Boardgames and Computational Thinking

how to identify games with potential to support CT in the classroom

MARCO SCIREA and ANDREA VALENTE, University of Southern Denmark

Boardgames exist that explicitly address Computational Thinking (CT for short) concepts and practices. Some are actual games, while others are more akin to gamified learning activities. And since CT has been formalized only recently, many existing boardgames unknowingly might support aspects of CT. To help educators and game practitioners navigate this complex landscape, we analyze a selected sample of analog games, and propose to categorize their features with respect to CT concepts and practices. The main contribution of this paper is a novel way to identify potential CT-relevant games, that leverages on the authors' experience with digital and analog games, playful and game-based learning. Although limited, this approach appears promising and practical for CT teachers and game designers interested in adapting existing games to the classroom or developing better CT-supporting boardgames.

CCS Concepts: • **Applied computing** → **Interactive learning environments**; • **Theory of computation** → *Models of computation*; • **General and reference** → *Surveys and overviews.*

Additional Key Words and Phrases: Boardgames, Computational Thinking, education, classification, guidelines

ACM Reference Format:
Marco Scirea and Andrea Valente. 2020. Boardgames and Computational Thinking: how to identify games with potential to support CT in the classroom. In *International Conference on the Foundations of Digital Games (FDG '20), September 15–18, 2020, Bugibba, Malta.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3402942.3409616

1 INTRODUCTION

Many countries are currently in the process of implementing Computational Thinking, CT for short, in their primary and secondary education systems. The practical aspects of this process and even the definition of CT itself vary greatly from country to country; CT as a subject is sometimes associated to math and other theory-heavy subjects, while in other context it is considered more similar to a craft, as art or design. Regardless of the approach to CT, the introduction of this new subject in the existing curriculum presents challenges and problems for teachers and learners, as well as opportunities.

In this paper we want to look at tabletop games and the various ways in which they can support CT learning. It is clear from the game-based learning research that games are a promising way to teach CT, especially to young pupils (as discussed in [Marchetti and Valente 2015]); and in fact, various digital and analog games exist to teach CT.

From previous work in Danish primary schools, we know that analog games are often used in the class and in particular we observed learning activities in which pupils are asked by their teachers to alter existing tabletop games, in order to show knowledge about specific subjects or topics. A classic example is a task in which pupils have to reflect about a book they read; the teacher divides them in small groups, and each group develops a variant of the goose game

Manuscript submitted to ACM

where the gameplay involves answering question about the book. Finally, pupils play each other's games to test their knowledge and the design skills of the other groups.

In our experience with tabletop games and learning, we notice that they can overcome one of the major problems of digital educational games: involving groups of players. Educational digital games have a tendency to cover the two extreme cases, that of a single player and that of massive multiplayer; however, educational digital games supporting small to medium groups of players are much more rear, and even at a conceptual level there is a known lack of game mechanics and design patterns to develop such small group supporting games. We see the capacity to involve small to medium groups as a strength of tabletop games, and a feature that could be very useful in the classroom. Another strength of tabletop games is their flexibility: as in the goose game task, these games are much easier to manipulate, modify, and extend for teachers and pupils than digital games (as known in the game-based learning research, for instance in [Lin and Hou 2016]).

In this paper we explore the spectrum defined by three categories of games: i) tabletop games explicitly declared to be relevant for CT, ii) analog games that have mechanics that can provide *scaffolding* for CT concepts; these are existing games that might implicitly support CT, and iii) educational, gamified activities, inspired by tabletop games (that are not actual games per se).

We find that games in these three categories have the potential to be useful in classroom situations, to present, discuss and practice CT, especially for groups of learners. We are interested in outlining the features of those games, and how their mechanics can support CT explorations. Knowing the features of games in this spectrum will allow teachers and game researchers to identify candidate tabletop games to adopt in a more confident and quick way; we consider this an important contribution to the practice of CT, given the sheer volume of available games to sift through. Moreover, a categorization of tabletop games with respect to their relationship with CT concepts will also shed light on possible new design patterns and game mechanics to adopt in digital CT-supporting games. Hence, this paper will provide an analysis of a select few games and discuss how they incorporate, support or more generally relate to CT concepts as part of their game mechanics.

Section 2 introduces CT, tabletop and digital games related to it. Section 3 presented our analysis of a selection of boardgames, followed by a discussion of interesting game features and mechanics, and how they could be used to support CT in education (section 4); criteria to help educators identify games of interest for the teaching of CT are also outlined. Section 5 concludes the paper.

2 BACKGROUND THEORY

2.1 Computational thinking

Computational thinking has been defined as "a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use" [Wing 2006]. That is a very vague definition, and there is still much debate over a more precise definition of what computational thinking is, and what are its components.

