Order Number 9010819

Learning Companion Systems

Chan, Tak-Wai, Ph.D.

University of Illinois at Urbana-Champaign, 1989

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

LEARNING COMPANION SYSTEMS

BY

TAK-WAI CHAN

B.Sc., University of Nottingham, 1981
M.S., University of Illinois, 1986

THESIS

Submitted in partial fullfillment of the Requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1989

Urbana, Illinois

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

THE GRADUATE COLLEGE

September 1989

WE HEREBY RECOMMEND THAT THE THESIS BY

Tak-Wai Chan

ENTITLED Learning Companion Systems

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF Doctor of Philosophy

Arthur B. Baskin, III
Director of Thesis Research

William J. Kubitz
Head of Department

Committee on Final Examination†

Arthur B. Baskin, III
Chairperson

J Richard Dennis

H George Friedman Jr.

Glenna Belford

† Required for doctor's degree but not for master's.

O-517

LEARNING COMPANION SYSTEMS

Tak-Wai Chan, Ph.D.
Department of Computer Science
University of Illinois at Urbana-Champaign, 1989

This thesis describes a new class of Intelligent Tutoring Systems (ITS) which I call the Learning Companion Systems (LCS). In the learning environment of such a system, there are three agents involved, namely, the human student, the computer learning companion, and the computer teacher. As implied by its name, the role of the computer learning companion is to act as a learning companion for the student. To this end, the companion performs the learning task at about the same level as the student, and both the student and the companion exchange ideas while being presented the same material by the teacher. In designing the prototype of LCS, Integration-Kid, in the domain of integration, I define some protocols of learning activities among the three agents. These protocols reflect the different learning stages of the learners in the learning process as well as appropriately restrict their possible unbounded interactions. Curriculum Tree, an architecture based on the domain structure is designed as the implementation framework of Integration-Kid. Production systems are used to simulate different agents' interactions via a common blackboard. The basis of the problem solver for both the companion and the teacher is a term rewriting system. Finally, the disadvantages of my design of the prototype and LCS in general, and the future directions of LCS related research are discussed.

To my Grandmother

陳襯色 女士

ACKNOWLEDGEMENTS

I would like to thank my parents for their patience and selflessness. I would also like to thank my advisor, Dr. Arthur Baskin. His help, advice, and generosity with his time are greatly appreciated. Special thanks is due to Dr. Richard Dennis for testing the system and suggesting corrections. Thanks is also due to Howard Aizenstein for his encouragement and hours of stimulating discussions. I would also like to thank my office-mates for their help. Finally, it is my particular pleasure to thank my wife Lisa. Without her understanding and help throughout the process, this work would take a lot longer. Her tremendous love and affection has made the time all worth it.

TABLE OF CONTENTS

CHAPTER

CHAPTER 1.

INTRODUCTION

1.1 Why There was One Who Studied with the Prince

In the past in China, there existed a child who studied with the Prince under the instruction of a royal teacher, as mentioned by the Chinese proverb "Studying with the Prince." Why? Perhaps the Queen recognized the importance of *learning companionship*. For whatever reason it evolved, the Prince was clearly expected to learn more effectively with the companion than by learning alone.

Examples of peer learning are numerous. Children acquire language in day care much more quickly than at home. Christians understand the Bible better through Bible study groups. Teachers find students keen on comparing answers. Study groups for law students have long been an institution in law school. Also, a beginning medical student shocked by the tremendously heavy school work believes that he can survive since other students can.

1.2 Traditional ITS -- a Two Agent Model

Private tutoring has proven to be as much as four times as educationally effective as a normal classroom setting and 98 percent of the students perform better with private tutors (Bloom, 1984). Intelligent tutoring systems, modeled after the idea of a private tutor, provide individualized instruction which offers the potential of better attention to individual student needs. This original vision remains strong today (Clancey, 1986). Realized as an interactive computer program, most ITS, naturally, can be conceived as a two agent model -- a learning environment with a *human student* and a *computer simulated teacher* (Figure 1.2).

