CTArcade: Learning Computational Thinking While Training AI Through Game Play

Tak Yeon Lee, Matthew Louis Mauriello and Awalin Sopan
Department of Computer Science
Human-Computer Interaction Lab
College Park, MD 20742 USA
tylee, mattm, awalin@umd.edu

ABSTRACT
In this paper we describe a web application framework for developing games that teach users about computational thinking and the preliminary research that led to its creation. CTArcade was developed to be the foundation for a library of several challenging games that engage the users’ innate computational thinking (CT) skills. These skills are employed when the users are asked to teach an artificial intelligence (AI) agent the rules and strategies of the game, Tic Tac Toe. In addition to providing a training module, users can direct their AIs to play individual matches against other AIs and participate in a regular server-wide tournament. These additional activities are designed to provide the users with motivation to improve their characters, which are anticipated to drive the improvement of fundamental skills. A brief pilot test was conducted using volunteers from the University of Maryland. The participants helped to assess the usability of CTArcade’s current implementation as an educational tool.

Categories and Subject Descriptors
H.5.m [Information interfaces and presentation]: Miscellaneous

General Terms
Design, Human Factors, Usability

Keywords
Computational Thinking, Learning, Programming by Demonstration, and Games

1. INTRODUCTION
In today’s age where information technology is so prevalent, it has become extremely important for people to be fluent in its use. Everyone can benefit from having the skills to find, understand, and add to the body of human information through the tools that we create. However, the educational system in the United States is lacking in its ability to teach computational thinking (CT) skills to its students. When students enter college and enroll in courses, fewer and fewer go into Science, Technology, Engineering, and Mathematics (STEM) courses with major gaps persisting between students across major demographic indicators such as gender and ethnicity[2]. Of those students that do enroll into STEM, many become disinterested and end up dropping out. The Computer Science (CS) community has become increasingly concerned with this fact because there is such a high drop out rate among first year students. The reason for this high rate is thought to be due to the extreme difficulty of programming and students’ lack of proper background in this field from their previous education[9]. There is an increased interest to develop new educational strategies that promote CS education for a broad range of students. Some of these initiatives have involved making online course work more available to the general public (i.e. MIT’s OpenCourseWare initiative[1], Stanford online Artificial Intelligence Course[2]), while other initiatives have involved the development of specialized tools.

From a pedagogical standpoint, CS education is evolving to stress CT skills earlier in a student’s education. When students begin CS education their courses should focus on making problems more engaging rather than on rote computer programming tasks. Wing[15] asserts that CT is fundamentally about learning how to solve problems through skills such as the abstraction of problems, the definition of appropriate representations, and the development of solutions. Google also promotes a set of four CT skills that it feels are fundamental to CS: decomposition of problems, pattern recognition, abstraction, and algorithmic thinking[4]. All of these skills are applicable to more than just CS education and would benefit students across many fields of studies.

In this paper, we present a formative study and a web application platform. The formative study examines how young children utilized CT skills as they created game-play strategies for the popular game, Connect Four. It also highlights design considerations for developing interfaces to help children to decompose their game play into logical strategies while conceptualizing games through algorithmic thinking. As a result of this study a web application platform, CTArcade, has been developed. The goal of CTArcade is to provide an online library of games that develop the users’ CT skills by allowing them to train an AI to play the deployed

1http://ocw.mit.edu/index.htm
2https://www.ai-class.com/
games.

2. RELATED WORK

There have been a number of strategies aimed at making CT more engaging and easy to grasp. Initial efforts focused on using programming such as Logo to allow students to easily program simulations, robots, and other projects[11]. Subsequent attempts to reduce the complexity of modern integrated development environments (IDE), while also making programming a visual experience, were highly successful. Such examples include Alice and Scratch which focus on teaching the primitives of programming languages such as loops, conditional statements, and event handling (e.g., mouse click) with attachable “drag and drop” visual components.

