The Ecology of Action Selection: Insights from Artificial Life. Electronic Supplementary Material

Anil K. Seth[1,2]

[1] *The Neurosciences Institute, 10640 John Jay Hopkins Drive, San Diego, CA 92121, USA*
[2] *The University of Sussex, Brighton, BN1 9QH, UK*

Supplementary Material S1: Minimal action selection

(*a*) *Description of the model*

The simulated environment is a spatially continuous unbounded area within which 3 types of objects can exist - 'grey food', 'black food', and 'traps' - in addition to the agent itself. Each object is a circle of radius 16 units, except for the agent which is a circle of radius 5 units. All objects appear within a 200 by 200 unit area of the environment (the agent is not constrained to remain within this area). There are 3 items of grey food, 3 of black food, and 9 traps.

The agent (figure 3A in the main text) possesses 2 wheels, and 3 sensor pairs, with each sensor pair responding to a different object type. It also possesses 2 internal batteries - one for grey food ($\mathcal{B}_g$) and one for black food ($\mathcal{B}_b$) - which diminish at a steady rate during the lifetime of the agent. Encounters with grey or black food replenish the corresponding battery, but if both reach zero, or if the agent encounters a trap, then it will die. The 3 sensor pairs respond to the distance from the agent to the nearest instance of each type of object, with each sensor ranging linearly from 100 (at the object) to 0 (200 or more units distant). If an object is to the left of the agent, the corresponding sensor on the left of the agent will respond with 20% greater activation (subject to the maximum output value of 100) than the sensor on the right, and *vice-versa* if the object is to the right of the agent.

The links between the sensors and the motors transform the sensor input signal (range [0,100]) into an output signal (range [-1, 1]) in a manner specified by a transfer function (figure 3B in the main text). It is the shape of this transfer function that is evolved, and it is also possible for this shape to be modified during the lifetime of the agent by the values of either $\mathcal{B}_g$ or $\mathcal{B}_b$, as described below. Each connection illustrated in figure 3B of the main text represents 3 independent and concurrently active links, and left/right symmetry is imposed such that both sets of 9 links are identical to each other. The link outputs are combined at the wheels; for each wheel the relevant link outputs are summed, passed through a sigmoid function, and then scaled to the range [-10,10] to set the wheel speed.

[†]Author for correspondence anil.k.seth@gmail.com.

The model is initialised by placing the objects and the agent randomly within the environment. The movement of the agent is then calculated on the basis of the wheel speeds: if both wheel speeds are set to 10 the agent moves forward at a maximum speed of 2.8 units per time-step. Each battery has a maximum (and initial) level of 200, which decreases by a single unit each time-step. If the agent encounters a food object, the appropriate battery level is restored to 200, and the object is replaced at a different random location. The agent has a maximum lifetime of 800 time-steps.

(*b*) *Genetic encoding scheme and algorithm*

As mentioned above, 9 links need to be specified for each agent. Each genotype consists of 83 integers in the range [0,99]; thus 9 integers for each of the 9 links, and one integer each for the left wheel and right wheel sigmoid threshold values. Figure 1 illustrates how the 9 integers for each link specify the shape of the transfer function. The offset and thresholds are set by scaling the first 6 integers onto the range [-100,100], with the restriction that the second threshold must follow the first. Note that all scaling is linear and maps onto continuous values. The gradients are set by scaling the relevant integers to the range $[-\pi/2, \pi/2]$, and then taking the tangent. The sigmoid thresholds are set by scaling to the range [-3,3]. Specifying the influence of the battery levels on the shape of the transfer function is accomplished as follows. If the 9th integer is even, then the shape of the function can be influenced by $\mathcal{B}_g$, and otherwise by $\mathcal{B}_b$. The relevant battery level modifies the shape of the transfer function in two ways. First, through 'offset modulation' γ, where γ is obtained by scaling the 8th integer to the range [-1,1]. Second, through 'slope modulation' α, where α is obtained by scaling the 7th integer to the range [0,1]. Let the output of the transfer function at time t, prior to battery modulation, be $s(t)$. Offset modulation is applied first:

$$s'(t) = s(t) + \frac{\gamma \mathcal{B}_r(t)}{2}. \quad (1)$$

where $\mathcal{B}_r(t)$ represents the level of the relevant battery at time t. Slope modulation is applied next:

$$s''(t) = s'(t)\left(1 + \frac{\alpha}{100}(\mathcal{B}_r(t) - 100)\right) \quad (2)$$

