

Learning as a Consequence of Selection

Samuel Delepouille^{1,2*}, Philippe Preux², and Jean-Claude Darcheville¹

¹ Unité de Recherche sur l'Évolution des Comportements et des Apprentissages (URECA), UPRES-EA 1059, Université de Lille 3, B.P. 149, 59653 Villeneuve d'Ascq Cedex, France, `lastname@univ-lille3.fr`

² Laboratoire d'Informatique du Littoral (LIL), Université du Littoral Côte d'Opale, UPRES-JE 2335, B.P. 719, 62228 Calais Cedex, France, `lastname@lil.univ-littoral.fr`

Abstract Since the end of the XIXth century, the influence of learning on natural selection has been considered. More recently, this influence has been investigated using computer simulations. However, it has not yet been shown how the ability of learning can be the product of natural selection. This point is precisely the subject of this paper.

1 Introduction

Since it has been proposed independently by Lloyd Morgan [12], Osborn [13] and Baldwin [2], it is known that the activity of organisms during their lifetime can bring long term modifications to their genomes, and therefore plays a role in natural selection. The Baldwin effect has first been experimented in the 1950's on *Drosophila* by Waddington [23, 24]. It is now widely recognized that genetic evolution and learning are deeply intertwined processes. Today, computer simulations provide a tool to investigate the interaction between natural evolution and learning. Even if the complexity of the agents that are simulated is rather crude with regards to living organisms, and even though the natural processes are much simplified when simulated, it has been argued that this type of work is useful [10]. Using simulations, Hinton and Nowlan [9] were the firsts to show that learning can guide and speed-up evolution. More generally, we refer to [22] for a recent up-to-date review regarding the interaction between evolution and learning. Among other points that have yet to be explored, the ability to learn should be explained by way of natural selection if we want to remain within a strict selectionist point of view of evolution. Indeed, if natural selection is invoked as the basic process of evolution, it has to create the structures that evolve, as well as all the other processes. Among these, is the ability of learning. At the most basic level, learning is the ability for an animal to modify its behavior according to the stimuli it receives from its environment. This has been modeled by Thorndike as the "law of effect", which says that the frequency of emission of a certain behavior increases when it has been followed by favorable

* Samuel Delepouille acknowledges the support of the "Conseil Régional Nord-Pas de Calais, France", under contract n 97 53 0283

consequences in the past [20, 21]. Subsequently, the law of effect has been studied and experimented in numerous works and by numerous researchers [4, 11]. Skinner proposed the principle of selection of behavior by its consequences [16, 18] which is basically the same thing as the law of effect, even though the theoretical framework has evolved since Thorndike [5]. The emphasis we put on the law of effect clearly distinguishes our work from others, such as [1]. In our study, agents are not supervised (at least, not in a strong sense such as involving backpropagation mechanisms or so); they behave and they eventually receive stimuli on their input sensors, and emit behaviors, getting neither reward nor even a value measuring any goodness of the emitted behavior; their lifetime activity selects them for providing offsprings to the next generation.

In the sequel, we first set up the stage by describing the model we use for agents, and processes of natural selection and learning. Then, we present the tasks the agents are facing, that is their environment. Afterwards, we present the result of the simulations. We finish with a discussion of this work.

2 The Model

In this section, we describe the agents that evolve, the processes of evolution and lifetime behavior. Natural selection is simulated using a genetic algorithm.

2.1 Evolved Structures

The agents that are evolving are made of a set of N input sensors (IS) to let them perceive their environment, a set of N behavior units (BU) to let them act onto their world, and a neural network which controls their activity and let them adapt to their environment during their lifetime (see Fig. 1). Agents are not located spatially in their world; they merely interact with each others. The neural network of an agent is made of C layers, each of N neurons. The IS's receive binary stimuli from the environment. Let us call "unit" either an input sensor, a behavior unit, or a neuron. Then, each neuron receives the output of the N units of the previous layer (input connections) and the output of the N units of the next layer (re-entrance connections); that is, each neuron receives $2N$ inputs. Owing to these connections, the neural network of an agent perceives its own behaviors since the BU's feed back the output layer of the network. Each BU is connected to one neuron of the last layer of the network in a one-to-one relationship. At each time step, only one BU is active, the one associated with the neuron having the highest potential, in a winner takes all fashion, ties being broken at random between neurons having the same potential. In this paper, C has always been set to 3, and N to 10. So, there is an input layer of neurons, an output layer, and a layer of hidden neurons. The characteristics of the neural network (that is, the characteristics of the neurons as well as those of the connections) are encoded in a genome. The response of each neuron is characterized by a boolean value which indicates whether the neuron is active or not, and by

