International Journal of Artificial Intelligence in Education 19 (2009) 103-104
IOS Press

Preface

Numerous empirical studies have demonstrated the effectiveness of Intelligent Tutoring Systems (ITSs). Students using state-of-the-art systems often learn significantly more in significantly less time when compared to alternative learning scenarios like classroom teaching or on-the-job training. Yet, too few students experience these benefits because ITSs are difficult to author. In order to be able to support students' learning, ITSs need several kinds of knowledge. First, ITSs need domain models, that is, domain knowledge represented in a way that allows the system to diagnose students' solutions and generate individualized instruction. Second, ITSs need to model students in order to be able to tailor pedagogical decisions towards each student. Third, ITSs need pedagogical knowledge in order to be able to adapt the teaching strategy. Finally, many ITSs either require sophisticated graphical user interfaces (e.g., to simulate the operation of a submarine) or use clever interface design to reduce the knowledge burden or enhance pedagogical effectiveness.

Due to the wide-ranging requirements for knowledge, the development of ITSs is a process with extremely high resource demands. The developer also needs a lot of knowledge and experience outside of the instructional domain – knowledge of programming, Artificial Intelligence, Cognitive Science, Psychology and Education is needed to develop effective ITSs. Researchers have reported 100-1000 hours of authoring time needed for one hour of instruction. It is therefore natural that authoring systems have been a topic of interest since the early days of ITSs in 1970s. Authoring systems are designed to reduce the time necessary for ITS development, and may also make it possible for non-programmers and even teachers to develop ITSs for their courses.

Tom Murray, Stephen Blessing and Shaaron Ainsworth edited an excellent volume in 2003 describing the research done on authoring tools for all kinds of learning environments (*Authoring Tools for Advanced Technology Learning Environments*, published by Kluwer Academic Publishers). As Murray pointed out in his overview chapter included in that volume (pp. 491-544), although a lot of progress had been made, it was still not clear which approaches were most promising, as the majority of authoring tools had been used in research labs by a small number of authors. Further, not enough had been done to evaluate claims of authoring tool developers regarding reduction in authoring time, effort, or skill requirements.

Since that remarkable collection of papers, however, there have been no similar collections or dedicated special issues of journals devoted to ITS authoring systems, although a lot of research has been reported in conference papers. We therefore thought it was high time to invite reports on the new developments in authoring systems and related research.

This special issue of IJAIED includes four papers presenting different approaches to authoring ITSs. Aleven, McLaren, Sewall and Koedinger discuss example-tracing tutors, a simple approach in which the author creates the tutor by designing the interface and demonstrating alternative solutions. The advantage of example-tracing tutors is that they are easier to develop than fully fledged model-tracing tutors. This paper presents estimates and an experimental evaluation of savings in authoring time.

Mitrovic and colleagues present ASPIRE, an authoring and deployment system for constraint-based ITSs. ASPIRE supports the authoring process by inducing constraints from the domain ontology developed by the author, as well as from examples of problems and their solutions. In addition to

presenting the features of ASPIRE and the support provided for authoring, the paper presents one ITS developed in ASPIRE and discusses its effectiveness.

Blessing, Gilbert, Ourada and Ritter present the Cognitive Model SDK authoring system that supports development of model-tracing tutors, focusing on the development of the cognitive model underlying the tutor. The author still develops the cognitive model him/herself, but the amount of programming required is reduced. As an alternative to an author-developed interface, Cognitive Model SDK also supports communication with third-party interfaces.

Hayashi, Bourdeau and Mizoguchi present their OMNIBUS ontology, which covers multiple instructional and learning theories and paradigms. Their paper also present SMARTIES, an authoring system that supports teachers in creating learning/instructional scenarios based on OMNIBUS.

The presented authoring systems ease the development process by automating a variety of authoring tasks, such as interface design (CTAT and ASPIRE), knowledge acquisition via machine learning (ASPIRE), supporting the author in developing the cognitive model (SDK) or their teaching strategy (SMARTIES) and supporting the development of ITSs with no or minimal programming required (ASPIRE and CTAT). All of these issues are of great importance for intelligent tutoring systems, and can help bring the area a step closer to achieving widespread use of ITSs.

GUEST EDITORS

Antonija Mitrovic
Ken Koedinger