Assessment of Computational Thinking (CT) in Scratch Fractal Projects: Towards CT-HCI Scaffolds for Analogical-fractal Thinking

Chien-Sing Lee[1][a] and Bo Jiang[2][b]
[1]*Department of Computing and Information Systems, Sunway University, Malaysia*
[2]*College of Educational Science and Technology, Zhejiang University of Technology, China*

Keywords: Computational Thinking, Fractal Thinking, Scratch Fractal Projects, HCI, Assessment.

Abstract: Learning from patterns and everyday creativity are two key trends in creative education. However, it is not easy to learn from or to create meaningful patterns. Fractals are repetitive patterns, which can result in interesting outcomes. Patterns can be based on a recursive whole or recursive modifications of decomposable parts of the patterns. However, developing fractals or relating fractals to real-life applications or creative innovations is not that easy. Since pattern recognition, recursion and relation to real-life applications are part of computational thinking (CT), we find potential in assessing CT skills. We scope our research to fractal projects at the Scratch website. We aim to identify correlations between the respective scores for each project's constructs corresponding to the respective total CT scores and to identify important human-computer interaction principles in scaffolding CT/fractal/fractal thinking development. Significance lies in identification of HCI design factors, possibility of using these findings as guides to better predict a student's performance/mastery and to identify areas and strategies for improvement. Future work within a Restorative Innovation Framework concludes.

1 INTRODUCTION

Computational thinking is gaining prominence and popularity in diverse computer science curricula. Wing's (2006) definition of computational thinking covers both the technical and human aspects, i.e., solving problems, designing systems and understanding human behaviour. Subsequently, curricula often include: abstraction as the key CT attribute, followed by data structures (symbols and representations), systematic information processing, algorithmic flow control (such as iteration, recursion, parallelism and conditional logic), efficiency (debugging and systematic error identification) and performance constraints.

Brennan and Resnick's (2012) definitions re-categorized these into three aspects: computational skills, practice, concepts. We find both definitions to be integral, but targeted at slightly different audiences, users.

On a broader front, the US National Research Council (2010 p. viii) highlights the importance of linking "the nature of computational thinking (with) its cognitive and educational implications." The Royal Society (2012) shares similar emphasis on the importance of linking computing with natural and artificial systems and processes around us.

Hence, Grover and Pea (2013) suggest refocusing on two aspects in relation to real-life application: big ideas and abstraction. Big ideas mooted by the National Science Foundation and the US College Board reframes computing practice as a creative human activity, which results in computational and hopefully innovative artefacts. Abstraction involves "identifying patterns and generalizing from specific instances." It is crucial as it enables simplification from big data to rules, enabling easier application and transfer of learning, which may lead to the development of creative and innovative artefacts.

More importantly, since most communities and professional bodies have agreed on CT definitions and recommended tools, Grover and Pea (2013) suggest that we should venture towards empirical/data-driven inquiries and assessments.

[a] https://orcid.org/0000-0002-4703-457X
[b] https://orcid.org/0000-0002-7914-1978

Lee, C. and Jiang, B.
Assessment of Computational Thinking (CT) in Scratch Fractal Projects: Towards CT-HCI Scaffolds for Analogical-fractal Thinking.
DOI: 10.5220/0007755401920199
In *Proceedings of the 11th International Conference on Computer Supported Education (CSEDU 2019)*, pages 192-199
ISBN: 978-989-758-367-4

However, different assessments of CT provide additional types of scaffolding. Hence, multiple types of assessments are necessary to assess different CT aspects to form a big picture of the student model.

For instance, Holbert and Wilensky's (2011) NetLogo-enhanced FormulaT evaluates results through *modelling and simulation* to further engage students in CT. Grover (2011) however, evaluates the development in students' use of computer science *vocabulary and language*. On the other hand, Werner, Denner, Campe and Kawamoto's (2012) Fairy assessment in *Alice* evaluates in terms of *understanding, use of abstraction, conditional logic and algorithmic thinking*. Similarly, Aho's (2012) assessment involves *formulation of problems* leading to solutions represented by computational steps and algorithms.

