In *Proceedings of the The Sixth International Conference on Cognitive Modelling* (pp 40–45). Mahwah, NJ: Erlbaum, 2004.

OPTIMIST: A New Conflict Resoution Algorithm for ACT–R

Roman V. Belavkin (R.Belavkin@mdx.ac.uk)
School of Computing Science, Middlesex University
London NW4 4BT, United Kingdom

Frank E. Ritter (ritter@ist.psu.edu)
School of Information Sciences and Technology,
The Pennsylvania State University, University Park, PA16802, USA

Abstract

Several studies have suggested recently that a more dynamic conflict resolution mechanism in the ACT–R cognitive architecture (Anderson & Lebiere, 1998) could improve the decision–making behaviour of cognitive models. This part of ACT–R theory is revisited and a new solution is proposed. The new algorithm (OPTIMIST) has been implemented as an overlay to the ACT–R architecture, and can be used as an alternative mechanism. The operation of the new algorithm is tested in a model of the classical Yerkes and Dodson experiment on animals' learning. When OPTIMIST is used, the resulting model fits the data better than the previous model (e.g. R^2 increases from $.85$ to $.93$ in one example).

Introduction

Conflict resolution is an important part of many intelligent systems, and from a cognitive science perspective it represents a model of a decision–making mechanism in the brain. In this paper, we introduce a new conflict resolution algorithm that can be used as an alternative to the standard mechanism in the ACT–R cogntivie architecture (Anderson & Lebiere, 1998). The new algorithm is called OPTIMIST (it stands for 'Optimism' plus 'Optimisation'), and recently it has been introduced as a search method (Belavkin, 2003).

Although OPTIMIST can, indeed, be used as a general purpose search strategy, its roots come from ACT–R models of cognitive development (Jones, Ritter, & Wood, 2000) and the effect of emotion on learning and decision–making (Belavkin & Ritter, 2003). These works exposed where to improve the well–established cognitive architecture.

The standard conflict resolution mechanism of ACT–R, its achievements and problems will be discussed in the first section of the paper. Then, the underlying theory of the new method will be explained, and the new algorithm will be presented. This section will repeat some results of the previous paper (Belavkin, 2003). The third section will demonstrate how the new algorithm works in a model. Early results suggest that a model with the OPTIMIST conflict resolution matches the data better than with the standard implementation of ACT–R.

The ACT–R Conflict Resolution

The symbolic level of ACT–R is organised as a goal–directed production system with declarative and procedural knowledge encoded in the form of chunks and production rules respectively. The chunks representing the current goal, some facts currently retrieved from the long term memory, and the states of perceptual and action buffers are compared with the patterns in the left–hand sides of the production rules. Then, after a set of all the rules that match the current working memory pattern has been created (the conflict set), a single rule has to be selected from this set and fired. This last step is called *conflict resolution*, and it is important how this rule selection occurs because it controls which 'decisions' the model makes and affects the search of the problem space.

In ACT–R, the conflict resolution uses subsymbolic information associated with the rules. During the model run the number of successes and failures of each rule (decision) is recorded by the architecture. In addition, ACT–R records the efforts (e.g. time) spent after executing the rule and actually achieving the goal (or failing). This information is used to estimate empirically the probability of success P_i and the average cost C_i of each rule

$$P_i = \frac{\text{Successes}_i}{\text{Successes}_i + \text{Failures}_i} \quad (1)$$

$$C_i = \frac{\text{Efforts}_i}{\text{Successes}_i + \text{Failures}_i} \,. \quad (2)$$

Here, Efforts_i is the sum of all costs, associated with previous tests of the ith rule: $\text{Efforts}_i = \sum_{j=0}^{k} C_{ij}$, where $k = \text{Successes}_i + \text{Failures}_i$ is the number of previous tests of rule i. For example, if cost is measured in time units, then equation (2) calculates the average time for exploring particular decision path. This way, probabilities and costs of rules are learned by the architecture.

