Hierarchy and Sequence vs. Full Parallelism in Action Selection

Joanna Bryson
MIT AI Lab
Cambridge, MA 02139
joanna@ai.mit.edu

Abstract

Hierarchical organization has become an unfashionable model of intelligent control within some communities of both natural and artificial intelligence. What has replaced it are models based on parallel distributed processes, both neural and behavior based, or dynamical systems theory, which denies modularity, let alone rigorous structure.

In this paper we present experimental results demonstrating an artificial reactive hierarchy-based system that outperforms fully parallel systems in a highly dynamic environment with a large number of conflicting goals. This work is conducted in Tyrrell's (1993) Simulated Environment and can be seen as an extension of his work on comparing action selection mechanisms. We observe that the hierarchical strategy has also been well demonstrated in nature. We argue that, for complex intelligences, preserving full reactivity may not be worth the cost in terms of the complexity of action selection.

1. Introduction

The use of both hierarchical and fixed sequential orderings of behavior for action selection has been postulated since the time of the early ethologists (Lorenz, 1981). The advantage of such systems are clear: they reduce the combinatorial complexity of control. The set of actions which may be selected from is determined by context, including state internal to the agent. Once an agent is attending to a particular strategy, actions associated with alternative strategies are unlikely to interfere with behavior production.

Early artificial intelligence research followed these hierarchical models, but ran into great difficulty in coping with dynamic environments in real time. Action selection was based on constructive planning, which was in turn based on heuristic search (Newell and Simon, 1981). This system has been both demonstrated and theoretically proven intractable without major alteration (Nilsson, 1984; Chapman, 1987). These results led researchers both inside and outside of artificial intelligence towards a new paradigm of *reactive intelligence* (Georgeff and Lansky, 1987; Agre and Chapman, 1990; Maes, 1991; Rosenschein and Kaelbling, 1995; Hendriks-Jansen, 1996; van Gelder, 1998; Bryson, in press). A reactive system is designed from the beginning to be situated in a complex, dynamic environment, which it must constantly monitor and to which it must instantly react.

A basic premise for many of these researchers is that a truly reactive system must have all aspects of its intelligence constantly active and sampling the environment (Maes, 1991; Brooks, 1991; Tyrrell, 1993; Hendriks-Jansen, 1996). This is associated with the *behavior based* approach to reactive intelligence, in which intelligence is composed of relatively simple modules: tightly coupled units of sensing and action. These behaviors are the perceptual system of the agent. If they are inactive, relevant information in the environment will be ignored. Some authors also reject hierarchy in a controller altogether, on the basis that it results in bottlenecks, staged responses, and governed, unreactive behavior (Maes, 1991; Hendriks-Jansen, 1996). Others design systems that exploit hierarchical order, but still maintain constant parallel processing (Tyrrell, 1993; Blumberg, 1996). Such approaches still neglect the main advantage of hierarchical organization described above.

This paper demonstrates that an agent may in fact ignore otherwise significant portions of its environment in pursuit of more important goals and still be very successful in a highly complex, dynamic and hostile environment. We use an established test base for reactive control, Tyrrell's Simulated Environment (Tyrrell, 1993). We compare two representative architectures originally developed for and tested on mobile robots. The representative fully informed architecture is also the architecture previously tested as best in Tyrrell's SE, the Extended Rosenblatt and Payton Architecture (Tyrrell, 1993); the architecture with selective attention is our own (Bryson and McGonigle, 1998). We then draw on biological evidence that animals (including humans) are not fully reactive, and argue that the complexity of the intelligent system necessary to handle full environmental information makes a selective hierarchy preferential to a fully parallel organization.

2. Experimental Approach

This section describes the tasks and environment used in running the action-selection experiments. It also describes the architectures representing the two forms of cognitive organization for control: fully informed vs. selectively attentive.

2.1 The Simulated Environment

Tyrrell's Simulated Environment, the SE, specifies an action-selection task which requires an agent to manage a large number of conflicting goals. He defines an environment in which a small omnivorous animal is required to survive and breed. He defines six subproblems for the animal to solve.

1. Finding sustenance. In addition to water, there are three forms of nutrition, satisfied in varying degrees by three different types of food.
2. Escaping predators. There are feline and avian predators, which have different perceptual and motion capabilities.
3. Avoiding hazards. Benign dangers in the environment include wandering herds of ungulates, cliffs, poisonous food and water, temperature extremes and darkness. The environment also provides various forms of shelter including trees, grass, and a den.
4. Grooming. Grooming is necessary for homeostatic temperature control and general health.
5. Sleeping at home. The animal is blind at night; its den provides shelter from predators and other hazards, and helps the animal maintain body temperature while conserving energy.
6. Reproduction. The animal is male, thus its reproductive task is reduced to finding, courting and inseminating mates. Attempting to inseminate unreceptive mates is hazardous.

The success of the animal is considered to be the number of times it mates in a lifetime. This is highly correlated with life length, but long life does not guarantee reproductive opportunities.

