markedly higher at 65.84 than that for the noise free case, at 15.46. There is also more variability in the valenced path lengths (as determined by the standard deviation, 45.99 as opposed to 1.34 for the noise free case). The Adisp = 0.75 trials resulted in a mean of 25.19 and a standard deviation of 14.42 under the same conditions. The learning curves in this case also showed a similar slope to the Adisp = 1.0 and 0.5 investigations.

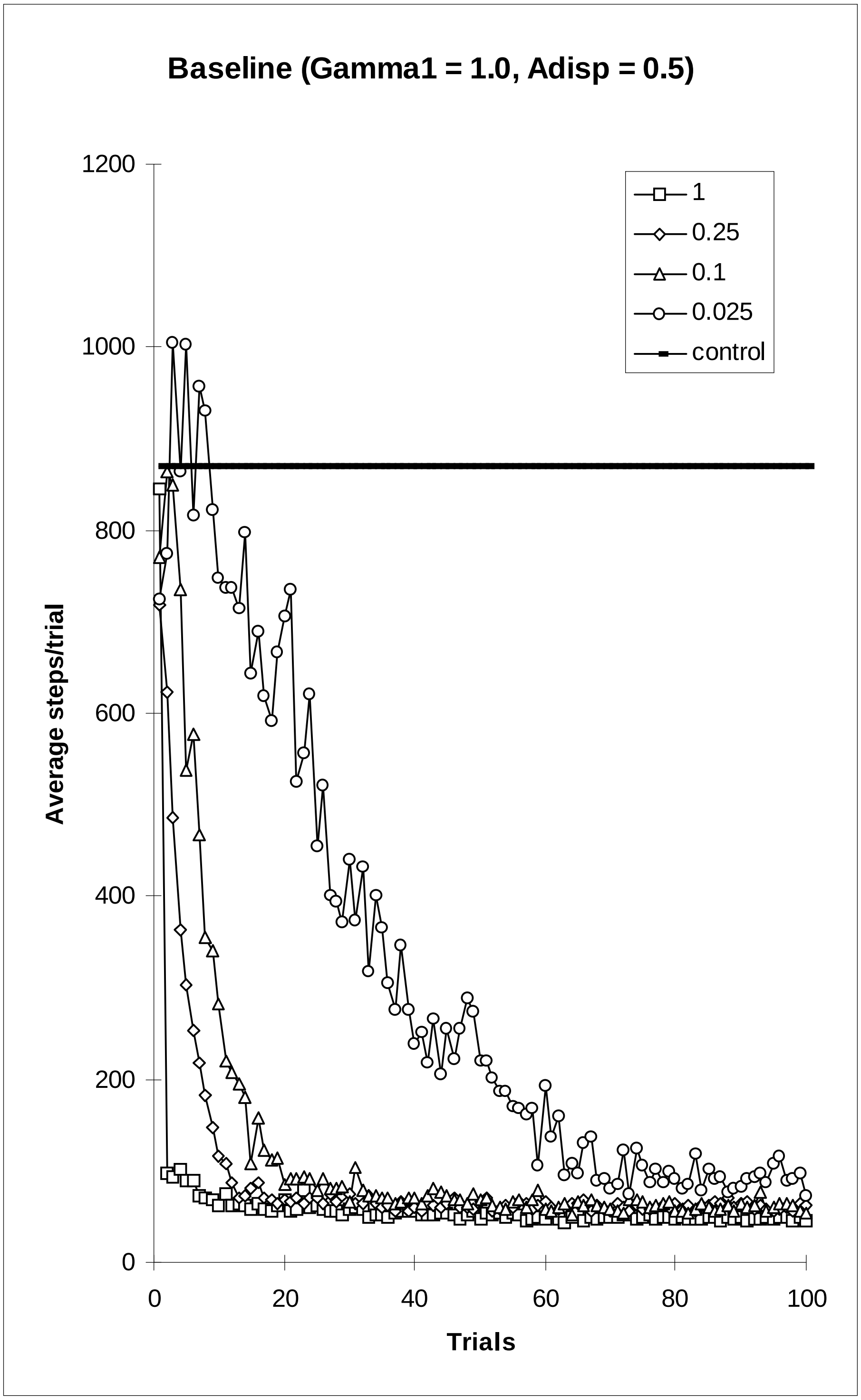


monolith\results\bse50all\bse50all.xls

Figure 6-4: Baseline Learning with Noise (Adisp = 0.5, Lprob = 1.0, 0.25, 0.1 and 0.025)

6.3.2.1. Tuning Parameters for Static Environments

The “standard” set of *selection factor* values (γ^1 = 0.0, γ^2 = 0.9, γ^3 = 0.1 and γ^4 = 0.0) was employed for the above investigations. These settings are appropriate to a changing environment, as the cost estimate values are biased toward more recent events. The experimental environment used here is essentially static, apart from the introduced noise, the level of which remains constant. The investigation with Adisp = 0.5 was repeated (for Lprob = 1.0, 0.25, 0.1 and 0.025 over 100 runs each of 100 trials), with the value of γ^1 set to 1.0 (so $\gamma^2 = \gamma^3 = \gamma^4 = 0.0$). Cost estimates are therefore directly related to the probability of successful prediction of each μ-hypothesis. The estimates are calculated from the unadjusted count of frequencies of satisfied expectations to total activations from the cycle on which the μ-hypothesis was created. Figure 6-5 shows the resulting learning curves. Conditions were identical to the results shown in figure 6-4, except as indicated.



monolith\results\gamma1\base_g1.xls

Figure 6-5: Baseline with Noise (Adisp = 0.5, γ^1 = 1.0)

The average valenced path length following stabilisation (as measured by the mean of the last 25 trials for Lprob = 1.0, 0.25 and 0.1, a total of 7,500 individual trials) is indeed lower, at 56.83 (stddev = 56.18), than for the γ^2 = 0.9 case, (65.84 steps/trial), but still higher than that for the noise free case (15.46 steps/trial).

These results indicate that alterations in the cost estimation parameters have some effect, but that this is not as pronounced as might have been expected under these conditions.

6.3.2.2.The Effects of Noise: Learning or Behaviour?

The question remains whether the decrease in animat goal seeking performance is primarily due to inaccuracies in the Dynamic Policy Map, or a consequence of the disruption due to the animat's individual action selections being thwarted by the noise process. This detailed investigation takes a specific individual and allows it to run for 100 trials with the noise parameter Adisp set to 0.5 (to replicate the baseline run). The investigator then regains manual control of the experiment and forces the value of Adisp to 1.0 (no dispersive noise), returns the animat to the start location, enables the standard goal and records the number of steps taken. Figure 6-6 compares the two subsequent trial paths, trial 101 with Adisp = 0.5, and trial 102 with Adisp = 1.0.

(A) Path with Adisp = 0.5 (trial 101)

```
Cycle 7299: World is 6 by 9
   0    0    0    0    0    1    1 ****   1X
   0    0 ****    0    1    5    3 ****   1
   2    1 ****    0    1    3    2 ****   3
   1    0 ****    1    1    0    1    3   3
   1    3    1    1    0 ****    0    0   0
   0    2    0    0    0    0    0    0   0
Mean = 0.914894, std dev = 1.185013
```

(A) Path with Adisp = 1.0 (trial 102)

```
Cycle 7314: World is 6 by 9
   0    0    0    0    0    0    0 ****   1X
   0    0 ****    0    0    0    0 ****   1
   1    1 ****    0    0    0    0 ****   1
   0    1 ****    1    1    1    1    1   1
   0    1    1    1    0 ****    0    0   0
   0    0    0    0    0    0    0    0   0
Mean = 0.319149, std dev = 0.483779
```

monolith\figures.ppt:slide 4

Figure 6-6: a) Path with Adisp = 0.5 (trial 101), b) Adisp = 1.0 (trial 102)

Inspection of the Valenced Path printout (figure 6-8) from the experiment trace log file confirms the soundness of the valenced path created under noise conditions. Figure 6-7 shows the policy map generated at the conclusion of trial 101. Each location shows the appropriate action except X=5, Y = 0 (bottom row, fourth back from right corner). μ-Hypothesis H223 ("S28<X5Y0> → D → S29<X6Y0>") has an estimated cost of 3.0, 14 of the 42 activations to date having succeeded. The "correct" μ-hypothesis, H121 ("S28<X5Y0> → R → S29<X6Y0>") has an estimated cost of 4.66, only three of the 14 trials to date having succeeded. Such is the consequence of probabilistic dispersive noise. Each action is selected independently, there is no guarantee at any point the ratio of the three possible actions reflects the 0.5:0.25:0.25 selection process. The location is away from the valenced goal path and consequently these policy recommendations were developed during the exploration period. Were this location to fall on the valenced path the system would naturally select H223. On the assumption it would fail in 75% of cases its estimated cost would eventually rise above that of H121, which would then become the preferred choice. Note that the majority of other estimated costs (line four in each location cell) more closely reflect the expected value of 2.0. Figures 6-6, 6-7 and 6-8 were all extracted from the latter ($\gamma^1 = 1.0$) investigation.

```
Policy map at cycle 7299
+--------+--------+--------+--------+--------+--------+--------+--------+--------+
|H164@14 |H378@13 |H29@12  |H45@11  |H276@10 |H380@9  |H148@8  |........|        |
|R       |R       |R       |R       |D       |D       |D       |........| GOAL   |
|   28.44|   26.64|   24.41|   22.28|   20.42|   18.68|   16.38|........|        |
|    1.80|    2.23|    2.13|    1.87|    2.00|    2.00|    1.70|........|        |
+--------+--------+--------+--------+--------+--------+--------+--------+--------+
|H1@15   |H18@14  |........|H109@10 |H48@9   |H49@8   |H301@7  |........|H493@1  |
|D       |D       |........|R       |R       |D       |D       |........|U       |
|   29.91|   28.20|........|   20.96|   18.42|   16.68|   14.68|........|    2.33|
|    1.38|    1.62|........|    2.55|    1.74|    2.33|    2.18|........|    2.33|
+--------+--------+--------+--------+--------+--------+--------+--------+--------+
|H70@14  |H176@13 |........|H305@9  |H213@8  |H50@7   |H331@6  |........|H492@2  |
|R       |D       |........|R       |R       |R       |D       |........|U       |
|   28.53|   26.58|........|   18.70|   16.52|   14.35|   12.50|........|    4.52|
|    1.94|    2.00|........|    2.18|    2.17|    1.85|    1.99|........|    2.19|
+--------+--------+--------+--------+--------+--------+--------+--------+--------+
|H192@13 |H20@12  |........|H100@8  |H294@7  |H382@6  |H383@5  |H384@4  |H422@3  |
|D       |D       |........|R       |R       |R       |R       |R       |U       |
|   26.66|   24.58|........|   16.26|   14.45|   12.67|   10.51|    8.51|    6.59|
|    2.03|    2.17|........|    1.81|    1.78|    2.15|    2.00|    1.93|    2.06|
+--------+--------+--------+--------+--------+--------+--------+--------+--------+
|H182@12 |H247@11 |H84@10  |H95@9   |H286@8  |........|H393@6  |H350@5  |H346@4  |
|R       |R       |R       |U       |U       |........|R       |U       |U       |
|   24.63|   22.41|   20.45|   18.22|   16.06|........|   12.33|   10.45|    8.63|
|    2.21|    1.96|    2.23|    1.96|    1.61|........|    1.88|    1.93|    2.04|
+--------+--------+--------+--------+--------+--------+--------+--------+--------+
|H185@13 |H205@12 |H207@11 |H209@10 |H210@9  |H223@8  |H227@7  |H407@6  |H426@5  |
|R       |R       |R       |R       |U       |D       |R       |R       |U       |
|   25.43|   23.38|   21.51|   19.66|   17.85|   17.00|   14.00|   11.65|   10.10|
|    2.06|    1.87|    1.85|    1.81|    1.79|    3.00|    2.35|    1.54|    1.48|
+--------+--------+--------+--------+--------+--------+--------+--------+--------+
```

Figure 6-7: Policy Map at Conclusion of Trial 101

Separate observations from a number of individual runs from both investigations, and from inspection of Dynamic Policy Maps ("M" command) confirm that the effects on valenced path length are mainly from the execution of the behaviour, rather than faults in the μ-hypothesis creation process or construction of the policy map. "Inappropriate" actions still appear in the DPM, and may do so at any point in the investigation due to the chance of long sequences of noise affected actions altering the relative strength of the μ-hypotheses relevant to the achievement of any given location in the path. Clearly this is more likely in the case where learning is biased towards recent events. In this instance a long sequence of noise affected actions will have a disproportionate effect at any point in the animat's existence. Where $\gamma^1 = 1.0$ the same sequence of noise affected actions will have greater effect while the total activations of the affected μ-hypothesis is low. In practice the system has shown itself (over thousands of trials) to be particularly tolerant of these chance events, re-establishing appropriate paths once the sequence of anomalous events is ended.

```
VBP @ 7256 = 285.322, bestcost = 28.5192
GOAL 46, Max valence level is 16
H70 predicts S5[X1Y3] from S0[X0Y3] (*active) after R (cost = 1.942029, total = 28.519203)
H176 predicts S6[X1Y2] from S5[X1Y3] after D (cost = 1.978261, total = 26.577173)
H20 predicts S7[X1Y1] from S6[X1Y2] after D (cost = 2.169492, total = 24.598913)
H247 predicts S22[X2Y1] from S7[X1Y1] after R (cost = 1.942308, total = 22.429422)
H84 predicts S23[X3Y1] from S22[X2Y1] after R (cost = 2.246154, total = 20.487114)
H95 predicts S26[X3Y2] from S23[X3Y1] after U (cost = 1.981482, total = 18.240959)
H100 predicts S20[X4Y2] from S26[X3Y2] after R (cost = 1.833333, total = 16.259478)
H294 predicts S25[X5Y2] from S20[X4Y2] after R (cost = 1.764706, total = 14.426144)
H382 predicts S33[X6Y2] from S25[X5Y2] after R (cost = 2.152542, total = 12.661438)
H383 predicts S40[X7Y2] from S33[X6Y2] after R (cost = 2.012987, total = 10.508896)
H384 predicts S42[X8Y2] from S40[X7Y2] after R (cost = 1.934307, total = 8.495909)
H422 predicts S44[X8Y3] from S42[X8Y2] after U (cost = 2.051020, total = 6.561603)
H492 predicts S45[X8Y4] from S44[X8Y3] after U (cost = 2.185185, total = 4.510582)
H493 predicts S46[X8Y5] (goal) from S45[X8Y4] after U (cost = 2.325397, total = 2.325397)
Valenced path in 14 steps, estimated cost 28.519203
```

Figure 6-8: Planned Valenced Path (trial 101)

6.3.3. Discussion

The introduction of dispersive noise into the SRS/E system is undoubtedly reflected in the performance of the animat under these controlled experimental conditions. These investigations also confirm that the learned component of the system is resilient to this form of noise (as is also claimed for certain *Q*-learning systems), actions derived from available μ-hypotheses at each choice point reflecting probabilities from past experience. The system may be made more or less reactive to change in the environment by the selection of parameters. Sutton (1990) suggests the possibility that a second order learning phenomena might be employed to determine the long term applicability to an individual animat of a particular strategy. Alternatively selection pressures within a population of individuals might be considered an appropriate strategy.

Dispersive noise, of the form investigated here is only one form of noise. The current implementation of SRS/E also allows for the introduction of random tokens into the input token stream. Such tokens emulate the presence of extraneous events, unrelated to the performance of the task. Using the postulate system described SRS/E will incorporate these random occurrences into μ-hypotheses as a matter of course. SRS/E will be sensitive to this form of noise. First in that it will precipitate the formation of spurious μ-hypotheses, diluting the Hypothesis List and adding computational overhead. Second in selecting whatever response was incorporated into the spurious μ-hypothesis at the time of its creation, inappropriate actions will be selected in pursuit of the current top-goal. As the availability of more effective μ-hypotheses increases, these spurious μ-hypotheses will contribute less to the behaviour of the animat and will eventually be expunged by the μ-hypothesis deletion procedures considered in chapter four.

6.4. Alternative and Multiple Goals

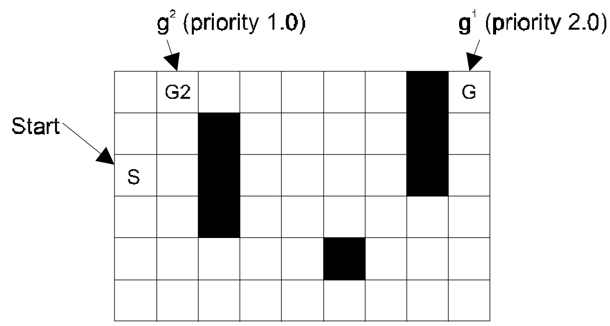
These investigations demonstrate the effect of the SRS/E system when confronted with several different goals, either sequentially or simultaneously. The results of these investigations illustrate the manner in which SRS/E handles goals and valenced behaviour, and highlights the differences between the Dynamic Expectancy Model and reinforcement learning methods that create a static policy map.

6.4.1. Description of Procedure

In investigation one of this experiment naïve animats are allowed an exploration period in the chosen environment, in this instance DynaWorld/Standard (figure 5-1). Each run uses the defined starting point ("S"). The initial unvalenced trial-and-error exploration period is chosen to allow the animat adequate opportunity to thoroughly explore its environment (1,000 execution cycles). An action repetition rate (Arep) value of 0.5 is selected to reduce initial random-walk time. The unvalenced time to reach the goal is noted. At the end of the exploration period the animat is returned to the known starting point, and the goal state ("G") is asserted with a priority of 1.0. The valenced time to reach to Goal is noted. On reaching the standard goal ("G") the original starting location ("S") is now asserted as the goal, with a priority of 1.0, and the valenced time for the animat to re-traverse the environment noted. To confirm these findings these two traversals are repeated, and the respective valenced path times noted.

As a control, investigation two of the alternating goal experiment repeats investigation one of the experiment with the start and goal locations reversed at every stage in the procedure. The procedure is repeated 10 times and the results tabulated. A single instance is selected and individual paths presented for detailed discussion.

The third investigation of this experiment presents the animat subjects with two goals simultaneously. The path generated to reach these two goals should verify the mechanism by which SRS/E seeks and satisfies elements on the Goal List $\mathcal{G}$. Individual naïve animats are given an identical training period to the previous investigations using the DynaWorld/Standard environment, before being returned to the start location “S”. Two goals are then enabled simultaneously, one of which is the original goal (“G”), with a priority of 2.0, and the other chosen to be at some location (“G2” at X = 1, Y = 5) on or near an expected valenced path between start and original goal. The goal “G2” is assigned a lower priority (1.0), figure 6-9.



Graphic 5.12 from monolith\mazes.cdr

Figure 6-9: Simultaneous Goal Locations

6.4.2. Results and Analysis of Experiment

Results for the first investigation are shown in table 6-1. The first column indicates the starting random seed, the second the number of actions taken during the random walk to reach the location “G”. The goal is not asserted and so has no special significance to the animat at this stage. The third shows the length of the valenced path for the first traversal from “S” to “G”. The fourth column records the length of the valenced path returning from “G” (as starting point) to “S” (now valenced as the goal). The fifth and sixth columns record the valenced path lengths from “S” to “G” (valenced) and then from “G” to “S” (valenced) respectively. The animat position is only changed by the investigator once, directly following the random-walk period.

Under these essentially ideal learning conditions the initial valenced path from "S" to "G" is close to the minimum. The variation observed is consistent with the observation that the 1000 random-walk cycles was insufficient to completely build the full potential Hypothesis List, so solution paths may be sub-optimal. The first return path ("G" → "S") consistently requires more cycles than would be expected following this level of experience. Figure 6-10 details the individual animat paths at different stages in a single experimental run and indicates the reason for the apparently anomalously extended path length. Figure 6-10a records (shown using the "W" command) the number of visits by the animat to each location during the exploratory, unvalenced, random-walk period. The location cell labelled "X" (X=8, Y=0) indicates the position of the animat when it was removed by the investigator to the start location for the first valenced run. Figure 6-10b shows the first valenced path, non-optimal at 16 steps, no doubt as a consequence of the greater degree of exploration in the upper part of the environment on this particular run.

Seed	1st visit "G"	S→G (1)	G→S (1)	S→G (2)	G→S (2)
10	915	16	23	15	14
20	317	14	28	13	14
30	216	14	18	13	15
40	101	15	15	13	16
50	534	14	19	15	14
60	167	14	14	15	13
70	379	14	18	15	16
80	265	16	27	15	14
90	134	14	33	15	16
100	140	14	29	13	14
Average	316.8	14.5	22.4	14.4	14.6

Table 6-1: Results for Investigation One of Dual Goal Experiment

Figure 6-10c shows the return path. The animat moves to location (X=8, Y=0) immediately and appears to become trapped there for some number of execution cycles, thereby increasing the overall path length to 27 (from a possible 14). This is an experimental artefact, demonstrating that this emulation of learning and behaviour requires as much care in the conduct of experimental procedure as does work with real animal subjects. The forcible movement of the animat to the start location caused a spurious μ-hypothesis ("H167: <X8Y0> → D → <X0Y3>")[28] to be created, which promises a short-cut to the current goal location. The μ-hypothesis H167 fails to deliver this promise at every trail. Its cost estimate contribution increases at each attempt until it exceeds that for the effective path, which is adopted at the next DPM rebuild. When this path is again valenced, the shorter path is adopted immediately, figure 6-10e.

