Electronic Communications of the EASST
Volume 45 (2011)

Proceedings of the Fourth International Workshop on Formal Methods for Interactive Systems (FMIS 2011)

Abstract Models and Cognitive Mismatch in Formal Verification

Rimvydas Rukšėnas and Paul Curzon

5 pages

Guest Editors: Judy Bowen, Steve Reeves
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Abstract Models and Cognitive Mismatch in Formal Verification

Rimvydas Rukšėnas and Paul Curzon

r.ruksenas@eecs.qmul.ac.uk, paul.curzon@eecs.qmul.ac.uk
School of Electronic Engineering and Computer Science
Queen Mary University of London, United Kingdom

Abstract: We present ongoing work to accommodate fine-grained analysis of interactive systems via model checking. We argue that this can be achieved by combining a basic abstract model of user behaviour and a separate constraint on the acceptable degree of cognitive mismatch. To explain the problem and illustrate our approach, we present a simple scenario related to number entry in infusion pumps.

Keywords: User models, cognitive mismatch, model checking, number entry

1 Introduction

In HCI, models of human cognition are used in different ways and for various purposes. One approach, computational cognitive models [RY01], draws from cognitive science and is applied to test and improve usability of interfaces, e.g., by predicting behaviour, time [JK96] and error [Gra00]. The analysis of system properties is based on individual simulation runs, meaning that the essentially deterministic models predict likely user behaviours and their properties. An alternative approach (e.g., [Rus01]), drawn from formal methods, focuses on safety-critical aspects of interactive behaviour. Compared to simulation-based approaches, the models are abstract and highly non-deterministic. This generates a wide range of behaviours and allows exhaustive exploration of the consequences of all possible model behaviours using automatic tools such as model checkers.

Models of cognition usually include consideration of the capabilities and limitations of the user. One aspect to consider is the human capability to create mental models of the device and interaction, and then to rely on those models to a certain extent while interacting with the device. Such mental models can be based on device manuals, training or simply previous experience with similar devices. They lead to certain expectations and assumptions about the device behaviour. Depending on situation, these expectations and assumptions may or may not be justified. This potentially leads to *cognitive mismatches* between the mental model and the actual state of the system, which can result in erroneous user actions [Rus01].

There are at least two ways of dealing with cognitive mismatches in interactive systems. The most obvious way is akin to the *safety* view of the reliability of computer systems. It tries to detect cognitive mismatches and fix them by modifying interfaces (e.g., Rushby's work on mode confusions [Rus02]). Though this is an essential and useful thing to do, in general, one cannot assume that users will form correct mental models, even when all the required information is present on the device interface or within the interactive system more generally. This might be due to high cognitive load, interruptions or other facets of the system being more salient, for

example. Consequently, the alternate *resilience* view to system safety accepts that cognitive mismatches are unavoidable and tries to safeguard against them.

We have developed a generic model of cognitively plausible behaviour based on empirical findings [RBCB09]. The model was applied, e.g., to perform GOMS-style analysis of timing [RCBB08] using the SAL model checker. In all our earlier work, the formalisations of cognitively plausible behaviour include built-in assumptions about how cognitive mismatches are handled. Consequently, the outcomes of our analyses were relative to those assumptions. A problem with this is agreeing on what assumptions are *generally* plausible, since usually cognitive plausibility is *situated*, i.e., dependent on the task, its context and on behavioural strategies. Including such assumptions into user models also makes their inspection and validation less tractable.

Here we consider an approach where assumptions on cognitive mismatch are split from the user model and illustrate the approach on a simple scenario related to number entry in infusion pumps. Infusion pumps require number input giving the rate of infusion and the volume to be infused. They use different styles of number entry. We focus on one incremental style using chevron buttons, though similar issues could arise when modelling interaction with pumps relying on other styles of number entry.

2 Motivation

Plausible behaviours Here we sketch how to develop a basic abstract model of plausible user interactions with an infusion pump. We focus on user beliefs related to number entry. Consider the subtask of setting the required rate of infusion. Let us assume, our infusion pump has four chevron buttons to perform this task (similarly as the Alaris family of pumps). Pressing the 'up' single chevron increases the number entered by a certain amount, say $delta$ in our model, whereas pressing the 'up' double chevron increases it in larger jumps, say $Delta$. The 'down' chevrons operate in the opposite way.