Brennan and Resnick developed a definition of computational thinking that involves three key dimensions: "*computational concepts*(the concepts designers employ as they program), *computational practices*(the practices designers develop as they program), and *computational perspectives*(the perspectives designers form about the world around them and about themselves)"[Brennan and Resnick 2012]. The computational concepts they identify are *sequences*, *loops*, *parallelism*, *events*, *conditionals*, *operators*, and *data*. Computational practices are defined as *being incremental and iterative*, *testing and debugging*, *reusing and remixing*, and *abstracting and modularizing*.

2.2 Analog games for teaching CT

Many activities have been designed to be used in classrooms to teach CT[1][2][3]. These activities often resemble more classical class activities, like exercises, homework, and small tasks, in general sharing very little of the characteristics of games. Closer to actual games we find activities that involve tinkering and the usage of play-fields, markers, tokens, scores, and other boardgame-related features. Many examples of these can be found on CS Unplugged[4], a collection of teaching material geared towards teaching CT concepts through games and puzzles without the mediation of a computer, but with the usage of physical, readily accessible objects (such as strings, paper, cards, etc.). A similar line of research is discussed in [Valente 2004], where a printable set of cards, called Computational Cards are defined and their expressive power explored.

An example of a commercial product geared towards CT is *Turing Tumble*[5], a gamified series of tasks and exercises in which the player creates computational machines. A more theoretical approach to using physical devices to create computational machines can be found in *Train and Track*[6], a system describing Turing machines using train tracks (also using Duplo tracks [7]). Finally, another typology of CT-supporting game/toys can be seen in hardware/software hybrids, such as robot-based teaching curricula[8], or the playful experience for very your pupils of Cubetto[9]

From the research perspective, a few boardgames have been designed with the explicit purpose of supporting CT concepts, the most notable are: *Crabs & Turtles* [Tsarava et al. 2018] and RaBit EscAPE [Apostolellis et al. 2014]. The former is discussed at length in section 3.8.

2.3 Computer games and CT

"Programming games" is a genre of videogames that is becoming increasingly popular. As of this writing, on Steam (a popular game distribution platform) there are 118 games that users have tagged as "programming" games[10]. These games can be quite different in how they approach their programming content, ranging from simple 2D sprites for representing relationships between objects as in *Baba is You* (Hempuli Oy, 2019), to simulating realistic computer systems as in *Hacknet* (Team Fractal Alligator, 2015), to presenting users with a low-level pseudo-assembly language as in *SHENZEN I/O* (Zachtronics, 2016) or *Human Resource Machine* (Tomorrow Corporation, 2015), to allowing the user to construct complex production machines either in 2D as in *Factorio* (Wube Software LTD., 2016) or in 3D as in *Infinifactory* (Zachtronics, 2015). Most of these games are single-player experiences, although there are notable exceptions such as *Hacknet* and *Gladiabots* (GFX47, 2019), which instead rely on player-vs-player mechanics.

Another interesting point is that most of these games, while supporting CT in some form, are not explicitly games-for-learning. This makes them quite hard to access to people that have no previous programming experience. An example of this can be seen in *SHENZEN I/O*, where the player's task is to code a solution to given problems by using extremely limited resources: an assembly-like language, code blocks of no more than 16 commands, very few memory registers, and having to deal with information transfer between code blocks (see Figure 1).

[1] https://www.stem.family/activities/computational-thinking-activities/
[2] https://code.org/curriculum/course3/1/Teacher
[3] https://www.barefootcomputing.org/homelearning
[4] https://csunplugged.org/
[5] https://www.turingtumble.com/
[6] http://www.cr31.co.uk/stagecast/trains/tt0_intro.html
[7] http://www.cr31.co.uk/stagecast/trains/tt8_duplo_lout.html
[8] https://kubo.education/
[9] https://www.primotoys.com/
[10] https://steamdb.info/tags/?tagid=5432

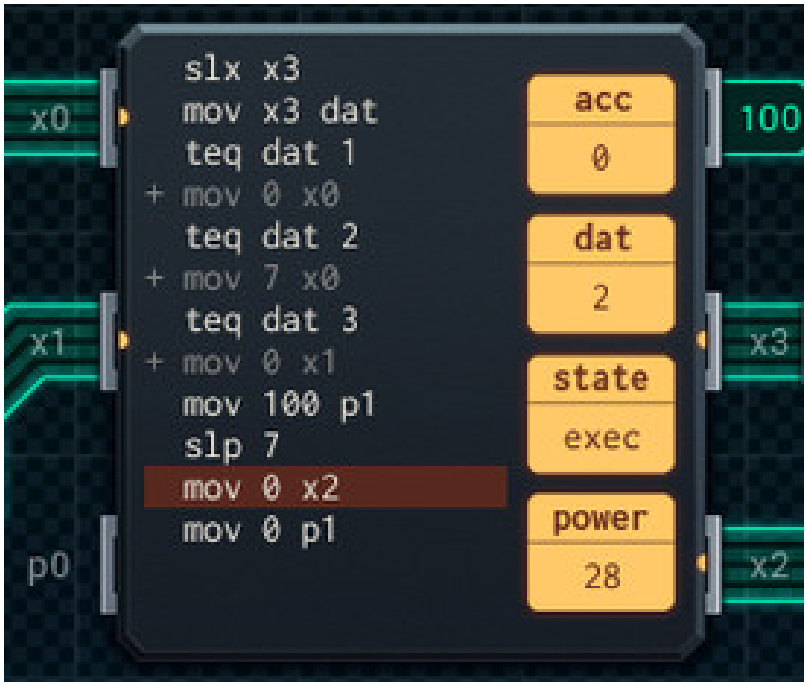

Fig. 1. An example of a block of code from SHENZEN I/O (Zachtronics, 2016).The data-lines that can be seen coming out of the block go to other ones, which together create the solution to the given problem.