(A) 1000 steps random-walk (rseed = 80)

```
Cycle 1001: World is 6 by 9
  95   56   22   47   21   10   27 ****    5
  54   28 ****   22   16   12   31 ****    4
  43   42 ****   21   12    6   19 ****   36
  16   32 ****   21   16   18   10    6   17
  14   13   13   14   14 ****    3    2   17
  12   20    7   23    5    8   10   15   46X
Mean = 21.297873, std dev = 17.189804
```

(B) S → G (1)

```
Cycle 1018: World is 6 by 9
   0    1    1    1    1    1    1 ****    1X
   0    1 ****    0    0    0    1 ****    1
   1    1 ****    0    0    0    1 ****    1
   0    0 ****    0    0    0    1    1    1
   0    0    0    0    0 ****    0    0    0
   0    0    0    0    0    0    0    0    0
Mean = 0.361702, std dev = 0.483779
```

(C) G → S (1)

```
Cycle 1046: World is 6 by 9
   0    0    0    0    0    0    0 ****    1
   0    0 ****    0    0    0    0 ****    1
   1    1X****    0    0    0    0 ****    1
   0    1 ****    1    1    1    1    1    2
   0    1    1    1    0 ****    0    0    2
   0    0    0    0    0    0    0    0   10
Mean = 0.595745, std dev = 1.501772
```

(D) S → G (2)

```
Cycle 1062: World is 6 by 9
   0    1    1    1    1    1    1 ****    1X
   0    1 ****    0    0    0    1 ****    1
   0    1 ****    0    0    0    1 ****    1
   0    0 ****    0    0    0    1    1    1
   0    0    0    0    0 ****    0    0    0
   0    0    0    0    0    0    0    0    0
Mean = 0.340426, std dev = 0.483779
```

(E) G → S (2)

```
Cycle 1077: World is 6 by 9
   0    0    0    0    0    0    0 ****    1
   0    0 ****    0    0    0    0 ****    1
   1X   1 ****    0    0    0    0 ****    1
   0    1 ****    1    1    1    1    1    1
   0    1    1    1    0 ****    0    0    0
   0    0    0    0    0    0    0    0    0
Mean = 0.319149, std dev = 0.483779
```

monolith\figures.ppt:slide 5

Figure 6-10: Animat Random and Valenced Paths (investigation 1, rseed = 80)

Table 6-2 records the results of investigation two of this experiment, where the roles of "S" and "G" from figure 6-1 are reversed throughout the procedure. The results are broadly similar to those of investigation one and clearly demonstrate that these results are independent of the actual start and goal locations.

[28]"H167 predicts S0[X0Y3] (goal) from S36[X8Y0] after D (cost = 1.818182, total = 1.818182)": from the valenced path summary recorded in the experiment trace file.

Seed	1st visit “S”	G→S (1)	S→G (1)	G→S (2)	S→G (2)
10	125	16	33	16	15
20	113	14	28	14	13
30	355	16	22	15	13
40	355	16	24	15	15
50	103	16	29	16	13
60	228	14	35	14	15
70	921	16	15	16	15
80	111	14	15	14	15
90	66	14	18	14	15
100	216	14	17	13	13
Average	259.3	15.0	23.6	14.7	14.2

Table 6-2: Results for Investigation Two of Dual Goal Experiment

Table 6-3 summarises the results obtained for the simultaneous goal procedures of investigation three. The effect of setting these two goals is to cause the animat to visit each in turn. In the majority of cases the animat visits the more distant, but higher priority goal first, and then doubles back to satisfy the secondary lower priority goal. The average valenced path length to the first goal is 14.33, and the average total travel to both goals is 32.44. The disruptive effects of the forced return to “S” are still apparent. In one instance the goals are visited in the reverse order (rseed = 80), with valenced path lengths of 3 and 16 respectively. This is purely because the secondary goal lay on the path taken by the animat to the primary goal. A goal is satisfied by being achieved, regardless of whether or not this was because of a valenced action specifically intended to satisfy that goal. The use of “cloned” animats for parts 1 and 3 of this experiment means the initial exploratory and first goal paths are identical.

Seed	1st Visit “G”	1st Goal	2nd Goal
10	915	16	29
20	317	14	39
30	216	14	27
40	101	15	32
50	534	14	27
60	167	14	27
70	379	14	35
80	265	3	16
90	134	14	37
100	140	14	39
Average	316.8	13.2	30.8

Table 6-3: Results for Investigation Three, Simultaneous Goals

Figure 6-11 shows two individual goal paths. Figure 6-11a records the path for rseed = 30, and is typical of the situation where the primary goal is visited first, then the secondary goal. Figure 6-11b shows the situation where the secondary goal is satisfied first because it happens to lie on the valenced path to the primary goal (rseed = 80).

(A) S → G1 → G2 (14/27 steps, seed = 30)

```
Cycle 1029: World is 6 by 9
   0    1    1    1    1    0    0 ****    1
   0    0X****    0    1    1    1 ****    2
   1    0 ****    0    0    0    1 ****    2
   1    0 ****    1    1    1    2    2    2
   1    1    1    1    0 ****    0    0    0
   0    0    0    0    0    0    0    0    0
Mean = 0.595745, std dev = 0.684167
```

(B) S → G2 → G1 (3/16 steps, seed = 80)

```
Cycle 1018: World is 6 by 9
   0    1    1    1    1    1    1 ****    1X
   0    1 ****    0    0    0    1 ****    1
   1    1 ****    0    0    0    1 ****    1
   0    0 ****    0    0    0    1    1    1
   0    0    0    0    0 ****    0    0    0
   0    0    0    0    0    0    0    0    0
Mean = 0.361702, std dev = 0.483779
```

monolith\figures.ppt:slide 5

Figure 6-11: Sample Simultaneous Goal Paths

6.4.3. Discussion

These investigations show substantial differences between existing reinforcement learning methods and the SRS/E algorithm. Goals may be selected at will from the available elements in the Sign List, and a Dynamic Policy Map built from the available μ-hypotheses to attempt a solution path. A standard reinforcement or *Q*-learning algorithm would presumably have to completely rearrange the static policy map over many trials before reasonable performance to the new goal is re-established. As reinforcement does not take place until the changed goal is achieved, if that new goal did not fall on the solution path to the previous goal, this might never happen. This result from the Dynamic Expectancy Model is considered a significant challenge to conventional reinforcement learning algorithms.

Investigation three of this experiment demonstrates SRS/E's flexibility and effectiveness in handling multiple goals. Much progress has been made in adapting reinforcement algorithms to build several policy maps to address multiple goals (section 2.4.2). This approach brings a severe computational cost penalty as the number of recorded goals increases, and means that all goals must be identified before learning can take place. These limitations do not apply to SRS/E. Section 7.2 proposes some extensions to SRS/E to modify its goal seeking behaviour to balance the estimated cost of achieving a goal with the given priority of the goal.

6.5. Multiple-Path, Blocking, Shortcut and Extinction Investigations

The individual investigations in this experiment series evaluate the performance of SRS/E in a range of conditions where multiple paths exist, become available, or cease to be available, between a constant start and constant goal location. The first investigation determines the learned behaviour of SRS/E in an environment where two distinct paths, one longer than the other, exist between start and goal (*multiple-path*). The animat has been allowed to adequately explore the environment fully before the start of the investigation. The investigation further determines the effect of blocking the preferred route.

In the second investigation the effects of blocking one previously explored and known path, and then two known paths is considered. This investigates the *extinction* phenomena, where a goal is abandoned as unattainable. The third investigation repeats a procedure reported by Sutton (1990) to determine the enhanced performance of his Dyna-Q+ system, compared with Dyna-PI, when presented with the situation where a known short path becomes blocked, and a previously unknown path is released (*path blocking*). Results of this latter investigation are presented in a manner comparable to that employed by Sutton. Finally the performance of SRS/E and programs from the Dyna family are considered in a situation where a previously unknown shortcut is introduced.

This series of investigations uses an experimental environment described by Sutton (1990) and shown in figure 6-12. Start "S" and Goal "G" locations are the same throughout the investigation. Obstructions are selectively added or removed during individual investigations at the points marked "A" and "B".

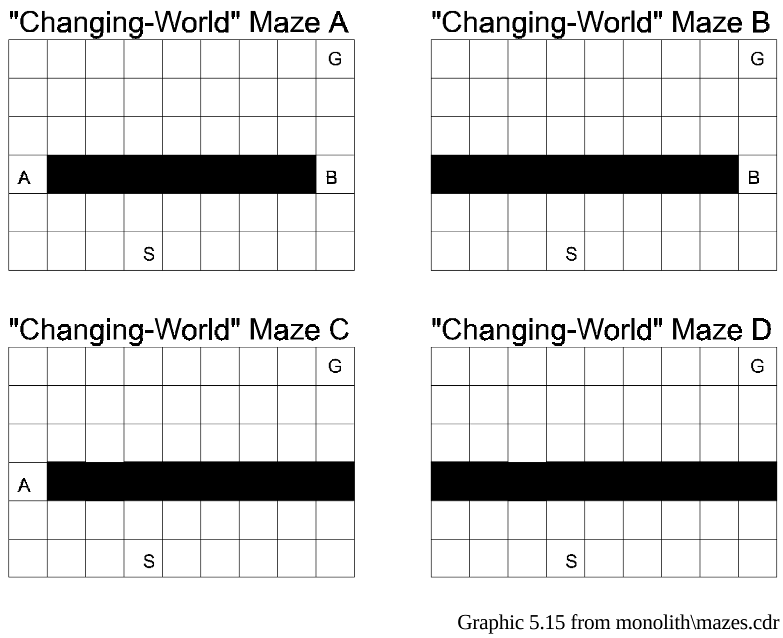


Figure 6-12: Changing World Environments

6.5.1. Investigation One (Multiple-Path), Procedure

This investigation determines the actions of an animat in an environment with two known paths, one of which is shorter than the other. Under these circumstances the animat is expected to take the shorter of the paths (that of lower estimated policy cost), but select the longer path should the shorter become unavailable. In this investigation the animat is allowed to explore the environment of figure 6-12a for 1000 cycles as a random walk with no goal asserted. With Arep is set to 0.5, this allows sufficient time for the environment to be completely explored. On completion of this first phase the animat is returned to "S" and goal "G" asserted with a priority of 1.0. The investigator confirms that the animat reaches the goal by the shorter of the alternative routes (i.e., via location "A"). The number of steps is noted. The animat is returned to “S” and location “B” is blocked. Goal “G” is again asserted with a priority of 1.0 and the behaviour of the animat noted. The animat is returned to “S”, “G” asserted and the resulting path noted.

6.5.2. Investigation One, Results and Analysis

Figure 6-13 shows the effect on animat behaviour of the procedure described for investigation one. The 1000 cycles of random walk provide ample opportunity for the animat to discover both available paths (figure 6-13a). Figures 6-13b, c and d show the animat path from “S” to “G” with no additional obstruction, the first run after location “B” is obstructed and the second run after “B” is obstructed respectively. This investigation was repeated with ten individual animats (rseed = 10, 20 .. 90, 100), the instance shown is with individual rseed = 10. With no dispersive noise and Lprob = 1.0 performance across these individuals is constant, the average first path length being 10 steps, and the third 16 steps. The average second path length is 39.7. Nine of the individuals took 39 steps. One 46 due to the appearance of a spurious shorter route μ-hypothesis introduced by handling during the procedure (the forced return move to “S” fell, by chance, in the lower right catchment area).

(A) 1000 steps random walk (seed = 10)

```
Cycle 1001: World is 6 by 9
  86   37   29   21   42   19   26   39   50
  68   23   12   10    7    8   13    8   14
  40   39   36X  25   22   20   16    4   16
  42 **** **** **** **** **** **** ****    9
  41   12   10    9    6    9    2    2    6
  43   12    6   10    5   12    6    7   22
Mean = 21.297873, std dev = 17.758427
```

(B) Trial One, “S” to “G”

```
Cycle 1012: World is 6 by 9
   0    0    0    0    0    0    0    0X   1
   0    0    0    0    0    0    0    0    1
   0    0    0    0    0    0    0    0    1
   0 **** **** **** **** **** **** ****    1
   0    0    0    0    0    0    0    0    1
   0    0    0    1    1    1    1    1    1
Mean = 0.234043, std dev = 0.437595
```

(C) Location “B” Blocked, “S” to “G”

```
Cycle 1051: World is 6 by 9
   1    1    1    1    1    1    1    1    1
   1    0    0    0    0    0    0    0    0X
   1    0    0    0    0    0    0    0    0
   1 **** **** **** **** **** **** **** ****
   1    1    1    1    1    1    1    1   13
   0    0    0    1    1    1    1    1    1
Mean = 0.847826, std dev = 1.876630
```

(D) Trial Three, “S” to “G”

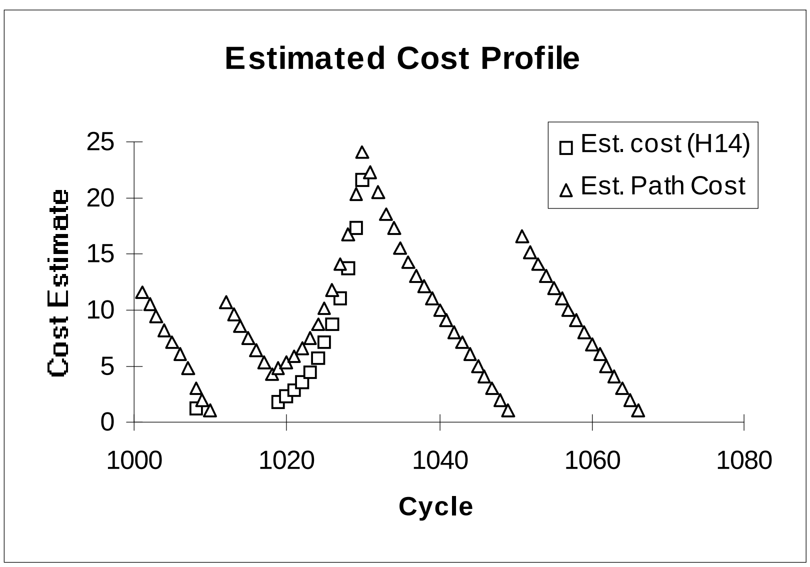
```
Cycle 1068: World is 6 by 9
   1    1    1    1    1    1    1    1    1X
   1    0    0    0    0    0    0    0    0
   1    0    0    0    0    0    0    0    0
   1 **** **** **** **** **** **** **** ****
   1    1    1    1    0    0    0    0    0
   0    0    0    1    0    0    0    0    0
Mean = 0.369565, std dev = 0.489010
```

monolith\figures.ppt:slide 7

Figure 6-13: Multiple Path Investigation, Individual rseed = 10

The mechanism by which SRS/E selects the original path, and then selects and stabilises on the new path after the obstruction is detected is straightforward. The first path is the lowest cost path computed by the Dynamic Policy Map from elements in the Hypothesis List. On the second trial run the DPM indicates the same path as run one. On reaching location X=8, Y=1 the previously reliable action "U" (from H14) fails, and the estimated cost of the step increases. The animat repeats this action until the estimated cost of the failed step raises the total estimated path cost above that for the alternative known route via location X=0, Y=2 (in the exemplar instance, 20.27). At this point the DPM is recomputed with the new shortest route and the animat pursues the new route to the goal.

Figure 6-14 details the cost estimate profile of the three valenced paths for the selected individual. The overall estimate for the remaining path is shown with triangle markers. The first series (cycles 1001 to 1011) shows the uninterrupted path from "S" to "G" via location "B". The second series (cycles 1012 to 1051) starts similarly to series one until the blocked location is detected. Estimated path cost increases as the cost contribution of the failed μ-hypothesis H14 increases (H14's contribution to the path cost is shown with square markers). Eventually the estimated cost of the preferred path exceeds that of the alternative, then the DPM policy estimates radically change and the animat follows the new path via location "A" without further interruption (cycles 1030 to 1051). The third series (cycles 1052 to 1067) confirms the preference for the new, longer, path.



monolith\results\chngwld\p14.xls

Figure 6-14: Estimated Cost Profile (Path and H14)

The apparent persistence with which the animat pursues the newly failed μ-hypothesis (H14) is determined primarily by the *extinction rate*, β. Within a normal population of individuals one might expect a range of values for this parameter and so the number of failed attempts to vary between individuals before the alternative path is adopted. The animat should not necessarily abandon its attempts at a known path too soon, as there are many circumstances where continued attempts are indeed better than not doing so. Mott's *ALP* robot controller being a case in point, the degree of persistence in goal seeking inadequately reflecting the rarity of the events sought. Other strategies could be proposed, including relating the degree of persistence rate to the existing quality and maturity of the μ-hypothesis in question.

6.5.3. Investigation One, Discussion

The ability of an animat to select an alternate, known, route if thwarted in pursuit of its preferred solution may appear as seemingly trivial. Yet this ability is an important discriminator between pure reinforcement learning systems and sensory-motor and intermediate level cognitive systems. Reinforcement learning systems (such as Dyna) which build a static policy map based on a current sensory pattern would not be expected to demonstrate the clear shift of behaviour presented by SRS/E, based as it is on a Dynamic Policy Map. Mimicking this ability therefore remains a challenge to conventional reinforcement learning systems. The distinction arises from the difference between categorising situations relative to a stable, but distant, reward and the encapsulation of situation and response as an independent unit disassociated from external reward.

6.5.4. Investigation Two (Goal Extinction), Procedure

This investigation determines the *goal extinction* behaviour of the animat when a single, known, path to the goal is obstructed, so that there is then no path to the goal. The animat is allowed to explore the environment shown in figure 6-12b for 1000 cycles (other conditions as for investigation one). The animat is returned to "S" and the goal location "G" asserted with a priority of 1.0. The animat's path to the goal noted. The animat is returned to "S", the location "B" blocked (so that there is no possible route to the goal) and goal "G" reasserted with priority 1.0. The behaviour of the animat in pursuing this unattainable goal is noted. The investigation is repeated with the initial conditions from investigation one (figure 6-12a), where there are two initially available paths, with both paths being blocked at the end of the period of random walk exploration. The behaviour of the animat is noted under these conditions.

6.5.5. Investigation Two, Analysis of Results

Figure 6-15 shows the stages in the goal extinction process. Sub-figures 6-15a and b show the initial stages for this investigation (for the individual rseed = 10), the random walk exploration and the demonstration of successful valenced goal seeking behaviour given an unblocked path. The path to the goal is blocked at this step, the animat returned to "S" and the goal "G" reasserted. Sub-figures 6-15c to h show the stages in the extinction process. Initially valenced goal seeking behaviour proceeds as normal. As there is no alternative path the animat repeats the failed μ-hypothesis (H14) until the estimated cost of the path exceeds that for the *valence break point* (VBP) value calculated from the original cost estimate (10.28) for the path. At this point the animat reverts to unvalenced behaviour for a period regulated by the *goal recovery mechanism*, figure 6-15d. This period of exploration allows the animat to discover some new and previously unknown path to the goal (it would have already tried other possible paths had they previously been identified during the exploration phase).

(A) 1000 steps random walk (seed = 10)

```
Cycle 1001: World is 6 by 9
  93   27   31   17   36   20   38   19   44
  52    9   13    7    5    8   14    7   15
  79   17   16    8    4    5   14   13   18
**** **** **** **** **** **** **** ****   32
  43   11   13   11   18   15X   8    8   11
  40   16   25   25   17   29   16   13   21
Mean = 21.760870, std dev = 17.820969
```

(B) Test Valenced Path, “S” to “G”

```
Cycle 1012: World is 6 by 9
   0    0    0    0    0    0    0    0   1X
   0    0    0    0    0    0    0    0    1
   0    0    0    0    0    0    0    0    1
**** **** **** **** **** **** **** ****    1
   0    0    0    0    0    0    0    0    1
   0    0    0    1    1    1    1    1    1
Mean = 0.239130, std dev = 0.442326
```

(C) Valenced to Step 1039

```
Cycle 1039: World is 6 by 9
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
**** **** **** **** **** **** **** **** ****
   0    0    0    0    0    0    0    0  21X
   0    0    0    1    1    1    1    1    1
Mean = 0.600000, std dev = 3.094799
```

(D) Unvalenced to Step 1140

```
Cycle 1140: World is 6 by 9
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
**** **** **** **** **** **** **** **** ****
  22  16X    6    3    4    9    1    1    7
  12    3    6    0    0    2    2    2    5
Mean = 2.244444, std dev = 4.553387
```

(E) Valenced to Step 1159

```
Cycle 1159: World is 6 by 9
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
**** **** **** **** **** **** **** **** ****
   0    1    1    1    1    1    1    1  12X
   0    0    0    0    0    0    0    0    0
Mean = 0.422222, std dev = 1.782632
```

(F) Unvalenced to Step 1360

```
Cycle 1360: World is 6 by 9
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
**** **** **** **** **** **** **** **** ****
  45   18    6    3   12    5   16    9    6
  27   14    1    1    3    6   12   13   4X
Mean = 4.466667, std dev = 8.615232
```

(G) Valenced to Step 1371

```
Cycle 1371: World is 6 by 9
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
**** **** **** **** **** **** **** **** ****
   0    0    0    0    0    0    0    0  10X
   0    0    0    0    0    0    0    0    1
Mean = 0.244444, std dev = 1.483240
```

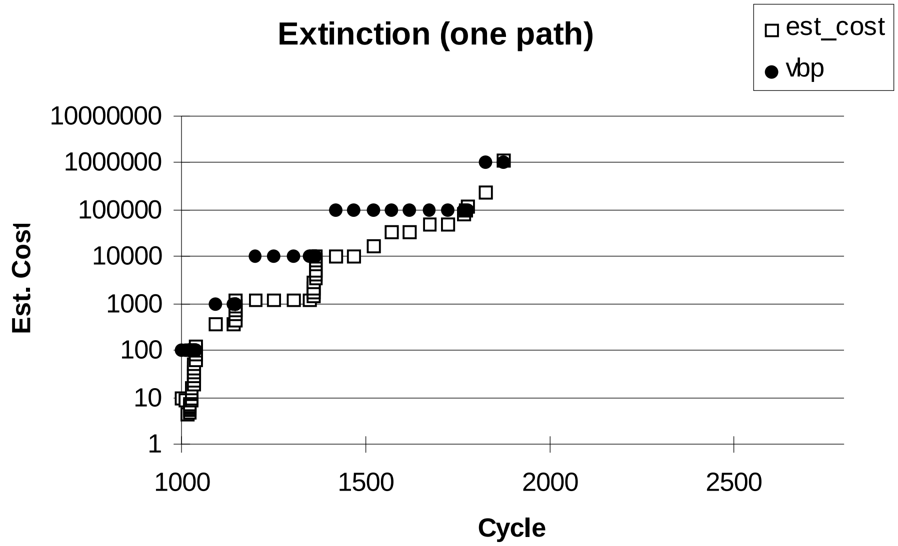
(H) Extinguished at Step 1593

```
Cycle 1593: World is 6 by 9
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0
**** **** **** **** **** **** **** **** ****
   0    4    6   10   10   12   11    9   43
   7    2    6   18  13X   11   11   23   26
Mean = 4.933333, std dev = 8.667949
```

monolith\figures.ppt:slide 8

Figure 6-15: Goal Extinction (rseed = 10)