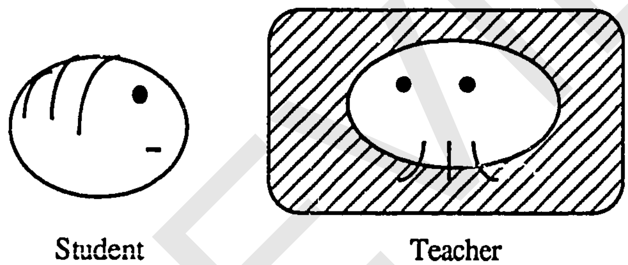

Figure 1.2 Traditional ITS Model where Computer is Simulated after a Human Tutor

In order to better understand the "knowledge dynamics" of various learning environments, we may look into the changing knowledge involved in each agent in the different environments. To do this, it seems helpful to view the current knowledge of a person in a given domain as his own interpretation of the domain. This interpretation evolves from his previous interpretation as a result of learning. Such a change of interpretation involves his background knowledge, historic culture and the current learning environment. To emphasize such individual dependencies, we call a person's knowledge of the domain his own *interpreted knowledge*. But, such interpreted knowledge of the domain varies over time, so, at a given point of time, we say that he has acquired a *version* of interpreted knowledge in the domain. Note that the interpreted knowledge may well consist of both correct and incorrect concepts. A version of interpreted knowledge is a *higher level* one than another if it articulates more knowledge, fewer misconceptions, and more flexible representation.

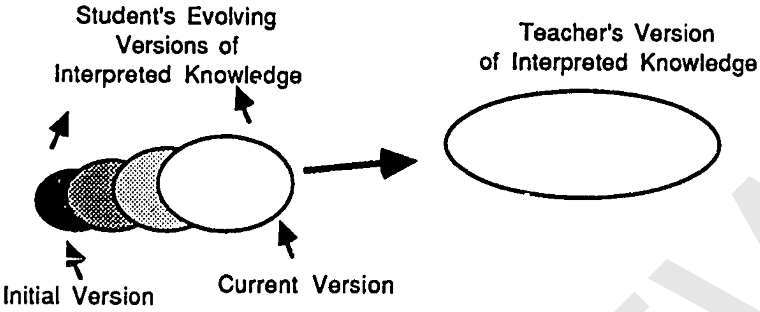

Figure 1.3 Cognitive Change in Traditional ITS Environment

In the learning environment of a traditional ITS, the goal of the teacher is to try to alter the student's evolving interpreted knowledge so that it *converges* (Figure 1.3) to his own (which is a much higher level but rather static one). An important function that the teacher exercises is monitoring of the learning activities and providing support for the student so that the convergence goes effectively.

1.3 Learning Companion System (LCS) -- a Three Agent Model

Learning companion system (LCS), the model proposed in this thesis, is a three agent model. In this model, the computer simulates not one, but two coexisting independent agents -- a tutor and a learning companion. The additional agent added to the traditional ITS model forms a small society with a richer social context. The simulated learning companion may act as a competitor or as a collaborator to the student. While being challenged by the companion, the student also observes sub-optimal performance of the companion. Misconceptions in learning are common, natural and need not be limited to the human student.

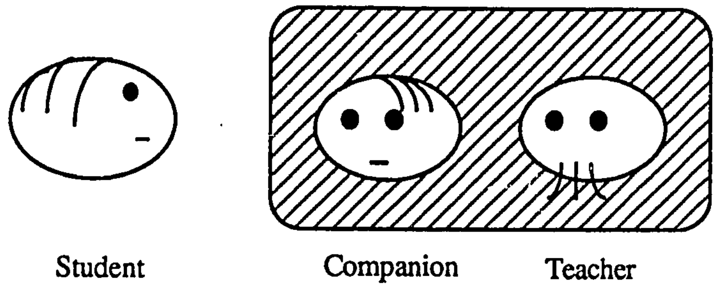

Figure 1.4 Learning Companion System Model where the Computer is Simulated as Two Agents, a Teacher and a Learning Companion

In the situation of learning companion system environment, while one student is looking at multiple perspectives, planning, evaluating new ideas, monitoring and assessing solutions, he is forced to unfold, examine, and defend his ideas when challenged by the other student, and in turn to keep an eye out for possible mistakes made by the other which may result in defeating a proposed idea. Such a process of mutual justification may not easily happen in a student-teacher situation because of different social status and knowledge levels, and thus different expectations. Thus, learning, in an LCS environment, is the *merging* (Figure 1.5) of two evolving versions of interpreted knowledge into the teacher's one.