Other aspects of interest highlighted by Grover and Pea (2013) involve determining grade and age-appropriate CT curricula, socio-cultural (situated, distributed and embodied cognition, interaction and discourse analyses), and cognitive learning (thinking skills, debugging, transfer, scaffolding), CT as a medium to teach other subjects, and development of learner identity/dispositions/ attitudes.

1.1 Objectives

We situate the study within PwC's (PwC, 2018) four worlds aimed at balancing each other (and corresponding prediction of work in 2030) and Cha's (2017) Restorative Innovation Framework. Another aspect of interest involves McCaffrey and Spector's (2012) obscure feature hypothesis framework to trigger/ prompt more interesting analogies.

The objective for this paper is to assess CT skills in Scratch fractal projects obtained from the Scratch website. More specifically, we aim to identify correlations between the respective scores for each project's concepts corresponding to the respective CT total score. Furthermore, since learning often involves human-computer interaction (HCI), we are interested to identify HCI design factors, which would scaffold the development of such computational-fractal thinking.

We are interested in Scratch fractal projects because fractals embody Wing's (2006) CT skills (pattern recognition and recursion) while fractal thinking builds on these to emphasize on abstraction and novelty transferable to real-life applications. Furthermore, if knowledge tracing such as by Jiang, Ye and Zhang (2018) can be carried out, intelligent agents can be designed to provide more personalized learning.

Furthermore, the findings from this study serve to inform another study in a Facebook-based learning environment called FunPlayCode (Lee & Ooi, 2019). Building on Lee (2010) and Lee and Wong (2015; 2017; 2018), Lee and Ooi (2019) hypothesize that curation of content can encourage knowledge sharing but with analogical transfers as a task requirement. In addition, FunPlayCode involves analogical-computational thinking in blocks of stories (videos/graphics/codes), similar to a certain extent, to block-chain building blocks. Hence, the level of abstraction and CT skills involved are higher.

Perceptions from experts on FunPlayCode's feasibility have been sought and findings indicate that Resnick's (2016) propositions for user-friendliness, attractiveness, inclusivity, and multiple paths need to be further looked into. Sample analogical derivatives are presented in Figs. 1a, b. An example of the coding-storytelling we are emulating in future study is Gibbs' (2018) codes illustrated in Fig. 2.

Figure 1: Sample analogical derivatives: (a) Minoru Mikaya's music and (b) expressions of connectivity.

```
public static void main(String[] args) {
  final double panic_rate = .02;
  int minute;
  minute = 0;
  double total_panic = 0;
  while(total_panic <= 1) {
   total_panic = panic_rate * minute;
   minute++;
  }
  System.out.println("Panic hits 100% after " + minute + " minutes.");
```

Figure 2: Gibbs' analogical expression of panic in code form.

Subsequently, the coding-storytelling-analogical thinking-computational thinking initiative switched

from the learning of Java to Python as Python is more popular among higher education students. Findings from Scratch and Scratch fractal projects, will inform the design and development of tools or techniques to improve CT concepts, practice and skills, in the process of learning Python (FunPlayCodev2) and subsequently, FunPlayCodev3 aided by agent-based interactions. Development of FunPlayCodev2 is expected to start by early to mid-2019.

2 RELATED WORK

Grover and Pea (2013) suggest multiple forms of assessments, depending on contexts and objectives. We agree as different learners have different aptitudes, dispositions and interests. We need to bring out students' strengths, not highlight their weaknesses.

In this study, we consider CT assessment via Dr. Scratch, design artefacts and implications to the design of human-computer interaction. As such, in the following subsections, we review Dr. Scratch and HCI principles which we use to assess the design artefacts.

2.1 Dr. Scratch

Moreno-León, Robles and Román-González's (2015) Dr. Scratch, a free/open-source web tool, analyses Scratch projects automatically. It assigns a CT score and detects potential errors or bad programming habits (debugging). It is chosen due to its close relation to Wing's (2006) definitions.

Students are categorized into three categories of CT competencies based on the seven CT concepts in the first column (Table 1). Furthermore, in terms of data representation, we note that there is a progression from modifying properties to creating variables to lists. Operations on lists are hints of the development of refactoring. In addition, rule-based statements are indicative of higher levels of competency because rules are generalizations. Since CT assessments can be in terms of coding, fractals and many other forms, and learning progression is often fuzzy, we propose participatory design with peers and/or students in defining CT concepts to suit one's objectives, tasks and assessments.

