More than 8,192 Ways to Skin a Cat: Modeling Behavior in Multidimensional Strategy Spaces

Mason R. Smith (masonrs@umich.edu)
Richard L. Lewis (rickl@umich.edu)
Department of Psychology, University of Michigan
Ann Arbor, MI 48109-1043 USA

Andrew Howes (HowesA@manchester.ac.uk)
Manchester Business School, University of Manchester
Manchester M15 6PB UK

Alina Chu (achu@umich.edu)
Department of Electrical Engineering and Computer Science, University of Michigan
Ann Arbor, MI 48109-2122 USA

Collin Green (cgreen@mail.arc.nasa.gov)
NASA Ames Research Center
Moffett Field, CA 94035 USA

Abstract

How can we model behavior on complex, real-world tasks with a large range of possible strategies that may vary along multiple dimensions? In this paper, we show show how an emerging approach to cognitive modeling, *cognitively bounded rational analysis*, can be applied to efficiently specify large, multidimensional strategy spaces, and to predict which strategies within the space are followed. The approach also supports a novel way of analyzing error control strategies, by directly modeling error recovery procedures and factoring these into strategy prediction. We apply this approach in a model of a typing task exemplifying three dimensions of strategic variability: decomposition of tasks into subtasks, parallel vs. serial processing of subtasks, and control of errors. We present empirical results showing the strategies people adopted on the task. The model successfully predicts the strategies used, by optimizing over the strategy space for a utility function defined as the performance-based payoff used in the experiment.

Keywords: Cognitive architectures; cognitively bounded rational analysis; HCI; strategies; errors

Introduction

To understand how people interact with a device, one needs to know which strategies people follow, out of the many possible strategies for accomplishing their task. However, for a real-world task, the space of possible strategies may be very large. For example, in the simple task of transcribing a string of digits on a computer, strategies may vary along several dimensions including:

(a) How tasks are decomposed into subtasks, e.g., whether the entire string is read and then typed, or whether the string is broken into groups;

(b) How subtasks are parallelized or interleaved, e.g., whether reading and typing processes overlap or occur in serial;

(c) How the incidence and costs of errors are controlled, e.g., how often people choose to check what they've typed for errors.

The entire strategy space for transcribing a digit string will be the combination of (at least) these three dimensions of strategic variability.

Current modeling approaches, including cognitive architectures such as ACT-R (Anderson et al., 2004) and EPIC (Meyer & Kieras, 1997), can be used for modeling behavior given a strategy, but do not directly support prediction of strategies used, relying on the modeler to identify or hypothesize the strategy through fitting to data or task analysis (see Howes, Lewis, and Vera (in revision) for a complete discussion). In previous work (Howes et al., in revision; Howes, Vera, Lewis, & McCurdy, 2004) we presented the *cognitively bounded rational analysis* approach which supports generation of strategy spaces and prediction of strategy selection. In this paper we report data from an experiment involving a typing task, and we apply cognitively bounded rational analysis to produce a model of the task that captures three types of strategic variability exhibited by subjects: adaptation in grouping the characters to be typed, in parallelizing encoding and typing processes, and in controlling error correction costs through error monitoring. The model breaks new ground in incorporating multiple dimensions simultaneously and in supporting reasoning about interactions between the dimensions. Furthermore, it represents a novel approach to modeling adaptation to error, in that error recovery is directly incorporated into the model and is taken into account in strategy prediction.

Cognitively Bounded Rational Analysis

Cognitively bounded rational analysis is an approach to modeling skilled task performance in which a space of possible strategies is generated subject to architectural and task constraints, and an optimal strategy relative to a given utility function is identified. Specifically, it consists of four components: strategy space specification, architecture specification,

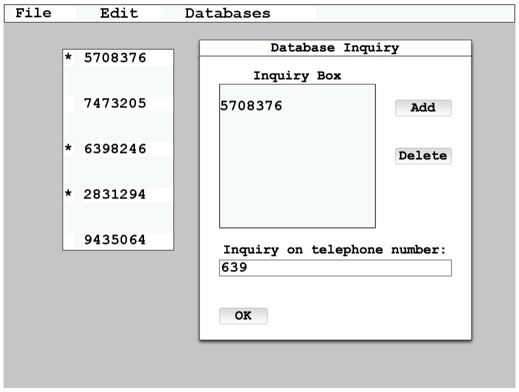

Figure 1: Screenshot of the database inquiry task, as the user transcribes a number from the list at left into the text box.

utility function specification, and strategy selection.