Computer games have an obvious advantage in presenting CT concepts, since they can simulate a system's execution much faster than can be done on tabletop games which supports the practices of being iterative and incremental, on top of making debugging and testing much easier. That said, the generally high inaccessibility to the games, the single-player focus, and the difficulty in adapting/modifying them for the classroom does not make them an obvious choice for teaching CT.

Various computer-aided resources have been designed to support teaching CT. What appears to be the most popular approach consists of providing a graphical simplified programming language, and a 2D graphic environment, as in *Scratch*[11] and *Hopscotch*[12]. Other notable examples of computer-aided resources are *AgentCubes*[13] which provides a way to simply create programs/games in a 3D environment, *App inventor*[14] which provides a framework to create apps that can run on mobile devices, and *Program your Robot* [Kazimoglu et al. 2012a,b] which also includes hardware elements.

We would also point that some research on using procedural content generation for creating adaptive activities supporting learning CT has recently been conducted. This research focused on the expression of the *sequence* concept through puzzles which adapt to the player's skill and understanding of the concept [Scirea 2020].

[11] https://scratch.mit.edu/
[12] https://www.gethopscotch.com/
[13] https://agentsheets.com
[14] http://appinventor.mit.edu/

Table 1. The games examined in this paper

Game	# of players	Versus/Coop
Robo rally	2-8	Versus
Twin tin bots	2-6	Versus
One zero one	2	Versus
SET	1-20	Versus
Qwirkle	2-4	Versus
The dragon & flagon	2-8	Versus
Space cadets	3-6	Coop
Crabs & turtles	unspecified	Versus

3 BOARDGAMES ANALYSIS

In this section we will look through the eight boardgames (see table 1) and discuss which CT concepts they integrate and how they are expressed through the game-mechanics. These games were selected by looking for commercial games of some renown that we thought appeared related to CT concepts, while being generally quite different games in both gameplay and setting. An exception is the last game, Crabs & Turtles [Tsarava et al. 2018], which is a game developed explicitly for teaching CT concepts. The reasoning to include this game is that it should allow us to observe similarities and differences between games that support CT explicitly and ones that can provide such support implicitly. From our experience in educational CT games, they have typically the same “ingredients" (and flaws) as this one price-awarded “representative", so we consider it to be good representative of its class.

3.1 Robo Rally

Robo Rally (Wizards of the Coast, 1994) is a racing game designed by Richard Garfield (better known for the collectible card game Magic: the Gathering), in which the players' objective is moving their token to various objectives as fast as possible. The player tokens (robots) are only controllable indirectly by creating a plan using cards that represent instructions to be "executed". Once the players have decided their plan in secret, these are executed by each player unveiling their actions one at a time. Of course, players can interfere with each other, which introduces a strategic element in the creation of the plan. Moreover, the game includes a modular board which can be used to create different maps, so games can be very different from each other.

The game includes some elements that makes it seem like a "programming" game, such as the action queue that the players use to create their action-plan, which instinctively make the game sound like it should include many CT concept. Disappointingly, once we give it a more thorough analysis, we can see how the game only utilizes the *sequences* concept, which is at the core of the gameplay. The game does not really include any more complex structures, such as loops, conditional, operators, and data. This is mainly due to the nature of the action cards, where only very simple actions are possible (move forward, turn clockwise, attack, etc.) with no way of using information from the board (or the state of the player) to influence the sequence.

A measure of parallelism can be observed, if we consider that the player actions are executed almost in parallel (the players take turns executing their respective action 1, then action 2, and so on). Another part of parallelism that is expressed is how the players' tokens can, purposefully or not, interfere with each other's plan.

When we consider computational practices expressed by the game, not much can be observed. The one-use nature of the sequences (execute once and throw away) combined with the changing state of the board don't allow for any

type of incremental/iterative methodology. The same can be said about testing and debugging, since the only way that would be possible is in the player's mind and using incomplete knowledge of the game-state. Reusing and remixing are also concepts that are not included in the game explicitly or implicitly, unless we consider the player learning from their own or other players' mistakes. Given the previously mentioned characteristics of the game, the game doesn't include any form of abstraction/modularization. Finally, we would note that, although not many CT concepts are expressed through Robo Rally, we do believe the game promotes algorithmic thinking since the core of the game is to create sequences of actions to solve a problem.

