A Petri-Net Based Approach to Measure the Learnability of Interactive Systems

Obaida Hanteer
Sapienza Università di Roma
obaida9hant@gmail.com

Andrea Marrella
Sapienza Università di Roma
marrella@dis.uniroma1.it

Massimo Mecella
Sapienza Università di Roma
mecella@dis.uniroma1.it

Tiziana Catarci
Sapienza Università di Roma
catarci@dis.uniroma1.it

ABSTRACT

We propose an approach to measure the learnability of an interactive system. Our approach relies on recording in a *user log* all the user actions that take place during a run of the system and on replaying them over one or more *interaction models* of the system. Each interaction model describes the expected way of executing a relevant task provided by the system. The proposed approach is able to identify deviations between the interaction models and the user log and to assess the weight of such deviations through a *fitness* value, which estimates how much a log adheres to the models. Our thesis is that by measuring the rate of such a fitness value for subsequent executions of the system we can not only understand if the system is learnable with respect to its relevant tasks, but also to identify potential learning issues.

CCS Concepts

•**Human-centered computing** → **HCI design and evaluation methods;** *Usability testing; HCI theory, concepts and models;*

1. INTRODUCTION

In the Human-Computer Interaction (HCI) community, *learnability* is recognized as one of the most relevant components of usability [1]. Learnability is addressed in ISO 9126-1 as *"the capability of the software product to enable the user to learn its application"* and applies to the nature of the performance change of a user when interacting with a system. A highly learnable system should allow a user to know *how to perform correctly* any relevant task of the system after having executed it a few times in the past.

Starting from this general idea of learnability, the work [2] surveys the existing learnability research spanning over the past four decades, and shows that there is no consistent agreement on how learnability of a system should be defined, evaluated and improved. This lack of consensus has led to

a lack of well-accepted metrics for measuring learnability; several metrics exist, but they are limited to measure specific aspects of the interaction. A classification of those metrics is presented in [2]. Some of them suffer from *evaluation subjectivity*, as they rely on associating learnability to the "quality" of the interaction by analyzing the user feedbacks *after* the interaction has happened. Others are *quantitative* metrics. They measure *objectively* the performances of a user executing a relevant task (e.g., completion times, error rates, etc.), but they are often performed in (controlled) lab environments under the guidance of an external evaluator.

In this paper, we customize an existing approach to perform conformance checking in Business Process Management [5] in order to *objectively quantify* the learnability of a system during its daily use. Our approach is based on verifying whether the user's "observed" behavior, which is recorded in a specific *user log* and reflects the user actions performed during the interaction with the system, matches the "intended" behavior represented as a *model of the interaction*. To tackle this issue, we introduce the concept of *alignment*, which is a pairwise matching between user actions recorded in the log and actions allowed by the interaction model. A perfect alignment between the log and the model is not always possible, thus making deviations be identified. Such deviations may reflect mistakes or slips made by the user during the interaction with the system, or may simply represent different strategies (with respect to the ones expected) that a user follows to accomplish a task. The result of the alignment activity is the so-called *fitness value*, i.e., how much the log adheres to a model of the interaction. The fitness value can vary from 0 to 1, where 1 means a perfect matching between the log and the interaction model, i.e., the user was able to accomplish her/his task as foreseen in the corresponding interaction model.

Our thesis is that in order to measure the learnability of a system we can analyze the rate of the fitness values corresponding to subsequent executions of the system over time. An increasing rate corresponds to a system that is easy to be learnt with respect to its relevant tasks. Vice versa, given an initial low fitness value, a not-increasing or stable rate may indicate the presence of some learning issues.

2. THE GENERAL APPROACH

Firstly, it is required to make explicit the *dialog* (i.e., the *interaction model*) between the user and the system. In the HCI literature, several notations and languages have
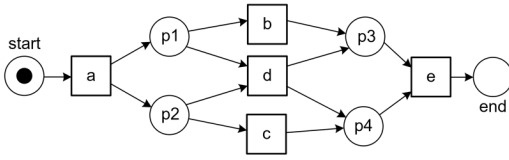
Figure 1: The Petri net used as running example.

$$\tau_1 = \begin{array}{|c|c|c|c|} a & c & b & e \\ \hline a & c & b & e \end{array} \qquad \tau_2 = \begin{array}{|c|c|c|c|c|} a & b & d & c & * \\ \hline a & b & * & c & e \end{array}$$

Figure 2: Alignment of traces $\tau_1 = \langle a, c, b, e \rangle$ and $\tau_2 = \langle a, b, d, c \rangle$ against the interaction model in Fig. 1.

been proposed for describing human-computer dialogs [1], but they are often characterized by an ambiguous semantics. However, to realize our technique we opted for *Petri nets* [3], which have a clear semantics and have proven to be adequate for modelling human-computer dialogs [4].

