

Christiansen Grammar Evolution for the Modelling of Psychological Processes

Emilio del Rosal, Alfonso Ortega, Manuel Alfonseca and Marina de la Cruz
Departamento de Ingeniería Informática
Universidad Autónoma de Madrid
C/Tomás y Valiente, 11 - Madrid 28049-Spain
e-mail: emilio.delrosal@uam.es

KEYWORDS

Evolutionary computing, Associative learning, Modelling, Model design.

Abstract

Psychologists have developed models of associative learning for more than 30 years. Despite the strong efforts made, they still suffer many shortcomings. We have tried to build an integral model of habituation, the simplest type of learning within the area of associative learning and the basic support for other types. To overcome the deficiencies of traditional models, we have made use of Christiansen Grammar Evolution. This evolutionary technique is capable of automatically search for a target expression (the model) in a given formal language (the formalism of the model). Under this perspective, that we call *Automatic Modelling*, we have found models of habituation with interesting characteristics.

1 INTRODUCTION

The present research has a multi-disciplinary character. We introduce the psychological area of associative learning (AL in the remainder of the paper), and the phenomenon of habituation, as well as the computer science procedure called grammatical evolution (GE from now on).

1.1 Models of associative learning

Since the pioneer work of Ivan Pavlov (Pavlov, 1927), animal associative learning has been widely studied by psychologists, mainly under the methodological perspective of behaviourism (O'Donohue and Kitchener, 1999). From this approach, the main object of the researcher is to find the functional relation between the stimuli of the environment and the responses of the animal.

AL recognizes four groups of phenomena. The first two are sensitization and habituation. Although they are actually non-associative processes, their role is essential for the functioning of AL. The other two, classical conditioning and operant conditioning, conform the core of associative learning (see (Mazur, 2002) for a more detailed explanation).

Current models of AL suffer from two main limitations. Firstly, most of them take into account only one of the four

types presented above, isolating it from all the others. It has been previously stated (Alonso et al., 2005; del Rosal et al., 2006) that this lack of integration is a big weakness in the current state of the art, specially if we consider that one kind of learning can affect the functionality of the others. Secondly, even after taking into account only one of the types, they cannot reproduce all its main characteristics.

Thus, our perspective to create an integral model of associative learning is the following: we understand that habituation, given its filtering role (see below) of stimuli and its simplicity, is the most appropriate type of learning to begin with. Once a satisfactory model of habituation has been built, the more complex types of learning could be modelled, integrating the previously modelled functionality of habituation. Finally, in our opinion, this challenge can be better confronted with support from the latest developments in computer science for the automatic resolution of problems (i.e. GE and its extensions).

Habituation acts by decreasing the innate response of an organism to a given stimulus. This decrease is a consequence of the repeated presentation of the stimulus. For example, the response of many organisms after hearing a novel sound decreases when the sound is presented many times. Habituation has an important function in the organism's learning, its role consisting of filtering irrelevant stimuli which should not be processed. See (Hall, 1991) for a fuller account. As an example, figure 1 shows the typical habituation curve with an inter-stimulus interval (ISI) of thirty seconds and the following *spontaneous recovery* after a period with no stimulus presentation.

To assess our modelling work we identified the defining characteristics of habituation. Each of our models was evaluated in terms of its ability to reproduce them. The list was built following (Thompson and Spencer, 1966) for the first 6 characteristics (the rows in table 1). The seventh row was supported by more modern empirical evidence (Byrne, 1982; Rankin and Broster, 1992). The main previously existing models (columns) were compared according to those characteristics, to make us able to discuss the value of our models within the current state of the art.

1.2 Evolutionary automatic programming and GE

Evolutionary automatic programming (EAP) follows the same principles as any other evolutionary computing tech-