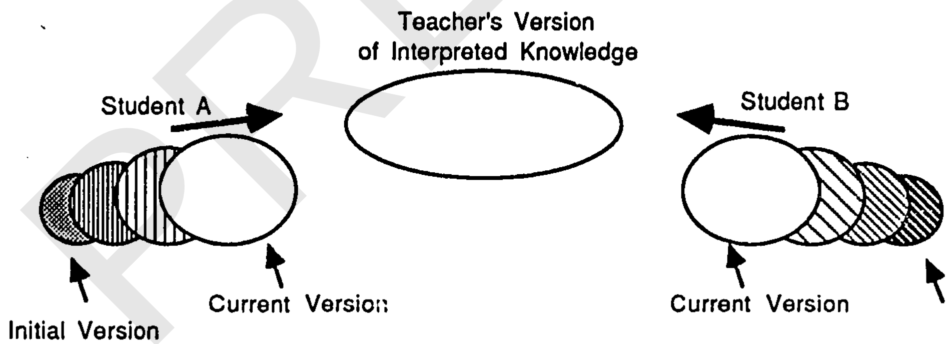

Figure 1.5 Cognitive Change in Learning Companion System Model

From this view, *teaching* may be regarded as the effort of altering the other's version of interpreted knowledge towards his own while *learning* is working the other way round. In this regard, it is quite easy to see that in the student-teacher environment, one is teaching and the other is learning. But, in student-student interaction, both agents are teaching and learning. Notice that, the level of a student's knowledge may never reach the level of the teacher's knowledge. For example, giving a problem of integral calculus to a student and to a professor of real analysis, even though they both solve the problem, we would never say that they have the same understanding of integral calculus. However, the presence of a teacher is necessary because of the imposing control on the student-student interaction; otherwise, their interpreted knowledge may not spontaneously converge to the knowledge defined by the learning goals. For young students, this is particularly necessary for they are not self conscious enough of their own learning.

1.4 Some Conventions

For minimization of potential ambiguity and convenience, we adopt some conventions for some words in this thesis. The conventions hold unless it is clear from the context that the word refers to its general meaning.

Companion or *learning companion* -- refers to the computer simulated learning companion.

Student or *learner* -- refers to the human student, the target user of the system.

Students or *learners*-- refers to both the student and the computer simulated companion.

Teacher and *tutor* -- are synonymous.

Also, the arbitrary constant c in indefinite integration, for example, $\int e^x \, dx = e^x + c$, is omitted for simplicity.

1.5 Application of the LCS idea to Integral Calculus

Instead of picking a well tailored domain, we choose a formal subject to test the idea of LCS so that the generality of its application may also be proved. Indefinite integration, at the level of a first year undergraduate, is the subject domain we chose to explore the design, construction, and operation of learning companion systems. For example, $\int x \log x \, dx$, $\int e^x \sin x \, dx$. Calculus is a domain that requires heuristic solution and a good deal of resourcefulness and intelligence. Many problems can be approached in a variety of ways — some of which are better than others. Also, this domain is not heavily dependent on other mathematical abilities. The prerequisites are some competence with differentiation and algebraic manipulation. Yet, students consistently have more difficulty in taking examinations and in doing miscellaneous exercises than they should despite many hours of working problems, as noted by Schoenfeld (1978). Part of the difficulty lies in their lack of adequate integration of separate concepts and techniques learned and therefore better judgement of the form of integrands. Being rather isolated and not easy to master, this domain will be a good vehicle for our attempt to examine how a student's learning can be affected through the interaction with a computer companion.

Since our primary goal is to explore the idea of LCS, we confine ourselves to a subset of indefinite integration. However, a complete small course is covered so that the implications of peer interaction can be reflected thoroughly in different stages of learning of the subject.

1.6 Knowledge of Learning Companion

When the student sits in front of an LCS, we may tell the student that the learning companion is *simulated* to be his classmate, or, that the learning companion is actually *learning* together with him. These different sayings correspond to two different approaches of implementing the knowledge of the companion -- simulating skill acquisition and actual machine learning. In the first approach, the performance capability of the companion is controlled by the system while in the second approach, the companion is required to be able to learn as the student does but by using the techniques of machine learning. In both approaches, the image presented to the student must be that of a companion whose skill advances in roughly the same way as that of the student. While the machine learning approach may arouse the student's curiosity about how the companion learns and could possibly be one of the most interesting application areas of machine learning, the sophistication of the learning companion seems beyond the scope of current learning systems. In the simulation approach, the growing domain knowledge of the companion can be viewed as selective use of the teacher's knowledge together with possible misconceptions of a typical student. In this thesis, we use the simulation approach.

1.7 Focus and Organization of the Thesis

The hypothesis of this thesis is: in the three agent model of learning companion system, more dimensions of learning may be achieved, higher motivation may be maintained, and better attitude of learning may be fostered, and thus, the three agent model is more educationally effective than the two agent model of traditional ITS. The research primarily focuses on exploring an *alternative architecture* to enable realization of peer interaction and its implications into a computer-based learning environment. Integration-Kid, an LCS prototype, has been completed using Common Lisp on a TI Explorer.