We support cognitively bounded rational analysis in the CORE modeling tool (Howes et al., in revision, 2004). CORE (for Constraint-based Optimizing Reasoning Engine) supports generation of strategies through the Information Requirements Grammar (IRG) formalism (Howes, Lewis, Vera, & Richardson, 2005). IRG is a generative grammar through which the modeler specifies a set of rules for successively decomposing tasks into component subtasks (an example is shown in Figure 3). The modeler may specify multiple rules to decompose the same task or subtask to reflect strategic variation in how that task may be accomplished. In the task hierarchy there may be many such subtasks with variable decompositions, allowing multiple dimensions of strategic variability to be explored simultaneously. Also, within the task rules the modeler may specify branches contingent on stochastic events (e.g. whether a typing error is detected). Finally, the task rules contain constraints on the order of subtasks, which limit the space of possible strategies.

The modeler specifies the architecture by defining the low-level processes (e.g. eye fixations, action selections) in which the task expansions bottom out, and by specifying the cognitive resources (e.g. vision, central executive) used by these processes. The resources used by the low-level processes determine which processes can run in parallel and which must be serialized.

The strategies are instantiated in *schedules* of the task's component processes generated by CORE (see Figure 4 for examples), and then evaluated according to a utility function specified by the modeler. The modeler can then compare the utilities of different strategies of interest, and identify the optimal strategy in the strategy space. For instance, the utility function could optimize for speed, accuracy, working memory load, bonus points (if modeling an experiment with a performance-based payoff), or a combination of such factors.

The Task

We modeled and collected data on a task that was studied in Nielsen and Phillips (1993), and subsequently modeled in John (1994), Salvucci and Lee (2003), and John, Prevas, Salvucci, and Koedinger (2004). In the task, subjects interact with a simulation of a database program (shown in Figure 1). The task involves querying the database for one to three seven-digit telephone numbers. The task includes many steps involving selection from menus and mousing on widgets (see Nielsen and Phillips (1993) for details) but in the present paper we focus on part of the task in which the subject transcribes a number shown on the screen into a text box.

Dimensions of Strategic Variability

The task illustrates three types of strategic variability which occur generally in skilled behavior: variability in task decomposition, variability in parallelization of subtasks, and variable adaptation to error correction costs.

Task Decomposition In transcribing the seven digits, people have a choice of how to break up the task: they may encode all seven digits and then type them from memory, or they may break the number into groups and switch between encoding and typing each group. Grouping incurs a task-switching cost in that the subject will have to switch their attention and their eyes more frequently between typing and encoding. Given the task-switching cost, it would never make sense for people to group—unless there are countervailing costs of *not* grouping. One possibility is that not grouping may incur a cost in working memory load. The modeling work presented in this paper suggests an alternative possibility, that by not grouping people would forfeit opportunities to parallelize components of the task.

Strategic variation in task decomposition is found in a wide range of tasks; for example, Brumby, Howes, and Salvucci (2007), Gray, Sims, Fu, and Schoelles (2006), and Salvucci, Taatgen, and Kushleyeva (2006) have also modeled strategic tradeoffs involved in grouping.

Parallelization of Subtasks If the digits are grouped, then people have the option of typing a group in parallel with encoding the following group. Parallelizing processes may save time, but may incur costs: for instance, processes may interfere with each other, or scheduling processes early may cause responses to be made out of order (Meyer and Kieras (1997) contains a detailed account of strategic variation in parallelization).