6 real numbers: $\alpha, \beta, \gamma, a, b \in [-100, 100], \epsilon \in [-1, 1]$. These 6 parameters characterize the response of the neuron with regards to surrounding neurons and its own past activity. As far as any neuron is connected to each neuron of the two surrounding layers, each neuron is also characterized by $2 \times N$ weights. Each weight is characterized by a quadruplet (V, E_a, E_b, E_{ab}) , where V is its initial value, while E_a, E_b and E_{ab} control how its value changes during learning. The value of these 4 parameters lies in $[-1, 1]$. Finally, the whole network is characterized by two numbers A_c and A_p which are discussed below (see Sect. 2.3). To sum-up, the genome of an agent encodes $C \times N$ neurons, each one constituted with one bit of activity, 6 real numbers, and $2 \times N$ weights, each weight being itself made of 4 real numbers, which totals in $CN(8N + 6) + 2$ numbers (plus one bit).

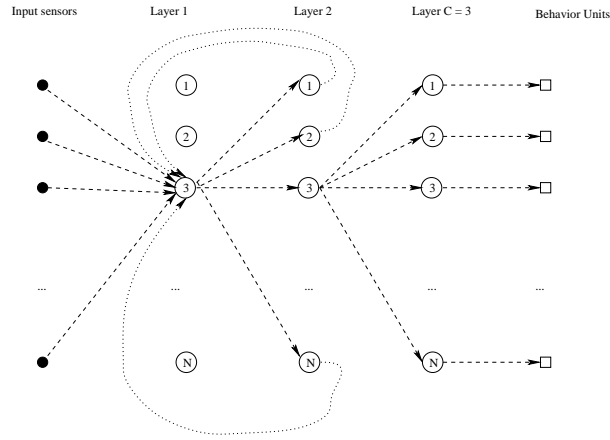


Figure 1. The internal structure of the agents being evolved. For the sake of clarity, not all units and not all connections are represented. On the neuron 3 of the first layer, we can see 4 out of the N input connections coming from the input sensors, and 3 out of the N re-entrance connections coming from the neurons of the second layer. Refer to the text for more details.

2.2 Evolution

The evolution process is simulated using a genetic algorithm acting on the previously described genomes. Basically, the algorithm that is performed is:

```

Initialize a population of agents
WHILE stopping criterion is not fulfilled DO
  // lifetime
  FOR i ∈ [1, life_duration] DO
    FOR j ∈ [1, number_of_agents] DO

```

```

        Activate
        Learn
    DONE
    DONE
    Evaluate the fitness of the agents
    // evolution
    Constitute the population of offsprings using genetic operators
    DONE

// Activate
BEGIN Activate
    Activate input neurons
    FOR i ∈ [1, Ac] DO
        Select one neuron at random
        Update its potential
    DONE
    Observe response
END Activate

// Learn
BEGIN Learn
    FOR i ∈ [1, Ap] DO
        Select one connection at random
        Update its weight
    DONE
END Learn

```

This algorithm is described in the following paragraphs.

