Improved Animal-Like Maintenance of Homeostatic Goals via Flexible Latching

Philipp Rohlfshagen
School of Computer Science, University of Birmingham
Edgbaston, Birmingham B15 2TT, United Kingdom
p.rohlfshagen@cs.bham.ac.uk

Joanna J. Bryson †
Department of Computer Science, University of Bath
Bath BA2 7AY, United Kingdom
jjb@cs.bath.ac.uk

Abstract

Controlling cognitive systems like domestic robots or intelligent assistive environments requires striking an appropriate balance between responsiveness and persistence. Basic goal arbitration is an essential element of low-level action selection for cognitive systems, necessarily preceding even deliberate control in the direction of attention. In

*Research funded by the British Engineering and Physical Sciences Research Council (EPSRC) grants GR/S79299/01 and EP/E058884/1. A preliminary version of this paper appeared in the AAAI Fall Symposium on Biologically Inspired Cognitive Architectures [P. Rohlfshagen, J.J. Bryson, Improved Animal-Like Maintenance of Homeostatic Goals via Flexible Latching, in: Proceedings of BICA-08, 2008]. This version contains results omitted from the workshop paper, as well as further and improved expositions.

†The majority of this work was carried out when PR was in the Department of Computer Science, University of Bath, Bath, BA2 7AY, United Kingdom, and JJB was on sabbatical at The Konrad Lorenz Institute for Evolution and Cognition Research, Adolf Lorenz Gasse 2, A-3422 Altenberg, Austria. JJB is the corresponding author.
natural intelligence, chemically-regulated motivation systems focus an agent’s behavioural attention on one problem at a time. Such simple durative decision state can improve the efficiency of artificial action selection by avoiding dithering, but taken to extremes such systems can be inefficient and produce cognitively-implausible results. This article describes and demonstrates an easy-to-implement, general-purpose latching method that allows for a balance between persistence and flexibility in the presence of interruptions. This appraisal-based system facilitates automatic reassessment of the current focus of attention by existing action-selection mechanisms. We propose a mechanism, flexible latching, and demonstrate that it drastically improves efficiency in handling multiple competing goals at the cost of a surprisingly small amount of extra code (or cognitive) complexity. We briefly discuss implications of these results to understanding natural cognitive systems.

Keywords: Action selection; drives; modularity; cognitive architectures

1 Introduction

The term action selection might seem to imply cognition, but this is merely due to anthropomorphic labelling. If we take cognition to be a process requiring time (probably a form of on-line search; [42]), and action selection to be any mechanism for determining the present course of action [11], then
much of action selection is really non-cognitive. Action choices in animals are limited both by evolution and individual skill learning; for adult animals many actions may be essentially reflexive [5, 7]. Such limiting is necessary if action selection is to be achieved in a timely manner [37, 15, 21]. However, there is no question that animals (including humans) do engage in cognition in some contexts. This article examines one such context: the arbitration between different goals. Even here, basic arbitration must necessarily be automatic. However, functional and efficient behaviour requires that the automated system can in some situations be interrupted and controlled cognitively [39]. Here we present a way to efficiently facilitate this capacity in artificial cognitive systems.

Budgeting time and pursuing multiple conflicting goals is a key aspect of any cognitive system [17, 22]. In the simulation of real-time animal-like intelligence considered in this paper, artificial agents must carry out a set of tasks, essential to their survival, while also interacting with dynamic surroundings, including other agents. Other-agent interactions in particular may include activities that are potentially essential to the species as a whole but not necessarily in the interest of the performing individual’s viability. This characterisation might suggest rather dramatic activities, e.g. fending off attack, but it can also apply to ordinary duties. In some sense, the tasks that the system was originally designed to carry out (e.g. mating in nature, or perhaps tea making for an office robot) are of lower immediate priority than making certain that the system maintains working order, since working
order (e.g. the ability to move and manipulate) is a precondition of any other activity. Nevertheless, it is clear that we require an agent to devote considerable time to the goals that motivated its construction. Such critical but non-urgent goals are common amongst animals, such as maintaining a social network, reproducing or keeping clean. All these behaviours require both time and energy, and it follows that agents possessing more efficient behaviour management should, in general, fare better than other agents with less efficient behaviour selection.

In this article, we demonstrate our goal-arbitration system using a simple artificial life task environment. Our agents must ensure they have the ability to store excess energy in order to pursue auxiliary behaviours. We discovered the need for an improved arbitration mechanism during the course of research on the evolution of primate social structures, so our examples derive from these models. The immediately urgent goals concern feeding, while the ultimately-important goals are social networking and exploration. Note that in nature such goals could also be considered survival-oriented, since socialising promotes long-term survival by facilitating group living [17, 25]. However, their payoff is more diffuse — it is seldom knowable when additional goodwill or information gathered may become critical, in contrast to starvation which has clear endogenous indicators. Thus we place essential behaviours at a high priority, but design an action-selection mechanism to ensure they are executed as efficiently as possible.