When several rules compete in the conflict set, ACT–R calculates their *utilities* by the following equation

$$U_i = P_i G - C_i + \xi(\sigma^2) \,. \quad (3)$$

Here, G is called the *goal value*, and it is measured in the same units as the cost (e.g. time); ξ is a random number taken from a normal distribution with zero mean and variance σ^2. Thus, the rational parts of the rules' utilities ($P_i G - C_i$) are corrupted by noise ξ of variance σ^2. Finally, the rule is selected according to utility maximisation: $i = \arg\max U_i$. Below is the summary of conflict resolution in ACT–R:

1. Set the goal value G and noise variance σ^2
2. Calculate P_i, C_i and $P_i G - C_i$ of the matched rules
3. Add noise $\xi(\sigma^2)$ to the utilities U_i
4. Fire rule $i = \arg\max U_i$

One can see that ACT–R learning equations (1) and (2) provide a kind of Bayesian estimation of rules' utilities. However, this mechanism only estimates the mean values ($P_i G - C_i$) of distributions from which the random utilities are drawn. The variances σ^2 remain the same for all rules and do not change. This issue will be addressed in the new mechanism.

The utility equation (3) has allowed ACT–R to model successfully some important properties of human and animal decision–making:

Probability matching. The choice in humans and animals decision–making is proportional to the probability of success. The use of P_i in the utility has allowed ACT–R to model the data of many probability matching experiments (e.g. see Anderson & Lebiere, 1998 for models on Friedman et al., 1964).

Stochasticity. The nondeterministic (irrational) property of choice behaviour is achieved by adding noise to the utility, and different variance σ^2 values were needed to simulate various experimental data (see Anderson & Lebiere, 1998). For example, the somewhat irrational behaviour of children could be simulated by a model of an adult with increased noise in conflict resolution (Jones et al., 2000). Moreover, it has been suggested that risk–taking behaviour characteristic to choice involving losses and negative emotions (Tversky & Kahneman, 1981; Johnson & Tversky, 1983) can be simulated by higher noise variance values, while low noise variance is better for simulating the risk–aversive behaviour associated with choice involving gains and positive emotions (Belavkin & Ritter, 2003).

Levels of stimulation. The reward (or penalty) values are known to influence choice. For example, higher pay–off leads to preferences towards decisions with higher success probabilities (Myers, Fort, Katz, & Suydam, 1963). This effect was modelled by using higher goal values G (Lovett & Anderson, 1995; Anderson & Lebiere, 1998). Also, it was shown that G can be used to represent different levels of aversive stimulation and even different levels of arousal (Belavkin & Ritter, 2003).

Recently, however, several problems in models' performance have been associated with the limitations of the ACT–R conflict resolution mechanism. In particular, it was noticed that ACT–R models usually produced more errors in the final stages of experiments than subjects. This effect was especially noticeable in models of tasks with incremental learning, such as the Tower of Nottingham (Jones et al., 2000) or the Yerkes and Dodson experiment (Belavkin & Ritter, 2003). Figure 1 shows such an example: The model matches the data well during the first five simulated days,[1] but produces more errors after day 5. Using smaller values of noise variance σ^2 could eliminate the problem, but would lead to a higher discrepancy with the data in the earlier stage of the curve. A similar lack of convergence was noticed by other researchers (Taatgen, 2001; Lebiere, 2003).

It has been suggested that noise variance σ^2 should not remain constant, but should gradually decrease. Taatgen used an exponential decay of σ^2 as a function of time and achieved better results. However, it was argued that noise variance should be an inverse function of success rate and should not

[1]Days A and B denote the preference series before training.