Genetic operators To constitute the population of offsprings, we use 6 operators: one recombination and 5 kinds of mutation. Each mutation acts at a certain level of the genome: weights, neurons, and their expression, and how the network learns. Basically, one point-crossover is used on two parents to produce one offspring. The crossover can only cut between two different neurons. The resulting genome inherits the two parameters A_c and A_p from one of its two parents drawn at random. With regards to mutation, the first one acts on a weight. It consists in choosing at random a weight in the genome and modify its value of at most 10%. This yields a mutation that has only a slight effect. The probability to apply this mutation is noted μ_w ; it can be rather high as long as its effects are not very disruptive. The second mutation acts on a neuron and it consists in resetting at random all the characteristics of a neuron. The probability to apply this mutation is noted μ_n ; obviously, its effects on the activity of the network can be rather important. Thus, we use a rather low value for μ_n . The third mutation concerns the expression of a neuron and merely acts on the activity bit of a neuron in the genome. It is applied with probability μ_e . Toggling this bit can have important consequences on the activity of the network. When inactive, a neuron can “travel” along generations without being noticed, and it can undergo mutations which do not modify the fitness of the agent (neutral mutations). When made active again, it can greatly modify the activity of the network and, thus, the fitness of the agent to which it belongs. The two last mutations concern the parameters A_c and A_p . With probability μ_p , each of these two variables can be modified independently. Their mutation changes slightly their value (± 10 units).

2.3 The Lifetime of an Agent

In this section, we describe how an agent learns during its lifetime. Before that, we describe how the neural network reacts to stimuli to produce its behavior, that is the procedure Activate of the algorithm.

Activate To come close to a concurrent activity of neurons, the neurons of an agent are activated as follows. Iteratively, A_c neurons are drawn at random, letting it possible that a neuron is drawn several times during a single invocation of “Activate”. Let $l \in [1, C]$ and $n \in [1, N]$ drawn at random be the layer and the number in the layer of a neuron to be activated. This neuron should have its expression gene turned on. Let us note $A_t(l, n)$ the potential of this neuron at time t . $A_{t+1}(l, n)$ can be written as a function of its current potential $A_t(l, n)$ and the weighted sum of its inputs $Se_t(l, n)$, the weighted sum of its re-entrance $Sr_t(l, n)$, and h_t a random noise uniformly drawn in $[-1, 1]$. Then, the “Update its potential” step of the algorithm is as follows:

$$A_{t+1}(l, n) = f(\alpha_{ln} \cdot Se_t(l, n) + \beta_{ln} \cdot Sr_t(l, n) + \gamma \cdot A_t(l, n) + \epsilon_{ln} \cdot h_t) ,$$

where

$$\begin{cases} Se_t(l, n) = \sum_{k=1}^N V_t^e(k, ln) \times A_t(l-1, k) \\ Sr_t(l, n) = \sum_{k=1}^N V_t^r(k, ln) \times A_t(l+1, k) \end{cases} ,$$

where $V_t^e(k, ln)$ is the weight at time t of the k^{th} input connection of neuron (l, n) , and $V_t^r(k, ln)$ is the weight at time t of the k^{th} re-entrance connection of the same neuron. The function $f(x)$ is linear by parts. It is determined according to the value of a_{ln} and b_{ln} :

– if $a_{ln} \neq b_{ln}$, then $g(x) = 2(x - a_{ln})(a_{ln} - b_{ln}) - 1$
and

$$\begin{cases} f(x) = -1 & \text{if } g(x) \leq -1 \\ f(x) = g(x) & \text{if } -1 < g(x) < +1 \\ f(x) = +1 & \text{if } g(x) \geq +1 \end{cases} ,$$

– if $a_{ln} = b_{ln}$, then

$$\begin{cases} f(x) = -1 & \text{if } x < a_{ln} \\ f(x) = +1 & \text{if } x \geq a_{ln} \end{cases} .$$

The potential of all neurons that are not updated remains unchanged. Finally, as long as a neuron has its expression gene turned off, its potential remains 0.

Learning This paragraph describes the “Learn” action in the algorithm. Learning is not determined by genes but by the variables (V, E_a, E_b, E_{ab}) that are genetically encoded that produce the way the network is activated. Hence, learning is not strongly genetically predetermined but remains under the influence of

the environment to a large extent. We call “Learning” the activation of the network according to the stimuli it receives from its environment. It consists in a modification of the weights of the network. This modification is not made deterministically but at random: a connection of the network is drawn at random, and its weight is modified according to the neurons to which it is connected. This modification is made iteratively A_p times. Again, a connection can be selected more than once in a single invocation of “Learn”.