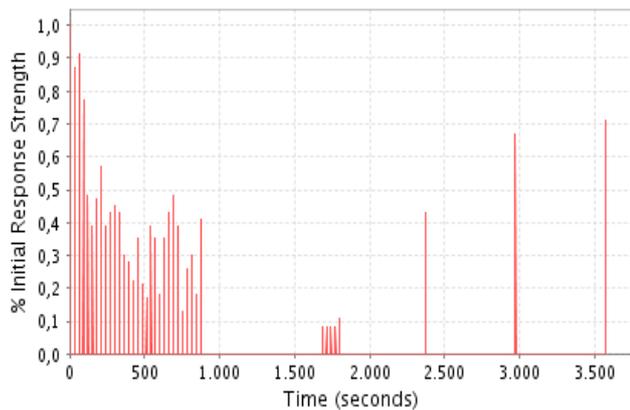


Figure 1: A stimulus is presented every 30 seconds (between seconds 0 and 1800 approximately). During this period the response of the animal to the stimulus decreases progressively. After some time with no exposure to the stimulus (seconds 1800-2400 approx.) three new stimulus presentations show a clear recovery of the animal's response. Data taken from (Rankin and Broster, 1992).

Property/Model	(Stanley, 1976)	(Wang, 1994)	(Staddon and Higa, 1996)	(del Rosal et al., 2006)
Response's exponential decrease	yes	yes	yes	yes
Spontaneous recovery	yes	yes	yes	yes
Lower intensity, faster habituation	yes	X	X	yes
Shorter ISI, faster habituation	X	yes	yes	yes
Dishabituation	X	X	X	yes
Long-term and short-term habituation	X	yes	X	X
Shorter ISI, faster recovery	X	X	yes	X

Table 1: Comparison of habituation models.

nique with the aim of creating programs automatically from a high-level description. From the beginning, genetic programming (Koza, 1992) (the main and first branch of EAP) has faced two main weaknesses. The first is getting over the closure problem, i. e., the generation of syntactically valid expressions (or programs) and the preservation of their validity under the action of the evolutionary operators (mutation, crossover...) The second is building a system independent of the different languages used.

In the last fifteen years, many advances in genetic programming have been related to the effort to overcome the closure problem (O'Neill and Ryan, 2003). In this context, Grammar-guided genetic programming (GGGP) has been used successfully (Rodrigues and Pozo, 2002). GE also plays an important role in this line of advances (O'Neill and Ryan, 2003). Like GGGP, it has the ability to produce only valid expressions in a general and parsimonious way, and manages arbitrary languages.

The GE defining features are (for a detailed explanation refer to (O'Neill and Ryan, 2003)):

- Individuals have two explicitly separate representations: genotype and phenotype. The former is a string of integers subject to the action of the evolutionary operators, the latter is written in the target programming language and is confronted to the fitness function.
- A predefined free-context grammar specifies the target language.
- A genotype-phenotype mapping transform the integer string in a *valid* expression of the programming language.

The main limitation, for our purposes, of the original GE system (as well as GGGP) is the use of context-free grammars. As we will see below, these algorithms based on context-free grammars have not enough expressive power to be used as an *automatic modelling tool*. However, recent extensions to GE make use of more powerful grammars (Ortega et al., 2007; Echeandia et al., 2005). We have employed Christiansen grammars (CG) (Ortega et al., 2007) an adaptable extension to attribute grammars (Knuth, 1968) whose first attribute is the CG that contains the rules applicable to each non-terminal in every derivation step. CG are adaptable because this attribute can change as words are generated. Like attribute grammars, CG can fully describe the solutions of any computable task, but their adaptability allows more comfortable approaches to some problems.

2 MODELLING METHOD

It is easy to show that there is no reason to restrict the set of languages to be used in CGE to the subset of *programming* languages. In fact, an evolutionary search can be done in any formal language which can be defined as a Christiansen grammar. Therefore, if we define the modelling task as finding an expression, written in a given language, which describes the behaviour of a real system, CGE can be understood as an *automatic modelling tool* whose grammar is the formalism in which the model is expressed and the fitness function determines the features of the target model. The original system GE has been used in previous works with the same general perspective: (O'Neill et al., 2002), economics and finance, or (Moore and Hahn, 2004), hierarchical Petri net modelling of complex genetic systems. As far as we know, this is the first time that CGE has been applied to a real problem.

With this in mind, our *modelling method* goes through the following steps:

- 1 **Grammar**: the grammar introduced in the CGE system determines the expressions landscape to be searched by the evolutionary algorithm. It implements the following aspects of the model.
 - (a) **General architecture of the model**: the formalism used to express the model.
 - (b) **Researcher's assumptions and bias**: due to theoretical reasons or model assumptions, some restrictions to the grammar may be introduced.
 - (c) **Other practical or technical constraints**.

