CONFLICT RESOLUTION BY RANDOM ESTIMATED COSTS

ROMAN V BELAVKIN

School of Computing Science, Middlesex University, London NW4 4BT, UK

Abstract: Conflict resolution is an important part of many intelligent systems such as production systems, planning tools and cognitive architectures. For example, the ACT–R cognitive architectire [Anderson and Lebiere, 1998] uses a powerful conflict resolution theory that allowed for modelling many characteristics of human decision making. The results of more recent works, however, pointed to the need of revisiting the conflict resolution theory of ACT–R to incorporate more dynamics. In the proposed theory the conflict is resolved using the estimates of the expected costs of production rules. The method has been implemented as a stand alone search program as well as an add–on to the ACT–R architecture replacing the standard mechanism. The method expresses more dynamic and adaptive behaviour. The performance of the algorithm shows that it can be successfully used as a search and optimisation technique.

keywods: conflict resolution, decision making, search, optimisation, rule–based systems, cognitive modelling.

1 INTRODUCTION

Many problems do not have a unique solution. Moreover, some problems may have infinitely large number of similar solution paths. A conflict occurs when several alternative decisions are available corresponding to different solution paths. Intelligent systems, such as rule-based systems, planning tools, cognitive architectures, rely on different strategies to resolve a conflict. The simplest method is a random or ordered choice of rules. Other strategies use recency or specificity of rules. More sophisticated methods can use statistical information about previous successes and failures of applying the rules to infer the probability of a success. In addition, some methods take into account costs of the rules, which represent the efforts (e.g time) required from the problem solver to perform the actions.

Statistical (Bayesian) methods proved to be very successful not only for a conflict resolution, but also for modelling some aspects of human behaviour. For example, the ACT–R cognitive architecture [Anderson and Lebiere, 1998] uses subsymbolic statistical information to choose a single production rule from a set of several rules matching the current goal (conflict set). ACT–R models have been successful in predicting many properties of human decision making, problem solving and learning. Despite the success, however, some recent works have pointed out several problems and limitations of the conflict resolution theory in ACT–R. These problems will be summarised in the first section of this paper.

The new method introduced in this paper relies on the same statistical information as in ACT–R, but uses it in a different way. The new method is more adaptable to a changing environment. Its dynamics is a consequence of the entropy reduction during problem solving. In addition, the method revises and unites several parameters in ACT–R. Many ideas were inspired by the progress in the theories of neural plasticity [Sejnowski, 1977a, 1977b; Bienenstock, Cooper, and Munro, 1982], as well as the information theoretic approach to cognitive models of decision making, learning and emotion [Belavkin and Ritter, 2003].

In this paper the theory and the algorithm will be presented in a general form, so that they could be applied to different domains. In the end of the paper a program demonstrating the method will be described, and its performance will be discussed. It will be suggested that the method can be used as a powerful optimisation and search technique.

2 CONFLICT RESOLUTION IN ACT–R

The ACT–R [Anderson and Lebiere, 1998] cognitive architecture uses a *utility* values U_i attached to every production rule i, and in the case of a conflict the rule with the highest U_i value is selected ($i = \arg\max U_i$). The utility of rule i is defined as

$$U_i = P_i G - C_i + \xi(\sigma^2)\,. \qquad (1)$$

Here P_i is the *expected probability* that the goal will be achieved after rule i has fired, and C_i is the average *cost* of the rule (average time required to achieve the goal). These rules–specific values are learned in ACT–R statistically using the records of past successes and failures as well as the efforts spent on each rule. Another two members of equation (1) are G — the *goal value* parameter (usually measured in time units), and $\xi(\sigma^2)$ — the *expected gain noise*, a random number derived from a normal distribution with zero mean and variance σ^2.

This conflict resolution scheme (1) allowed ACT–R to model many characteristics of human problem solving, such as probability matching and the effect of the pay–

Proceedings 17th European Simulation Multiconference
 ISBN 3-936150-25-7

off (reinforcement) on choice behaviour. Indeed, one can see from (1) that utility is a function of probability of success and the goal value.