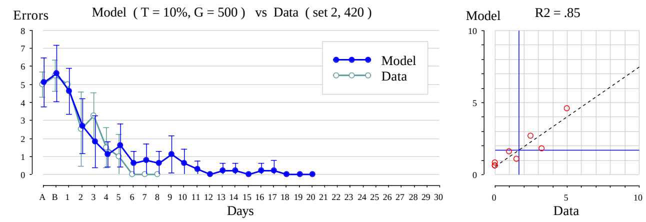

Figure 1: A model of the Yerkes and Dodson experiment compared with the data. Left: Error curves. Right: Regression plot of errors per day.

necessarily always decrease, but may increase if more failures occur (Belavkin, 2001). This would not only improve the models' match with the data, but also optimise the decision–making in a way similar to a simulated annealing heuristic. An alternative method was proposed to control noise variance by the entropy of success parameter (Belavkin & Ritter, 2003). Indeed, uncertainty of achieving success, estimated by the entropy, decreases as a result of learning, but may increase locally if more failures occur. The experiments demonstrated consistently that models with such a control matched the data better. In this interpretation, noise can be seen as compensation for missing information about the utilities of rules. The idea that noise should be proportional to the uncertainty (lower expertise) may explain also why children were simulated better by a model with higher noise variance (Jones et al., 2000).

The increase of expertise is not only reflected in the form of statistical information about the production rules. An ACT–R model may learn new rules using the production compilation mechanism. Taatgen proposed that noise should affect these new rules more than the 'older' rules in the system. This would provide a smooth transition in a model from the use of old to the more recently learned rules.[2] Again, this is not possible in the current ACT–R implementation, because σ^2 is a global parameter which does not depend on rules' creation times.

Another concern expressed is regarding the goal value parameter G, which is used as a constant in the current implementation. In the real–world situations, however, the value of the goal may change due to various reasons: Environmental change, re–evaluation of the efforts required, change of motivation due to boredom or anxiety and so on. Moreover, it was shown that G controls the problem space search strategy, and an increase of G from small to high values implements the best–first search heuristic (transition from breadth–first to depth–first) that can greatly optimise the search (Belavkin, 2001).

Unfortunately, the current ACT–R theory does not account for such dynamics. Let us summarise the new properties desirable for the conflict resolution algorithm:

1. Noise variance should be rule specific.
2. Noise variance should be inversely proportional to the rate of success, and should decrease on average with time.

[2]This effect has been achieved by using the production strength parameter and strength learning mechanism.

3. The goal value should be dynamic and increase on average.

In the next section, a new algorithm that implements the above properties is introduced.

The OPTIMIST Conflict Resolution

The OPTIMIST algorithm (Belavkin, 2003) has been derived in attempt to address the issues discussed in previous section. In the first part of this section, we present some theoretical background that helped derive the new algorithm, and in the second part, we describe the algorithm and its properties.

Theoretical background

It is well–known that many problems can have several solutions. Moreover, in the real world, applying even the same solution to one problem several times may produce slightly different outcomes. For example, using the same strategy to reach the goal in some task in several experiments may take slightly different amounts of time due to slightly different initial conditions or other uncontrolled factors in the environment. In view of this, it is natural to consider the cost C (e.g. time) needed to achieve the goal as a random variable, and the expected cost is thus

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

where $P(C)$ is the probability that the goal will be achieved exactly at cost C, and the summation is made across all possible values of C (or an integral on $t \in [0, \infty)$ if C is continuous, such as time). Note, that in this notation, probability distribution $P(C)$ (or probability density $\varphi(t)$) defines the probability of achieving success on any time interval on $[0, \infty)$, which is different from the ACT–R notation.

Knowledge of distribution functions $P_i(C)$ for different alternative decisions $i \in [1, \dots, N]$ would allow us to calculate their expected costs $E_i\{C\}$, and to choose the best rule. Indeed, better decisions should have smaller expected costs. For example, we can use several strategies to assemble a Rubik's cube puzzle. One such strategy can be a random rotation of edges of the cube, and it may eventually assemble the puzzle, but it will probably take much longer than by using some more sophisticated rules. Thus, one can choose the rule by minimising the expected cost: $i = \arg\min E_i\{C\}$ (optimisation).