3.2 Twin tin bots

Twin tin bots (TTB) (Flatlined Games, 2013) is a game that in many ways appears similar to Robo Rally: the players are in charge of creating sequences of actions to move their tokens (robots) around the board to collect as many resources as possible. That said, a few key differences make TTB have a very different gameplay and express other CT concepts. The main differences are that:

- in TTB each player is controlling not one, but two robots
- action sequences are shorter than in Robo Rally: each robot can be "programmed" with up to 3 actions, compared with the 5 used in Robo Rally
- action sequences are not discarded after use, but can only be slowly modified in following turns, this means that "changing the robots' programming" is a limited resource
- action sequences are not secret, each player can observe the current programming of each of the other players at any time

TTB has a static field of play, but resource placement can create different challenges for the players. The CT concepts expressed by TTB have a lot of similarities with what we discussed for Robo Rally in the previous section: sequences are a key component of the game, while most other concepts do not really appear. The reasons are the same as discussed before, mainly a consequence of the simplicity of the possible actions.

The one concept that is more prominently portrayed is parallelism: in TTB each player's two robots move at the same time. This requires the player to not only consider possible interference by the other players but also how to program their own robots so that they do not conflict with each other. Another feature that supports parallelism is how the players can observe the other players' planned programming, allowing them to consider that in their strategy. Moreover, the already mentioned limitations in changing sequences makes sure that a player is not able to completely subvert the other players' expectations of their plan.

Another new computational practice that appears in TTB is being incremental and iterative. We would argue that in TTB being iterative is more prominent, since the game only allows small changes to the programming of the robots, but since 1) the state of the game continuously evolves and 2) the sequences are very short, there isn't really an incremental definition of an algorithm to reach a specific goal. The game also supports algorithmic thinking, since the players need to create sequences that would be working in a long-term timeframe, since modifications are limited.

3.3 One Zero One

One Zero One (OZO) (The Game Crafter, 2013) is a strategy card game for two players. We selected this game since it presents aesthetics that evoke how memory is represented in computers. In the game each player has an identical deck of cards with a 0 or 1 symbol plus (possibly) an extra "command" that allows for manipulation of the cards on the table.

The game presents itself as an area-influence game, in which the players' objective is to fill the board with a larger amount of the players' symbol (either 0 or 1). The playing of cards is presented as filling memory registers with bits (the cards), the "commands" featuring on the cards are also evoking programming concepts (e.g. goto, if/then, cut, save, etc.).

When looking at the CT concepts expressed by the game, we can immediately see how the gameplay itself has very little in common with CT apart from the theme of the game. Since each player alternates playing a card there is no mention of sequences, loops, parallelism, operators, or events. That said, the game can be seen as modifying the bits in a memory register so we could argue that it does express the data concept. Sadly, the way that special actions are constructed in the game have little to no connection with how memory is dealt in computers, so it is questionable how useful from a learning perspective this actually is. Another characteristic that we firstly observe in this game is the use of conditionals, since some cards have some instructions that change depending on the placement of the card. Unfortunately, this is a quite basic use of conditionals, since the player has no way to define their own ones but can only influence them by choosing where to place the card.

3.4 SET

SET (Set Enterprises, Inc., 1988) is a real-time card game designed by Marsha Falco about abstract pattern recognition, specifically the players are tasked to recognize a "set" here defined as three cards that are either all alike or all different in each attribute depicted on the card. For example, if all the cards have the same color, but different shapes, number of shapes, and shading, then those cards are a set. The game originated as a coding system the designer used in her job as a geneticist. Moreover, the game gained some interest from the combinatorics aspect of the game[15], including a proof a general version of the game being NP-complete [Kalyanasundaram and Schintger 1992].

This game was selected since pattern recognition and being able to abstract concepts are concepts related to CT. However, on closer inspection, the game does not really display any CT concepts since the only abstract concept is the definition of the "set" and the patterns only apply to this one concept. We consider this game as a "false positive", a game that on the surface might appear related to CT but does not express any CT concepts.

3.5 Qwirkle

Qwirkle (MindWare, 2006) is similar to SET, and is a variation of the classic domino. It is turn based and uses tiles with various shapes and colors. The game supports 2 to 4 players, from age 6 on, and the playtime is declared to be around 45 minutes. The main game mechanics of this game is to look for sequences (of tiles), and since sequential thinking and pattern recognition are parts of CT, we consider this game related to those CT topics. Moreover, the game exhibits some parallelism, since players have to look for sequences in two dimensions, to maximize score. A player gets more points if they are able to complete multiple sequences at the same time, in 2 dimensions, by placing her tile strategically. Data structures and types are fundamental elements of Computer Science and CT, and this game has some elements of those topics: in fact, tiles that have both a shape and a color, and the players have to reason about combinations of two types in a tuple-like data type. Other aspects of CS that are present in minor forms are: incremental and iterative thinking, because players build sequences in a step-by-step fashion, reusing, remixing and modularizing, since players can only combine and reason about sequences of the given tiles. Finally, algorithmic and heuristics thinking are supported by this game: heuristic reasoning is required to maximize the score by exploiting the types of tiles and the 2-dimensional,