Strategic Adaptation to Error Correction Costs In the task, people will sometimes make typing errors, and will either have to detect and correct the errors, or incur some cost from leaving them uncorrected. People have a strategic choice of how frequently to check what they've typed, which is subject to a tradeoff: the more frequently one checks for errors, the sooner errors will be caught and the less time will be required for correcting them; however, one incurs a cost in time for each check performed.

Other modeling work that has included strategic adaptation to error includes Meyer and Kieras (1997), Salvucci et al. (2006), and Sperling and Dosher (1986). However, our model

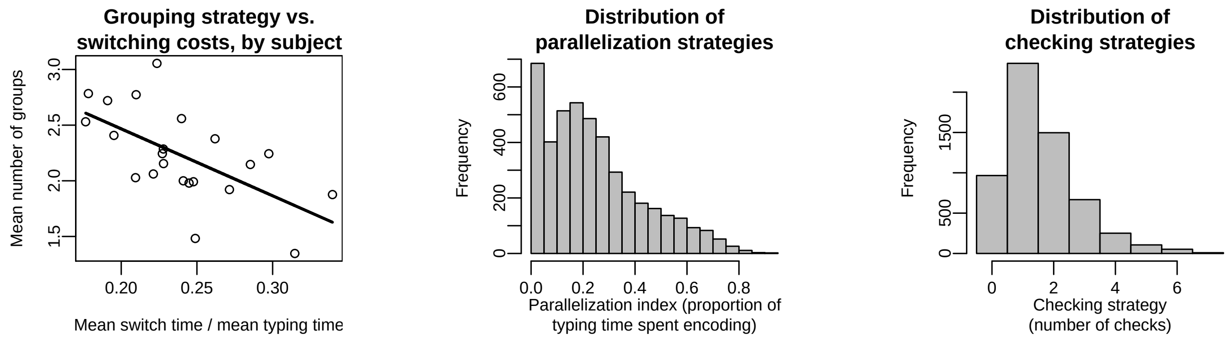

Figure 2: Observed strategic variation in typing task, along three dimensions: grouping, parallelization, and checking.

distinguishes itself from previous approaches in that it both directly incorporates *error recovery routines* (e.g. backspacing and retyping) and factors these into strategy prediction.

The Experiment

Method

We collected eyetracking, mousing, and keystroke data on this task from 22 subjects. The experiment consisted of two sessions, each of which contained 81 trials divided into nine blocks. Each block contained an equal number of one-, two-, and three-number query tasks. Subjects were awarded bonus points for each trial depending on their speed and accuracy in completing the task: $150 - 15(RT - 4)/Q$ points for correct trials, where RT is the time to complete the task and Q is the number of queries; or -150 points for incorrect trials (e.g. an error in the phone number entered).

Results

For each phone number typed we used the eyetracking data to compute the number of groups into which the subject broke the phone number. Each look at the phone number that was followed by a look at the hands or text box and the typing of at least one digit was counted as a group. For each subject we computed the mean number of groups the subject used, to get a measure of the subject's grouping strategy preference. Subjects' mean grouping strategies ranged from 1.3 groups to 3.1 groups, with an overall mean of 2.2 groups.

In addition to variations in typing speed, subjects varied in the time they took to initiate typing after encoding. (See Larochelle (1983) for a study of variations in typing with skill level). Our preliminary work on the model presented in this paper suggested the hypothesis that subjects' grouping preferences might be affected by the cost of switching from encoding to typing: subjects with high switch costs would be expected to use fewer groups so as to switch less frequently. (Gray et al. (2006) report a related study of the effects of switch costs on grouping.) To test this hypothesis, for each subject we calculated the average time between the end of encoding and the beginning of typing of each group. We then normalized these mean switch times by the subjects' mean typing time. Subjects' grouping preferences were significantly correlated with their normalized mean switch time ($p < 0.005$); the relationship is shown in the left panel of Figure 2.

For each phone number typed, we computed a parallelization index defined as the time encoding and typing were occurring in parallel divided by the time spent typing. The distribution of this parallelization index across all subjects is shown in the center panel in Figure 2. For 10% of the phone numbers typed, there was no parallelization; for the remainder, the mean parallelization index was 0.27.

