Reflective Analysis on Teaching and Learning Computational Thinking Online

Li Xu
University of Arizona South
1140 North Colombo Avenue
Sierra Vista, AZ 85635
lxu@email.arizona.edu

Abstract: In 2006, Wing proposed Computational Thinking (CT) as a fundamental skill to everyone (Wing, 2006). To promote CT and address the limitations to teach CT in a traditional face-to-face classroom, the Computer Science (CS) program at the University of Arizona South has offered an online class to teach and learn CT since 2013. In this paper, based on the collected teaching/learning practice data, we evaluate on the online course reflectively by analyzing course elements including student, instructor, course content, and the relationships between them as well as the course as a whole by putting all elements together. The reflective analysis shows that the online course is capable of supporting students to learn CT effectively and actively. The analysis also provides informed data to evolve the online course development so as to continually promote CT to students.

Introduction

Wing (2006) proposed Computational Thinking (CT) as a fundamental skill to everyone. In a later paper she published (Wing, 2011), she defined CT as "the thought process involved in formulating problems and their solutions so that the solutions are represented in a form that can be carried out by an information-processing agent." In addition, Hu (2011) emphasized that people can "gain different kinds of critical thinking capabilities" by being capable of the variety of means in CT and CT "is present not only because of the nature of computation but also because of the way how people think critically." Essentially, CT uses a set of concepts drawn from Computer Science (CS) to solve problems, understand human behaviors, and design systems. To help students to understand and practice CT, it is necessary for them to develop a foundation of CT concepts and techniques, and practice how to synthesize the concepts and techniques in critical thinking, problem solving, and system design.

The Computer Science (CS) program at the University of Arizona South has committed to support students to learn how to think computationally and critically. Since Wing proposed CT in 2006, how best to help students to understand and master CT has remained an open question. However, research done on thinking processes convinced that thinking skills were most effectively taught when teaching them directly and deliberately (Bono, 1992). Guzdial (2008) also pointed out "the metaphors and ways of thinking about computing must be explicitly taught." Moreover, teaching CT in a traditional face-to-face classroom had various barriers and challenges including limited course access and constrained student-instructor communication approaches and contact time in classroom. With the support of online learning technology, since 2013 the CS program has started to offer an online class to teach and learn CT to promote CT and address limitations to teach CT in a traditional face-to-face course. The CS program developed and delivered the online CT course by employing the Desire2Learn (D2L) system, which is a Web-based course management system offering teaching and learning tools for course development, delivery and management.

In this paper, we intend to evaluate the online teaching and learning CT reflectively by analyzing course elements including student, instructor, course content, and the relationships between them as well as the course as a whole by putting the elements all together. Section Elements of Online Teaching and Learning CT presents the reflective analysis over the isolated course elements including student, course content, instructor as well as relationships between them; and Section Putting All Together as a Whole describes the online learning by synthesizing the various course elements. Finally, Section Conclusions draws conclusions of the study.

Elements of Online Teaching and Learning CT

We discuss the online teaching and learning on CT by addressing the course elements including student, course content, instructor, and the relationships between them.

Student

Students enrolled in the course since 2013 were across disciplines. In 2013, 12 students enrolled including seven in CS, one in Mathematics, two in Applied Sciences, one in Management of Information Systems, and another one from School of Information. All students were majored in computing related programs. One student, who

was in CS, didn't pass the course. In 2014, 17 students enrolled including five in CS, six in Applied Sciences, and another six in non-computing fields. Among them three students including one Applied Sciences major and two non-computing majors, failed the course. In 2015, 15 students enrolled including seven in Applied Sciences, one in Information Science, four in CS, one in Mathematics, one double major in both CS and Mathematics, and another one in a non-computing major. One Applied Sciences major failed the course. In 2016, 25 students enrolled including two in CS, 21 in Applied Sciences, and another two in non-computing majors. Only one student, who majored in CS, failed the course.

Students enrolled in the online CT course engaged in actively planning and conducting learning activities, and persistently monitoring and reflecting their own learning. Simultaneously, the leaners formed a learning community where students actively supported each other. Frequently, students raised questions to challenge both their peers and the instructor. They also took efforts to answer questions and share their learning approaches and experiences. Student evaluations and reflections reported that the online course needed "constant information passing" from students and provided intensive and valuable interactions with fellow classmates as well as the instructor.

There were observed differences among the students regarding the online learning CT. Probably the most obvious difference between CS and non-CS students was their capabilities to do programming in Python and bash, which were two programming languages used in the course. Even if the CS majors had little or none Python/bash programming experience, they could quickly started to do programming to solve a few well-structured problems. In contrast, the other group of students was able to read Python code or bash scripts based on their learning on elements of the programming languages but writing code turned out a great challenge to most of them.