More precisely, let $l \in [1, C]$, $n, k \in [1, N]$, and $r \in \{\text{false}, \text{true}\}$ drawn at random.

If $r = \text{false}$, the weight of the k^{th} input connection is updated according to:

$$V_{t+1}^e(k.ln) = V_t^e(k.ln) + E_a \times A_t(l, n) + E_b \times A_t(l-1, k) + E_{ab} \times A_t(l, n) \times A_t(l-1, k) ,$$

If $r = \text{true}$, the weight of a the k^{th} re-entrance connection is updated according to:

$$V_{t+1}^r(k.ln) = V_t^r(k.ln) + E_a \times A_t(l, n) + E_b \times A_t(l+1, k) + E_{ab} \times A_t(l, n) \times A_t(l+1, k) .$$

The description of the model is now finished.

3 Simulation

The agents have been the subject of three tasks. In each case, the fitness function is directly related to the behavior of the agents.

3.1 The Tasks

We describe three conditions under which the agents have been evolved, namely a discrimination task, a task known in the psychological literature as “mutual fate control”, and a derived task we call “mutual fate control with selection of behavior”.

Discrimination Task aims at selecting those networks that are able to learn to emit behaviors which emission have been followed by favorable consequences in the past. This task consists in discriminating two stimuli S1 and S2. In the presence of S1, the behavior of the agent must be B1; in the presence of S2, it should be B2. S1 and S2 are input on two different IS’s. B1 and B2 correspond to the activation of two different BU’s. When the required behavior is emitted, a stimulus is put on a certain IS (distinct from the IS’s that receive S1 and S2): this stimulation acts as the positive consequences (a reward). To obtain networks that are not only able to emit B1 when facing S1, and B2 with S2, the task is made of a series of epochs. In each epoch, one combination is rewarded: either S1-B1, S2-B2, or S1-B2, S2-B1. The agent receives no signal so as to know which of the two combinations is the rewarded one at a given moment; at the beginning of each epoch, one combination is selected at random with

probability 0.5 to be the one which will be rewarded during the epoch. Owing to this, agents should have adaptive capabilities, that is, they should be able to learn the good combination of stimulus-behavior, and they should be able to learn to adapt their behavior along their “lifetime” according to the stimuli they receive from their environment. Each epoch is made of 1 000 stimuli. As the agents have no means to measure time, this task is not markovian. Each stimulus is emitted with probability 0.5 at each time step. The population is made of 10 agents. The fitness function is defined as the cumulated number of rewards that are received along the 10 epoches. One generation is made of 10 epoches. The maximal fitness is then 10 000. Initially, the population is drawn at random. The two worst individuals are removed from the population at the end of each generation. They are replaced by two offsprings of the two fittest agents of the current population obtained by way of the genetic operators. μ_w is set to 0.05, μ_n is set to 0.01, and μ_e is set to 0.05. μ_p is set to 1.0. Initially, the values of A_p and A_c are drawn uniformly at random in $[1, 1000]$.

Mutual fate control (MFC) is drawn from the field of social psychology and deals with a situation of cooperation. It was introduced in 1957 [15]. The idea is to confront two agents A and B which have two possible behaviors, B1 and B2. Their behaviors have only consequences for their party as follows:

- if the behavior of A is B1, then B receives positive consequences,
- if the behavior of A is B2, then B receives negative consequences,
- if the behavior of B is B1, then A receives positive consequences,
- if the behavior of B is B2, then A receives negative consequences.

So, an agent does not receive the consequences of its own behaviors but the consequence of the behaviors emitted by its party (hence, one controls its party’s fate). This situation leads to complex dynamics which have been discussed in [6, 7]. As in the previous task, the goal is to select adaptive agents, not agents that are only able to answer systematically by behavior B1. Thus, we perform 10 epoches and for half of the epoches, the consequences are reversed: if A (resp. B) emits B2, then B (resp. A) receives positive consequences, while if A (resp. B) emits B1, then B (resp. A) receives negative consequences. Once again, the agents receive no information with regards to the fact that a new epoch begins and in which among the two cases it falls in. For this task, the population is initialized at random. The selection step as well as the probability to apply the operators are identical to those of the first task. During 10000 iterations, two agents are facing each other. We do not select these two agents at random because this would lead to difficulties with regards to our goal. Indeed, whatever an agent does, its fitness is fully determined by its party. Hence, an agent that would provide its party lots of rewards while receiving no reinforcer from its party would be rated very bad, while the second would be rated very high: this would be totally unfair, and would not match our goal: we want agents which provide rewards to their party to be fit, while those which do not provide rewards to their party fit poorly. To avoid this effect, we select one genome in the

