Int J Artif Intell Educ (2016) 26:512–535
DOI 10.1007/s40593-015-0083-7

ARTICLE

Reciprocal Tutoring: Design with Cognitive Load Sharing

Chih-Yueh Chou[1] · **Tak-Wai Chan**[2]

Published online: 1 December 2015

Abstract *Reciprocal tutoring*, as reported in "Exploring the design of computer supports for reciprocal tutoring" (Chan and Chou 1997), has extended the meaning and scope of *intelligent tutoring* originally implemented in standalone computers. This research is a follow-up to our studies on a *learning companion system* in the late 1980s and its network version, Distributed West, in the early 1990s. In this commentary paper, we first provide the history of and rationale behind our research. We pose and discuss six design dimensions that comprise 12 design questions. This is done on the basis of our previous experience and current knowledge as well as by reexamining the design approach, *cognitive load sharing*, in the original paper. Our purpose is to shed light on the future design of reciprocal tutoring. O*ne-to-one* classrooms, in which students learn with their personal computing devices (Chan et al. 2006), are becoming prevalent in practice; therefore, we expect that reciprocal tutoring—learning-by-tutoring and learning-by-being-tutored—will also become widespread.

Keywords Reciprocal tutoring · Cognitive load sharing · Learning by teaching · Learning-by-tutoring · Learning-by-being-tutored · Collaborative learning · Learning companion

"Teaching and learning mutually enhance one another."

Book-of-Rites

✉ Tak-Wai Chan
chan@cl.ncu.edu.tw

Chih-Yueh Chou
cychou@saturn.yzu.edu.tw

[1] Department of Computer Science and Engineering, Yuan Ze University, Zhongli, Taoyuan City, Taiwan, Republic of China

[2] Graduate Institute of Network Learning Technology, National Central University, Zhongli, Taoyuan City, Taiwan, Republic of China

From Learning Companion to Reciprocal Tutoring and Beyond

Examining the foundations of prior research may facilitate future research. Reciprocal tutoring systems (RTSs; Chan and Chou 1997) were an intermediate outcome of a sequence of research that began with the development of learning companion systems (LCSs; Chan and Baskin 1988) in the late 1980s, when Chan was working on his PhD project. An LCS, considered an alternative breed of intelligent tutoring systems (ITSs) in the late 1980s, was modeled after two virtual agents—a virtual learning companion (as a peer student) and a virtual tutor—while interacting with a student (Fig. 1).

Conceptually, the goal of the virtual tutor in the traditional ITS is "to try to alter the student's evolving interpreted knowledge so that it *converges* to the tutor's own, which is a much higher level but rather static one" (Chan 1991, page 4). The virtual tutor mainly monitors learning activities and supports the student to enable "the convergence to be effective". Different from traditional ITS, when students engaged in identifying and evaluating emerging ideas in addition to planning and developing solutions in LCS, they may be challenged by peer students. Hence, students have to defend, unfold, examine, and reflect on their own ideas while identifying possible mistakes of their peers. "Such a process of mutual justification may not easily happen in a student-tutor situation because of different social status and knowledge levels, and thus different expectations. Thus, learning, in an LCS environment, is the *merging* of two evolving versions of interpreted knowledge into the tutor's one" (Chan 1991, page 4) (Fig. 2).

In the general research area of technology enhanced learning, the paper on learning companion published in 1988 was possibly the first paper adopting Vygotsky's theory (1978) on the *proximal zone of development* and *scaffolding*. From Vygotsky's perspective, scaffolding is a temporary form of social assistance where a *more capable* peer assists a student by providing feedback to the student or sharing part of the learning task; this assistance fades gradually as the student masters the task.

How does a student interact with a learning companion? Three "protocols" of learning activities (presently are termed as "scripts" in the area of computer supported collaborative learning, Kobbe et al. 2007) were designed in a prototype system called Integration-Kid (Chan and Baskin 1988) (Fig. 3). The first protocol involved using "the

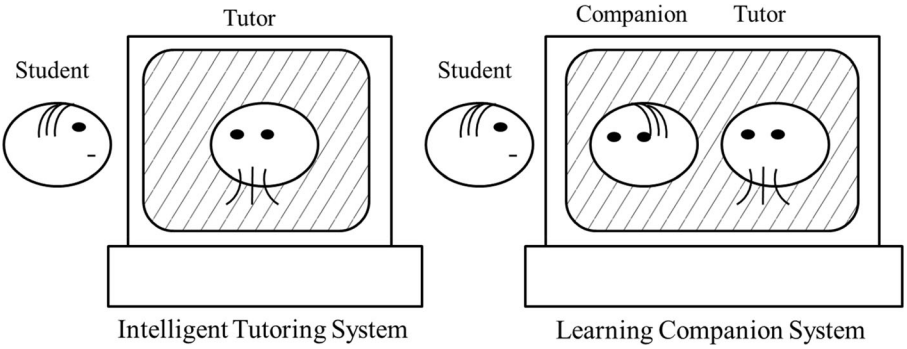

Fig. 1 Intelligent tutoring system and learning companion system

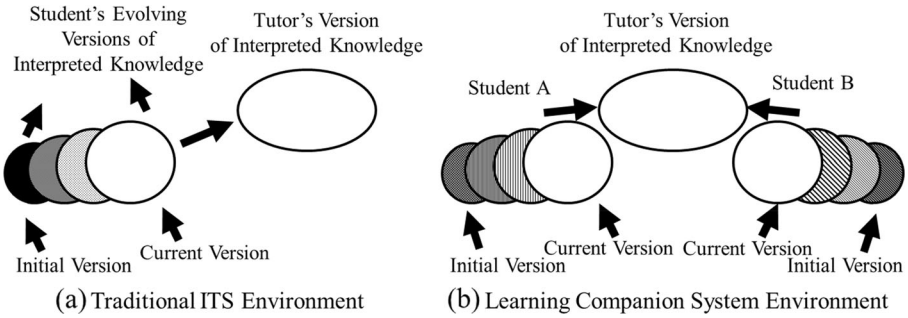

Fig. 2 Dynamic cognitive changes (Chan 1991)

companion as a competitor"; in this system, the student and companion compete when solving a set of problems independently, where the teacher judges the students' work and offers hints. The second and third protocols involved collaborative learning: "one works while the other watches" and "working on the same problem through responsibility sharing." In these two collaborative learning protocols, the student and companion alternate roles when solving problems, whereas the teacher issues problems, demonstrates examples, and offers retrospective comments. The LCS paradigm actually represents a broad spectrum of studies "because of possible variations in the number and identities of agents." For example, the student serves as a tutor and "*learns how to learn by teaching the learning companion*" (Chan and Baskin 1988, page 199).

Distributed West was developed to explore the spectrum of the learning companion studies. In Distributed West, two students used their personal computers connected by a network cable to collaborate with or compete against each other with or without a computer simulated companion (Chan et al. 1992) (Fig. 4). Thus, it extended the first LCS—Integration-Kid—from a standalone system to a networked system and from interacting with two virtual agents (virtual tutor and virtual learning companion) to with more than two agents (including the human agent on another connected computer). Although Distributed West was a natural extension of the first LCS, it is interesting to

