International Journal of Child-Computer Interaction 2 (2014) 26–33

Contents lists available at ScienceDirect

International Journal of Child-Computer Interaction

journal homepage: www.elsevier.com/locate/ijcci

CTArcade: Computational thinking with games in school age children

Tak Yeon Lee [a,b,*], Matthew Louis Mauriello [a,b], June Ahn [a,c,d], Benjamin B. Bederson [a,b,c]

[a] *Human-Computer Interaction Lab, University of Maryland, College Park, MD 20742, USA*

[b] *Computer Science Department, University of Maryland, College Park, MD 20742, USA*

[c] *iSchool, University of Maryland, College Park, MD 20742, USA*

[d] *College of Education, University of Maryland, College Park, MD 20742, USA*

ARTICLE INFO

Article history:
Received 22 July 2013
Received in revised form
14 June 2014
Accepted 28 June 2014

Keywords:
Computational thinking
Learning
Games
Human factors
Design

ABSTRACT

We believe that children as young as ten can directly benefit from opportunities to engage in computational thinking. One approach to provide these opportunities is to focus on social game play. Understanding game play is common across a range of media and ages. Children can begin by solving puzzles on paper, continue on game boards, and ultimately complete their solutions on computers. Through this process, learners can be guided through increasingly complex algorithmic thinking activities that are built from their tacit knowledge and excitement about game play. This paper describes our approach to teaching computational thinking skills without traditional programming—but instead by building on children's existing game playing interest and skills. We built a system called CTArcade, with an initial game (Tic-Tac-Toe), which we evaluated with 18 children aged 10–15. The study shows that our particular approach helped young children to better articulate algorithmic thinking patterns, which were tacitly present when they played naturally on paper, but not explicitly apparent to them until they used the CTArcade interface.

1. Introduction

Computational thinking (CT) is a set of thinking patterns that includes understanding problems with appropriate representation, reasoning at multiple levels of abstraction, and developing automated solutions [1]. CT has become a fundamental skill and should be cultivated by everyone, not just computer scientists [2,3,1]. Researchers in other disciplines discover breakthroughs by using the same CT skills valued by computer scientists. Many everyday activities using information technology can be done more efficiently when the person performing them has CT skills. However, one challenge in fostering CT is the lack of opportunities to improve one's CT skills.

Building one's own game has been a popular task in educational programming environments. Papert [4] coined the term "constructionism" and observed that learning is most effective when learners construct a meaningful product. Researchers have contributed foundational work to create visual programming environments that allow children to create their own games by manipulating graphical (and tangible) blocks rather than relying on textual syntax. For instance, AgentSheets [5] helped students recognize patterns of CT while designing games, and allowed them to apply the patterns in science simulations. Storytelling Alice [6] taught basic programming constructs to middle school girls by giving them tools for creating interactive animations. HANDS [7] showed that the usability of programming environments is crucial for children to learn CT skills. Although game creation is an effective way to learn basic CT skills, building a playable game from scratch is still challenging for novice programmers. This level of challenge means that only strongly-motivated children can build playable games that involve sophisticated CT skills even though these skills are readily available in the natural game play exhibited by their peers.

Learning scientists and educational experts have acknowledged that children playing various genres of games have shown a variety of sophisticated CT patterns that emerge in their game play. For example, racing games encourage children to internalize one-to-one mapping of game action to kinematic concepts [8,9]. Different game activities are associated with various characteristics of CT skills [10]. However, a formative study [11] found out that simply playing games does not help children articulate CT skills unless the game is equipped with carefully designed support for articulating CT skills.

* Corresponding author at: Human-Computer Interaction Lab, University of Maryland, College Park, MD 20742, USA. Tel.: +1 202 330 2506.

E-mail addresses: tylee@umd.edu (T.Y. Lee), mattm@cs.umd.edu (M.L. Mauriello), juneahn@umd.edu (J. Ahn), bederson@cs.umd.edu (B.B. Bederson).

http://dx.doi.org/10.1016/j.ijcci.2014.06.003
2212-8689/

Another approach is to combine the advantages of natural game play and educational programming environments by using a programmable game character strategy as done by a small number of simulation game platforms. One example of this approach is AlgoArena [12], which enabled students to program their own wrestler characters in order to win matches against other wrestlers. RoboCode [13] is another game environment where each player develops AI using the Java language to manoeuver a robot tank in order to search out and destroy other tanks. This problem-driven learning has several advantages. First, it allows students to focus on advanced algorithmic logic without requiring them to build basic functionality of the games. Second, when the children develop the AI, they transform tacit knowledge about game play into formal logic, which is a common way of learning CT skills. Third, social interaction on the tournament servers provides extra motivation. Unfortunately, currently available educational games, requiring professional programming skills, are not accessible to novice programmers.