Noise ξ in conflict resolution proved to be a good candidate for modelling different levels of expertise. It was shown in [Jones, Ritter, and Wood, 2000] that by increasing the noise variance σ^2 a model of an adult problem solver may begin to behave more like a young problem solver. Thus, learning may result in reduction of noise with time. It was suggested during the ACT–R workshop in August 2001 that such dynamics could potentially improve the fit of some models to data.

Moreover, it has been proposed that noise variance should follow the uncertainty of a success [Belavkin and Ritter, 2003], and its changes may play an important role for optimising the choice behaviour. In addition, it was shown that goal value G, if controlled dynamically, may optimise the expenditure of efforts [Belavkin, 2002]. Indeed, noise and goal value control breadth and depth of the search for a solution respectively.

Modern cognitive architectures have mechanisms not only for learning the statistics of existing rules, but also they may learn new rules (e.g. chunking in SOAR [Newell, 1990] or production compilation in ACT–R). It was proposed that noise ξ should be rule–specific and affect the new rules more than the 'older' rules in the system.

Unfortunately, this is not possible in the current conflict resolution mechanism because the goal value G and noise variance σ^2 are global parameters affecting all the rules simultaneously. Moreover, they are constants, and the theory does not explain their dynamics.

As we can see from the discussion above that being a successful theory of conflict resolution for some aspects of human problem solving, the current conflict resolution mechanism in ACT–R may not yet be complete. The new method introduced here is an attempt to overcome the described problems.

3 COST AND SUCCESS PROBABILITY

One may question the need of having the success probability P, as well as the goal value G and cost C in the utility equation (1). Let us consider the cost C of achieving the goal as a random variable, and let $P(C)$ be the probability that the goal will be achieved at the cost C (probability that the cost is exactly C). The expected value of the cost is

$$E\{C\} = \sum_C C\,P(C) \quad \left(E\{t\} = \int_0^\infty t\,P(t)\,\mathrm{d}t\right) ,$$

where the summation is made across all possible values of C (or an integral on $t \in [0, \infty]$ if C represents continuous time). The distribution function $P(C)$ gives the value of success probability for any cost. That is the expected probability P for given C or G in (1).

Knowledge of distribution functions $P(C)$ for different decisions would allow the problem solver to calculate their expected costs $E\{C\}$, and to choose the best rule (or strategy) to solve the problem. Of course the difficulty here is that when solving a problem for the first time nothing is known about $P(C)$. The only way to sample these distributions is by trying to solve the problem using different strategies. Moreover, even if we were determined to find out what the costs are by trial, we would soon realise that some costs are very hard to 'measure' directly.

For example, random rotation of the edges of a Rubik's cube may eventually assemble the puzzle, but the chance, as we say, is very low. More correctly would be to say that the probability of assembling the puzzle quickly is very low. This means that most likely the cost of such random rotation strategy is very high, and one would have to spend a lot of time waiting for the result. The question is when to give up and try another strategy?

The ability to give up on hopeless solutions without exploring them in full is a very important property of human problem solving. One of the important property of the algorithm that will be introduced is that it specifies exactly how deeply an alternative should be explored.

4 PROBLEM SOLVING AS AN OBSERVATION OF A POISSON PROCESS

Let us imagine that a computer solves some problem using a particular algorithm, and each time after the goal state has been achieved, the computer is restarted and is given to solve the same problem again (1).

Figure 1: A computer running an algorithm in a loop. The goal state is observed at a rate $\lambda = \frac{1}{E\{C\}}$, where $E\{C\}$ is the expected cost.

Now, if the expected cost of the solution that the computer is using is $E\{C\}$, then we shall observe the goal state every $E\{C\}$ seconds, or at a rate $\lambda = \frac{1}{E\{C\}}$. We may consider the occurrence of the goal state as a Poisson process. The probability of observing n events by the time t is

$$P(n \mid \lambda) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} , \qquad n = 0, 1, 2, \ldots . \quad (2)$$

Here λ is called the mean count rate ($\lambda = 1/E\{C\}$), and $n = 0, 1, 2, \ldots$ is the number of observations of the event by the time moment t.