These problems vary along several axes: homeostatic vs. non-homeostatic, dependency on external vs. internal stimuli, periodicity, continual vs. occasional expression, degree of urgency and finally, whether prescriptive or proscriptive with regard to particular actions. In addition to these problems, the environment is highly dynamic. Food and water quantities, temperature and light vary, and animals move. Sensing and action are uncertain. Perception in particular is extremely limited and severely corrupted with noise; the animal usually misperceives anything not immediately next to it, unless it chooses to spend time and expose itself by rearing up and "looking around" in an uncovered area.

Tyrrell separates the problems of learning and navigation from the problem of action selection by providing these elements in his simulation. Thus the animal has available as primitives a direction in which it thinks it remembers its home or recently observed food and water. The animal's sense of location with respect to its den decays over time and distance, thus keeping track of its bearing is a part of the "sleeping at home" sub-problem, even though the precise mechanism of navigation is not explicit.

The results of the animal's performance is heavily dependent on chance implementation details of the environment. In his thesis, Tyrrell attempts to check the independence of his results from these details by running tests in four different "worlds". Besides the standard model he first designed, there are three variants. As documented (p. 162) these vary in the following ways.

1. Perception is altered by affecting the animal's visibility according to the time of day, the amount of vegetation, and the animal's own activity.
2. The noise variance for navigational information is tripled, and the size of the remembered map is halved.
3. Motor control is compromised by making it more probable that incorrect actions are taken, and by changing the conspicuousness and energy consumption of various actions.

2.2 The Extended Rosenblatt and Payton Action Selection Mechanism

Tyrrell used this environment to test five different architectures from ethology and artificial intelligence. The first four were drive theory (Hull, 1943), the psycho-hydraulic system of Lorenz (1950, 1981), spreading activation networks of Maes (1991), and the connectionist, hierarchical, feed-forward networks of Rosenblatt and Payton (1989). The fifth, which he recommends not only as best but as nearly optimal, is his own extension of the latter system, the Extended Rosenblatt and Payton architecture (ERP). In this model, all the constituent elements operate continuously in parallel. These elements constantly evaluate their own relevance with respect to the current environment and the animal's needs. This

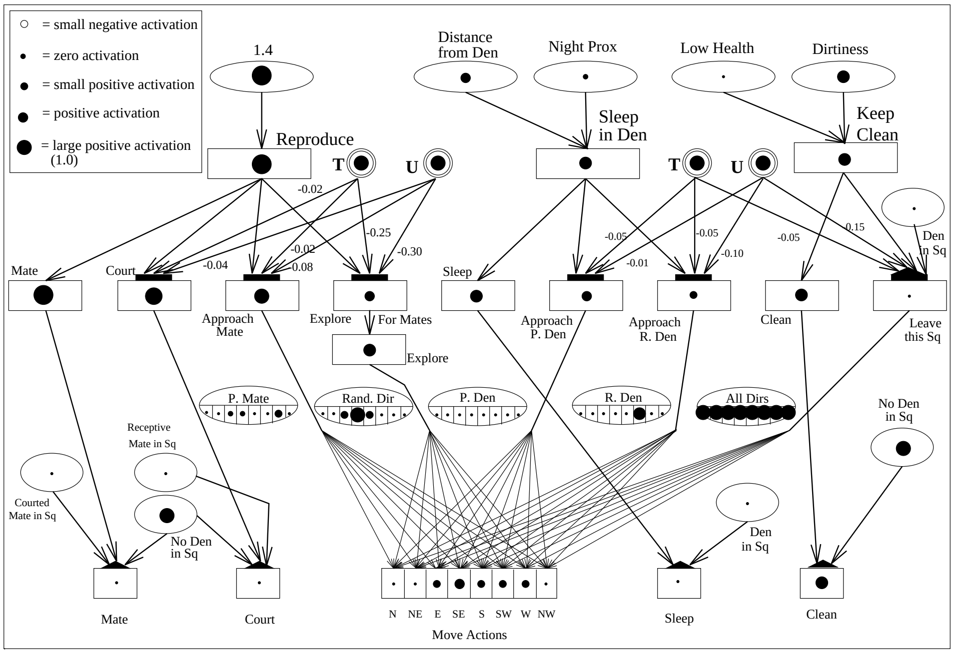

Figure 1: A fraction of the Extended Rosenblatt & Payton action selection network for controlling an animal in Tyrrell's Simulated Environment. **T** and **U** are temporal and uncertainty penalties for actions that take time or pursue a goal uncertain to succeed. P stands for perceived, R for remembered. (Tyrrell, 1993, page 247).

relevance is used to weight the element's recommendation for a next action, which is also constantly in computation. Ultimately a winner-take-all algorithm is used to determine the expressed action, which may be a combination of the desires of several elements.

Tyrrell rejects the subsumption architecture of Brooks (1986) as being too ill-defined to be considered a true architecture for action selection. Regardless of the accuracy of this criticism, the ERP shares many features in common with subsumption, such as full concurrency. The primary difference is the system of prioritization between behaviors: the ERP's is more orderly with a clear directionality between any two behaviors. Subsumption allows for cycles within a single layer, though in practice these seldom exist. Also, the somewhat complicated system of inhibition and suppression in subsumption is replaced by a uniform pipeline metaphor in the ERP, with no obvious loss of expressiveness.