population and generate two clones out of it which then face each other in the task. As long as agents learn during their lifetime, this is not the same as if one agent was facing itself; actually, we can see the experiment as two twins facing each other. Moreover, we can add that it would also be interesting to study the evolution of the population where two different agents would face each other in the task. This has to be done and, it is not so clear that, finally, the evolution of the fitness of the population would be very different.

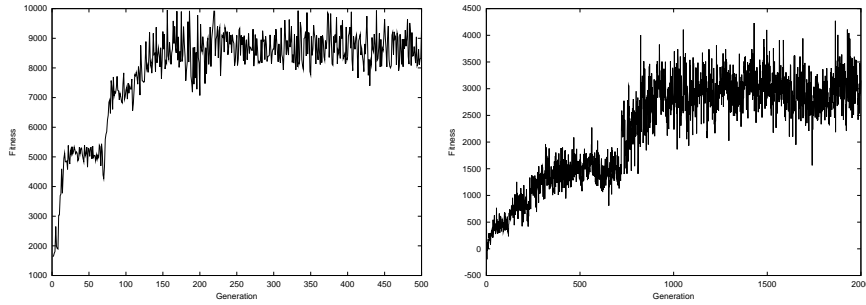
Mutual fate control with selection of behavior is strictly identical to the previous one except with regards to how the initial population is set. Instead of being random, the population is initialized with agents that have been able to pass a test procedure. This test procedure is made in such a way that, to pass it, the agent should be able to perform a very basic learning. This procedure consists in systematically rewarding a certain behavior. If the frequency of emission of this behavior is higher than for other behaviors, the test is passed by the agent. 813 agents have been drawn at random to obtain a population of 10 agents that pass the test 10 times in a row. Once constituted, the population undergoes MFC.

3.2 Results

The agents and the processes that have been described have been implemented in Java to perform the simulations. This section presents the results of these simulations, task by task.

Discrimination task Fig. 2(a) plots against generations the average performance of the population of agents in the discrimination task. Basically, after a rapid increase of performance, it levels for a while, then, it increases again to reach a much higher level (approximately 8500). The first level corresponds to a population where agents are able to receive the reinforcer one time out of two: the ordinate is 5000 whereas the maximum that can be obtained is 10000 (see Sect. 3.1). Then, in this population, the ability to discriminate appears later: after 200 generations, approximately 90 % of the agents are able to perform the discrimination task.

This simulation shows that selection is able to produce the ability to discriminate, that is to learn its behavior from the consequences its emission receives from the environment. It should be emphasized that the environment is dynamic: the reinforcers are not received deterministically after the emission of a behavior; nothing in the environment helps the agents know in which condition they are. The ability to emit a behavior that have brought favorable consequences in the past is the core of the law of effect and, subsequently, of the principle of selection of behaviors by their consequences. Thus, we have shown that this can be produced by natural selection. Based on that observation, the next two simulations show that the ability of learning confers a great advantage.



(a) Average performance of the agents facing the discrimination task against time. The performance is measured as the number of reinforcers that are received. The maximum is 10000.

(b) Average performance of the agents facing MFC against time. The performance is measured as the number of reinforcers that are received. The maximum is 10000. The initial population is made of agents which genome is drawn at random.

Figure 2. Evolution of performance of agents along time on the first two tasks.

Mutual fate control Fig. 2(b) plots the evolution of the average performance of agents in the MFC. Clearly, genetic selection leads to agents that have an increasing ability to control their party. This evolution is made of different phasis. Sharp increases in the performance happens from time to time, separated by periods that are rather steady. It is noticeable that after 2000 generations, the performance of the agents is still rather low: it receives only 35 % of the rewards they could receive.