Despite the challenges, student comments and feedbacks showed that most students got inspired while learning CT and they believed that the course set a good groundwork for them to think computationally. In 2013, 2015, and 2016 about 80% of the enrolled students were able to learn effectively and concluded their learning with a grade either A or B. In 2014, however only 47% (eight) of the enrolled 17 students performed satisfying or excellent learning performance.

Course Content

The course content of the online class centers on CT definitions. CT captures how computer scientists think and its definitions typically draw metaphors and concepts from CS. The essential concepts identified by the International Society for Technology in Education (ISTE, 2014) include data collection, data analysis, data representation, problem decomposition, abstraction, algorithms, automation, parallelization, and simulation. Since the CS program started to offer the online CT course, the course content typically has covered all concepts except parallelization. Below is the list of the online CT course topics:

- Introduction to computational thinking
- Algorithm
- Data abstraction
- Programming languages
- Python programming
- Shell programming in UNIX
- Thinking Object Oriented
- Responsibility Driven Design
- Theory of computation
- Research/programming project

The last topic was also the designated final assessment on student's learning on CT. While working on the project, students chose a problem of interest to address or a system to design using computational thinking and doing. By the end of each spring term during which the course was offered, students presented and shared their project results in project posters online.

In addition to the CT topics, the online course content also composed of weekly learning schedules, learning assessment activities, students' performance and reflection data, instructor's annotation notes on the course materials, course formal evaluation surveys, and instructor-student interaction data. At the beginning of each week, the course site's announcement page would display a list of learning activities including reading notes and book chapter sections, working on quiz and assignment questions, writing one or more reflective discussion posts, and conducting project research. Students contributed to the course content by making input on the course design and delivery informally at online discussion forums as well as by the formal course delivery surveys. The course content

also included student works, including reflective notes, raised questions and question discussions, quizzes, assignments, project proposals, progress reports, and final project posters. Every student work typically had instructor's grading and/or feedback data associated. Instructor also continually wrote complementary notes on the course materials to reflect and revise the course content. Since the CS program initiated the online CT course in 2013, the course content has evolved over the long time span.

Instructor

The role of the instructor during the online teaching and learning CT processes was supporting students and facilitating online learning. It happened that the same instructor delivered the course during the past four spring terms. However, no teaching or learning repeated exactly same during the terms and in practice every delivery demanded specific responses by the instructor. As the course deliveries progressed the instructor utilized the current and previous course practice data to reflect her teaching persistently and further revise the course.

The biggest concern while delivering the course was the lack of physical presence of the instructor, which is critical when teaching a traditional face-to-face class. Based on the student learning performance data and student weekly reflections, the physical presence of an instructor in a traditional class seemed replaceable for most students enrolled in the online CT course. The Teacher-Course-Evaluation (TCE) data collected at the end of each term in general were positive regarding teaching effectiveness and instructor comparison. In 2013, 2015, and 2016 both teaching effectiveness and instructor comparisons were above the comparison group data, which mostly composed of TCE data from face-to-face CS courses. In 2014, both scores were also comparable. Note that the students enrolled in 2014 were the most diverse group and the term was also two weeks shorter than the others.

Student---Content

To effectively engage students participating in learning CT online, the online course development had its focus on CT concepts and techniques considering the student populations and needs during the learning processes. The course content was built based on the preference matrix, which Kaplan & Kaplan (1983) developed as a theory of humans as information processors. When applying the preference matrix theory as a pedagogical tool to design CS courses, Paxton (2006) described that the two key dimensions that lead to a preferred learning environment included (1) whether a student can make sense of the learning environment and (2) whether a student can be involved through learning and/or exploration.

The online CT course promoted "make sense" by including: course content clearly stated student learning objectives on mastering and practicing CT; each course module included learning goals expected to be accomplished by students; and weekly news announcements to students specified an action plan constituting of learning materials, expected learning goals, and assessment activities. To foster student involvement in the online learning involvement, the online CT learning modules had specific features, including: students conducted regular online discussions addressing essential questions on the weekly learning CT topic and persistently reflected on their own learning; students worked on quizzes to assess their recognition and understanding on CT concepts and techniques; students worked on programming/writing assignments to analyze problems, integrate and resolve solutions; students solved problems or designed systems in a final project to conclude their course learning. Especially, students had almost complete controlling on the learning process over the final project, in which they chose what/when/how they wanted to do to conclude their learning on CT.

The online teaching and learning challenged and changed students' perception on CT. In their course evaluations and student reflections, students reported that by learning the course content they realized that CT was not programming but programming played a critical tool for them to use to automate problem solutions. In addition, students also mentioned that they were challenged by the amount they learned about CT and the online course content was engaging and allowed a philosophical approach for them to develop CT skill.