The users of the pump will have their own beliefs about these delta values. Let us denote them $mdelta$ and $mDelta$, respectively. Though these beliefs might be right in most cases, there is no reason to assume that a cognitive mismatch is impossible, especially, since the interface does not show the actual delta values. Furthermore, they are not constant: when the number increases, the delta values are increased at certain points too.

The infusion rate, $rate$, is displayed on the pump screen as the number is incrementally entered using the chevrons. Though the user can check it on the display, there is no guarantee of them doing so on each button press. Hence, we have another variable, $mrate$, to represent the user's belief about the current rate. To increase the rate value, the users can perform either the 'small rate increase' or 'big rate increase' action. Let us consider the latter, associated with pressing the 'up' double chevron, and specify state transitions related to the relevant belief updates. Here, the relevant beliefs are represented by variables $mrate$ and $mDelta$.

For $mDelta$, there are two plausible alternatives. The user model may assume that the delta value is unchanged ($mDelta' = mDelta$), or it may correct a wrong belief by updating $mDelta$ with the actual delta value ($mDelta' = Delta$). Likewise, two alternatives associated with the 'big rate increase' action are plausible for the user's beliefs about the infusion rate. The model may

check the current rate value on the display and update their belief, increasing it by the assumed delta value ($mrate' = rate + mDelta$). Alternatively, it may rely on the rate believed to be current and increase it similarly ($mrate' = mrate + mDelta$). Similar specifications could be given for the 'small rate increase' action and the corresponding rate decrease actions, as well as for the actions associated with the volume to be infused. Note that, at this level of abstraction, we do not postulate how and when such an update may occur, for which there could be many triggers.

Next, we discuss issues that may arise when such abstract model is used for the formal analysis of user interaction with the infusion pump.

Issues Being underspecified, the above model can generate extremely large numbers of behaviours: the 'big rate increase' action as specified above can lead to 4 alternative transitions in each step (2 for $mDelta$ multiplied by 2 for $mrate$). The number of such steps can be large if anything close to the actual range of numbers in infusion pumps (e.g., 0 – 9999) is used in the model. Furthermore, the model would include other plausible number changing actions that could occur at the same time. Each will in turn generate a similar number of different transitions.

The automatic checking of liveness properties ("something useful is eventually done"), is complicated by having huge numbers of behaviours. The following example liveness property is relevant to our scenario: "The user model combined with the pump model, in all possible scenarios, eventually sets the target rate of infusion within the specified time bound." Formally, this can be specified in LTL (linear temporal logic) using the temporal operator Until as follows:

$$(t < T_{max}) \text{ Until } (rate = target)$$

where T_{max} is a time bound, and $target$ is a target rate.

Model checking properties of this kind in a practically useful time is problematic for abstract models as above. Furthermore, the above property will most likely be falsified for any reasonable bound T_{max} because of the model generating behaviours that, though plausible in principle, are unlikely. For example, as long as the beliefs represented by $mrate$ and $mDelta$ remain wrong, the user model can keep performing the 'big rate increase' action, even though the actual rate value may already have been higher than the target rate for many steps. Obviously, such behaviour is unrealistic; even more so, when the pump user is a nurse trained to use infusion pumps.

3 Cognitive Mismatches and Formal Analysis

The above model is simple and, therefore, easy to inspect and reason about. It assumes little about pump users, which means that a very wide range of plausible behaviours are generated by it. As such, it is appropriate for the analysis of safety properties ("something bad never happens") related to user interaction. An example of such a property for our scenario is the following one: "The user model never confirms a wrong rate of infusion."

Setting parameters and starting infusion quickly may be critical in, e.g., operating theatres. As discussed above, our abstract model is less feasible for analysing liveness properties such as timing. Although performance of interactive systems can be analysed by other means [JK96], formal timing verification could provide stronger guarantees that critical interactions are sufficiently efficient by exploring a range of plausible but not necessarily optimal behaviours.

