Plan Recognition to Aid the Visually Impaired

Marcus J. Huber* Richard Simpson†

*Intelligent Reasoning Systems
4976 Lassen Drive
Oceanside, California, 92506
marcush@marcush.net

†Dept of Rehabilitation Science and Technology
University of Pittsburgh
Forbes Tower, Suite 5044
Pittsburgh, PA 15260
ris20@pitt.edu

Abstract. Less than half of the individuals of working age with visual impairments are employed and a significant barrier to employment is effective computer access. Screen reader applications offer some help but have limited context sensitivity and are of limited use in applications with dynamic "interfaces" like web pages. Sophisticated screen readers provide aid through application-specific scripts but their full potential is reduced by limited awareness of the scripts and the difficulty in programming and modifying scripts. Technologies such as plan recognition and automated script generation and optimization that provide a more adaptive interface for the user will significantly improve computer accessibility to the visually impaired. In this paper, we discuss the addition of probabilistic plan recognition capabilities and supporting framework to a leading screen reader in order to improve accessibility of computers to the visually impaired at work and at home.

1 Introduction

Effective access to computers is becoming increasingly crucial for academic and vocational success. It is predicted that 60 percent of U.S. jobs will require computer skills within the next five years. Currently, less than half of the individuals of working age with visual impairments are employed, and a significant barrier to employment is effective computer access. In particular, manipulating information on the WWW, our application focus, is rapidly becoming a crucial computer skill. Screen readers, applications that audibilize the text on computer screens, provide some support. However, to find relevant information on web pages, this can involve dozens if not hundreds of manual navigation actions and is very difficult to repeat. Scripts can automate such navigation tasks, but screen readers are still limited in a number of ways, including not being able to adapt their behavior to the semantic contents of specific web pages and the difficulty users have in using the scripting capabilities of screen reader.

The software developed during this project[1] has been integrated with a sophisticated screen reader with scripting capability called JAWS [2]. JAWS has been extended to do several things that other current screen readers cannot: recognize scripts to identify when the user is manually performing a task (possibly with mistakes) for which a script is available and prompt the user to make use of the script instead (saving tremendous

[1] Funded by the National Science Foundation under grant DMI-0091590.
[2] Freedom Scientific, 11800 31st Court North, St. Petersburg, Florida 33716.

time); know something about the goal of a set of actions and produce a script that is much shorter and more efficient than that of a simple macro recorder (simplifying and speeding script generation); and use existing scripts as the basis for recognizing scripts that can be used with minimal tailoring (also simplifying and speeding script generation). What this means to the end user is that they will be made aware of the existance of the scripts that they are performing manually, they can make errors during script generation and web page navigation, scripts automatically generated will be compact and optimized, and tailoring existing scripts will be significantly easier.

2 Enhancing A Screen Reader

One capability that provides a significant improvement in screen reader technologies is the ability to identify the user's intentions as the user is performing a task. This capability, called *plan recognition*, provides assistance to the user as well as to provide useful contextual information to our script generation and optimization extensions discussed below. Our plan recognition mechanism is based on the ASPRN (Automated Synthesis of Plan Recognition Networks) system's probabilistic modeling theories and implementation[3, 4]. ASPRN takes procedures as its input, in this case scripts, and outputs specially constructed probabilistic models that we call Plan Recognition Networks (PRNs), a particular instantiation of belief networks, that models those procedures. Because PRN computations are based on probabilistic models and relationships, belief networks are well suited for dealing with uncertain, conflicting, or extraneous information, something often encountered in user interfaces. Plan recognition algorithms using probabilistic representations (e.g., [3, 5]) are well suited for application to screen readers. Users will almost certainly deviate from standard task templates intentionally or unintentionally, something non-probabilistic representation schemes (e.g., [1, 2]) cannot deal with in natural or pragmatic way.

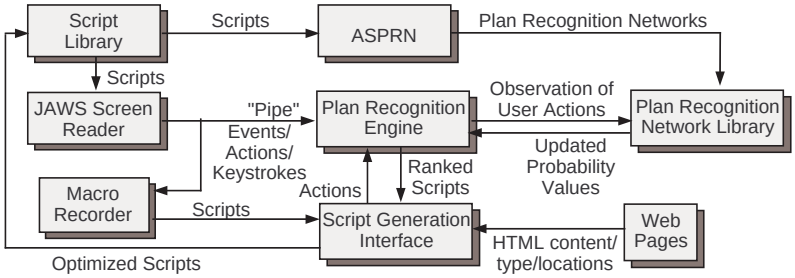

Fig. 1. Architectural diagram of our enhanced screen reader system.