[15] https://henrikwarne.com/2011/09/30/set-probabilities-revisited/

Table 2. Observed CT concepts in the analyzed games. Where the concept was clearly observed it has been marked with "+" , while where there are elements of the concept but in an incomplete/ambiguous fashion it was marked with "~"

	Concepts						
Game	Sequences	Loops	Parallelism	Events	Conditionals	Operators	Data
Robo rally	+		~	~			~
Twin Tin Bots	+		+				~
One Zero One					~		+
Set				~			
Qwirkle	+		~				~
The dragon & flagon	~	~			+		~
Space cadets	~	~	+	~	~	~	+
Crabs & Turtles	+	+	~	~	+	+	~

Table 3. Observed CT practices in the analyzed games. Where the practice was clearly observed it has been marked with "+" , while where there are elements of the practice but in an incomplete/ambiguous fashion it was marked with "~"

	Practises				
Game	Incremental/Iterative	Testing/Debugging	Reusing/Remixing	Abstracting/Modularizing	Algorithmic thinking/Heuristics
Robo rally			~		+
Twin Tin Bots	+		~		+
One Zero One					~
Set					
Qwirkle	~		~		+
The dragon & flagon	~		~	~	+
Space cadets	~	~	~	~	+
Crabs & Turtles	~	~	~	~	+

multiple sequences on the game table. We find that even if this game does not explicitly declare so, some CT concepts can in fact be supported and experienced by playing Qwirkle.

3.6 The dragon & flagon

This game (Stronghold Games, 2016) reminds of to the 1980s classic *HeroQuest*; it has a board, as well as cards, miniatures and many 3-dimensional plastic items. The game is turn-based, with time points to control how many actions each player can perform in each turn. Games last around 1 hour, and the players can be from 2 to 8, from 10 years old. The gameplay is quite complex, since the game takes place in a tavern. The state of the game is kept in the playing board, which is divided in cells, and populated with items and the miniatures representing the players. The players play by enacting a brawl, throwing items at each other (mugs and chairs for example), and trying to cause damage to each other, in a humorous atmosphere. Brute force and magic spells can be used, and each player also keep track of her score and reputation points in a separate *data board*.

Given the turn-based nature of this game a global *time marker* is present, to keep track of the turns and how many actions can be performed by each player (in turn); from a CT point of view, this sequential time tracking system is reminiscent of a program counter. Moreover, the way turns work here can be related to loops (a CT concept), at least in a superficial form. Players are faced with plenty of options and must reason on what action to take, how to use the time points, and how to inflict max damage or get most reputation points; this is relevant for the CT concept of *conditionals*. Another CT concept that this game supports at least in a partial way is that of data structures: each player has to keep a table of data, using physical markers and writing down points for her character. In programming this data structure is similar to an object. The fact that the same pieces are reused but randomized from game to game, suggest a minimal

relation with the CT concepts of reusing, remixing and modularization; and finally the game offers good possibilities to practice algorithmic and heuristics thinking, in particular in the strategies that players have to develop to maximize their score/reputation points.

3.7 Space cadets

Space cadets (Stronghold Games, 2012) is another game from the same designers and publishers of The dragon & flagon; however, this game is much more complex, offering more roles for the players, multiple boards, various decks of cards and many different types of tokens. In a sense this game can be considered an analog simulation of a videogame. The gameplay is highly cooperative and distributed, divided in multiple mini-games, each with emphasis on resource management and coordination among players. The games can last 1 to 2 hours and involve 3 to 6 players (from age 8 up).

The presence of the many mini-games and management tasks, together with a turn-base structure, makes this game moderately related to sequential thinking. The distribute and cooperative nature of the gameplay, and the needed coordination among players offers a good playground for practicing the CT concept of parallelism. The game is real time, and involves rather continual and fast decision making, and these features suggest at least partial support for reasoning about events and conditionals. Given that the different mini-games and many roles require players to master multiple sets of operations, the CT concept of *operators* is also somewhat present in Space cadets. Operations and data are often interrelated, and this game presents data in many forms: players have to keep multiple boards of data as they play, for each character and mini-game.

Planning missions and repeating steps to incrementally accomplish goals are central parts of this game. This suggests that incremental and iterative thinking are required at least in some form. Finally, to play this game efficiently, the players have to think algorithmically, plan to solve puzzles and adopt different heuristics depending on the roles and mini-games.

3.8 Crabs & Turtles

In [Tsarava et al. 2018] the authors discuss their game series called Crabs & Turtles. This game is an example of gamified educational activity and has won several awards in the Game-Based Learning community. The game series is aimed at groups of 10 to 15 players, in the age range from 6 to 12. According to the game's authors the main target group are primary and secondary school pupils, but the group can be extended to include older kids and adults with no technical background too. A typical play session lasts around 2 hours, and the game is composed of three sub-games: the Treasure Hunt, the Race, and Patterns.