For each phone number typed we also recorded the error monitoring strategy employed: how many of the seven digits typed were accompanied by a look at the text box (presumably to monitor the accuracy of what had been typed). Most often, subjects monitored the number once (usually at the end of the number). The distribution of error monitoring strategies is shown in the right panel of Figure 2.

Model Description

Architecture specification

The model incorporated low-level cognitive processes, and cognitive resources occupied by those processes, based on the ACT-R cognitive architecture (Anderson et al., 2004). The architecture specification duplicated that of earlier CORE models which emulated ACT-R (Howes et al., in revision, 2004). Cognitive resources in the architecture included separate motor and visual resources, allowing visual and motor processes to occur in parallel, and a cognition resource whose capacity was limited to a single process at a time.

In addition to the ACT-R processes, we defined some processes specialized for the typing task, whose durations were estimated from the data. These included a visual encoding process, a cognition process for initiating a new group of keystrokes, and a cognition process for initiating individual keystrokes within groups. We varied the duration of the cognition group-initiation process to model the variability in subjects' encoding-to-typing task switching times (seen in Figure 2, left panel) and to explore the effects of this variation on strategy selection. The distinction between group-initiation and keystroke-initiation processes, and the between-subject

```
transcribe STRING - CURSOR                - DONE
  -->
  encode STRING                           - ENCODED,
  type STRING after ENCODED and CURSOR    - DONE.

transcribe STRING - CURSOR                - DONE
  -->
  parameter(concatenate(GROUP1 GROUP2) STRING),
  encode GROUP1                           - ENCODED1,
  type GROUP1 after ENCODED1 and CURSOR   - DONE1,
  transcribe GROUP2 - DONE1               - DONE.
```

Figure 3: IRG rules specifying the strategy space for grouping digits to be transcribed.

variability in group-initiation times, were also suggested by Larochelle (1983). Finally, a parameter representing frequency of typing errors per keystroke was also estimated from the data. Note that these parameters were not chosen to maximize the fit of the model's predictions to the data, but were measured directly from specific quantities in the data (e.g. frequency of typing errors) independent of the model's predictions (see (Howes et al., in revision) for an in-depth discussion of model calibration from data).

Strategy generation

Our model of the task sought to capture all combinations of grouping, parallelizing, and checking strategies.

Grouping The model assumed that subjects may either follow a strategy of encoding the entire number and then typing it, or may follow strategies involving breaking the number into any number of groups of any size, and encoding and typing each group individually. To specify this strategy space in CORE, we used recursive IRG rules, shown in Figure 3. (For clarity, these rules are simplified from the actual code used in the model.) These rules specify the possible expansions of `transcribe` into `encode` and `type` subtasks. Each possible expansion of `transcribe` corresponds to a different strategy for grouping the digits. For instance, expanding `transcribe` directly into `encode` and `type` by the first rule corresponds to a strategy of not breaking the number into groups; expanding `transcribe` into `encode`, `type`, and `transcribe` (second rule) and then expanding the second `transcribe` into `encode` and `type` (first rule) corresponds to a strategy of breaking the number into two groups; and so forth. (The `parameter` keyword in the second rule is used to establish constraints, in this case that two substrings must add up to the string being transcribed.)

Parallelization The model assumed that if a number was broken into groups, subjects may follow a range of strategies for parallelizing, serializing, or interleaving the processes comprising the encoding and typing of each group. In CORE, this flexibility emerged from the composition of the `encode` and `type` subtasks partly from low-level processes which use separate vision and motor resources by our architectural assumptions. CORE is free to schedule such processes at overlapping times, or not, depending on what is optimal relative to the objective function. The strategy space for parallelization is not totally unconstrained however: some of component processes of the `type` and `encode` subtasks share the cognition resource, and cannot execute in parallel.