In this thesis, we briefly deal with the human social and cognitive issues that relate to learning companion systems. Based on such issues, we design several protocols of learning activities. A prototype of LCS for the domain of integration is then designed and implemented. Chapter 2 reviews Intelligent Tutoring Systems (ITS), related work in peer learning, collaborative partners and apprenticeship learning. Chapter 3 discusses the major design issues which must be solved in building an LCS. Chapter 4 presents protocols of learning activities which form the basis for the detailed design and implementation of an LCS system for integration. Chapter 5 describes the overall design, in particular, the curriculum tree architecture, and the implementation of the prototype, Integration-Kid. Chapter 6 are example uses of the system. Chapter 7 is a discussion of advantages and limitations of our design. Chapter 8 presents some future directions of LCS research and the final conclusions of this work.

CHAPTER 2.

BACKGROUND

This chapter provides the readers with the background knowledge of LCS. An overview of ITS with a general description of its components and discussion of some existing ITS is given. Following that, we point out some of the problems and difficulties in ITS research. For the cognitive part, we offer some psychological perspectives of peer learning. Finally, we relate our work with the idea of collaborative partner and apprenticeship learning.

2.1 Overview of ITS

Given that ITS and LCS share the same goal and there are various approaches of developing ITS, LCS should be viewed in the context of ITS research. This section is a short overview of the ITS research. The best and most thorough review of the history of ITS is provided in Sleeman and Brown (1982). Brief summaries of the early work are also available in Barr and Feigenbaum (1982), and a more recent and thorough description is given by Wenger (1987).

2.1.1 Components of an ITS

A natural division of the task of ITS can fall into four distinct components (Figure 2.1). Seldom has a system paid an equal amount of attention to all the components. The boundaries between the components are not always sharp. Also, various tasks do not necessarily correspond to a distinct component in an actual system. For instance, evaluations of the student's answer can be shared between the domain knowledge component and the teaching expert component. Furthermore, such a division is not even a guideline for ITS implementation. Nevertheless, the distinction serves as a useful level of abstraction for understanding the functionality of ITS and LCS.

Domain Knowledge Component

One of the most important things for a teacher to possess is the knowledge of the subject being taught. This component has two basic functions. First, it acts as the source for the knowledge to be presented. This includes generating explanations and responses to the student as well as tasks and questions. Second, it serves as a standard for evaluating the student's performance. For this function, it must be able to generate solutions to problems in the same context as the student does, so that their respective answers can be compared. Furthermore, if the tutoring system is to guide the student in solving problems, the component must also generate sensible solution paths, so that intermediate steps can be compared.

In many domains, it is important to represent situational specific knowledge (e.g. draw analogy to a previous problem), since neither the student nor the teaching system can have complete knowledge of the subject domain. Insufficiency and implicit contradictions may always be lurking, only to be revealed upon the acquisition of further knowledge.