In the above model, a major source of the many behaviours is cognitive mismatch. Realistically, cognitive mismatches are unavoidable, temporarily at least. Therefore, a formal model should not rule them out completely. When analysing liveness properties such as timing, however, it is reasonable to impose certain constraints on the degree of cognitive mismatch assumed by the model. For our scenario, an example of such a constraint could be as follows: "If the belief maintained by the model about the rate value is correct ($mrate = rate$), then the belief about the delta value is correct too ($mDelta = Delta$)." One can also postulate that the difference between the actual rate value and user belief is maintained within a certain limit, e.g.:

$$rate - mrate < k \times Delta$$

Intuitively, this states that the user model corrects a wrong belief about the rate no later than performing the 'big rate increase' action for the k-th time in succession.

While constraints as above are plausible for the analysis of efficiency, they are an instance of *situated* assumptions. For example, it would be generally impossible to justify any specific (and not unreasonably large) k value in the above constraint. Therefore, incorporating such situated constraints into the user model could compromise the verification of safety properties, when the same model is used for that. A formal model of plausible behaviour explicitly represents assumptions that must be consistent with actual users for the properties proved to be useful and meaningful in assessment of an interactive system. At the same time, for keeping track of the assumptions made and their inspection and validation, it is advantageous to work with the same set of assumptions (user model) when checking all the relevant properties.

We tackle the dichotomy between generality and situatedness by combining two ingredients: a basic abstract model of user behaviour and a separate constraint on the acceptable degree of cognitive mismatch. To specify this constraint, we define a measure of cognitive mismatch. For example, in our scenario, this measure, say d, could be defined as

$$|rate - mrate| \text{ div } Delta$$

The user model is fixed and captures generic assumptions about the users of a specific device. Constraints on cognitive mismatch are part of the properties analysed and can vary between properties. For example, the earlier property for timing analysis could be restated as follows:

$$(t < T_{max}) \text{ Until } (rate = target \vee \neg(d < k))$$

where $d < k$ is the constraint on mismatch just defined. Intuitively, behaviours that violate this constraint are pruned during model checking when the modified property is used.

4 Conclusions

The separation of the user model and the constraint on cognitive mismatches offers several advantages. A fixed user model supports consistency throughout the analysis of different aspects of a system. Constraints on cognitive mismatch, on the other hand, provide flexibility: they can vary depending on the properties explored. Furthermore, the correlation between the assumptions made and the property verified remains explicit, thus easier to analyse. In this way, one can

explore, for example, the effect that different degrees of the allowed cognitive mismatch make on timing, or the ability of the user model to achieve a task goal.

Acknowledgements: Funded by two EPSRC research grants: CHI+MED (Computer–Human Interaction for Medical Devices), EP/G059063/1, and Extreme Reasoning, EP/F02309X/1.

Bibliography

[Gra00] W. Gray. The nature and processing of errors in interactive behavior. *Cognitive Science* 24(2):205–248, 2000.
doi:10.1016/S0364-0213(00)00022-7

[JK96] B. E. John, D. E. Kieras. Using GOMS for user interface design and evaluation: which technique? *ACM Trans. Comput.-Hum. Interact.* 3:287–319, 1996.
doi:10.1145/235833.236050

[RBCB09] R. Rukšėnas, J. Back, P. Curzon, A. Blandford. Verification-guided modelling of salience and cognitive load. *Formal Aspects of Computing* 21:541–569, 2009.
doi:10.1007/s00165-008-0102-7

[RCBB08] R. Rukšėnas, P. Curzon, A. Blandford, J. Back. Combining Human Error Verification and Timing Analysis. In Gulliksen et al. (eds.), *Engineering Interactive Systems*. Lecture Notes in Computer Science 4940, pp. 18–35. Springer Berlin / Heidelberg, 2008.
doi:10.1007/978-3-540-92698-6_2

[Rus01] J. Rushby. Analyzing Cockpit Interfaces Using Formal Methods. *Electronic Notes in Theoretical Computer Science* 43, 2001.
doi:10.1016/S1571-0661(04)80891-0

[Rus02] J. Rushby. Using Model Checking to Help Discover Mode Confusions and other Automation Surprises. *Reliability Engineering and System Safety* 75(2):167–177, 2002.
doi:10.1016/S0951-8320(01)00092-8

[RY01] F. E. Ritter, R. M. Young. Embodied models as simulated users: introduction to this special issue on using cognitive models to improve interface design. *Int. J. Human-Computer Studies* 55:1–14, 2001.
doi:10.1145/320719.322590