A Petri net is a directed graph with two nodes called *places* and *transitions*. Places are represented by circles and transitions by rectangles. At any time, a place can contain zero or more *tokens*, drawn as black dots. The state of a Petri net, a.k.a. *marking*, is determined by the number of tokens in places. In any run of a Petri net, the number of tokens in places may change, i.e., the Petri net marking can evolve according to the following firing rule: if all the places with arcs going into a transition have a token, then the transition can *fire* and all the tokens from the input places are removed and new tokens are placed on the outputs of the transition.

When Petri nets are used to represent interaction models, transitions are associated with the user actions (windows opened, check boxes clicked, text entered, etc.) required to accomplish a relevant task of the system (e.g., the purchase of a train ticket). Therefore, we can define as many interaction models as are the relevant tasks of the system. In Fig. 1 we show a Petri net that depicts an interaction model of a fictitious relevant task. We suppose that transitions a, b, c, d, e are associated with user actions of the same name. The marking with one token in place $start/end$ (and no tokens in any other place) is the *initial/final* marking.

Secondly, the system to be evaluated must capture the user actions taking place during a single execution of the system itself and record them in a *user log*. Hence, a user log consists of a sequence of user actions potentially related to the execution of several relevant tasks. Given a specific interaction model describing the execution path to accomplish a single relevant task, our approach allows to extract only those fragments of a user log (called *traces*) that are related to the execution of the relevant task under consideration. To perform correctly such an extraction it is important that the actions in the user logs are related to transitions in the interaction model, and vice versa. For example, given the interaction model of Fig. 1, two possible traces extracted from a user log can be $\tau_1 = \langle a, c, b, e \rangle$ and $\tau_2 = \langle a, b, d, c \rangle$. In the context of a single run of the system a relevant task can be executed several times, i.e., several traces related to the relevant task may be extracted from the user log.

Thirdly, given a set of user logs (representing subsequent executions of the system) and a relevant task for which we developed an interaction model, we can construct the *alignment* between any of the traces extracted from these logs and the interaction model. The alignment activity consists of *replaying* the extracted traces over the interaction model. We basically need to relate "moves" in a trace to "moves" in the interaction model. A move (x, y) is said to be *synchronous* if the action x in the trace can be replayed by the transition y of the interaction model. For example, as shown

in Fig. 2 (note that moves are represented vertically), all the moves of τ_1 are synchronous. If some of the moves in the trace can not be replayed by the interaction model (and vice versa), we call them "no moves". They are explicitly denoted by $'*'$ and allow us to identify where deviations occur, e.g., the third/fifth move of τ_2 is a move in trace/model, resp.

In order to define the severity of a deviation, we exploit a cost function on trace/model moves, which can be used to favor one type of explanation for deviations over the other, and depends on the complexity of the interaction model to be investigated. In general, several possible alignments exist between an interaction model and a trace, since there may exist manifold explanations why a specific interaction with the system is not conforming. The alignment with the *lowest cost* allows us to infer the *fitness value*, i.e., the extent to which the traces of a user log can be associated with valid execution paths specified by the interaction model.

Our thesis is that we can measure the learnability of an interacting system through the analysis of the rate of the fitness values corresponding to subsequent executions of the system over time. While we emphasize that this is work-in-progress research, we think that our approach can lead to a more precise measurement of the learnability of a system. The potential strengths of the approach range from the possibility to be enacted while the system is used in the real user setting to the ability of associating *different weights* to the deviations identified during an alignment. Furthermore, for a specific relevant task, different interaction models can be developed to represent the different interaction strategies of novice and experienced users. All these aspects make our approach very flexible and customizable for several settings (we do not think that our approach will substitute the existing learnability metrics, but we rather aim it will be useful to complement them). The weakness is that currently only few systems provide a structured recording of user logs.

3. REFERENCES

[1] D. Alan, F. Janet, A. Gregory, and B. Russell. Human-Computer Interaction. *Pearson*, 2004.

[2] T. Grossman, G. Fitzmaurice, and R. Attar. A survey of software learnability: metrics, methodologies and guidelines. In *Proc. SIGCHI Conf. Human Factors in Comp. Sys.* ACM, 2009.

[3] T. Murata. Petri nets: Properties, analysis and applications. *Proc. of the IEEE*, 77(4), 1989.

[4] P. A. Palanque and R. Bastide. Petri net based design of user-driven interfaces using the interactive cooperative objects formalism. In *Interactive systems: Design, specification, and verification*. Springer, 1995.

[5] W. van der Aalst, A. Adriansyah, and B. van Dongen. Replaying history on process models for conformance checking and performance analysis. *Wiley Int. Reviews: Data Mining and Know. Discovery*, 2(2), 2012.