2 **Fitness function:** used to assess the quality of the expressions generated by the algorithm. This function has to sort models according to their quality, to make comparisons possible.

3 Parameter tuning and search

4 **Interpretation and study of the model:** in some contexts, the fitness function may not provide enough information to conclude that the model found is appropriate. In other cases, the model found may fit the conditions imposed, beside bringing some other knowledge in terms of the way it performs the task, thus encouraging theoretical discussion.

We consider that this scheme can be applied to any modelling context. In our case, it has been used to model habituation.

2.1 Prior assumptions and constraints

To represent our models of habituation we have used the formalism and conditions of one of the latest habituation models (del Rosal et al., 2006), together with other practical considerations.

1. Since habituation features happen in time, the variables of the model must be functions of time. We have used the iterated functions formalism, where time is the independent variable. Iterated functions have the form $f(t+1) = g(f(t))$, with initial condition $f(0)$.
2. As this model is to be integrated with a more general model of associative learning, its architecture should have the ability to manage more than one stimulus and their interactions.
3. Each stimulus is represented by three variables:
 - $S_i(t)$ represents the intensity of stimulus i . Its value is a number in the $[0, 1]$ interval. Any value greater than zero indicates that the stimulus is present.
 - $A_i(t)$ represents the strength of the response of the organism to stimulus i .
 - $D_i(t)$ is an auxiliary variable.
4. For each pair of stimuli (i, j) , $T_{ij}(t)$ represents their interaction.
5. **Initial conditions:** we decided to maintain the same initial conditions as in (del Rosal et al., 2006). $\forall i, A_i(0) = T_{ij}(0) = 0$ and $D_i(0) = 1$.
6. **Complexity.** Since the model is to be used by the researcher as a representation of the mechanism that produces habituation, it must not be too complex, so that the researcher can manipulate it as a prediction tool and for theoretical discussions.

2.2 The Christiansen Grammar

Before presenting the grammar, we will explain its main features.

All the iterated functions are defined by means of clauses such as $A_i(t) = \langle \text{arithmetic-expression} \rangle$. For clarity, the time step specification remains implicit. The arithmetic expressions are written in prefix notation. The following operators can be combined in the expressions:

- Binary operators: **+**, **-**, *****, **/**, **max**, **min** and **pow** (with the usual meaning).
- Unary operators: **log**, **abs**, **int**, **sqrt** and **exp** (with the usual meaning).
- **sum<n>**: a summation operator with some constraints. It adds all the values in an array. For instance, the expression $sum_k, D_i, sum_l, A_k, A_l$, is equivalent in classic notation to $\sum_{k=0}^n D_i \sum_{l=0}^n A[k]A[l]$ (where n is the number of stimuli).

Variables may be subscripted by referring to a single stimulus, as in i or j , or to all of them, as in $\langle n \rangle$. The following are legal subscripted variables: $A_i, D_i, S_i, A_{\langle n \rangle}, D_{\langle n \rangle}, S_{\langle n \rangle}, T_{i\langle n \rangle}, T_{ij}$ and $T_{\langle n \rangle j}$.

It can easily be seen below that the grammar we have used is a context-free grammar, except for two elements, which require the use of the CG's capabilities:

- **Summation indices:** a variable with subscript index of type $\langle n \rangle$ does not make sense outside the scope of a summation operator. Our CG has rules to generate this kind of variables only when the non-terminal is inside the scope of a summation operator. This is a context-dependent property.
- **Model complexity:** as mentioned before, the complexity of the model should not be too large. The Christiansen grammar includes an attribute ($\uparrow c$) to store the number of operators in each expression. If this number gets larger than a given threshold, the expression is dismissed (function *tooComplex(c)* in the grammar).

We present the grammar below. Our notation follows (Watt and Madsen, 1977), where inherited attributes have down-arrows symbols (\downarrow) and synthesised attributes have up-arrows (\uparrow). ($\downarrow g$) stands for the Christiansen Grammar attribute. More details in (del Rosal, 2006).