Alice and its derivative, Storytelling Alice, is an authoring tool to develop short interactive 3D digital movies. One of the factors that make Alice successful as an authoring tool is its graphical user interface, which makes programming easy for non-programmers. The interface allows users to program using intuitive actions such as the ability to drag and drop methods into their work. Features like these also have the added benefit of reducing user frustration with common errors involving programming syntax. This benefit makes it easier for users to develop their skills while keeping them interested in the activity of programming. Storytelling Alice adds storytelling support to the interface by repackaging methods like move, turn, and orient and converting them into a language that is more natural to storytelling (e.g., walk off-screen, turn away from, and fall-down). While Alice was effective at teaching users CT skills, Storytelling Alice was found to increase the interest of young women in programming[7].

Alice and its derivative, Storytelling Alice, is an authoring tool to develop short interactive 3D digital movies. One of the factors that make Alice successful as an authoring tool is its graphical user interface, which makes programming easy for non-programmers. The interface allows users to program using intuitive actions such as the ability to drag and drop methods into their work. Features like these also have the added benefit of reducing user frustration with common errors involving programming syntax. This benefit makes it easier for users to develop their skills while keeping them interested in the activity of programming. Storytelling Alice adds storytelling support to the interface by repackaging methods like move, turn, and orient and converting them into a language that is more natural to storytelling (e.g., walk off-screen, turn away from, and fall-down). While Alice was effective at teaching users CT skills, Storytelling Alice was found to increase the interest of young women in programming[12].

Another similar application, Scratch, has translated programming elements into bricks that can be tinkered with in a manner similar to Lego bricks. The program was developed by MIT following a successful partnership with Lego that resulted in the production of Lego Mindstorms. The motivation behind this project is similar to CT Arcade in that it aims to give young students, referred to as “digital natives”, with a high fluency in digital technologies the means to not only consume media but become producers of games, animations, and simulations. Although Scratch is similar to Alice in many ways, it also allows users to collaborate extensively on ideas and projects. This ability to collaborate has spawned an active, global, and dynamic community capable of creating complex projects of various types.

While these visual programming environments make it significantly easier to understand and build programs, the range of problems that can be solved by user is somewhat limited to what is specified by the IDEs. Another approach is to make traditional syntax learning less complex by building smarter programming consoles. Codecademy3, an interactive online tutorial for learning Javascript, takes a more direct approach to train a potential programmer by guiding the user through step-by-step instruction. Codecademy is more or less designed as a training tool for the job market; however, the interface and guidance the system provides makes it just as applicable to traditional classroom activities in CS education.

We observe a commonality in these projects. From a learning perspective, they follow a similar pedagogical trajectory. Users must first learn the primitive syntax of the language as a means to then solve the problems they are interested in.

In our work, we are exploring an alternative cognitive trajectory for CS learners. Instead of moving from syntax to problem solving, recent thought in CT suggests that one could start with natural human pursuits and then connect CT to these activities in situ. For example, Berland and Lee[1] show that CT skills such as debugging and distributed computation occur naturally when individuals play a collaborative board games. Furthermore, other studies of learning show that human beings engage in natural CT during everyday activities such as when African-American boys play dominoes or young children play racing games such as Mario Kart[6, 8].

This distinction between learning processes is subtle, but important. If one would like to embed CT within natural activities, such as game playing, there are different design challenges. In particular, a significant problem concerns the design of tools that help individuals become cognizant of their intrinsic CT (i.e. game playing strategy) - and then translating their natural thinking patterns to a related computational syntax or vocabulary. Researchers find that individuals can frequently describe computational ideas such as if-then logic, looping, and iteration in their everyday lives; however, they have extreme trouble translating this tacit knowledge into a generalized CS vocabulary or programming conventions[5, 10].

