In Biologically Inspired Cognitive Architectures, Proceedings of the First Annual Meeting of the BICA Society.
Editors: A.V. Samsonovich, K.R. Johannsdottir, A. Chella and B. Goertzel. Amsterdam: IOS Press.

Validating a High Level Behavioral Representation Language (HERBAL): A Docking Study for ACT-R

Changkun Zhao[1], Jaehyon Paik[2], Jonathan H. Morgan[1], and Frank E. Ritter[1]
[1] *The College of Information Sciences and Technology*
[2] *The Department of Industrial and Manufacturing Engineering*
The Pennsylvania State University
University Park, PA 16802
cuz111@psu.edu, jaehyon.paik@psu.edu, jhm5001@psu.edu, frank.ritter@psu.edu

ABSTRACT. We present a docking study for Herbal, a high-level behavioral representation language based on the problem space computational model. This study docks an ACT-R model created with Herbal to one created by hand. This comparison accomplishes several things. First, we believe such studies are necessary for achieving and demonstrating the theoretical rigor and repeatability promised by high-level representation languages. Second, it is necessary to evaluate the effectiveness and efficiency of high-level cognitive modeling languages if they are to make a significant impact in either the cognitive or social sciences. Third, this kind of study provides an opportunity to test Herbal's ability to produce ACT-R models from a GOMS-like representation that contains hierarchical methods, memory capacity, and control constructs. Finally, this study provides an example model for future validation work in this area. Our study addresses each of these points by docking Pirolli's [1] price finding model in ACT-R with the same model written in Herbal. We extended Herbal to support more memory types, and in the process may have extended the PSCM.

Keywords. Model validation, Docking, Herbal, and ACT-R

Introduction

This paper addresses the challenges associated with comparing and validating cognitive models across cognitive architectures. Cognitive models implemented in cognitive architectures have successfully modeled the effects of bounded rationality[2] on cognition[3-6] and to some extent on social interactions[7] Achieving a domain of validity for cognitive models has, however, proven more difficult for several reasons. These reasons include the following: the inability to hold high-level abstractions constant across architectures while testing a specific model; ambiguity regarding the relationship between a model and unique aspects of its host architecture; and confounding variables introduced by architectural differences in generating models[8].

High-level cognitive languages such as Herbal (High-level Behavioral Representation Language)[9], HLSR (High-Level Symbolic Representation)[8], and

Icarus[10] offer an approach for addressing these issues. Each has a representation structure that can serve as a baseline abstraction from which to compare models. These abstractions share a core set of commonalities found in most cognitive architectures including: declarative and procedural memory, memory retrieval mechanisms, goals, methods for responding to external events, and iterative decision-making[8]. Using high-level cognitive languages for model validation, however, requires first testing the ability of these languages to replicate the results of models developed in their supported architectures (Soar, ACT-R, and Jess in the case of Herbal).

This study is an initial docking study of Herbal's ACT-R compiler[11, 21]. Herbal is an open source cognitive modeling language based on Newell et al.'s[6] Problem Space Computational Model (PSCM). It uses a modified version of the PSCM to represent a set of common cognitive modeling knowledge structures to create models in three cognitive architectures (Soar, ACT-R, and Jess). Users can either develop models using a GUI editor or by editing Herbal's XML code directly[12].

To test Herbal, we extend validation approaches developed for social modeling[13, 14] to the validation of cognitive models. Specifically, we test the equivalence of two versions of Pirolli's[1] price finding model (one developed in ACT-R, the other in Herbal) using an alignment, or "docking", study methodology.

1. Docking cognitive models

Cognitive Science has historically compared simulation results to human data to assess validity. While this method remains vital, it provides no clear criteria for establishing either model equivalence or subsumption within or across cognitive architectures. In addition, this method provides no means of isolating implementation effects from the premises of the theory; the theory and the implementation in essence are indistinguishable[8]. Docking studies conducted within a high-level cognitive architecture provide both criteria for establishing equivalence for models testing the same phenomena and a way of disambiguating implementation effects from the theory's premises. Cooper, Fox, Farringdon, and Shallice[15] have suggested a similar direction for cognitive modeling.