- $\langle \text{Model} \rangle(g) ::= A_i(t) = \langle \text{ExpTypeAD} \rangle_A(\downarrow g) : D_i(t) = \langle \text{ExpTypeAD} \rangle_B(\downarrow g) : T_{ij}(t) = \langle \text{ExpTypeT} \rangle(\downarrow g)$

$$\{$$

$$\langle \text{ExpTypeAD} \rangle_A.\downarrow g = \langle \text{Model} \rangle.g$$

$$\langle \text{ExpTypeAD} \rangle_B.\downarrow g = \langle \text{Model} \rangle.g$$

$$\langle \text{ExpTypeT} \rangle.\downarrow g = \langle \text{Model} \rangle.g$$

$$\}$$

Below, rules for the non-terminal $\langle \text{ExpTypeAD} \rangle$:

- $\langle \text{ExpTypeAD} \rangle(\downarrow g, \uparrow c) ::= \langle \text{BinaryOp} \rangle(\downarrow g, \uparrow c)$

$$\{$$

$$\langle \text{BinaryOp} \rangle.\downarrow g = \langle \text{ExpTypeAD} \rangle.\downarrow g$$

$$\}$$

```

<ExpTypeAD>_A.↓g = <ExpTypeAD>.↓g
<ExpTypeAD>_B.↓g = <ExpTypeAD>.↓g
<ExpTypeAD>.↑c = <ExpTypeAD>_A.↑c +
<ExpTypeAD>_B.↑c + 1
tooComplex(<ExpTypeAD>.↑c)
}
• <ExpTypeAD>(↓g, ↑c) ::= <UnaryOp>(↓g),
<ExpTypeAD>_A(↓g, ↑c)
{
<UnaryOp>.↓g = <ExpTypeAD>.↓g
<ExpTypeAD>_A.↓g = <ExpTypeAD>.↓g
<ExpTypeAD>.↑c = <ExpTypeAD>_A.↑c + 1
tooComplex(<ExpTypeAD>.↑c)
}
• <ExpTypeAD>(↓g, ↑c) ::= sum<indexAD>(↓g, ↑new_g),
<ExpTypeAD>_A(↓g, ↑c)
{
<indexAD>.↓g = <ExpTypeAD>.↓g
<ExpTypeAD>_A.↓g = <indexAD>.↑new_g
<ExpTypeAD>.↑c = <ExpTypeAD>_A.↑c + 1
tooComplex(<ExpTypeAD>.↑c)
}

```

As we are inside the scope of a summation operator, new rules are include to allow indices of type $\langle n \rangle$.

```

• <indexAD>(↓g, ↑new_g) ::= a | b | c | d | e | f | g | h | k | m |
o | p | q | r | s | t | u | w | x | y | z
{
rule1 = <ExpTypeAD>(↓g) ::= A_{\langle n \rangle(\downarrow g)} | D_{\langle n \rangle(\downarrow g)}
| S_{\langle n \rangle(\downarrow g)} | T_{i \langle n \rangle(\downarrow g)}
{
<n>.↓g = <ExpTypeAD>.↓g
}
rule2 = <n> ::= "THE ACTUAL INDEX"
<indexAD>.↑new_g = <indexAD>.↓g + rule1 + rule2
}
• <ExpTypeAD>(↓g, ↑c) ::= <RealNumber>(↓g)
{
<RealNumber>.↓g = <ExpTypeAD>.↓g
<ExpTypeAD>.↑c = 0
}
• <ExpTypeAD>(↓g, ↑c) ::= S_i | A_i | D_i
{
<ExpTypeAD>.↑c = 0
}

```

Rules for $\langle \text{ExpTypeT} \rangle$ are equivalent to those for $\langle \text{ExpTypeD} \rangle$ except for the variables allowed. We omit equivalent rules:

```

• <indexT>(↓g, ↑new_g) ::= a | b | c | d | e | f | g | h | k | m | o
| p | q | r | s | t | u | w | x | y | z
{