A remarkable feature of this game is the large board, which makes it more a *floor* game than tabletop. The first sub-game reminds of a space- and time- discrete LOGO simulation (as discussed in [Resnick et al. 1990]). The second game is a *goose game* variant, where the players have to answer math and Computer Science questions in order to advance. And the third is a variation of games like SET (or the Lynx board game, by Educa Borras).

The three sub-games were designed to support specific CT concepts and activities: sub-game 1 for example supports sequences, decomposition, algorithms and evaluation (according to [Tsarava et al. 2018]); sub-game 2 instead supports operators, constants and variables, conditionals, events, loops and abstraction. The third sub-game supports conditionals as well, but also patterns.

The major disadvantage of this game seems to be its educational origin: the game is composed of lengthy teaching-like activities, all of which require adult supervision to keep the participants within the game rules and possibly maintain

Table 4. In this table we show what game mechanics are related to CT concepts and practice. The labels for the mechanics are a collection of all the game mechanics annotated for the analysed games on the boardgame database \url{https://www.boardgamegeek.com}, with the addition "resource management'. We decided to include this last one since it is relevant to some of the games we discussed and we find it one of the few characteristics that is connected with the operators concept. Where the concept was clearly observed it has been marked with "+" , while where there are elements of the concept but in an incomplete/ambiguous fashion it was marked with "~"

	Action queue	Modular board	Simultaneous action selection	Simulation	Area majority/influence	**Hand management**	**Pattern recognition**
Sequences	+						
Loops							
Parallelism			+				
Events			+				
Conditionals	~					~	
Operators	~						
Data		~				~	~
Incremental/Iterative	~	+		~			
Testing/Debugging	~						
Reusing/Remixing		~					
Abstracting/Modularizing		~					~
Algorithmic thinking/Heuristics	+	~	+	+	~	+	

Set collection	**Variable player powers**	**Variable setup**	**Pattern building**	**Tile placement**	**Cooperation**	**Real-time**	**Turn-based**	**Resource management**
							~	
							~	
					+	+		
						+		
					~	~		~
								+
~	~					~	~	+
		~		~				
		~		~				
~		~	~					
			+	+	+			+

the motivation to play. The scoring rules are complex and can require an external judge (again a teacher, or other adult/expert) and might not be very *algorithmic* (following [Tsarava et al. 2018]). The game appears to be a good gamified learning activity, but lacking in intuitive and fun gameplay, which instead characterize most games analyzed in this paper.

Looking more closely at the game and its sub-games, we could see that sequences, conditionals and loops are clearly well covered, while parallelism is only marginally present, especially since the game are mainly competitive. Operators and data are also covered (even if they are not explicitly mentioned by the authors); in fact, sub-game 2 makes use of many math operators, even if they are not directly related to CT or programming. We consider data involved in the evaluation of constants and variables (in sub-game 2); moreover, the turtles in the first sub-game have attributes, and a structure that resembles objects or character boards in other of the games analyzed in this paper. Given that this game was designed with CT-learning in focus, we were surprised to find almost no presence of testing/debugging related thinking or practice. For example, it is not clear what happens if a player moves a turtle or crab in a *wrong way*, during sub-game 1. That could be a place to present players with the need to try and correct the turtles code-like rules.

Finally, algorithmic and heuristics thinking are well represented in this game: planning and step-by-step reasoning are central to many parts of the sub-games, and problem solving is required by the teams of players to solve the (math) puzzles.

4 DISCUSSION

In the previous section we have analyzed our sample games and discussed which CT concepts and practices are included or supported by them, as summarized by tables 2 and 3. We have also defined a set of game mechanics that appear in one or more of these games and can be of relevance for CT concepts and practices. This set was defined by re-categorizing the

game mechanics declared for each game in the boardgame database Board Game Geek[16]. This resulted in 15 mechanics, to which we added *resource management*, as it felt strongly relevant to most of the discussed games. The following discussion is based on these 16 *supportive* mechanics and their relationships to CT, as shown also in table 4.

4.1 Supportive mechanics

The mechanics that seem to be the most supportive to CT concepts are *action queues*, *simultaneous actions*, *board modularity*, *cooperation*, and *resource management*.

Action queues are unsurprisingly the single mechanic that seems to support the most concepts, since it can be very close to the concept of programming. Parallelism and events seem to be related to real-time or simultaneous action execution mechanics. Modular boards can support incremental thinking, especially when expanding a map. This is especially true in games where the players have to build the game-board as the game progresses. An example can be found in Carcassonne (Hans im Glück, 2000), where tile placement is directly connected with strategy and is the largest part of the players' interaction with the game-world. Nevertheless, it's important to note that compared to incremental thinking, iterative thinking is much more seldom, since very few games allow the player to create a sequence of actions, try them, and improve them before using them again. *Reusing/mixing* and *abstracting/modularizing* could also be supported by modular or reusable elements (tiles, cards, tokens, etc.); however, we consider this relationship weak, since reusing/mixing is almost entirely an emergent behavior, and the abstracting is done by the game designer and not on player-generated structures.