Content---Instructor

Each course delivery composed of design and implementation phases by the instructor. To develop the online CT course content, the instructor modeled and implemented multiple instructional strategies including asking questions, scaffolding problem solving, eliciting good questions and explanations, guiding knowledge transferring from term recognition to knowledge application and creation, and coaching project managements of final projects. When the course was initiated in 2013, the instructor started the course design phase about one year before the first class day. The instructor determined the course topics as well as learning objectives, student assessment activities. In the later three spring terms, during the design phase, the instructor revised the modules based on student feedback comments, learning assessment data, online discussions, and instructor's reflective notes. During every implementation phase, while the instructor progressed her teaching CT online, she dynamically created course

content including feedbacks to student works, weekly learning schedules, relevant online discussion questions, course announcements and reflective notes to address specific student requests and needs.

The instructional scaffolding proved especially critical for the CT class to succeed due to how student learning was initiated and progressed. During the design and implementation phases, the instructor considered instructional scaffolding as an essential element of effective teaching, to set up successive levels of learning modules and activities. When designing the course content, the instructor intentionally related online modules to each other, related theoretical ideas to practice CT, and organized and structured contents into a coherent whole that served the desired learning objectives. Especially, the instructor used scaffolding to bridge learning gaps —i.e., the difference between what students had learned and what they were expected to know and be able to do at a certain time instant. For example, after the instructor received student messages regarding how to initiate shell programming by connecting to the University's computer system, the instructor created stepwise examples to demonstrate how to connect to the U system, login using the U account, run UNIX commands, and write and execute bash scripts.

Instructor---Student

There were four communication channels for interactions between the instructor and her students. First, online discussion has been one critical means for effective instructor-student communications. Every week the instructor created relevant weekly online discussion questions. During the week, students regularly reflected their learning and raised questions on the CT topics. Meanwhile, the instructor also closely monitored the input students made to identify problems. If necessary, the instructor would develop new materials online to facilitate online learning. At the online discussion forums, the instructor sometimes provided public posts to the whole class, if she believed the information would be beneficial to all students, as well as private feedbacks to engage individual student learning. Second, email exchanges turned out to be the most effective way to discuss programming problems and debugging solutions. Third, the instructor always provided prompt, detailed feedback notes to each student regarding how he or she had done on a learning activity such as a quiz or assignment. Lastly, virtual meeting tools such as Skype and Adobe Connect embedded in D2L course sites allowed conference meetings between students and the instructor. The last approach however was not used often during the past four course deliveries. However, the online meeting tools were able to alleviate learning anxieties of a small number of students who preferred or even demanded the instructors' physical presence.

One benefit gained from teaching CT online was that the D2L course-development platform allowed the instructor to support students to practice CT using a systematic structure with preparations, actions, reflections, and revisions. As student learning was progressing, students' participation and performance data was incrementally collected by the system for the instructor to analyze student learning and reflect on teaching effectiveness. The learning and teaching processes provided statistical data for the instructor to identify and address student needs.

In the previous subsection, instruction scaffolding was presented as an instructional strategy for the instructor to improve course content. Another goal of instructional scaffolding, which was implemented dynamically during communications between the instructor and students, was to reduce the negative emotions and self-perceptions that students might experience when they got frustrated, intimidated, or discouraged when attempting a difficult task, such as programming or conducting research on the final project.

Putting All Together as a Whole

After analyzing student, instructor, course content, and relationships between them, we now synthesize all elements together and have an in-depth look at the course as a whole.

Student-Centered Teaching and Learning

The online CT course offered a student-centered approach to teach and learn CT online. The course was divided into a sequence of small-scaled learning processes driven by a list of weekly learning activities. The instructor carefully crafted the course materials including learning activities using the preference matrix to inspire online learning and foster student participation and involvement. Students had expectations on where to find course content and how to get support from the instructor during the learning processes. As the course evolved during the past four spring terms, the instructor was free from giving lecturing as her major teaching activities. Instead, the instructor's primary role was supporting students' online learning activities.

Note that the course deliveries didn't require students to perform complete self-directed learning. That is, students didn't have the complete learner autonomy to control the whole learning process on CT. Instead, the course deliveries emphasized self-directed learning within a short time period—typically one week. Students could learn when they wanted at their own pace within a week. However, the course progressed and directed by the instructor during the course-delivery term so that students could follow a timeline to actively master the concepts and

techniques that form the foundation of CT. Thus, the course delivery model required students to be proactive but also allowed the instructor to get involved in and closely coach the online learning. Based on comments and reflections by students, the instructor's facilitations and coaching, such as designing the relevant course materials and intensively communicating with students including being present at discussion forums and providing prompt feedbacks to student emails and submissions, certainly made the course easy to progress.