In this article we present a comparative study of three variants of a simple
action-selection mechanism designed to improve the agent’s capacity for goal arbitration. Our primary motivation is a potential inefficiency that may occur when an agent attempts to acquire a buffer of excess satisfaction before pursuing its next goal. We propose that if an agent is interrupted at any stage during this period, a choice needs to be made concerning whether to continue with the current goal or whether to attend to other, possibly more relevant behaviours. Persistence avoids the inefficiency of *dithering* between multiple goals. Dithering is inefficient because there is typically a significant start-up cost to pursuing new goals before consummatory actions can take place. However, some degree of flexibility avoids the inefficiency of pursuing a goal which is no longer urgent and has locally become excessively costly.

We look to biological motivation systems for inspiration because these have presumably evolved to manage this trade off. However, here we do not attempt a perfect or neurological model nature. Rather, our emphasis in this article is engineering. We present and evaluate a simple control mechanism that achieves the requisite level of flexibility at minimal cost. In fact, two types of costs are kept minimal: both the advance, coding-time costs for the agent’s designers and the real-time, cognitive-processing costs for the agents. We use a basic latching system augmented with the ability to detect potentially relevant interruptions. This threshold-based addition triggers a reevaluation of priorities already present in the agents’ overall action-selection system.
2 Methods

In this section we first describe the particular agent architectures we use to test our new goal arbitration system. Although we use a single system here, it is an example of a common type of action-selection system, and we describe the augmentation in general terms so that it may be applied on other systems as well. We then describe the specific goals to be manipulated in the experiments, and define the metrics of success in terms of these. Next, we describe the various latching mechanisms we have implemented for comparison. Finally, we describe the testing scenarios, including the agents’ operating environment, followed by the presentation and discussion of our results.

2.1 Basic Action Selection

The agents are specified using the behaviour-oriented design (BOD) methodology [12], a system that produces complete, complex agents consisting of (a) modules that specify details of their behaviour and (b) dynamic plans that specify agent-wide, cross-modular priorities. Actions are produced by the modules; action selection (where there is contention) is carried out using the Parallel-rooted, Ordered Slip-stack Hierarchical (POSH) dynamic plan system [10].

We chose BOD as a fairly simple example of an architectural consensus achieved in the late 1990s for real-time, situated systems: That AI is best
constructed using a combination of modularity, for providing intelligent primitives, and structured hierarchical plans, for encoding priorities [24, 26, 8]. Even mainstream cognitive architectures such as Soar and ACT-R can be described in this way [28, 38]. Such approaches have been somewhat neglected in the academic literature in the last decade due to an emphasis on machine learning approaches to action selection. However, in applied human-like AI such as games programming and cognitive robotics, such modular, hand-coded approaches are still very much the norm [23, 31].

The details of the structured action-selection system are unimportant to the mechanism presented in this paper. All that is assumed is

- some mechanism for storing temporary values of long-term state (e.g. learning),
- some mechanism of expressing a variety of goals and their associated actions, and
- the notion of a trigger or precondition as part of the mechanism for choosing between goals and actions.

A single POSH plan was used to specify the priorities of all the agents tested here. That is, all the agents have the same priorities and therefore the same dynamic plan, though of course their expressed behaviour will vary due to their environment and their previous experience. What differs between conditions in the experiments described below are only the action-selection mechanisms and the testing environments.
The plan, shown in Figure 1, assumes four basic behaviours (drives): \(B_1\) to \(B_4\). In POSH, the top level of a plan hierarchy (the drive collection) is checked on every cycle of the controller. Control is passed to the highest-priority drive element whose trigger (line-labels in Figure 1) is true. All but behaviour \(B_4\) further contain a sub-plan, in POSH called a competence. Competences also contain elements each with their own trigger, but these are plans for the purpose of pursuing a single goal, and as such require less sophisticated scheduling than the drive collection. Competences maintain decision memory and control behaviour until they either terminate, pass control to a child competence of their own, or the main drive collection takes control back for a higher-priority problem. Their execution is similar to teleo-reactive plans [32] or indeed to the generalised plans created by STRIPS [18].

The first two behaviours, which are of the highest (and equal) priority, fulfil consumption-related needs, such as eating or drinking, the neglect of which would cause the agent to die. Behaviours \(B_3\) and \(B_4\) are of lower priority and are only considered for potential execution if \(B_1\) and \(B_2\) are not triggered. It should be noted that these behaviours are of lower priority simply because behaviours \(B_1\) and \(B_2\) are essential to the agent’s immediate survival. This does not imply, however, that lower-priority behaviours are not important, they could be critical to the agent’s mission. Since our experimental environment represents primate social behaviour, these behaviours in fact relate to increasing the probability of longer life. As such, behaviour \(B_3\) represents social networking through grooming, which requires two agents to
Figure 1: The POSH plan that determines priorities for the agents: the drive collection (SDC) is called at every time step and its elements checked in order: \( \{ B_1 = \text{eat}, B_2 = \text{drink}\}, \{ B_3 = \text{groom}\}, \{ B_4 = \text{explore}\} \). The highest-priority element whose trigger is true is executed. Equal priority elements (i.e., \( B_1 \) and \( B_2 \)) are checked in random order.

interact with one another. The final behaviour (\( B_4 \)) is exploration, possibly to find new food sources. In a POSH plan, the lowest-priority goal serves as a default behaviour and should always be triggerable. Thus if an agent with this plan is efficiently arbitrating between goals, it should be able to spend most of its time exploring new space.
2.2 Metrics of Efficient Behaviour

The primary focus of our investigations then is on behaviours $B_3$ and $B_4$. Lower priority behaviours may only be executed if all higher priority behaviours are managed efficiently and for artificial agents, the ‘lower’ behaviours are typically the ones that define and justify the agent’s mission. Despite their significance these behaviours are necessarily of lower priority than those that facilitate the survival of the agent so it can perform these tasks. It is therefore paramount that these higher-level behaviours are managed efficiently enough to allow agents to pursue other behaviours as well.