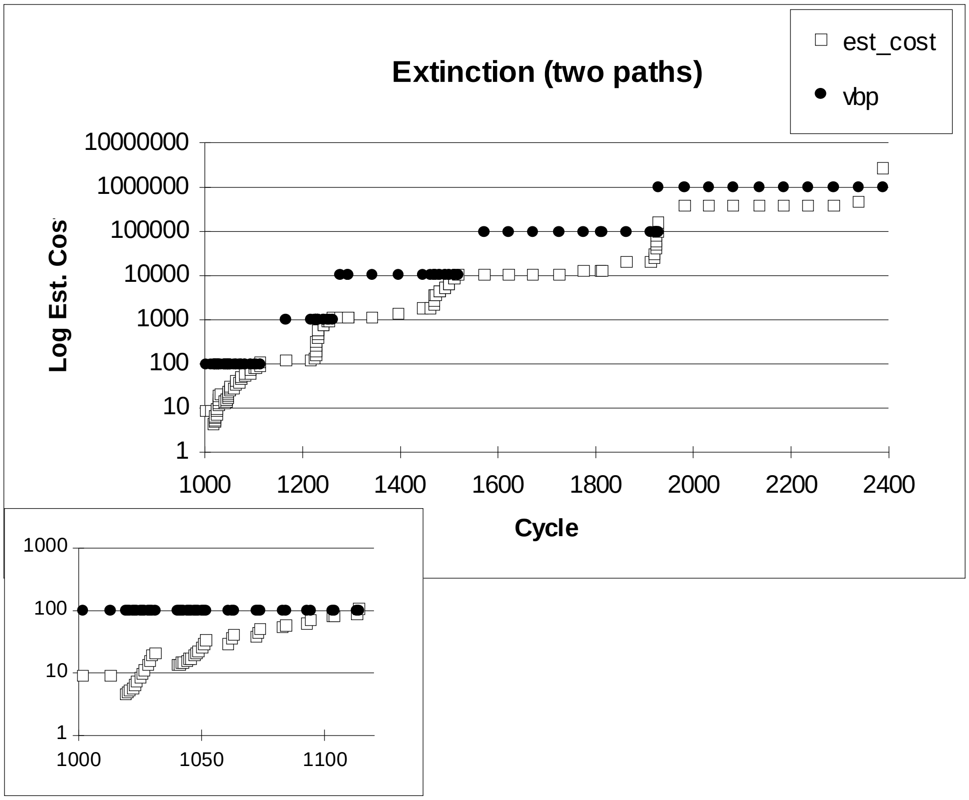
This process is repeated with alternating periods of valenced and unvalenced (trial and error) behaviour until the total cost estimate for the goal path exceeds the *goal cancellation level*, Ω, figure 6-15h. At this point g^1 is forcibly removed by SRS/E from the Goal List. The Innate Behaviour List $\mathcal{B}^g$ might reassert the goal, but to little useful effect. Figure 6-16 records the relative values of the cost estimate for the goal path and the computed value of VBP. Note in particular that the estimated cost rises quickly to meet the VBP at the end of each period of unvalenced behaviour. Note also that the estimated cost can rise during this unvalenced period due to the animat testing μ-hypothesis on the valenced path, but purely as a consequence of trial and error activities. This is particularly apparent in the latter stages of the extinction process and is in no small part due to the confined space in which the animat operates.



monolith\robtest\chngwld\extnct1g.xls

Figure 6-16: Goal Extinction, Comparison of Cost Estimate to VBP

This investigation was repeated with both paths ("A" and "B") available during the 1000 step random walk exploration phase (figure 6-12a). Both paths are then blocked before starting the extinction phase (as figure 6-12d). The animat behaviour is modified to appearing to scuttle back and forth between the two previously effective paths during the periods of valenced activity. Figure 6-17 shows the resulting estimated cost and VBP values of this investigation. The insert to the figure shows the detailed effect of this scuttling behaviour. Each rise in the cost estimate arises from the animat attempting the blocked μ-hypothesis, first at one end, and then at the other. The animat appears decreasingly persistent in its attempts to traverse each blocked path with each attempt. Gaps between the rises indicate the cycles during which the animat is (under valenced control) travelling between the two places where the known paths had been located. Note that the cost estimate and VBP are not shown during these periods as they are only recomputed when an event causes changes in Δ or δ that exceed `REBUILDPOLICYTRIP`. The net effect is to increase the number of cycles that elapse before goal extinction takes place. Over 10 separate trials (rseed = 10, 20 .. 100) the average time to extinction was 870.9 cycles for the single path case, and 1,443.2 cycles for this dual path case.



monolith\figures.ppt: slide 9 (monolith\robtest\chngwld\extnct2c.xls)

Figure 6-17: Goal Extinction (Two Path), Cost Estimate and VBP

6.5.6. Investigation Two, Discussion

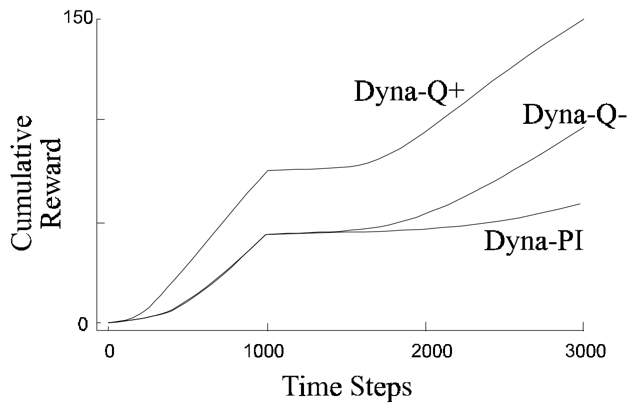
Goal extinction phenomena are well documented for natural learning, and are supported by a wealth of experimental data. The rate at which extinction takes place appears to be highly variable. Razran (1971, p. 167) points out that under some operant conditioning regimes pigeons will continue with ineffective pecking behaviour (introduced with food reward) for over 10,000 events, expending more energy than would have been obtained from the reward. Classical conditioning regimes tend to demonstrate much more rapid extinction phenomena (Razran posits a median conditioning-extinction ratio of 36:1). The number of unrewarded actions required to produce goal extinction appears to depend on many factors including experimental conditions and procedures, the nature of the reward, its presentation and subject animal.

The onset of extinction can be continuously delayed by occasional reward (as in variable reward ratio regimes). Such is also the case in SRS/E where a single valid prediction restores the value of `bpos` for any μ-hypothesis disproportionately to the effect of a failed prediction. In its current implementation SRS/E does not demonstrate any spontaneous recovery of extinguished valenced behaviour. Such phenomena might be implemented by either an explicit second order term in the cost estimate function or by the inclusion of a specific *habituation* process disadvantaging μ-hypotheses used repeatedly. This would reflect Hull's approach to the extinction process (section 2.2, eqn. 2-1).

The presentation of data in figures 6-16 and 6-17 mirrors that for experimentally observed extinction patterns in animals (figure 3-1). Note that while these two presentations appear superficially similar they are not directly comparable, though they may indicate a similarity in underlying mechanism. The data in the figures presented in this chapter record internal values, those for animal experiments record externally observed events. Extinction in natural learning is a subtle phenomenon, no doubt deserving of a more sophisticated model that currently provided for in the SRS/E algorithm.

6.5.7. Investigation Three (Path Blocking), Procedure

This investigation determines the behaviour of an animat when faced with a block to a known path, but where a previously unknown path is simultaneously made available. To locate the new path the animat must balance exploration of the environment with exploitation of the previously known, and successful, solution path. In this investigation the animat is allowed a period of 1,000 cycles of continuously valenced activity using the maze shown in figure 6-12b (shorter path). The animat is always started at "S", with "G" asserted as goal. Once the animat reaches "G", it is returned to "S" and "G" reasserted. The other investigations in this experiment allowed random walk exploration during this initial phase. As in pervious experiments a small number of run-on cycles are permitted to ensure SRS/E may learn the steps leading directly to the goal. At cycle 1000 the location "B" is blocked and the previously blocked location "A" opened. The animat must discover the new path and continue to traverse from "S" to "G" as in the first phase of the investigation. Figure 6-18 shows the results obtained by Sutton (1990) for this blocking task with the Dyna family of reinforcement learning algorithms. The procedure used here follows that employed by Sutton. Effects of slight variations in experimental procedure will be noted and discussed. The procedures for this investigation are available as a fixed schedule within SRS/E.



Graphic 5.21 from monolith\dyna.cdr

Figure 6-18: Average Performance of Dyna Systems on a Blocking Task

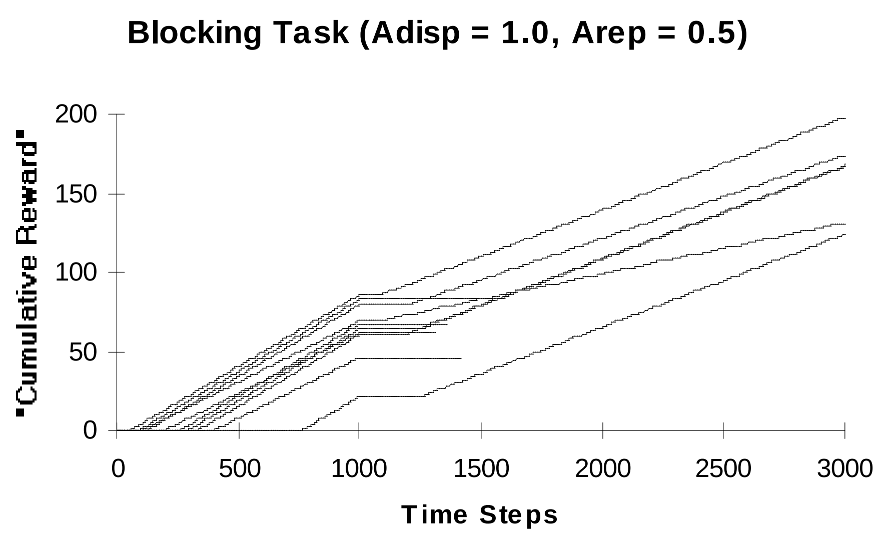
From Sutton (1990), p 222.

This investigation retains a cumulative record of the number of visits to the goal location, referred to as *cumulative reward* in figure 6-18. The slope of the line reflecting the frequency with which the goal is achieved. The shorter path allows the slope to be steeper, a flat period indicates a section in the investigation during which no “reward” is received, after location “B” is blocked and “A” opened. Results are plotted as curves recording individual animat performance and as an average of many individuals. Results for SRS/E are obtained with no dispersive noise (Adisp = 1.0), and with 10% dispersive noise (Adisp = 0.9).

6.5.8. Investigation Three, Results and Analysis

Figure 6-19 shows 10 individual performance curves for the conditions described by Sutton for the path blocking experiments (rseed = 10, 20 ... 100). As with figure 6-18 the slope of each curve indicates the path length from “S” to “G”, the steeper the slope the more frequently the goal is visited. This form of presentation is analogous to that often used in *Skinner box* experiments to record the bar pressing activity of experimental animals in relation to reward delivery. Flat sections on a curve indicate periods where no reward is obtained. The first flat section indicates the initial random walk trial and error path to the goal. As Lprob is set to 1.0 in these investigations the slope of the curve represents the length of the learned path (sometimes optimal, 7 cases of ten, sometimes not).

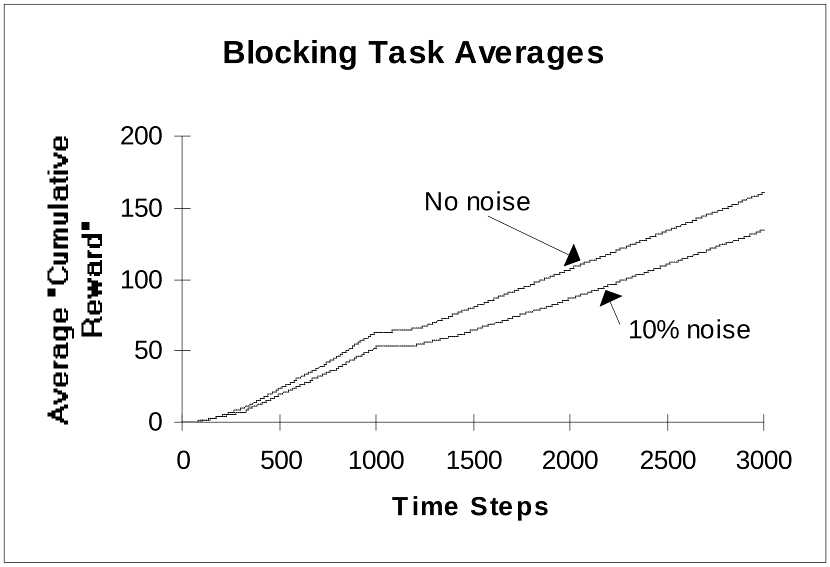
The second flat portion represents the time taken for the longer path to be located by trial and error random walk during the unvalenced parts of the goal extinction process. In four of the ten instances (individuals with rseed = 10, 50, 60 and 80) goal extinction took place before the alternative route was located. The cumulative curve ends abruptly in these cases. Members of the Dyna family of systems do not employ this mechanism. Of the remaining six individuals four found the shortest path from “S” to “G”.



monolith\robtest\blocking\block1.xls

Figure 6-19: Investigation Three, Individual “Cumulative Reward” Curves

Figure 6-20 shows the averaged results of the ten individual trials described above. The performance of SRS/E under these conditions is comparable with the best of the Dyna series, *Dyna-Q+*, under similar experimental conditions (see discussion below). Addition of 10% dispersive noise (lower curve) has a consistently adverse effect on the performance of this system. The advantage of any additional exploratory effect being completely masked by the extra effort required to reach the goal. This finding appears consistent with previous conclusions about the effects of dispersive noise.



monolith\robtest\blocking\averages.xls

Figure 6-20: Investigation Three, Average "Cumulative Reward" Curves

6.5.9. Investigation Three, Discussion

Being fully aware of the difficulties of taking accurate measurements from a published graph (figure 6-18), a line drawn tangential to the first portion of the Dyna-Q+ curve indicates a slope of 10.76 steps/reward and for the second portion of the curve a slope of 18.2 steps/reward. Minimum path lengths are 10 and 16 respectively. Compensating for run-on cycles called for in the current experimental procedures, SRS/E attains average slope values of 10.6 and 18.33 respectively. It would be unreasonable to directly compare the total number of cumulative rewards at cycle 3000 (about 150 for Dyna-Q+, 160.33 for SRS/E) as the four worst instances in SRS/E were abandoned due to the extinction process. By adjusting the parameters involved SRS/E could be tailored to allow greater periods of random walk exploration during the unvalenced stages of the goal extinction process.

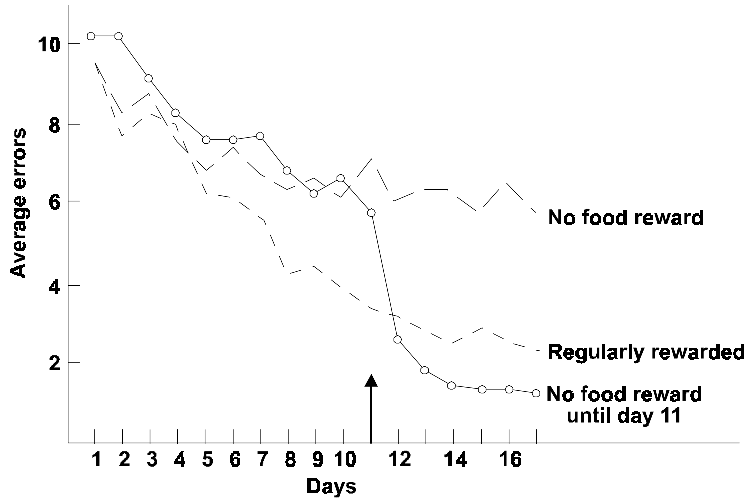
Sutton also tested members of the Dyna family of systems on a shortcut task. Animats were set a repeated goal seeking task using maze C (figure 6-12) in which only the longer path via "A" is available initially. After 3,000 cycles the shorter path "B" is also made available. Dyna-Q+, with its additional exploration component demonstrated some improvement in performance, indicating the shorter path had been discovered and adopted. SRS/E has no explicit mechanism for exploration during valenced goal seeking behaviour. Consequently, if SRS/E is continuously tasked it will always adopt the best known path. Such wilful overtasking is a pathological case for SRS/E, the system expects to be presented with a range of tasks and to have periods where no goal is asserted. Under such conditions SRS/E has every opportunity to locate and subsequently employ the shortcut route.

6.6. Latent Learning

The demonstration of *latent learning* phenomena was a significant step in the historical development of learning theory. Each of the major behaviourist learning theories is based on the notion that learning takes place in response to a reward (or conversely a punishment). If it were to be demonstrated that learning had occurred without any reward then the findings of the behaviourist school would be called into question. Clearly a demonstration of this type would have suited Tolman in the promotion of his expectancy theory.

A classic "latent learning" experiment is replicated with SRS/E. In the original Tolman and Honzik (1930) tested three groups of food deprived rats in a maze apparatus. The first group were allowed to wander the maze and obtained a food reward at the end location. The second group were allowed to wander the maze, but on reaching the end location they received no food reward. Each rat was placed in the maze once per day before being returned to their normal accommodation. Once the rat had reached the end location it was prevented (by a one-way door) from re-entering the body of the maze. Sufficient time was allowed in the end location to prevent any reward effects associated with food availability in their normal accommodation. On the eleventh day (i.e., after 11 runs through the maze) the second group were given access to food reward in the end location. A third, control, group was allowed to run the maze with no food reward throughout the duration of the experiment.

Tolman found that the performance of the second group on the twelfth daily run (the first after the introduction of reward) was as good as or better than that on the first group that had been rewarded on every run, who had shown a gradual improvement in performance. Tolman's maze was constructed from 14 multiple T units, with doors between the units to prevent the rats retracing their steps in the maze. Tolman interpreted this as clear evidence that reward was not required for learning to take place. Tolman and Honzik's results are reproduced in figure 6-21. The measure of performance is the number of errors made by the experimental animal in traversing the maze.



Graphic from monolith\latent.cdr

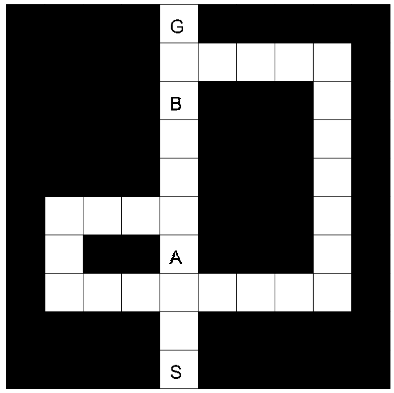
Figure 6-21: Tolman and Honzik's Latent Learning Results

adapted from Bower and Hilgard (1981, p. 338)

6.6.1. Description of Procedure

A latent learning schedule is available as a fixed procedure in the SRS/E program. Figure 6-22 shows the experimental environment selected for this investigation. It is characterised by having three distinct paths of varying length from the defined start "S" to defined goal or finishing location "G". The maze arrangement used here differs from that of Tolman and Honzik.

In the procedure 100 "clone" animats are selected for each of the three groups (i.e., each of the three groups comprises 100 individuals with rseed = 1000, 1001 ... 1099). All 16 traversals of the maze by the first group are valenced. The first 11 traversals of the second group are unvalenced, but the twelfth and subsequent traversals are. All traversals by the control group are unvalenced. The essential parameters are: Arep = 0.5, Adisp = 1.0, Lprob = 0.25, the other learning parameters are standard.



Graphic 5.25 from monolith\mazes.cdr

Figure 6-22: The SRS/E Latent Learning Environment

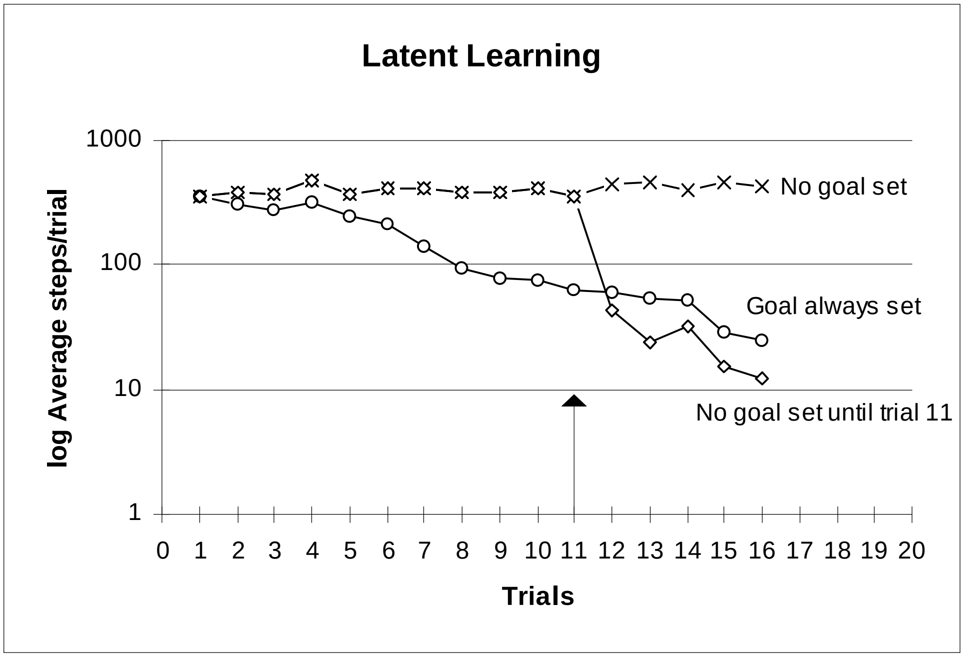
6.6.2. Results and Analysis of Experiment

Figure 6-23 shows the results of the experiment, indicating that the essential properties of the Tolman and Honzik experimental results are present. The first group show a gradual improvement in performance throughout the procedure. The second group show a dramatic improvement following the introduction of goal valencing. The third, control, group shows no significant change in performance. Note the different representation of performance, steps/trial rather than errors. A logarithmic representation of the performance axis has been used for cosmetic reasons. Neither of these factors should materially affect the interpretation of the results.