Note, that for $\lambda = 0$ (or $E\{C\} = \infty$) the probability (2) becomes zero for any t, which corresponds to a case when the event is impossible. Thus, for an event to be possible the rate must be $\lambda > 0$ (or $E\{C\} < \infty$). Perhaps, when solving a problem, one must assume that the goal state is possible (be optimistic). That is $\exists G < \infty : E\{C\} \leq G$.

Now, let us consider some special cases of the Poisson probability (2).

Failure probability is the probability that the event will not occur ($n = 0$), and according to (2) it is

$$q(t) = P(n = 0) = e^{-\lambda t} . \quad (3)$$

The shape of the above function is shown by a declining dashed curve on Figure 2.

Success probability is the probability that the event will occur at least once ($n > 0$):

$$p(t) = P(n > 0) = 1 - q(t) = 1 - e^{-\lambda t} . \quad (4)$$

The success probability is shown on Figure 2 by an inclining dashed curve. One can see that it increases with time if $\lambda > 0$.

When solving a problem, especially for the first time, what we are interested in is the first occurrence of the goal state. Moreover, often we do not need to solve exactly the same problem again. Therefore, the probability of the very first success is of special interest.

First success probability that the event will occur exactly once ($n = 1$) is

$$p_1(t) = P(n = 1) = \lambda t e^{-\lambda t} . \quad (5)$$

The shape of the above function is shown by a solid curve on Figure 2. One may notice that it increases with time up to a certain maximum and then decreases again.

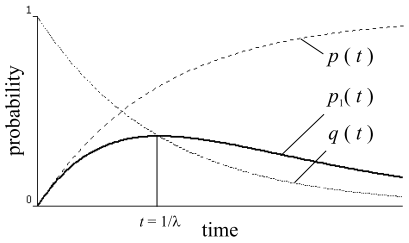

Figure 2: Probability of failure $q(t)$ decreases with time (declining dashed curve), probability of success $p(t)$ increases with time (inclining dashed curve). Probability of the first success $p_1(t)$ (solid curve) has a unique maximum in $t = 1/\lambda = E\{C\}$.

Let us find the time moment corresponding to the maximum of the first success probability:

$$\dot{p}_1(t) = \frac{\mathrm{d}}{\mathrm{d}t}\lambda t e^{-\lambda t} = 0 \quad \Rightarrow \quad t = \frac{1}{\lambda} .$$

We can see that this time moment corresponds to the expected cost $E\{C\}$ (the most likelihood cost). Note, that at this point the probability of the first success equals the probability of failure:

$$p_1(t) = q(t) = e^{-1} , \quad t = 1/\lambda .$$

It can be shown that for a system with two outcomes (first success and its complement) this is the moment of the maximum entropy, and hence it is the best moment to make a new estimate of λ using new information. If the new estimate turns out to be much greater than expected, then it may also be the optimal moment to change strategy or give up.

5 ESTIMATION OF THE EXPECTED COST

Up to this point we have been talking about problem solving as an observation of a Poisson process with known rate λ. Indeed, equation (2) describes the conditional probability $P(n \mid \lambda)$ of observing n events for a given value of λ (t is parameter). In reality, when solving a problem, the rate λ is unknown, and the expected cost is what we are trying to estimate. What is known is the number of successes n and the amount of time (or cost) that have been spent.