Our system's design is shown in Figure 1. All of the components illustrated in Figure 1 have been implemented and are operational. The JAWS screen reader forms the basis for all of our work, providing much of the original functionality and internal representations for providing computer access to the visually impaired. The Script Library maintains all of JAWS' application-specific scripts, each of which accomplishes a specific, small task. Based on feedback from visually-impaired clients, we identified sev-

eral tasks that the visually impaired frequently perform with web browsers (e.g., retrieving a weather forecast). We developed scripts for performing each of these tasks and also a number of subscripts that perform more elemental functions. These subscripts will ultimately play an important role in distinguishing between scripts when multiple strategies can be used interchangeably within a single high-level task and when multiple high-level tasks are differentiated based on which strategies can be used. Scripts within the Script Library are converted by ASPRN into PRNs and placed in the PRN Library. The Plan Recognition Engine (PRE) accesses the PRNs and uses these probabilistic models in combination with information about the user actions to determine the scripts most likely associated with the user's actions. If the user is doing something that an existing script can perform, the PRE suggests this to the user, thereby improving productivity. The Script Generation Interface (SGI) takes a recorded sequence of user actions from the Macro Recorder and, by using plan recognition and hand-coded algorithms, create an optimized script which is then placed in the Script Library for later use.

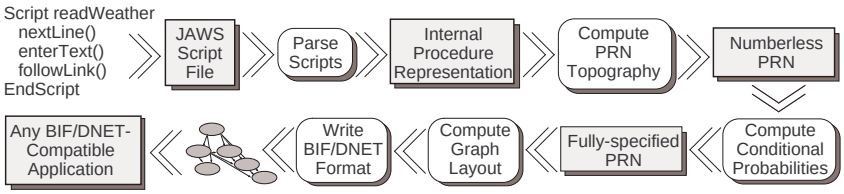

Fig. 2. The ASPRN process of taking a JAWS script and constructing a specially-designed belief network called a Plan Recognition Network (PRN).

Figure 2 shows how ASPRN parses a script into a generic (reusable) internal procedural model, computes a PRN topography based on this, computes the conditional probability values associated with the PRN instance, determines a default visual layout of the PRN for belief network visualization applications, and finally writes out the PRN for adding to the PRN Library and subsequent use by the PRE. The basic modeling theories, algorithms, and key application concepts from the original ASPRN system [3, 4] were modified to suit the specifics of the JAWS scripting language and the task of providing assistance to the visually impaired. Some redesign was necessary to account for the simpler scripting language that JAWS uses compared to the agent-based plan languages that ASPRN has thus far been applied.

Figure 3 shows how script constructs are transformed into belief networks. These belief network transformations are composable and are therefore linked together according to the script being modeled, creating a single belief network with dozens to hundreds of probabilistic nodes (depending upon the length of the script). While we cannot go into detail in this short of a paper, ASPRN supports belief network-based probabilistic modeling of the key aspects of JAWS' script constructs, including sequences of script commands, function and sub-script invocation, conditional execution, and iteration. ASPRN models a script's actions and a script's temporal, conditional, and hierarchical relationships. Each modeled low-level action also has an associated evidence node that is used to model the reliability of observations that has/has not

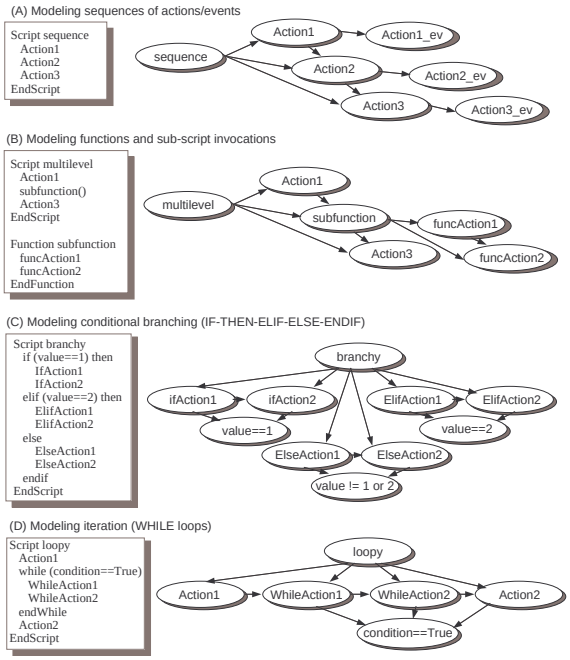

Fig. 3. PRN sub-constructs that model various JAWS script sub-constructs.

been performed. The conditional probabilities within the PRNs are specified initially by case-based templates. Due to the regularity and composability with which all script components are modeled, all possible cases have been represented within ASPRN such that a completely-specified belief network is constructed. The conditional probability cases and values are based on the type of node being specified and the types of the nodes linking to it. Probability values for these generic templates were determined empirically in [3, 4] to result in the natural, convex progression of posterior probabilities given correct observation sequences. Actual, in-use, statistical probabilities which will eventually be learned in future work.