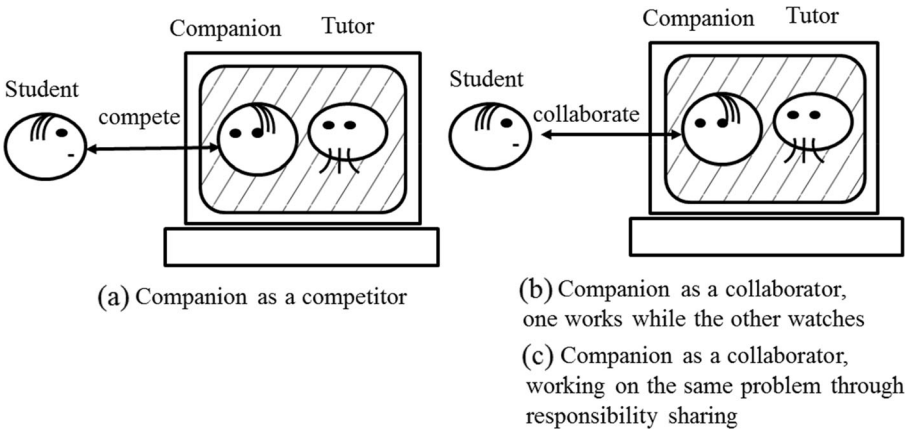

Fig. 3 Three protocols of Integration-Kid

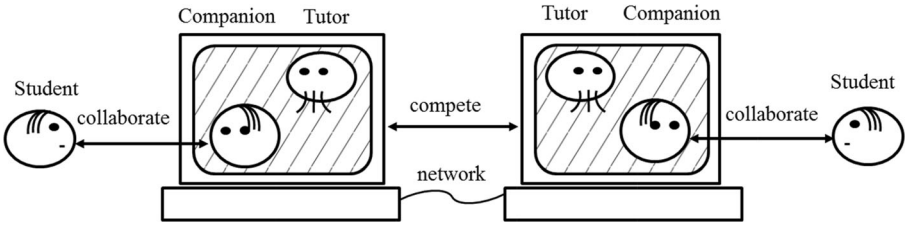

Fig. 4 A protocol of Distributed West

note that it was also possibly the first network learning system dedicatedly developed for students to interact in real time. It supported students to learn collaboratively and competitively with a learning game.[1] When a student played the Distributed West game for learning binary number computation, a virtual tutor was available on each student's computer to offer hints for the learning task if necessary. After implementing the prototype system, we conducted a trial experiment in which, we connected all of the computers in a personal computer lab in pairs to enable a class of freshmen undergraduate students to learn and use the system in pairs (RTSs were a sequel to Distributed West in which two students had a tutor–tutee relationship for one task and switched roles for another task). We foresaw that this computer lab might resemble a classroom in the future, thus we called it a *futuristic intelligent classroom* (Chan et al. 1992). It also indicated that such a classroom with connected computers might represent a network learning community. Currently, we refer to these classrooms as *one-to-one* (*1*:*1*) classrooms (Chan et al. 2006), in which every student learns with a personal computing device, such as notebook, tablet PC, or smart phone. It enables all students to work with their own computers. Notably, in Chinese, the word "computer" is literally translated as "electronic brain." In a classroom with 30 students (thus 30 human brains), if all students are working with their own computers ("electronic brains"), then, as designers, we must evaluate how every student's brain can derived the highest benefit from the other 59 brains: 30 "electronic brains" in addition to 29 human brains. Another perspective on human companions and classrooms with connected computers is to build a network *learning society*. EduCity, started in 2000, was the first attempt to realize this vision. It overcame the confines of school walls and was comprised of a hierarchy of communities that reached more than 1.5 million students, in 15,000 classrooms, within 1800 schools (Chan et al. 2001; Chan 2010; Chang et al. 2003; Atkins et al. 2010).

Another perspective of the learning companion studies is to explore the time span of a student's relationship with their learning companion. This relates to the notion of a "lifelong learning companion," which accompanies a student from childhood to adulthood (Chan et al. 2001; Chou et al. 2003). Cloud computing and storage technology enables storing a student's lifelong learning profiles, hence, establishing students' lifelong learning companions. Furthermore, with the advancement in robot technology, such a companion may become a *lifelong robot companion*.

[1] West was a classic computer learning game (Burton and Brown 1979). Distributed West, developed based on West, was a network version of West.

Emerging Design Problems

LCS research has expanded ITS research by introducing variations on the number, roles, relationships, and knowledge levels of the agents (Chan and Baskin 1988; Chan et al. 1992; Chou et al. 2002a). An agent can be either a human or a computer-simulated character. The agent's role can be a tutor, peer competitor, peer tutor, peer tutee, or collaborator (Chou et al. 2003). An LCS may contain multiple companions, while the agents' levels can be weak or strong as well as similar or different when compared with the human student (Hietala and Niemirepo 1998; Chou et al. 2002a; Uresti and Boulay 2004). We enumerated 768 possible LCS learning protocols by assessing various possible numbers and roles of human students and computers as well as other possible factors affecting student interactions (Chan et al. 1992). One possible protocol involves two agents that have a tutor–tutee relationship for one task and switch roles for another task. The protocol has become the basic form of an RTS. We further explored the design space of RTSs with various combinations of six design dimensions: *task partition*, *social scaffolding* (virtual, robot, and human agents), *scaffolding tools*, *scaffolding and fading*, *student modeling*, *and benefit-cost-tradeoffs* (Chan and Chou 1997) (Fig. 5). Scaffolding talks about how students are assisted or supported to perform a task beyond their abilities as well as how the assistance fades to enable them to perform a task by themselves (Pea 2004). There are two types of scaffolding: social scaffolding and technological scaffolding. Social scaffolding refers to the assistance offered by agents, which may be virtual, robot, or human. Technological scaffolding denotes computer-enabled scaffolding tools to assist learning. The design space of RTSs is similar to the components of a collaboration script: participants, activities, roles, resources, and groups (Kobbe et al. 2007). In RTSs, the main sub-tasks or activities are tutoring (learning-by-tutoring, also termed as learning-by-teaching) and tuteeing (learning-by-being-tutored); the activities are dispatched to multiple agents (participants and group), where humans or computer-simulated characters play such roles as tutor, tutee, collaborator, and competitor. Some RTSs provide scaffolding tools (resources) to assist students in role-playing to complete sub-tasks. Student modeling supports adaptivity to optimize student learning. Benefit-cost-tradeoffs concern the tradeoffs of the benefits and implementation costs of the design. The problems encountered in the design space of RTSs are subsequently discussed.

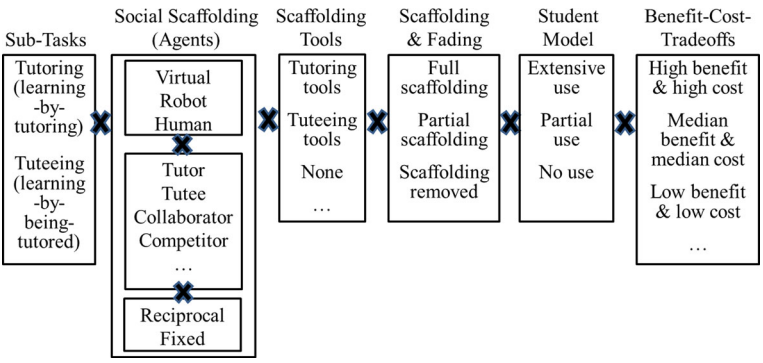

Fig. 5 Design space of RTSs

As defined in the original paper: "*Reciprocal tutoring is a form of cooperative learning by adopting the 'divide and conquer' strategy. A learning task is partitioned into sub-tasks of tutoring and 'tuteeing' where tuteeing is learning by working on the solution while being tutored. These sub-tasks are distributed to different agents and each agent is responsible for a sub-task, while the rest of the work is taken care of by other agents*" (Chan and Chou 1997, page 3). Figure 6 illustrates a reciprocal tutoring example scenario, where a student plays the role of a tutor who tutors a student tutee, where the system plays the role of the tutor's tutor. We can also phrase such as follows:

"The tutor's tutor tutors the tutor who tutors the tutee."

That is, the dyad takes turns playing the roles of tutor and tutee in various learning tasks. Thus, reciprocal tutoring is a type of peer tutoring that enables peer students to tutor other students (Delquadri et al. 1986; Topping 1996; King 1998; King et al. 1998; Rittschof and Griffin 2001). The scope of this paper is to design protocols and computer supports for reciprocal tutoring, such as sub-task partition, roles of agents, virtual agents, and computer scaffolding tools.

An RTS designer must access four "brains" (two students and two computers) involved in a single learning task. The designer must analyze *how the learning task is decomposed*, *which* "*brain*" *performs which part of the learning task*, *and how these* "*brains*" *should interact*. Students may not engage in a collaborative learning activity sufficiently by themselves (Barron 2003); therefore, collaborative scripts are generally designed as instructions and scaffolds to improve collaboration by scripting learning objectives, types of activities, sequencing, role distribution, and types of representation in order to structure the interactive processes among collaborators (Kollar et al. 2006). In designing scripts or protocols for RTSs, learning objectives are learning-by-being-tutored and learning-by-tutoring. The activities are tuteeing and tutoring. Agents, virtual or human, serve as tutors or tutees. Sequencing involves determining whether the roles of the agents are reciprocal. The type of representation involves determining how scripts are presented to students, such as in textual, oral, or graphical presentations. We proposed a cognitive load sharing (CLS) approach for designing RTSs (Chan and Chou 1997).

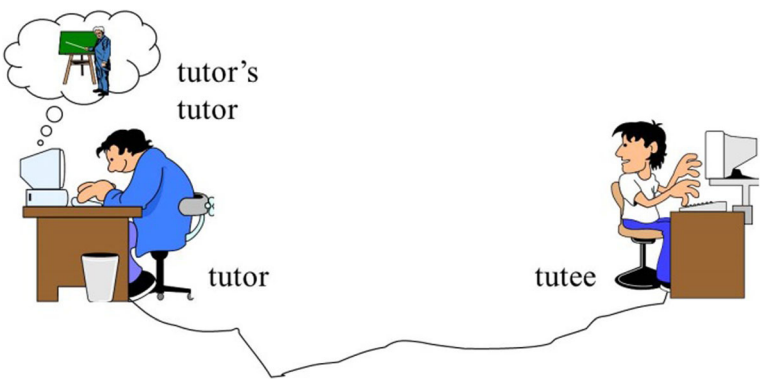

Fig. 6 Reciprocal tutoring example scenario (Chan et al. 2001)

Cognitive Load Sharing Approach

A complex learning task (activity) generally involves many sub-tasks (sub-activities). For instance, solving a problem is a learning task, and comprises sub-tasks such as searching for approaches, constructing a solution, diagnosing a solution if it is not effective, considering alternatives or correcting possible errors, and modifying or reconstructing a solution. As described in the "Conditions of Learning" by Gagne (1985), as parts of a learning task, various factors may be evaluated such as how to draw attention, arouse curiosity, and activate prior knowledge before undertaking the task as well as how to sustain motivation after finishing the task to encourage students to complete another task within the same domain. Gagné suggested that the learning prerequisites that must be completed by a student can be identified by analyzing a learning task. This task analysis process is intended to decompose a learning task and identify which actions or cognitive processes must be undertaken by a student; the analysis process can help a designer gain a reasonable understanding of a learning task. Therefore, to design an RTS, we must conduct a task analysis process in order to identify the actions of the tutor, tutee, and their computers. We referred to this process as CLS.