Docking studies are commonly used in social modeling[16], systems engineering[17], and bioinformatics[18] to test model equivalence[13]. Model equivalence is evaluated by comparing the tested models' output or results after processing identical inputs. Equivalence, in this approach, is further defined in one of three ways: numerical, statistical, or relational equivalence. These notions of equivalence differ in their strictness and are appropriate in different settings. The most rigorous of these tests, numerical equivalence refers to comparing the models' output to see if the numeric results are identical. Numerical equivalence is seldom used because it inapplicable when testing stochastic models. When validating stochastic models, researchers generally test for statistical equivalence by comparing the models' distributions over multiple runs. When, however, the models' inputs or outputs differ, relational equivalence is assessed, for example to what degree the same internal relationships exist across levels of aggregation.

Docking studies entail a process of translation that isolates the core premises of a model from the implementation effects associated with its host environment. Achieving behavioral equivalence between models is, however, nontrivial. In addition

to aligning the two models' parameters, this process requires isolating the models' core processes and ensuring those processes are consistent across both models. This procedure is commonly referred to as establishing component equivalence. Establishing component equivalence across cognitive models is complicated by the layered nature of cognitive modeling.

A cognitive model generally operates within a cognitive architecture to perform a specific task in a given environment. Each layer (the model, the architecture, the task, and the environment) complicates the validation process; however, the embedded nature of cognitive models poses a unique validation problem. Cognitive models implemented in cognitive architectures often represent more than a single theory but a theory of theories, a layered representation specifying different but interrelated aspects of cognition across multiple levels of abstraction. A cognitive architecture is, in itself, a theory of cognition while versions of that architecture can be viewed as elaborations to or revisions of that theory. Furthermore, cognitive models developed in a given architecture can vary with respect to what architectural components they utilize depending upon model's specified task.

In addition, a cognitive model's output can be complex, making assessing equivalence difficult. Cognitive models frequently produce traces of activity that list the number, types, timing, and information content of the steps performed by the model in a given task. These steps generally rely on stochastic processes whose parameters are specified by the architecture but can be adjusted. For models capable of learning, these processes can vary retention rates, obscure instances where learning has occurred, and produce differences in the models' learned behaviors despite performing the same steps in the same way. These factors make general predictions about other task sequences or other tasks challenging, but not impossible. We expect these factors will also complicate validating complex cognitive models; however, these confounding variables are known[15] and can be controlled[19]. Also, in many cases, the comparisons and docking procedures do not lead to simple summative measures but formative measures giving rise to insights about the task, the cognitive architecture, and the human behavior.

2. Comparative study of ACT-R and Herbal Models

For this validation study of Herbal's ACT-R compiler, we compare three versions of Pirolli's[1] Price Finding Model (PFM), the original in ACT-R 5: one in ACT-R 6, and the third in Herbal 3.0.5.

The PFM has two distinct advantages that made it suitable for this initial study. First, while the PFM is a simple model, we expected its sophisticated manipulation of declarative memory elements would test both Herbal's ACT-R compiler and the ability of its ontology to represent a broader array of cognitive models. Second, while the PFM's use of ACT-R's declarative, procedural, and goal retrieval systems is consistent with more complex ACT-R models, the PFM does not utilize either ACT-R's perceptual-motor or its subsymbolic computations, simplifying the validation process.

