Evaluating Cognitive Models and Architectures

Marc Halbrügge
Human Factors Institute, Universität der Bundeswehr München
marc.halbruegge@unibw.de

Abstract

Cognitive modeling is a promising research area combining the fields of psychology and artificial intelligence. Cognitive models have been successfully applied to different topics like theory generation in psychology, usability testing in human-computer interface research, and cognitive tutoring in algebra teaching.

This paper is focused on two major problems concerning the evaluation of cognitive models and architectures. First, it is shown that current approaches to quantify model complexity are not sufficient and second, the claim is reinforced that the flexibility of cognitive architectures leads to the problem that they cannot be falsified and are therefore too weak to be considered as theories.

Cognitive architectures

From a psychologist's point of view, the main goal of cognitive modeling is to simulate human cognition. A cognitive model is seen as an instantiation of a theory in a computational form. This can then be used to test a certain theory empirically.

Cognitive architectures like SOAR (Laird, Newell, & Rosenbloom 1987), EPIC (Kieras & Meyer 1997), and ACT-R (Anderson *et al.* 2004) are useful frameworks for the creation of cognitive models. In addition, they are often regarded as useful steps towards a unified theory of cognition (Newell 1990).

When a computational model is used to evaluate a theory, the main goal is (generalizable) concordance with human data. But this is not a sufficient proof of the theory. The following problems are often stated in the literature:

- *Compliance with the theory.* The underlying theory and its implementation (in a cognitive architecture) may diverge.
- *Irrelevant specification* (Reitman 1965). Most of the models need to make additional assumptions. It is very hard to tell which parts of the model are responsible for its overall performance and which parts may be obsolete.

  Newell (1990) demanded to "listen to the architecture" when creating new models in order to get around this problem. While this is a recommendable request, it is hard to verify that the modeler has complied with it.
- *Underlying theories.* A special case of the irrelevant specification problem can occur when the architecture incorporates knowledge from previous research, for example Fitts' law (Fitts 1954) about the time needed to perform a rapid movement of the forearm. These underlying empirical laws may provide a big part of the fit of the cognitive model in question.
- *Interpretability*. Most of the time, one needs to look at the code of a model in order to understand what it really does. As many psychologists do not hold a degree in computer science, this hinders them from participating in the cognitive modeling research field.

The usage of a cognitive architecture – as opposed to modeling everything from scratch – does not solve these problems completely, but weakens them noticeably. Cognitive architectures are subject to social control by the research community. This way, theory compliance, interpretability, and rejection of irrelevant parts are enforced at least on the level of the architecture.

An informative evaluation of cognitive models has been done by Gluck & Pew (2005). They applied several cognitive architectures to the same task and compared the resulting models not only with regard to goodness-of-fit but also to modeling methodology, features and implied assumptions of the particular architecture, and the role of parameter tuning during the modeling process.

This approach of Gluck & Pew is well suited for addressing the problems of model evaluation besides goodness-of-fit and therefore I'll recommend it as standard practice.

Evaluation methodology for cognitive models

I stated above that from a psychologist's point of view concordance with human data is the main goal of cognitive modeling. This concordance is typically assessed by computing goodness-of-fit indices. This approach has been critized sharply during the last years (Roberts & Pashler 2000), initiating a discussion that concluded with the joint statement that a good fit is necessary, but not sufficient for the validity of a model (Rodgers & Rowe 2002; Roberts & Pashler 2002).

In order to assess the fit of a model or to compare fits between models, it is important to be aware of the complexity of the models. Models with many degrees of freedom usually achieve better fits, but bear the risk of *overfitting*, the siamese twin of the irrelevant specification problem described in the previous section. In addition, the principle of parsimony demands to prefer a less complex model to a more complex one, given comparable fit. Therefore measurements of complexity are needed.

Pitt, Myung, & Zhang (2002) have developed statistics for model selection that take into account both goodness-of-fit and complexity. These statistics aim at closed form mathematical models, where fitting means to choose the member of a class of functions that represents some empirical data best. Examples for this type of model range from simple linear regression to complex neural networks.

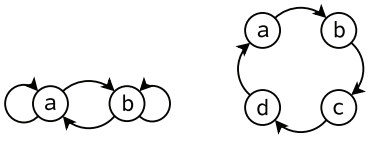
In order to expand the notion of complexity to cognitive models, Baker, Corbett, & Koedinger (2003) used the number of free weights of the subsymbolic part of an ACT-R model as measure of the degress of freedom of the model.

As will be show in the following, this is not sufficient. While some parts of cognitive models can be described by mathematical formulas (i.e. retrieval latencies, spreading activation), they also include an algorithmic part. Additional degrees of freedom can emerge from the algorithmic structure of a model. Therefore the complexity of a model can not be adequately quantified by the number of numerical weights in its implementation.

Example: Memory retrieval in ACT-R

This claim shall be illustrated by an example. Imagine two simple cognitive models using ACT-R, both of them retrieving a series of chunks from declarative memory. Which chunk is to be retrieved next depends on the production that currently fires. The models stop when the memory retrieval fails which depends mostly on the amount of noise added to chunk activation.