The operational system works in two basic modes, one as the user is interacting normally with the system in pursuit of their task objectives, and one while the user is creating or modifying scripts. While the user is navigating between and within web pages, JAWS observes the user's actions and screen events. After each observed action, the PRE adds this information as evidence into the PRNs, performs Bayesian inferencing, and then sorts the scripts based on the subsequent posterior probabilities. Figure 4 illustrates how PRNs react to evidence of the user's actions. PRNs modeling scripts that do not quite match the evidence pattern do not react as strongly and result in lower posterior probabilities, allowing the PRE to discriminate between alternatives. Once a script has a sufficiently high likelihood of being pursued (user studies will determine an appropriate value for this), the system informs the user that a script exists that can perform their task much more efficiently, or is close but needs to be tailored slightly for the user's current task. Bayesian learning algorithms that will be added at a later stage

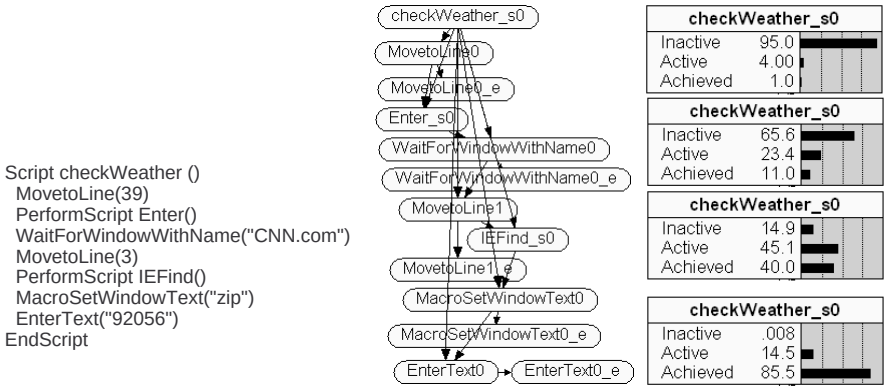

Fig. 4. Left, a segment of an actual script for reading weather at CNN.com. Center, the PRN constructed by ASPRN. Right, the probability distribution for the script being performed by the user with zero observations (i.e., priors) at top, MoveToLine observed (second), Enter observed (third), and all actions observed (bottom). Note the gradual progression from Inactive to Achieved (i.e., completed).

will take the user's acceptance or rejection of this suggestion and adjust the internal PRN model appropriately to gradually adjust the PRNs to the user's preferences.

During generation of scripts, the user records a raw sequence of user actions using the Macro Recorder. The SGI then analyzes the sequence of actions and events and uses plan recognition and other algorithms to determine sub-sequences of the user's actions (which possibly contains mistakes) that perform the equivalent operations. If any strong matches are found then the sub-sequence is recommended as a replacement to the user (this is another possible Bayesian learning application area).

Other future work includes exploring variations and specializations of PRN topographies and probabilities, including aggressiveness and temporal accuracy. User trials are scheduled to evaluate all aspects of the system's efficacy. Our continuing work will also focus on the vocational and educational uses of the proposed software involving off-line applications.

References

1. C. Broverman, K. Huff, and V. Lesser. The Role of Plan Recognition in Design of an Intelligent User Interface. In *Proceedings of the IEEE Systems, Man, and Cybernetics Conference*, pages 863–868, 1987.
2. B. Goodman and D. Litman. Plan Recognition for Intelligent Interfaces. In *Proceedings of the Sixth Conference on Artificial Intelligence Applications*, pages 297–303, 1990.
3. M. Huber. *Plan-Based Plan Recognition Models for the Effective Coordination of Agents Through Observation*. PhD thesis, The University of Michigan, 1996.
4. M. Huber, E. Durfee, and M. Wellman. The Automated Mapping of Plans for Plan Recognition. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 344–351, July 1994.
5. G. Kaminka, D. Pynadath, and M. Tambe. Monitoring teams by overhearing: A multiagent plan-recognition approach. *Journal of AI Research*, 2002.