Social factors have been proven to be important in education, however, not many CT educational systems involve social interaction as a main characteristic. As previously mentioned, Scratch is one example of a successful social environment where children and adult users share their code and improve together. Another example of collaborative learning is the success of the FIRST LEGO League, which gets students involved in team learning through the Lego Mindstorms
The study was conducted with a team of children, ages 7 - 11, at the University of Maryland’s Human Computer Interaction Lab (HCIL). The HCIL works with a dedicated team of children co-designers (Kidsteam) using methods of Cooperative Inquiry. Using this method, children and researchers design new technologies together. The children were guided through several activities that modeled how to decompose Connect Four into game playing rules and strategies. The children were then shown how to combine these rules and strategies into an algorithm, or recipe, to play against a computer. The children were also shown how to create one board state and then conceptualize another. For example, one participant created a rule where if three pieces were present in a row the next move would be to win with four pieces in a row. After this exercise, the participants were placed into pairs to play Connect Four. One player assumed the role of the “computer” and these players were restricted to only play particular rules in an algorithm (i.e. a rule sheet that could be applied top to bottom until a pattern was recognized). The other participant played freely as one normally would.

The study found that children could easily grasp the rules of the game and were able to verbalize game play strategies. However, the young players had difficulty representing their game play strategies in abstract terms using paper prototypes. The children were also able to create their own rules, but only with significant assistance by the adult helpers. A significantly challenging aspect for the children was representing various board states. For example, the children were able to verbalize the idea of blocking an opponent’s winning move, but they could not express this idea through the process of creating an initial game board state along with a second game board state representing the move. Finally, children preferred to play their own moves on one game board exhibiting highly effective winning strategies, but had trouble grasping the abstraction of the games rules using the paper interfaces to the point of frustration. When being asked to play as the “computer,” they had no problem grasping the exercise of “being the computer”.

The challenges uncovered during the session offer some insights into how games could be better designed to promote CT education for children. The first design consideration should be to attempt to integrate tacit knowledge within the game interface because children express complex CT strategies during game play activities. The point at which tacit knowledge is abstracted and generalized may be a critical point and implies that we should design game interfaces very carefully. These interfaces should help the learners explicitly link their game actions to the abstracted representations and algorithms that describe them. The next design consideration should be moving from concrete to abstract computational thinking, because traditional learning theory suggests that children progress from thinking concretely first and then to abstract principles. Following that, the idea of concreteness fading should be applied. Interfaces designed in this way increasingly highlight broader algorithmic strategies while gradually reducing the salience of the specific game situation. Finally, the design should reduce the cognitive load on the user because the split screen representation of boards states was difficult on the children while developing rules and asking them to recreate board states; however, when a child was asked what they would do given

---

3http://www.cs.umd.edu/hcil/

4http://en.wikipedia.org/wiki/Connect_Four
a particular situation on a single game board, they could easily make a move and justify their choice. It is important to remember also that the children did not express any issue with playing the role of the computer, which would imply that educational tools can be developed using interactions with AI.

4. CTARCADE

CTArcade is a web application platform with three main components: a trainer, a match reviewer, and a lobby. The user’s goal is train his/her own AI in the trainer component, and then play matches against other AIs. The lobby displays the ranking of all the AIs to foster competition among the user population. Currently, Tic Tac Toe is the only game available in CTArcade and this was purposefully done to allow preliminary testing of the various components to begin.

4.1 Intro

Another area of CTArcade is the Intro page, figure 1, which contains background information about and our motivations for developing this project. Additionally, this area also serves as a “Frequently Asked Questions” list if users are having issues with using the application during testing and in general.

4.2 Trainer

As mentioned earlier CTArcade is built on the assumption that computational thinking is already embedded in user’s game play. Instead of code editing panels in common IDEs, CTArcade users train their AIs while playing the game together. Trainer mode (not Programming mode) UI is where users can extract new rules from game play and teach his/her own AI. As shown in Figure 3, Trainer mode has a game board where users can directly play with AI. Initially the user’s AI has only one predefined rule - TAKE RANDOM - which means it always take any randomly-chosen empty tile. How to extract and teach new rules is one of our research questions, and we came up with several ways to do that.

First of all, users can select one of the predefined rules that match the user’s latest move. For example, in Figure 3, users took the top-middle tile and CTArcade found two predefined rules - TAKE ANY SIDE and TAKE RANDOM. If the AI does not know one of the rules, the user can click on a list of suggested rule possibilities to teach the rule to the AI. The table 1 describes the predefined rules.