In addition to such architectural constraints, there are constraints imposed by the task: for instance, typing of a group cannot begin before encoding of that group is completed, or before typing of the previous group is completed. Such task constraints are enforced by specifying information flows, represented by variables (distinguished by capital letters) passed between subtasks as arguments. For example, in the first rule, passing the last argument of `encode`, `ENCODED`, to `type` as its second argument will constrain typing to begin after encoding is finished. Arguments of subtasks are, through the subtasks' expansions, grounded in the inputs and outputs of specific low-level processes, so that passing the last argument of `encode` to the second argument of `type` has the concrete effect of binding the output of a component process of `encode` to the input of a component process of `type`, constraining the second process to follow the first. On the other hand, the rules in Figure 3 contain no constraints on when the component processes of `encode` begin, leaving CORE free to schedule encoding of a group before the completion of typing of the preceding group.

Error monitoring Our model assumed that after each of the seven digits typed, subjects may either check what they've typed up to that point, or not check, resulting in a space of 2^7 or 128 strategies for checking, ranging from not checking at all to checking after each digit is typed. To implement this in IRG, the task expansion of the `type` tasks included a `monitor` task for each digit typed, which had two possible expansions. One expansion was trivial, "do nothing". The other expansion involved looking at the digits typed on the screen and then branching: if an error was committed, press the backspace key until the error is reached, and then retype; if no error was committed, do nothing further. In this way, the costs incurred by errors was implicitly included in the task specification.

Utility function specification

We set the utility function to be equal to the bonus point payoff function used in the experiment. The strategies in the space described above were then evaluated by CORE according to this utility function.

Model Results

Predicted strategy The optimal strategy predicted by the model is shown in the top panel of Figure 4. The horizontal bars in the figure show the cognitive processes comprising task execution, with the component processes of encode, type, and error monitoring subtasks shown in green, blue, and red, respectively. The optimal strategy had the following features:

(a) Numbers were broken into two groups;

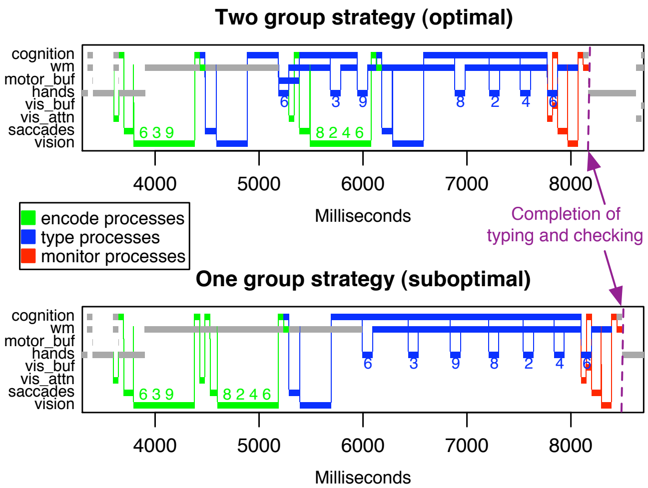

Figure 4: Schedules of two-group and one-group check-once strategies for typing the number ”6398246”. Keystroke execution and eye fixation processes are labeled with the digits being typed / encoded. Parts of the task preceding and following typing are not shown.

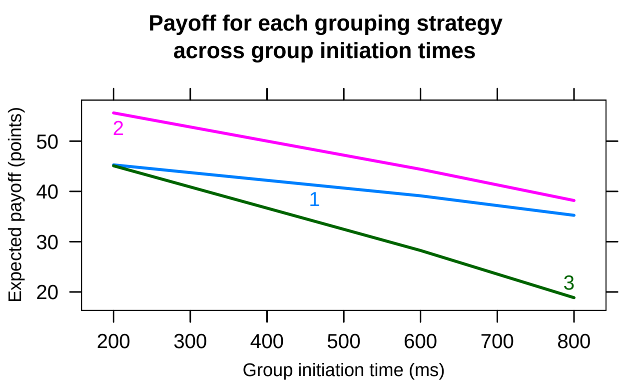

Figure 5: Grouping strategy utilities for different values of the group initiation time parameter in the model.