2.1. Methodology

We tested for numerical equivalence between the three versions of the PFM by comparing three major outputs: the models' total number of cycles, the number of sub-cycles, and the models' best price. While there were no changes to the memory systems used by the PFM between ACT-R 5 and 6, we compared the PFM implemented in ACT-R 6 to a version implemented in Herbal because the original model was lost. After collaborating with the author, we were able to re-implement the PFM in ACT-R 6 in approximately 6 hours, and confirmed numerical equivalence between the original and our re-implemented version (noted in **Table 1**) for model cycles and best price found. We expected numerical equivalence between the three models for two reasons: first, the inputs and means of interaction for all three models were identical; second, the PFM does not utilize the stochastic processes in ACT-R, namely its perceptual-motor or subsymbolic computations. We confirmed the PFM's results were constant across both ACT-R versions by running the new model multiple times (n=10).

2.2. Models' Description: Establishing component equivalence

Establishing component equivalence between the ACT-R versions of the PFM and its Herbal counterpart required changes to Herbal's ACT-R compiler. Though Herbal's ACT-R compiler has supported hierarchical task analyses[20, 21], supporting the PFM required developing a new retrieval function and changing the interface to enable users to designate chunks as either goal or retrieval chunks. We describe these changes and the steps taken to establish component equivalence below. We first describe the processes used by the PFM in ACT-R 6, and then compare it to those developed for Herbal's ACT-R complier.

2.2.1. Pirolli's Price Finding Model: ACT-R 5 and ACT-R 6

Pirolli's PFM consists of six productions—four productions for the general goal (*start*, *first-link*, *next-link*, and *done*) and two productions for the subgoal (*minimum-price-stays-the-same* and *new-minimum-price*). The model uses the general goal for finding and storing the best price and the subgoal for comparing new prices with the current best or minimum price. Out of the two prices, the PFM selects the lower price one.

The production *start* initializes the goal and retrieval buffers; the production *first-link* uses the first price accessed from declarative memory as the initial best price; the production *next-link* activates a subgoal to compare a new price with the best price; and the productions *new-minimum-price* and *minimum-price-stays-the-same* determine whether a new best price is selected. If the price selected is less than the best price, the subgoal cycle returns the new price as the new minimum price. Otherwise, it returns the current best price as the minimum price. Then, production *next-link* sets the returned minimum price as the best price in the general goal cycle. The model uses a competitive iterative loop consisting of the productions *next-link* and *done* to drive the information foraging process. The PFM determines whether to fire the *next-link production* by evaluating two external functions[1]: an *expected savings rate* and *labor*

[1] These are external in the sense that they are not intrinsic to ACT-R.

value. Calculating the *price*, *expected savings rate*, and the *labor value*, the model fires the *next-link* production when there is an expected savings associated with searching for a new price; and fires the *done* production when the expected saving is less than the labor cost.

2.2.2. Pirolli's Price Finding Model: Herbal 3.0.5

As noted previously, we achieved component equivalence in two ways: by modifying Herbal's interface to enable users to specify whether a component will be used as a chunk-type in the retrieval buffer, and by adding a "retrieve" function as an action element, allowing the model to retrieve a particular value in the retrieval buffer. We created two Herbal types: a *link* type and a *find* type. The *link* type corresponds to the chunk-type *link* in the ACT-R versions of the PFM while the *find* type is a goal that replicates the chunk-types *find* and *minimum*.