Only a few matches will be enough to teach 3 5 basic rules, however, if users cannot demonstrate some of the rules or the AI is not able to meet the required condition, more matches will be needed. For example, if the AI is very weak, it will be difficult to demonstrate a move matching with the Block Win rule.

To address the issue of difficulty demonstrating rules, another method of training the AI called the history mode (accessed by clicking left / right arrow button at the bottom to trace back the game) was added. Under this mode, the user can directly demonstrate a move and teach the matching rule to the AI.

The last method is using a Custom Rule Maker shown in figure 4. With this component, users can teach a rule which is not predefined by CTArcade. First, the user needs to define a base pattern on the board (by assigning one of five cases for some board tiles) and then generalize the pattern by applying various transforming operations. For example, the board pattern in the figure defines a required board condition (where left and middle tiles in the first row are players) and the next move (the rightmost tile on the first row). The pattern can cover only one case of all possible winning situations, thus the user might want to generalize it by applying five transforming operations in table 2.

Rule prioritization is another crucial activity of training, as rules are applied top-to-bottom. If an upper rule (having higher priority) is applicable, all the rules below are ignored. To change the priority of rules, users can drag-and-drop any rule vertically.

4.3 Match Review

After training his/her own AI, user can test it with other users’ AIs in Match review mode. Thus Match review corresponds to debuggers in traditional programming environments. As game play may vary because of random variables.

<table>
<thead>
<tr>
<th>Title</th>
<th>Required Condition</th>
<th>Next Move</th>
</tr>
</thead>
<tbody>
<tr>
<td>WIN</td>
<td>If two tiles in a row/column/diagonal are occupied by the player and another tile is empty,</td>
<td>Take the empty tile</td>
</tr>
<tr>
<td>BLOCK WIN</td>
<td>If two tiles in a row/column/diagonal are occupied by the opponent and another tile is empty,</td>
<td>Take the empty tile</td>
</tr>
<tr>
<td>TAKE CENTER</td>
<td>If the center title is empty</td>
<td>Take the center tile</td>
</tr>
<tr>
<td>TAKE ANY SIDE</td>
<td>If any non-corner tile on the border is empty</td>
<td>Take one of side tiles</td>
</tr>
<tr>
<td>TAKE ANY CORNER</td>
<td>If any corner title is empty</td>
<td>Take one of corner tiles</td>
</tr>
<tr>
<td>TAKE RANDOM</td>
<td>No condition required</td>
<td>Take any empty tile</td>
</tr>
</tbody>
</table>

Table 1: Predefined Rules in CTArcade Tic Tac Toe

<table>
<thead>
<tr>
<th>Transforming operations</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row permutation</td>
<td>Change the order of rows in all possible ways (e.g. [1,2,3] &gt; [1,3,2], [2,1,3], [2,3,1], [3,1,2], [3,2,1])</td>
</tr>
<tr>
<td>Column permutation</td>
<td>Change the order of columns in all possible ways</td>
</tr>
<tr>
<td>Rotation</td>
<td>Rotate the board pattern 90, 180, 270 degrees</td>
</tr>
<tr>
<td>Flip</td>
<td>Mirror the board pattern horizontally and vertically</td>
</tr>
<tr>
<td>Move around</td>
<td>If the pattern is smaller than the board, move it to any place without cropping it</td>
</tr>
</tbody>
</table>

Table 2: Transforming Operations Generalizes A Basic Board Pattern
The CTArcade project was initiated by Dr. Ben Bederson. Initial research was performed by the Human Computer Interaction Lab (HCIL) at the University of Maryland. The HCIL worked to develop an activity session that provided challenges for gauging the computational thinking (CT) skills of the group of children through the KidsTeam project initiative. Analyses of this session suggested that a computational thinking tool would be beneficial to teachers and students.

The primary motivation behind this project is to develop a framework for computational thinking tools that teach users through a simple interface that enhances existing internalized skills, as opposed to a complex graphical programming tool. Rather than having users program logical rules directly in a bottom-up approach, users will "discover" and rank rules to train an "artificial intelligence" (AI). These AI will play for the users on an automated tournament server composed of other similarly trained entities. Playing on the server will result in a rank and tournament score. Users can also challenge individual AI and review the history of the games played during the challenge. These two artifacts are intended to provide motivation for the user to reflect and improve on their AI by learning from past experiences.