```

```

rule1 = <ExpTypeT>(↓g) ::= A_{\langle n \rangle(\downarrow g)} | D_{\langle n \rangle(\downarrow g)} |
S_{\langle n \rangle(\downarrow g)} | T_{i \langle n \rangle(\downarrow g)} | T_{\langle n \rangle j(\downarrow g)}
{
<n>.↓g = <ExpTypeT>.↓g
}
rule2 = <n> ::= "THE ACTUAL INDEX"
<indexT>.↑new_g = <indexT>.↓g + rule1 + rule2
}
• <ExpTypeT>(↓g, ↑c) ::= S_i | A_i | D_i | S_j | A_j | D_j | T_{ij}
{
<ExpTypeT>.↑c = 0
}
• <BinaryOp> ::= + | - | * | / | max | min | pow
• <UnaryOp> ::= 1 | abs | int | sqrt | exp
• <RealNumber>(↓g) ::= <IntPart>(↓g).<DecPart>(↓g)
{
<IntPart>.↓g = <RealNumber>.↓g
<DecPart>.↓g = <RealNumber>.↓g
}
• <IntPart> ::= 0 | 1 | 2 | ..... | 8 | 9 | 1<IntPart>_A |
2<IntPart>_A | .... | 9<IntPart>_A
{
<IntPart>_A.↓g = <IntPart>.↓g
}
• <DecPart> ::= 0 | 1 | 2 | ..... | 8 | 9 | <DecPart>_A1 |
<DecPart>_A2 | .... | <DecPart>_A9
{
<DecPart>_A.↓g = <DecPart>.↓g
}

```

2.3 The fitness function

The fitness function compares our models to empiric data with respect to the characteristics mentioned in table 1. (Rankin and Broster, 1992), experiment 1, is used for properties 1, 2, 4 and 7. (Rankin et al., 1990) is used for property 5. The comparison is done in terms of the absolute differences between the points of the simulated and empiric curves. Therefore, the values go from zero to ∞ . The values are in terms of the percentage of the initial response to the stimulus, as usually in the psychological literature. For those models with negative values, their curves are scrolled up to avoid them, since negative values has no meaning as an activation value to a stimulus. The mathematical expression is:

$$Fitness = \sum_{i=0}^{i=n} \sum_{k=1}^{k=m} \left| \frac{P_i[k]}{P_i[0]} - \frac{E_i[k]}{E_i[0]} \right| * weight(i, k)$$

Leaving apart the *weight* function, E stands for the empiric points' vector. P symbolize the simulated points' vector. The subscript index "i" represents each experiment in the data set and "k" represents each point. Finally, "n" is the number of experiments, while "m" is the number of points of the given experiment.

The *weight* function increases the influence of points that are specially important. Every point has a *weight* of 1 by

default, except for a few points that strongly characterize habituation. These points are those that represent recovery in (Rankin and Broster, 1992), experiment 1, (weight = 3) and the one that reflects dishabituation in the experiment taken from (Rankin et al., 1990), (weight = 6). These values are somehow arbitrary and could be subject to further tuning.

3 EXPERIMENTS AND RESULTS

During our experiments we tried to find the best combination of evolutionary parameter values, in terms of their influence to reach better fitness values. Only a subset of them varied: mutation, cross-over and the generational gap (in a steady-state population model). The rest remain constant with the following values: *population* = 10.000, fixed *number of codons* = 1.000, *wrapping* = 3, *codon range* = [0,256], *parent selection* = "fitness proportional", and *survivor selection* = "replace worst scheme". The termination condition was the existence of a fitness below 1 (a very small number that reflects an almost perfect matching between the empiric and simulated data) or reaching the maximum number of generations, which is determined by the maximum number of fitness evaluations, an amount between 30000 and 120000 depending on the CPU performance. With this we tried to avoid very long experiments (a normal experiment lasted 4-6 days).

We tried to find the best parameter combination considering the following values: mutation rates of 10%, 50% or 100%, generational gaps of 1% and 50% and cross-over rates of 10%, 50% or 100%. The best combination of values was mutation=50%, cross-over=10% and GG=1%. We ran at least four experiments for each combination. It is worthy to note at this point that, in our program, mutation rate is not defined in the traditional way. Rather than applying mutation to every bit with a given probability, our 50% rate gives the probability that a single codon in the genotype will be mutated. This corresponds to a much smaller rate under the traditional interpretation.