The problem is, of course, that usually there is little information about $P_i(C)$, especially when making a decision for the first time, and in order to estimate the expected cost even for one decision one would have to apply this decision several times to get a sample estimate: $\bar{C} = \frac{1}{k}\sum_{j=1}^{k} C_j \approx E\{C\}$, where k is the number of tests.

We suggested to use Poisson distribution to approximate $P_i(C)$, if the cost is continuous, such as time (Belavkin, 2003). Indeed, if the expected cost of some strategy was known, then one could repetitively solve a problem by this strategy and would expect to observe the goal state at a rate $\lambda \equiv 1/\theta$, where $\theta \approx E\{C\}$ is the average waiting time. The probability of observing $n = 0, 1, 2, \dots$ number of successes by the time t in such a process is given by the Poisson distribution

$$P(n \mid \lambda) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} , \qquad n = 0, 1, 2, \dots . \tag{4}$$

Here, $\lambda = 1/\theta$ is called the *mean count rate*. Note, that for $\lambda \to 0$ (or $\theta \to \infty$) the probability (4) becomes zero. The corresponding waiting times until the success events (costs) are distributed according to Gamma distribution with mean $\mu = n\theta$ and variance $\sigma^2 = n\theta^2$. Thus, a strategy with a higher success rate λ should not only have a smaller expected cost, but also the variance of costs is smaller.[3]

Equation (4) describes the conditional probability of n successes on time interval $[0, t]$ for a known λ. However, in our case λ is unknown, and the expected cost $E\{C\} \approx \theta$ is what we are trying to estimate after observing $n = 0, 1, \dots$ successes on time interval $[0, t]$. This can be done using posterior probability density $\varphi(\lambda \mid n)$, which can be obtained from Bayes' formula

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

In the worst case scenario when no information about prior $\varphi(\lambda)$ is available, we should assume that all λ are equally probable (the maximum entropy principle). We can use the following exponential function $\varphi_\varepsilon(\lambda) = \varepsilon e^{-\varepsilon\lambda}$ with $\varepsilon \to 0$ for such an unbiased estimate. Using this prior and Poisson likelihood $P(n \mid \lambda)$, one can show that

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

Note, that after some finite period of time, if no successes have been observed, the distribution of λ becomes a decreasing function with high rates having smaller probabilities. Now, the posterior mean estimate of λ is:

$$E\{\lambda\} = \int_0^\infty \lambda\,\varphi(\lambda \mid n)\,\mathrm{d}\lambda = \frac{n+1}{t} \quad \left(E\{C\} \approx \frac{t}{n+1}\right)$$

Here, t and $n + 1$ correspond to the Efforts_i and Successes_i parameters in ACT–R equations (1) and (2).[4] Note that we can use the above estimate even when $n = 0$ (i.e. when no successes have occurred). This property is very important, because it means that we do not have to explore all the solution paths in full trying to succeed. Indeed, in our probabilistic interpretation of cost, any decision or strategy may eventually lead to the desired goal (optimistic approach), although the chance may be very small. An illustration of this idea can be the classical example from the probability theory of a monkey randomly typing on a keyboard. The probability that it will come up with a literature text, such as *War and Peace*, is in fact non–zero. Therefore, it is desirable for such 'impractical' decision paths to be explored only partially, and after accepting a failure ($n = 0$) the system should give up and try another decision (or strategy).

[3] We use the term *success rate* to describe the number of successes per time, and term *success ratio* as the number of successes per attempts.

[4] In ACT–R, $\text{Successes}_i = 1, 2, \dots = n + 1$ with 1 being the default value.