Below is a list of frequently asked questions about using CTArcade, as well as the project in general. If you have any problems using CTArcade or if you need any additional information, please send an email to: ctarcade@gmail.com

What is the CTArcade?
Is the CTArcade in beta?
What are the requirements to use CTArcade?
How do I sign up and login?
How do I train my AI?
What is the lobby?
How do I challenge the AIs of other users?
How are the results of challenges displayed?
How can I change my strategy in between challenges?
How are tournaments scheduled and how do they work?
How do I know who won the tournament?
What is next for CTArcade?
What games will be added to CTArcade?

Figure 1: Intro mode provide the overview of CTArcade project.

Welcome to the CTArcade lobby!
You can review your rank and score against other players using CT Arcade.
You can also select a player to challenge from the list below.

<table>
<thead>
<tr>
<th>Players</th>
<th>Score/Rank</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>yun1</td>
<td>79</td>
<td>Challenge</td>
</tr>
<tr>
<td>hardNPC</td>
<td>12</td>
<td>Challenge</td>
</tr>
<tr>
<td>moderateNPC</td>
<td>12</td>
<td>Challenge</td>
</tr>
<tr>
<td>krist</td>
<td>11</td>
<td>Challenge</td>
</tr>
<tr>
<td>tak</td>
<td>5</td>
<td>Challenge</td>
</tr>
<tr>
<td>ask</td>
<td>4</td>
<td>Challenge</td>
</tr>
<tr>
<td>easyNPC</td>
<td>4</td>
<td>Challenge</td>
</tr>
<tr>
<td>hyunjong</td>
<td>3</td>
<td>Challenge</td>
</tr>
<tr>
<td>ben</td>
<td>2</td>
<td>Challenge</td>
</tr>
</tbody>
</table>

Time until next tournament: 13:34:44

Administration:
Run Server Tournament

Figure 2: Lobby shows the leader board of all the players.
Figure 3: Trainer mode. (Left) Rules that the AI currently knows. (Center) Tic Tac Toe board where user can play against own AI. (Right) Console showing rules that have been applied to the last move made.

Figure 4: Custom Rule Maker in Trainer mode. User can manually specify a tile pattern and a next move. A specified rule can be generalized with various transforming operations such as row/column permutation, rotation and flip.
in some rules, the CTArcade server will automatically run a predefined number of matches and present the summary of results (number of winning/losing/tie games and four visualizations). The user’s goal is to find important board states and improve his/her own AI to win more games. The list of the AI’s rules on the left side allows users to quickly change the priority.

As we can see in figure 5 match review can present four different visualizations for match results. 1) List view simply shows all the matches and board states in order. 2) Step-wise animation allows user to replay all the matches in a forward / backward direction. 3) Group by winner shows winning and losing games separately to find a pattern for each result easily. 4) Game tree graph compresses all the board states into two graphs (separated by who took the first move) and merges multiple board states into one pattern if they are identical after rotation or flip operations. Patterns taken from the same match are then connected by edges; edge thickness represents the number of times this path occurred in the match. Each path represents a group of similar matches starting from the left-most board to the right-most ones. Boards having colored borders represent winning/losing matches, and by clicking on a board the user can highlight all the paths passing to the board. When the mouse cursor hovers over a tile, a tool-tip shows the rule applied for the move.

4.4 Lobby
Lobby, figure 2, has a leader board showing the ranking of all the AIs. Everyday at a fixed time, the CTArcade server runs a single elimination tournament between all the AIs and updates the ranking.

5. PILOT TEST
Before assessing the ability of CTArcade to improve computational thinking among children, we needed to perfect the design of the system and its interface. We conducted a pilot test for the system with three users. These users are all graduate student from the department of computer science and have experience on development and design of user interface. We had both male and female participants. The test was conducted on two sessions and each session took one hour on average. The goals of the pilot test was to assess the usability of the system and get general feedback, determine which part of the interface was the most easy or most difficult, how to improve the interface, and to analyze user interaction flow and determine the workflow suitable for the next phase testing.