Each behaviour is composed of numerous elements, some of which may be classified as secondary actions. In the case of feeding, the secondary actions would be ‘locating food source’ and ‘move towards food source’. The primary action would correspond to ‘eat’. For all behaviours, executing the primary action with a high frequency relative to the secondary actions determines the degree of efficiency with which the behaviour is executed. Dithering, the rapid switching between goals, results in secondary actions being performed excessively in proportion to primary ones. In our example, each behaviour $B_i$ has one such primary action which will be denoted as $B_i^{α}$. The frequency at which primary actions are executed determines the degree to which all behaviours may be executed and thus defines the metric of success at the centre of our investigation.
2.3 Agents and State

Each behaviour $B_i$ is associated a single-valued internal state $E_i$. Here, for the sake of clarity and without loss of generality, we use the concept of energy to denote the internal state of the agent: each behaviour $B_i$ has a current level of energy $E_i$. The agents live in a toroidal, discrete-time world with dimensions of $600 \times 600$ pixels. Time is considered to be discrete and at every time step, all agents in the environment are updated simultaneously.

In particular, at every time-step, all energy states $E_i$ are decreased by $e_i^-$. If a given behaviour is vital to the agent’s survival, death is imminent once $E_i \leq 0$. For each behaviour, we define a threshold $\delta_i$ such that $B_i$ is triggered once $E_i < \delta_i$. Once $B_i$ is triggered, the agent will execute the actions associated with that particular behaviour. The behaviours $B_1$ and $B_2$ in our example correspond to sustenance activities (eating or drinking): The agent first locates an energy source, moves towards the energy source (at a speed of 2 pixels/time step) and consumes the source once in close proximity. This consumption raises the agent’s internal state by $e_i^+$. Clearly we must ensure that $e_i^+ \gg e_i^-$, $\forall i$ as otherwise an agent would never be able to satisfy a need (and in the case of essential behaviours, the agent would eventually die). Here we have chosen the same values for all behaviours: $e^+ = 1.1$ and $e^- = 0.1$ and hence drop the behaviour-dependent subscript $i$ from here on. Since we are interested in the execution of lower-priority behaviours, an individual choice of energy gain/loss across the different behaviours would require the adjustment of the individual thresholds (which are tightly related
to the net energy gain), unnecessarily complicating the model. Overall, this
gives a net energy gain of $e^\pm = 1$ for any primary action.

Lower-priority behaviours (i.e. $B_3$ and $B_4$) may only be executed if $B_1$
and $B_2$ are satisfied. What it means for a behaviour to be ‘satisfied’ depends
upon the implementation of the agents’ action selection — the basis of this
article which we describe next.

### 2.4 Conditions

We use three different action selection mechanisms and evaluate their impact
on the efficiency of the agent: unlatched, strict latch and flexible latch.

#### 2.4.1 Unlatched

As mentioned in the previous section, a behaviour $B_i$ is triggered if $E_i < \delta_i$.
In the basic unlatched model, the drive terminates as soon as $E_i \geq \delta_i$ and the
time spent at the energy source is expected to be relatively short (although
this depends strictly on $\delta_i - E_i$ which may vary depending on the number of
equal-priority behaviours). Furthermore, no excess energy is stored and the
behaviour is triggered again very shortly after it is satisfied\(^1\). When there
are multiple such behaviours, the agent will continue to oscillate between
them (dithering). Even if there is only a single top-priority behaviour, the
agent will spend its entire time in close proximity to the energy source as the

\(^1\)The theoretical maximum possible excess energy in this case given the values of $e^+$
and $e^-$ is 0.9 which will last for 9 time steps.
acquired energy is always insufficient to pursue anything else.

2.4.2 Strict latch

In the latched models, the agent only terminates the drive once \( E_i \geq \phi_i \) where \( \phi_i \geq \delta_i \). Now the agent has an energy reserve of \((\phi_i - \delta_i)/e\) time steps before the behaviour is triggered again. If all high-priority drives are latched in this way and the latch is sufficiently large (see next section), the agent is able to eventually follow lower-priority drives. This form of latching is very inefficient, however, if the agent inhabits a world where unexpected interruptions may occur. If an agent is almost finished with one activity but gets interrupted, the agent will continue to pursue this activity independent of other, lower-or-same priority needs. For example, an agent that is grooming and whose partner has left, might pursue another partner for five minutes when only another five seconds of grooming would have satiated it. This is true even if \( E_i = \phi_i - \epsilon \) where \( \epsilon \ll \phi_i - \delta_i \) and hence this form of latching is referred to as strict.