We showed using the maximum entropy principle that the optimal moment to make new estimation of the expected cost and its posterior probability density $\varphi(\theta \mid n)$ is at $t = \theta \approx E\{C\}$ (see Belavkin, 2003). If after the new estimation another rule has smaller expected cost, then this also would be the best moment to give up and try another alternative. The following recursive procedure can be used to estimate $E\{C\}$ of one decision

$$\Delta t_0 = C_{\min}\ , \quad \Delta t_{k+1} = \theta_k = \frac{\sum_{i=0}^{k} C_i}{n+1}\ .$$

Here, k is the cycle number, and Δt_k is the time (or cost) interval, on which, after the decision has been made, we expect to achieve a success. If success does not occur before the end of Δt_k, then a failure is accepted. The number of successes (n) in this case does not change, but the efforts (t) increase by $C_k = \Delta t_k$. Thus, on failures the estimate increases. If the success occurs before Δt_k, then n increases by one, and efforts increase by $C_k < \Delta t_k$. Thus, on successes the estimate decreases. Figure 2 shows an example of $E\{C\}$ estimation over 20 cycles: After increasing above the $E\{C\}$ level, its estimate quickly converges to $E\{C\}$.

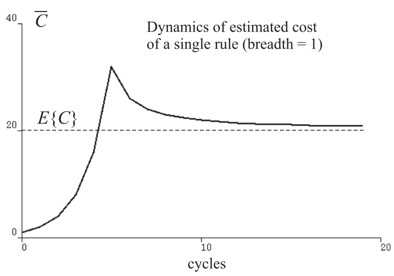

Figure 2: Estimated cost of one rule (vertical axis) as a function of test cycles (horizontal axis). Estimated cost converges to the expected cost $E\{C\}$ with cycles $k \to \infty$.

Now, if several alternative decisions (rules) are being considered, the choice can be made by selecting the decision with the smallest estimate.

Algorithm Description

The OPTIMIST algorithm uses the same subsymbolic information as that of the standard ACT–R implementation — the number of successes and the overall efforts associated with each rule. However, instead of calculating probabilities P_i and average costs C_i (equations (1) and (2)), OPTIMIST estimates the expected costs of achieving the success by each rule

$$\theta_i = \frac{\text{Efforts}_i}{\text{Successes}_i} \approx E_i\{C\}\ . \tag{5}$$

Next, the estimates θ_i of all the rules in the conflict set are replaced by random numbers ξ_i, which we call *random estimated costs*. Ideally, ξ_i should be drawn from Gamma distributions with parameters θ_i that have just been estimated. As has been mentioned earlier, these distributions have means $\mu_i = \theta_i$ and variances $\sigma_i^2 = \theta_i^2$ (we are expecting only the fist success, or $n = 1$). Thus, ξ_i represent a sample approximating the distribution of expected costs for several rules in the conflict set. Finally, the rule choice is made by minimisation of random estimated costs:

$$i = \arg\min \xi_i\ .$$

Below is the summary of the OPTIMIST algorithm:

1. Calculate the estimates θ_i of rules' expected costs
2. Replace θ_i by corresponding random ξ_i
3. Fire rule $i = \arg\min \xi_i$

For computationally efficiency, instead of Gamma distributions, we used the following uniform distributions to generate the random estimated costs

$$\varphi(\xi_i) = \begin{cases} \frac{1}{2\theta_i} & \text{if } |\xi_i - \theta_i| < \theta_i \\ 0 & \text{otherwise} \end{cases}\ . \tag{6}$$

These distributions have means $\mu_i = \theta_i$ and variances $\sigma^2 = \theta^2/3$. In addition, the algorithm can be made sensitive to the successes per attempts ratio by using the following functions to generate ξ_i:

$$\xi_i = \frac{k_i\,\theta_i + \text{rand}(2\theta_i)}{k_i + 1}\ , \tag{7}$$

where $k_i = \text{Successes}_i + \text{Failures}_i$ is the number of attempts.