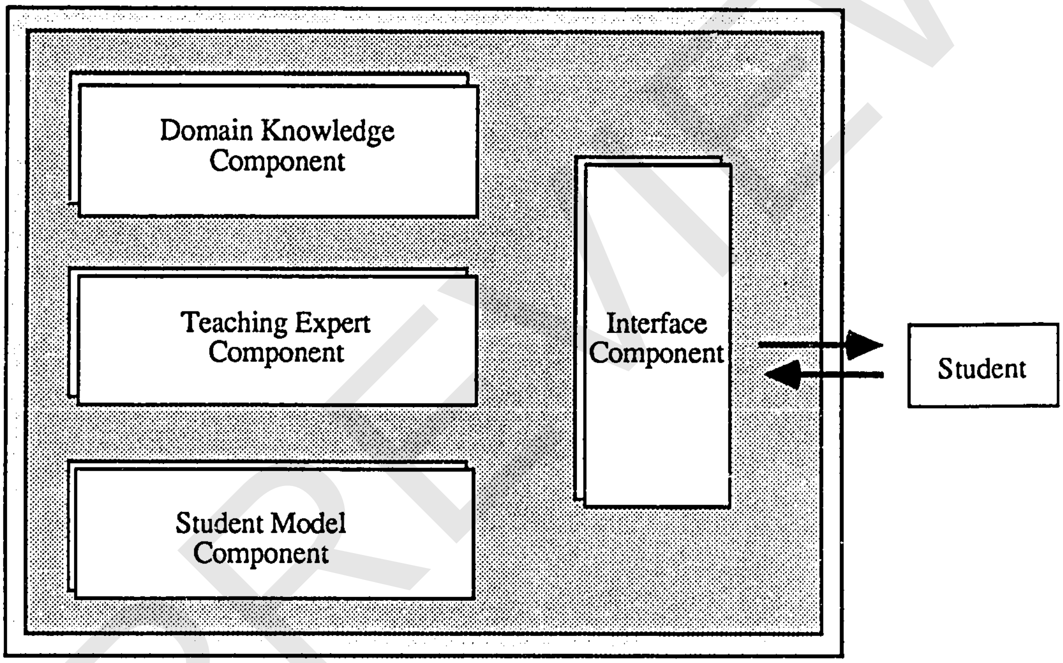

Figure 2.1 Intelligent Tutoring System Configuration

Teaching Expert Component

Once an ITS has represented the subject domain knowledge, some way must be found of actually imparting this knowledge to the student. This includes what to say and how to say it given a situation, as well as generation of a sequence of problems. Surprisingly, there has not been much emphasis on how to provide ITS with teaching expertise (McCalla & Greer, 1987). An exception is the LISP Tutor (Anderson et al., 1984a) which encodes a number of tutoring principles as a set of production rules. These principles are derived from Anderson's theory of cognition, ACT*[1] (Anderson, 1983).

Student Model Component

A student model represents the student's understanding of the material to be taught. The purpose of this component is to make hypotheses about his misconceptions and suboptimal performance strategies, so that the teaching expert component can point them out, indicate why they are wrong, and suggest corrections. Representation of the domain expertise is often insufficient to model student's errors. The task of constructing such a student model is obviously not simple for computer-based systems. A solution is to construct the student model from the primitive elements of the domain that spans both correct and incorrect knowledge. In practice, it is difficult to account for all observed errors with a consistent set of primitive elements that is computationally tractable (Sleeman, 1983).

Interface Component

The interface component processes the flow of communication in and out. In both directions, it translates between the system's internal representation and an interface language understandable to the student in its finalized form. People combine data from a wide variety of sources, like voice effects or facial expressions. But the computer can neither see nor hear. Variation or noise of the student's responses that may come from changes over emotion, mood, instinct, physical condition or external interruption could be totally out of expectation of the computer. The limited communication channels — a keyboard, a mouse and a screen — permitted by the system are inevitably handicapped compared to human communication.

Early ITS tried to use natural language as their communication language. With widespread availability of bit-mapped displays, most current ITS have essentially avoided the problem of natural language understanding. The use of fixed menus with multiple-choice answers, direct manipulation of visuals or hypertext written materials, shifts communication from words to pictures.

2.1.2 Motivations for Development of ITS

We all know that education is highly labor intensive. In school, teachers are heavily burdened by their role of transferring knowledge to students — so heavily that they rarely find

[1]ACT* (ACT- star) is a recent version of ACT (Adaptive Control of Thought) which is a general theory of cognition with a particular emphasis on skill acquisition.

enough time to pay special attention to student's individual needs. School learning is an important part of human life in a modern society, yet remains an inefficient process, nor has it fully benefited by modern technology. With the rapidly dropping price of computers and the increasing power of hardware, a piece of good software, even if it just takes up a small part of the teaching role of teachers, can save many years of human work in the future (Westcourt et al., 1977; Papert, 1980). Of all the computer applications that are possible today, it seems to Schank (Schank, 1984), that education is the most important and potentially beneficial.

Learning behavior is difficult to understand. Furthermore, it evolves over time. The ideas of Artificial Intelligence (AI) and its techniques that have been developed have had a pervasive impact on many application domains today. It is the potential capability of such techniques for exploration of the process of learning and teaching that makes ITS research one of the most interesting and socially significant applications of AI.

Why are intelligent tutoring systems "intelligent"? This name in part reflects the history of the research (Sleeman and Brown, 1982; Wenger, 1987). The people who began this work wanted to contrast their work with traditional, computer-aided instruction, so they called their programs, based on AI programming techniques, Intelligent Tutoring Systems. The name Intelligent Computer Aided Instruction (ICAI) means the same thing.

2.1.3 Artificial Intelligence Contributions in ITS

Almost every "core" area of AI plays a part in instructional programs: knowledge representation, problem solving, natural language, and learning. An important reason for attributing intelligence to these programs is their ability to solve the same problems that they present to the students (Clancey, 1987). This capability enhances its understanding of the student and its ability to offer explanation. The underlying approach is to make latent knowledge manifest. This is accomplished by different forms of reasoning, simulation of problem-solving and/or planning.

A number of representation techniques have been developed in AI research, and most of them have been employed in ITS. Such representations include production rule systems, frame-based systems, semantic networks, and object-oriented inheritance systems. For example, in an early ITS, the SCHOLAR's subject knowledge about South American geography is represented in a semantic network (Carbonell, 1970).