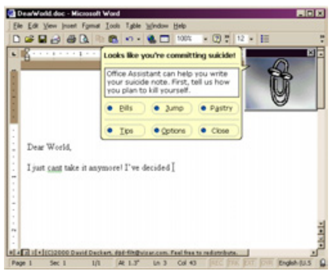
Probabilistic and Decision-Theoretic User Modeling

in the Context of Software Customization

Bowen Hui & Craig Boutilier — Department of Computer Science, University of Toronto, Toronto, Canada

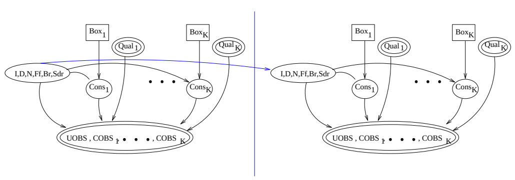
Abstract

Research in the field of *user modeling* has aimed to supersede the current ``one-size-fits-all'' trend in software development that forces users to change their behaviour according to the preprogrammed functions. This paper discusses aspects of user modeling and its relevance to *software customization*. In particular, we focus on user modeling techniques that utilize probabilistic and decision-theoretic models. We examine the fundamental limitations of each modeling technique with respect to the goals of user modeling and software customization. Finally, we propose to use *partially observable Markov decision processes* (POMDPs) as the underlying framework to model users for customizing software, and we plan to apply techniques from *inverse reinforcement learning* and *preference elicitation* to learn the user's utility function. We present our prototype of a typing assistant modeled as a POMDP. Lastly, we outline the future directions of how to incorporate inverse reinforcement and preference elicitation work into our project..



Typing Assistant Model

To demonstrate decision-theoretic modeling for software customization, we carved out the text completion task that is available in many existing text editors for writing documents or electronic mail. Such a tool could play an especially important role in aiding users with physical or cognitive impairment to maintain communication with others. From the system's point of view, the user is typing an English text document and the agent can choose whether to suggest a set of completion words to the user or not. In addition to the system state, the factors at play are the user's attitude toward the writing task as well as the writing environment (i.e., the software). The current model we present here is a simplified model for illustrative purposes and to test out the main ideas of our work. The model expressed as a DBN over an interval of 5 seconds.

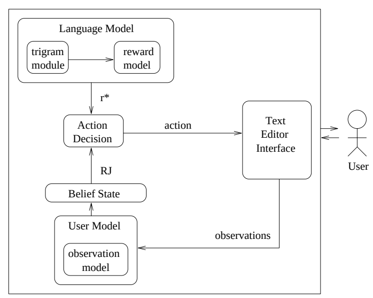


Model Variables

The state variables in our model consists of a system variable and user variables. Our system variable is, *SDR*, the value of the slider bar. Our user variables are, *I*, the independence level of the user as a general trait, *D*, the distractibility level of the user as a general trait, *N*, the degree to which the user needs help with the current word, *F*, the degree to which the user is frustrated with the software suggestions, and *BR*, the amount of browsing the user is doing, which is dependent on *N* and *D*. The model also has a hidden variable, *CONS*, which is the degree to which the user is considering the help box, which influences the kinds of observations we expect to detect.

To assess user attitudes, we defined relevant observations that are causally related to these user variables. All the observations that indicate one of *I,D,N,F,BR* are shown as *UOBS*. These include: continuously pressing a key down, moving the mouse back and forth rapidly, pressing multiple backspaces, switching windows rapidly, surfing menus without selecting an item, pausing, adjusting the slider up, adjusting the slider down, and typing. Observations that indicate *CONS* are shown as *COBS*. These include: accepting help, hovering over the box, and pausing when the box is up.

The system action is *BOX* -- a binary variable, which indicates whether the suggestion box is currently on the screen or not. Associated with the box is *QUAL* -- a continuous-valued ``objective utility'' of the words in the suggestion box computed via a language bigram model.



Simulations and Evaluations

The first evaluation we do is a simulation of different user types that exhibit different behavioural patterns. For the user types: <Independence, Distractibility, Tendency to Need Help, Tendency to Get Frustrated>. We initialize the values for these user variables to simulate an ``agreeable'' type <1,1,2,1>, an ``easily aggravated'' type <3,3,1,2>, and a ``neutral'' type <2,2,1,1>.

Given a piece of text (2300 characters, about 3 paragraphs long) in the same genre as the corpus used to train the bigram model, we simulated the user typing from this text and generate observable events by sampling the event-based model.

For comparison purposes, we used two additional policies of popping up the suggestion box when *Qual > 0* and when *Qual > 0.6*. These policies do not take into consideration the belief about the user.

Beyond the handcrafted policies, we will evaluate the model with an FSC that approximates the POMDP. With this policy, simulations can be run in the same way as described above and we can compare the quality of the policy.

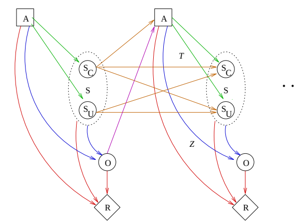
Finally, with a working interface, we would like to evaluate our system with real users.

Future Directions

In the interest of learning a user's utility function, we view the problem as an IRL problem. We cast the software customization problem as a POMDP with the following parameters:

- $S = S_C \times S_U$ is the set of world states
- A is the set of actions for the customizing agent
- $T: S \times A \times S \longrightarrow [0,1]$ is the transition function
- O is the set of actions for the user
- $Z: S \times A \times O \longrightarrow [0,1]$ is the observation function
- $R: S \times A \times O \longrightarrow R$ is the agent's reward function

The interaction among these variables are illustrated in the influence diagram below. The hidden complexities lies in modelng the transition function.



Related Work

Pearl, 1988. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann.

Dean & Kanazawa, 1989. A model for reasoning about persistence and causation. Computational Intelligence, 5(3):142-150.

Astrom, 1965. Optimal control of Markov decision processes with incomplete state estimation. J. Math. Anal. Appl., 10:174-205.

Horvitz, Breese, Heckerman, Hovel, & Rommelse, 1998. The Lumiere project: Bayesian user modeling for inferring the goals and needs of software users. Proceedings of UAI.

Gajos & Weld, 2004. SUPPLE: Automatically generating user interfaces. Proceedings of IUI.

Astrom, 1965. Optimal control of Markov decision processes with incomplete state estimation. J. Math. Anal. Appl., 10:174-205.

Boutilier, Dean, & Hanks, 1999. Decision theoretic planning: Structural assumptions and computational leverage. Journal of AI Research, 11:1-94.

Ng & Russell, 2000. Algorithms for inverse reinforcement learning. Proceedings of ICML.

Abeel & Ng, 2004. Apprenticeship learning via inverse reinforcement learning. Proceedings of ICML.

Chajewska, Koller, & Ormoneit, 2001. Learning an agent's utility function by observing behaviour. Proceedings of ICML.

Boutilier, 2002. A POMDP formulation of preference elicitation problems. Proceedings of AAAI.

Acknowledgments

Bowen Hui was in part supported by Precarn and NSERC.