If the learning task is too complex, students may not be able to address all sub-tasks. For instance, writing a Lisp program is complicated, where students may not be able to understand why their programs are ineffective. When we designed RTSs, we considered, "*cognitive load sharing is a salient feature of most of the systems …* (Chan and Chou 1997, page 7)". The design rationale of the CLS approach is to "divide and conquer" and which entails distributing sub-tasks to computer scaffolding tools or various agents, such as human student tutors, human companions, virtual tutees, or virtual tutors; this enables students to focus on their own sub-tasks, while other agents and computer scaffolding tools address other sub-tasks (e.g., identifying errors and suggesting corrections). Cognitive load, in this context, is the cognitive process that students must use to learn their own assigned sub-tasks. In reciprocal tutoring, students are assumed to do complex learning tasks; additionally, it is assumed that the cognitive load on a single student to complete an entire task is too heavy. Therefore, a learning task must be divided into sub-tasks and distributed to various agents and computer tools in order to share the cognitive load. In the CLS approach, scaffolding is provided in social and technological dimensions by providing real agents, virtual agents, and scaffolding tools (Pea 2004). Csikszentmihalyi's flow thoery (1975) suggests that the challenge associated with a task and a student's skills should be matched to prevent the student from becoming anxious or bored. Specifically, to remain engaged in their tasks, students should undertake tasks with appropriate cognitive loads. Therefore, after becoming proficient in their assigned sub-tasks, students are asked to immediately undertake additional sub-tasks to maintain an appropriate challenge, which constitutes fading.

As we investigated the CLS approach, we were unaware that other researchers were also investigating into the "cognitive load" topic mainly from the working memory perspective, which was reported in the book by Sweller et al. (2011). These researchers proposed cognitive load theory and two types of cognitive load: *Intrinsic* cognitive load, imposed by the intrinsic nature of a material, and *extraneous* cognitive load, imposed by the manner in which a material is presented and can be reduced. An excessive cognitive load leads to ineffective learning. Researchers have thus proposed

many instructional principles for reducing extraneous cognitive load to improve learning. However, if a learning task is fixed and student knowledge levels remain constant, intrinsic cognitive load remains constant. From the perspective of cognitive load theory, the objective of CLS is to alter the intrinsic cognitive load of students by adaptively dividing learning tasks among various agents.

To explore the RTS design, we developed and compared seven RTS variations presented in our previous study (Chan and Chou 1997). Table 1 lists a comparison of the designs of related RTSs according to RTS design space. The configuration includes the number and roles of agents and whether agents are virtual characters or real humans. Companion(s) indicates the agent(s) that interact with a human student.

Upon reexamining our CLS approach, we pose and discuss 12 research questions or concerns that may emerge in the design of reciprocal tutoring. These research questions assist developers in adopting the CLS approach to designing RTSs. They are divided into six design dimensions: task partition (questions #1 and #2), social scaffolding (questions #3 and #4), scaffolding tools (questions #5 and #6), scaffolding and fading (questions #7, #8, and #9), student modeling (questions #10 and #11), and benefit-cost-tradeoffs (question #12)(Table 2). Although we focus on reciprocal tutoring, the questions and principles we address can be applied to other learning activities.

Task Partition Dimension

This dimension consists of two questions: the 'when' question and the 'how' question. The premise question is the 'when' question, concerning whether the task is appropriate for being partitioned into sub-tasks. The How-To-Partition question is the 'how' question, dealing with how a task is to be partitioned into sub-tasks so that they can be distributed to and undertaken by different agents.

#1 Premise Question: Is the CLS Approach Effective? What Are the Premises?

The effectiveness of the CLS approach depends on some premises. The first premise is that a learning task (activity) can be partitioned into several sub-tasks (sub-activities) for various agents. Partitioning some learning tasks into sub-tasks shared among various agents may be difficult. The second premise is that the benefit of the cognitive load reduced by sharing should be greater than the damage of the increased effort resulting from sharing sub-tasks and collaboration among agents. Dillenbourg (2002) argued that over-scripting may force students to interact and solve problems through non-natural methods, thereby increasing their cognitive load. Agents are responsible for parts of sub-tasks, but must collaborate with others to combine sub-tasks in order to complete entire learning tasks. Communicating and working with other agent(s) increases effort and may offset the reduction in cognitive load through sharing. Individual differences exist in this premise. Some students are proficient in collaboration, whereas others are not. The third premise is that other agents must be able to complete their sub-tasks. To explain CLS, consider, for example, a task involving moving a piano. A piano may be too heavy for a single person to move. Four persons can move a piano if they move the piano together. If they lack coordination or if some persons do not contribute to the strength of the group, the group cannot move the piano.

Table 1 Design of related reciprocal tutoring systems

System	Configuration	Companion(s)	Task(s) of student	Reciprocal/fixed	Scaffolding Tool(s)
Integration-Kid (Chan and Baskin 1988; Chan 1991)	RL + VT + VT/RT + VT + VL	VT + VT/VT + VL	Tuteeing/Tutoring	Reciprocal	
Learning by Teaching (Palthepu et al. 1991)	RT + VL	VL	Tutoring	Fixed	
DENISE (Nichols 1994)	RT + VL	VL	Tutoring	Fixed	
STEPS (Vanlehn et al. 1994; Ur and VanLehn 1995)	RT + VL	VL	Tutoring	Fixed	
Distributed Reciprocal Tutoring (Chan and Chou 1997)	RT + RL + VT	RT/RL + VT	Tuteeing/Tutoring	Reciprocal	PLS/DHT
Distributed Responsibility Sharing (Chan and Chou 1997)	RT + RL + RL	RL + RL/RT + RL	Tutoring/Tuteeing	Reciprocal	PLS-SNL/PLS-KEY/DHT
Centralized Reciprocal Tutoring (Chan and Chou 1997)	RL + VT/RT + VL	VT/VL	Tuteeing/Tutoring	Reciprocal	PLS/DHT
Intelligent Tutoring System (Chan and Chou 1997)	RL + VT	VT	Tuteeing	Fixed	PLS
Learning-By-Tutoring (Chan and Chou 1997)	RT + VL + VT	VL + VT	Tutoring	Fixed	DHT
Self Diagnosing (Chan and Chou 1997)	RL + VT	VT	Tuteeing	Fixed	PLS/DHT
Working Alone (Chan and Chou 1997)	RL	No	Working alone	Fixed	PLS
PAL (Reif and Scott 1999)	RL + VT/RT + VL	VT/VL	Tutoring/Tuteeing	Reciprocal	
DwestAgent (Chou et al. 2002a)	RT + VL/RL + VT	VL/VT	Tutoring/Tuteeing	Fixed	Recommendation menu
Reciprocal Tutoring System (Wong et al. 2003)	RT + VL/RL + VT/RT + RL/	VL/VT/RL/RT	Tutoring/Tuteeing	Reciprocal	PLS/DHT/Dialogue templates
LECOBA (Uresti and Boulay 2004)	RT + VL	VL	Tutoring	Fixed	Teaching window
Betty's Brain (Biswas et al. 2005; Leelawong and Biswas 2008)	RT + VL + VT	VL + VT	Tutoring	Fixed	Concept map editor
My-Pet (Chen et al. 2007, 2012; 2013)	RT + VL	VL	Tutoring	Fixed	
SimStudent (Matsuda et al. 2010, 2013)	RT + VL	VL	Tutoring	Fixed	Option menu
APTA (Walker et al. 2011, 2014)	RL + RT + VT	RL+ VT/RT	Tutoring/Tuteeing	Reciprocal	Adaptive help-giving support
Moby (Okita and Schwartz 2013)	RT + VL	VL	Tutoring	Fixed	Propositional and matrix menu
OPAL (Evans and Moore 2013)	RT + RL	RL/RT	Tutoring/Tuteeing	Variable reciprocity	a tutor pool and a tutoring ticket list
ProJo (Okita 2014)	RL/RT + VL	No/VL	Working alone/Tutoring	Reciprocal	

The Working Alone system, which has a PLS tool and does not have companions, is used as a baseline for comparison with other systems

V virtual, *R* real human, *T* tutor, *L* learner (tutee), *PLS* Petal-Like-System, *DHT* Diagnosis-Hint-Tree, *PLS-SNL* Petal-Like-System in Semi-Natural Language, *PLS-KEY* Petal-Like-System with KEY in

Table 2 Research questions of six dimensions of RTS design space in CLS approach

Dimension	Questions
Task partition	#1 Premise Question #2 How-To-Partition Question
Social scaffolding	#3 Role-Assignment Question #4 Agent-Genus Question
Scaffolding tools	#5 Scaffolding-Tool-Type Question #6 Agent-Tool-Combination Question
Scaffolding and fading	#7 Scaffolding-Effect Question #8 Fading-Effect Question #9 Combined-Scaffolding-Effects Question
Student modeling	#10 Student-Modeling-Benefit Question #11 Big Data Question
Benefit-cost-tradeoffs	#12 Benefit-cost-tradeoffs

Issues of how to distribute sub-tasks to different agents involve two questions: how to partition a learning task (activity) into several sub-tasks (sub-activities) for various agents (Question #2) and how to appropriately assign roles of agents for sub-tasks or sub-activities (Question #3).