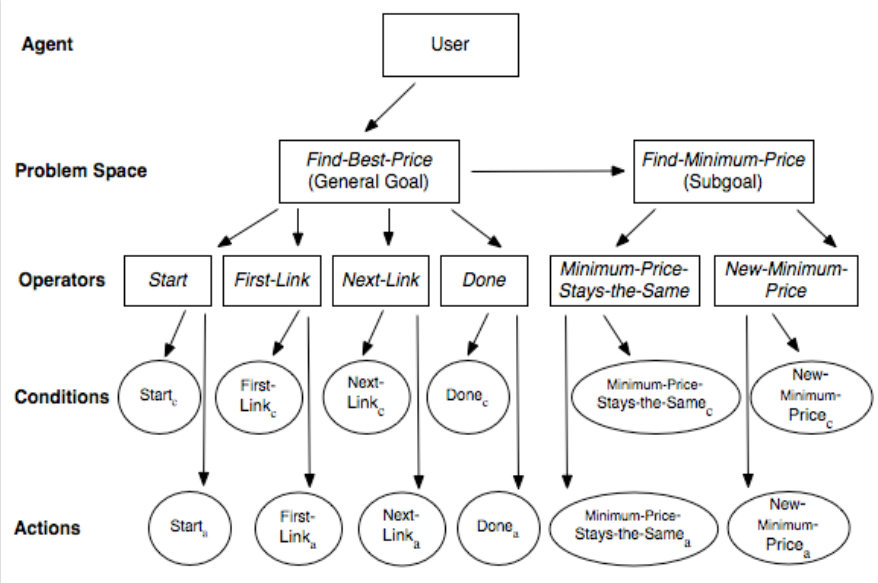

Figure 1. Herbal's ontological representation of the PFM

Figure 1 shows Herbal's ontological representation of the PFM. This representation contains five levels: agent, problem space, operators, conditions, and actions. The highest level is the user agent, which includes two problem spaces, the *find-best-price* and the *find-minimum*. The *find-best-price* problem space searches a list of prices and stores the best price. The *find-minimum* problem space compares the best price to a new price and returns the lower price. The two problem spaces consist of six operators that are created to match the six productions found in the ACT-R versions of the PFM. The *start*, *first-link*, *next-ink*, and *done* operators are associated with the *find-best-price* problem space while the *price-stays-the-same* and *new-as-minimum* operators are associated with the *find-minimum* problem space. Each operator contains one condition and one action to replicate the buffer testing process and the buffer managing process found in ACT-R.

2.3. Results

The ACT-R 6 version of the PFM used the 30 declarative memory elements a particular *expected saving rate*. Our ACT-R model examined 27 declarative elements before firing the production *done*. By the 27th memory element, the best price was $66, with an *expected saving rate* of $8 per hour. At this price, the *expected saving rate* was less than the assumed labor cost $10 per hour, resulting in the model firing the *done* production and terminating the whole process.

The Herbal version of the PFM used 30 declarative memory elements, examining 27 before firing the *done* production. By the 27th memory element, the best price was $66, with an *expected savings rate* of $8 per hour. Table 1 shows the details of result.

Table 1. Experimental results for three tested variables.

Model	Total Cycles	Sub-Cycles	Best Price
Price Finding Model (ACT-R 5)	53	25	66
Price Finding Model (ACT-R 6)	53	25	66
Price Finding Model (Herbal 3.05)	53	25	66

3. Discussion and conclusion

Our docking study began with recreating Pirolli's[1] price finding model. The recreated model performed identically across two versions of ACT-R for the dimensions we tested. After reconstructing the PFM in ACT-R 6, we created a version of the PFM in Herbal. To do this, we had to extend Herbal's ACT-R compiler to include a retrieval function. The resulting Herbal model was numerically equivalent in the following ways: the number of total cycles, sub-cycles, and the best price.

We also compared the hand-coded ACT-R source code with Herbal's auto-generated source code to determine to what degree Herbal can replicate the ACT-R models. We found that the two versions of the source code are structurally equivalent because they have the same number of productions and chunk types, and the elements of "IF" (conditions) and "THEN" (actions) in every production are nearly the same. They differ, however, in that the Herbal code checks the goal buffer before operating each goal type slot, whereas the hand-coded model only checks the buffer at the beginning of each production's condition and action. Nevertheless, the Herbal model passed the syntax test, running with no errors.

Reimplementing the PFM in ACT-R 6 took 6 hours after corresponding with the author, while implementing the PFM in Herbal took approximately an hour. In this case, using Herbal decreased the time required to build the model. Extending Herbal's interface and the Herbal's ACT-R compiler, however, took a couple of days to fully develop and test. The features necessary to better support ACT-R's declarative memory retrieval functions constitute an extension of not only Herbal's technical abilities but also its ontology.