Table 1: Moreno-León, Robles and Román-González's(2015) three categories of CT competencies.

CT Concept	Competence Level			
	Null (0)	Basic (1 point)	Developing (2 points)	Proficiency (3 points)
Abstraction and problem decomposition	-	More than one script and more than one sprite	Definition of blocks	Use of clones
Parallelism	-	Two scripts on green flag	Two scripts on key pressed, two scripts on sprite clicked on the same sprite	Two scripts on when I receive message, create clone, two scripts when %s is > %s, two scripts on when backdrop change to
Logical thinking	-	If	If else	Logic operations
Synchronization	-	Wait	Broadcast, when I receive message, stop all, stop program, stop programs sprite	Wait until, when backdrop change to, broadcast and wait
Flow control	-	Sequence of blocks	Repeat, forever	Repeat until
User Interactivity	-	Green flag	Key pressed, sprite clicked, ask and wait, mouse blocks	When %s is >%s, video, audio
Data representation	-	Modifiers of sprites properties	Operations on variables	Operations on lists

2.2 HCI Principles

The design of user interface and interaction influences the degree of cognitive access/the learning curve. We next review standard user interface and interaction design principles. Table 2 presents three standard human-computer interaction principles, encompassing the design of user interfaces to interaction design and implicitly, user experience.

Both Nielsen (1995) and Shneiderman (1998) focus on user interface design, towards effective human-computer interaction. Norman (1998) regards the gulf of execution and the gulf of evaluation as the two main problems in user interface/user interaction design. He applies cognitive science and usability engineering to identify 6 concise principles.

Preece, Rogers and Sharp (2002) concur and highlight the importance of going beyond user interface and user interaction to a heavier and more explicit emphasis on user experience. Past research in the creative industries (Lee & Wong, 2015) and computing and information systems (Lee & Wong, 2017) concur on the crucial role of problem formulation, user interaction and user experience among students in two Malaysian universities.

Table 2: Three standard HCI standards/principles.

Nielsen's (1995) 10 Usability Heuristics for *User Interface* Design	Shneiderman's (1998) 8 Golden Rules of *User Interface* Design	Norman's (1998) 6 principles for *interaction design*/good UX
1. Visibility of system status 2. Match between system and the real world 3. User control and freedom 4. Consistency and standards 5. Error prevention 6. Recognition rather than recall 7. Flexibility and efficiency of use 8. Aesthetic and minimalist design 9. Help users recognize, diagnose, and recover from errors 10. Help and documentation	1. Strive for consistency 2. Enable frequent users to use shortcuts. 3. Offer informative feedback. 4. Design dialogue to yield closure. 5. Offer simple error handling. 6. Permit easy reversal of actions. 7. Support internal locus of control. 8. Reduce short-term memory load.	1. Visibility 2. Feedback 3. Affordance 4. Mapping 5. Constraints 6. Consistency

3 METHODOLOGY

The public dataset by Hill & Monroy-Hernández (2017) and the TUDelftScratchLab's Scratch dataset from GitHub are downloaded. The data set consists of 25 fractal projects. Different types of fractal tools have been used such as Mandelbrot fractal tree and Fractal pen.

For analysis, a mixed method approach is adopted: First, for quantitative analysis, the total scores for each project based on Dr. Scratch's CT metrics were calculated and then sorted from lowest to highest. Subsequently, the total scores for each CT concept were calculated to identify which CT concept contributes more to the total CT score. Scores lower than 30, are regarded as non-contributing CT factors.

Second, qualitative evaluation is carried out by considering the type of instructions/explanations students included into their fractal projects.

Since Scratch is a public website, we present only the higher scoring projects with no identifying information or images or examples. Permission to use the data has been obtained from Hill and Monroy-Hernandez (2017). TUDelftScratchLab's Scratch dataset is from GitHub, an open-source site.

4 FINDINGS

We present our findings based on the following outline: Concepts contributing to higher CT scores (quantitative analysis), influence of tool to remixing/transformation, type of instructions/ explanation (qualitative analysis), and finally preliminary insights.