Let us estimate the rate λ (and, hence, $E\{C\}$) from the observed number of successes n after spending t amount of time. This can be done using posterior probability $P(\lambda \mid n)$, which can be obtained by the Bayes' formula

$$P(\lambda \mid n) = \frac{P(n, \lambda)}{P(n)} = \frac{P(n \mid \lambda)P(\lambda)}{P(n)} .$$

One can show that when *a priori* all the values of λ are equally probable, and the likelihood probability is described by the Poisson distribution (2), then the posterior probability can be found from the likelihood:

$$P(\lambda \mid n) = t\, P(n \mid \lambda) .$$

Now, using the above formula for the posterior probability, we can find the posterior mean estimate of λ:

$$E\{\lambda\} = \int_0^\infty \lambda P(\lambda \mid n)\, \mathrm{d}\lambda = \frac{n+1}{t} ,$$

and hence $E\{C\} \approx \frac{t}{n+1}$. Note that this estimation is biased towards successes (optimistic). Such an estimate is more useful than $\frac{t}{n}$ because it can be used even when no successes have yet been observed ($n = 0$). For this reason the algorithm was named OPTIMIST.

Interestingly, the OPTIMIST algorithm finds support in several studies on kinetics of choice and animal learning. Myerson and Miezin [1980] found that the response frequency in rats can be explained by the Poisson distribution [see also Mark and Gallistel, 1994].

6 RECURSIVE ESTIMATION

Let us return to the example of a computer solving a problem (Figure 1), but with one difference: the expected cost $E\{C\}$ is unknown. Also, this time we control when the computer is restarted. Our goal is to restart the computer in such a way, that the goal state appeared at the highest rate possible.

Let us denote by Δt the time interval after which we restart the computer. If we restart the computer too late $\Delta t > E\{C\}$, then obviously the rate at which the goal state occurs will not be the highest. On the other hand, if we restart the computer too early $\Delta t < E\{C\}$, then often the computer will not have enough time to finish solving the problem.

Let us conduct a series of trials registering the first occurrence of the goal state during time intervals Δt: if the goal state is registered, then we shall restart the computer immediately after it; otherwise, restart the computer after Δt. One may notice that there are only two possible outcomes of such trials (binomial trials):

Failure: the goal state has not been achieved, the number of successes n does not change, the overall effort (time) spent increases by $C = \Delta t$.

Success: the goal state is achieved, the number of successes n increases by one, and the effort increases by $C \leq \Delta t$.

By counting n number of successes and summing up the time spent $t = C_0 + ... + C_k$ in k trials, we can estimate the expected cost $E\{C\}$ using posterior mean formula:

$$E\{C\} \approx \bar{C} = \frac{t}{n+1} \,. \tag{6}$$

Now, starting with some small $\Delta t = C_{\min}$ let us set each next Δt equal to the last estimation of $E\{C\}$:

$$\Delta t_{k+1} = \bar{C}_k = \frac{\sum_{i=0}^{k} C_i}{n+1} \,.$$

An example of step by step calculation of $\bar{C}_i$ and Δt_{k+1} for ten trials ($k = 0, ..., 10$) is shown in Table 1. The dynamics of the estimated cost (6) during 20 trials is shown on Figure 3. With no successes registered ($n = 0$) the estimated value grows exponentially until it becomes greater than the expected cost $E\{C\}$, which means that the system spends enough efforts (time) to register first successes ($n = 1, 2, ... > 0$). As the number of successes n increases, the estimated value $\bar{C}$ decreases converging to the expected cost $E\{C\}$.

One can see that if the expected cost of the algorithm our computer is using is finite $E\{C\} < \infty$, then with trials the estimated cost $\bar{C}$, and hence the restarting time Δt, will converge to $E\{C\}$:

$$\lim_{k\to\infty} \Delta t_{k+1} = \lim_{k\to\infty} \bar{C}_k = E\{C\} \,.$$

Also, as a result, the goal state will occur at the highest rate possible.