Cooperative games appear to be more supportive of parallelism, since they usually require players to coordinate actions and form a cohesive multi-agent strategy.

When considering sequential and parallel actions in games, we found that turn-based and real-time gameplay seem to support rather different, possibly opposite, CT concepts. Where real-time supports events and parallelism, given that each player can act at any time (usually with atomic actions), turn-based games support reasoning in terms of creation and use of sequences of actions. Interestingly, hybrid approaches exist as well, as the one discussed for Robo Rally, where the players' "turn" is executed in parallel for all players, showing a possible way to reconcile the creation/use sequences and parallelism.

Considering the mechanic of resource management, we found that it relates to the CT concept of *data*. While the concept of *operators* is not typically supported in the games we analyzed, *data* seems present at least weakly in almost all games (see table 4). Interesting data-related game elements are: typed or color-coded cards or items in a game, character sheets and tables to keep scores, and complex usages of tokens to trace the state of game activities (for example many sub-games in Space Cadets use a variety of tokens to represent allocation and management of resources over time). Another example of quite explicit data management and operations can be seen in other types of analog games, like for example *Magic: the Gathering*, where cards have specific "classes" (monster, spell, land, etc.) and each specific card is an "instance" of the particular class, showing concepts of Object-Oriented Programming languages. Managing one's cards, thinking in terms of their types and features, is a central problem in these kinds of games, and in CT terms it becomes a problem of applying operations to manipulating possibly complex objects using the resources available to the player.

Finally, strategic thinking in boardgames is also found to be interesting with respect CT, even if it is not a mechanic per se. We found for instance that most mechanics that enable the player to elaborate complex strategies to play efficiently, or to maximize score by considering multiple goals at once, lead to algorithmic or (at least) heuristic thinking.

[16] url{https://www.boardgamegeek.com}

We can observe that, out of the 16 mechanics we have identified, half of them are somewhat related to this CT practice. Unsurprisingly, most games include one or more of them, as most games require some form of strategy. Counterexamples are games like SET (section 3.4), where the gameplay is driven by pattern recognition, and not so much by heuristic thinking.

4.2 Missing concepts

A CT practice that is rarely present (or weakly present at best) in the games we analyzed is that of testing/debugging. In our experience with playful learning ([Brown and Scirea 2018], [Valente and Marchetti 2020]) this can be because both CT practices require that the learners iterate between the *design space* and the *use space* (sometimes called *problem domain* and *user domain* in software development). For instance: when pupils are given the task to design a tabletop game based on a book they read, they will have to create prototypes (with their *designers hat* on, as discussed in [De Bono 2017]), then test them themselves (with their *players hat* on), and perhaps discover misalignment between what they wanted as designers and what they actually got as players. The fact that testing and debugging arises more naturally in situations that have 2 contexts, makes these practices difficult to find in boardgames, since it would require a gameplay somehow broken in two phases, design and use, instead of the more typical gameplay where players are directly involved in the actions and decisions of the game.

Loops are another rare concept, since in most cases the players' actions are discarded after use (or can be disregarded in the future). However, a weak connection can be established between the CT concept of loops and the gameplay-*loop* present in boardgames, especially in turn-based games, where the player is often limited to some specific amount/type of actions. One notable example of a game that does incorporate the concept of loops is the aptly named Loop Inc. (Eagle-Gryphon Games, 2015), in which the players' actions in one turn must be executed in the subsequent turns.

4.3 How to identify CT-relevant games

In this section we want to establish criteria to help educators and game designers navigate the rather vast space of boardgames and identify with better confidence which could be adapted and deployed in CT classes. We are interested in ways to spot potential CT-relevant boardgames even when these games are not declared explicitly to support CT; an advantage of this approach is that it allows to include also older games, which otherwise might be missed.

Given our analysis we are aware of the complex spectrum of features and parameters to consider when categorizing boardgames, hence we will not try to define a precise ranking, nor to suggest a quantitative metric. However, in our experience adapting existing tabletop games to learning contexts can present various degrees of difficulty and can succeed to different extents. So, we decided to define a fictional *best CT-supporting game*, that represents a game that is very clearly easy to adopt in CT classes, and can cover most CT concepts and practices. Informally, this *best* game can be thought as a reference, and we suggest to use it as a way to define a distance with respect to any other boardgame: the closer a boardgame is to our best game, the more easy and useful it would be for teachers and other game designers, working within the CT domain.