Tyrrell's work and the ERP architecture in particular have been very influential. For example, Humphrys (1996) evaluates a number of strategies for final action selection, favoring a selection on the basis of maximizing the least unhappiness of the various units. Blumberg (1996) has divided his behavior elements into classes that do not need to compete for physical resources on the agent, and may therefore express themselves simultaneously, as in the joyous or woeful gaits for humans and dogs. However, the experiments in this paper compare directly to to the Extended Rosenblatt and Payton architecture. This is for two reasons: first, because it had already been stringently tested against four other popular models of actions selection, and second, because a solution for the SE implemented by its original author already exists, removing several of the sources of bias possible in such a comparison exercise.

The Rosenblatt & Payton model was developed and first tested as a robot architecture (Rosenblatt and Payton, 1989). The main feature of this architecture is what Tyrrell terms a *free-flow hierarchy.* Nodes in the hierarchy receive activation from internal and external stimuli and higher elements of the hierarchy. They pass energy to their children. What differentiates this model from the drive model or other hierarchies is that no decisions are made until the leaf or action nodes. This allows for the selection of *compromise candidate* behaviors: behaviors that satisfy or express multiple drives. See for example Figure 1.

Tyrrell's extensions to Rosenblatt & Payton's original system are as follows:

- The addition of penalties for temporal delays and uncertainty of rewards for items that are distant or based on poor memory or perception. This allows the animal to chose a small but apparent near food source rather than a large but distant, poorly remembered food source.

- The creation of explicit penalties for appetitive vs. consummatory actions[1].

[1]Failing to favor consumatory actions is also a problem for Maes' architecture, which performs particularly poorly in the SE.

- The addition of a new rule for combining the inputs to a single node that explicitly distinguished the maximum of inputs adding positive energy to a particular node and the minimum of inputs opposing the node. This was significantly more successful than simple summation of the activations (See further Humphrys, above).

2.3 Edmund

The ERP is able to model a hierarchical structure of data flow, but it does not represent decisions or a focus of attention — normally considered basic elements of a cognitive model. This reflects a strong belief in the agent community that traditional hierarchical structure is overly rigid and leads to neglect, and must be replaced by more dynamic, fully parallel architectures (Maes, 1991; Brooks, 1991; Hendriks-Jansen, 1996). The reason Tyrrell argues that the ERP is a near optimal architecture is due to its ability to monitor all goals simultaneously, thus enabling it to select more optimal actions in particularly contentious situations. He emphasizes the idea of *compromise candidate* actions, which satisfy more than one set of goals.

The argument of this paper is that an adequately reactive hierarchy can surpass the performance of a fully reactive system, despite the loss of information and compromise candidates. The simplification in control structure that results from taking advantage of selective attention makes an intelligent approach easier to develop, whether by natural evolution or by human design.

The experiments in this paper use an agent architecture called Edmund (Bryson and McGonigle, 1998). Edmund is a behavior-based architecture with Parallel rooted, Ordered, Slip-stack Hierarchical (POSH) action selection. POSH action selection exploits hierarchy both to combat the combinatorics of action selection and provide persistence in behavior. At the same time, it maintains a high level of reactivity through its parallel roots and strictly limited stack size. Edmund is available on line from http://www.ai.mit.edu/~joanna/edmund.html. The code for the simulations reported in this paper is also available at the bottom of the same page.

2.3.1 Behavior Oriented Design and POSH Action Selection

Edmund is a behavior based architecture with a separate, specialized action selection mechanism. In this respect it is similar to several mainstream architectures, such as RAPS (Firby, 1987) and PRS (Georgeff and Lansky, 1987). However, the behavior modules in Edmund represent a higher level of abstraction than the simple control primitives they have come to represent in many hybrid architectures. A behavior in Edmund is a group of related functions for sensing and acting upon the world, and the state they need in order to make intelligent discriminations. For example, an agent may have a behavior for navigating through a world, which is dependent on a stored set of experiences pertaining to the location and accessibility of various landmarks or salient items. Or, it might have a greeting behavior which selects from a variety of possible expressions, which in turn reflect the agent's mood or familiarity.

The addition of variable state makes the behaviors in Edmund less like those in standard behavior-based systems, which were highly influenced by the edicts against variable state due to Brooks (1991). On the other hand, Edmund behaviors are actually more like the semi-autonomous, perception-rich behaviors Brooks originally described than are those reduced to being primitive function calls. Edmund behaviors and action-selection scripts can be implemented using a modified version of object oriented design (e.g. Coad et al., 1997) called behavior oriented design, or BOD (Bryson, 2000a).

The behaviors of an Edmund agent are responsible for monitoring the parts of the environment that are salient to them, for remembering relevant experiences (such as the agent's own current motivation or previous social encounters,) and for controlling expression of their associated actions. However, a behavior does not on its own determine *when* an action is expressed; that is left to the action selection mechanism.

POSH action selection is based on reactive plans, which are composed of five possible element types. The most fundamental level of control are *primitive actions* and *sense predicates*. Both of these are simply interfaces to the behaviors which control their expression. As is standard for behavior-based control, the primitive "actions" are themselves fully guided by their own, specialized perception. The sense primitives are not particularly different, except that they provide information, and have no side effects on the real world. Sense primitives are used for informing the decisions of the action selection mechanism.