Note that the algorithm does not use the goal value parameter G. However, to some extent it is identical to the expected cost estimation θ (or more precisely its minimum $\min\theta_i$). Thus, the goal value in OPTIMIST is dynamically learned through equation (5). Figure 3 shows the dynamics of $\min\theta_i$ for twenty rules in an example conflict set.

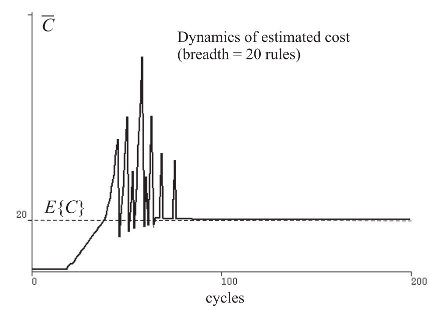

Figure 3: Dynamics of the smallest estimated cost for a conflict set of 20 rules as a function of test cycles.

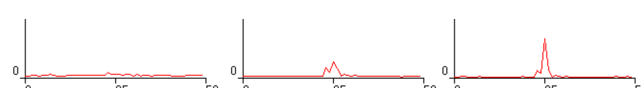

Figure 4: Dynamics of choice proportion (vertical axis) for different rules (horizontal axis) as a function of time. From left to right: The choice concentrates on more successful rules.

Also, one has been mentioned earlier, θ_i controls the variance of the costs' distributions. Thus, the noise variance in the conflict set also increases on failures and decreases on successes. In addition, the variance may also decrease with k_i — the number of times a rule has been used (see Eq. 7). Figure 4 shows from left to right the dynamics of choice proportion between fifty rules (horizontal axis), with the best rule placed in the middle. One can see that the choice quickly concentrates on the best rule.

Finally, because both θ_i and k_i are rule specific parameters, the randomness is different for all the rules in the system. In general, the less successful rules in the system (smaller rates $1/\theta_i$) as well as newer rules (smaller k_i) are more 'noisy' than rules with higher success rates and rules used more frequently. One can see from above that OPTIMIST possesses all three desired properties stated in the previous section: Noise variance is rule specific, dynamic and proportional to the success rate; Goal value is also dynamic and increases on average. Table 1 provides a comparison between the current ACT–R mechanism and OPTIMIST.

Table 1: Comparison of ACT–R and OPTIMIST conflict resolution mechanisms.

	ACT–R	OPTIMIST
n. of successes	$n_i + 1$	$n_i + 1$
n. of attempts	k_i	k_i
efforts spent	t_i	t_i
success probability	$P_i = \frac{n_i+1}{k_i}$	$P_i(n_i \mid \lambda_i)$
expected cost	$C_i = \frac{t_i}{k_i}$	$\theta = \frac{t_i}{n_i+1}$
goal value	$G = \text{const}$	$\min \theta_i$
noise variance	$\sigma^2 = \text{const}$	$\sigma_i^2 \sim \theta_i^2$
utility	$U_i = P_i G - C_i$	θ_i
conflict resolution	$\max U_i$	$\min \theta_i$

A Model Example and Additional Parameters

The OPTIMIST conflict resolution algorithm was put into a test in a model of the Yerkes and Dodson experiment. In this classical animal learning task, mice were trained over several days to escape a discrimination chamber (a box with two doors) through one particular door. Ten tests per day were performed with each mouse, and the number of errors was recorded. Figure 5 shows one example of distribution of errors, produced by the model with the OPTIMIST algorithm and compared with the experimental data. Horizontal axis represents the day numbers, and vertical axis shows the number of errors per day.

These first tests demonstrated that the new algorithm works, and the model produces behaviour comparable to both the data and a model with the standard conflict resolution mechanism. Moreover, because noise variance in OPTIMIST version has decreasing dynamics, the models with the new algorithm do not suffer from lack of convergence discussed earlier in the paper (see Figure 1). In fact, when using the weighted average version of the algorithm (7), the convergence is sometimes too fast. Therefore, a parameter has been introduced to enable OPTIMIST to retain some level of noise.