Nevertheless, this study provides a model for testing high-level cognitive architectures, as well several lessons for future docking studies. For the later comparison and development of models, researchers should publish or archive their models. While rewriting the PFM was not onerous, reconstructing a larger model would be. The use of unstructured model archives, such as act.psy.cmu.edu, is clearly

helpful while more structured archives[22, 23,26], is better yet. Herbal may be able to provide such an archive, at least for the models created in it. Our study suggests that ACT-R 5 and 6 were functionally equivalent for this model. While the PFM was simple, it used core aspects of the architecture. Testing the effects of changes to complex architectures will, however, require docking multiple models of greater complexity than the PFM. Finally, developing and validating models in Herbal or other high-level cognitive modeling languages facilitates both documentation and reuse. Having the model in a formal representation with more explicit entry and exit points (operators in this case) encourages reuse by putting the model in a more comprehensible format.

Furthermore, using docking studies incrementally to compare increasingly more complex models affords us the opportunity to identify core processes through testing, helping us retain some degree of parsimony in our theories. This study, thus, provides an incremental testing and development strategy for high-level cognitive architectures. More specifically, operationalizing a concept of component equivalence allows us to identify a set of core processes. Over time, this approach may allow us to more fully realize a unified theory of cognition by establishing more concrete methods of model subsumption.

Newell[5] noted that there is more in cognitive architectures than we have dreamed. Examining hand-written models though docking is one way to better understand the capabilities in and salient differences between cognitive architectures. The way we make models has implications for developing more unified theories of cognition. While many models fully utilize the core attributes of their host architectures, others do not. For instance, when we reversed compiled a Soar models[24] we found that not all published Soar models follow the PSCM. In some instances, this may be appropriate and extends their use. These extensions and uses can be supported by a high level language. On the other hand, non-canonical use makes replicating previous work more difficult, and frustrates efforts to build a more coherent body of knowledge in the Cognitive Sciences.

4. Future work

In the future, we will explore three directions regarding Herbal. First, we will continue to extend the Herbal ACT-R compiler because it cannot fully support ACT-R currently. Second, we will choose a more complex model with subsymbolic computation such as the Diag Model[25.] Third, we will extend Herbal's Soar compiler to compile our PFM model into Soar source code. In this case, we can validate the effectiveness of cross-architecture modeling in Herbal and compare the behavioral differences between Soar and ACT-R.

5. Acknowledgement

This work also is supported by the grants from DTRA (HDTRA1-09-1-0054) and ONR (N00014-09-1-1124). We would like to thank Dr. Mark Cohen and Dr. Jong Kim for their useful suggestions and comments.

References

[1] Pirolli, P. *Information foraging theory: Adaptive interaction with information*. Oxford University Press, New York, NY, 2007.

[2] Simon, H. A. A behavioral model of rational choice. *The Quarterly Journal of Economics,* **69** (1955), 99-118.

[3] Anderson, J. R. *How can the human mind exist in the physical universe?* Oxford, New York, NY, 2007.

[4] Laird, J. E., Rosenbloom, P.S., & Newell, A. Chunking in Soar: The anatomy of a general learning mechanism. *Machine Learning,* **1**(1986), 11-46.

[5] Newell, A. *Unified theories of cognition*. Harvard University Press, Cambridge, MA, 1990.

[6] Newell, A., Yost, G. R., Laird, J. E., Rosenbloom, P. S., & Altmann, E. Formulating the problem space computational model. *Carnegie Mellon Computer Science: A 25-Year commemorative*. ACM-Press (Addison-Wesley), Reading, MA, 1991, 255-293.

[7] Carley, K. M., & Newell, A. The nature of the social agent. *Journal of Mathematical Sociology,* **19** (1994), 221-262.

[8] Jones, R. M., Crossman, J. A., Lebiere, C., & Best, B. J. An abstract language for cognitive modeling. In *The Seventh International Conference on Cognitive Modeling*, Trieste, Italy, 2006, 160-165.