#2 How-To-Partition Question: How Can a Learning Task (Activity) Be Partitioned into Several Sub-Tasks (Sub-Activities) for Various Agents?

First, some learning activities include several sub-activities. For example, the reciprocal teaching method, which entails training students in comprehension, includes four activities: summarizing, questioning, clarifying, and predicting (Palinscar and Brown 1984). These sub-activities can be distributed to various agents. Second, if a learning task covers several topics, the task can be partitioned into sub-tasks of mastering various topics, such as the Jigsaw method (Aronson et al. 1978).

In reciprocal tutoring, learning tasks are generally partitioned into tuteeing and tutoring for two agents. However, the types of tuteeing and tutoring vary among different domains and tutoring approaches. For examples, tuteeing in the Distributed Reciprocal Tutoring system involves designing and implementing a program to solve a problem (Chan and Chou 1997). Tuteeing in DwestAgent involves composing an arithmetic expression with three specific numbers to progress forward toward the goal (Chou et al. 2002a). Tutoring involves activities, such as critiquing, detecting and fixing errors, hinting, and demonstrating. However, there are different tutoring approaches involving different activities. For example, tutoring in DENISE involves demonstrating a causal qualitative model of economics (Nichols 1994). Tutoring in the Distributed Reciprocal Tutoring system involves diagnosing and providing hints to assist tutees (Chan and Chou 1997). RTSs are intended to help student tutees learn by being tutored by tutors (learning-by-being-tutored) and help student tutors learn by tutoring tutees (learning-by-tutoring). One design problem is determining the type of tutoring approach that can help student tutees to learn effectively. Vanlehn (2011) compared the effectiveness of human tutoring and ITSs with various granularities of tutoring interaction: answer-based, step-based, and substep-based tutoring. The results revealed that step-based, substep-based, and human tutoring are more effective than

answer-based tutoring. Another design problem is determining the type of tutoring approach that can help student tutors effectively learn by teaching. Researchers have reported that students benefit from learning-by-teaching through preparing to teach, presenting learning materials, monitoring tutees' tasks and finding errors, and observing recursive feedback from tutees as well as demonstrating, explaining, and responding to tutee's questions (Bargh and Schul 1980; Renkl 1995; Roscoe and Chi 2007, 2008; Okita and Schwartz 2013; Okita 2014). If an RTS involves student tutor(s) and tutee(s), tutoring interactions should be designed for help student tutor(s) and tutee (s). Determining the type of tutoring approach may involve a trade-off between helping student tutors and helping student tutees. The trade-off can be solved by providing virtual agents (Question #4).

Social Scaffolding Dimension

After partitioning the task into sub-tasks, we have to consider how to design social scaffolding. There are two design questions in this dimension. The role-assignment question refers to how these sub-tasks are being assigned to different agents. In order to match the natures of different sub-tasks to assignees, the assignment has to take into account the number and roles of the agents as well as whether the roles of the agents are fixed or reciprocal. For agents, we also have to consider whether an agent is virtual, robot, or human—the agent-genus question.

#3 Role-Assignment Question: How Are Agents Appropriately Assigned Roles for Sub-Task(s) or Sub-Activities?

When sub-tasks are assigned to various agents, collaboration scripts should be clearly defined. This also applies to specifications and relationships among agents, in addition to learning sub-tasks, such as the roles of agents and sequences of activities. Doing so enables agents to be responsible for their learning sub-tasks and collaborate to complete the entire learning task. This question involves the number and roles of agents and whether the roles of the agents are reciprocal or fixed.

Number and Roles of Agents

Sub-task(s) may be assigned to agents through various methods. A sub-task can be dispatched to an agent or group of agents. In contrast, an agent can be responsible for several sub-tasks. For instance, a student is assigned to play the role of designing a solution, while another student as translator to implement the solution in the Distributed Responsibility Sharing system; meanwhile, a student is assigned to play the role of tutee to design and implement a solution in the Distributed Reciprocal Tutoring system (Chan and Chou 1997). The agent's role can be a tutor, peer competitor, peer tutor, peer tutee, or collaborator (Chou et al. 2002a, 2003). The competence of the agents also influences agent roles, such as weak companion and strong companion (Hietala and Niemirepo 1998). Researchers have found that students have various preferences regarding the roles and competencies of agents (Hietala and Niemirepo 1998; Chou et al. 2002a; Uresti and Boulay 2004). The appropriate number and roles of agents

depend on the amount and difficulty of sub-tasks and capabilities of students. A higher number of sub-tasks, greater difficulty of sub-tasks and lower ability among students may necessitate sharing sub-tasks among higher number of agents; such may lead to a higher complexity and effort regarding communication and collaboration. A protocol is required to assist agents in playing their roles to complete their sub-tasks and to collaborate in completing the entire learning task. For example, a student plays the role of solution designer; then a student plays the role of translator to implement the designed solution in the Distributed Reciprocal Tutoring system (Chan and Chou 1997). Simultaneously, a student plays the role of tutor to tutor the designer and translator.

Reciprocal or Fixed

Agent roles can be reciprocal or fixed (see Table 1). A reciprocal protocol enables students to alternate between learning various sub-tasks in order to understand all sub-tasks; whereas a fixed role enables students to focus on learning particular sub--tasks. However, some sub-tasks may be more difficult than others. For example, a student may be able to solve a problem (sub-tasks for a tutee), however, unable to tutor other students (sub-tasks for a tutor). Students with different capabilities can be assigned to various roles for sub-tasks involving various difficulty levels. For example, students with higher capabilities can be assigned to tutor those with lower capabilities. The roles can be changed according to the advances in the capability levels of students. For instance, novice students can be assigned to play the role of tutee, intermediate students can be assigned to reciprocally play the roles of tutee and tutor, and expert students can be assigned to play the role of tutor. Evans and Moore (2013) proposed variable reciprocity in which students play the roles of tutor and tutee for various problems depending on whether or not they accurately solve the problems. Students are tutors for problems they solved and are tutees for problems they do not solve. Students who accurately solve a problem gain authority to be tutors for the problem in order to help other students (tutees for the problem) solve the problem.

#4 Agent-Genus Question: What Types of Agents Can Share Cognitive Load (Sub-Tasks)?

Two types of agents can share cognitive load: human and artificial agents. Artificial agents are computer simulated characters, including virtual and robot agents. Although most of the artificial agents in existing RTSs are virtual agents (see Table 1), robot agents exhibit benefits in enriching interactions between a system and students, while increasing a student's engagement (Hsu et al. 2007; Chen et al. 2014). RTSs generally involve both human and virtual agents, however, may only involve human agents, such as in a Distributed Responsibility Sharing system (Chan and Chou 1997). Human agents can be recruited from classmates, senior students, and teachers. Teachers can be recruited as effective tutors (Chi et al. 2001), however, such increases teachers' loads. Providing virtual agents as teaching assistants can therefore serve to benefit teacher load (Shindo and Matsuda 2001; Chou et al. 2011). Students can be recruited as peer tutors or collaborators; however, they may not be able to complete their sub-tasks, sufficiently collaborate with other agents, or satisfy the demands of roles, such as strong

companions. Evans and Moore (2013) proposed variable reciprocity to better ensure that tutors for a program have correctly solved the problem. Some systems provide scaffolding tools for remedying these problems (Question #5).

Virtual agents, which share cognitive load, are simulated by computers, such as virtual tutors and tutees (see Table 1). Providing virtual agents can eliminate the problem of lacking appropriate human agents; additionally, they can solve the tradeoffs of determining a tutoring approach to assist both student tutors (learning-by-tutoring) and student tutees (learning-by-being-tutored). Adaptive virtual agents can be developed to meet individual requirements of students by such methods as creating adaptive collaboration scripts (Diziol et al. 2010), providing adaptive scaffolding feedback (Kinnebrew et al. 2013), and playing various roles with different knowledge levels (Chou et al. 2002a). Intelligent virtual tutors have been proven to be effective in helping students (Anderson et al. 1995; Koedinger and Corbett 2006; Vanlehn 2011). Researchers have also reported that virtual agents' adaptive prompts for self-regulated learning strategies (Kinnebrew et al. 2013) and recursive feedback (Okita and Schwartz 2013) were helpful for learning-by-teaching. However, developing such virtual educational agents is complex and labor-intensive (Murray 1999; Chou et al. 2002b, 2003). Generally, interaction data among real agents must be collected to develop virtual agents.