2.4.3 Flexible latch

If the agent is able to detect interruptions, the interruption could trigger a decision that determines its subsequent activities. Such a decision might be conscious, but here we simply relax the latching by using yet another threshold, \( \psi_i \), that is situated in-between the previously two established ones, \( \delta_i \leq \psi_i \leq \phi_i \). This gives rise to two different scenarios. If the interruption
occurs when:

1. $\delta_i < E_i < \psi_i$, the drive remains ‘unsatisfied’
2. $\psi_i < E_i < \phi_i$, then the drive is considered ‘satisfied’

Note that for $\delta_i < E_i < \phi_i$ the status of any latch is path or history dependent — if $E_i$ was more recently below $\delta$ the drive is now unsatisfied, if it was more recently satiated (about $\phi$) than it is not. What is new for the flexible latch is that if an interruption occurs in the third scenario, where $E_i$ had been below $\delta$ but has now been raised above $\psi_i$, this path dependency is dismissed.

2.5 Threshold Selection

The previous section has discussed different thresholds that require initialisation and the choice of parameters is crucial to the outcome of the simulation.

First, it should be noted that the flexible latch is simply a generalisation of the strict latch, which in turn is a generalisation of the unlatched technique:

- Flexible latch \( \delta \leq \psi \leq \phi \)
- Strict latch \( \delta \leq \psi = \phi \)
- Unlatched \( \delta = \psi = \phi \)

In this investigation, we have two primary points of interest, which are closely related: Survival and efficiency. The survival of the agent crucially depends on the choice of $\delta$. Efficiency, on the other hand, refers to the agent’s ability
to pursue all its behaviours, not just high-priority ones, and depends on the
choice of $\phi$ and $\psi$. In order for an agent to survive, any vital behaviour must
be triggered such that the agent has enough energy to approach the energy
source (locating an energy source can be done in a single time-step and is
subsequently excluded from the following discussion):

$$\delta_i \geq E'_i$$  \hspace{1cm} (1)

where $E'_i$ is the energy required to reach the source: $(d_{\text{max}}/d_{\text{mov}}) \times e^-$, where
$d_{\text{mov}}$ is the distance an agent can move in a single time step and $d_{\text{max}}$ is the
maximum possible distance an agent can travel$^2$. If there are $n$ equally vital
behaviours, $\delta_i$ has to be adjusted accordingly:

$$\delta_i \geq \sum_{j=1}^{n-1} (E'_{j} + E'^{c}_{j}) + E'_i$$  \hspace{1cm} (2)

where $E'^{c}_{i}$ is the energy required to raise the energy level to the appropriate
level:

$$E'^{c}_{i} = \frac{\delta_i - E_i}{e^\pm}$$  \hspace{1cm} (3)

$^2$The theoretical maximum in this case is simply $\sqrt{(\text{width}/2)^2 + (\text{height}/2)^2} \approx 424$
and it would take the agent a maximum of $424/2=212$ time steps to reach the target,
consuming $212 \times 0.1 = 21.2$ units of energy.
The value of $\phi$, on the other hand, has to be set such that enough energy is stored to pursue all vital needs:

$$\phi_i \geq \delta_i + \sum_{j=1}^{n} (E_{ij}^r + E_{ij}^c)$$

Any excess energy is subsequently devoted to the other, lower-priority behaviours. This choice of $\phi_i$ necessarily affects $E_c$ as now more time is spent at the energy source (a difference of $\phi_i - \delta_i$). Interruptions drastically alter $E_c$ and the energy required to satisfy a latched behaviour given $m$ interruptions is simply:

$$E_{ij}^c = \sum_{j=1}^{m} (E_{ij}^r + E_{ij}^c)$$

At each interruption, the agent should, in theory, decide whether it is worth pursuing the currently executed behaviour (i.e. if there is a positive or negative energy ratio). Usually there is insufficient knowledge available to make an informed decision due of the complexity or indeterminacy of the environment. Consequently, heuristic values must be used. Nature selects for agents with appropriate or at least adequate thresholds; here we test a range of values for $\psi$ to find which is appropriate for our particular simulations.

### 2.6 Experiment and Simulation Details

Our experiments are organised into two sets. The first set uses sim1, a very well defined setup that allows a great degree of control over all aspects in-
Figure 2: The two simulation environments used to test the overall efficiency of the agents: a completely controlled scenario (a) where energy sources are maximum distance apart, all agents are initially grouped at the centre and interruptions are externally induced, and a more realistic scenario (b) where agents and energy sources are placed randomly.

Investigated, particularly the frequency of interruption (see Figure 2(a)). The second set use sim2 (Figure 2(b)), a more realistic simulator where interruptions are caused by the dynamics of the environment itself. For our experiments we consider two types of interrupts. The first type occurs when the source of satisfaction is depleted or otherwise removed (e.g., an agent loses his current grooming partner). The second type of interrupt is caused by higher priority drives that are triggered.

In both simulations, there are 5 identical agents. Furthermore, sim1 positions the energy sources such that they are maximum distance from one another\(^3\). In this simulation, we exactly control the number of interruptions an agent is exposed to throughout the execution of a single behaviour. Once

\(^3\)The simulation is toroidal and agents are able to move, for example, from the far left to the far right in one move.
an agent is interrupted, it is forced to consider an alternative energy source (it is not allowed to remain at the current one). The second simulation is somewhat more realistic and is used to verify the results obtained from the first set of experiments. In sim2, energy sources are scattered randomly across the world. Each energy source has a certain load that depletes as an agent consumes it. Once depleted, the energy source vanishes, but, at the same time, a new energy source appears elsewhere in the world. The load of any energy source has a maximum of 50 units and depletes by 2 units if consumed. All energy sources gain 1 unit per time step.