The gradual improvement seen in the control group of Tolman and Honzik's results is not replicated by SRS/E. This might be interpreted as evidence that some other form of reinforcement is available to the animal prior to the main reward (Bower and Hilgard, 1981, p. 339). Alternatively it might be noted that rats (and many other mammals) show a quite distinct *curiosity*[29], seeking out the novel and then ignoring it once it is no longer novel. The design of Tolman and Honzik's maze has many dead-ends, which once discovered can be safety ignored in

[29]As MacCorquodale and Meehl (1953, p. 204) put it "*No one who has observed rats during their early exposure to a maze could dismiss the exploratory disposition as of negligible strength*".

subsequent traversals of the maze - leading to a reduction in measured error rate. SRS/E differs in that it responds to novelty, but does not seek it out. An additional mechanism, such as *prioritized sweeping* of Moore and Atkeson (1993), might be adapted for use in SRS/E to demonstrate the gradual improvement findings in the control group.



monolith\results\latent\lat100.xls

Figure 6-23: Results of the SRS/E Latent Learning Experiment

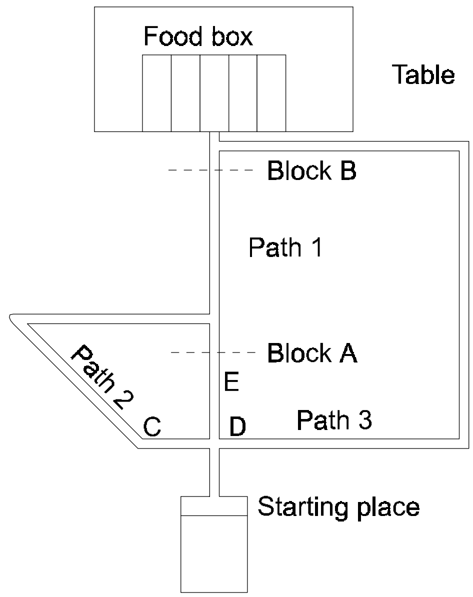
6.6.3. Discussion

That SRS/E should demonstrate latent learning is hardly in doubt, nor a surprise. Reinforcement is generated internally, and is not dependent on external reward. Given the revival of interest in behaviourist and reinforcement learning methods for machine learning models it is nevertheless a timely reminder that these are well-trodden paths. Latent learning has been extensively researched. Thistlethwaite (1951) identifies and evaluates over 30 different latent learning experiments under a variety of different experimental conditions. MacCorquodale and Meehl (1953) placed considerable emphasis on the latent learning phenomenon, indeed stating that it provided the main motivation to add their contribution toward the formalisation of expectancy theory. MacCorquodale and Meehl note that not all experiments to demonstrate latent learning actually do so, in part, no doubt, due to variations in experimental design and procedure. Observation of the latent learning phenomenon places a considerable strain on behaviourist and reinforcement based theories, whereas the absence of the phenomenon has little impact on expectancy based models.

SRS/E's demonstration of the latent learning phenomena arises from one by now well explored propensity - to pursue a known route to a valenced goal in preference to exploring for a possible better alternative. With group one (always valenced) some, typically small, proportion of the individuals traverse the maze to the goal location by one of the longer paths during the first trial. Once they have that path, those individuals tend to continue to use it, as their behaviour is always valenced while in the maze. Gradual improvement in performance is a consequence of the choice of Lprob = 0.25, and is consistent with the learning rates previously shown in the baseline investigations of figure 6-2. Group two has adequate opportunity to explore the maze by random walk during the 11 unvalenced trials. Once the goal location becomes valenced individual animats have invariably encountered, and so use, the shortest route. Consequently, on average, the performance of group two exceeds that for group one, once the goal is valenced. The control group have no reason to treat the "goal" differently from any other location, and show no performance improvement.

6.7. Place Learning

Tolman also devised a *place learning* experiment, again using rats in an experimental maze to demonstrate what he referred to as "inferential expectation" or "insight" in these animals (Tolman and Honzik, 1930b). In this classic demonstration experimental rats were placed in a maze of the form shown in figure 6-24. With adequate experience of the maze rats show a clear preference for the shorter of the available routes, path 1. When path 1 was blocked the rats showed a distinct preference for path 2 and when path 2 was also blocked, then the rats would adopt the longer path 3. The key to the experiment is the placing of the block. Tolman argued that if the block was placed at point B a rat guided by blind habit would first try path 2, its choice at this decision point being directed by the response previously associated with the stimulus at that point. However, one capable of cognitive "inferential expectation" or "insight" would conclude that the block also affected path 2 and would consequently employ path 3 directly. He found this to be the case.



Graphic 5.27 from monolith\tolmaze.cdr

Figure 6-24: Tolman and Honzik's "Insight" Maze

adapted from Bower and Hilgard (1981, p. 337)

6.7.1. Description of Procedure

These "insight" experiments are replicated with SRS/E using the experimental environment of figure 6-22. The procedure replicates the major functional features of Tolman and Honzik's "insight" maze. In the replication of this experiment naïve animats are allowed to explore the maze for 2,000 cycles by unvalenced random walk. This allows sufficient time for the animats to explore every path. Each animat is then given one valenced trial from "S" ("G" asserted as goal) with no path blocked to confirm that the animat will select the most direct route. In the next step the location at point "A" is blocked. The animat is returned to "S" and "G" is valenced. The number of steps required to traverse the environment to the goal is noted. The animat is returned to "S", "G" is valenced and the number of steps required to reach the goal location again noted. In the next step the block at location "A" is removed and a block added at location "B", the animat is returned to "S". The goal location "G" is valenced and the number of steps to traverse the modified environment noted. The animat is returned to "S" and the number of steps to complete another valenced traversal to the goal location again noted. This experiment uses the standard learning parameters and Arep = 0.5, Adisp = 1.0, Lprob = 1.0.

6.7.2. Results and Analysis of Experiment

Figure 6-25 shows the performance in this experimental procedure by a single individual (rseed = 10). Sub-figure 6-25a confirms that each path has been fully explored, though by no means evenly. Sub-figure (b) confirms the animat takes the direct route when "G" is valenced. Sub-figure (c) shows the effect of the first valenced run after block "A" is set. After 10 failed attempts to traverse path 1, the animat proceeds along path 2, as Tolman would have predicted. Sub-figure (d) confirms the new path on the next valenced run. Sub-figure (e) shows the effect of the first valenced run after block "A" is cleared and block "B" set. As the animat is valenced it follows the known available route (via path two) until the unexpected block is encountered at "B". After a number of failed attempts to traverse the now blocked location "B" the animat backtracks down path one and round to the goal location via path three. The still longer path involving path two is ignored. Sub-figure (f) confirms the new route via path 3 on the next valenced run.

(A) 2000 steps random walk (seed = 10)

```
Cycle 2001: World is 10 by 10
**** **** **** ****  337 **** **** **** **** ****
**** **** **** ****   55    5    3   22   15 ****
**** **** **** ****   84 **** **** ****    6 ****
**** **** **** ****   73 **** **** ****    1 ****
**** **** **** ****   84 **** **** ****   19 ****
****   25    4   21  119 **** **** ****   18 ****
****    6 **** ****  141 **** **** ****    6 ****
****   92   51   77   70   67   64    7   33 ****
**** **** **** ****  221X**** **** **** **** ****
**** **** **** ****  198 **** **** **** **** ****
Mean = 64.133331, std dev = 75.047981
```

(B) Confirm Path 1

```
Cycle 2011: World is 10 by 10
**** **** **** ****    1X**** **** **** **** ****
**** **** **** ****    1    0    0    0    0 ****
**** **** **** ****    1 **** **** ****    0 ****
**** **** **** ****    1 **** **** ****    0 ****
**** **** **** ****    1 **** **** ****    0 ****
****    0    0    0    1 **** **** ****    0 ****
****    0 **** ****    1 **** **** ****    0 ****
****    0    0    0    1    0    0    0    0 ****
**** **** **** ****    1 **** **** **** **** ****
**** **** **** ****    1 **** **** **** **** ****
Mean = 0.333333, std dev = 0.483046
```

(C) Add Block "A"

```
Cycle 2036: World is 10 by 10
**** **** **** ****    1X**** **** **** **** ****
**** **** **** ****    1    0    0    0    0 ****
**** **** **** ****    1 **** **** ****    0 ****
**** **** **** ****    1 **** **** ****    0 ****
**** **** **** ****    1 **** **** ****    0 ****
****    1    1    1    1 **** **** ****    0 ****
****    1 **** **** **** **** **** ****    0 ****
****    1    1    1   10    0    0    0    0 ****
**** **** **** ****    1 **** **** **** **** ****
**** **** **** ****    1 **** **** **** **** ****
Mean = 0.862069, std dev = 1.800383
```

(D) Confirm Path 2

```
Cycle 2052: World is 10 by 10
**** **** **** ****    1X**** **** **** **** ****
**** **** **** ****    1    0    0    0    0 ****
**** **** **** ****    1 **** **** ****    0 ****
**** **** **** ****    1 **** **** ****    0 ****
**** **** **** ****    1 **** **** ****    0 ****
****    1    1    1    1 **** **** ****    0 ****
****    1 **** **** **** **** **** ****    0 ****
****    1    1    1    1    0    0    0    0 ****
**** **** **** ****    1 **** **** **** **** ****
**** **** **** ****    1 **** **** **** **** ****
Mean = 0.551724, std dev = 0.525226
```

(E) Remove Block "A", Add Block "B"

```
Cycle 2098: World is 10 by 10
**** **** **** ****    1X**** **** **** **** ****
**** **** **** ****    1    1    1    1    1 ****
**** **** **** **** **** **** **** ****    1 ****
**** **** **** ****   15 **** **** ****    1 ****
**** **** **** ****    2 **** **** ****    1 ****
****    1    1    1    2 **** **** ****    1 ****
****    1 **** ****    1 **** **** ****    1 ****
****    1    1    1    2    1    1    1    1 ****
**** **** **** ****    1 **** **** **** **** ****
**** **** **** ****    1 **** **** **** **** ****
Mean = 1.586207, std dev = 2.559633
```

(F) Confirm Path 3

```
Cycle 2116: World is 10 by 10
**** **** **** ****    1X**** **** **** **** ****
**** **** **** ****    1    1    1    1    1 ****
**** **** **** **** **** **** **** ****    1 ****
**** **** **** ****    0 **** **** ****    1 ****
**** **** **** ****    0 **** **** ****    1 ****
****    0    0    0    0 **** **** ****    1 ****
****    0 **** ****    0 **** **** ****    1 ****
****    0    0    0    1    1    1    1    1 ****
**** **** **** ****    1 **** **** **** **** ****
**** **** **** ****    1 **** **** **** **** ****
Mean = 0.620690, std dev = 0.491304
```

monolith\figures.ppt:slide 10

Figure 6-25: Results from "Insight" Experiment

As with the latent learning experiment the key to successful demonstration of the phenomenon under investigation is careful experimental layout and procedure. Where the latent learning procedure called for careful rationing of experience in the maze during the initial stages of the sequence, this procedure calls for adequate exploration. Without this the various routes may not be fully known to the animat, and consequently it will not select the preferred (by the experimenter in this case) routes. Other researchers subsequently found Tolman and Honzik's results repeatable, but prone to disruption, apparently due to elements in experimental design.

6.7.3. Discussion

SRS/E confirms Tolman's view of "insight". It seems unlikely that Tolman will have won much approbation from his peers by the use of the term, implying as it does, a level of intelligence well above that normally associated with the laboratory rat. Perhaps paradoxically, and with the benefit of hindsight, we may see that this behaviour is fully explicable in terms of problem solving, at best a minor form of “insight”. Nevertheless, the capabilities demonstrated by Tolman’s rats and replicated by the SRS/E algorithm in this procedure still present considerable difficulties to the behaviourist and reactive agent schools of thought that promote reinforcement learning by explicit reward.

6.8. Chapter Summary

This chapter has described a series of experiments that investigate the properties of the SRS/E algorithm as an implementation of the Dynamic Expectancy Model. To facilitate direct comparison with previously published algorithms, Sutton’s (1990) Dyna family of reinforcement learning programs, the experimental conditions employed for those previously published works have been replicated. In the baseline investigations of section 6.2 the performance of the SRS/E algorithm was directly compared to that of Sutton’s *Dyna-PI* algorithm. SRS/E shows a marked performance gain over Sutton’s algorithm. Under “ideal learning conditions” SRS/E was clearly able to master the maze traversal problem within a single trial (the Lprob = 1.0 curve of figure 6-2), whereas Dyna-PI is recorded as requiring over 80 trials (the “zero planning steps” curve of figure 6-1). It may be estimated that this represents approximately a forty-fold improvement in learning efficiency, in terms of the overall number of steps required to master the given task. The improved curves shown for Dyna-PI are achieved by added internal computation, the degraded curves for SRS/E are created by restricting the effectiveness of the learning process (Lprob < 1.0).

Sutton did not report on the performance of Dyna-PI in the noise disrupted environment he described. However, these investigations were performed with the SRS/E algorithm, and are reported in section 6.3. The results obtained are summarised in figures 6-4 and 6-5. The figures demonstrate that while the rate at which the task is learned is not markedly affected by the addition of this form of noise, the overall learned task performance is degraded by the presence of the noise. It was subsequently argued in section 6.3.2.2 that the Dynamic Policy Map is indeed correctly formed by the learning process. It is the task performance that is disrupted by the presence of noise in the test trials. When this noise is removed, animat task performance is restored to near optimal levels.

The alternative and multiple goal experiments described in section 6.4 highlight a significant difference between the Dynamic Expectancy approach and that of conventional *Q*-learning algorithms. By recomputing the policy map on demand it becomes clear that <u>any</u> sign known to the system may be treated as a goal and selected on some arbitrary basis, not just those signs that were assigned as goals during the learning process. The SRS/E algorithm may therefore address situations where the animat is faced with goals that vary over time, and where several goals, of varying priority, must be tackled in an appropriate order.

The investigations of section 6.5 explored the response of the SRS/E algorithm to a variety of situations in which different paths from a starting point to a fixed goal point are presented to the animat. These tasks are essentially beyond the capabilities of conventional *Q*-learning algorithms of the form described by Watkins (1989). The performance of Sutton's *Dyna-Q+* algorithm, an adaptation of the *Q*-learning approach, was compared directly with the unmodified form of the SRS/E algorithm. Even though the mechanism by which new paths are discovered is radically different in the two algorithms, the apparent recorded performance was generally very similar. This is something of a surprise, as it might be thought that the inclusion of a continuously active exploratory component in the Dyna-Q+ algorithm would degrade its otherwise optimal levels of performance. Exploration is only invoked in SRS/E when an obstruction to the policy map path is encountered. The provision of an extinction mechanism in the SRS/E algorithm is a radical departure from the Dyna approach, and has some biological plausibility.

The demonstration of latent learning, described in section 6.6, highlights a substantive difference between the Dynamic Expectancy Model and previous reinforcement learning techniques. Learning is demonstrated to take place in the absence of external reward. This result, for which there is a substantial body of corroborating literature from animal learning experiments, would be wholly unexpected from a conventional reinforcement learning mechanism.

Similarly the place learning experiments, described in section 6.7, demonstrate the ability of the SRS/E algorithm to negotiate obstructions in its policy path in a manner that would be unpredicted from any algorithm employing a static policy map. Again, these results are consistent with findings from well-established animal learning experiments.

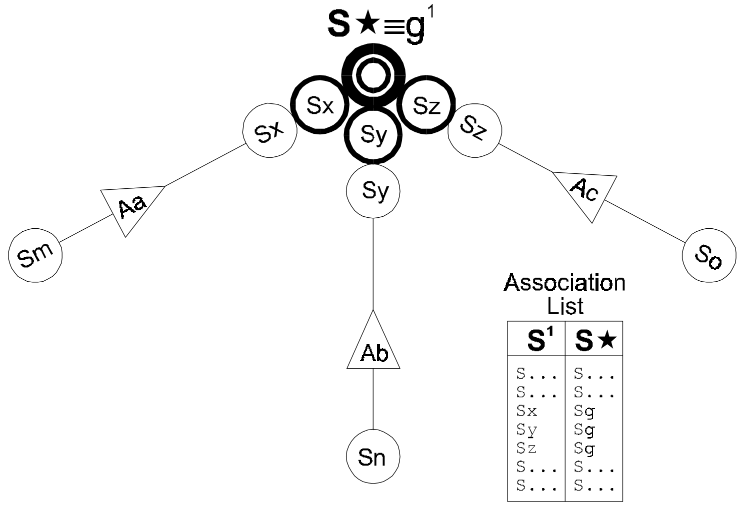
Chapter 7

7. Extensions to SRS/E and Further Work

SRS/E is an experimental system. By their nature such experimental systems are vehicles for extension and enhancement. The SRS/E algorithm is a working and workable implementation of the Dynamic Expectancy Theory, but there is scope for additional capability. This section describes a small number of the possibilities.

7.1. An Association List

A component part of MacCorquodale and Meehl's interpretation of Tolman's *expectancy theory* proposed a separate sign to sign associative effect (denoted "$S_2S^\star$"). Such pairings may in particular record the association of arbitrary signs (S_2) to signs ($S^\star$) specifically identified as relating to desirable goal situations; the *secondary cathexis* postulate. The creation of a separate *Association List*, $\mathcal{A}$, within SRS/E would allow the attachment of multiple (secondary) goal states to a single (primary) goal definition. Signs detected as occurring concurrently with, or slightly preceding (giving a predictive element to the association) a predefined goal sign would be paired with the desired sign and this association saved on $\mathcal{A}$, figure 7-1. The strength of this association being subject to strengthening by *mnemonization* and weakening by extinction processes based on the frequency and temporal adjacency of the pairing.



Graphic 6.1 from monolith\dpmex.cdr

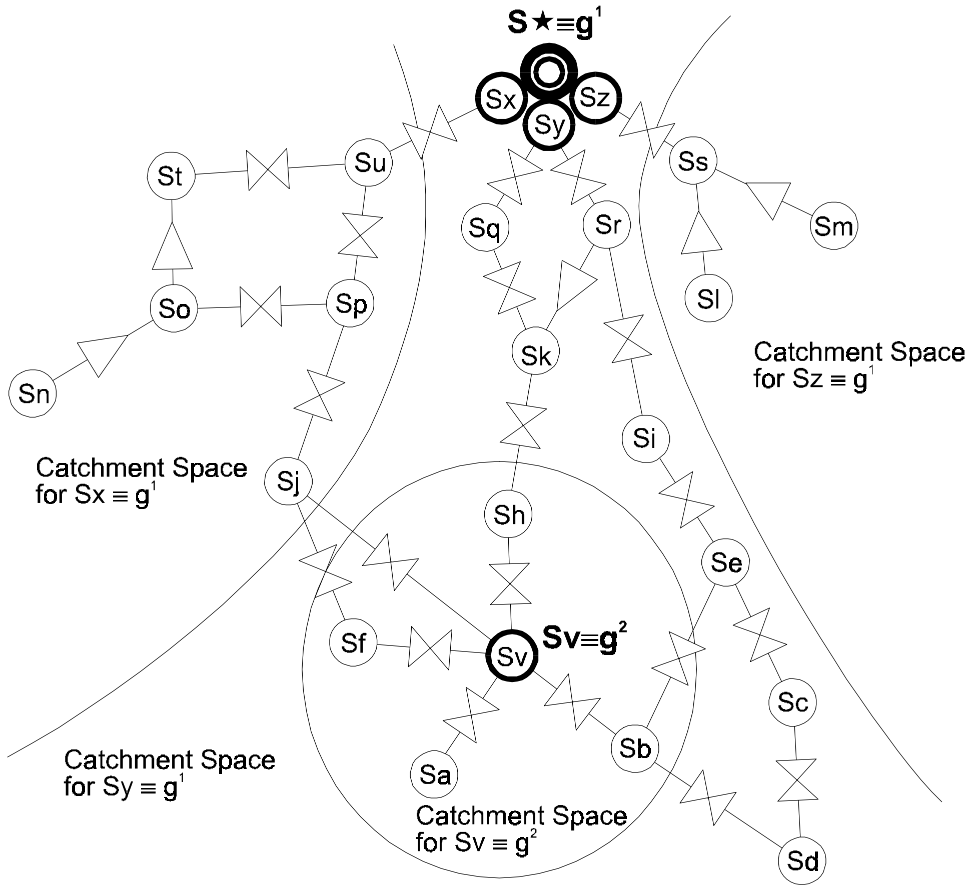
Figure 7-1: Sign-Sign Associations (Secondary Cathexis)

This arrangement allows greater flexibility in selecting goals from the Behaviour List, as there is no longer a requirement for the originator to identify specific tokens or signs to describe the goal. This form of association is different from the association phenomena described in the *classical conditioning* literature, in that it is not dependant on an unconditioned response (UR). The $S_2S^{\star}$ *sensory preconditioning* effect has been demonstrated under controlled experimental conditions, what relationship it may or may not have to classical conditioning phenomena is a matter of some conjecture. Bower and Hilgard (1981, pp. 330-331) review some of the evidence.

7.2. Seeking Multiple Goals Simultaneously

Multiple goals may be pursued in a more effective manner than the sequential strategy currently employed by SRS/E. Given several goals active on $\mathcal{G}$, the SRS/E algorithm currently actively seeks the top-goal, and will pass secondary goals by, regardless of how close they are to the current path, or of the overall estimated cost of achieving the main goal and subsequently continuing to the secondary one. This was demonstrated in section 6.4. The algorithm normally takes the path of least estimated cost to the top-goal. Where a secondary goal is on the path, by either good-fortune or chance, then it is satisfied in passing.