The most basic plan structure is the *action pattern*, which executes a set sequence of behavior. Action patterns significantly reduce the combinatorics of action selection, but they are not particularly reactive so tend to be fairly short. A more flexible structure is provided by the *competence*. A competence is a prioritized collection of plan elements, the behavior of which tends to converge on a particular goal. A competence is a single tier of a conventional reactive plan, somewhat like a triangle table or teleo-reactive plan (Nilsson, 1994). A competence consists of a set of elements which may be either action patterns or other competences. The elements of a competence are not only prioritized, but may have triggers. The highest priority element that triggers successfully is allowed to execute. An example of how a simple plan composed of one competence can operate to solve problems under a variety of conditions is provided in Appendix A.

Generally, the very highest priority element of a competence is a goal trigger, which has no action, but rather recognizes whether the intended task has been completed. When a goal trigger fires it terminates the competence. Competencies also terminate if none of the elements can fire.

The root of a POSH plan is a special persistent form of competence which allows for the parallel processing of its elements. This is called a *drive collection*, and its elements are analogous to individual drives in ethology. In real-time versions of Edmund, the execution of a drive collection employs course-grained, pseudo-parallel, best-attempt scheduling for shifting attention between its various elements. The drive collection retain the triggers and prioritization of elements that competences have. Resources can when necessary be monopolized by higher level drives, usually by arresting the attention of the drive collection.

2.3.2 Reactive Mechanisms in Edmund

POSH action selection exploits the advantages of hierarchical and sequential control for combating the combinatorial complexity of the action selection problem. At the same time, *posh* action selection is intended to provide adequate reactivity for operating in a dynamic environment.

Reactivity under Edmund comes from several sources. It is grounded in the parallel and semi-autonomous behaviors, which maintain their own perceptual state largely independently of action selection. It is then filtered through POSH control cycle. At every program cycle, attention begins at the parallel roots of the posh hierarchy, the drive collection. Attention then shifts to the highest priority drive which has been triggered and is not habituated on this cycle. Each drive element typically has a plan element in some state of execution. For example, it may be part way through an action pattern, or it might be attending to a particular competence. Attention is focussed on that plan element until one primitive action has been operated, then the program cycles again. This is the source of the course-grained parallelism: no primitive action should in itself take a significant time to execute and return. Long, continuous actions are controlled from the behaviors themselves; in their case the primitive actions usually serve as signals to begin or persist until the next anticipated signal time.

The competences themselves serve as a source of reactivity, since they are sufficiently flexible to handle failed operations and unexpected changes in the environment. They are essentially basic reactive plans, similar to RAPS (Firby, 1987) and teleo-reactive programs (Nilsson, 1994). Further, although competences are hierarchical in that they can contain action patterns and other competences as elements, no stack is allowed to grow. When a competence is activated, it has received activation from a particular drive element. When it then in turn passes attention to one of its elements, it replaces itself with that element in the drive's current attention. This is the slip-stack mechanism, the 'S' in POSH.

Each drive element is a hierarchy root. Each drive element keeps track of this root as well as the element to which it is currently attending. When a competence or action-pattern terminate, then the drive that was attending to it switches its attention back to its root. This way, every decision that was made in activating that element is revisited before returning attention to the terminating element's parent. If the parent is again selected, it will then revisit the decisions that activated the element in the first place. If the element has failed for some reason the parent will retry it, unless a specific mechanism of habituation or episodic memory blocks that repeated strategy. If the element has succeeded and the environment has not otherwise changed, then presumably a higher-priority plan element will be able to trigger, and the plan will progress.

Besides providing reactivity, the slip-stack approach supports chains of arbitrary length, including cycles, in a control structure. An example where a cycle might be useful is in controlling the alternate paces of two-legged walking until a destination is reached or some other goal interferes. Another example is natural language production.

2.3.3 Summary

The following aspects of Edmund facilitate its reactive nature:

- Edmund is a behavior-based architecture. Each behavior maintains its own perceptual state concurrently.

- Each element of the reactive plans that make up Edmund's POSH action selection is prompted to perform by its own perceptual requirements. There is no error passing or assumption of success for an operation.

- The top level of the POSH action selection hierarchy, the drive collection, executes its elements in parallel. This allows for the constant monitoring of important sense modalities or internal states and the possibility of interruption for urgent or opportunistic conditions.

- No stack is maintained as control traverses through a hierarchy of competencies.

POSH action selection thus has the potential for expressing fairly dynamic behavior, but it does not have the complete opportunism or full democracy of Maes' and Tyrrell's architectures. Any given hierarchy rooted under the drive collection may branch an arbitrary number of times, and only one branch will be followed at any particular time. A POSH reactive plan allows for the expression of both sequences and prioritization. It allows for the utilization of selective attention to ensure particular tasks are completed fluently.

Because Tyrrell's simulated environment provides nearly all of the memory, action and sensory primitives required for its task, the experiments below only make use of POSH action selection. Nearly all of the primitives are rooted directly in the behavioral competences provided by the SE. The single additional behavior facilitated the combination of perceptual inputs in selecting directions to run in when pursuing mates or being pursued by predators.