The experiments are executed over 15 distinct trials. Each trial executes the simulation for 5000 time steps. All internal states are initialised such that $E_i = \delta_i$, thus all behaviours are triggered immediately once the simulation begins. At each time step, the agent may execute a single action. The results are simply the number of times each primary action has been executed, averaged over all agents and trials. In all cases, a two-tailed t-test is used to test for significance with a confidence of 0.995. We chose the same threshold settings across all behaviours and again, we drop the subscripts from here on. Furthermore, we set $\delta = 200$ in all experiments, giving an agent sufficient energy for $200/e^- = 2000$ time steps before $E$ falls to zero after a behaviour has been triggered.
Table 1: Comparing latched and unlatched behaviours. The latches are chosen to be $\phi - \delta \in \{0, 10, 50, 100\}$.

<table>
<thead>
<tr>
<th>action</th>
<th>no latch $\phi = \delta$</th>
<th>latched 10</th>
<th>50</th>
<th>100</th>
<th>significance 0-10</th>
<th>10-50</th>
<th>50-100</th>
</tr>
</thead>
<tbody>
<tr>
<td>$B_1^o$</td>
<td>443</td>
<td>452</td>
<td>478</td>
<td>494</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>$B_2^o$</td>
<td>443</td>
<td>452</td>
<td>479</td>
<td>498</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>$B_3^o$</td>
<td>0</td>
<td>0</td>
<td>454</td>
<td>468</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_4^o$</td>
<td>0</td>
<td>0</td>
<td>1414</td>
<td>2037</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>total</td>
<td>886</td>
<td>903</td>
<td>2824</td>
<td>3498</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3 Results

3.1 Controlled Environment: Sim1

The first experiment compares the unlatched version with the strictly latched one. The results are shown in Table 1. The data confirms that in the unlatched case, dithering prevents the agent from pursuing any of the lower priority behaviours. The latch effectively solves this problem, although only if the latch is sufficiently large. A latch of size 10 does increase the activity of the primary actions for behaviours $B_1$ and $B_2$ but still does not allow for the lower-priority behaviours $B_3$ and $B_4$ to be executed. Once the latch increases sufficiently in size, so does the activity of the lower-priority behaviours. This result is not surprising. Note though that too large a latch might also lead to neglect of lower-priority behaviours, since the highest-level goals might never be satisfied.

The next experiment investigates the efficiency of strict latching once an agent is confronted with interruptions. The data for this experiment
Table 2: The performance of the agents given $\phi - \delta \in \{10, 50, 100\}$ and 1, 3 or 5 interruptions. Significance is checked for $\phi = 100$. Cases without interruptions (0) are taken from the results shown in table 1 (not shown in this table).

<table>
<thead>
<tr>
<th>action</th>
<th>10</th>
<th>50</th>
<th>100</th>
<th>significance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>0-1</td>
</tr>
<tr>
<td>$B_1^\alpha$</td>
<td>458</td>
<td>442</td>
<td>420</td>
<td>478</td>
</tr>
<tr>
<td>$B_2^\alpha$</td>
<td>454</td>
<td>441</td>
<td>429</td>
<td>474</td>
</tr>
<tr>
<td>$B_3^\alpha$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>277</td>
</tr>
<tr>
<td>$B_4^\alpha$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>95</td>
</tr>
<tr>
<td>total</td>
<td>912</td>
<td>882</td>
<td>850</td>
<td>1324</td>
</tr>
</tbody>
</table>

is summarised in Table 2. Even in the case of a single interruption, the
frequency of primary actions executed drops significantly. The right-most
column in the table compares the performance of a latch of size 100 with 0,
1, 3 and 5 interruptions and the differences for the lower-priority actions are
almost always significant.

The final experiment using $sim1$ determines the performance of the flexi-
ble latch using the same settings as in the experiment before. Here, different
values for the intermediate threshold $\psi$ are tested. The value of $\psi$ is denoted
as the percentage of the latch itself. If, for example, $\delta = 100$ and $\phi = 120$, a
value of 25% would indicate that $\psi = 105$. The results are shown in Table 3
and a setting of $\psi = \delta$ seems most successful. However, as shown in Table 4,
the differences are usually not significant. In the absence of significant differ-
ence, the zero setting is still to be preferred as it also allows us to simplify the
action-selection mechanism. We can effectively eliminate $\psi$ altogether but
always reconsider priorities when interrupted. Comparing the flexible latch
Table 3: The performance of the agents with flexible latching. $\psi = \delta + p(\phi - \delta)$ where $p \in \{0, 0.25, 0.5, 0.75\}$, $\delta = 200$, $\phi = 300$ and frequency of interruptions equal to 1, 3 and 5. Significance of results shown in table 4.