Both models consist of four productions (resembling four state transitions), but differ in their number of chunks in the declarative memory. The structures of the models are displayed in figure 1.



Model 1 Model 2

Figure 1: Structure of the two models. Circles represent the internal states, arrows productions. Initial and final states and corresponding productions are not shown.

Model 1 consists of two states a and b, represented by chunks in the retrieval buffer. The four productions (arrows in figure 1) allow any possible state transition (a→a, a→b, b→b, b→a).

Model 2 consists of four states represented by chunks a, b, c and d. The productions allow state transitions from every state to its single successor (a→b, b→c, c→d, d→a).

Notice that although the second model looks deterministic, the retrieval failure that breaks the cycle of firings may appear at any moment. All of the four activation weights are equally important for the outcome of the model.

Production utility learning is not used in the models. The only influential weights are the activations of the chunks.

Now the question of model complexity shall be addressed: According to the weight counting approach (Baker, Corbett, & Koedinger 2003), the second model is more complex than the first one. It uses more memory objects with each single one being associated with a numerical activation. But because of the arrangement of states and productions (see figure 1), this model can only create outputs of the form abcd-abcda and so on. The first model in contrast can output any sequence of a's and b's. Therefore model 1 should be considered the more complex one.

Even if model run time is used as dependent variable, you can spot the difference in complexity. Imagine that two of the four possible transitions in the models take a lot of time compared to the other two (i.e. 1000 ms vs. 100 ms). While the overall distribution of the run time will stay comparable for the two models, the second model will only be able to produce a limited set of values (i.e. 100, 1100, 1200, 2200, 2300).

In order to test this, I created the two models,[1] ran each one 10000 times and recorded the simulated run time. The results are displayed in figure 2. The greater restriction of model 2 is apparent.

A close inspection of the models reveals that model 2 possesses just one degree of freedom – the total run time: Depending on the noisy chunk activation, the model may run shorter or longer. But when the number of states visited during a run is known, the sequence of states is known as well.

On the other hand, model 1 has more degrees of freedom because it can take multiple paths of production firings to reach a specific state at a given position in the sequence of states.

Taken together, the example provides two important findings:

- Even for two simple models as the ones described above (about 100 lines of code each), it is very hard to assess the model complexity when only the number of chunks and productions is known. In order to be able to judge a model, one has to look into the code.
- Knowledge of numerical parameters is not sufficient for the estimation of model complexity as more degrees of freedom can emerge from the algorithmic structure.

  The strategy of "eliminating parameter estimation" (Anderson *et al.* 2004, p. 1057) in order to deal with the criticism of model fitting (Roberts & Pashler 2000) falls short in this case.

[1] The source code can be downloaded from `http://www.unibw.de/lrt11/halbruegge/ACT-R`

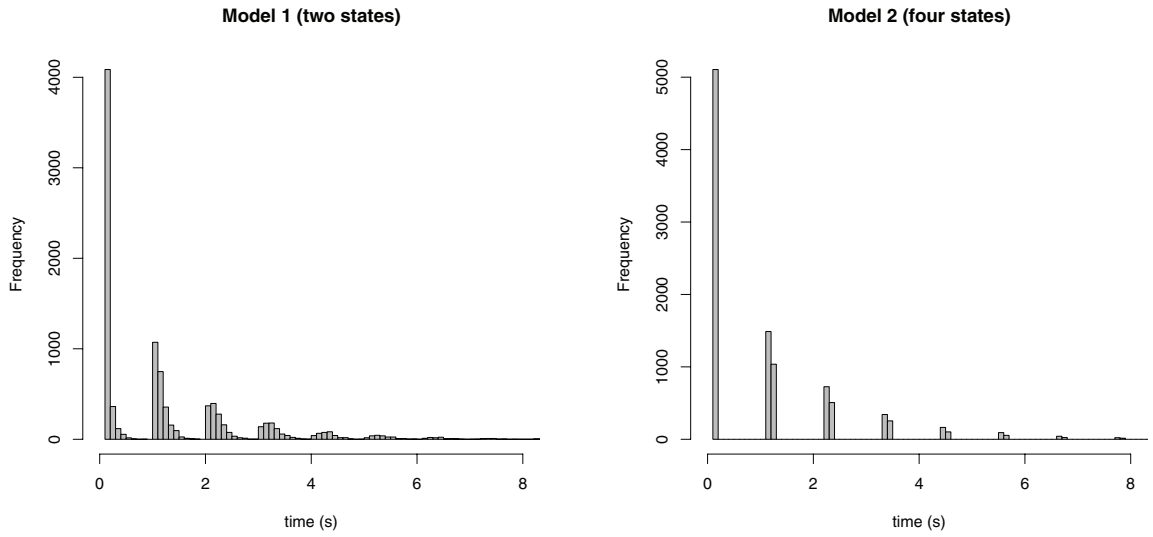


Figure 2: Histograms of run times for model 1 and 2 (N=10000)

As long as there are no better methods available for the measurement of model complexity, I would suggest that scientific publications in the cognitive modeling domain should always be accompanied by a source code release.