Changes to the goal seeking process may be implemented either by building a single DPM where the action selected depends on both cost to goal and relative *goal priority*, or by computing several DPMs, and selecting an action on the basis of some, as yet undetermined *goal strength function*, *f*(`estimated_cost`, `goal_priority`), thus combining cost and priority. This would allow the animat to divert to secondary goals when they are close to the primary path. This, coupled to the proposed Association List, allows several paths to the desired specified goal state to be defined and pursued concurrently. Figure 7-2 illustrates the concept.



Graphic 6.2 from monolith\dpmex.cdr

Figure 7-2: Enhanced Goal Acquisition

In this example each goal (or goal by association) has a "catchment area", defined by the goal strength function. For each recomputation of the Dynamic Policy Map every sign in S will fall within the catchment area of one of the prioritised goals. So in this example if "Sb" was active ("Sb*"), the animat would use the μ-hypothesis "Hbv" to satisfy the lower priority goal g^2, even if the path "Sb*"-"Se" represented the lowest estimated cost path to g^1. The animat would then proceed to the original g^1, possibly via the path "Sh"-"Sk" and so on. In the current implementation the animat selects from the available alternative paths "Sb*"-"Sv", "Sb*"-"Se" or "Sb*"-"Sd" entirely according to the lowest estimated cost path to g^1, and so may increase the total path to satisfy both goals unnecessarily. Even in the proposed regime the animat would pursue the path "Sd*"-"Sc" if that represents the lowest cost path, as "Sd" falls outside the catchment area defined for "Sv".

Goodwin and Simmons (1992) describe a decision theoretic approach to the balancing of multiple goals for a *HERO 2000* series mobile robot. Haigh and Veloso (1996) describe *Rogue*, a system for generating and executing plans with multiple interacting goals, where goal tasks may be interrupted or suspended.

7.3. An Explicit Template List

This extension to the SRS/E algorithm proposes an additional list type, the *Template List*, $\mathcal{T}$, to record the pattern of signs and actions used to build a new μ-hypothesis. Templates may at first be created at random, much in the manner that μ-hypotheses are in the present version of SRS/E. After a period of corroboration the effectiveness of each template may be assessed by reference to the confidence measures of the μ-hypotheses it was responsible for creating. Future bias being then given to those templates that are demonstrated to give rise to successful μ-hypotheses. This *meta-level learning* may be instrumental in explaining *learning-to-learn* phenomena described in the natural learning literature (although these phenomena may also be in part due to an increase in overall competence). The provision of a Template List would further allow the originator to bias the learning strategy of the animat according to pre-conceived notions of an intended environment or behavioural strategy.

The provision of a separate Template List equates, in some small measure, to *Popper's* notion of a "theory". Individual μ-hypotheses are generated from these meta-level objects, and in turn these meta-level objects may be judged according to the performance of their generated descendants.

7.4. Directing Learning Effort

The SRS/E algorithm is an implementation of an expectancy theory, reinforcement for individual μ-hypotheses is contingent upon their effectiveness as a predictive element. This reinforcement is not, in the system and experiments so far described, contingent on any notion of the value (as defined in the ethogram or elsewhere) in achieving goals defined for the system. There is a huge body of evidence that learning is indeed contingent upon the achieving a "desired" outcome (i.e. one which "reinforces".) An absolute distinction between predictive outcome and desirability is therefore an unnecessary one, and ultimately potentially disadvantageous to the system.

MacCorquodale and Meehl (1953, pp. 238-239) suggest increasing the expectancy-growth strength to a greater rate according to valence level. This is equivalent to increasing the value of the learning rate parameter α when a reward is detected as a result of satisfying a highly valenced prediction. In practice, adopting this strategy will have only a marginal effect on the system's overall observable behaviour. It also serves to confound two quite separate issues - the reliability of an expectancy and the usefulness of an expectancy. The reliability (as reflected in the various confidence measures) of the μ-hypothesis is properly determined by the ratio of successful to unsuccessful predictions, as has been the case. If an outcome is useful, then emphasis should be placed on the acquisition of μ-hypotheses that achieve it either directly or indirectly.

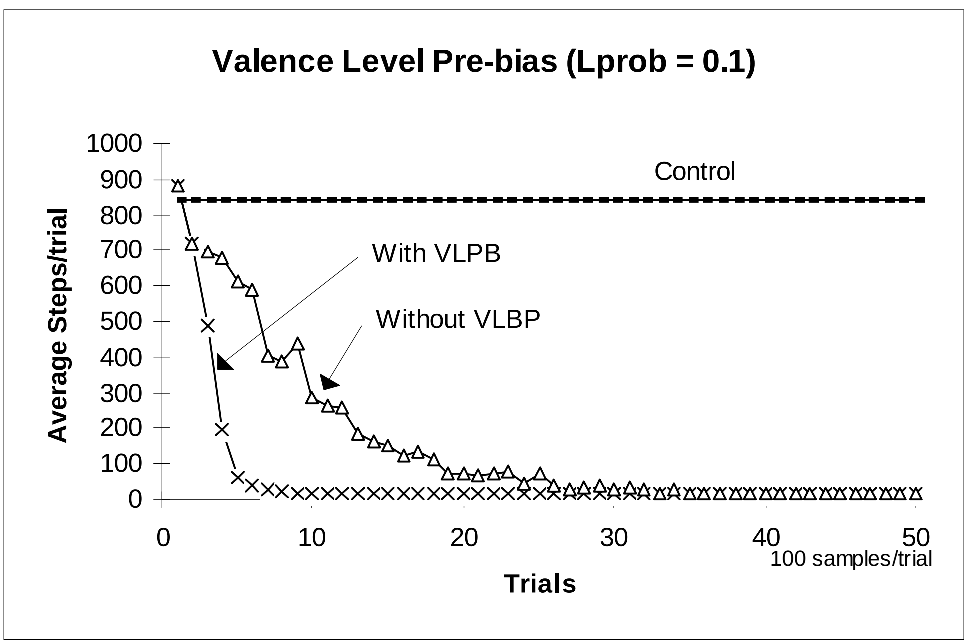
Each sign in $\mathcal{S}$ may therefore be graded according to the highest valence level it has achieved in the past in various Dynamic Policy Maps created by the system. Therefore, if a sign s has been nominated as a goal in the past, the learning sub-system should always create a new μ-hypothesis if the opportunity arises. If the sign s has been implicated at valence level two, then the learning system should be strongly biased to create a new μ-hypothesis, and so on, reducing as the highest recorded valence level for s falls away. In a practical system the probability of learning would reasonably be a function of (1) the highest ("best") valence level achieved by the sign; (2) the priority of the goal giving rise to the valencing; and (3) how recently the goal was valenced. Thus:

P(creation) ← *f*(`best_valence_level*goal_priority*recency_of_goal`)

eqn. (7-1)

Giving a situation where higher valence levels and greater goal priorities increase the probability that the unexpected occurrence of ȣ will give rise to the formulation of a new μ-hypothesis to predict that sign. The probability further decreasing as time elapses since the goal was last asserted.

The current implementation of the SRS/E algorithm records the most significant valence level assigned to every element of the Sign List in the value `best_valence_level`. In an optional process to be referred to as *valence level pre-bias* μ-hypothesis creation by unexpected event (SRS/E step 8.2) always creates a new expectancy if the unpredicted sign has any valence level defined for it. This has no effect when the *learning probability rate* (Lprob) is 1.0, as all opportunities to learn are exploited unconditionally. The results of the experiments described in section 6.2 show the deterioration in learning performance as Lprob is reduced. Figure 7-3 compares the effect of enabling the valence level pre-bias option for the data in figure 6-2 (where Lprob = 0.1, Adisp = 1.0 and Arep = 0.0) against the original results.



monolith\results\vlpb\vlpb.xls

Figure 7-3: The Effect of Valence Level Pre-Bias

The dramatic improvement in learning performance is explained by the rate at which the valence level may propagate from the goal sign. With Lprob = 0.1, there is effectively only 10% chance that the crucial μ-hypothesis that connects the goal to a sign at valence level one will be created. Without this critical link, no DPM can be built, and goal seeking performance is restricted to random walk search. Once this link is created the catchment area within the DPM is widened and the corresponding random search time reduced.

The step-like performance shifts for many individual trials (which appear as the classical negatively decelerating learning curves when averaged over many trials) are a consequence of the abrupt connection of the growing network of latently learned expectancies, with those connected to the goal. By ensuring that the final connection is made (by pre-biasing it), and that the second connection is made on the next attempt, and so on, the portion of the graph connected to the goal is guaranteed to expand by at least one valence level on each trial. In figure 7-3 this would be a maximum of 14 trials. In practice this is reduced to around half this figure due to latent learning of the graph made during the trial-and-error search period of each trial.

7.5. Aversion

The discussion of SRS/E up to this point has only considered goals that are actively sought, and has not included situations where an action is to be avoided as it may lead to an undesirable outcome. There is a considerable body of evidence (Campbell and Masterson, 1969; Schwartz, 1989, Ch. 6) that animals and humans will actively avoid situations leading to certain sensations, variously described as undesirable, unpleasant or *painful*. The mechanism by which sensations are characterised in these ways in nature is not entirely clear.

For the purposes of the SRS/E algorithm it is sufficient to designate certain sensations, as encoded as input tokens or signs, as undesirable. This is a function of the ethogram design. μ-Hypotheses that predict the occurrence of these outcomes may be disadvantaged by additional cost estimates. The degree of this disadvantage being related to the given degree of undesirability of the resulting sensation, and the confidence with which the outcome is predicted. It may be inappropriate to chain these aversions, in the manner of the positive goal seeking activities, as this may lead to a form (or analogue) of a *phobia*. Actions are avoided on the basis they might lead to an undesirable outcome at some time in the future, irrationally, as many actions may be taken to easily avoid the undesirable outcome. Clinical symptoms of phobias in humans seem unlikely to be related to this mechanism.

Chapter 8

8. Discussion and Conclusions

8.1. Reactive or Cognitive?

The initial problems remain. Is behaviour in animals and animats primarily or wholly according to responses mediated by the immediate reaction to impinging stimuli? Is learning simply a matter of strengthening or weakening the connections between stimulus and response, as the reactive or situated agent behaviourists would have us believe? Or is behaviour primarily instigated by "goals", internal states of the animat set and satisfied according to the physiological needs of the animat, with the processes of the animat selecting actions to pursue those goals?

These questions have been hotly debated for nearly a century, with a mountain of evidence accumulated for both viewpoints. Brooks (1991b) has argued (and many before him), much of what we observe in animal and human behaviour can be perfectly adequately explained with a purely stimulus-response analysis. Yet from the time of Tolman (1932) psychologists have argued that reactive behaviourism is wholly inadequate to explain the behavioural abilities of the human species and, as demonstrated through ingenious experiment, to explain all the behavioural abilities of animals.

8.2. Expectancy Model as "Missing Link" in Learning Theory

The Dynamic Expectancy Model may be thought of as the "missing link" between pure S-R behaviourism and the "cognitive", goal based, approach. While the Dynamic Policy Map is created by a goal driven process, utilising the three part representation of the μ-hypothesis, a purely cognitive notion, immediate <u>behaviour</u> is selected only on the basis of the current stimulus set, and so may be thought of as purely reactive. In many experimental designs the two may appear almost indistinguishable from one another. A similar distinction has been developed in the idea of universal planning, which is considered in more detail later in this chapter. Critically, and in keeping with the observation that reward is most effective if applied immediately following an event, reinforcement is still applied directly to the main unit of learning, the μ-hypothesis, immediately the outcome (of the prediction) is known. The adaptive component of the learning process is pure reinforcement; behaviour due to the combination of these units to produce goal seeking behaviour by the *spreading activation* process. Direct reinforcement relative to a known system "motivation" is not excluded, as demonstrated by the *valence level pre-bias* experiments. There is also no restriction to the re-ordering or strengthening of elements of the Behaviour List $\mathcal{B}$ in a manner entirely consistent with a pure S-R behaviourist reinforcement regime.

Given the obvious diversity of both physical and behavioural characteristics across all the species of the animal kingdom, it would appear idle to suggest that there would not be a similar diversity of behavioural and learning strategies. Some animals with simple behavioural strategies may employ no adaptive ability, or limited learning strategies. In others the number and complexity of these strategies increase, manifest as improved behavioural ability. Razran (1971, p. 252) has proposed an "evolutionary ladder of reactions", which argues for a correlation between an animal's place on the evolutionary scale with the appearance of experimental evidence for various learning strategies at the different levels. In this context adoption of different and varied reinforcement strategies, and similar strategies to varying extents, by different species seems inevitable.

8.2.1. Types of Reinforcer

The conventional view of a *reinforcer* is related to underlying biological needs, such as "*food, water or sexual contact for appropriately deprived individuals*" (Bower and Hilgard, 1981, p. 268). It is exactly these needs that can repeatedly be demonstrated as the motivations or drives to initiate and sustain behaviour. It makes design sense to learn behaviours relating directly to those aspects that will be most germane to the everyday existence of the animat. Such *primary reinforcers* may be easily identified and categorised into phenomena that do, and those which do not, act to modify behaviour. In SRS/E, with the valence level pre-bias (VLPB) option enabled, any sign placed on the Goal List will subsequently adopt the role of a primary reinforcer.

It is clear that phenomena other than direct biological need can act as a learning reinforcer. Such *secondary reinforcers* may include "*money, praise, social approval, attention, dominance and the spoken exclamation "good"*" (Bower and Hilgard, 1981, p. 268). At a level below even the primary reinforcers, notions of "pleasure" and "pain" appear to "pre-classify" stimuli and sensations into desirable phenomena, to be sought and undesirable phenomena, to be avoided. The existence of specific nerve types to detect "painful" stimuli would indicate that this is a very primitive mechanism, one it is easy to argue will have a very immediate impact on the survival rate of an organism. "Pleasure", on the other hand, seems to be associated with a much higher level of neural organisation. In this context the application of expectation satisfaction appears as a bridging reinforcer. Expectation satisfaction is neither a primary reinforcer - it serves no direct biological need, nor a secondary reinforcer - as it does not require a social infrastructure implicit in the list of secondary reinforcers.

8.3. Relationship to Policy Maps and Universal Plans

A feature of the *Dynamic Policy Map* is that it indicates the most appropriate action to take in the specific set of circumstances defined by the goal being sought and by the prevailing sensory pattern. In SRS/E this pattern may include elements from the trace of past sensations. In this respect the action selection mechanism has many similarities to the *policy map* described for reinforcement and *Q*-learning procedures. These procedures suffer in comparison to the DPM when the goal definition changes, or the path to the goal becomes blocked or radically altered. Schoppers (1987, 1989, 1995) develops the notion of *universal planning* that addresses the plan/react issue from a different direction.

In Schoppers' system a conventional planner builds a problem-solution path using goal reduction operators. The resulting structure is converted into a decision tree. This may be traversed for each current situation to determine the action appropriate to the prevailing conditions defined by a set of known and predetermined predicate tests, a *cache* of pre-formulated step solutions. The reactive nature of the universal plan overcomes a form of brittleness inherent in conventional planning, where failure of any stage during execution causes failure of the plan as a whole. Universal plans react to successes and failures in activity without recourse to additional computationally expensive replanning.

Ginsberg (1989) argues against the universal plan as a useful approach. He argues that the size of the cache will grow exponentially with the number of sensors, that there will be only a minor computational cost saving, and that this will be at the expense of greater storage requirements. Ginsberg's exponential growth argument is based on the notion that all sensors are independent, and that each sensor may be connected to every action. He further argues that, unless the "universal plan" covers all eventualities it should properly be referred to as an *approximate universal plan*.

Strict application of the exponential complexity argument is specious. The world is clearly non-uniform. Were the world "uniform" then it would make no difference which action was taken under what circumstances, and such is palpably not the case. All associationist, behaviourist and cognitive models are based on the exploitation of this non-uniformity. Rivest and Schapire (1990) have presented an algorithm to detect and utilise equivalence in detectable conditions. Using this algorithm the 10^{19} states of the sometime popular children's toy the *Rubik's Cube* may be reduced to 54 conditions. Yet it may be that important conditions in the environment are poorly distinguishable, either because they are in some true sense similar, or because the sensory capabilities used to differentiate between them are ineffective. Under these conditions the behavioural (and learning) mechanisms will be obliged to incorporate a broader spectrum of sensations to disambiguate between candidate options.

If we view the evolution of species as nature's "universal plan generator" (as made manifest in an individual's ethogram), it becomes clear that these exponential complexity pre-conditions relating to sensors do not hold. As discussed in an earlier section, nature apparently tailors and tunes otherwise undifferentiated sensory apparatus to each task. Tinbergen's birds responded quite specifically to certain "predator" silhouettes, but were apparently oblivious to other shapes. SRS/E and other like systems may take advantage from similarly tuned sensory apparatus, but even without this advantage will seek to identify those combinations of sensations that are significant, and ignore the remainder. In summary there is no need for sensory apparatus to be uniform or homogeneous.

Classical AI planning systems have two potential advantages over reactive and policy based approaches. First, they are (or should be) incorporated into formally correct search procedures. More significantly this implies that the operators defined must themselves be correct; that is achieve the outcome they promise, under the conditions they promise them. Second, the classical planner may take different actions based solely on its current position in its internal solution path, although the incoming sensor vector is identical. The current detectable conditions are used for confirmation, or not at all. Purely reactive systems based on the current sensor vector do not have this advantage. SRS/E addresses this problem by the use of activation traces and recency values. Other approaches may allow recirculation of sensory data (for instance, Becker's proposal to re-circulate kernels into STM,) or some other method for the explicit recording of past events into the representation.

However, classical AI planning can lead to a form of brittleness. If the operators are not correct the solution path generated will not be correct. Advantage gained from the correctness of the search procedure is compromised. SRS/E operators, the μ-hypotheses, are, by their nature, only an estimate of the described transition. The Dynamic Policy Map allows the SRS/E algorithm to select actions on the basis of combined probabilities, as manifest in the cost estimation procedures, and then to update its confidence in individual μ-hypotheses on the basis of the outcome. It is particularly robust in the face of unexpected outcomes caused, among other reasons, by faulty or unconfirmed μ-hypotheses. It takes advantage of serendipitous transitions forward to the goal where the cost estimate unexpected falls; and may continue along some other route to recover from a failure to traverse the expected path.

In a wide range of circumstances speed of response is the critical issue in behaviour. The tardy prey, absorbed in careful planning of its escape, might expect no quarter from the stooping hawk. Perhaps predictably, Schoppers (1989) in his reply to Ginsberg argues in favour of the increased space utilisation for the cache to achieve responsiveness. Given the incompleteness of most behavioural repertoires, and of the scope of the current generation of formal planners, “universal plan” may indeed be something of a misnomer.

8.4. One-Shot Learning Phenomena

The SRS/E model clearly demonstrates the *one-shot learning* phenomena. As soon as the μ-hypotheses is created the animat has a possible path between the two points in the “cognitive” map represented by the signs “s1” and “s2” embedded in a μ-hypothesis. An effective μ-hypothesis becomes rapidly adopted as the path of choice, and the animat will appear to learn quickly, possibly as a result of a single trial. Because its outcome is successfully predicted, discovery of an effective solution also has the effect of suppressing further learning activity related to the sign “s2”. If, as is more likely, the new μ-hypothesis fails to encapsulate all the conditions necessary for a perfect prediction, further learning may occur at each instance of an imperfect prediction. At some point it may be that there are sufficient imperfect μ-hypotheses to ensure that every instance of “s2” is predicted, and learning for this restricted sub-domain will cease, at least temporarily.

This procedure may serve to explain the conundrum (described by Bower and Hilgard, 1981, p. 341) of why a rapidly learned path is quickly extinguished, yet one that is learned over an extended period takes longer to disappear. Individual μ-hypotheses are (in SRS/E at least) extinguished at an essentially equal rate, on the basis of activations, not elapsed time. Where one-shot learning has taken place, a single μ-hypothesis is available to reach the solution while the goal is asserted. No further μ-hypotheses being created as none are required. The observed extinction time is therefore equivalent to that for a single μ-hypothesis. Where several such alternative, albeit imperfect, μ-hypotheses exist, more than one path will be available through the Dynamic Policy Map. As each path fails, another will be selected from the recomputed DPM. The animat will continually swap between the alternatives as the estimated policy cost shifts (at a rate determined by the parameters previously discussed) due to prediction failures. Eventually one, then another and finally all the different paths are extinguished and the goal is finally abandoned as unachievable in the normal way.

Overall time to extinction, as measured by the count of actions ascribed to pursuing the goal, is then (in the SRS/E algorithm at least) a function of the number of alternative paths through the DPM. Alternative paths arise through imperfect μ-hypothesis formulation, which extends learning time. Therefore, extended learning times lead to extended extinction times. Careful examination of results from extinction experiments (section 6.5) reveal this effect, which is particularly apparent in the *dual-path extinction* procedures (figure 6-17).

Taken to a natural conclusion, SRS/E attempts to build a hypothesis about every sign it might detect, and also to predict every occurrence of those signs. Under certain circumstances these conditions can hold true, for instance those described by some *Markov Decision Processes* (MDP) worlds. In the finite and deterministic (FDMSSE) environment the SRS/E algorithm will stabilise with a μ-hypothesis to predict every sign and for every appearance of each possible sign.

8.5. Expectancy Theory and XBL - a Proposal

The development of expectation based learning directly impacts one of the long standing conundrums associated with machine learning; how to make learning truly autonomous. Autonomous learning means that the animat or learning program can learn without any form of external supervision or guidance as to what represents a "good" or "bad" choice. In the case of the novel Dynamic Expectancy Model described in this thesis, and tested in the form of the SRS/E algorithm and implementation, a reinforcement signal is generated internally from successful and failed predictions.