This program of experiments gave two main results: a) most experiments ended with a big convergence in the population, b) the models with better fitness needed to be studied in detail, as they showed some of the features listed in table 1. They thus represent interesting novel models of habituation. We will look in depth at two of them. We have translated the models to a more usual notation. We have also omitted the $T_{ij}(t)$ function since is irrelevant in both models. The first is represented by the expression $A_i(t) = D_i : D_i(t) = S_i - (D_i * \ln 2.67)$. Figure 2 shows its behaviour during the habituation process (with the same conditions as in (Rankin and Broster, 1992), experiment 1, ISI 30). We can see how it reproduces the two main features of habituation: exponential response decrement and spontaneous recovery.

The second model $A_i(t) = D_i : D_i(t) = (\sum_k S_k * 4.7) - \max(D_i, \sqrt{\text{abs}(D_i / \sqrt{343.8})})$ was confronted to a habituation process followed by a tentative of dishabituation (same conditions as in (Rankin et al., 1990)), corresponding to features 1 and 5 in table 1. Dishabituation is a sudden recovery of the response after the presentation of a novel stimulus (second 397 in our case). Figure 3 shows how this model exhibits a basic exponential response decrement and disha-

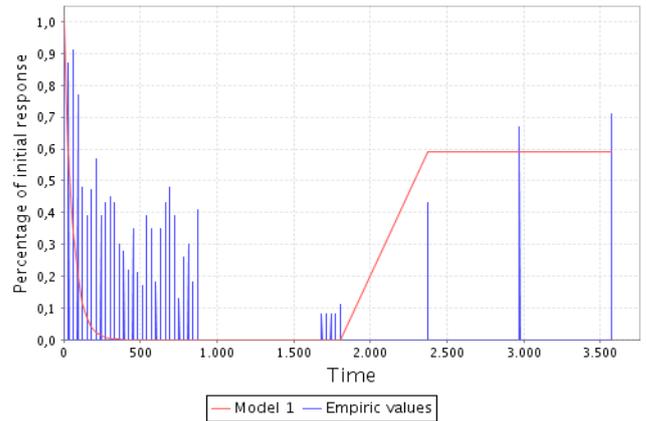


Figure 2: Empiric data Vs Model 1. Habituation and spontaneous recovery

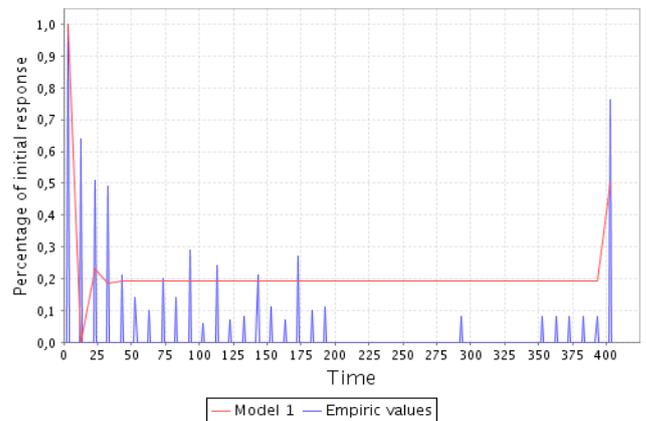


Figure 3: Empiric data Vs Model 2. Habituation and dishabituation

bituation. This last feature is specially important, since it has been neglected by all the existing models in the literature, except for (del Rosal et al., 2006).

4 DISCUSSION AND FURTHER WORK

The curves presented in this paper show that our models have, only partially, some of the characteristics that these models have to exhibit. This circumstance becomes evident when confronting to the fitness function our model and some of the literature: one of the last models of habituation Alonso et al. (2005) gets a value of 22.11, on the other hand, the best models we have found never gets a fitness value below 24. We have, therefore, further work to do in terms of parameter tuning and other elements that could improve the evolutionary automatic technique we have developed.

Some of the future improvements concern the maintenance of diversity throughout the experiment, which can be done by modifying the genetic algorithm running in the background of the CGE system with mechanisms to preserve diversity. Other improvements will affect the grammar: adding new semantic constraints, in the form of new variables and their features, might make it possible to find better models than those

in the literature, specially if these new conditions are based on knowledge from experts in the field of habituation. In fact, our grammar includes the assumptions and constraints of the latest models in the literature (del Rosal et al., 2006). A more flexible grammar (for instance, with no limitation on the number of variables) could also open new possibilities for novel models.