3. Experiments

3.1 Procedure

The code for controlling an agent using the Edmund architecture within the SE was written over a three week period. Total programming time was approximately six days. Considerable time had to be devoted to evaluation; the highly dynamic nature of the SE led to very volatile results, so 6000 simulations (representing no more than 10 days of life each) had to be run for statistically significant results.

The first POSH plan was the most simple one that could generate a genetic fitness measure – one that only pursued mating. Thus the root drive "life" had a single competence "mate", which consisted of:

life (D)	mate (C)	inseminate (courted_mate_here t)	copulate
		court (mate_here t)	strut
		pursue	pick_dir_mate go

See Appendix A for an explanation of the notation.

This simple program performed fairly well. Though on average it lived under half a day, it had a genetic fitness of slightly more than 2.0, immediately outperforming Tyrrell's best implementation of Maes (1991). Other obvious behaviors were added gradually, testing the new additions and ordering over approximately 800 runs of the simulator each. Simplicity was favored unless the program improved very significantly. After this phase of development the components of life in order of priority were as below. Notice that here the label for the drive has been omitted for clarity, but the drive is still represented by the leftmost line. The label *X::Y* indicates drive elements which habituate. Homing in the evening and looking for predators take priority intermittently, but habituate for at least 5 cycles. This allows other, lower priority behaviors to express themselves immediately after they have fired.

flee (C) (sniff_predator t)	freeze (see_predator t) (covered t)	hold_still
	run_away (see_predator t)	pick_safe_dir go_fast
	look	observe_predator
mate (C) (sniff_mate t)	*as above*	
home *1::5* (late t) (at_home nil)	pick_dir_home go	
check *1::5*	look_around	
exploit (C) (day_time t)	use_resource (needed_res_avail t)	exploit_resource
	leave	pick_dir go
sleep_at_home (at_home t) (day_time nil)	sleep	

At this point, the genetic fitness of the Edmund animal was approximately 65% of the ERP's. The next phase of development was to examine the animal's performance step-by-step in the SE graphical user interface. This resulted in the following observations and changes:

- The animal was often killed by feline predators while freezing. Fixed by adding a precondition preventing the freezing action pattern in the presence of felines.
- The animal seemed too concerned with staying in shelter and failed to explore much unless chasing mates. Altered by reducing the desire for shelter in pick_dir.
- Less concerned with shelter, the animal now sometimes explored so far afield it lost bearing and could no longer find its den at nightfall. Fixed with the addition of the following action pattern:

triangulate (getting_lost t)	pick_dir_home go

  This AP was prioritized just above homing, with no habituation.
- The animal often had unforced accidents with irrelevant animals and cliffs, particularly while it was pursuing mates. Fixed by altering pick_mate_dir to take these into account.

Another class of problems could not be fixed so programmatically, because they dealt with tradeoffs. For example, the animal's erratic perception frequently leads it to hallucinate a predator nearby. Too high of priority to perceived predators disrupts behavior coherence sufficiently to interfere with mating, while too little attention to predators reduces mating opportunities by reducing the expected lifespan. This class of problem was addressed by running a large number of simulations with randomly selected values for six critical thresholds: four for detecting predators (observing or fleeing each of the two types), one for seeking shelter and one for avoiding dangerous places.

Linear regression on this data was uninformative, because the variables relationships to the animals success were neither linear nor independent. Instead, we selected the variable sets for the top few performing individuals out of 4400 trials, and tested each over 800 additional runs. The best set improved the animal's mean performance by 25 percent.

At this point Edmund's animal bested the ERP's in all the test worlds except for world 3. Examining the simulator code showed that world 3's change primarily differed by substantially reducing the energy consumption for certain activities (such as mating and running) as well as significant changing the conspicuousness of some behaviors, both positive and negative. This led to the postulate that Edmund's mouse was wasting too much time or conspicuousness on grazing. Consequently the **exploit** competence and its elements were divided into their constituent parts, eliminating the one case of "compromise candidate" determination that had made up Edmund's solution. Also, flight from predators was changed to be the same as normal cautious exploration, only faster.

3.2 Results

Edmund's mouse performed better (that is, mated more), but not significantly so (critical ratio (F) = 1.36) in Tyrrell's worlds. However, it was very significantly better than the ERP in every world except for variant world 3, and also than any other architecture reported in (Tyrrell, 1993). The final results (over 6600 trials) were:

World	Edmund	ERP	F
Standard	9.12 (0.19)	8.09 (0.17)	3.95**
Var. 1	4.02 (0.09)	3.61 (0.09)	3.1**
Var. 2	9.67 (0.2)	8.16 (0.16)	5.73**
Var. 3	11.23 (0.23)	13.38 (0.23)	−6.58**

where the parenthetical numbers indicate standard error — the standard deviation of the mean scores. A negative sign is used with an F value to indicate the significance is in favor of the ERP.

