OPTIMIST Conflict Resolution Overlay for the ACT–R Cogntivie Architecture[*]

Roman V. Belavkin
Middlesex University, School of Computing Science
London, NW4 4BT, United Kingdom

June 17, 2005

Abstract

This is a documentation on the OPTIMIST overlay for the ACT–R cognitive architecture (Anderson & Lebiere, 1998). The overlay implements an alternative conflict resolution algorithm. The main difference of the new algorithm from the 'standard' is the use of Gamma distributed noise, which has rule–specific and dynamic variance as opposed to normal distributed noise with global and constant variance.

Introduction

ACT–R (Anderson & Lebiere, 1998) is a general purpose hybrid cognitive architecture for developing cognitive models that can vary from simple reaction tasks to simulations of pilots navigating airplanes and operators of airtraffic control systems. ACT–R follows the approach of unified theories of cognition (Newell, 1990), in which several theories about different aspects of cognition are used in a single simulation system. Today, ACT–R has emerged as the architecture of choice for many cognitive modelling problems.

The symbolic level of ACT–R is organised as a goal–directed production system with declarative and procedural types of knowledge encoded in the form of chunks and production rules respectively. The subsymbolic level accounts for fuzzy or probabilistic properties of cognition, and it uses activation values, utilities, associations and other parameters to control the way symbolic knowledge is used. The dynamics of these parameters is described by a set of equations, that arise from neuroscience and cognitive psychology theories.

Conflict resolution is an important part of the subsymbolic level of ACT–R, and it represents a model of the decision–making mechanism. Several studies have suggested recently that a more dynamic conflict resolution mechanism in the architecture could significantly improve the decision–making behaviour of cognitive models. The current mechanism of ACT-R has been revised and a new algorithm has been proposed (see Belavkin, 2003; Belavkin & Ritter, 2004). The new algorithm is called

[*]The OPTIMIST overlay has been initially developed at University of Nottingham and then at the Middlesex University in London, United Kingdom.

OPTIMIST (stands for 'Optimism' plus 'Optimisation') and it has been implemented as an overlay to the ACT–R architecture. Thus, it can easily be used as an alternative mechanism. This document explains how to load and use OPTIMIST with the ACT–R cognitive architecture. For the motivations and theory behind the OPTIMIST algorithm see Belavkin (2003); Belavkin and Ritter (2004).

1 Conflict Resolution

After a set of all the rules matching the current working memory pattern has been created (the conflict set), a single rule has to be selected from this set and fired. This 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

$$P_i = \frac{\text{Successes}_i}{\text{Successes}_i + \text{Failures}_i} \tag{1}$$

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

Here, Efforts$_i$ is the sum of all costs, associated with previous tests of the ith rule: Efforts$_i = \sum_{j=0}^{k} C_{ij}$, where $k =$ Successes$_i +$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

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

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: $i = \arg\max U_i$.

The main motivation for the new conflict resolution are as follows:

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.
3. The algorithm should allow for different reinforcements from the environment.

The main idea behind the OPTIMIST algorithm is an assumption that the time intervals between the successes can be described 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 production rule with a higher success rate λ should not only have a smaller expected cost, but also the variance of costs is smaller.

The algorithm uses posterior estimation of the mean waiting times $\theta_i \equiv 1/\lambda_i$ for each rule in the conflict set:

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

Here, t and $n + 1$ correspond to the Efforts$_i$ and Successes$_i$ parameters in ACT–R equations (1) and (2). Note that the mean waiting times θ_i are not equivalent to the average costs C_i in ACT–R ($\theta_i \geq C_i$). Usually, more successful rules would have smaller θ_i than those of less successful rules.

The mean waiting times θ_i are used to generate a Gamma distributed noise $\xi(\theta_i)$. This is done in the following way. Once $\lambda \equiv 1/\theta$ has been estimated, the probability of failure according to (4) is $q = P(0 \mid \lambda) = e^{-\lambda}$, and probability of success is $p = 1 - e^{-\lambda}$. The inverse pdf of success is $F^{-1} = -\theta \ln(1 - p)$. Probability p is generated using uniform distribution on $(0, 1)$, and the random θ is computed using the inverse pdf.