Generally machine learning algorithms fall into two categories, supervised and unsupervised learning. In the former category a teacher is on hand to indicate to the system the appropriateness of its actions and so provide the feedback to guide the learning mechanism. In the latter case information about the task to be learned has been embedded in the code. Buchanan, Smith and Johnson (1979) refer to this component as the *critic*. The critic compares the outcome of the *performance element*, responsible for the overt (and possibly faulty behaviour) with the predefined desired behaviour and supplies an error or difference signal to a *learning element*, which modifies the performance element accordingly. Their *model of machine learning* is a general one, but the form in which each of the elements appears and the nature of the signals passed between them is particularly diverse.

Expectation Based Learning (XBL)[30], based on the principles laid down for the Dynamic Expectancy Model, at last releases the *etho-engineer*[31] from the obligation, but not the option, to specify goal or purpose related criteria for the learning element. Evaluation of an SRS/E μ-hypothesis on the basis of its predictive ability forms a measure of the effectiveness of that μ-hypothesis. Its usefulness is a separate issue, related to the degree to which it enables the performance element to pursue some pre-defined or otherwise generated purpose. The *valence level pre-bias* (VLBP) experiment demonstrates that when learning and performance are indeed linked, both may be advantaged.

Drescher (1991) suggests the term "Schema Based Learning" be adopted as appropriate to the class of intermediate level cognitive models. Notwithstanding the importance of the tri-partite representation adopted by SRS/E, ALP and JCM, it, however, does not align directly with the notion of expectancy. The satisfaction of an expectancy is not tied to this particular representational formulation. It is possible that the notion of an expectation and its subsequent satisfaction may prove to be applicable to a wide range of other otherwise quite conventional structures already employed in the fields of Artificial Intelligence, Machine Learning and Adaptive Behaviour research.

---

[30] XBL, rather than EBL, as this term is already in widespread use ("*Explanation Based Learning*", Minton *et al*, 1990)

[31] One who engineers ethograms - for want of a more apposite term

9. Appendix One

This appendix collates all the stages in a single execution cycle of the SRS/E algorithm, as previously described in Chapter 4.

1.0 Gather Tokens and Update Sign-list

Initialise $\mathcal{S}^{new} \leftarrow \{\}$; $\mathcal{I}^{*} \leftarrow \{\}$; $\mathcal{S}^{*} \leftarrow \{\}$;

1.1 Accept tokens into buffer, for each `token_string` do

1.1.1 $\iota \leftarrow \mathcal{I}($`token_string`$)$ [convert input string]

[note: $\mathcal{X}(y)$ convert element of type y to element of type $\mathcal{X}$]

1.1.2 if $\iota \notin \mathcal{I}$ [a token previously unknown to the system]

1.1.2.1 $\mathcal{I} \leftarrow \mathcal{I} + \iota$ [append ι to $\mathcal{I}$]

1.1.2.2 $\mathcal{S}^{new} \leftarrow \mathcal{S}^{new} + \mathcal{S}(\iota)$ [create a sign containing ι]

1.1.3 $\mathcal{I}^{*} \leftarrow \mathcal{I}^{*} + \iota$

1.2 $\mathcal{S} \leftarrow \mathcal{S} + \mathcal{S}^{new}$

1.3 For each s where $s \in \mathcal{S}$

1.3.1 if (`EvalSignConjunction`(s))

$\mathcal{S}^{*} \leftarrow \mathcal{S}^{*} + s$ [eqn. 4-3]

1.4 $\mathcal{G} \leftarrow \mathcal{G} - (\mathcal{S}^{*} \cap \mathcal{G})$ [cancel satisfied goals]

2.0 Evaluate μ-Experiments on Basis of Prior Prediction

Initialise $\mathcal{S}^{pred} \leftarrow \{\}$;

2.1 for every p ($p \in \mathcal{P}$), such that `predicted_time`(p) = `now`, do

2.1.1 if `predicted_sign`(p) $\in \mathcal{S}^{*}$ [prediction succeeds]

2.1.1.1 Update `predicting_hypo`(p) [according to α, eqn. 4-11]

2.1.1.2 $\mathcal{S}^{pred} \leftarrow \mathcal{S}^{pred}$ + `predicted_sign`(p)

2.1.2 if `predicted_sign`(p) $\notin \mathcal{S}^{*}$ [prediction fails]

2.1.2.1 Update `predicting_hypo`(p) [according to β , eqn. 4-12]

2.1.2.2 `rebuildpolicynet` $\leftarrow$ `rebuildpolicynet` + δ

2.1.3 $\mathcal{P} \leftarrow \mathcal{P} - p$ [remove spent prediction]

2.2 $\mathcal{S}^{unexpected} \leftarrow \mathcal{S}^{*} - \mathcal{S}^{pred}$ [record unpredicted signs]

3.0 Select Innate Action and Set Goals

Initialise $\mathcal{B}^* \leftarrow \{\}$;

3.1 `candidate_action` ← `SelectRandomAction`($\mathcal{R}$)

3.2 for each $\mathcal{b}$ where `action`($\mathcal{b}$) $\in \mathcal{B}^r$ **AND** `condition`($\mathcal{b}$) $\in \mathcal{S}^*$

    3.2.1 $\mathcal{B}^{r*} \leftarrow \mathcal{B}^{r*} + \mathcal{b}$;

3.3 `innate_action` ← `action`(max(`behaviour_priority`($\mathcal{B}^{r*}$))) [innate action]

3.4 `innate_priority` ← max(`behaviour_priority`($\mathcal{B}^{r*}$))

3.5 for each $\mathcal{b}$ where `action`($\mathcal{b}$) $\in \mathcal{B}^g$ **AND** `condition`($\mathcal{b}$) $\in \mathcal{S}^*$

    3.5.1 $\mathcal{G} \leftarrow \mathcal{G} + \mathcal{b}$ [build Goal List]

3.6 $\mathcal{G} \leftarrow$ order(`goal_priority`($\mathcal{G}$)) [order Goal List by priorities]

3.7 if(`innate_priority` > ε) [above basal threshold?]

    3.7.1 `candidate_action` ← `innate_action`

3.8 if(`goal_priority`(g^1) < `innate_priority`) [select goal or innate]

    3.8.1 skip to step 6.0

4.0 Build (re-build) Dynamic Policy Map (`Hypo::BuildPolicyNet()`)

Initialise $\mathcal{H}^£ \leftarrow \{\}$; $\mathcal{S}^v \leftarrow \{\}$; $\mathcal{S}^£ \leftarrow \{\}$;

`rebuildpolicynet` ← 0; `pathavailable` ← **FALSE**;

`bestcost` ← **MAXVALUE**; `vn` ← 1 [valence level one]

Rebuild map if goal changed or 'rebuild' greater than threshold

4.1 while ($g^1 \in \mathcal{S}^*$) [top-goal already satisfied]

    4.1.1 $\mathcal{G} \leftarrow \mathcal{G} - g^1$ [so remove]

    4.1.2 $g^1 \leftarrow$ max(`goal_priority`($\mathcal{G}$)) [and select next highest]

4.2 if($\mathcal{G}$ = {}) skip to step 6.0 [no goals on Goal List]

4.3 (if $g^1 = g^{1@t-1}$ **AND** `rebuildpolicynet` < `REBUILDPOLICYTRIP`)

    skip to step 5.0 [no need to rebuild DPM]

Stage 1 - create first valence level

4.4 for each $\mathcal{h}$ such that `s2`($\mathcal{h}$) = g^1

    4.4.1 $\mathcal{h}^£ \leftarrow$ `GetCostEstimate`($\mathcal{h}$) [eqn. 4-13]

    4.4.2. $\mathcal{S}^{v+1} \leftarrow \mathcal{S}^{v+1}$ + `s1`($\mathcal{h}$) [record valenced sub-goals]

    4.4.3 $\mathcal{H}^£ \leftarrow \mathcal{H}^£ + \mathcal{h}^£$ [cost of transition s1 to goal]

    4.4.4 $\mathcal{S}^£ \leftarrow$ s1($\mathcal{h}^£$) [record sign cost]

    4.4.5 if(`s1`($\mathcal{h}$) $\in \mathcal{S}^*$)

        `pathavailable` ← **TRUE** [path solution found]

    4.4.6 if(`bestcost` > $\mathcal{h}^£$) `bestcost` ← $\mathcal{h}^£$

Stage 2 - continue spreading activation until done

4.5 `vn` ← `vn` + 1

4.6 if($\mathcal{S}^{v}$ = {}) skip to step 5.0 [expansion complete]

4.7 for each h such that `s2`(h) ∈ $\mathcal{S}^{v=vn}$ [expand each sub-goal]

 4.7.1 $h^{£}$ ← s2($\mathcal{S}^{£}$) + `GetCostEstimate`(h) [eqn. 4-13]

 4.7.2 $\mathcal{H}^{£}$ ← $\mathcal{H}^{£}$ + $h^{£}$ [record total cost of path]

 4.7.3 if(`s1`(h) ∉ $\mathcal{S}^{v}$ **OR** s1($h^{£}$) > s1($\mathcal{S}^{£}$)) [new or better path]

  4.7.3.1 $\mathcal{S}^{v+1}$ ← $\mathcal{S}^{v+1}$ + `s1`(h) [new sub-goals]

  4.7.3.2 $\mathcal{S}^{£}$ ← $\mathcal{S}^{£}$ + s1($h^{£}$) [record lower sign cost]

 4.7.4 if(`s1`(h) ∈ $\mathcal{S}^{*}$)

  `pathavailable` ← **TRUE** [solution path found]

 4.7.5 if(`bestcost` > $h^{£}$) `bestcost` ← $h^{£}$

4.8 return to step 4.5 [expand next valence level]

5.0 Select Valenced Action (`Hypo::SelectValencedAction()`)

5.1 `VBP` ← `GetValenceBreakPoint`() [establish `VBP`]

5.2 if (`pathavailable` = **FALSE**) `VBP` ← 0 [no path to goal]

5.3 else if (`VBP` ≤ 0 **OR** `VBP` > `bestcost`) [compute `VBP`]

 `VBP` ← `bestcost` * `VALENCEBREAKPOINTFACTOR`

5.4 $\mathcal{H}^{\#£}$ ← $\mathcal{H}^{£}$ ∩ (`s1`(h) ∈ $\mathcal{S}^{*}$) [candidate active signs]

5.5 h ← min($\mathcal{H}^{\#£}$) [select least policy cost]

5.6 `valenced_action` ← `r1`(h)

5.7 if(`policy_value`(h) ≤ `VBP`) [break-point reached?]

 `candidate_action` ← `valenced_action` [no, use valenced action]

5.8 if(`policy_value`(h) > Ω) [goal cancellation level?]

 5.8.1 $\mathcal{G}$ ← $\mathcal{G}$ - g^{1} [so cancel top-goal]

6.0 Perform Action

6.1 `DoAction(candidate_action)` [reify candidate action]

6.2 $\mathcal{R}^{*}$ ← `candidate_action` [record in trace]

7.0 Conduct µ-Experiments (`Hypo::EvaluateHypotheses()`)

initialise $\mathcal{H}^{*}$ ← {};

7.1 for all h, such that `s1`(h) ∈ $\mathcal{S}^{*}$ **AND** `r1`(h) ∈ $\mathcal{R}^{*}$

 7.1.1 $\mathcal{H}^{*}$ ← $\mathcal{H}^{*}$ + h [record activation]

 7.1.2 $\mathcal{P}$ ← $\mathcal{P}$ + $\mathcal{P}$(h, `s2`(h), `now` + t) [make prediction]

8.0 Hypothesis Management (`Hypo::NewHypo()`)

Creation on the basis of novelty

8.1 for each $\mathcal{s}^{new}$ such that ($\mathcal{S}^{new} \neq \{\}$ **AND** $\mathcal{s}^{new} \in \mathcal{S}^{new}$)

8.1.1 if (rand(0.0 .. 1.0) > λ) skip to step 8.1.7

8.1.2 `s1` $\leftarrow$ `Select(`$\mathcal{s}^{x} \in \mathcal{S}^{*@-t}$)

8.1.3 `r1` $\leftarrow$ `Select(`$\mathcal{r}^{x} \in \mathcal{R}^{*@-t}$)

8.1.4 `s2` $\leftarrow$ $\mathcal{s}^{new}$

8.1.5 $\mathcal{H} \leftarrow \mathcal{H} + \mathcal{H}$(`s1`,`r1`,`s2`$^{@+t}$), where `s1` $\neq$ `s2`

8.1.6 `rebuildpolicynet` $\leftarrow$ `rebuildpolicynet` + Δ

8.1.7 $\mathcal{S}^{new} \leftarrow \mathcal{S}^{new}$ - $\mathcal{s}^{new}$

Creation on the basis of unpredicted event

8.2 for each $\mathcal{s}^{unexpected}$ such that ($\mathcal{S}^{unexpected} \neq \{\}$ **AND** $\mathcal{s}^{unexpected} \in \mathcal{S}^{unexpected}$)

8.2.1 if (rand(0.0 .. 1.0) > λ) skip to step 8.2.7

8.2.2 `s1` $\leftarrow$ `Select(`$\mathcal{s}^{x} \in \mathcal{S}^{*@-t}$)

8.2.3 `r1` $\leftarrow$ `Select(`$\mathcal{r}^{x} \in \mathcal{R}^{*@-t}$)

8.2.4 `s2` $\leftarrow$ $\mathcal{s}^{unexpected}$

8.2.5 $\mathcal{H} \leftarrow \mathcal{H} + \mathcal{H}$(`s1`,`r1`,`s2`$^{@+t}$), where `s1` $\neq$ `s2`

8.2.6 `rebuildpolicynet` $\leftarrow$ `rebuildpolicynet` + Δ

8.2.7 $\mathcal{S}^{unexpected} \leftarrow \mathcal{S}^{unexpected}$ - $\mathcal{s}^{unexpected}$

Specialisation (differentiation)

8.3 for all $\mathcal{h}$, such that $\mathcal{h} \in \mathcal{H}^{*}$ **AND** `hypo_maturity(`$\mathcal{h}$) > Ψ
AND `hypo_prob(`$\mathcal{h}$) > θ **AND** `hypo_prob(`$\mathcal{h}$) < Θ

8.3.1 `s1` $\leftarrow$ $\mathcal{S}$(`s1(`$\mathcal{h}$) + $\mathcal{x}^{@-t}$) [differentiate s1]

8.3.2 `r1` $\leftarrow$ `r1(`$\mathcal{h}$) [copy action]

8.3.3 `s2` $\leftarrow$ `s2(`$\mathcal{h}$) [copy s2]

8.3.4 $\mathcal{H} \leftarrow \mathcal{H} + \mathcal{H}$(`s1`,`r1`,`s2`$^{@+t}$) [install new μ-hypothesis]

8.3.5 $\mathcal{S} \leftarrow \mathcal{S}$ + `s1` [install new sign in $\mathcal{S}$]

8.3.6 `rebuildpolicynet` $\leftarrow$ `rebuildpolicynet` + Δ

Deletion (forgetting) under competition

initialise $\mathcal{H}^{\#} \leftarrow \{\}$;

8.4 for all $\mathcal{h}$, such that $\mathcal{h} \in \mathcal{H}^{*}$ **AND** `hypo_maturity(`$\mathcal{h}$) > Ψ

AND `hypo_prob`($\mathcal{h}$) < Θ

8.4.1 $\mathcal{H}^{\#} \leftarrow \mathcal{H}^{\#} + \mathcal{h}$ [build candidate list]

8.5 $\mathcal{h}^{\text{delete}} \leftarrow$ min(`hypo_prob`($\mathcal{H}^{\#}$)) [select a deletion candidate]

8.6 $\mathcal{H} \leftarrow \mathcal{H} - \mathcal{h}^{\text{delete}}$ [update Hypothesis List]

8.7 `rebuildpolicynet` ← `rebuildpolicynet` + Δ

9.0 Return to step 1

10. References

Agre, P.E. (1995) “Computational Research on Interaction and Agency”, Artificial Intelligence, Vol. 72, pp.1-52

Agre, P.E. and Chapman, D. (1987) “Pengi: An Implementation of a Theory of Activity”, Proceedings of the Sixth National Conference on Artificial Intelligence. Los Altos, CA: Morgan Kaufmann, pp. 268-272.

Albus, J.S. (1981) “Brains, Behavior, and Robotics”, Peterborough, NH: Byte Books/McGraw-Hill

Altman, J. and Sudarshan, K. (1975) “Postnatal Development of Locomotion in the Laboratory Rat”, Animal Behaviour, Vol. 23, pp. 896-920

Arbib, M.A. and Cobas, A. (1991) “Schemas for Prey-Catching in Frog and Toad”, in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior “From Animals to Animats”, pp. 142-150

Arbib, M.A. and Lee, H.B. (1993) “Anuran Visuomotor Coordination for Detour Behavior: From Retina to Motor Schemas”, in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 42-51

Aylett, R. (Ed.) (1994) “Models or Behaviours, Which Way Forward for Robotics?”, AISB94 Workshop Series, University of Leeds, 12th and 13th April 1994

Baerends, G.P. (1976) “The Functional Organization of Behaviour”, Animal Behaviour, Vol. 24, pp.726-738

Ball, N. (1994) “Organizing an Animat’s Behavioural Repertoires Using Kohonen Feature Maps”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 128-137

Barto, A.G. and Sutton, R.S. (1982) “Simulation of Anticipatory Responses in Classical Conditioning by a Neuron-like Adaptive Element”, Behavioural Brain Research, Vol. 4, pp. 221-235

Baird, L.C. and Klopf, A.H. (1993) “Extensions to the Associative Control Process (ACP) Network: Hierarchies and Provable Optimality”, in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 163-171

Becker, J.D. (1970) “An Information-processing Model of Intermediate-level Cognition”, Stanford Artificial Intelligence Project, Memo AI-119, Computer Science Dept., Stanford University (Ph.D. thesis)

Becker, J.D. (1973) “A Model for the Encoding of Experiential Information”, in: Schank, R.C. and Colby, K.M. (Eds.) “Computer Models of Thought and Language”, San Francisco: W.H. Freeman and Company, pp. 396-434

Beer, R.D. and Chiel, H.J. (1991) “The Neural Basis of Behavioral Choice in an Artificial Insect”, in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior “From Animals to Animats”, pp. 247-254

Berkson, W. and Wettersten, J. (1984) “Learning from Error: Karl Popper’s Psychology of Learning”, LaSalle, IL: Open Court

Blackman, D. (1974) “Operant Conditioning: An Experimental Analysis of Behaviour”, London: Methuen & Co.

Boden, M.A. (1994) “Autonomy and Artificiality”, AISB Quarterly, No. 87 (Spring 1994), pp. 22-28

Bolles, R.C. (1979) “Learning Theory”, New York: Holt-Rinehart-Winston

Bonarini, A. (1994) “Some Methodological Issues about Designing Autonomous Agents which Learn Their Behaviors: The ELF Experience”, in: Trappl, R (Ed.) “Cybernetics and Systems Research” (Proc. 12th Euro. Meeting of Cybernetics and Systems Research), Singapore: World Scientific Publishing, pp. 359-366

Bond, A.H. and Mott, D.H. (1981) “Learning of Sensory-motor Schemas in a Mobile Robot”, in: Proceedings of the Joint Conference on Artificial Intelligence, (IJCAI-81), pp. 159-161

Booker, L.B., Goldberg, D.E. and Holland, J.H. (1990) “Classifier Systems and Genetic Algorithms”, in: Carbonell, J. G. (Ed.) “Machine Learning: Paradigms and Methods”, Cambridge, MA: The MIT Press (a Bradford Book), pp. 235-282

Booker, L.B. (1991) “Instinct as an Inductive Bias for Learning Behavioral Sequences”, in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior “From Animals to Animats”, pp. 230-237

Bower, G.H. and Hilgard, E.R. (1981) “Theories of Learning” Englewood Cliffs: Prentice Hall Inc. (Fifth edition)

Brooks, R.A. (1986) “A Robust Layered Control System For A Mobile Robot”, IEEE Journal of Robotics and Automation, Vol. RA-2, No. 1, March 1986, pp. 14-23

Brooks, R.A. (1990) “The Behavior Language; User’s Guide”, MIT Artificial Intelligence Laboratory, A.I. Memo 1227, April 1990

Brooks, R.A. (1991a) “Intelligence Without Reason”, MIT AI Laboratory, A.I. Memo No. 1293, April 1991. (Prepared for Computers and Thought, IJCAI-91, pre-print)

Brooks, R.A. (1991b) “Intelligence Without Representation”, Artificial Intelligence, Vol. 47, pp. 139-159

Brooks, R.A. and Maes, P. (Eds.) (1994) “Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems”, Cambridge, MA: The MIT Press

Buchanan, B.G., Mitchell, T.M., Smith, R.G. and Johnson, C.R., Jr. (1979) “Models of Learning Systems” Stanford Heuristic Programming Project Memo HPP-77-39 (Computer Science Dept. Report No. STAN-CS-79-692), January 1979

Campbell, B.A. and Masterson, F.A. (1969) “Psychophysics of Punishment”, in: Campbell, B.A. and Church, R.M. (Eds.) “Punishment and Aversive Behavior”, New York: Appleton-Century-Crofts, pp. 1-42

Carbonell, J. G. (Ed.) (1990) “Machine Learning: Paradigms and Methods”, Cambridge, MA: The MIT Press (a Bradford Book)

Catania, A.C. and Harnad, S. (Eds.) (1988) “The Selection of Behavior, The Operant Behaviorism of B.F. Skinner: Comments and Consequences”, Cambridge: The Cambridge University Press

Catania, A.C. (1988) “The Operant Behaviorism of B.F. Skinner”, in: Catania, A.C. and Harnad, S. (Eds.) “The Selection of Behavior”, Cambridge: The Cambridge University Press, pp. 3-8.