Artificial agents can be presented not only on screens as virtual agents (Johnson et al. 2000) but also as physical robot agents (Hsu et al. 2007; Chen et al. 2014). The physical robots enrich interactions between a system and students by enabling tangible inputs, such as physical manipulation and touch, and by providing outputs through robots, such as sounds, motions, and light. An augmented reality technique similarly enables students to manipulate physical objects, however, receive outputs from screens (Oh and Woo 2008; Chen et al. 2014). Physical robots and an augmented reality technique can enhance a variety of artificial agents; this approach can enrich student learning experiences and increase student engagement by combining multiple sensorial media, such as visual, auditory, kinesthetic, and tactile.

In CLS, agents are used to share cognitive load and to increase student motivation. When students collaborate with other agents, students may feel responsible and make efforts to complete their sub-tasks. Agents may also encourage, criticize, or praise students to enhance student motivation and, consequently, learning performance. For example, the My-Pet system provides a trainable animal companion to enhance student motivation (Chen et al. 2012, 2013). Studies have also revealed that various students prefer different levels of agents (Hietala and Niemirepo 1998; Chou et al. 2002a; Uresti and Boulay 2004). Regarding CLS, the effects of agents on student motivation may vary among students and should be investigated further.

Scaffolding Tools Dimension

After being dispatched a sub-task, the human agent may need some scaffolding tools to accomplish the sub-task. There are two design questions in this dimension. The scaffolding-tool-type question is about what types of scaffolding tools are needed by the human agent. The agent-tool-combination question is about seeking the most appropriate combination of agents and tools in order to assist the student.

#5 Scaffolding-Tool-Type Question: What Types of Scaffolding Tools Can Be Applied for Reciprocal Tutoring or Other Learning Activities Involving CLS?

Computer-enabled scaffolding tools help students in completing learning tasks and sharing cognitive load. Two types of scaffolding tools are generally used for reciprocal tutoring or other learning activities involving CLS: (1) cognitive tools for assisting agents in completing learning sub-tasks, such as tuteeing tools for tutees and tutoring tools for tutors; (2) tools for assisting agents in communication and collaboration, such as mechanisms for finding and suggesting potential peer helpers in the PHelpS system (Greer et al. 1998) and dialogue templates in the Reciprocal Tutoring System (Wong et al. 2003). These scaffolding tools provide scaffolding to reduce the cognitive load of students playing various roles. For instance, the Distributed Reciprocal Tutoring system provides a tuteeing tool, PLS (Petal-Like-System), to assist human tutees in constructing a recursive Lisp program without making syntax errors and a tutoring tool; further a DHT (Diagnosis-Hint-Tree) is offered to assist human tutors in locating errors and providing hints via an interactive diagnosis interface and several adaptive hints (Chan and Chou 1997). PHelpS enables peer helper selection and mediates communication (Greer et al. 1998). OPAL includes the mechanisms of a tutor pool and a tutoring ticket list for enabling students to find available tutors with verifiable skills (Evans and Moore 2013). The APTA system provides adaptive support to enhance the tutoring ability of peer tutors (Walker et al. 2011, 2014). Reiser (2004) proposed two scaffolding mechanisms of tools: structuring and problematizing. The structuring mechanism "reduces complexity and choice by providing additional structure to the task." The problematizing mechanism "increases the utility of the problem solving experience for learning" by helping students "view something as requiring attention and decision-making." For instance, PLS, as a tuteeing tool, provides code chunks as a structure, and DHT, as a tutoring tool, has an interactive diagnosis structure to reduce complexity (Chan and Chou 1997). In PLS and DHT, students are required to indicate whether each line of a recursive Lisp program code is a recursive or base case. The indication is a problematizing mechanism to focus attention by marking critical features of a recursive program.

Cognitive tools should also be faded to enable students to complete sub-tasks without cognitive tools, if this is the learning goal. Some tools are domain-specific, and some are domain-general. Domain-specific tools generally provide more in-depth assistance than other tools do; however, they are applicable only to specific domains. By contrast, domain-general tools are applicable to several domains, however, may provide only limited assistance. Some tools are adaptive to students and are thus more helpful than non-adaptive tools; however, developing adaptive tools involves student modeling (Question #10) and is complex.

#6 Agent-Tool-Combination Question: Which Combinations of Agents and Tools Are Helpful for Learning?

Numerous combinations of agents and tools can be used in CLS designs. Which combinations are helpful for learning? In the original paper, the learning effectiveness of seven systems was compared with various combinations of types and roles of agents and tools (Chan and Chou 1997). The preliminary results revealed that students who reciprocally played the roles of tutor and tutee with the assistance of a tutoring and

tuteeing tool registered higher posttest scores than did students who maintained the role of tutor with the assistance of a tutoring tool and who were not supported by the scaffolding of load sharing. This may because those student tutors who maintained the role of tutor with the assistance of a tutoring tool did not solve problems by themselves. However, the preliminary experimental trial included few participants; this problem must be investigated further. This problem leads to the next question regarding how to evaluate various types of CLS designs.

Scaffolding and Fading Dimension

The mechanisms of scaffolding and fading must be coupled together. There are three questions in this dimension. The scaffolding-effect question concerns whether the scaffolding provided can really assist students in learning as well as whether or not students are able to learn by themselves after scaffolding has been removed. The fading-effect question deals with when and how fading should be carried out. The combined-scaffolding-effects question discusses about planning and monitoring how and when various scaffoldings should be provided to the student while some of the scaffoldings are fading.

#7 Scaffolding-Effect Question: What Are the Effects with and of Various Types of CLS Designs?

Traditionally, scaffolding comprises two phases: cognitive process support and fading of support, which can be expressed as follows:

$$\text{scaffolding} = \text{cognitive process support} + \text{fading}$$

CLS is a form of cognitive support. Therefore, scaffolding involving CLS can be expressed as follows:

$$\text{scaffolding} = \text{CLS} + \text{fading}$$

CLS provides students with scaffolding by offering social scaffolding (real agents, virtual agents, and robot agents) and technological scaffolding (scaffolding tools) in order to share cognitive load. Furthermore, CLS fades scaffolding by gradually reducing sharing cognitive load. The total effects of CLS designs includes those with (i.e. with scaffolding) and of (i.e. after the removal of scaffolding) the designs. The effects with CLS are concerned with whether students can complete learning tasks under the scaffolding condition. The effects of CLS are concerned with whether or not students still complete learning tasks without the scaffolding (i.e., after fading). A perfect result is that students can complete learning tasks with scaffolding and after the removal of scaffolding. However, the effect of CLS can be diminished under specific conditions. For example, researchers have reported that some students may be "gaming" an ITS by excessively asking for help from the system in order to complete the problems (Baker et al. 2008). Although these students may complete learning tasks with the help of the system, they may not learn the material, which leads to Question #8.

#8 Fading-Effect Question: How Can CLS Scaffolding Be Appropriately Faded?

When students become proficient in their assigned sub-tasks, the tools should be removed, or more sub-tasks should be distributed to these students in order to maintain appropriate difficulty levels; this is called a fading mechanism. The question pertains to when and how fading should be carried out. The objective of fading mechanisms is to make students responsible for more sub-tasks when they show improved skill levels until such students can complete all sub-tasks by themselves. However, fading too slowly might hinder students from learning sub-tasks whereas fading too rapidly might frustrate students. An appropriate fading of scaffolding guides students to learn and prevents them from excessively depending on scaffolding. Designing a protocol or some rules might enable students to perform fading processes by themselves. For example, an RTS applies a scoring mechanism, which calculates scores according to a tutee's help requests; further, tutors receive assistance from tutoring tools; such is offered in order to prevent tutees from excessively depending on tutors and to prevent tutors from excessively depending on the scaffolding of tutoring tools (Chou et al. 2002b; Wong et al. 2003). However, fading can be conducted adaptively by systems per each student. Establishing student models of student knowledge and behavior is crucial in appropriately fading scaffolding, which leads to Question #11. For instance, Del Solato and Du Boulay (1995) implemented an ITS to model student performance, confidence, effort, and independence in order to provide adaptive instruction.

#9 Combined-Scaffolding-Effects Question: How Can a CLS Be Designed to Be Effective With Combinations of Various Scaffolding and Fading Mechanisms?

A complete CLS design includes combinations of various scaffolding and fading mechanisms in order to manage appropriate cognitive load. Many possible combinations of scaffolding and fading mechanisms can be used. The combined-scaffolding-effects question is concerned with planning and monitoring how and when various scaffoldings should be provided to the student while some of the scaffoldings are fading. Sub-tasks may have different difficulty levels or order relationship; hence, the combined-scaffolding-effects question is concerned with how to organize sequences of sub-tasks and scaffoldings in order to maintain manageable cognitive load for students. For instance, Table 3 lists an example of scaffolding and fading mechanisms within different learning stages for a novice student. A novice student is initially assigned as a tutee with the scaffolding of a tuteeing tool, with a virtual tutor in stage 1. The

Table 3 An example of scaffolding and fading mechanisms within different learning stages

	Stage 1	Stage 2	Stage 3	Stage 4
Student subtasks	Tuteeing	Tuteeing	Tuteeing/Tutoring	Tuteeing/Tutoring
Student roles	Tutee	Tutee	Tutee/Tutor	Tutee/Tutor
Social scaffolding	Virtual tutor	Virtual tutor	Virtual tutor/Virtual tutee	Virtual tutor/Virtual tutee
Scaffolding tools	Tuteeing tools		Tutoring tools	

scaffolding of the tuteeing tool fades when the student performs adequately as a tutee in stage 2. Thereafter, the student is assigned to play the roles of tutee and tutor reciprocally, in order to become more effective via learning-by-tutoring in stage 3. The cognitive load of the student increases when the student is asked to play the role of tutor. A tutoring tool can be provided as a scaffolding to lessen the increased cognitive load. Thereafter, the tutoring tool fades in stage 4. Finally, students serve as tutees and tutors by themselves. The changes of different scaffoldings can be designed within an obvious change of learning stages or as an imperceptible change during student learning.

