A Script-Based Approach to Modifying Knowledge-Based Systems

Marcelo Tallis
Department of Computer Science and Information Sciences Institute
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
tallis@isi.edu

Modifying knowledge-based systems (KBSs) is a complex activity that needs to be performed very often. The occurrence of changes in a KBSs environment, requests for extending the system's functionality, and the debugging of inadequate knowledge are some of the events that demand modifications to a KBSs. One of the difficulties of KBS modifications is that they might require the modification of several related portions of the system. Determining what portions have to be changed and how to change them requires a deep understanding of how the elements of the KBS interact. This requirement is especially hard for users when the rationale behind the design of a KBS or the details of its implementation are unknown.

Although current Knowledge Acquisition (KA) tools can lessen the burden of modifying KBSs, they do not directly address the issue of how to modify related portions of a KBS in coordination. Consequently, their support is limited. Some of these KA tools, like SALT (Marcus and McDermott 1989), support KBS modifications based on strong assumptions about the underlying problem-solving strategy used by the KBS. These tools can only support modifications to the domain-dependent knowledge manipulated by the system, which is a strong limitation of these tools. Other tools, like EXPECT (Gil 1994), support KBS modifications based on the understanding of how KBS components interact. These tools can support modifications to any aspect of the KBS. Unfortunately, the understanding of these interactions alone is insufficient to determine how related components have to be modified.

In this research, I aim to develop an approach for building KA tools that both, provide strong guidance to users and allow a wide range of modifications to a KBS. My proposed approach consists of providing a knowledge acquisition tools with *knowledge acquisition scripts* (or *KA scripts*) that represent prototypical procedures for modifying knowledge-based systems (KBSs). More specifically, a KA script describes a prototypical sequence of changes for following up the consequences of previous changes to the KBS. An example of a KA script is:

if a) a change to a KBS makes a goal more general and
b) the modified goal cannot be achieved
then generalize method that achieved the original goal

KA scripts enable a KA tool to relate changes concerning different parts of the KBS.

We have implemented a Script-based KA tool called ETM that supports modifications of EXPECT KBSs. The role of ETM is to help a user to bring a KBS back to a coherent state after a user has performed some initial changes to the KBS using EXPECT conventional KA tool (the KBS is assumed to be coherent before starting its modification). ETM engages in a loop in which 1) it identifies unattended consequences of previous changes, 2) suggests KA Scripts that take care of those consequences, and 3) guides the user throughout the application of one of these KA Scripts (chosen by the user). A complete modification usually requires of the execution of several KA scripts. This is because some changes can have several consequences that have to be followed up by different KA scripts, and because the execution of some KA scripts have consequences that also need to be followed up.

In developing ETM we addressed several research issues that concerned the development of a KA script library, the coordination of the execution of KA scripts, and the model of interaction with the user. We have designed a KA script library that contains around 75 KA scripts (only a dozen were implemented).

We carried out a preliminary evaluation of ETM that compared the performance in modifying KBSs for subjects using ETM vs. subjects using EXPECT. The experiment showed that subjects using ETM outperformed the ones using EXPECT, especially in the more complex modification tasks. These results suggests that script-based knowledge-acquisition is a promising technology. In this first experiment we chose subjects that were already familiar with EXPECT but not with ETM. We expect that the difference in performance will be more significant in our future experiments involving subjects not familiar with EXPECT either.

References

Gil, Y. 1994. Knowledge Refinement in a Reflective Architecture. In AAAI-94, Seattle, WA, 1994.

Marcus, S. and McDermott, J. 1989. SALT: A Knowledge Acquisition Language for Propose-and-Revise Systems. *Artificial Intelligence*, 39(1):1-37.