Mutual fate control with selection of behavior When the initial population of agents that face MFC is composed of agents that have passed the test procedure, the evolution of performance of the population is very different. Fig. 3 shows this difference where A is the evolution of a random population while B is the evolution of the agents that have passed the test. At the beginning, the performance of the two populations are rather identical. But after very few generations, population B shows a much better performance than population A. After 200 generations, population B obtains 85-90 % of the available reward. The discrepancy to 100% is due to the fact that at the beginning of each epoch, the agents have to adapt their behavior.

4 Conclusion and Discussion

In this paper, using computer simulations, we have shown that a population of agents can acquire the ability to learn during their lifetime by way of natural selection, giving way to the possibility of the Baldwin effect. Learning means that

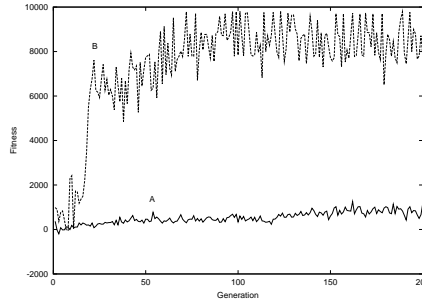


Figure 3. Performance of two populations of agents facing the MFC against time: A is a population of agents initially drawn at random (that is, this is the same population as that of Fig. 2(b)); B is a population which is initially constituted of agents that pass a simple test of learning ability. This clearly advantages very much the population which performs well after 200 generations. As far as the environment is changing without any notice to the agents, 85-90% successes is a very good performance.

a certain structure in the agents is able to learn favorable associations between stimuli and the behavior to emit. Learning does not mean acquiring a stimulus-response reflex: the environment is dynamic so that the agents have to be able to adapt their behavior to changing environments during their lifetime. Technically, this is known as operant learning, or instrumental conditioning. This work has thus to be considered as a step further the following known facts that have been shown using evolutionary algorithms: natural selection can produce fitter and fitter individuals along generations in static and in dynamic environments; natural selection can produce neural networks that act as control architecture of animats; the Baldwin effect can speed-up evolution. These points have already been raised and studied from the point of view of engineers to obtain good solutions for optimization problems, either numerical, or symbolic: hybrid algorithms have been a favorite theme of the EA literature for many years. We have also shown that once agents are able to select their behavior according to its consequences, the interaction between agents shows complex dynamics [7], and that complex behaviors can be acquired [14, 8]. Clearly, an other originality of our work is that we put a strong emphasis on the interaction between two adaptive agents, not merely on the evolution and adaptation of a single agent in its environment. We think that this is an important point to obtain models and simulations that can bring something to the scientists who study life. Finally, we can also emphasise that the agents are facing non markovian environment.

At this point, we would like to relate this work to reinforcement learning. Clearly, the law of effect has inspired temporal difference (TD) methods as it has been argued in different places (see for example [3, 6, 19]). However, the formalization of the law of effect under an algorithmic form is far from straightforward. For sure, TD is appealing but there remains many unclear points, such as the definition of states and actions, the evolution law of Q-values or connection weights, and the role of time in the dynamics of behavior [7]. Furthermore,

the status of the reinforcement in TD is different from what we consider here. In TD, the agents are optimizing the amount of their reinforcements because this is what the algorithm has been designed to do. In our case, the reinforcement is a mere stimulus like any other stimulus received on an input sensor, but, the agent is not programmed to optimize its amount, the agents are selected according to their ability to be sensible to these stimuli; a punishment is considered as a negative reinforcement by TD although this is against experimental evidences: according to behavioral analysis, a punishment is definitively not a negative reinforcement [17]. In our work, reinforcements and punishments are considered as different stimuli, thus one is not merely the opposite of the other: they are two different things. Finally, in the present work, our goal is to show that the law of effect can be the product of natural selection; we could have tried to evolve structures (using genetic programming for example) leading to TD; instead of that, we have favored an other approach which also leads to a reinforcement architecture based on neural networks. This approach has been chosen as being more “natural” to us, and in which we have tried to minimize the number of hypothesis as well as their remoteness from natural structures: neurons do exist (they can be seen, touched and even operated) while things like states are more elusive.