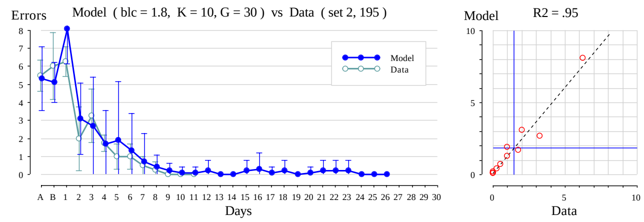

Figure 5: A model with OPTIMIST conflict resolution compared with experimental data (Yerkes & Dodson, 1908). Left: Error curves. Right: Regression plot of errors per day.

In the current implementation, this is achieved by limiting the number k used in equation (7).

Another adjustment to the algorithm concerns different levels of stimulation. In ACT–R, different values of pay–off can be represented by the goal value parameter G. In OPTIMIST, if the cost is only measured by time, then there is no way of distinguishing between different levels of a pay–off (i.e. values of reward or penalty). Indeed, the time spent on choosing an option with a prize worth \$10 is the same as for \$100. In order to account for these effects, the current OPTIMIST implementation uses reinforcement mechanism, which can modulate the costs of particular outcomes:

- If a rule fired has explicit `:failure` flag, then *penalty* value increases the cost of of the outcome and hence increases the expected cost of the rule associated with the failure.
- If a rule fired has `:success` flag, then the cost of the outcome is reduced by the *reward* amount.

The values of penalty and reward are defined by the corresponding variables in the system, and in fact they describe characteristics of the environment and interaction of a cognitive model with the environment, rather than internal state of the model. Moreover, this implementation allows a modeller to define several different rewards and penalties in various places of the simulated environment (unlike global G).

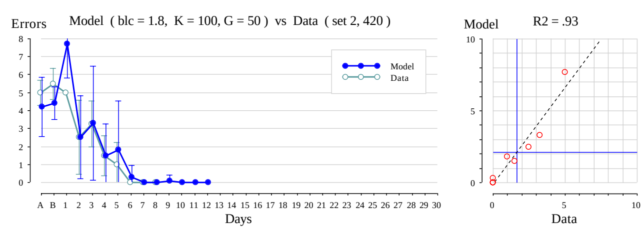

Figure 6: The effect of reinforcement: A model and data for an experiment with higher level of stimulation (Yerkes & Dodson, 1908).

Figure 6 shows the results of a model of the Yerkes and Dodson experiment with higher value of stimulation (in the original experiment it was an electrical stimulus). The model uses penalty value that modulates the costs of rules that are choosing the wrong door leading to an error. As a result, the model learns faster. Note that the data set the model is compared with is the same as shown on Figure 1. One can see that the OPTIMIST model matches the data better than the model with standard mechanism of ACT–R version 5 (R^2 has increased from .85 to .93), which indicates in favour of the new algorithm.

Discussion and Conclusions

In this paper, we challenged one of the most important mechanisms of a well–established cognitive architecture — the conflict resolution of ACT–R. Many cognitive models in ACT–R rely on and use the utility equation (3) and its parameters. One of such models, mentioned in this paper, is the model of the Yerkes and Dodson experiment. The limitations of the current ACT–R implementation, as encountered by this model, has become the main motivation for the new algorithm. For example, we have shown that a model with dynamic control of noise variance by means of entropy reduction improves significantly the match between the model and data (Belavkin & Ritter, 2003). As has been discussed earlier in this paper, similar concerns have been expressed by other researchers.