We also tested both architectures in an additional set of worlds which were identical to Tyrrell's except that food was significantly scarcer. The range of possible initial food supplies was modified as follows:

Food	Original	Sparse
fruit	[50, 81]	[5, 35]
cereal	[45, 26]	[2, 27]
prey	[25, 36]	[2, 15]

which resulted in the following performances:

World	Edmund	ERP	F
Standard	8.17 (0.19)	4.77 (0.12)	15.01**
Var. 1	3.56 (0.09)	2.46 (0.06)	9.59**
Var. 2	10.79 (0.18)	4.56 (0.12)	27.6**
Var. 3	10.74 (0.24)	12.53 (0.23)	−5.47**

As can be seen, Edmund also coped substantially better than the ERP with this more difficult situation, though again with the exception of the third variant world, to which Tyrrell's implementation of the ERP seemed particularly well adapted. In this world, however, the overall performance of the Edmund animal was significantly better than the ERP's.

Another important result pertaining to this paper's thesis is the relative complexity of the two systems. Th Edmund agent required 20 action primitives and 22 dedicated sensing primitives. It also had 7 competencies and 23 action sequences defined in its final program script. The ERP agent had 61 sensing nodes and 278 other nodes, of which 216 were intermediate points in its hierarchy. Edmund had 26 thresholds embedded in its primitives, of which 6 were not one of four standard values (0.2, 0.4, 0.6 or 0.8). The ERP had 264 weights, of which 189 were either 1.0 or -1.0. However, the other 75 weights took 37 separate values. In summary, the control of the Edmund agent was approximately an order of magnitude more simple from a design standpoint than that of the ERP agent.

4. Analysis and Implications

Although there is an acknowledged advantage to following the original research and thus having a target to beat, both the relative lack of complexity of the POSH action selection and the significantly superior results are strong evidence that managing complexity through selective attention can be more important than having a fully reactive architecture.

This result is supported by the fact that natural evolution has also selected attention focusing strategies. Animals control not only for what information is worth attending to, but when. Ethology has long had examples such as the digger wasp, which treats caterpillars as food only when they are engaged in appetitive behavior (Lorenz, 1981). Rats also will not engage in any appetitive behavior (such as foraging or eating) without the appropriate triggering of their limbic system (Carlson, 1994). Recent work on decoding hippocampal firing patterns indicates that cells can have different roles and participate in different ensembles depending on the particular behavioral context the rat believes itself to be in (Wiener, 1996). The human brain is extremely context sensitive. Even the retina has more neurons feeding into it then taking information out — even at the very earliest levels of sensing, expectations grounded in current context significantly affects what is perceived.

The work described in this paper addresses the criticisms leveled by researchers such as Maes, Brooks, Goldfield, Hendriks-Jansen and van Gelder at not only artificial intelligence, but also psychology and philosophy. These researchers have claimed that the use of hierarchy for control is intractable for agents needing to survive in a dynamic world. In so doing, they have neglected significant evidence for structured control in observed natural behavior (e.g. Tinbergen, 1951; Dawkins, 1976), through reasoned argument (e.g. McGonigle and Chalmers, 1998), and in the experience artificial intelligence (e.g. Bonasso et al., 1997; Kortenkamp et al., 1998; Bryson, in press) in an over-reaction against the less realistic AI systems that preceded their work. While there can be no doubt that animals have dedicated parallel mechanisms for monitoring the occurrence of significant, plan-altering events, there can also be no question of the importance of controlling the combinatorial complexity of action selection, nor of the critical nature of the design process. Whether the control for an agent is evolved, learned, designed by hand, or developed through some combination of these, the agent's architecture must provide the right representation of control in order to make that development process likely to succeed.

Although considerable space has been dedicated to describing the exact workings of the Edmund architecture, we do not believe that the findings of this paper are unique to the particular architectures examined. Tyrrell himself examined five other architectures, both hierarchical and parallel. He settled on what he believed to be the optimal action-selection scheme on the basis of maximum information with appropriate bias. We believe that the reason Edmund out-performed the hierarchical action-selection mechanisms previously tested in the SE is because of its combination of hierarchical and reactive elements. We have been working to demonstrate that the principles of POSH action selection can be applied under a number of other architectures, including Ymir (Thórisson, 1999), PRS (Georgeff and Lansky, 1987) and JAM (Huber, 1999). These architectures also provide for both hierarchy-based action selection and for interrupting and redirecting control attention. Other architectures also have these features, such as the most recent version of Sloman's architecture (Sloman, 2000), which includes high-level interrupts from "alarms", and ATLANTIS (Gat, 1991). We chose Edmund and the ERP as *representatives* of their respective approaches. We believe both approaches to be particularly well represented because their agents in the SE were designed by the same people who designed the architectures.

We have discussed at length elsewhere the validity of testing action-selection mechanisms in a simulated environment rather than on a robot (see Bryson, 2000b, , Section 4.5). Although purpose-built simulations can include biases which overlook significant issues, the same can be said of many robot experiments. In this case, the simulation has been well established in the literature, and has been previously demonstrated with a large variety of action-selection mechanisms. Further, using this simulation guaranteed that we had not in any way misrepresented the ERP, since we used its code directly as provided by the original author. We were also able to confirm the original reported experimental results, and to perform the large number of trials necessary to establish significant results in a task so thoroughly complex that it creates a very high variance in individual outcomes. This is not to say that we deprecate the use of robots as experimental platforms: the Edmund architecture and accompanying design methodology was significantly refined and improved when moved from a simulated blocks world to a real mobile robot. However, we do believe that robots are not necessary for doing valid action-selection AI research. Neither are simple robot experiments sufficient; we have yet to see a robot experiment that supports the number and diversity of competing goals present in the SE. The nearest contender is probably RoboCup (Kitano et al., 1997).