(b) The typing of the second group occurred in parallel with encoding of the first group;

(c) The number was checked once, at the end.

(a), (b) and (c) were in agreement with the strategies most commonly followed by subjects, as shown in the left, center, and right panels, respectively, of Figure 2.

Interactions between grouping, parallelization, and checking The bottom panel of Figure 4 shows the best of the one-group strategies considered by CORE. The contrast between the one- and two- group strategies illustrates how the two-group strategy saves time through the parallelization of encoding processes (blue) with typing processes (green); this benefit outweighs extra costs incurred by switching between encoding and typing. On the other hand, for the three-group strategy (not shown), diminishing returns from parallelization

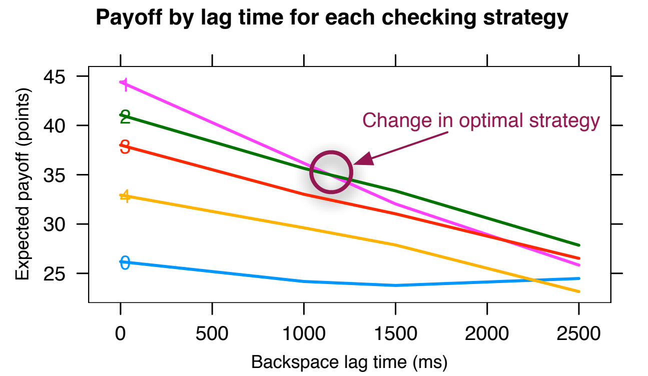

Figure 6: Checking strategy utilities for different backspace lag times in the model.

of smaller groups are outweighed by the extra switch costs. If the model was restricted to only consider serial strategies, then the one-group strategy became optimal. In contrast to the parallelization dimension, the checking dimension of the strategy space was independent of the grouping dimension in that the grouping strategies retained the same utilities relative to each other if checking was omitted from the model; also, the check-once strategy was optimal regardless of how the number was grouped.

Varying switch costs The optimal strategy remained consistently optimal when we varied the duration of the process for initiating typing of a new group (thereby varying the encoding-to-typing switch cost). However, the utility of the different grouping strategies varied depending on this group initiation time parameter; Figure 5 shows the expected payoff for each grouping strategy for different values of the parameter. The graph shows the one-, two- and three-group strategies with the highest utility. At lower group initiation times, the three-group strategy becomes less suboptimal relative to the two-group strategy, whereas at higher typing initiation times, the one-group strategy becomes less suboptimal compared to the two-group strategy. If the model's group initiation time is taken to include the variable task switching processes observed in subjects, then the predicted effect of group initiation time on the relative utility of strategies can explain the observed variation in subject's strategy preferences as a function of their switch times, as shown in Figure 2 (left panel).

Varying error correction costs The model's prediction of a check-once strategy emerged from a tradeoff between costs of checking for errors, and costs of recovering from errors or of failing to detect errors. To further explore the role of the error recovery routines in strategy selection, we ran variations of the model in which we varied the time necessary to execute each backspace (i.e., adding lag to the backspace key). Figure 6 shows the predicted payoff for each checking strategy for different values of the lag parameter. The best one-check, two-check, three-check, four-check, and no-check strategies at each level of lag are shown. A lag of 0 corresponds to the original model of the experiment described above. At higher levels of lag, the strategy involving one check loses in utility relative to the strategies involving more checks, with the two-check strategy becoming optimal when the lag is greater than one second. This is because under the multiple-check strategies, errors are caught sooner on average than they are under the one-check strategy, so that the backspace key needs to be used less.