Student Modeling Dimension

Student modelling supports adaptivity and optimization of student learning, which are desirable in the social scaffolding, scaffolding tools, and scaffolding and fading dimensions. There are two questions in this dimension. The student-modeling-benefit question handles how student modeling can be applied in these dimensions for two purposes. Another question is the big data question. Since big data can be obtained in a networked learning community, the question pertains to how big data can be used for building student modeling and virtual agents as well as assessing and improving CLS designs.

#10 Student-Modeling-Benefit Question: What Are the Roles and Benefits of Applying Student Modeling in CLS?

Student modeling entails using several techniques to detect student domain cognition, meta-cognition, affection, cognitive and learning style, and interaction in groups; such is key to providing students with individual, adaptive, and intelligent instructions or support (Greer and McCalla 1994). The student model question in CLS involves how to apply student modeling to adapt and optimize sub-task partition, agents, tools, and scaffolding and fading mechanisms to the needs of students. However, adaptation and optimization are difficult challenges needing evaluative results from various CLS designs, which lead to Question #11. Student modeling can be applied to building virtual tutors (ITSs) for providing adaptive tutoring (Woolf 2008), and to providing adaptive support for collaborative and peer assisted learning, such as finding potential peer helpers (Greer et al. 1998), group formation, domain-specific support, and peer interaction support (Magnisalis et al. 2011). Student modeling can also be used to simulate virtual tutees (Chou et al. 1999), and to provide adaptive scaffolding tools, such as DHT, a tutoring tool for peer tutors (Chou et al. 2002b). Student models can be open for students to raise self-reflection and interaction among students (Bull 2004; Bull and Kay 2007). In addition, open student models can be applied to portray animal companions or avatars in order to motivate students to learn and help others (Chen et al. 2007, 2012). Open student models can also be applied in CLS. Student models can also serve as a clone (virtual avatar) of a student, which can interact with other students when the student is off-line or unavailable (Chen 2014). In summation, the possible roles and applications of student modeling in CLS are diverse and require further investigation.

#11 Big Data Question: How Can Big Data Be Used for Implementing Student Modeling and Virtual Agents As Well As Evaluating and Refining CLS Design?

Implementing virtual agents and student models is complex and labor-intensive. Computer systems can collect, in real-time, a large quantity of user data in a runtime stream (big data); therefore, big data regarding interactions among agents can be used for designing and implementing virtual agents as well as providing adaptation to optimize student learning in real-time. For example, big data can be collected and analyzed in real-time in order to identify possible problems when students play various roles, to provide useful assistance for students, and to offer virtual agents in various roles. However, real agents and big data regarding real agents can also be used to complement machine intelligence (virtual agents and adaptive tools) and human intelligence (assistance from real agents) in order to share cognitive load and reduce the complexity of creating virtual agents and student models (Chou et al. 2003, 2011). Machine intelligence can be applied to extend or reuse human intelligence in order to reduce the load on humans; conversely, human intelligence can be used to share the load on machines. For instance, a virtual teaching assistant system collects a teacher's hint for a specific student problem situation and reuses the hint for students in the same situation (Chou et al. 2011). PHelpS applies student modeling to determine appropriate peer helpers to assist students (Greer et al. 1998).

Designing appropriate RTSs to adapt to different students and to optimize student learning is challenging because the design space of RTSs contains many possibilities and students have various preferences, personalities, and knowledge levels. The CLS provides an approach to design RTSs by considering previous research questions. However, answering previous research questions and developing adaptation of CLS in RTSs require evaluative results from different CLS designs in RTSs. Because RTSs extend the meaning and scope of ITSs, evaluation methodologies for ITSs (Mark and Greer 1993) can also be applied in evaluating CLS designs in RTSs. Researchers suggest the emerging trends of applying big data in evaluations (Greer and Mark 2015) and individualization of computer assisted learning systems (Mayer-Schönberger and Cukier 2014). Big data can be used to evaluate the effectiveness of CLS, offer answers to the previous research questions, refine CLS schemes, and adapt CLS in RTSs. For example, big data can be used to determine whether or not a design works, task partition and role assignment are suitable for students, scaffolding and fading are appropriate, tools are helpful, and virtual agents are necessary. The big data question concerns data that should be collected, how such data should be analyzed, and how this analysis can be used to refine the design, implementation, and individualized adaptations of RTSs.

Benefit-Cost-Tradeoffs Dimension

The gaining of benefits of social scaffolding, scaffolding tools, scaffolding and fading, and student modeling are usually at the expense of the implementation cost. Designers have to consider the tradeoffs of the benefits and costs. There is only one question in this dimension.

#12 Benefit-Cost-Tradeoffs Question: What Are the Tradeoffs of Benefits and Implementation Costs of Various Types of CLS Designs?

The benefits and costs of CLS designs should be evaluated. For example, although scaffolding tools and virtual agents may be useful and adaptive, their development procedures are complex and labor-intensive. In addition, developing virtual agents generally requires using data collected from real agents. Human agents are easier to find, however, might be incapable of performing sub-tasks. Although domain-specific tools may be more helpful than domain-general tools, they require considerable implementation effort. Therefore, designers must balance the costs and benefits of various types of CLS designs.

In summary, applying the CLS approach to designing an RTS involves the practical issues of analyzing and partitioning sub-tasks (Questions #1 and #2), determining agent roles and agent genuses for social scaffolding (Questions #3 and #4), selecting scaffolding tools (Questions #5 and #6), planning scaffolding and fading mechanisms (Questions #7, #8, and #9), applying student modeling to support adaptivity in order to optimize student learning (Questions #10 and #11), and considering benefit-cost-tradeoffs of the design (Question #12).

Final Remarks

As argued by Chan (2010), when we talk about pedagogical models for deep and complex learning, we usually relate them to inquiry-based learning, collaborative problem learning, reflective learning, group competition games, and multimedia storytelling, among others. Learning-by-tutoring could actually incorporate most elements of these models. For example, a comprehensive implementation of learning-by-tutoring demands "learning about the material which means learning and re-learning to ensure full understanding of the materials; searching for supplementary material on the web; composing teaching materials by designing and constructing expository exercises, problems, questions, and answers; negotiating with peers when synthesizing their different constructed lectures into one; and conducting instruction through expository instruction, demonstrations, explanations, and asking or answering questions" (Chan 2010, page 40).

Also, from the design point of view, learning-by-tutoring, to some extent, subsumes learning-by-being-tutored. This means that if we design learning-by-tutoring first, then some elements developed in learning-by-tutoring can be reused in the design of learning-by-being-tutored, thereby making the design of the later simpler.

In this paper, we explore the design space of RTSs in six dimensions and propose related design questions. The design space exhibits a variety of RTSs from a basic form to complex forms. The basic form of RTSs is partitioning a learning task into several sub-tasks and dispatching these sub-tasks to different agents, without including scaffolding tools, student modeling, and fading mechanisms. The basic form works to share cognitive load although it is simple. Scaffolding tools, student modeling, and fading mechanisms enrich the design of RTSs and enable RTSs to assist students further in learning. However, the enrichment increases the complexity, which may increase student effort to engage in learning, thus, leads to the Premise Question. The enrichment also increases implementation costs, which lead to the Benefit-Cost-Tradeoffs Question. The

design space provides designers with various possible configurations of RTSs with different CLS mechanisms; however, involving as many CLS mechanisms as possible may not lead to a good RTS. Designers can create a RTS initially in a basic form, assess the effect of the basic form, and evaluate the necessity of adding scaffolding tools, student modeling, and fading mechanisms. If the evaluation reveals such a necessity, the designers can determine to add more CLS mechanisms in the RTS. After adding more CLS mechanisms, the new version of the RTS must be compared to the previous version of the RTS in order to evaluate the effect of the added mechanisms. In sum, the design space of RTSs provides a framework for designing and investigating RTSs in the CLS approach.