<table>
<thead>
<tr>
<th>action</th>
<th>1</th>
<th>3</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0%</td>
<td>25%</td>
<td>50%</td>
</tr>
<tr>
<td>$B^a_1$</td>
<td>499</td>
<td>491</td>
<td>489</td>
</tr>
<tr>
<td>$B^a_2$</td>
<td>492</td>
<td>490</td>
<td>496</td>
</tr>
<tr>
<td>$B^a_3$</td>
<td>481</td>
<td>476</td>
<td>479</td>
</tr>
<tr>
<td>$B^a_4$</td>
<td>1723</td>
<td>1689</td>
<td>1528</td>
</tr>
<tr>
<td>total</td>
<td>3195</td>
<td>3146</td>
<td>2991</td>
</tr>
</tbody>
</table>

Table 4: Significance results for table 3. Increasing $p$ has the most impact on the lowest-priority behaviour. The right-most column compares the strictly and flexibly latched implementation for the different frequencies of interruptions.

$$
\text{Table 3: The performance of the agents with flexible latching. } \psi = \delta + p(\phi - \delta) \text{ where } p \in \{0, 0.25, 0.5, 0.75\}, \delta = 200, \phi = 300 \text{ and frequency of interruptions equal to 1, 3 and 5. Significance of results shown in table 4.}
$$

$$
\text{Table 4: Significance results for table 3. Increasing } p \text{ has the most impact on the lowest-priority behaviour. The right-most column compares the strictly and flexibly latched implementation for the different frequencies of interruptions.}
$$

$$
to the strict latch shows a significant improvement in at least one behaviour’s primary action for any number of interruptions tested (compare Table 2 with Table 3; significance is indicated in the right-most column of Table 4).
$$

$$
Figure 3 shows graphically how the ability to detect interruptions improves the agent’s overall efficiency. The graph plots the number of time steps spent executing the actions of interest given different frequencies of interruption. Furthermore, as a reference value, the unlatched and uninterrupted latched cases are also shown. It is evident that the performance of the strict latch degrades very quickly while the flexible latch substantially
$$

21
Figure 3: A graphical comparison of strict and flexible latching ($\sum_{i=1}^{4} B_i^\alpha$). The top and bottom lines are shown for reference, indicating the latched but uninterrupted and unlatched cases. For uninterrupted latches, the strict and flexible cases are indistinguishable.

reduces the impact of interruptions.

3.1.1 Death Rates

In the previous experiments, efficiency was judged by the capacity to devote time to all behaviours. For these experiments, the value of $\delta$ has been set such that agents would always survive. In nature, such a threshold would evolve in species like primates that invest a great deal in individual survival and life histories. Nevertheless, exceptionally extreme environments or other unusual circumstances may cause a threshold setting to become (temporarily) insufficient.

In the present experiment, we set $\delta$ such that survival in an uncertain environment is no longer guaranteed ($\delta = 40$). We then compare death
rates between strict and flexible latches. The latch is also set at a relatively
low level of \( \phi = 45 \). The results are shown in Table 5. The flexible latch
shows a significantly reduced death rate in all three relevant conditions (as
determined by the number of interruptions). Furthermore, it is interesting
to note that now, even with the smaller latch, the flexible implementation
performs significantly better in almost all cases when compared to the strictly
latched version.

Finally, it is possible to reduce the death rate even further. In another sce-
nario we utilise the agents’ ability to deal with interruptions: Equal-priority
behaviours are allowed to interrupt one another if they reach a critical thresh-
hold \( \psi \). We set \( \psi = 20 \), as per the calculations described in Section 2.5 above.
This critical threshold essentially corresponds to the minimum energy re-
quired to satisfy a single need. The addition of the threshold changes the
death rates from 0, 861, 1143 to 60, 417, 472. Interestingly, the death rate
is actually slightly higher in the first case but noticeable lower in the other two cases. The differences are relatively weakly significant for this $N$, with a confidence of $p < 0.05$ for both the two- and three-interrupt conditions.

### 3.2 Random Environment: Sim2

The previous results showed that in $sim1$, latching is necessary to allow an agent to execute lower-priority behaviours, and that it is best to abort a latched behaviour immediately upon interruption. We now examine these results in a system with a more “natural” setup using $sim2$, where the timing and frequency of interruption depends on the dynamics of the environment itself.

Table 6 compares all three implementation on $sim2$. The overall results are similar to before although there are some striking differences. Now, a latch of size 10 is sufficient to generate at least some frequency of execution for behaviours $B_3$ and $B_4$ whether or not it is flexible and indeed the flexibility makes no significant difference at this size latch. The change is due to the random environment providing more opportunities, which either implementation is able to exploit. Once the size of the latch increases, flexibility creates a noticeable (as well as significant) difference for behaviour $B_4$, but no difference for $B_3$. This indicates $B_3$’s primary action is already executed sufficiently even without the flexibility in the latch — the flexibility in the environment provides sufficient opportunities for it to satiate at the threshold levels we’ve specified. Nevertheless, the massive increase of opportunity for
expressing the exploratory behaviour shows the power of flexible latching.

4 Discussion

We have considered three variants of a simple threshold-based action selection mechanisms. The completely unlatched condition may seem unrealistic, but several well-known reactive architectures have added latching only as an afterthought, handled with rather inelegant exception mechanisms [35, 16]. Others assume latching can be handled by intelligent planning [6, 39]. This, however, requires a high cognitive load and in general, reasoning about time and distant rewards is difficult even for cognitive, symbolic systems [1].