5. Conclusion

This paper has presented work indicating that even (perhaps especially) in an extremely dynamic and dangerous environment, an agent may be better off ignoring some of the information available to it. This is because finding the correct design, whether by hand-coding, learning or natural evolution, is the key bottleneck to developing intelligence. Reducing information reduces the complexity of the task. This point has been demonstrated by directly comparing two representative architectures in an experimental setting: a fully informed architecture, Tyrrell's Extended Rosenblatt and Payton architecture, which had previously been evaluated against several of the other leading fully reactive architectures; and an architecture utilizing selective attention, called Edmund, which was shown to have significantly better performance and to allow for significantly simpler design.

Acknowledgments

I would like to thank Toby Tyrrell, The University of Edinburgh, and the CMU AI Archives for making the SE freely and publicly available. I would also like to thank Will Lowe for his technical assistance with the statistics packages and his proofreading. This research was conducted in the Laboratory for Cognitive Neuroscience and Intelligent Systems of the University of Edinburgh.

References

Agre, P. E. and Chapman, D. (1990). What are plans for? In Maes, P., (Ed.), *Designing Autonomous*

Agents: Theory and Practice from Biology to Engineering and Back, pages 17–34. MIT Press, Cambridge, Massachusetts.

Blumberg, B. M. (1996). *Old Tricks, New Dogs: Ethology and Interactive Creatures.* PhD thesis, MIT. Media Laboratory, Learning and Common Sense Section.

Bonasso, R. P., Firby, R. J., Gat, E., Kortenkamp, D., Miller, D. P., and Slack, M. G. (1997). Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental & Theoretical Artificial Intelligence*, 9(2/3):237–256.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2:14–23.

Brooks, R. A. (1991). Intelligence without reason. In *Proceedings of the 1991 International Joint Conference on Artificial Intelligence*, pages 569–595.

Bryson, J. (2000a). Making modularity work: Combining memory systems and intelligent processes in a dialog agent. In Sloman, A., (Ed.), *AISB'00 Symposium on Designing a Functioning Mind.*

Bryson, J. (2000b). The study of sequential and hierarchical organisation of behaviour via artificial mechanisms of action selection. MPhil Thesis, University of Edinburgh.

Bryson, J. (in press). Cross-paradigm analysis of autonomous agent architecture. *Journal of Experimental and Theoretical Artificial Intelligence.*

Bryson, J. and McGonigle, B. (1998). Agent architecture as object oriented design. In Singh, M. P., Rao, A. S., and Wooldridge, M. J., (Eds.), *The Fourth International Workshop on Agent Theories, Architectures, and Languages (ATAL97).* Springer-Verlag.

Carlson, N. R. (1994). *Physiology of Behavior.* Allyn and Bacon, Boston, 5 edition.

Chapman, D. (1987). Planning for conjunctive goals. *Artificial Intelligence*, 32:333–378.

Coad, P., North, D., and Mayfield, M. (1997). *Object Models: Strategies, Patterns and Applications.* Prentice Hall, 2nd edition.

Dawkins, R. (1976). Hierarchical organisation: A candidate principle for ethology. In Bateson, P. P. G. and Hinde, R. A., (Eds.), *Growing Points in Ethology*, pages 7–54. Cambridge University Press, Cambridge.

Firby, J. (1987). An investigation into reactive planning in complex domains. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 202–207.

Gat, E. (1991). *Reliable Goal-Directed Reactive Control of Autonomous Mobile Robots.* PhD thesis, Virginia Polytechnic Institute and State University.

van Gelder, T. (1998). The dynamical hypothesis in cognitive science. *Behavioral and Brain Sciences*, 21(5):616–665.

Georgeff, M. P. and Lansky, A. L. (1987). Reactive reasoning and planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, pages 677–682, Seattle, WA.

Goldfield, E. C. (1995). *Emergent Forms: Origins and Early Development of Human Action and Perception.* Oxford University Press.

Hendriks-Jansen, H. (1996). *Catching Ourselves in the Act: Situated Activity, Interactive Emergence, Evolution, and Human Thought.* MIT Press, Cambridge, MA.

Horswill, I. D. (1995). Visual routines and visual search. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal.

Huber, M. J. (1999). JAM: A BDI-theoretic mobile agent architecture. In *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 236–243, Seattle.

Hull, C. (1943). *Principles of Behaviour: an Introduction to Behaviour Theory.* D. Appleton-Century Company.

Humphrys, M. (1996). Action selection methods using reinforcement learning. In Maes, P., Matarić, M. J., Meyer, J.-A., Pollack, J., and Wilson, S. W., (Eds.), *From Animals to Animats 4 (SAB96)*, Cambridge, MA. MIT Press.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., and Osawa, E. (1997). RoboCup: The robot world cup initiative. In *Proceedings of The First International Conference on Autonomous Agents.* The ACM Press.