Validating a cognitive architecture

Anderson & Lebiere (2003) proposed the following list of 12 criteria for the evaluation of cognitive architectures: flexible behavior, real-time performance, adaptive behavior, vast knowledge base, dynamic behavior, knowledge integration, natural language, learning, development, evolution, and brain realization. The publication of such a list is partly inspired by Newell's request to "play a cooperative game of *anything you can do, I can do better*" (Newell 1990, page 507).

Although Anderson & Lebiere make some important points, they miss the problem of falsifiability. Before we ask the question "*How* do we validate a cognitive architecture?", we should answer the question "Can we validate a cognitive architecture *at all*?" or, according to Popper (1989), "Can we *falsify* a cognitive architecture?"

Can a cognitive architecture be falsified?

This section concentrates on ACT-R, but should be applicable to other cognitive architectures as well.

Anderson & Lebiere state that ACT-R is *not* a "general computational system that can be programmed to do anything" (page 597). Although I do agree that the computational power of ACT-R can be limited, the cognitive architecture as such is Turing complete from a more formal viewpoint.

This claim has first been made by Wexler (1978) and has not been ruled out since then. The logic is simple: Cognitive architectures (at least ACT-R) are complete Turing machines. That means that they can compute *anything* that is computable by this sort of machine. Therefore, a cognitive architecture cannot be falsified. It is always true, like the sentence "Tomorrow it will rain or it won't rain".

What *can* be falsified are additional assumptions, e.g. the ubiquitious 50 ms granularity of the production cycle or mathematical models of spreading activation. Nevertheless, the architecture itself cannot be validated. As Tadepalli (2003) wrote, in concluding a similar remark: "This may [...] mean that we have to put aside the problem of evaluating cognitive architectures, for now or forever".

Conclusions

At the beginning of this paper it was stated that cognitive architectures are often viewed as efforts to create a unified theory of cognition. I have tried to show that this view is not sustainable. Cognitive architectures should be regarded as "theory languages", similar to programming languages.

This does not mean that cognitive architectures are useless. Cognitive architectures provide modeling frameworks that ease both model creation, the communication between modelers, and the combination of models. But when it comes to validation, the focus should not lie on architectures but on models. These can be tested empirically.

In order to evaluate and compare cognitive models, we need a way to assess their complexity. As long as we lack a formal methodology to do this, we should alternatively release the (well commented) code of our models. This is the only way to judge the actual algorithm, and hereby the plausibility – or face validity – of a cognitive model.

Acknowledgments

This work was supported by the German Research Foundation (DFG) within the excellence cluster "Cognitive Techni-

cal Systems"; it is part of the junior research group "Cognitive Modeling in Ergonomics".

References

Anderson, J. R., and Lebiere, C. 2003. The newell test for a theory of cognition. *Behavioral and Brain Sciences* 26(5):587–601.

Anderson, J. R.; Bothell, D.; Byrne, M. D.; Douglass, S.; Lebiere, C.; and Qin, Y. 2004. An integrated theory of the mind. *Psychological Review* 111(4):1036–1060.

Baker, R. S.; Corbett, A. T.; and Koedinger, K. R. 2003. Statistical techniques for comparing act-r models of cognitive performance. In *Proceedings of the 10th Annual ACT-R Workshop*.

Fitts, P. M. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology* 47(6):381–391.

Gluck, K. A., and Pew, R. W., eds. 2005. *Modeling Human Behavior with Integrated Cognitive Architectures*. Mahwah, NJ: Lawrence Erlbaum Associates.

Kieras, D. E., and Meyer, D. E. 1997. An overview of the epic architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction* 12:391–438.

Laird, J. E.; Newell, A.; and Rosenbloom, P. S. 1987. Soar: an architecture for general intelligence. *Artificial Intelligence* 33(1):1–64.

Newell, A. 1990. *Unified Theories of Cognition – The William James Lectures, 1987*. Cambridge, MA, USA: Harvard University Press.

Pitt, M. A.; Myung, I. J.; and Zhang, S. 2002. Toward a method of selection among computational models of cognition. *Psychological Review* 109(3):472–491.

Popper, K. R. 1989. *Logik der Forschung*. Tübingen: Mohr, 9 edition.

Reitman, W. R. 1965. *Cognition and Thought*. New York: Wiley.

Roberts, S., and Pashler, H. 2000. How persuasive is a good fit? a comment on theory testing. *Psychological Review* 107(2):358–367.

Roberts, S., and Pashler, H. 2002. Reply to rodgers and rowe (2002). *Psychological Review* 109(3):605–607.

Rodgers, J. L., and Rowe, D. C. 2002. Theory development should begin (but not end) with good empirical fits: A comment on roberts and pashler (2000). *Psychological Review* 109(3):599–604.

Tadepalli, P. 2003. Cognitive architectures have limited explantory power. *Behavioral and Brain Sciences* 26(5):622–623. Commentary on Anderson & Lebiere, 2003.

Wexler, K. 1978. A review of john r. anderson's language, memory, and thought. *Cognition* 6(4):327–351.