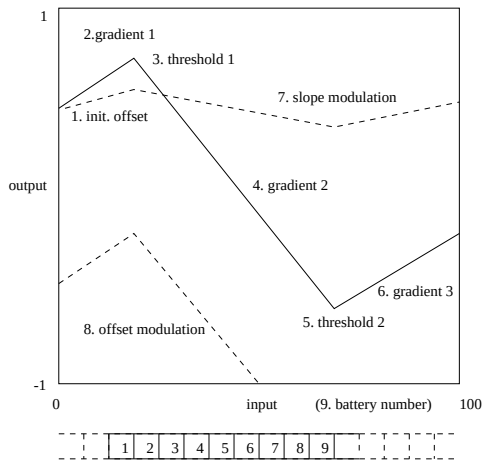

Figure 1. Genetic encoding scheme. Each sensorimotor link requires 9 integers to specify the various link parameters. The first 6 integers specify the basic shape of the transfer function transforming the sensor input into an output signal, and the final 3 integers specify how this shape can be modified by battery levels. The shape of each link can be modified by either $\mathcal{B}_g$ (if integer 9 is even) or $\mathcal{B}_b$ (if integer 9 is odd), and the level of this battery can then influence both the overall gradient of the function (to a degree specified by integer 7), and the offset (to a degree specified by integer 8).

where s'' represents the final link output. Note that there is no requirement that any given transfer function, connecting a particular sensor type to a wheel, should be influenced by the battery corresponding to the object type that its sensor responds to.

To evolve the shapes of transfer functions, we employed a distributed GA with a population size of 100. The fitness function rewards a high average battery level and is calculated incrementally for each time-step that the agent is alive:

$$\mathcal{F} = \frac{\mathcal{B}_g + \mathcal{B}_b}{400} \tag{3}$$

This function rewards agents that live long (by keeping at least one battery level above zero and by avoiding traps), and that visit food items as often as possible.

The details of the GA are as follows. A population of 100 random genotypes is arranged on a 10 by 10 toroidal grid and the fitness of each assessed. The following cycle then repeats. A random grid position is chosen, and a 'pool' of nine genotypes constructed from the 3 by 3 sub-grid surrounding this position. Two of the fittest members from this pool are chosen as parents using stochastic rank-based selection (with replacement) and used to generate a new genotype by means of crossover and mutation. The new genotype is placed back into the population in place of the weakest pool member (again chosen using a stochastic rank-based scheme) and evaluated; the parents are re-evaluated with a probability of 0.8. Crossover probability is set at 0.5, with a 0.04 probability of point mutation per allele. Each point mutation shifted the value of the allele by value selected from the (integer) range [-5,5] according to a Gaussian distribution; if the shift transgresses the allowable range [-100,100], the post-mutation value is set by selecting a random value between the pre-mutation value and the bound in question. One generation of the GA corresponds to 100 repetitions of this cycle. Fit individuals consistently evolved in about 200 generations.

Supplementary Material S2: Probability matching

(*c*) *Description of the model*

The environment comprises an infinite plane containing 4 grey and 4 black food items (radius 8). Both food items and agents (radius 2.5) are initialised within a 200 by 200 'patch' centered at the origin. Each item type is associated with a probability that encounter will lead to full replenishment of the agent's internal battery. These probabilities (P_{gry}, P_{blk}) represent the resources available from each item type; black food items *always* replenish the battery ($P_{blk} = 1.0$), and P_{gry} is experimentally manipulated. Each time an agent encounters a food item, the item disappears to be immediately replaced in another random location in the patch, thus ensuring a constant density of available resources. The agent's battery level depletes at a rate of one unit per time-step; if the battery level reaches zero the agent dies. Encounters with conspecifics (if any) have no effect.

Each agent possesses 5 sensors, 4 of which are sensitive to food items (in 2 left/right pairs) and one of which reflects the battery level. The internal architecture of each agent comprises a feedforward neural network. We use a neural network architecture because such architectures are known to be flexible and, in contrast to the transfer function architecture of the previous section, require little introduction. In the present network, input units scale sensor values to the range [0,1], with all interconnecting weights in the range [-1,1]. Each unit applies a sigmoid function to the sum of its inputs (plus a threshold value), with each output scaled to the range [0,1]. Motor outputs are scaled to the range [-10,10] to set wheel speeds; a value of 10 on both wheels translates to a speed of 2.8 distance units per time-step. A genotype of length 46 is required in this model, 42 loci to specify the weights and thresholds of the network (5 input units, 5 hidden units, and 2 output units), and 4 to specify how well the agent is able to discriminate between the food types, as described in the following.