The teaching and online style used in the online CT course seemed matching most of enrolled students' degree of self-direction. But it didn't fit everyone. In 2013 and 2016, the only two students who failed the course majored in CS, which suggested the course might not be inspiring enough to engage CS majors to study CT. Some students commented suggestions to make the course assignments a little harder for people who had some programming experiences. And yet others pointed out that the topics were "intensive and demanding" and homework assignments should be geared to beginners. The teaching practice also indicated that students, especially the ones in non-computing majors, needed longer time and a slower learning pace to master the CT concepts and techniques and demanded intensive, meaningful instructor-student interactions. In Spring 2014, the 14-week course delivery brought up more anxieties among the non-computing majors due to the shortened term to complete the course. In addition, students showed difficulty to synthesize their learning when programming and conducting research on the final project. Despite the challenges, students reported that the online CT course not only supported them to build ground works on CT but also increased their learning skills.

Computational Thinking in Problem Solving

In every course delivery, students actively participated in CT learning by solving problems presented in course assignments and the final project. The course intentionally taught the means and tools in CT so that students were able to practice using CT in problem solving. During the teaching and learning processes, the instructor presented the assignment problems using well-structured problem statements in assignment specifications. Usually students were able to understand the CT concepts rapidly but sometimes had difficulty to apply them together. Especially the non-CS majors often needed more support from the instructor on how to come up algorithms to approach the problems and how to program their algorithms in Python and bash.

Although students reported that practicing CT in the assignments and the final project increased their professional skills, it was also obvious that students had difficulty with synthesizing the CT topics into their final project. By analyzing the learning performance on project proposals, progress reports, and project posters, it was obvious that students' prior research experience as well as problem-solving and critical thinking skills impacted on their applications of CT skills to solve problems and design systems. In the past four course deliveries, a significant portion of the enrolled students including CS majors were not able to identify and structure the problems they wanted to solve. Students also reported that they needed to constantly adjust their project scopes and the project took efforts and time to mature.

Moreover, the final project in the online CT course allowed students to decide what/when they wanted to do except that students needed to propose their project, report their project progress, and publish the research result. Taking an active role to manage the final project turned out another great challenge confronted by students. Fortunately, the teaching and learning practice data accumulated online has helped the instructor identify the needs to provide more learning structures, project examples, and other scaffolding problem-solving materials online to support and motivate students. In the most recent two terms, the instructor provided project poster examples and urged students to share their learning approaches and project experiences at online discussion forums. She also gave individuals detailed study guide to ease learning anxieties. Note that due to the diversity in student groups, very frequently the instructor was not an expert for some applications approached by students. Still, the instructor actively collaborated with students during the learning by providing feedbacks on project management, searching resources, and acquiring new knowledge like how a computer scientist works in various fields. Frequently, fellow classmates also raised clarification and probing questions, and made comments to support peers to continually carry on the project learning tasks.

Conclusions

In this paper we have made an in-depth reflection on an online CT course development by analyzing student, course content, instructor as well as relationships between them. After evaluating the various course elements, we then synthesized them altogether and reflected over the whole course. While analyzing the course as a while, we emphasized the student-centered teaching and learning CT as well as student learning on CT in problem solving. Throughout the reflective analysis on the elements and the online course as a whole, we conclude that the online CT Teaching and learning supported students to form a foundation of CT concepts and techniques, learn how to learn

actively, and be competent in computational thinking and problem solving. While analyzing the course reflectively, we become informed to continually improve the course development in order to promote CT in future.

References

Bono, E. D. (1992). *Six thinking hats for schools*. Hawker Brownlow.

Guzdial, M. (2008). Education: Paving the way for computational thinking. *Communications of the ACM*, 51(8), 25-27.

Hu, C. (2011). Computational thinking: what it might mean and what we might do about it. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education: ITiCSE '11*, 223-227. New York, NY, USA: ACM.

ISTE. (2014). *http://www.iste.org.*

Kaplan, S., & Kaplan, R. (1983). *Cognition and environment: Functioning in an uncertain world*. Ann Arbor, MI: Ulrich's.

Paxton, J. (2006). The preference matrix as a course design tool. In *Proceedings of the 6th baltic sea conference on computing education research: Koli calling 2006*, 124-127. New York, NY, USA: ACM.

Wing, J. (2006). Computational Thinking. *Communications of the ACM*, 49(3), 33-35.

Wing, J. (2011). Computational thinking: What and Why.
http://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why.