The basic latched approach is inspired by theories of affect and action selection, as well as basic control theory. LeDoux [29] for example promotes the theory that emotions place the brain in a cognitive context appropriate for a particular course of action. Neuroscience tells us that interrupting such emotional responses is a cognitive capacity requiring frontal-lobe inhibition

<table>
<thead>
<tr>
<th>action</th>
<th>unlatched</th>
<th>strict latched</th>
<th>flexible latched</th>
<th>significance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>10 50 100</td>
<td>10 50 100</td>
<td></td>
</tr>
<tr>
<td>$B_1^a$</td>
<td>451</td>
<td>454 470 500</td>
<td>454 466 468</td>
<td>*</td>
</tr>
<tr>
<td>$B_2^a$</td>
<td>452</td>
<td>454 475 490</td>
<td>455 466 469</td>
<td>*</td>
</tr>
<tr>
<td>$B_3^a$</td>
<td>0</td>
<td>178 365 452</td>
<td>154 423 471</td>
<td>*</td>
</tr>
<tr>
<td>$B_4^a$</td>
<td>0</td>
<td>71 264 689</td>
<td>22 704 1289</td>
<td>*</td>
</tr>
<tr>
<td>total</td>
<td>903</td>
<td>1156 1574 2131</td>
<td>1084 2058 2697</td>
<td></td>
</tr>
<tr>
<td>dead</td>
<td>0</td>
<td>0 0 0</td>
<td>0 0 0</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Comparing the unlatched, strictly and flexibly latched implementations in $sim^2$ using latch sizes of $\phi - \delta \in \{10, 50, 100\}$ and $\psi = \phi$. All cases have frequent interruptions (see main text).
of the emotional response [14]. Of course, the frontal-lobe inhibition system must itself be a fairly automatic gating mechanism. But this mechanism provides an opportunity for an alternative plan to become most salient [34].

Our system for determining appropriate thresholds for the flexible latches is also inspired by animal mechanisms through ethology. In particular, Dunbar’s time-budget theory [17, 25] suggests that animal drives have evolved to ensure individuals are likely to spend the appropriate amount of time in behaviours, where appropriate is determined by what is adaptive. Our work here can be seen both as support for this theory and possibly as an elaboration, to the extent that our mechanism helps connect the time budget to the underlying neuroscience others have proposed (e.g. [34].)

In AI in contrast, there have been surprisingly few recent attempts to propose general-purpose architectural features for homeostatic control. Those that exist tend to create detailed biomimetic representations of hormone levels [41, 27]. Gadanho [20] has a similar perspective to our work, using emotions to control the temporal expression of behaviour. However, she focuses on modelling specific emotions and their impact on reinforcement learning systems, rather than focusing directly on control mechanisms. In contrast, our flexible latch is simple to implement and incorporate into any standard module-based agent architecture. Also, she uses rising levels of emotions as the source of interruptions, rather than dealing with inefficiencies caused by interruptions generated by the external environment.

Interestingly, several established models of consciousness are similar to
our new model of flexibly-latched drives. Norman and Shallice [33] describe
consciousness as a higher-cost attentional system which is brought on line
whenever the more basic, reliable, low-cost action-sequencing mechanism is
unable to proceed. Our system of flexible latching also operates by recogniz-
ing interruptions. It would be plausible in a system with modules capable
of deliberation to have interruptions trigger these rather than the simple re-
assesment of existing goals demonstrated above. More recently, Shanahan
[36] proposes a model of mutually-inhibiting motives in a global workspace.
We do not agree with Shanahan that such models can account for all of
action selection. Tyrrell [40] provides an extensive critique of a
very similar spreading-activation architecture, The Adaptive Neural Archi-
tecture [30] (more commonly referred to as Maes’ Nets [19]), explaining why
spreading-activation models cannot scale to a full action-selection mecha-
nism. The problem is simple combinatorics — a problem that architectures
like ACT-R and IDA address by focussing on just one plan subset of the full
network [19, 2]. This focussing makes these architectures functionally simi-
lar to script-based dynamic-planning systems, although their actual action-
selection mechanisms are far more complex. However, as this paper makes
clear, we do think that a system like Shanahan’s or Maes’ could well account
for high-level goal arbitration.

IDA is a cognitive architecture specifically designed to implement a the-
ory of consciousness [3]. IDA is not only a model, but also a working AI
architecture which has been used to create recommender systems for the US
Navy. Its newest version, LIDA provides the functionality of flexible latches through “timekeeper codelets” [4, p. 30] which keep a proposed action salient long enough for a variety of options to be debated. This system could well be effective, and is certainly more conducive to human-like metacognition than the system proposed here. However, our flexible latches are simpler and probably sufficient for most autonomous AI applications.

The problems Tyrrell identified with spreading activation models are to some extent addressed by [22], who recommend generating a system of attractors in the networks. This achieves an effect similar to the latching shown here. However, again the mechanism and architecture presented here are much simpler than spreading activation, even without the attractor system [9].