The OPTIMIST conflict resolution uses the following utilities:

$$U_i = P_i G - \chi(\theta_i) + \xi(\sigma^2) , \tag{5}$$

where $\chi(\theta_i)$ are Gamma distributed random numbers. All other parameters are the same as in equation (3). Parameters G and σ^2 are used for compatibility, and the OPTIMIST algorithm should work when $G = 0$ and $\sigma^2 = 0$. Goal value G can be used to reduce globally the effect of the Gamma noise.

2 Reinforcement Mechanism

Time is not the only objective that should be considered by the utility. Indeed, for example, time spent on choosing an option with a prize worth $10 can be the same as for $100. In order to account for such 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 they can describe characteristics of the environment. The level of reinforcement may change during the interaction of a cognitive model with the environment. This implementation allows a modeller to define several different rewards and penalties in various places of the simulated environment. This is different from the parameter G, which is global affecting all the rules in the model.

Note that only rules with the flags `:failure` or `:success` set to T can be reinforced.

3 Loading OPTIMIST

There are two versions of the overlay: One for ACT–R Version 4 and one for Version 5 (files `optimist-for-act4.lisp` and `optimist-for-act5.lisp` respectively). To use the OPTIMIST algorithm, load the required file after ACT–R:

```
(load ACTR.lisp)
(load optimist-for-actr*.lisp)
```

4 Parameters

There are four additional global parameters defined in the OPTIMIST. The values of these parameters can be set in the model or by a simulation during the model run. Below is the description of the parameters.

```
*optimist*
```

This parameter turns the OPTIMIST algorithm on and off. Default value is `T`. If `NIL`, the standard ACT–R mechanism is used.

```
*minimal-cost*
```

Sets the prior estimate of θ. It is equal to the `*default-action-time*` by default (i.e. 50 milliseconds).

```
*reward*
```

The amount of positive reinforcement of a rule with a `:success` flag. Default value is `NIL`. If a number, then this value is subtracted from the efforts on completion of the goal set by the rule.

```
*penalty*
```

The amount of negative reinforcement of a rule with a `:failure` flag. Default value is `NIL`. If a number, then this value is added to the efforts on completion of the goal set by the rule.

5 Example Model

File `cr-test-model.lisp` provides an example model using the OPTIMIST overlay. This model works with both ACT–R Versions 4 and 5. It contains five production rules:

```
Do-Again
Good-Rule
Bad-Rule
Goal-Success
Goal-Failure
```

The first rule sets a goal that is matched by two rules: `Good-Rule` and `Bad-Rule`. These rules modify the goal in a slightly different way. The remaining two rules are fired depending on whether the goal has been modified by a `Good` or `Bad-Rule`. The goal is removed off the goal stack with either a success or a failure result. The parameters (i.e. numbers of successes, failures and the efforts) of the `Good-Rule` or the `Bad-Rule` are updated by the parameters learning mechanism of ACT–R. The `Do-Again` rule will repeat the process.

There is only one conflict in this model: Between the `Good` and `Bad-Rule`. There is no symbolic difference between these rules, and thanks to the subsymbolic learning and the conflict resolution, the model quickly learns the preference to use the `Good-Rule`. However, how quickly this preference is formed depends on several parameters: Goal value G and the amount of reinforcement (i.e. `*reward*` or `*penalty*`).

6 Feedback and Support

All the files and documentation can be accessed from

`http://www.cs.mdx.ac.uk/staffpages/rvb/software/optimist/`

Feedback and suggestions can be directed to `R.Belavkin@mdx.ac.uk`

References

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

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. (2004). Optimist: A new conflict resolution algorithm for ACT–R. In *Proceedings of the Sixth International Conference on Cognitive Modelling* (pp. 40–45). Mahwah, NJ: Lawrence Erlbaum. (ISBN 0-8058-5426-6)

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