[9] Cohen, M. A., Ritter, F. E., & Haynes, S. R. Applying software engineering to agent development. *AI Magazine,* **31**, 2010, 25-44.

[10] Langley, P., & Choi, D. A unified cognitive architecture for physical agents. *In Proceedings of the twenty-first national conference on artificial intelligence.* AAAI Press, Boston, 2006.

[11] Haynes, S. R., Cohen, M. A., & Ritter, F. E. Designs for explaining intelligent agents. *International Journal of Human-Computer Studies,* **67** (2009), 99-110.

[12] Friedrich, M., Cohen, M. A., & Ritter, F. E. *A gentle introduction to XML within Herbal*. ACS Lab, The Pennsylvania State University, University Park, PA, 2007.

[13] Axtell, R., Axelrod, R., Epstein, J. M., & Cohen, M. D. Aligning simulation models: A case study and results. *Computational and Mathematical Organization Theory,* **1** (1996), 123-141.

[14] Loui, M., Carley, K. M., Haghshenass, L., Kunz, J., & Levitt, J. Model comparisons: Docking OrgAhead and SimVision. In *Proceedings of the 1st Annual Conference of the North American Association for Computational Social and Organization Science* 6 (Day 4). NAACSOS, Pittsburgh, PA, 2003.

[15] Cooper, R., Fox, J., Farringdon, J., Shallice, T. A systematic methodology for cognitive modelling. *Artificial Intelligence***, 35** (1996), 3-44.

[16] Xu, J., Gao, Y., & Madey, G. A docking experiment: Swarm and repast for social network modeling. In *Proceedings of the Seventh Annual Swarm Researchers Meeting (Swarm 2003)*. Notre Dame, IN, 2003.

[17] Barry, P. S., Koehler, M. T. K., McMahon, M. T., Tivnan, B. F. Agent-directed simulation for systems engineering: Applications to the design of venue defense and the oversight of financial markets. *International Journal of Intelligent Control and Systems,* **14** (2009), 20-31.

[18] Fillizola, M., & Weinstein, H. The study of G-protein coupled receptor oligomerization with computational modeling and bioinformatics. *The Federation of European Biochemical Societies Journal,* **272** (2005), 2926-2938.

[19] Gluck, K. A., & Pew, R. W (Eds.). *Modeling human behavior with integrated cognitive architectures: Comparison, evaluation, and validation.* Erlbaum, Mahwah, NJ, 2005.

[20] Paik, J., Kim, J. W., Ritter, F. E., Morgan, J. H., Haynes, S. R., Cohen, M. A. Building Large Learning Models with Herbal. *Proceedings of ICCM -2010- Tenth International Conference on Cognitive Modeling*. Philadelphia, USA, 2010, 187-192.

[21] Cohen, M. A., Ritter, F. E., & Haynes, S. R. Applying software engineering to agent development. *AI Magazine*. **31**(2010), 25-44.

[22] Cornel, R., Amant, R. S., & Shrager, J. Collaboration and modeling support in CogLaborate. In *Proceedings of the 19th Conference on Behavior Representation in Modeling and Simulation*. BRIMS Society, Charleston, SC, 2010, 10-BRIMS-129, 146-153.

[23] Wong, T.J., Cokely, E. T., & Schooler, L. J. An online database of ACT-R parameters: Towards a transparent community-based approach to model development. *Proceedings of ICCM - 2010 - Tenth International Conference on Cognitive Modeling*. Philadelphia, USA, 2010, 282-286

[24] Girouard, A., Smith, N. W., & Ritter, F. E. Lessons from decompiling an embodied cognitive model. In Cognitio 2006 Workshop, cognitio.uqam.ca/index.php?section=posters&lng=en,2006

[25] Ritter, F. E., & Bibby, P. A. Modeling how, when, and what learning happens in a diagrammatic reasoning task. *Cognitive Science*. **32**(2008), 862-892.

[26] Myung, J., & Pitt, M. Cognitive Modeling Repository. *Cognitive Science,* 2010.