In the general case each agent has i sensor pairs, each of which is associated with a set $\{d_{gry}, d_{blk}, ..., d_g\}$, with g indexing resource types. Each d lies in the range [0,1], and specifies the probability with which the associated sensor will perceive an item of type g; all d values are genetically specified. In the present case, with 2 sensor pairs and 2 d values per pair, 4 additional loci are required. The scheme functions in the following way. Every time an item is initialised it is tagged with the identity of each sensor pair that can perceive it. For example, if an agent has $[d_{gry} = 1, d_{blk} = 0]$ for its first sensor pair (s_1) and $[d_{gry} = 0.5, d_{blk} = 1]$ for its second (s_2), then a grey food item initialised within the range of the agent will be tagged as perceivable by s_1, and also by s_2 if (and only if) $\mathcal{R} < 0.5$ (where $\mathcal{R}$ is a random number in the range [0,1]). During each sensorimotor cycle, each sensor pair of each agent responds to its nearest perceivable item (if any), ranging linearly from 100 (at an item) to 0 (≥ 200 distance units away). This sensor scheme was designed to provide a source of stochasticity in agent behavior which would allow agents to generate flexible behaviour (see [1] for further details).

(*d*) *Experimental details*

A distributed GA was used to evolve populations of genotypes in each of 8 conditions; 4 involving a single isolated agent (the S - single agent - condition set) and 4 involving 3 agents derived from the same genotype (the M - multiple

agent - condition set). The fitness function used - in all conditions - was:

$$\mathcal{F} = \sum_{t=1}^{800} \frac{\mathcal{B}}{200}, \tag{4}$$

where t indexes time-steps and $\mathcal{B}$ represents the battery level (at time t), with each evaluation lasting for a maximum of 800 time-steps. This function rewards agents that live long and forage efficiently. Each genotype was evaluated 4 times each generation.

Both the S and M condition sets involved evolving and testing agents in environments distinguished by the value of P_{gry}. Four values of P_{gry} were employed in each condition set; 1, 0.66, 0.33, and 0, with $P_{blk} = 1$ in all conditions. Genotypes of high fitness, in both S and M condition sets, reliably evolved after about 400 generations in each condition, but in each case the population was left until 1000 generations had been completed. The fittest genotypes from each condition were then evaluated, in the same conditions as experienced during evolution in each case, with the average number of responses to grey and black items (over 1000 evaluations) being recorded. The entire set of evolutions (and analyses) was repeated 12 times to obtain overall averages.

S-agents and M-agents were also assessed in a 'forced-choice' task. In this analysis, the fittest genotype from each condition was decoded into a *single* agent, which was then assessed *in isolation* by being placed equidistant from a single grey food item and a single black food item; no other items were present. Each trial was stopped as soon as one or other of the items had been visited, and again each agent was tested 1000 times. It is important to emphasise that these tests always involved a *single* agent, even if evolution had occurred in a multi-agent environment. The trial-to-trial variability in behavior during this analysis is accounted for by the stochastic nature of the sensors, as described above.

Supplementary material S3: Matching and the ideal free distribution

(*e*) *Forager intake*

Consider two patches, A and B. Following [2] and [3], we define the per forager intake rate W_i (s^{-1}) to both the forager density N_i and the resource availability F_i on each patch i as:

$$W_i = \frac{Q F_i F^*}{{N_i}^m}, \tag{5}$$

in which Q (ms^{-1}) is a measure of patch-independent forager efficiency, $F_i \in [0, 1]$ (dimensionless) represents the resource fraction in patch i, F^* (=200) represents the total resources available, and m (dimensionless) is the interference constant, which varies between 0 (no interference) and 1 (high interference). Across two patches A and B, assuming $W_A = W_B$ [the ideal free distribution (IFD) condition]:

$$log\frac{N_A}{N_B} = \frac{1}{m} log\frac{F_A}{F_B}, \tag{6}$$

Taking the total forager number to be N_T ($= N_A + N_B$), it is possible to predict both N_A and N_B directly [4]:

$$N_A = \frac{N_T}{(10^{-c} + 1)}, \qquad c = \frac{log\frac{F_A}{F_B}}{m}. \tag{7}$$

In the model, we used equation 7 to assess the fit of a population to the IFD, and equation 5 to determine the per-agent intake rate.

(*f*) *Description of the model*

The full definition of ω-sampling is provided in Appendix A of the main text. It will not be repeated here.

To assess fit to the IFD, we recorded the equilibrium distribution (after 1000 time-steps) of populations of 100 ω-samplers, for each of 9 different resource distributions across two patches A and B. Two separate populations were analysed, corresponding to two different levels of interference ($m = 1.0$ and $m = 0.3$). In each case, agents were initially randomly allocated to either A or B. Then, each time-step, the resource obtained by each agent was calculated (equation 5), the ω-sampling heuristic applied for each agent, and the new agent distribution determined. The final equilibrium distributions were compared with the predictions of the IFD (equation 7).