5.1 Procedure
Each participant used the system on separate computers and each action and comment they made were recorded. The users were given instructions on how to register, sign in, and use the system. The objectives of the system were also explained verbally, although the homepage included basic instructions as well. The user’s were told how to train their AI in regards to using predefined rules, creating new rules, and prioritizing rules. They were also taught how to play matches between their own trained AI and other user’s AI. Each user trained their AI by playing at least 4 games, after which they reviewed the match results between their AI and other AIs. Finally they were debriefed on their experiences using the system.

5.2 Findings and Improvement
All the participants were positive about the visual aesthetics of the interface. Registration and sign in functionalities caused no problems. In the trainer mode, when the participants were supposed to train the AI, they suggested that the system needed to be more explanatory to distinguish between the tasks of training the AI and playing against an opponent. After the completion of each step in the training mode, the dialog to advance further in the system process was confusing for one of the participants. There were two interfaces for the rule of priority list, a vertical and a horizontal. The horizontal list caused a lot of confusion for the user’s. Therefore, we removed the horizontal list and adjusted the interfaces throughout the application.

The interface for creating custom rules was both challenging and engaging to the first participant. The user was excited to have the option to create customizable rules, but the icons were confusing. We fixed this problem by including more explicit icons and labels. As a result of this update, the next two participants found the system easier to use when creating custom rules. They suggested that for different variations of same rule, the interface is too abstract, therefore more visual cues should be utilized to help the players when creating custom rules.

For the match review component, we have provided several visualization options. Among these options, the horizontal tree visualization for match the overview was well received by the participants. The other visualization techniques were also able to convey meaningful insights about the game patterns and helped the participants analyze and respond to game situations through the creation of new rules and updated rule prioritizes. Participants also gave feedback about making the system more social and appealing for the participants by letting them create profiles for their AIs, including the implementation of avatars.

After observing how the participants interacted with the system, we came up with the activity flow that we should follow for future testing.

6. DISCUSSION AND FUTURE WORK
We will continue to develop CTArcade and its systems to complete the framework for deploying more advanced games. During this development will be regularly testing the usability and viability of the system as a learning tool. The most immediate concern is a full review of the pedagogical material looking for ways to improve upon the design of the system while strengthening the CT components.

One of the research questions we could not fully answer in this initial development of CTArcade was whether or not the social interaction between users can strengthen the effectiveness of learning. The next step is to add features promoting social interaction such as synchronous game play in which two players can play against each other at the same time or share one’s custom rules with other players.

The pilot test also suggested that the basic concept of CTAr-
Figure 5: In Match Review mode user can analyze AI's play against other AIs using four visualization methods.
CADE (managing AI) requires more explanation. An interactive tutorial and increased informational support for users will be added soon.

Additionally, it was also suggested that we increase the user’s ability to customize their profiles and AIs. After completing several more usability tests, the next major test of CTArcade will be as a classroom activity with a Kidsteam, and it will also be in a larger classroom setting.

Assuming the successful completion of all the various tests, the next big concern is selecting the appropriate games for CTArcade. Tic Tac Toe was a reasonable choice as the first game, however, it has a few limitations. First, Tic Tac Toe has a low ceiling that an AI can achieve. That means two reasonably good AIs will always draw. Second, some people may want to collaborate with others rather than compete. To address these issues, the next game would be more challenging and promote collaboration between users.

7. CONCLUSION
In this paper we introduced the first implementation of the CTArcade platform and its original game, Tic Tac Toe. CTArcade has a unique goal: engaging users to learn computational thinking skills while training his/her own AI that plays with/against other AIs. CTArcade also seeks to innovative approaches such as training by demonstration, debugging with visualization. While the pilot test results show big potential for the system, it still leaves room for improvements.

Future work falls in three categories: to promote social interaction, add more games with various characteristics, and a test of its pedagogical effectiveness through extensive user study.

8. REFERENCES