In this paper, we explain the background and rationale behind our investigation into computer support for reciprocal tutoring. On the basis of the CLS approach, we identified 12 design questions for future RTS design tasks. When we researched LCS in the late 1980s, the world was dominated by standalone computers. In the mid-1980s, some researchers explored *asynchronous* learning through the Internet (the term "Internet" became popular in the late 1980s) developed by the U.S. government. The Virtual Classroom (Hiltz 1994) project implemented at the New Jersey Institute of Technology is an example. Distributed West development initiated in 1990, was perhaps the first dedicated network system developed for *synchronous* learning; in other words, this system enabled students to learn with their own computers while interacting with one another in real time. Within Distributed West, students were collaborators or competitors; while in RTSs, they engaged in tutor-tutee relationships with one another. Currently, mobile, cloud, and big data computing are prevalent; regarding learning, we are entering the one-to-one educational computing, seamless learning, and game-based learning era (Chan et al. 2006; Chan 2010) that will soon be pervasively adopted in real world education. As described in the CLS approach, when a student is working with a computer, the roles of the computer mainly fall into one of four categories: cognitive tools, virtual agents (called non-player characters in computer games), communication mediators with other online human agents (which may appear as avatars), or clones (virtual avatars) that represent human agents who are off-line or unavailable. Additionally, peer relationships mainly fall into one of three categories: collaborators, competitors, or tutor-tutee relationships. Furthermore, numerous studies in our field have proven that learning-by-being-tutored and learning-by-tutoring are effective and valuable (Anderson et al. 1995; Bargh and Schul 1980; Koedinger and Corbett 2006; Okita and Schwartz 2013; Okita 2014; Renkl 1995; Roscoe and Chi 2008; Vanlehn 2011). Therefore, we expect reciprocal tutoring to become widespread in one-to-one classrooms over the next 10 years.

Currently, our perspective on our field has broadened as the world has undergone rapid changes over the past 25 years. When we worked on LCS and Distributed West between the late 1980s and early 1990s, researchers in our field considered our work as "alternatives to one-to-one tutoring." For example, our paper on Distributed West in the proceedings of the ITS 1992 conference was placed in the category of "alternatives to one-to-one tutoring." This view reflected the domain of standalone computers. However, the picture is clearer now as reciprocal tutoring is a natural extension of intelligent tutoring, from the standalone-computer world to the current networked world. Principally, reciprocal tutoring may be a form of classroom learning that potentially helps to resolve Bloom's two-sigma problem (Bloom 1984), considered by many as the 'Holy Grail' of ITS research.

This paper reexamines the CLS approach for deigning RTSs and poses 12 design questions. The answers, however, to most of these design questions remain unclear and demand further exploration. The key objective of the CLS approach is to enable students to learn effectively under adaptive scaffolding in order to maintain appropriate cognitive load. Cognitive load theory (Sweller et al. 2011) provides approaches to measure students' cognitive load in learning and principles to design effective learning systems; it is helpful for examining the CLS approach and provides answers to design questions. From the perspective of cognitive load theory, the aim of the CLS approach is to alter the intrinsic cognitive load of students by adaptively dispatching learning tasks to various agents and tools. The CLS approach, however, may lead to an increase in extraneous cognitive load. The search for answers of the design questions posed in this paper may lead to the establishment of the principles of the CLS approach that can be used to design effective RTSs. Nevertheless, we hope that this paper paves a road for future research on learning-by-tutoring and learning-by-being-tutored in the era of the connected and robotic world.

References

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: lessons learned. *The Journal of the Learning Sciences*, *4*(2), 167–207.

Aronson, E., Blaney, N., Stephan, C., Sikes, J., & Snapp, M. (1978). *The jigsaw classroom*. Beverly Hills, CA: Sage.

Atkins, D. E., Bennett, J., Brown, J. S., Chopra, A., Dede, C., Fishman, B., et al. (2010). *National educational technology plan 2010, draft, office of educational technology*. U.S. Department of Education.

Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A., & Koedinger, K. (2008). Why students engage in "gaming the system" behavior in interactive learning environments. *Journal of Interactive Learning Research*, *19*(2), 185–224.

Bargh, J. A., & Schul, Y. (1980). On the cognitive benefits of teaching. *Journal of Educational Psychology*, *72* (5), 593–604.

Barron, B. (2003). When smart groups fail. *The Journal of the Learning Sciences*, *12*(3), 307–359.

Biswas, G., Leelawong, K., Schwartz, D., Vye, N., & The Teachable Agents Group at Vanderbilt (2005). Learning by teaching: a new agent paradigm for educational software. *Applied Artificial Intelligence*, *19* (3–4), 363–392.

Bloom, B. (1984). The 2 sigma problem: the search for methods of group instruction as effective as one-to-one Tutoring. *Educational Researcher*, *13*(6), 4–16.

Bull, S. (2004). Supporting learning with open learner models. *Planning*, *29*(14), 1.

Bull, S., & Kay, J. (2007). Student models that invite the learner in: the SMILI: open learner modelling framework. *International Journal of Artificial Intelligence in Education*, *17*(2), 89–120.

Burton, R. R., & Brown, J. S. (1979). An investigation of computer coaching for informal learning activities. *International Journal of Man-Machine Studies*, *11*(1), 5–24.

Chan, T. W. (1991). Integration-kid: A learning companion system. In J. Mylopolous, & R. Reiter (Eds.), *Proceedings of the 12th international conference on artificial intelligence* (vol. 2, pp. 1094–1099). Australia, Morgan: Kaufmann Publishers, Inc.

Chan, T. W. (2010). How East Asian classrooms may change over the next 20 years. *Journal of Computer Assisted Learning*, *26*(1), 28–52.

Chan, T. W., & Baskin, A. B. (1988). "*Studying with the prince" the computer as a learning companion*, *In Proceedings of 1988 International Conference on Intelligent Tutoring System (pp. 194–200)*. Canada: Montreal.

Chan, T. W., & Chou, C. Y. (1997). Exploring the design of computer supports for reciprocal tutoring. *International Journal of Artificial Intelligence in Education*, *8*, 1–29.

Chan, T. W., Chung, Y. L., Ho, R. G., Hou, W. J., & Lin, G. L. (1992). Distributed learning companion systems - WEST revisited. In C. Frasson, G. Gauthier & G. McCalla (Eds.), *Proceedings of the 2nd International Conference of Intelligent Tutoring Systems*, *Lecture Notes in Computer Science* (Vol. 608, pp. 643–650), Springer-Verlag.

Chan, T. W., Hue, C. W., Chou, C. Y., & Tzeng, O. J. L. (2001). Four spaces of network learning models. *Computers & Education*, *37*, 141–161.

Chan, T. W., Roschelle, J., Hsi, S., Kinshuk Sharples, M., Brown, T., et al. (2006). One-to-one technology-enhanced learning: an opportunity for global research collaboration. *Research and Practice in Technology Enhanced Learning*, *1*(1), 3–29.

Chang, L. J., Yang, J. C., Deng, Y. C., & Chan, T. W. (2003). EduXs: multilayer educational services platforms. *Computers and Education*, *41*(1), 1–18.

Chen, Z. H. (2014). Facilitating learning preferences and motivation of different ability students for social-competition or self-competition. *Educational Technology & Society*, *17*(1), 283–293.

Chen, Z. H., Chou, C. Y., Deng, Y. C., & Chan, T. W. (2007). Active open learner models as animal companions: motivating children to learn through interacting with my-pet and our-pet. *International Journal of Artificial Intelligence in Education*, *17*(2), 145–167.

Chen, Z. H., Chou, C. Y., Biswas, G., & Chan, T. W. (2012). Substitutive competition: virtual pets as competitive buffers to alleviate possible negative influence on pupils. *British Journal of Educational Technology*, *43*(2), 247–258.

Chen, Z. H., Chao, P., Chao, Y., Hsu, M. C., & Teng, C. H. (2013). Level up, my-pet: the effects of level-up mechanism of educational agents on student learning. *Educational Technology & Society*, *16*(4), 111–121.

Chen, Z. H., Wang, S. C., Lu, H. D., & Chou, C. Y. (2014). Tangible animal companions in traditional Chinese character learning. In *Proceedings of the 22th International Conference on Computer in Education (ICCE 2014)* (pp. 680–682).

Chi, M. T., Siler, S. A., Jeong, H., Yamauchi, T., & Hausmann, R. G. (2001). Learning from human tutoring. *Cognitive Science*, *25*(4), 471–533.

Chou, C. Y., Lin, C. J. & Chan, T. W. (1999). User modeling in simulating learning companions. Proceedings of *AI-ED 99*, *9th International Conference on Artificial Intelligence in Education*, (pp. 277–284), Le Mans, France.

Chou, C. Y., Chan, T. W., & Lin, C. J. (2002a). An approach of implementing general learning companions for problem solving. *IEEE Transactions on Knowledge and Data Engineering*, *14*(6), 1376–1386.

Chou, C. Y., Lin, C. J., & Chan, T. W. (2002b). An approach to developing computational supports for reciprocal tutoring. *Knowledge-Based Systems*, *15*(7), 407–412.

Chou, C. Y., Chan, T. W., & Lin, C. J. (2003). Redefining the learning companion: the past, present, and future of educational agents. *Computers & Education*, *40*(3), 255–269.

Chou, C. Y., Huang, B. H., & Lin, C. J. (2011). Complementary machine intelligence and human intelligence in virtual teaching assistant for tutoring program tracing. *Computers & Education*, *57*(4), 2303–2312.

Csikszentmihalyi, M. (1975). *The psychology of optimal experience*. New York: Harper & Row.

Del Solato, T., & Du Boulay, B. (1995). Implementation of motivational tactics in tutoring systems. *Journal of Artificial Intelligence in Education*, *6*, 337–378.

Delquadri, J., Greenwood, C. R., Whorton, D., Carta, J. J., & Hall, R. V. (1986). Classwide peer tutoring. *Exceptional Children*, *52*, 535–542.