The action selection behavior of isolated agents was assessed under the four reinforcement schedules described in the main text (basic, VR VR, VI VI, and VI VR). For the each schedule, single ω-samplers foraged in isolation for 1000 time-steps under each of 9 different resource distributions.

For the basic schedule, forager intake was determined - as in the IFD experiments - by equation 5.

For the VR VR schedule, F_i was interpreted as specifying a probability that patch i will yield the fixed resource quantity F^*. Under this assumption, agent intake is determined by the random variable:

$$W_i = \begin{cases} \frac{QF^*}{{N_i}^m}, & p(F_i) \\ 0, & p\,(1 - F_i) \end{cases} \tag{8}$$

VI VI was implemented by using F_i to set delay intervals (D_i) such that $D_i = 20(1.0 - F_i) + r$, with $r \in [-2, 2]$ an integer random number. The first response to option i on each evaluation procured the full reward F^* and initialised D_i. Subsequent responses to i went unrewarded until D_i time-steps had elapsed, after which a response would again procure F^* and re-initialise D_i, with the incorporation of r ensuring that the schedule was indeed 'variable interval'.

VI VR was implemented by applying VI to one option (A), and VI to the other (B).

Both the IFD analysis and the matching law analysis were repeated 30 times each, enabling means and standard deviations to be calculated.

(*g*) *Parameter values*

Analysis of ω-samplers required selecting values for ϵ and γ (see Appendix A in the main text). We chose these parameters in two different ways: (i) by selecting fixed values ($\epsilon = 0.052$ and $\gamma = 0.427$), and (ii) by *evolving* parameter values for each condition separately. The results obtained did not depend on the method used, and were robust to variations in the fixed set [5, 6]. The motivation for the evolutionary approach was to identify parameter values that would enable an agent to perform as well as possible in a given condition [7]. For completeness, we describe this approach below.

Random initial populations (size 100) were evolved in two conditions ($m = 1.0$ and $m = 0.3$). Each agent (in each condition) possessed a genome of 2 real numbers (range [0,1]) specifying ϵ and γ. Each condition applied a tournament

	$m = 1$	$m = 0.3$
ϵ	.061	.040
γ	0.21	0.65

Table 1. Evolved parameters. Columns labelled by interference level m

GA for 100 generations (mutation rate 0.01, each mutation drawn from Gaussian distribution radius 0.13; range transgressions were truncated). Fitness was averaged over 10 separate evaluations. Each evaluation randomly assigned values for F_A and F_B ($F_A + F_B = 1.0$, total resource $F^* = 200.0$ in all conditions), and randomly allocated agents between A and B. The fitness of each agent was determined by total accumulated resources after 1000 cycles. The entire GA process was repeated 10 times in each condition, from which average (condition-specific) parameter values were computed (see table 1).

REFERENCES

[**1**] A.K. Seth. Modeling group foraging: Individual suboptimality, interference, and a kind of matching. *Adaptive Behavior*, 9(2):67–91, 2001.

[**2**] W.J. Sutherland. Aggregation and the 'ideal free' distribution. *Journal of Animal Ecology*, 52:821–828, 1983.

[**3**] M. Milinski and G.A. Parker. Competition for resources. In J.R. Krebs and N.B. Davies, editors, *Behavioural ecology*, pages 137–168. Blackwell Scientific Publishers, Oxford, 1991. 3rd edition.

[**4**] T. Tregenza, G.A. Parker, and D.J. Thompson. Interference and the ideal free distribution: Models and tests. *Behavioral Ecology*, 7(4):379–386, 1996.

[**5**] A.K. Seth. *On the relations between behaviour, mechanism, and environment: Explorations in artificial evolution*. PhD thesis, University of Sussex, 2000.

[**6**] A.K. Seth. Competitive foraging, decision making, and the ecological rationality of the matching law. In B. Hallam, D. Floreano, J. Hallam, G. Hayes, and J.-A. Meyer, editors, *From animals to animats 7: Proceedings of the Seventh International Conference on the Simulation of Adaptive Behavior*, pages 359–368, Cambridge, MA, 2002. MIT Press.

[**7**] A.K. Seth. Agent-based modeling and the environmental complexity thesis. In B. Hallam, D. Floreano, J. Hallam, G. Hayes, and J.-A. Meyer, editors, *From animals to animats 7: Proceedings of the Seventh International Conference on the Simulation of Adaptive Behavior*, pages 13–24, Cambridge, MA, 2002. MIT Press.