Table 1: Example of estimation of $E\{C\}$ in 10 trials with failures in the first two and successes in the following eight trials.

k	n	$\bar{C}_k$	Δt_{k+1}
0	0	$C_{\min}$	C_0
1	0	Δt_1	$\bar{C}_1$
2	0	$\Delta t_1 + \Delta t_2$	$\bar{C}_2$
3	1	$\frac{\Delta t_1 + \Delta t_2 + \Delta t_3}{2}$	$\bar{C}_3$
...		...	...
10	8	$\frac{\Delta t_1 + ... + \Delta t_{10}}{9}$	$\bar{C}_{10}$

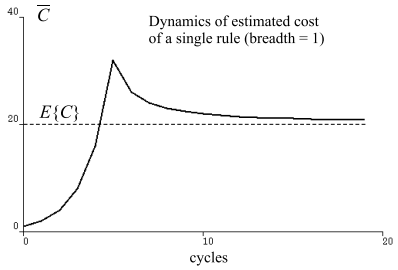

Figure 3: Estimated cost $\bar{C}$ converges to the expected cost $E\{C\}$ with cycles $k \to \infty$ and the number of successes $n > 0$.

7 CONFLICT RESOLUTION

In previous sections we discussed how to estimate the cost of one particular strategy (algorithm, decision or production rule) by estimating the rate of a hypothetical Poisson process. As an illustration we used the example of a computer set into an endless loop solving a problem (Figure 1). In a similar manner let us represent a conflict by a choice of several such computers tackling the same problem, but with only one computer used at a time.

Let us denote the options (choice of computers, strategies or rules) by x, and suppose that each computer uses different algorithm with a different expected cost $E\{C(x)\}$. Our goal is to find the fastest (cheapest) x.

Let us record for each option x the following information: $k(x)$ — the number of times x was used, $n(x)$ — the number of successes for x, $t(x) = C_0(x) + ... + C_{k(x)}(x)$ — the efforts (all time) spent using x. After trying each option we can estimate its expected cost $E\{C(x)\} \approx \bar{C}(x)$ using equation (6). In order to resolve the conflict we introduce a *random estimated cost*:

$$\tilde{C}(x) = \frac{k(x)\bar{C}(x) + \xi(\bar{C}(x))}{k(x)+1} \,. \tag{7}$$

Here $\xi(\bar{C}(x))$ is called a *random prediction*, and it is a random variable defined in such a way that its expected value equals the estimated cost $\bar{C}(x)$ ($E\{\xi\} = \bar{C}(x)$). For example, we can use the following function: $\xi(\bar{C}(x)) = \text{rand} \in (0, 2\bar{C}(x))$.

The conflict is resolved by selecting an alternative x with the smallest random estimated cost:

$$x = \arg\min \left[\tilde{C}(x)\right] \,.$$

8 PROPERTIES OF THE RANDOM ESTIMATE

One can see from (7) that the random estimated cost is a mixture of two components: the estimated cost $\bar{C}$ and the random predication ξ made based on the last estimated cost $\bar{C}$. The contribution of the latter component for individual rule x depends on the number of trials k (experience), and it decreases. This means that if new production rules are learned during problem solving, their expected cost will be more affected by the random prediction ξ.

The expected value of the random estimated cost $\tilde{C}$ equals the expected cost $E\{C\}$, which follows from its definition (7). Moreover, with trials k the value of the random estimated cost $\tilde{C}$ not only converges to the the expected cost $E\{C\}$, but also its value becomes more stabilised (less plastic) as the number of samples k increases. This property recalls of an important plasticity effect known for neural networks and explained by the *covariance* learning rule [Sejnowski, 1977a, 1977b; Bienenstock et al., 1982].

Because with successes the estimated cost $\bar{C}$ decreases, so does the expected value of the random ξ, as by definition $E\{\xi\} = \bar{C}$. This way, with successes, the random estimated cost $\tilde{C}$ decreases on average and becomes less random. We may compare this process with cooling the system down in simulated annealing [Kirkpatrick, Gelatt, and Vecchi, 1983].

On the contrary, if the number of failures increases, then the estimated cost $\bar{C}$ grows exponentially. As a result, the contribution of the random prediction ξ in (7) becomes more and more noticeable. This way, with failures, the random estimated cost $\tilde{C}$ increases on average and becomes more random. We may compare this process with heating the system up.