Nevertheless, the results prove that the methodology we have followed is capable of automatically building models of habituation with interesting features. This encourages us to continue refining the algorithm, and supports the idea that CGE can be used as an *evolutionary automatic modelling* tool, since it has been able to tackle, with a significant degree of success, a modelling problem as difficult as associative learning.

Acknowledgments

This work has been partially sponsored by the Spanish Ministry of Education and Science (MEC), project number TSI2005-08225-C07-06.

References

- Alonso, L., Moreno, R., Vázquez, M., del Rosal, E., and Santacreu, J. (2005). Spontaneous recovery of conditioned response in an autonomous agent. *Estudios de Psicología*, 26(3):365–376.
- Byrne, J. H. (1982). Analysis of synaptic depression contributing to habituation of gill-withdrawal reflex in *aplysia californica*. *Journal of Neurophysiology*, 48:431–438.
- del Rosal, E. (2006). Automatic modelling; grammatical evolution applied to the problem of modelling habituation in psychology. Master's thesis, Escuela Politecnica Superior. Universidad Autonoma de Madrid.
- del Rosal, E., Alonso, L., Moreno, R., Vázquez, M., and Santacreu, J. (2006). Simulation of habituation to simple and multiple stimuli. *Behavioural Processes*, 73:272–277.
- Echeandia, M. D., de la Puente, A. O., and Alfonseca, M. (2005). Attribute grammar evolution. *Artificial Intelligence and Knowledge Engineering Applications: a Bioinspired Approach, PT 2, Proceedings*, 3562:182–191.
- Hall, G. (1991). *Perceptual and associative learning*. Oxford: Clarendon.
- Knuth, D. E. (1968). *Mathematical Systems Theory*, volume 2, chapter Semantics of Context-Free Languages, pages 127–145.
- Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA.
- Mazur, J. E. (2002). *Learning and behavior*. Upper Saddle River (New Jersey). Prentice-Hall.
- Moore, J. H. and Hahn, L. W. (2004). An improved grammatical evolution strategy for hierarchical petri net modeling of complex genetic systems. *Applications of Evolutionary Computing*, 3005:63–72.
- O'Donohue, W. T. and Kitchener, R. (1999). *Handbook of behaviorism*. San Diego: Academic Press.
- O'Neill, M., Brabazon, A., and Ryan, C. (2002). *Genetic Algorithms and Genetic Programming in Economics and Finance*, chapter Forecasting market indices using evolutionary automatic programming. A case study. Kluwer Academic Publishers.
- O'Neill, M. and Ryan, C. (2003). *Grammatical Evolution. Evolutionary Automatic Programming in an Arbitrary Language*. Kluwer Academic Publishers.
- Ortega, A., de la Cruz, M., and Alfonseca, M. (2007). Christiansen grammar evolution: grammatical evolution with semantics. *IEEE Transactions on Evolutionary Computation*, 11-1:77–90.
- Pavlov, I. P. (1927). *Conditioned reflexes*. London: Oxford University Press.
- Rankin, C. H., Beck, C. D. O., and Chiba, C. M. (1990). *Caenorhabditis-elegans* a new model system for the study of learning and memory. *Behavioral Brain Research*, 37(1):89–92.
- Rankin, C. H. and Broster, B. S. (1992). Factors affecting habituation and recovery from habituation in the nematode *caenorhabditis-elegans*. *Behavioral Neuroscience*, 106(2):239–249.
- Rodrigues, E. and Pozo, A. (2002). Grammar-guided genetic programming and automatically defined functions. *Advances in Artificial Intelligence, Proceedings*, 2507:324–333.
- Staddon, J. E. R. and Higa, J. J. (1996). Multiple time scales in simple habituation. *Psychological Review*, 103(4):720–733.
- Stanley, J. C. (1976). Computer-simulation of a model of habituation. *Nature*, 261(5556):146–148.
- Thompson, R. F. and Spencer, W. A. (1966). Habituation: a model phenomenon for the study of neuronal substrates of behavior. *Psychological Review*, 73:16–43.
- Wang, D. L. (1994). A neural model of synaptic plasticity underlying short-term and long-term habituation. *Adaptive Behavior*, 2:111–129.
- Watt, D. A. and Madsen, O. L. (1977). Extended attribute grammars. Technical Report 10, University of Glasgow.