The new algorithm uses some elements of statistical decision–making theory and estimates expected costs of production rules using a Poisson distribution. Interestingly, several studies on kinetics of choice in animals learning have suggested earlier that estimation of the Poisson rate λ (or equivalently $\theta \equiv 1/\lambda$) may explain animals' choice behaviour. In particular, Myerson and Miezin (1980) used a Poisson distribution to explain the change of response frequency in rats (see also Mark & Gallistel, 1994). Moreover, an attempt to incorporate this into the ACT–R theory has been already made in a form of the *events discounting* mechanism (Lovett & Anderson, 1996; Lovett, 1998). Unfortunately, this mechanism suffers from computational overhead and turns out to be impractical for complex models. The new algorithm, introduced in this paper, directly estimates the rate of a Poisson process. In addition, the algorithm is computationally efficient and uses the standard subsymbolic information of ACT–R, so it is relatively easy to implement.

The first implementation of the algorithm as an overlay to ACT–R has been created, and it is available for download on

`http://gold.mdx.ac.uk/~rvb/software/optimist/`

The operation of the algorithm has been demonstrated on a model. Early results are in favour of the new algorithm and suggest that it indeed may improve the match of some cognitive models to data. However, more tests in different models and on other data sets still have to be done. It is, therefore, suggested to use the new algorithm in addition to the standard to provide valuable feedback for further development.

References

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.

Belavkin, R. V. (2001, March). The role of emotion in problem solving. In C. Johnson (Ed.), *Proceedings of the AISB'01 Symposium on Emotion, Cognition and Affective Computing* (pp. 49–57). Heslington, York, England: AISB. (ISBN 1-902956-19-7)

Belavkin, R. V. (2003). Conflict resolution by random estimated costs. In D. Al-Dabass (Ed.), *Proceedings of the 17th European Simulation Multiconference (ESM2003)* (pp. 105–110). Nottingham, UK. (ISBN 3-936150-25-7)

Belavkin, R. V., & Ritter, F. E. (2003, April). The use of entropy for analysis and control of cognitive models. In F. Detje, D. Dörner, & H. Schaub (Eds.), *Proceedings of the Fifth International Conference on Cognitive Modelling* (pp. 21–26). Bamberg, Germany: Universitäts–Verlag Bamberg. (ISBN 3-933463-15-7)

Friedman, M. P., Burke, C. J., Cole, M., Keller, L., Millward, R. B., & Estes, W. K. (1964). Two–choice behaviour under extended training with shifting probabilities of reinforcement. In R. C. Atkinson (Ed.), *Studies in mathematical psychology* (pp. 250–316). Stanford, CA: Stanford University Press.

Johnson, E., & Tversky, A. (1983). Affect, generalization, and the perception of risk. *Journal of Personality and Social Psychology, 45*, 20–31.

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

Lebiere, C. (2003, April). *Personal communication.*

Lovett, M. C. (1998). Choice. In *The atomic components of thought* (pp. 255–296). Mahwah, NJ: Lawrence Erlbaum.

Lovett, M. C., & Anderson, J. A. (1995). Making heads or tails out of selecting problem solving strategies. In *Proceedings of the 17th Annual Conference of the Cognitive Science Society* (pp. 265–270). Hillsdale, NJ: Lawrence Erlbaum.

Lovett, M. C., & Anderson, J. A. (1996). History of success and current context in problem solving: Combined influences on operator selection. *Cognitive Psychology, 31*, 168–217.

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

Myers, J. L., Fort, J. G., Katz, L., & Suydam, M. M. (1963). Differential monetary gains and losses and event probability in a two–choice situation. *Journal of Experimental Psychology, 77*, 453–359.

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

Taatgen, N. A. (2001, July). Production compilation. In *Eigths annual post–graduate summer school*. Retrieved from `act-r.psy.cmu.edu`.

Tversky, A., & Kahneman, D. (1981). The framing of decisions and the psychology of choice. *Science, 211*, 453–458.

Yerkes, R. M., & Dodson, J. D. (1908). The relation of strength of stimulus to rapidity of habit formation. *Journal of Comparative Neurology and Psychology*, *18*, 459–482.