Conclusion

In this paper we have demonstrated the application of cognitively bounded rational analysis to the problem of modeling and predicting behavior on tasks with large, multidimensional strategy spaces. The approach was applied to modeling experimental data from a typing task, with a focus on three dimensions of strategic variability: decomposition of tasks into subtasks, parallel vs. serial processing of subtasks, and control of errors. The model used a generative grammar formalism, IRG, to efficiently specify the strategy space, and the actual strategies employed were predicted by optimizing over the strategy space with respect to a utility function identical to the payoff used in the experiment. The model predicted a strategy consistent with the grouping, parallelization, and error monitoring strategies used by subjects, as well as predicting interactions among these dimensions; in particular, observed grouping strategy preferences were explained as emerging from a tradeoff between task switching costs and parallelization benefits.

In addition, we have demonstrated a new way of producing models which directly incorporate error recovery routines, and which predict strategic adaptation under the costs incurred by error recovery. One immediate goal for future work is to validate this capability by running experiments to observe strategy choices under varying error correction costs. However, this same approach can be applied to modeling any kind of stochasticity internal or external to the subject. In particular we are currently applying this approach to modeling strategic adaptation to task interruptions, and to rare emergencies.

Acknowledgments

We would like to thank Bonnie John for helpful discussions about the task, and Danielle Elskens, Cortney Sera, Bradford Terry, and Amy Woodward for assisting with data collection. This work was supported by NASA Awards NNA05CS65A and NNA04CK14A, and Office of Naval Research Award N000140310087.

References

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

Brumby, D. P., Howes, A., & Salvucci, D. D. (2007). A cognitive constraint model of dual-task trade-offs in a highly dynamic driving task. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 233–242). New York, NY, USA: ACM Press.

Gray, W. D., Sims, C. R., Fu, W. T., & Schoelles, M. J. (2006). The soft constraints hypothesis: A rational analysis approach to resource allocation for interactive behavior. *Psychological Review*, *113*(3), 461–482.

Howes, A., Lewis, R. L., & Vera, A. (in revision). Cognitively bounded rational analysis. *Psychological Review*.

Howes, A., Lewis, R. L., Vera, A., & Richardson, J. (2005). Information-requirements grammar: A theory of the structure of competence for interaction. *Proceedings of the 27th Annual Meeting of the Cognitive Science Society*, 977–983.

Howes, A., Vera, A., Lewis, R., & McCurdy, M. (2004). Cognitive constraint modeling: A formal approach to supporting reasoning about behavior. In *Proceedings of the 26th Annual Meeting of the Cognitive Science Society* (pp. 595–600). Hillsdale, NJ, USA: Lawrence Erlbaum Associates.

John, B. E. (1994). Toward a deeper comparison of methods: A reaction to Nielsen and Phillips and new data. In *Proceedings companion of CHI 1994* (pp. 285–286). New York, NY, USA: ACM Press.

John, B. E., Prevas, K., Salvucci, D. D., & Koedinger, K. (2004). Predictive human performance modeling made easy. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 455–462). New York, NY, USA: ACM Press.

Larochelle, S. (1983). A comparison of skilled and novice performance in discontinuous typing. In W. E. Cooper (Ed.), *Cognitive aspects of skilled typewriting.* New York, NY, USA: Springer-Verlag.

Meyer, D. E., & Kieras, D. E. (1997). A computational theory of executive cognitive processes and multiple-task performance: Part 1. Basic mechanisms. *Psychological Review*, *104*(1), 3–65.

Nielsen, J., & Phillips, V. (1993). Estimating the relative usability of two interfaces: Heuristic, formal, and empirical methods compared. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 214–221). New York, NY, USA: ACM Press.

Salvucci, D. D., & Lee, F. J. (2003). Simple cognitive modeling in a complex cognitive architecture. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 265–272). New York, NY, USA: ACM Press.

Salvucci, D. D., Taatgen, N. A., & Kushleyeva, Y. (2006). Learning when to switch tasks in a dynamic multitasking environment. In *Proceedings of the Seventh International Conference on Cognitive Modeling* (pp. 268–273). Oxford, UK: Taylor & Francis/Psychology Press.

Sperling, G., & Dosher, B. A. (1986). Strategy and optimization in human information processing. In *Handbook of perception and human performance: Volume 1. Sensory processes and perception.* New York, NY, USA: John Wiley and Sons.