Chapman, D. (1989) “Penguins Can Make Cakes”, AI Magazine, Vol. 10, No. 4 (Winter 1989) pp. 45-50

Chesters, W. and Hayes, G. (1994) "Connectionist Environment Modelling in a Real Robot", in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior "From Animals to Animats 3", pp. 189-197

Chrisman, L. (1992) "Reinforcement Learning with Perceptual Aliasing: The Perceptual Distinctions Approach", in: Proceedings of the American Association for Artificial Intelligence (AAAI-92), San Jose, CA, pp. 183-188

Cliff, D. (1991) "Computational Neuroethology: A Provisional Manifesto", in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior "From Animals to Animats", pp. 29-39

Cliff, D. (1994) "AI and A-Life: Never Mind the Blocksworld", AISB Quarterly, No. 87 (Spring 1994), pp. 16-21

Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) (1994) Proceedings of the Third International Conference on Simulation of Adaptive Behavior "From Animals to Animats 3", Cambridge, MA: The MIT Press

Dawkins, R. (1986) "The Blind Watchmaker", Harmondsworth: Penguin Books Ltd.

Dewsbury, D.A. (1978) "Comparative Animal Behavior", New York: McGraw-Hill Book Company

Dolhinow, P.J. and Bishop, N. (1972) "The Development of Motor Skills and Social Relationships Among Primates Through Play", in Dolhinow, P.J. (Ed.) "Primate Patterns", New York: Holt Rinehart Winston, pp. 312-337

Dorigo, M. (1995) "ALECSYS and the AutonoMouse: Learning to Control a Real Robot by Distributed Classifier Systems", Machine Learning, Vol. 19, pp. 209-240

Dorigo, M. and Bersini, H. (1994) “A Comparison of *Q*-learning and Classifier Systems”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 248-255

Dorigo, M. and Colombetti, M. (1994) “Robot Shaping: Developing Autonomous Agents Through Learning”, Artificial Intelligence, Vol. 71, pp. 321-370

Drescher, G.L. (1987) “A Mechanism for Early Piagetian Learning”, in: Proceedings of the American Association for Artificial Intelligence (AAAI-87), pp. 290-294

Drescher, G.L. (1991) “Made-up Minds: A Constructivist Approach to Artificial Intelligence”, Cambridge, MA: The MIT Press

Fikes, R.E. and Nilsson, N.J. (1971) “STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving”, Artificial Intelligence, Vol. 2, pp. 189-208

Foner, L.N. and Maes, P. (1994) “Paying Attention to What's Important: Using Focus of Attention to Improve Unsupervised Learning”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 256-265

Friedman, L. (1967) “Instinctive Behavior and its Computer Synthesis”, Behavioral Science, Vol. 12, pp. 85-108

Gaussier, P. and Zrehen, S. (1994) “A Topological Neural Map for On-Line Learning: Emergence of Obstacle Avoidance in a Mobile Robot”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 282-290

Ginsberg, M.L. (1989) “Universal Planning: An (Almost) Universally Bad Idea” AI Magazine, Vol. 4, No. 10 (Winter 1989), pp. 40-44

Giszter, S. (1994) “Reinforcement Tuning of Action Synthesis and Selection in a ‘Virtual Frog’”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 291-300

Goodwin, R. and Simmons, R. (1992) “Rational Handling of Multiple Goals for Mobile Robots”, in: “Proceedings of the First International Conference on Artificial Intelligence Planning Systems”, June 1992, College Park, MD. (pre-print)

Haigh, K.Z. and Veloso, M.M. (1996) “Planning with Dynamic Goals for Robot Execution”, in: “Proceedings of the AAAI Fall Symposium "Plan Execution: Problems and Issues"”, AAAI Press, November 1996 (pre-print, to appear)

Hall, J.F. (1966) “The Psychology of Learning”, Philadelphia and New York: J.B. Lippincott Company

Hallam, B.E., Hallam, J.C.T. and Halperin, J.R.P. (1994) “An Ethological Model for Implementation in Mobile Robots”, Adaptive Behavior, Vol. 3, No. 1, pp. 51-79

Harrison, P. (1983) “Operational Research: Quantitative Decision Analysis”, London: Mitchell Beazley Publishers

Hartley, R. (1993) “Propulsion and Guidance in a Simulation of the Worm *C. Elegans*”, in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 122-128

Heitkötter, J. and Beasley, D. (Eds.) (1995) “The Hitch-Hiker’s Guide to Evolutionary Computation: A List of Frequently Asked Questions”, Available via anonymous FTP from “rtfm.mit.edu:/pub/usenet/news.answers/ai-faq/genetic/”, about 90 pages

Hergenhahn, B.R. and Olson, M.H. (1993) “An Introduction to Theories of Learning”, Englewood Cliffs, N.J.: Prentice-Hall (fourth edition)

Hess, E.H. (1959) “Imprinting: An Effect of Early Experience”, Science, Vol. 130, pp. 133-141

Highfield, R. (1996) “Working Out How Time Flies”, Science Feature in the Daily Telegraph, Wednesday February 21, 1996, p. 14

Hinde, R.A. (1970) “Animal Behaviour: A Synthesis of Ethology and Comparative Psychology”, New York: McGraw-Hill (second edition)

Hinton, G.E. (1986) “Learning Distributed Representations of Concepts”, in: “Proceedings of the Eighth Annual Conference of the Cognitive Science Society”, Amsherst, MA

Hinton, G.E. (1990) “Connectionist Learning Procedures”, in: Carbonell, J. G. (Ed.) “Machine Learning: Paradigms and Methods”, Cambridge, MA: The MIT Press (a Bradford Book), pp. 185-234

Holland, J.H. (1975) “Adaptation in Natural and Artificial Systems”, Ann Arbor: The University of Michigan Press

Hubel, D.H. and Wiesel, T.N. (1962) “Receptive Fields, Binocular Interaction, and Functional Architecture in the Cat’s Visual Cortex”, Journal of Physiology, Vol. 160, pp. 106-154

Humphrys, M. (1995) “W-learning: Competition Among Selfish *Q*-learners” University of Cambridge Computer Laboratory Technical Report No. 362, April 1995.

Jahoda, G. (1969) “The Psychology of Superstition”, London: Allen Lane, The Penguin Press

Jochem, T.M., Pomerleau, D.A. and Thorpe, C.E. (1993) “MANIAC: A Next Generation Neurally Based Autonomous Road Follower”, in: “Proceedings of the International Conference on Intelligent Autonomous Systems: IAS-3”, Pittsburgh, Pennsylvania, (pre-print)

Jones, T.L. (1971) “A Computer Model of Simple Forms of Learning”, Technical Report AD-720 337 (AI-TM-236). Massachusetts Institute of Technology, Cambridge, Massachusetts

Kaelbling, L.P. (1994) “Associative Reinforcement Learning: Functions in *k*-DNF”, Machine Learning, Vol. 15, pp. 279-298

Kaelbling, L.P. (1996) “Introduction” [to special issue on reinforcement learning], Machine Learning, Vol. 22, pp. 7-9

Kamin, L.J. (1969) “Predictability, Surprise, Attention, and Conditioning”, in: Campbell, B.A. and Church, R.M. (Eds.) “Punishment and Aversive Behavior”, New York: Appleton-Century-Crofts, pp. 279-296

Kearsley, G. (1996) “Explorations in Learning and Instruction: The Theory Into Practice Database”. George Washington University Psychology Dept. technical report at “http://gwis2.circ.gwu.edu/~kearsley/”

King, N. (1987) “The First Five Minutes”, London: Simon and Schuster

Kirsh, D. (1991) “Today the Earwig, Tomorrow Man?”, Artificial Intelligence, Vol. 47, pp. 161-184

Klahr, D. (1994) “Children, Adults, and Machines as Discovery Systems”, Machine Learning, Vol. 14, pp. 313-320

Kleitman, N. and Crisler, G. (1927) “A Quantitative Study of a Salivary Conditioned Reflex”, American Journal of Physiology, Vol. 79, pp. 571-614 (cited Bower and Hilgard, 1981, p. 51)

Klopf, A.H. (1988) “A Neuronal Model of Classical Conditioning”, Psychobiology, Vol. 16, No. 2, pp. 85-125

Klopf, A.H., Morgan, J.S. and Weaver, S.E. (1993) “Modeling Nervous System Function with a Hierarchical Network of Control Systems that Learn”, in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 254-261

Knuth, D.E. (1973) “The Art of Computer Programming, Volume 3: Sorting and Searching”, Reading, MA: Addison-Wesley Publishing Company

Koch, S. (1954) “Clark L. Hull”, in: Estes, W.K., Koch, S., MacCorquodale, K., Meehl, P.E., Mueller, C.G., Schoenfeld, W.N. and Verplanck, W.S. (Eds.) “Modern Learning Theory: A Critical Analysis of Five Examples”, New York: Appleton-Century-Crofts, pp. 1-176

Krechevsky, I. (1933) “"Hypotheses" in Rats”, Psychological Review, Vol. 39, pp. 516-532

Langley, P. (1996) “Elements of Machine Learning”, Palo Alto: Morgan Kaufmann Publishers

Langton, C. (Ed.) (1989) “Artificial Life, The Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems”, SFI Studies in Sciences of Complexity, Los Alamos, September 1987, Reading, MA: Addison-Wesley Publishing Co.

Levine, M. (1970) “Human Discrimination Learning: The Subset-sampling Assumption”, Psychological Bulletin, Vol. 74, No. 6, pp 397-404

Levy, S. (1992) “Artificial Life, The Quest for a New Creation”, Harmondsworth: Penguin Books

Liaw, J-S. and Arbib, M.A. (1993) “Neural Mechanisms Underlying Direction-Selective Avoidance Behavior”, Adaptive Behavior, Vol. 1, No. 3 (Winter 1993), pp. 227-261

Lieberman, D.A. (1990) “Learning, Behavior and Cognition”, Belmont, CA: Wadsworth Publishing Company

Lin, L-J. (1991) “Programming Robots Using Reinforcement Learning and Teaching”, in: Proceedings of the American Association for Artificial Intelligence (AAAI-91), pp. 781-786

Lin, L-J. and Mitchell, T.M. (1993) “Reinforcement Learning with Hidden States”, in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 271-280

Littman, M.L. (1994) “Memoryless Policies: Theoretical Limitations and Practical Results”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 238-245

Lofts, B. (1970) “Animal Photoperiodism”, London: Edward Arnold (Publishers) Ltd.

Lorenz, K.Z. (1950) “The Comparative Method in Studying Innate Behaviour Patterns”, in: “Physiological Mechanisms in Animal Behaviour, Symposium of the Society of Experimental Biology”, Vol. 4, London: Academic Press, pp. 221-268

MacCorquodale, K. and Meehl, P.E. (1953) “Preliminary Suggestions as to a Formalization of Expectancy Theory”, Psychological Review, Vol. 60, No. 1, pp. 55-63

MacCorquodale, K. and Meehl, P.E. (1954) “Edward C. Tolman”, in: Estes, W.K., Koch, S., MacCorquodale, K., Meehl, P.E., Mueller, C.G., Schoenfeld, W.N. and

Verplanck, W.S. (Eds.) “Modern Learning Theory: A Critical Analysis of Five Examples”, New York: Appleton-Century-Crofts, pp. 177-266

Maclin, R. and Shavlik, J.W. (1996) “Creating Advice-Taking Reinforcement Learners”, Machine Learning, Vol. 22, pp. 251-281

Mahadevan, S. and Connell, J. (1991) “Scaling Reinforcement Learning to Robotics by Exploiting the Subsumption Architecture”, in: Birnbaum, L.A. and Collins, G.C. (Eds.) “Machine Learning, Proceedings of the Eighth International Workshop (ML91)”, San Mateo, CA: Morgan Kaufmann Publishers, Inc., pp. 328-332

Maes, P. (1989) “The Dynamics of Action Selection”, Proceedings of the 11th International Joint Conference for Artificial Intelligence (IJCAI-89), pp. 991-997

Maes, P. (1991) “A Bottom-up Mechanism for Behavior Selection in an Artificial Creature”, in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior “From Animals to Animats”, pp. 238-246

Maes, P. (1993) “Behavior-based Artificial Intelligence”, in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 2-10

Maes, P. and Brooks, R.A. (1990) “Learning to Coordinate Behaviors”, in: Proceedings of the American Association for Artificial Intelligence (AAAI-90), pp. 796-802

Maes, P., Mataric, M.J., Meyer, J-A., Pollack, J. and Wilson, S.W. (Eds.) “From Animals to Animats 4” Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior, Cambridge, MA: The MIT Press

McCallum, R.A. (1995) “Instance-Based Utile Distinctions for Reinforcement Learning with Hidden State”, in: Proceedings of the 12th International Machine Learning Conference, Lake Tahoe, CA. (pre-print)

McCulloch, W.S. and Pitts, W.H. (1943) “A Logical Calculus of the Ideas Immanent in Nervous Activity”, Bulletin of Mathematical Biophysics, Vol. 5, pp. 115-133

McFarland, D. and Sibly, R.M. (1975) “The Behavioural Final Common Path”, Philosophical Transactions of the Royal Society (Series B), Vol. 270, pp. 265-293

Meyer, J-A. and Wilson, S.W. (Eds.) (1991) Proceedings of the First International Conference on Simulation of Adaptive Behavior “From Animals to Animats”, Cambridge, MA: The MIT Press

Meyer, J-A. and Guillot, A. (1991) “Simulation of Adaptive Behavior in Animats: Review and Prospect”, in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior “From Animals to Animats”, pp. 2-14

Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) (1993) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior, Cambridge, MA: The MIT Press

Michalski, R.S. (1980) “Pattern Recognition as Rule-Guided Inductive Inference” IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No. 4, pp. 349-361

Millán, J. del R. (1994) “Learning Efficient Reactive Behavioral Sequences from Basic Reflexes in a Goal-Directed Autonomous Robot”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 266-274

Millán, J. del R. and Torras, C. (1991) “Learning to Avoid Obstacles through Reinforcement”, in: Birnbaum, L.A. and Collins, G.C. (Eds.) “Machine Learning, Proceedings of the Eighth International Workshop (ML91)”, San Mateo, CA: Morgan Kaufmann Publishers, Inc., pp. 298-302

Minsky, M. (1963) “Steps Toward Artificial Intelligence”, in: Feignbaum, E.A. and Feldman, J. (Eds.) “Computers and Thought”, New York: McGraw-Hill, pp. 406-450

Minsky, M. (1985) “The Society of Mind”, New York: Simon and Schuster (A Touchstone Book)

Minsky, M. and Papert, S. (1969) “Perceptrons: An Introduction to Computational Geometry”, Cambridge, MA: The MIT Press

Minton, S., Carbonell, J.G., Knoblock, C.A., Kuokka, D.R., Etzioni, O. and Gil, Y. (1990) “Explanation-Based Learning: A Problem Solving Perspective”, in: Carbonell, J. G. (Ed.) “Machine Learning: Paradigms and Methods”, Cambridge, MA: The MIT Press (a Bradford Book), pp. 63-118

Moore, A.W. and Atkeson, C.G. (1993) “Prioritized Sweeping: Reinforcement Learning With Less Data and Less Time”, Machine Learning, Vol. 13, pp. 103-130

Mott, D.H. (1981) “Sensori-motor Learning in a Mobile Robot”, AI Laboratory Memo. Department of Computer Science and Statistics, Queen Mary College, University of London, Ph.D. Thesis.

Mowrer, O.H. (1956) “Two-factor Learning Theory Reconsidered, with Special Reference to Secondary Reinforcement and the Concept of Habit”, Psychological Review, Vol. 63, pp. 114-128

Mura, F. and Franceschini, N. (1994) “Visual Control of Altitude and Speed in a Flying Agent”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 91-99

Nehmzow, U. and McGonigle, B. (1994) “Achieving Rapid Adaptations in Robots by Means of External Tuition”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 301-308

Newell, A. and Simon, H.A. (1972) “Human Problem Solving”, Englewood Cliffs, N.J.: Prentice-Hall, Inc.

Nilsson, N.J. (1965) “Learning Machines: Foundations of Trainable Pattern-Classifying Systems”, New York: McGraw-Hill

Nilsson, N.J. (1980) “Principles of Artificial Intelligence”, New York: Springer-Verlag (Symbolic Computation Series)

Norman, D.A. (1969) “Memory and Attention, an Introduction to Human Information Processing”, New York: Wiley & Sons, Inc.

Payton, D.W., Rosenblatt, J.K. and Keirsey, D.M. (1990) “Plan Guided Reaction”, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 20, No. 6, pp. 1370-1382

Peng, J. and Williams, R.J. (1992) “Efficient Learning and Planning Within the Dyna Framework”, in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 281-290

Peng, J. and Williams, R.J. (1996) “Incremental Multi-Step *Q*-learning”, Machine Learning, Vol. 22, pp. 283-290

Pinker, S. (1994) “The Language of Instinct, The New Science of Language and Mind”, Harmondsworth: Penguin Books Ltd.

Pomerleau, D.A. (1994) “Neural Network-Based Vision for Precise Control of a Walking Robot”, Machine Learning, Vol. 15, No. 2, pp. 125-136

Popper, K.R. (1959) “The Logic of Scientific Discovery”, London: Routledge. This is the 1992 reprint of the 1959 translation into English of the 1934 original (in German) “*Logik der Forschung*”.

Premack, D. (1976) “Intelligence in Ape and Man”, Hillsdale, NJ: Lawrence Erlbaum Associates

Puterman, M.L. (1994) “Markov Decision Processes, Discrete Stochastic Dynamic Programming”, New York: John Wiley and Sons

Razran, G. (1971) “Mind in Evolution: An East-West Synthesis of Learned Behavior and Cognition”, Boston: Houghton Mifflin Company

Rescorla, R.A. (1988) “Pavlovian Conditioning, It's Not What You Think It Is”, American Psychologist, Vol. 43, No. 3, pp. 151-160

Restle, F. (1962) “The Selection of Strategies in Cue Learning”, Psychological Review, Vol. 69, No. 4, pp. 329-343

Reynolds, V. (1976) “The Origins of a Behavioural Vocabulary: the Case of the Rhesus Monkey”, Journal of the Theory of Social Behaviour, Vol. 6, No. 1, pp. 105-142

Ring, M. (1993) “Two Methods for Hierarchy Learning in Reinforcement Environments”, in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 148-155

Riolo, R.L. (1991) “Lookahead Planning and Latent Learning in a Classifier System”, in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior “From Animals to Animats”, pp. 316-326

Rivest, R.L. and Schapire, R.E. (1990) “A New Approach to Unsupervised Learning in Deterministic Environments”, in: Kodratoff, Y. and Michalski, R.S.

(Eds.) “Machine Learning: An Artificial Intelligence Approach, Volume III”, San Mateo, CA: Morgan Kaufmann, pp. 670-684

Roitblat, H.L., Moore, P.W.B., Nachtigall, P.E. and Penner, R.H. (1991) “Biomimetic Sonar processing: From Dolphin Echolocation to Artificial Neural Networks”, in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior “From Animals to Animats”, pp. 66-76

Rosenblatt, F. (1962) “Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms”, New York: Spartan Books

Rosenblatt, J.K. and Payton, D.W. (1989) “A Fine-Grained Alternative to the Subsumption Architecture for Mobile Robot Control”, Proceedings of the IEEE/INNS International Joint Conference on Neural Networks, Vol. II, pp. 317-323

Ross, S. (1983) “Introduction to Stochastic Dynamic Programming”, New York: Academic Press

Rumelhart, D.E., Hinton, G.E. and Williams, R.J. (1986) “Learning Representations by Back-Propagating Errors”, Nature, Vol. 323, pp.533-536

Schmajuk, N.A. (1994) “Behavioral Dynamics of Escape and Avoidance: A Neural Network Approach”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 118-127

Schölkopf, B. and Mallot, H.A. (1995) “View-Based Cognitive Mapping and Path Planning”, Adaptive Behavior, Vol. 3, No. 3 (Winter 1995), pp. 311-348

Schoppers, M.J. (1987) “Universal Plans for Reactive Robots in Unpredictable Environments”, Proceedings of the Tenth International Joint Conference on Artificial Intelligence, Milan, August 23-28, 1987, pp. 1039-1046

Schoppers, M.J. (1989) “In Defense of Reaction Plans as Caches” AI Magazine, Vol. 10, No. 4 (Winter 1989), pp. 51-59

Schoppers, M.J. (1995) “The Use of Dynamics in an Intelligent Controller for a Space Faring Rescue Robot” Artificial Intelligence, Vol. 73, pp. 175-230

Schwartz, B. (1989) “Psychology of Learning and Behavior”, New York: W.W. Norton & Co. (third edition)

Scutt, T. (1994) “The Five Neuron Trick: Using Classical Conditioning to Learn How to Seek Light”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 364-370

Sejnowski, T.J., Koch, C. and Churchland, P.S. (1988) “Computational Neuroscience”, Science, Vol. 241, pp. 1299-1306

Sejnowski, T.J. and Rosenberg, C.R. (1987) “Parallel Networks that Learn to Pronounce English Text”, Complex Systems, Vol. 1, pp. 145-168

Shen, W-M. (1993) “Discovery as Autonomous Learning from the Environment”, Machine Learning, Vol. 12, pp. 143-165

Shen, W-M. (1994) “Autonomous Learning from the Environment”, New York: Computer Science Press/W.H. Freeman and Company

Shettleworth, S.J. (1975) “Reinforcement and the Organization of Behavior in Golden Hamsters: Hunger, Environment, and Food Reinforcement”, Journal of Experimental Psychology: Animal Behavior Processes, Vol. 104, No. 1, pp. 56-87

Simon, H.A. (1983) “Why Should Machines Learn?”, in: Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (Eds.), “Machine Learning: An Artificial Intelligence Approach”, Palo Alto, CA: Tioga Publishing Company, pp. 25-37

Singh, S.P. and Sutton, R.S. (1996) "Reinforcement Learning with Replacing Eligibility Traces", Machine Learning, Vol. 22, pp. 123-158

Skinner, B.F. (1948) "Superstition in the Pigeon", Journal of Experimental Psychology, Vol. 38, pp. 168-172

Sutton, R.S. (1988) "Learning to Predict by the Methods of Temporal Differences", Machine Learning, Vol. 3, pp. 9-44

Sutton, R.S. (1990) "Integrated Architectures for Learning, Planning, and Reacting Based on Approximating Dynamic Programming", in: Porter B.W. and Mooney, R.J. (Eds.) Machine Learning, Proceedings of the Seventh International Conference on Machine Learning, Morgan Kaufmann Publishers, pp. 216-224