References

- [1] D. Ackley and M. Littman. Interactions between learning and evolution. In Christopher Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, Santa Fe Institute Studies in the Sciences of Complexity, pages 487–509. Addison-Wesley Publishing Company, 1992.
- [2] J.M. Baldwin. A new factor in evolution. *The american naturalist*, 30, 1896.
- [3] A.G. Barto. Reinforcement learning and adaptive critic methods. In D.A. White and D.A. Sofge, editors, *Handbook of intelligent control: neural, fuzzy, and adaptive approach*, pages 469–491. Van Nostrand Reinhold, 1992.
- [4] C. Catania. Thorndike’s legacy: learning, selection, and the law of effect. *Journal of the experimental analysis of behavior*, 72:425–428, 1999.
- [5] P. Chance. Thorndike’s puzzle boxes and the origins of the experimental analysis of behavior. *Journal of the Experimental Analysis of Behavior*, 72(3):433–440, 1999.
- [6] S. Delepouille. *Coopération entre agents adaptatifs ; étude de la sélection des comportements sociaux, expérimentations et simulations*. PhD thesis, Université de Lille 3, URECA, Villeneuve d’Ascq, October 2000. Thèse de doctorat de Psychologie.
- [7] S. Delepouille, Ph. Preux, and J-Cl. Darcheville. Dynamique de l’interaction. In B. Chaib-Dra and P. Enjalbert, editors, *Proc. Modèles Formels de l’Interaction, Toulouse*, pages 141–150, 2001.
- [8] S. Delepouille, Ph. Preux, and J-Cl. Darcheville. Selection of behavior in social situations — application to the development of coordinated movements. In *Applications of Evolutionary Computing*, volume 2037 of *Lecture Notes in Computer Science*, pages 384–393. Springer-Verlag, April 2001.
- [9] G.E. Hinton and S.J. Nowlan. How learning can guide evolution. 1:495–502, 1987.

- [10] J. Maynard-Smith. When learning guides evolution. *Nature*, 329:761–762, October 1987.
- [11] D. McFarland. *Animal Behavior. Psychology, Ethology and Evolution*. Longman Science and Technology, 1998.
- [12] C. Lloyd Morgan. On modification and variation. *Science*, 4:733–740, 1896.
- [13] H.F. Osborn. Ontogenetic and phylogenetic variation. *Science*, 4:786–789, 1896.
- [14] Ph. Preux, S. Delepouille, and J-Cl. Darcheville. Selection of behaviors by their consequences in the human baby, software agents, and robots. In *Proc. Computational Biology, Genome Information Systems and Technology*, March 2001.
- [15] J.B. Sidowski, B. Wyckoff, and L. Tabory. The influence of reinforcement and punishment in a minimal social situation. *Journal of Abnormal Social Psychology*, 52:115–119, 1956.
- [16] B.F. Skinner. *The behavior of organisms*. Appleton-Century Crofts, 1938.
- [17] B.F. Skinner. *Science and human behavior*. MacMillan, 1958.
- [18] B.F. Skinner. Selection by consequences. *Science*, 213:501–514, 1981.
- [19] R.S. Sutton and A.G. Barto. *Reinforcement learning: an introduction*. MIT Press, 1998.
- [20] E.L. Thorndike. Animal intelligence: An experimental study of the associative process in animals. *Psychology Monographs*, 2, 1898.
- [21] E.L. Thorndike. *Animal Intelligence: Experimental Studies*. Mac Millan, 1911.
- [22] J. Urzelai. *Evolutionary Adaptive Robots: artificial evolution of adaptation mechanisms for autonomous systems*. PhD thesis, EPFL, Lausanne, Suisse, 2000.
- [23] C. Waddington. Genetic assimilation for acquired character. *Evolution*, 7:118–126, 1953.
- [24] C. Waddington. Genetic assimilation of the *bithorax* phenotype. *Evolution*, 10:1–13, 1956.