4.1 CT Concepts Contributing to Higher CT Scores (Quantitative Analysis)

Moreno-León, Robles and Román-González's (2015) evaluation matrix presents individual projects in rows and CT concepts in columns. In terms of CT concepts, Fig. 3 shows that the two highest total scores for all projects (not just the high scoring projects) were for flow control (46) and data representation (45), followed by logic (42) and abstraction (41). Flow control and data representation correspond with algorithm and data structures in programming. Logic and abstraction correspond with relational and object-oriented schema and processes. Thus, these CT concepts are suitable predictors.

In terms of individual project scores, the highest scoring projects (project ID 21-25) scored as follows in descending order (Fig. 3):

a) abstraction (score of 3), followed by
b) logic (score of 3), data representation (score of 3), and mastery (score of 3), followed by
c) flow control (score of 2) and user interactivity (score of 2).

We thus predict that for all students, when logic and abstraction increase, the CT score for mastery will also increase. These high scoring projects used the fractal tool (fractal pen) and the design artefacts are artistic and/or game-like. They were also able to include instructions on how to play and explain what they hope to achieve. The best scoring project also added in a specific music to create a certain mood. The sense of pride/accomplishment was clear and encouraging. A summary of the findings is presented in Table 3.

	A	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1		total-view	total-remi	total-favo	total-love	is-remix		Abstractic	Parallelisr	Logic	Synchroni	FlowConti	UserInter	DataRepre	Mastery	Clones	CustomBl	Instances	Total
2	1	1	0	0	0	0		0	0	0	0	2	1	1	4	0	0	0	8
3	2	3	0	1	0	0		0	0	0	1	2	1	1	5	0	0	0	10
4	3	1	0	1	1	0		0	0	0	1	2	2	1	6	0	0	0	12
5	4	1	0	0	0	0		1	1	0	0	2	1	1	6	1	0	0	13
6	5	1	0	0	0	0		0	0	1	0	3	1	2	7	0	0	0	14
7	6	11	0	2	2	0		2	0	2	0	1	1	1	7	0	1	0	15
8	7	9	0	2	2	0		2	0	2	0	1	1	1	7	0	1	0	15
9	8	13	0	2	1	0		0	0	2	0	3	1	2	8	0	0	0	16
10	9	1	0	0	0	0		0	0	2	0	3	1	2	8	0	0	0	16
11	10	3	0	0	0	0		2	0	1	0	1	2	2	8	0	1	0	17
12	11	3	0	0	0	0		2	0	1	0	1	2	2	8	0	1	0	17
13	12	3	0	1	1	0		2	0	2	0	2	1	1	8	0	1	0	17
14	13	1	0	0	0	0		2	0	2	0	1	2	1	8	0	1	0	17
15	14	2	0	0	0	0		1	1	0	1	2	2	2	9	0	0	0	18
16	15	2	0	0	0	0		2	0	2	0	1	2	2	9	0	1	0	19
17	16	4	0	0	0	0		2	0	1	1	1	2	2	9	0	1	0	19
18	17	2	0	2	2	0		2	0	2	0	2	2	2	10	0	3	0	23
19	18	1	0	0	0	0		2	1	3	1	2	1	2	12	0	1	0	25
20	19	1	0	0	0	0		2	1	2	2	2	1	1	11	1	5	0	28
21	20	21	0	8	2	0		2	0	3	3	2	2	2	14	0	1	0	29
22	21	10	0	0	0	0		3	3	2	2	2	1	2	15	0	1	1	32
23	22	1	0	0	0	0		3	1	3	1	2	2	3	15	0	1	1	32
24	23	5	1	1	1	0		3	1	3	1	2	2	3	15	0	1	1	32
25	24	2	0	0	0	0		3	1	3	1	2	2	3	15	0	1	1	32
26	25	1	0	0	0	1		3	1	3	1	2	2	3	15	0	1	1	32
27		103	1	20	12	1		41	11	42	16	46	38	45	239	2	23	5	508
28								8%	2%	8%	3%	9%	7%	9%	47%	0%	5%	1%	100%
29																			

Figure 3: CT scores for each Scratch fractal project.

Table 3: Three standard HCI standards/principles.