Sutton, R.S. (1991) "Reinforcement Learning Architectures for Animats", in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior "From Animals to Animats", pp. 288-296

Sutton, R.S. (1992) "Introduction: The Challenge of Reinforcement Learning", Machine Learning, Vol. 8, pp. 225-227

Tenenberg, J., Karlsson, J. and Whitehead, S.D. (1993) "Learning via Task Decomposition", in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) "From Animals to Animats 2" Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 337-343

Thistlethwaite, D. (1951) "A Critical Review of Latent Learning and Related Experiments", Psychological Bulletin, Vol. 48, No. 2, pp. 97-129

Thorndike, E.L. (1911) "Animal Intelligence", New York: Macmillan Company (1965 facsimile of 1911 edition, New York and London: Hafner Publishing Company)

Tinbergen, N. (1951) “The Study of Instinct”, London: The Oxford University Press

Toates, F. (1994) “What is Cognitive and What is not Cognitive”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 102-107

Tolman, E.C. and Honzik, C.H. (1930a) “Introduction and Removal of Reward, and Maze Performance in Rats” University of California Publ. Psychol., Vol. 4, pp 257-275 (cited Bower and Hilgard, 1981, pp 338-340)

Tolman, E.C. and Honzik, C.H. (1930a) “"Insight" in Rats” University of California Publ. Psychol., Vol. 4, pp 215-232 (cited Bower and Hilgard, 1981, pp 335-338)

Tolman, E.C. (1932) “Purposive Behavior in Animals and Men” New York: The Century Co. (Century Psychology Series)

Tolman, E.C. (1938) “The Determiners of Behavior at a Choice Point”, Psychological Review, Vol. 45, No. 1, pp. 1-41

Tolman, E.C. (1948) “Cognitive Maps in Rats and Men”, Psychological Review, Vol. 55, pp. 189-208

Travers, M. (1989) “Animal Construction Kits”, in: Langton, C. (Ed.) “Artificial Life, SFI Studies in Sciences of Complexity”, Reading, MA: Addison-Wesley Publishing Co., pp. 421-442

Tyrrell, T. (1993) “Computational Mechanisms for Action Selection” University of Edinburgh, Ph.D. thesis

Tyrrell, T. (1994) “An Evaluation of Maes’s Bottom-Up Mechanism for Behavior Selection”, Adaptive Behavior, Vol. 1, No. 4, pp. 387-420

Venturini, G. (1994) “Adaptation in Dynamic Environments Through a Minimal Probability of Exploration”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 371-379

Verplanck, W.S. (1954) “Burrhus F. Skinner”, in: Estes, W.K., Koch, S., MacCorquodale, K., Meehl, P.E., Mueller, C.G., Schoenfeld, W.N. and Verplanck, W.S. (Eds.) “Modern Learning Theory: A Critical Analysis of Five Examples”, New York: Appleton-Century-Crofts, pp. 267-316

Vershure, P.F.M.J. and Pfeifer, R. (1993) “Categorization, Representations, and The Dynamics of System-Environment Interaction: A Case Study in Autonomous Systems”, in: Meyer, J-A., Roitblat, H.L. and Wilson, S.W. (Eds.) “From Animals to Animats 2” Proceedings of the Second International Conference on Simulation of Adaptive Behavior , pp. 210-217

Walter, W.G. (1953) “The Living Brain”, London: Gerald Duckworth & Co. Ltd.

Watkins, C.J.C.H. (1989) “Learning from Delayed Rewards”, King's College, Cambridge University (Ph.D. thesis)

Watkins, C.J.C.H. and Dayan, P. (1992) “Technical Note: *Q*-learning”, Machine Learning, Vol. 8, pp. 279-292

Webb, B. (1994) “Robotic Experiments in Cricket Phonotaxis”, in: Cliff, D., Husbands, P., Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior “From Animals to Animats 3”, pp. 45-54

Widrow, B. and Hoff, M.E. (1960) “Adaptive Switching Circuits”, 1960 WESCON Convention Record, Part IV, pp. 96-104

Wilson, S.W. (1985) “Knowledge Growth in an Artificial Animal”, in: Greffenstette, J.J. (Ed.) “Proceedings of the First International Conference on

Genetic Algorithms and Their Applications”, Hillsdale, NJ: Lawrence Erlbaum Associates, pp. 16-23

Wilson, S.W. (1991) “The Animat Path to AI”, in: Meyer, J-A. and Wilson, S.W. (Eds.) Proceedings of the First International Conference on Simulation of Adaptive Behavior “From Animals to Animats”, pp. 15-21

Whitehead, S.D. and Ballard, D.H. (1991) “Learning to Perceive and Act by Trial and Error”, Machine Learning, Vol. 7, pp. 45-83

Whitehead, S.D. and Lin, L-J. (1995) “Reinforcement Learning of Non-Markov Decision Processes”, Artificial Intelligence, Vol. 73, pp. 271-306

Wyatt, J. (1995) “Issues in Putting Reinforcement Learning onto Robots” Presented at AISB 1995 Robotics Summer School. DAI, Edinburgh University, March 1995.

11. SUBJECT and AUTHOR INDEX

A

α *See* reinforcement rate
ACP model 43
action 73
action cost 74, 113, 149
action dispersion probability 156, 157
action patterns 75
action repetition rate 156
action selection 80
Action Selection Mechanisms 12, 105
activation 48, 70
activation trace 69
actuators 73
Adisp *See* action dispersion probability
ADROIT 14
AGIL 38
Agre, P.E. 11, 94
Albus, J.S. 16, 42
ALECSYS 38
ALP 51, 89, 188
alternative path experiments 126
Altman, J. 20
ALVINN 41
analogic-matching 50
animat 11
animat-centric view 127
appetitive actions 13
approximate universal plan 219
Arbib, M.A. 20, 72
Arep *See* action repetition rate
Artificial Neural Networks 38
Association List 208
association unit 39
at (@) notation 97, 127
Atkeson, C.G. 35, 200
autonomy (behavioural) 133
aversion 27, 76, 214
Aylett, R. 10

B

β *See* extinction rate
backpropagation 40
Baerends, G.P. 14
Baird, L.C. 43
Ball, N. 38
Ballard, D.H. 30, 94
Barto, A.G. 25, 26
basal level threshold 103, 137
Beasley, D. 37
Becker, J.D. 49, 57
Beer, R.D. 20
behavior language 15
behavior units 14
behaviour based model 11
Behaviour List 93, 102
behaviour priority 84
Berkson, W. 68
Bersini, H. 38
best_valence_level 213
bestcost 125, 138
bid amount 37
biological plausibility 21
biomimetics 21
Bishop, N. 78
Blackman, D. 69, 80, 81
blind-trials 66
Boden, M.A. 106
Bolles, R.C. 23
Boltzmann distribution 33, 170
Bonarini, A. 55
Bond, A.H. 49
Booker, L.B. 35, 149
Bower, G.H. 18, 23, 29, 45, 58, 81, 168, 209, 221
Brooks, R.A. 9, 11, 14, 31, 57, 216
Buchanan, B.G. 222
bucket-brigade algorithm 37, 104

C

cache (universal plan) 218
Campbell, B.A. 214
Carbonell, J.G. 24, 223
Catania, A.C. 43, 46
cathexis 47, 77, 208
causality 127
CFSC2 55
Chapman, D. 94
Chesters, W. 41

Chiel, H.J. *20*
Chrisman, L. *30*
Churchland, P.S. *20*
circadian rhythm 73
classical conditioning 23, 24, 209
Classifier Systems 35
classifiers 36
Cliff, D. *10, 20, 106*
CMAC 42
Cobas, A. *20, 72*
cognitive map 48
cognitive viewpoint 44
Colombetti, M. *38*
composite action 53, 100
compound actions 74
computational neuroethology 20
computational neuroscience 21
conditioned reinforcer 80
conditioned stimulus 24
conflictor link 17
Connection Machine 89
connectionism 38
Connell, J. *30*
consummatory actions 13
corroboration 60
cost estimate 79, 113
creation (hypothesis) 62, 133, 143
creation bonus 110, 114, 143
credit assignment problem 30
Crisler, G. *67*
critic 222
cumulative reward 194
curiosity 199

D

Δ (δ) *See* rebuildpolicynet
Dawkins, R. *38*
Dayan, P. *32*
default (exploratory) behaviours 85
deictic marker 94
deictic representation 94
Dewsbury, D. *20*
differentiation 25, 51, 62
discriminated operant 42
Dolhinow, P.J. *78*
don't care (#) 36, 99
Dorigo, M. *38*
DoWorldAction 148
Drescher, G.L.49, 57, 58, 89, 107, 223
drive 28
dual path blocking 117
dual-path extinction 222
Dynamic Expectancy Model 21, 57, 86, 89, 146
Dynamic Policy Map 78, 112, 138, 218
dynamic programming 32, 170
Dyna-PI 149, 205
Dyna-Q+ 34, 114, 149, 195, 206
DynaWorld/Standard 149

E

ε *See* basal level threshold
elicitor-cathexis 48
encapsulation 70
Estes, William K. 29, 168
etho-engineer 223
ethogram 11, 57
ethology 12
Etzioni, O. *223*
Execution Cycle 131
expectancy 46
expectancy postulates 46
expectancy theory 45, 120, 208
expectandum 47
Expectation Based Learning 22, 223
experience replay 31
experimental extinction 25
Explanation Based Learning 223
exploration bonus 35, 114
explore-exploit tradeoff 33, 38
extended context 54
extended result 54
extinction 46, 77, 81
extinction (goal) 184
extinction rate 61, 110, 188

F

falsification 68
FDMSSE 32, 64
FIFO buffer 49
Fikes, R.E. *115*
filter.exe 155
first appearances effect 128
fixed action patterns 13
fixed interval schedule 81
fixed ratio schedule 81
fixed schedule experiments 158
Foner, L. *72*
forgetting 63

Franceschini, N. *20*
Friedman, L. *14*
FSMSSE 32, 64

G

γ *See* selection factor
Gaussier, P. *41*
General Problem Solver 107
generalisation 25, 129
generalised inference 47
genetic algorithm 38, 129
genetic crossover 38
Genghis (robot) 31
Gil, Y. *223*
Ginsberg, M.L. *219*
Giszter, S. *31*
goal (definition) 76
goal cancellation level 126, 141, 190
goal extinction 77, 141, 188
goal extinction point 77, 124
Goal List 93, 104
goal priority 76, 104, 112, 153, 210
goal recovery mechanism125, 141, 189
goal satisfaction 76, 105
goal setting behaviours 84, 102, 137
goal strength function 210
goal valence 76
goal_recovery_rate 125
GOFAI 106
Goldberg, D.E. *35*
Goodwin, R. *211*
graph-search 112
Guillot, A. *48*

H

habituation 19, 193
Haigh, K.Z. *211*
Hall, J.F. *23*
Hallam, B.E. *14*
Hallam, J.C.T. *14*
Halperin, J.R.P. *14*
Harnad, S. *43*
Harrison, P. *130*
Hartley, R. *20*
hash table 95
Hayes, G. *41*
Heitkötter, J. *37*
Hergenhahn, B.R. *23*
HERO (robot) 211
Hess, E.H. *20*
Highfield, R. *72*
Hilgard, E.R. *18, 23, 29, 45, 58, 81, 168, 209, 221*
Hinde, R.A. *78*
Hinton, G.E. *40, 41*
Hoff, M.E. *30*
Holland, J.H. *35, 38*
Honzik, C.H. *159, 201*
Hubel, D.H. *72*
Hull, Clark L. 28, 104, 113
Humphrys, M. *35*
Husbands, P. *10*
hypo_activation_trace 108
hypo_age 108, 114
hypo_cneg 109
hypo_cpos 109
hypo_first_seen 108
hypo_identifier 108
hypo_last_seen 108
hypo_maturity 114, 128
hypothalamus 72
Hypothesis List 93, 105
hypothesis template 127
Hypothetico-Corroborative 59, 69
Hypothetico-Deductive 59, 69

I

implicit activation 107, 133
Imprinting 20, 102
induced valence 78
induction, learning by 129
inference 47
innate behaviour 102
innate releasing mechanism 13
Input Token List 93, 94, 148
instrumental conditioning 42
instrumental learning 127
intelligence without reason 11
internal kernel 98

J

Jahoda, G. *69*
JCM 49, 89
Jochem, T.M. *41*
Johnson, C.R. Jr. *222*
joint probability 130
Jones, T.L. *55*

K

Kaelbling, L.P. *29*
Kamin, L.J. *62*

Karlsson, J. *35*
Kearsley, G. *23*
Keirsey, D.M. *16*
kernels 49
Khepera (robot) 41
King, N. *128*
Kirsh, D. *11*
Klahr, D. *66*
Kleitman, N. *67*
Klopf, A.H. *26, 43*
Knoblock, C.A. *223*
Knuth, D.E. *95*
Koch, C. *20*
Koch, S. *28*
Kohonen feature map 38
Krechevsky, I. *65*
Kuokka, D.R. *223*

L

λ *See* learning probability rate
Langley, P. *24*
Langton, C.G. *9*
latent extinction 81
latent learning 45, 55, 197
law of effect 26
learning element 222
learning probability rate 143, 156, 158, 164, 213
Learning, definition of 18
learning-to-learn 211
Lee, H.B. *20*
Levine, M. *66*
Levy, S. *9*
Liaw, J-S. *72*
Lieberman, D.A. *23*
Lin, L-J. *30, 31*
LISP 146
list (SRS/E) 90
list element values 90
list elements 90
Littman, M.L. *149*
LIVE system 56
Lofts, B. *73*
log file 155
logic of scientific discovery 67
Long Term Memory 49
Lorenz, K.Z. *13*, 20
lower confidence bound 129, 144
Lprob *See* learning probability rate

M

MacCorquodale, K. *45, 110, 212*
machina docilis 26
machina speculatrix 26
machine learning, model of 223
Maclin, R. *34*
Maes, P. *9, 10, 11, 16, 31, 57, 72, 77*
Mahadevan, S. *30*
Mallot, H.A. *72*
MANIAC 41
marginal attribution 54
Markov Decision Process 222
Markov environment 31
markov property 32
Masterson, F.A. *214*
Mataric, M.J. *10*
mathematical learning theory 29
maturation 19, 101, 102
maturity threshold 128, 130, 144
McCallum, R.A. *30*
McCulloch, W.S. *38*
McFarland, D. *75*
McGonigle, B. *41*
means ends analysis 107
means-end-field 48
means-ends-readiness 107
Meehl, P.E. *45, 110, 212*
meta-level learning 211
μ-experimentation 60
μ-experiments 59
Meyer, J-A. *10, 48*
μ-hypothesis 59, 60, 93, 105
Michalski, R.S. *99*
Microsoft Windows 146
Millán, J. del R. *31*
Minsky, M. *18, 30, 40, 147*
Minton, S. *223*
Mitchell, T.M. *30, 222*
mnemonization 46, 61, 208
modus ponens 46
modus tolens 67
Monte-Carlo method 32
Moore, A.W. *35, 200*
Moore, P.W.B. *21*
Morgan, J.S. *43*
motivation 28
motivational kernel 52
Mott, D.H. *49, 57*

Mowrer, O.H. *44*
multiple goals 209
multiple-path 184
Mura, F. *20*
mutation 38

N
Nachtigall, P.E. *21*
need strength 47
negatively accelerating curve 110
Nehmzow, U. *41*
net response strength 28
Newell, A *107*
Nilsson, N.J. *40, 112, 115*
Norman, D.A. *89*
novel event 62, 126, 143
novelty 52

O
object permanence 53, 98, 111
occam's razor 127
occult occurrence 69, 107, 110, 130
Olson, M.H. *23*
one-shot learning 29, 169, 221
operant conditioning 42
originator 59
oscill 113, 170
oscillatory factor 28, 113

P
pain 214
panic reaction 125
Papert, S. *40*
parturition 63, 83, 167
path blocking 126, 184, 193
pathavailable 138
pattern extraction 126, 128
Pavlov, Ivan P. 24
Payton, D.W. *16*
Peng, J. *34, 149*
Penner, R.H. *21*
Perceptron 39
performance element 222
persistence (of behaviour) 124
Pfeifer, R. *106*
phobia 215
Piaget, Jean 53, 58
Pinker, S. *11*
Pitts, W.H. *38*
place cells 72
place learning 45, 201
play 78, 114
policy map 57, 116, 218
policy value 79, 80, 115, 125
Pollack, J. *10*
Pomerleau, D.A. *41*
POP-2 52
Popper, K.R. *67*, 127, 211
predecessor link 17
prediction 60
Prediction List 94, 109
Premack, D. *12*
primary behaviours 84, 102, 137
primary generalization 47
primary item 53, 73
primary reinforcers 217
primitive item 94
prioritized sweeping 35, 200
punishment 27
Puterman, M.L. *31*

Q
θ *See* lower confidence bound
Θ *See* upper confidence bound
Q-learning 32, 116
QMC Mk. IV robot 75
quality-values 32

R
R2 (robot) 41
rand() 156
random walk 156
rapid extinction conundrum 221
raw_sign_prob 100
Razran, G. *18, 26, 192, 217*
reaction potential 48
reactive agent 11
rebuildpolicynet 116, 136, 138, 143
REBUILDPOLICYTRIP 117, 123, 138, 191
recency 111
redundant attribution 55
reification 74
reinforcement 61, 217
reinforcement learning 26, 116
reinforcement measure 61, 110
reinforcement rate 61, 110
Rescorla, R.A. *26*
Response List 93, 100, 149
response_activation_trace 101
response_cost 101
response_identifier 101

response_string 101, 148
Restle, F. 66
Reynolds, V. 75
Rhesus Monkey 75
Riolo, R.L. 38, 55, 62, 149
Rivest, R.L. 219
Rogue 211
Roitblat, H.L. 10, 21
Rosenberg, C.R. 41
Rosenblatt, F. 39
Rosenblatt, J.K. 16
Ross, S. 32, 170
rseed 156, 167
Rubik's Cube 219
Rumelhart, D.E. 40

S

sampling strategy 127, 143
Schapire, R.E. 219
schema confidence weight 50, 108
schema representation 49
Schmajuk, N.A. 44
Schölkopf, B. 72
Schoppers, M.J. 218, 220
Schwartz, B. 23, 214
Scutt, T 26
secondary cathexis 48, 208
secondary reinforcers 218
Sejnowski, T.J. 20, 41
selection factor 114, 172
sensitisation 19
sensitive period 20
sensory preconditioning 209
Shavlik, J.W. 34
Shen, W-M. 56, 62
Shettleworth, S.J. 75, 128
Short Term Memory 49, 72, 97
siamese fighting fish 14
Sibly, R.M. 75
sign 70
Sign List 93
sign_activation_trace 99
sign_count 99
sign_first_seen 99
sign_identifier 99
sign_last_seen 99
sign-gestalt 73
Simmons, R. 211
Simon, H.A. 18, 107
Singh, S.P. 34
single-hypothesis assumption 66
situated agent 11
Skinner box 43, 126, 130, 150, 194
Skinner, B.F. 42, *69*
Smith, R.G. 222
specialisation (learning by)51, 128, 144
spontaneous recovery 82
spreading activation 17, 112, 217
spreading valence 79, 112
S-R behaviourism 11, 27, 44, 217
SRS/E 22, 89, 130, 208
stationary policy 32
stimulus sampling theory 29, 160, 168
STM to LTM encoding 51
stochastic policy 32
strength of expectancy 110
strength value 37
STRIPS 115
sub-goal 79
sub-set sampling assumption 66
substantia nigra 72
subsumption architecture 14, 31
subsumption point 134
sub-valence 112
successor link 17
Sudarshan, K. 20
superstitious learning 69, 107
Sutton, R.S. 25, 26, 29, 30, 34, 57, 114, 149, 171, 185, 205
synthetic item 53, 98

T

tabula rasa 86, 126
Template List 211
temporal differences method 30, 34
temporal discrimination 69, 96
Tenenberg, J. 35
terminating condition 63, 133
Thistlethwaite, D. 200
Thorndike, E.L. 26
Thorpe, C.E. 41
three-spined stickleback 13
three-term contingency 46
threshold unit 39
time_shift 108, 142, 143
timebase shifting 62, 143
Tinbergen, N. 57, 71
Toates, F. 44

token 70
token negation 97
token_first_seen 95
token_identifier 95
token_last_seen 95
token_prob 95
token_string 95
tokenisation 70, 94
tokens 69
Tolman, E.C. *44*, *48*, *65*, 101, *159*, *201*, *216*
top-goal 76, 93, 104
Torras, C. *31*
Towers of Hanoi 56
Travers, M. *14*
trial and error 27, 54, 85, 86, 124, 156
triune brain hypothesis 16
Tyrrell, T. *12*, *15*, *18*, *77*

U

unconditioned reflex 24
unconditioned stimulus 24
unexpected event 62, 126
universal plans 218
unsupervised learning 126
unvalenced actions 125
upper confidence bound 129, 144

V

valence 47, 76
valence break point 80, 125, 141, 189
valence level 112
valence level pre-bias 213, 217, 223
valence path 115, 155
VALENCE_BREAK_POINT_FACTOR 125
variable interval schedule 81
variable ratio schedule 81
Veloso, M.M. *211*
Venturini, G. *38*
Verplanck, W.S. *43*
Vershure, P.F.M.J. *106*
Visual C++ 146
VL21 99

W

Ω *See* goal cancellation level
Walter, W.G. *26*
Watkins, C.J.C.H. *32*, *57*, *83*, *206*
Watson, John B. 27
Weaver, S.E. *43*
Webb, B. *20*
Wettersten, J. *68*
Whitehead, S.D. *30*, *35*, *94*
Widrow, B. *30*
Wiesel, T.N. *72*
Williams, R.J. *34*, *40*, *149*
Wilson, S.W. *10*, *11*, *38*
W-learning 35
world tally 153
Wyatt, J. *169*

X

XBL *See* Expectation Based Learning

Y

Ψ *See* maturity threshold

Z

Zrehen, S. *41*