The information acquired during the binomial tests of rules acts as reward or penalty signals in reinforcement learning theories [Barto, 1985; Barto and Anandan, 1985]. Indeed, initially all the alternatives x have equal chances to be selected from the conflict set, because no posterior information is available, and the choice is random. On successes the estimated cost $\bar{C}$, and consequently the expected value of $\tilde{C}$, decreases. As a result, the chance of the successful rule to be selected on the next trial increases. This is similar to excitation effect in neural networks. On the contrary, if a failure occurs, then the estimated cost $\bar{C}$, and consequently the expected value of $\tilde{C}$, increases. This leads to inhibition of the failed alternative x, because its chance to be selected next time decreases.

9 METHOD PERFORMANCE

There are currently two applications in which the OPTIMIST algorithm has been tested: an add–on to the ACT–R architecture replacing the conflict resolution mechanism [see Belavkin, 2002], and a demonstration search program (all implemented in Common Lisp). The interface of the latter is shown on Figure 4. The program presents a search space with several gadgets controlling its parameters. The alternatives x representing different choice of strategy are located along the horizontal axis (breadth of the search space). The cost (depth of the search) is represented by the vertical axis.

When a particular rule x is selected, it is depicted by a vertical beam going up from the corresponding position (second alternative is shown selected on Figure 4). The height of the beam represents the current maximal cost (Δt). The outside world is represented by a distribution of the real costs. They are represented by thick horizontal bars. The distribution shape is controlled by the user, and it is not known to the algorithm. Figure 4 shows parabolic distribution of the real costs with the smallest (optimal) cost positioned in the middle. If the cost payed by the algorithm is enough to achieve the goal (the beam is higher than the real cost bar), then a success is registered (goal achieved); otherwise, a failure occurs. The horizontal line represents the latest estimation of the expected cost. The estimated costs for each alternative are stored in the memory of the program and are represented by thin horizontal bars.

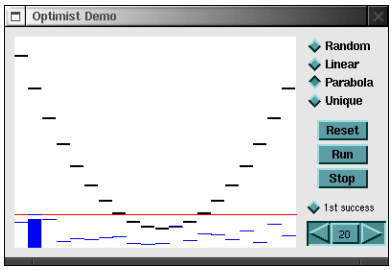

Figure 4: Interface of the OPTIMIST demonstration program

The program demonstrated the following behaviour of the algorithm with the four distinguishable stages. In the beginning the search is completely random and not very deep (the heights of the beams are small). When no successes are registered the estimated costs begin to grow exponentially (the beams begin to rise higher). The system 'heats up'. When the depth explored by the algorithm is greater than the real cost, the first successes occur, and the estimated cost decreases. For some period of time the number of successes is comparable to the number of failures, and the system appears as 'boiling'. When the algorithm finds more optimal solutions the system starts to cool down which is represented by a decrease of the estimated cost, and the choice is concentrated more on the successful entries with smaller (more optimal) costs. Finally, the system stabilises choosing only the optimal alternative and exploring just enough to reach the success. This stage can be compared with crystallisation.

If the distribution of the real costs changes after crys-

tallisation, the system heats up again until the algorithm finds another solution. This demonstrates the ability of the system to adapt to a changing environment. The speed of adaptation, however, decreases with the 'age' of the system.

Figure 5 shows from left to right the dynamics of choice proportion for parabolic distribution of the real costs with the optimal in the middle (breadth set to 50 alternatives). One can see that the breadth of the search decreases. The dynamics of the estimated cost is shown on Figure 6, and it illustrates the depth of the search as a function of trials (cycles).

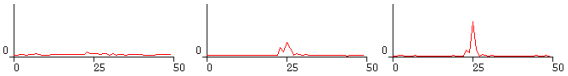

Figure 5: Dynamics of the choice proportion (from left to right).