The difficulties in scaling spreading activation networks draw attention to an important limit of our work. Although we have shown substantial efficiency improvements, temporal costs still increase linearly with the number of interruptions. Further, some forms of interruptions will necessarily increase with the number of potential behaviours — in particular those that are generated by the action-selection mechanism itself as higher priorities trigger. What this implies is that agents should have a limited number of high-level motivations which are contested this way.

What we present here is a cognitively-minimal mechanism which makes substantial improvements to an otherwise reactive action-selection system. Elsewhere, we explore in more detail the earlier suggestion that due to
LeDoux that the psychological entities called *drives* and *emotions* may be seen as a chemically-based latching system, evolved to provide persistence and coherence to the otherwise electrically-based action selection provided by the central nervous system [13]. We hypothesise that in nature, each drive or emotion — with its associated pattern of hormonal regulators and species-typical actions — might be viewed as serving one such high-level goal or need. We recommend that a system such as our flexible latch should similarly be used for each high-level goal an agent has that requires a time budget in an artificial cognitive system.

5 Conclusions

In this paper we have presented a relatively simple way to introduce flexible latching into an autonomous system and presented an analysis of how to determine appropriate thresholds that govern the execution of lower-priority behaviours. The agents we considered have been specified using the behaviour-oriented design methodology: each agent consists of a set of modules that specify specific behaviours as well as a dynamic plan that prioritises amongst these behaviours. We take this as a fairly standard modular architecture using scripted dynamic plans for action selection, and then demonstrate how to extend that action selection to improve its efficiency.

We demonstrate our system using four behaviours derived from a tool for modelling primate social behaviour. Two behaviours — eating and drinking
— are essential to the immediate survival of the agent and are of highest (and
equal) priority. The third, grooming, represents a mission-critical behaviour
though it is not essential for immediate survival. This and the fourth, default
behaviour (exploring) can only be executed if the higher priority behaviours
are managed efficiently. Each behaviour is composed of a number of indi-

dividual actions and we distinguish between primary and secondary actions.
Secondary actions are those required to perform the primary action; the pri-
mary action is the core consumatory action of the behaviour and satisfies
the agent’s need that triggers the behavioural module. Efficient execution
of behaviours requires the agents to (a) minimise the execution of secondary
actions, and (b) acquire sufficient satisfaction (energy in our case) to be able
to carry out lower-priority behaviours.

The behaviour- (or action-) selection mechanism we have introduced con-

sists of three thresholds: A lower threshold $\delta$ that triggers the behaviour
depending on the agent’s internal state, an intermediate threshold, $\psi$, that
acts in case the agent is interrupted and an upper threshold, $\phi$, that causes
the behaviour to terminate. The addition of these thresholds does not al-
ter the priorities of the behaviours (which are still governed by the dynamic
plan) but may delay (or not) the execution of lower-priority behaviours and
may have a significant impact on the ratio of secondary to primary actions
performed by the agent. We demonstrated their efficacy in two experimental
settings. Without latching (i.e., only a lower threshold), the agent dithers
between food sources, leaving no time to execute lower-priority behaviours.
Latching (i.e., lower and upper threshold) allows for persistence but may be hugely inefficient in the presence of interruptions. The persistent pursuit of unsatisfied behaviours may lead to an unsustainable frequency of secondary task executions.

The experiments allowed us to determine the most useful setting for the intermediate threshold, above which an interrupted agent may reconsider its behaviour priorities. The results show that the utility of latching, as long as the latch is sufficiently large, where there is a significant cost of switching between goals. Flexible latching addresses a reduction in performance of latches when there are interruptions. We found however that the intermediate threshold is usually not required, or more precisely, can be set to be equal to the lower threshold. In our experiments, it was optimal for agents to reconsider priorities whenever interrupted. This result may not hold if interruptions are more frequent and/or the size of the latch is smaller, since either case would increase the probability that persistence is needed. Finally, we also explored the case where the agent may die if essential behaviours are carried out inefficiently. We found that latching significantly improves the rate of survival of the agent.

We have discussed how this mechanism, despite its simplicity, or because of it, may be relevant to numerous existing artificial cognitive architectures, and we have drawn parallels to animal-like decision making processes. Although the validation presented here is admittedly limited, these results do match expectations derived from our observations in nature concerning the
life-history strategies for species that tend to be correlated with more cognitive ability. At the same time, the work presented here also allows for extremely simple implementations such as hand-coding heuristic indicators of interruption.

There are numerous possible avenues to be explored in the near future. In our experiments, we chose the same thresholds for all behaviours, allowing a centralised approach that involves little overhead. However, it would be interesting to highlight potential differences in the efficiency of an agent’s action selection when all behaviours have individual threshold settings. Furthermore, the thresholds may be adjusted dynamically over time (e.g., using a simple feedback control loop) or in artificial life contexts might be individually evolved.

Acknowledgements

Hagen Lehmann first identified that strict latching seemed inappropriate for modelling primate behaviour; his model provided the initial behaviour code that was extended for this research. The research was conducted by PR based on a suggested fix by JJB. This research was funded by the British EPSRC AIBACS programme, grant number GR/S79299/01 and EPSRC grant EP/E058884/1. It was first presented at the AAAI 2008 Fall Symposium on Biologically Inspired Cognitive Architectures; we thank the organisers of that meeting, the other participants and the reviewers both for AAAI and Cognitive Computation.
References