Our best game has action queues, cooperation between the players, resource management, modular pieces, and a hybrid turn-based/real-time game progression. This game should have a good set of non-trivial goals for the player to maximize, and with reusable components to be mixed in various ways before or during play. There should also be items of multiple, possibly orthogonal, types. To compensate for the difficulty of covering (or even supporting) the testing/debugging practice, this "optimal" game can be presented to the learners in the class as an iterative activity of *design for others to play*.

5 CONCLUSION

Given the increasing interest in playful ways to support learning of Computational Thinking, we decided to look at the space of boardgames to study the relationships among game mechanics, play styles and other game elements, and CT concepts and practices. The result is an in-depth analysis of a selection of boardgames, and a proposal for a method to help CT educators and game designers to recognize the potential CT-supporting qualities of existing boardgames.

The main limitation of this paper's results is how the set of game mechanics we obtained in this study obviously does not cover all the CT related ones. Conducting a similar analysis on a larger corpus of games would likely yield a larger set of promising mechanics, but we expect that the ones we discussed in this paper should already be very helpful in finding boardgames that can support CT learning. Another limitation, that is exemplified by the game of SET, is that the mechanics we have discussed can be, *but are not necessarily*, supportive of CT. This means that our identification criteria are only to be considered as a heuristic, useful but limited to reducing the space of possible game candidates; teachers should still make sure that the mechanics of the candidate games are used in a way that supports the desired concepts and practices.

Future work would include workshops with teachers to assess the quality of our assumptions and criteria to identify CT-relevant board games. Moreover, from a more technical, programming-related point of view, the games we analyze in this paper offer direct inspiration for tasks that CT teachers can assign their pupils. A possible scenario we are planning to test is one in which a primary school teacher who works with CT and Scratch or Python organizes play sessions for groups of pupils, using existing boardgames. The game is then analyzed in class and is the basis for possible programming challenges.

To summarize, the paper presents a set of mechanics that we have found to support different aspects of CT, which can be useful when selecting existing boardgames to use in the classroom in the context of teaching CT.

REFERENCES

Panagiotis Apostolellis, Michael Stewart, Chris Frisina, and Dennis Kafura. 2014. RaBit EscAPE: a board game for computational thinking. In *Proceedings of the 2014 conference on Interaction design and children*. 349–352.

Karen Brennan and Mitchel Resnick. 2012. New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada*, Vol. 1. 25.

Joseph Alexander Brown and Marco Scirea. 2018. Procedural Generation for Tabletop Games: User Driven Approaches with Restrictions on Computational Resources. In *International Conference in Software Engineering for Defence Applications*. Springer, 44–54.

Edward De Bono. 2017. *Six thinking hats*. Penguin UK.

Bala Kalyanasundaram and Georg Schintger. 1992. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics* 5, 4 (1992), 545–557.

Cagin Kazimoglu, Mary Kiernan, Liz Bacon, and Lachlan MacKinnon. 2012a. Learning programming at the computational thinking level via digital game-play. *Procedia Computer Science* 9 (2012), 522–531.

Cagin Kazimoglu, Mary Kiernan, Liz Bacon, and Lachlan Mackinnon. 2012b. A serious game for developing computational thinking and learning introductory computer programming. *Procedia-Social and Behavioral Sciences* 47 (2012), 1991–1999.

Yi-Hui Lin and Huei-Tse Hou. 2016. Exploring young children's performance on and acceptance of an educational scenario-based digital game for teaching route-planning strategies: a case study. *Interactive Learning Environments* 24, 8 (2016), 1967–1980. https://doi.org/10.1080/10494820.2015.1073745

Emanuela Marchetti and Andrea Valente. 2015. Learning via Game Design: From Digital to Card Games and Back Again. *Electronic Journal of e-Learning* 13 (2015), 167–180.

Mitchel Resnick, Stephen Ocko, and Seymour Papert. 1990. *LEGO/logo–learning through and about design*. Epistemology and Learning Group, MIT Media Laboratory Cambridge, MA.

Marco Scirea. 2020. Adaptive Puzzle Generation for Computational Thinking. In *International Conference on Human-Computer Interaction*. Springer, 471–485.

Katerina Tsarava, Korbinian Moeller, and Manuel Ninaus. 2018. Training Computational Thinking through board games: The case of Crabs & Turtles. *International Journal of Serious Games* 5, 2 (Jun. 2018), 25 – 44. https://doi.org/10.17083/ijsg.v5i2.248

Andrea Valente. 2004. Exploring theoretical computer science using paper toys (for kids). In *IEEE International Conference on Advanced Learning Technologies, 2004. Proceedings.* IEEE, 301–305.

Andrea Valente and Emanuela Marchetti. 2020. StickAndClick: sticking and composing simple games as a learning activity. In *Proceedings of Learning and Collaboration Technologies: Designing Learning Experiences. HCII 2020 (Lecture Notes in Computer Science).* Springer, Germany.

Jeannette M. Wing. 2006. Computational thinking. *Commun. ACM* 49, 3 (March 2006), 33–35. https://doi.org/10.1145/1118178.1118215