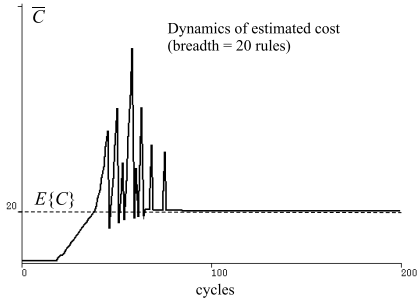

Figure 6: Dynamics of the estimated cost (optimal cost set to 20) for a conflict set of 20 alternatives (breadth 20).

The depth of search converges to the optimal. Moreover, the first solution found is not necessarily, but most likely to be the optimal. Indeed, the greater is the real cost of a solution, the less is its chance to be explored in full. On the contrary, the optimal solution path has the highest probability to be explored in full.

10 CONCLUSIONS

A new conflict resolution algorithm has been introduced, which united some of the parameters of the ACT–R cognitive architecture. The introduced learning and conflict resolution scheme addresses several problems of the ACT–R conflict resolution. It also implements other theories on kinetics of choice as a computational algorithm. In addition, the theory is general enough to be employed as a search and optimisation technique. The performance and cheap computational cost of the algorithm is encouraging for its application in various areas of computer science.

ACKNOWLEDGEMENTS

This work is sponsored by ESRC Credit and the ORS Award Scheme. Frank Ritter, David Elliman and David Wood provided useful comments and support for this work.

REFERENCES

Anderson, J. R., and Lebiere, C. 1998. *The atomic components of thought.* Mahwah, NJ: LEA.

Barto, A. G. 1985. "Learning by statistical cooperation of self–interested neuron–like computing elements." *Human Neurology*, *4*, 229–256.

Barto, A. G., and Anandan, P. 1985. "Pattern–recognizing stochastic learning automata." In *IEEE Transactions on Systems, Man and Cybernetics* (Vol. 15, pp. 360–375).

Belavkin, R. V. 2002. *On emotion, learning and uncertainty: A cognitive modelling approach.* PhD Thesis, The University of Nottingham, United Kingdom.

Belavkin, R. V., and Ritter, F. E. 2003. "The use of entropy for analysis and control of cognitive models." In F. Detje, D. Dörner, and H. Schaub (Eds.), *Proceedings of the Fifth International Conference on Cognitive Modelling* (pp. 21–26). Bamberg, Germany: Universitäts–Verlag Bamberg.

Bienenstock, E. L., Cooper, L. N., and Munro, P. W. 1982. "Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex." *Journal of Neuroscience*, *2*, 32–48.

Jones, G., Ritter, F. E., and Wood, D. J. 2000. "Using a cognitive architecture to examine what develops." *Psychological Science*, *11*(2), 93–100.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, J. M. P. 1983. "Optimization by simulated annealing." *Science*, *220*(4598), 671–680.

Mark, T. A., and Gallistel, C. R. 1994. "Kinetics of matching." *Journal of Experimental Psychology*, *20*(1), 79–95.

Myerson, J., and Miezin, F. M. 1980. "The kinetics of choice: An operant systems analysis." *Psychological Review*, *87*(2), 160–174.

Newell, A. 1990. *Unified theories of cognition.* Cambridge, Massachusetts: Harvard University Press.

Sejnowski, T. J. 1977a. "Statistical constraints on synaptic plasticity." *Journal of Mathematical Biology*, *69*, 385–389.

Sejnowski, T. J. 1977b. "Storing covariance with nonlinearly interacting neurons." *Journal of Mathematical Biology*, *4*, 303–321.

BIOGRAPHY

Roman Belavkin was born in Moscow in 1971. He graduated from Physics Department of the Moscow State University, but his interests switched from computer modelling of solar radiation to AI and cognitive modelling. He completed his PhD Thesis in 2002 in the University of Nottingham. He is currently based in Middlesex University in North London. His research is on cognitive modelling of the effects of emotion on decision making and learning. In his work he uses information theoretic approach for analysis of learning in cognitive models and architectures.