Dillenbourg, P. (2002). Over-scripting CSCL: The risks of blending collaborative learning with instructional design. *Three worlds of CSCL. Can we support CSCL?*, Nederland, Heerlen, 61–91.

Diziol, D., Walker, E., Rummel, N., & Koedinger, K. R. (2010). Using intelligent tutor technology to implement adaptive support for student collaboration. *Educational Psychology Review*, *22*(1), 89–102.

Evans, M. J., & Moore, J. S. (2013). Peer tutoring with the aid of the internet. *British Journal of Educational Technology*, *44*(1), 144–155.

Gagne, R. (1985). *The conditions of learning* (4th ed.,). New York: Holt, Rinehart & Winston.

Greer, J., & Mark, M. (2015). *Evaluation methods for intelligent tutoring systems revisited*. Journal of Artificial Intelligence in Education in press.

Greer, J. E., & McCalla, G. (Eds.) (1994). *Student modelling: The key to individualized knowledge-based instruction*. Heidelberg: Springer.

Greer, J. E., McCalla, G. I., Collins, J., Kumar, V., Meagher, P., & Vassileva, J. (1998). Supporting peer help and collaboration in distributed workplace environments. *International Journal of Artificial Intelligence in Education*, *9*, 159–177.

Hietala, P., & Niemirepo, T. (1998). The competence of learning companion agents. *International Journal of Artificial Intelligence in Education*, *9*, 178–192.

Hiltz, S. R. (1994). *The virtual classroom: Learning without limits Via computer networks*. Intellect Ltd..

Hsu, S. H., Chou, C. Y., Chen, F. C., Wang, Y. K., & Chan, T. W. (2007). An investigation of the differences between robot and virtual learning companions' influences on students' engagement. *Proceedings of the First IEEE International Workshop on Digital Game and Intelligent Toy Enhanced Learning* (*DIGITEL 2007*), 41–48.

Johnson, W. L., Rickel, J. W., & Lester, J. C. (2000). Animated pedagogical agents: face-to-face interaction in interactive learning environments. *International Journal of Artificial Intelligence in Education*, *11*, 47–78.

King, A. (1998). Transactive peer tutoring: distributing cognition and metacognition. *Educational Psychology Review*, *10*(1), 57–74.

King, A., Staffieri, A., & Adelgais, A. (1998). Mutual peer tutoring: effects of structuring tutorial interaction to scaffold peer learning. *Journal of Educational Psychology*, *90*(1), 134.

Kinnebrew, J. S., Biswas, G., Sulcer, B., & Taylor, R. S. (2013). Investigating self-regulated learning in teachable agent environments. In *International Handbook of Metacognition and Learning Technologies* (pp. 451–470). Springer New York.

Kobbe, L., Weinberger, A., Dillenbourg, P., Harrer, A., Hämäläinen, R., Häkkinen, P., et al. (2007). Specifying computer-supported collaboration scripts. *International Journal of Computer-Supported Collaborative Learning*, *2*(2–3), 211–224.

Koedinger, K. R., & Corbett, A. T. (2006). Cognitive tutors: technology bringing learning science to the classroom. In K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences*. Cambridge, MA: Cambridge University Press.

Kollar, I., Fischer, F., & Hesse, F. W. (2006). Collaboration scripts–a conceptual analysis. *Educational Psychology Review*, *18*(2), 159–185.

Leelawong, K., & Biswas, G. (2008). Designing learning by teaching agents: the betty's brain system. *International Journal of Artificial Intelligence in Education*, *18*(3), 181–208.

Magnisalis, I., Demetriadis, S., & Karakostas, A. (2011). Adaptive and intelligent systems for collaborative learning support: a review of the field. *Learning Technologies, IEEE Transactions on*, *4*(1), 5–20.

Mark, M. A., & Greer, J. E. (1993). Evaluation methodologies for intelligent tutoring systems. *Journal of Artificial Intelligence in Education*, *4*, 129–129.

Matsuda, N., Keiser, V., Raizada, R., Tu, A., Stylianides, G., Cohen, W. W., et al. (2010). Learning by teaching simstudent: technical accomplishments and an initial use with students. In *Proceedings of the international conference on intelligent tutoring systems* (pp. 317–326). Springer: Berlin Heidelberg.

Matsuda, N., Yarzebinski, E., Keiser, V., Raizada, R., Stylianides, G. J., & Koedinger, K. R. (2013). Studying the effect of a competitive game show in a learning by teaching environment. *International Journal of Artificial Intelligence in Education*, *23*(1–4), 1–21.

Mayer-Schönberger, V., & Cukier, K. (2014). *Learning with Big data: The future of education*. Houghton Mifflin Harcourt.

Murray, T. (1999). Authoring intelligent tutoring systems: an analysis of the state of the art. *International Journal of Artificial Intelligence in Education*, *10*, 98–129.

Nichols, D. M. (1994). Issues in designing learning by teaching systems. In P. Brusilovsky, S. Dikareva, J. Greer, & V. Petrushin (Eds.) *Proceedings of the East-West International Conference on Computer Technologies in Education* (*EW-ED'94*), (Vol. 1, pp. 176–181), Crimea, Ukraine.

Oh, S., & Woo, W. (2008). ARGarden: augmented edutainment system with a learning companion. *In Transactions on Edutainment I, LNCS*, *5080*, 40–50.

Okita, S. Y. (2014). Learning from the folly of others: learning to self-correct by monitoring the reasoning of virtual characters in a computer-supported mathematics learning environment. *Computers & Education*, *71*, 257–278.

Okita, S. Y., & Schwartz, D. L. (2013). Learning by teaching human pupils and teachable agents: the importance of recursive feedback. *Journal of the Learning Sciences*, *22*(3), 375–412.

Palinscar, A. S., & Brown, A. L. (1984). Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. *Cognition and Instruction*, *1*(2), 117–175.

Palthepu, S., Greer, J., & McCalla, G. (1991). Learning by teaching. In *The proceedings of the international conference on the learning sciences, AACE* (pp. 357–363).

Pea, R. D. (2004). The social and technological dimensions of scaffolding and related theoretical concepts for learning, education, and human activity. *The Journal of the Learning Sciences*, *13*(3), 423–451.

Reif, F., & Scott, L. A. (1999). Teaching scientific thinking skills: students and computers coaching each other. *American Journal of Physics*, *67*(9), 819–831.

Reiser, B. J. (2004). Scaffolding complex learning: the mechanisms of structuring and problematizing student work. *Journal of the Learning Sciences*, *13*(3), 273–304.

Renkl, A. (1995). Learning for later teaching: an exploration of mediational links between teaching expectancy and learning results. *Learning and Instruction*, *5*(1), 21–36.

Rittschof, K. A., & Griffin, B. W. (2001). Reciprocal peer tutoring: Re-examining the value of a co-operative learning technique to college students and instructors. *Educational Psychology*, *21*(3), 313–331.

Roscoe, R. D., & Chi, M. T. (2007). Understanding tutor learning: knowledge-building and knowledge-telling in peer tutors' explanations and questions. *Review of Educational Research*, *77*(4), 534–574.

Roscoe, R. D., & Chi, M. T. (2008). Tutor learning: the role of explaining and responding to questions. *Instructional Science*, *36*(4), 321–350.

Shindo, Y. & Matsuda, H. (2001). Design and implementation of scenario language for cyber teaching assistant, *Proceedings of Enhancement of Quality Learning Through Information & Communication Technology*, 2, 643–650.

Sweller, J., Ayres, P., & Kalyuga, S. (2011). *Cognitive load theory*. Springer New York.

Topping, K. J. (1996). The effectiveness of peer tutoring in further and higher education: a typology and review of the literature. *Higher Education*, *32*(3), 321–345.

Ur, S., & VanLehn, K. (1995). Steps: a simulated, tutorable physics student! *Journal of Artificial Intelligence in Education*, *6*, 405–437.

Uresti, J. A. R., & Boulay, B. D. (2004). Expertise, motivation and teaching in learning companion systems. *International Journal of Artificial Intelligence in Education*, *14*(2), 193–231.

Vanlehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, *46*(4), 197–221.

Vanlehn, K., Ohlsson, S., & Nason, R. (1994). Applications of simulated students: an exploration. *Journal of Artificial Intelligence in Education*, *5*, 135–135.

Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes*. Harvard University Press.

Walker, E., Rummel, N., & Koedinger, K. (2011). Designing automated adaptive support to improve student helping behaviors in a peer tutoring activity. *International Journal of Computer Supported Collaborative Learning*, *6*(2), 279–306.

Walker, E., Rummel, N., & Koedinger, K. (2014). Adaptive intelligent support to improve peer tutoring in algebra. *International Journal of Artificial Intelligence in Education*, *24*(1), 33–61.

Wong, W. K., Chan, T. W., Chou, C. Y., Heh, J. S., & Tung, S. H. (2003). Reciprocal tutoring using cognitive tools. *Journal of Computer Assisted Learning*, *19*(4), 418–428.

Woolf, B. P. (2008). *Building intelligent interactive tutors: student-centered strategies for revolutionizing e-learning*. Boston: Morgan Kaufmann Publishers.