Overall group	Highest scoring groups	Predictors (difference between overall group and highest scoring groups)
• flow control (46) • data representation (45) • logic (42) • abstraction (41).	• abstraction (score of 3), • logic (score of 3), • data representation (score of 3), • mastery (score of 3), • flow control (score of 2) • user interactivity (score of 2).	• when logic & abstraction increase, the CT score for mastery will also increase; contributing to weighted criteria analysis; • added in music, have a different perspective e.g. from artistic view. • included instructions on how to play and explain what they hope to achieve.

4.2 Instantiation/Cloning/ Remixing/ Transformation

Scratch encourages curating, connecting and co-designing/co-appropriating. Fractal thinking, best exemplified in Scratch, focuses on novelty and abstraction, remixing and transformation. Remixing in Scratch 3.0 comes from the possible change of backdrop, sprites, extensions and the introduction of a surprise object (Fig. 4). The surprise object can trigger/prompt novel ideas, remixes and transformation.

For the Scratch fractal dataset, students used sprite instances a total of 5 times but are a bit apprehensive when it comes to cloning, with a total score of 2. Since the students are using fractal tools, the low scores for these two CT concepts are understandable. This might be due to certain perceived connotations towards cloning as copying, i.e., of not being original.

For fractals, transformation is more natural and makes more sense than remixing. Remixing in fractals based on the tools used may come at a later stage, depending on what the tool allows and/or as a means to or a form of transformation.

4.3 Type of Instructions/Explanation and Outcomes (Qualitative Analysis)

We also note that some projects received lesser views and likes. We conjecture that perhaps students did not see the relevance of learning fractals or its relevance to the real-world. This is true for many unless they are Mathematicians or designers/artists/ industrial

engineers. We think, with time and better tools, and scaffolds linking fractals and fractal thinking to their curriculum and the real-world, appreciation of its nature and possible use would increase.

We see this possibility in the projects themselves. Although the better projects initially had minimal views, the number of views and likes for these projects increased over time. One project increased from 11 views to 16 but another increased from 21 views to 129, almost a six-fold increase (Fig. 5).

One possible difference was due to the former not providing instructions and/or explanations whereas the latter instructed on speed, how to restart, trying to be realistic and specifying a particular music as a finishing touch. In short, communicating with the audience could be a possible reason for more views and more likes.

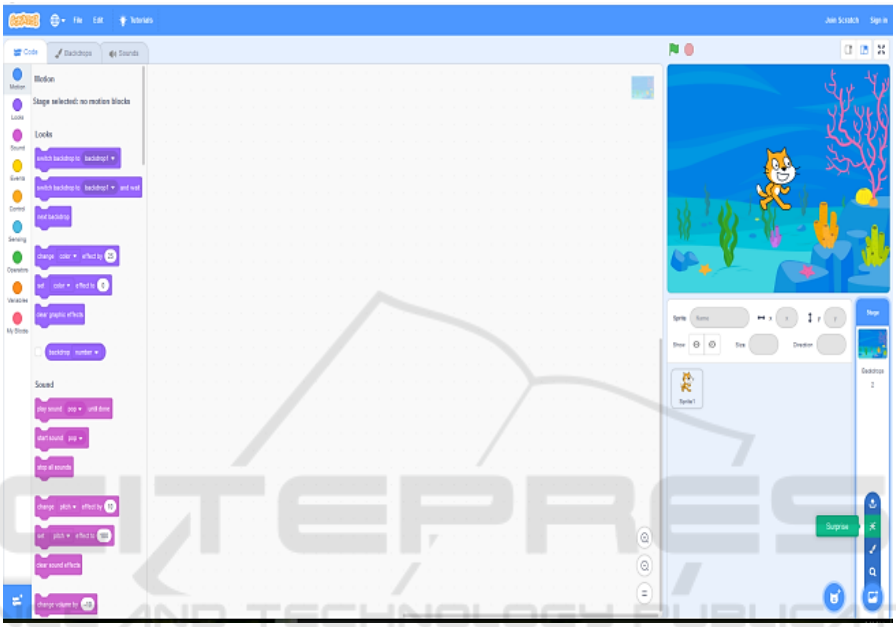

Figure 4: Scratch 3.0's latest layout, palette, stage and additions (change backdrop, surprise, add extensions).