In this paper, we describe CTArcade—an educational game environment that enables children to articulate CT-related thinking patterns while playing games. We developed Tic-Tac-Toe as the first game to utilize the CTArcade environment. We report a "think aloud" user study that observes 18 children playing the CTArcade version of Tic-Tac-Toe and the original version on paper. We compared the gameplay of these environments through (1) quantitative analysis of code counts for instances of different kinds of CT skills, and (2) descriptive examples of the CT skills utilized by the children. The contributions of this paper are threefold: (1) we articulate an approach to developing CT skills through gameplay; (2) we provide a demonstration of the CTArcade system using a Tic-Tac-Toe game that implements a scaffolded learning structure; (3) and an exploratory user study with 18 children that shows evidence that scaffolded gameplay in CTArcade helped them articulate more algorithmic thinking skills compared to playing naturally on paper.

2. Design process

The CTArcade design was strongly motivated by a formative study [11] that examined how young children (aged 7–11) utilized computational thinking skills as they play the paper-based game, Connect Four. We observed that children naturally used complex CT skills (e.g. recognizing the winning and losing conditions of the board). However, they expressed difficulty developing algorithmic representation of their gameplay strategy. The study provided design considerations about how to use games to teach CT and we directly applied them in the design of CTArcade. We then designed the initial set of Tic-Tac-Toe strategy and AI agents based on the strategy created during the formative study. After building the first working prototype, we iteratively conducted pilot tests and redesigned it with children. We had three goals in the design of CTArcade:

- *Integrate tacit knowledge within the game interface*. Decomposing innate thinking into abstract representations is difficult for children. Providing a natural way of formalizing tacit knowledge during game play has a critical place to help learners explicitly link their game actions to CT. The CTArcade Trainer employs a mixed-initiative approach [14] that infers and suggests potential strategic rules corresponding to the most recent move made by the learner. Using this method allows the learner to either pick the right rules or directly specify a new rule in the creator mode.
- *Moving from concrete to abstract computational thinking*. In order to help learners transform their tacit knowledge about game play into formal logic, CTArcade guides them to create rules for specific game situations first and then to generalize them later. Such progress requires the process of *concreteness fading* [15], CTArcade provides features that gradually reduce the salience of the specific game situation to an abstract strategy.

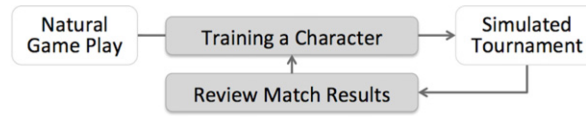

Fig. 1. CTArcade framework focuses on creating a learning loop for articulating CT skills, where children can train their own virtual characters and review tournament results.

- *Reduce cognitive load and split attention*. Having separate elements of information (e.g. a game board and a rule sheet) to pay attention to would significantly raise the cognitive workload for learners. We designed the CTArcade platform to minimize the split attention effect.

We designed CTArcade to scaffold the learning of CT skills via natural game play. Specific elements of the user interface (UI) and environment were designed to help children articulate CT thinking patterns:

- *Templates* for game strategy help learners conceptualize their game play. For example, the game strategy of Tic-Tac-Toe can be described as a prioritized list of rules, where each rule is defined by a pre/post-condition of the game board. Templates not only support different CT skills to be learned but also prevent syntactic/semantic errors. Different games may have different templates (see Future Work section for examples).
- *Learners teach virtual characters* instead of programming. We believe "teaching" provides learners with a strong motivation to conceptualize their own strategy.
- *Suggestions* are provided that infer algorithmic patterns related to the learner's game play.
- *Tournament feedback* helps learners to evaluate their own strategies and learn from other people, which can lead them to iterative reflections and fine-tuning of one's gameplay logic.
- *Visual analytic tools* enable learners to find patterns from a number of tournament results.

3. CTArcade: designing to teach computational thinking

CTArcade is a web-based educational gaming environment that extends simple games with scaffolded learning activities (see Fig. 1). In CTArcade children start by naturally playing Tic-Tac-Toe against their own game characters, which know very little about game play strategies. While playing the game, a child will conceptualize his/her game play into abstract algorithmic rules that the character will then follow. The programmed characters will then join the tournament server and play against other characters. The children can review results, recognize patterns of losing matches with four different visualizations, and make refinements to their strategy in iterative manner. The learning loop highlights how the CTArcade framework complements and bridges the gap between natural game play and educational programming environments. When children play games, they naturally experience situations that require CT skills. However, these skills are applied rapidly in a natural context and children do not inherently conceptualize their play as abstract logic. On the other hand, visual programming environments help to lower the technical barrier of programming, but do not necessarily take advantage of natural activities requiring CT skills. Our goal with CTArcade is to complement these approaches by helping young children internalize CT skills while converting their natural game play into a formal strategy via a scaffolded model. The following sections illustrate components of CTArcade learning loop in detail.