Kortenkamp, D., Bonasso, R. P., and Murphy, R., (Eds.) (1998). *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems.* MIT Press, Cambridge, MA.

Lorenz, K. (1950). The comparative method in studying innate behaviour patterns. In *Symposia of the Society for Experimental Biology*, volume 4, pages 221–268.

Lorenz, K. (1981). *Foundations of Ethology.* Springer-Verlag, 2 edition.

Maes, P. (1991). A bottom-up mechanism for behavior selection in an artificial creature. In Meyer, J.-A. and Wilson, S., (Eds.), *From Animals to Animats*, pages 478–485, Cambridge, MA. MIT Press.

McGonigle, B. and Chalmers, M. (1998). Rationality as optimised cognitive self regulation. In Oaksford, M. and Chater, N., (Eds.), *Rational Models of Cognition*. Oxford University Press.

Newell, A. and Simon, H. A. (1981). Computer science as empirical inquiry: Symbols and search. In Haugeland, J., (Ed.), *Mind Design*, chapter 1, pages 35–66. MIT Press, Cambridge, MA.

Nilsson, N. (1984). Shakey the robot. Technical note 323, SRI International, Menlo Park, California.

Nilsson, N. (1994). Teleo-reactive programs for agent control. *Journal of Artificial Intelligence Research*, 1:139–158.

Rosenblatt, K. and Payton, D. (1989). A fine-grained alternative to the subsumption architecture for mobile robot control. In *Proceedings of the IEEE/INNS International Joint Conference on Neural Networks*.

Rosenschein, S. J. and Kaelbling, L. P. (1995). A situated view of representation and control. *Artificial Intelligence*, 73.

Sloman, A. (2000). Models of models of mind. In Sloman, A., (Ed.), *AISB'00 Symposium on Designing a Functioning Mind*.

Thórisson, K. R. (1999). A mind model for multimodal communicative creatures & humanoids. *International Journal of Applied Artificial Intelligence*, 13(4/5):519–538.

Tinbergen, N. (1951). *The Study of Instinct*. Clarendon Press, Oxford.

Tyrrell, T. (1993). *Computational Mechanisms for Action Selection*. PhD thesis, University of Edinburgh. Centre for Cognitive Science.

Ullman, S. (1984). Visual routines. *Cognition*, 18:97–159.

Whitehead, S. D. (1992). Reinforcement learning for the adaptive control of perception and action. Technical Report 406, University of Rochester Computer Science, Rochester, NY.

Wiener, S. I. (1996). Spatial, behavioral and sensory correlates of hippocampal CA1 complex spike cell activity: Implications for information processing functions. *Progress in Neurobiology*, 49(4):335.

Appendix A

The following is a detailed example of the working of a POSH competence, taken from (Bryson, 2000a). Consider an example in blocks world. Assume that the world consists of stacks of colored blocks, and that the goal of the agent is to hold a blue block. A possible plan would be[2]:

grasp_blue (C)	*goal* (holding block) (block blue)	
	drop (holding block)	$\boxed{\text{drop\_held\_object}}$
	grasp 3::0 (fixated_on blue)	$\boxed{\text{move\_attn\_to\_stack\_top} \\ \text{grasp\_attended\_object}}$
	look (blue_in_scene)	$\boxed{\text{fixate\_blue}}$

Each plan element starts with a label — a (C) next to the label indicates a competence, a (D) a collection of drives. Preconditions follow the plan label, and are in parentheses. Action patterns are in boxes, in this simple plan most of them have only one element. For a competence or drive collection, the highest priority element is at the top of the line next to the label.

For this competence, the highest priority element is a special form which recognizes that a goal has been achieved. A competence does not need to have a goal, and drive collections seldom do. A competence will terminate if either the goal is achieved or if no elements can execute. Otherwise, the highest priority element that can run is executed.

Consider the case where the world happens to include a stack with a red block sitting on the blue block. If the agent has not already fixated on the blue block before this competence is activated, then the first operation to be performed would be element **4**. Otherwise, if for example the previously active competence has already fixated on blue, **4** would be skipped. Once a fixation is established, element **3** will trigger. If the grasp is successful, this will be followed by element **2**, otherwise **3** will be repeated. Assuming that the red block is eventually grasped and discarded, the next successful operation of element **3** will result in the blue block being held, at which point element **1** should recognize that the goal has been achieved, and terminate the competence.

Infinite retries on the grasp is prevented through habituation without recovery. The grasp will be tried 3 times. Elements of a drive collection may recover from their habituation. For example, a label 1::5 indicates that after being triggered once, the element will not be available for triggering for another 5 turn cycles. In real-time applications such as robots, drive habituation recovery is scheduled by real times (see Bryson and McGonigle, 1998).

[2] This task is taken from (Whitehead, 1992). The perceptual operations are based on the visual routine theory of Ullman (1984), as implemented by Horswill (1995). Fixate_blue puts visual attention on a blue object (blue is considered to be a "pop-out property".) Move_attn_to_stack_top is actually an abbreviated action which requires two attention markers.