	D	E	F	G	H	I
1	Project	total-views	total-rem	total-favo	total-love	is-remix
2	1	2	0	0	0	0
3	2	13	0	2	1	0
4	3	11	0	2	2	0
5	4	1	0	0	0	0
6	5	4	0	0	0	0
7	6	3	0	0	0	0
8	7	3	0	0	0	0
9	8	3	0	1	1	0
10	9	9	0	2	2	0
11	10	10	0	0	0	0
12	11	1	0	0	0	0
13	12	1	0	0	0	0
14	13	1	0	0	0	0
15	14	1	0	0	0	0
16	15	2	0	2	2	0
17	16	1	0	0	0	0
18	17	1	0	0	0	0
19	18	5	1	1	1	0
20	19	3	0	1	0	0
21	20	2	0	0	0	0
22	21	21	0	8	2	0
23	22	1	0	0	0	1
24	23	1	0	0	0	0
25	24	1	0	1	1	0
26	25	2	0	0	0	0
27						

Figure 5: Number of views, likes, remix.

5 INSIGHTS AND SIGNIFICANCE

CT as a medium to teach other subjects and development of learner identity/dispositions/ attitudes has often been emphasized in many national policies. At a macro level, socio-cultural considerations (situated, distributed and embodied cognition, interaction and discourse analyses) as well as cognitive learning (thinking skills, debugging, transfer) are important aspects in determining grade and age-appropriate CT curricula.

In terms of tools, Resnick (2005), Grover and Pea (2013) and many CT researchers have rightly pointed out, it is important to design tools for users of different competencies based on multi-pronged CT assessments to triangulate data as pointed out by successful tools reviewed above. What are the scaffolds and affordances which we need to build into the tool?

Fractals developed by the students were afforded by the respective fractal generating tools. This corresponded with Norman's (1998) gulf of execution and the gulf of evaluation. Scratch has pointed out that it is important to scaffold these tools, similar to standard programming tools, i.e., with libraries, templates, and sample scenarios or stories. Hence, it would be interesting to develop these for fractal development too.

In terms of user interface design, all fractal generating tools utilized by the students were minimalist, and matched closely with Nielsen and Shneiderman's principles except for feedback. This creates room for agent-based feedback/interactions, given sufficient prior user interactions to develop the knowledge base.

Significance of the study lies in:

a) confirming the importance of quantitative and qualitative multi-pronged CT assessments;
b) identifying design factors and correlated attributes for higher computational thinking scores, which may correspond very well with fractal thinking. These CT design factors can be used as guides to better predict a student's CT performance/mastery and CT areas and strategies to help him/her to improve further. These predictors may also lead to more informed design of agent-based feedback such as prior work on knowledge tracing (Jiang, Ye, & Zhang, 2018). Given sufficient datasets, to better understand students' learning and to provide more agent-assisted help or surprises to trigger context-aware associations or reminders.
c) connecting CT assessment with HCI design by highlighting the need to not only have clear understanding of the objectives, skills, contexts, tasks and assessments but also how the tool can better scaffold modelling and simulation, formulation of problems. Future studies may look into the connections between HCI design and the use of computer science vocabulary and language, understanding, use of abstraction, conditional logic and algorithmic thinking as reviewed in Section 1 above.
d) Since findings contribute to FunPlayCode, where analogical-fractal thinking may become suitable playgrounds for simulating various analogies and insights, there are possibilities of deriving clearer links between CT, HCI, and fractals/fractal thinking/analogical thinking.
e) Hopefully, with seed data and user interaction, more useful scaffolding can be introduced, aimed at the crux of CT: solving problems, designing systems and understanding human behaviour.

5.1 Limitations of the Study

Different forms of fractal tools are used. We are not sure whether the students who submitted the projects are exposed to only one fractal tool or diverse tools. Second, since the submissions are just for fun and not for grades, we would like to qualify our findings as very preliminary, as working towards a better understanding of the link between CT, HCI and fractals/fractal thinking.

The learning curve is naturally difficult for anyone. We do not judge. Our aim is only to identify CT concepts, which correlate higher with CT and to use these as guides to develop more effective learning environments and learning.

6 CONCLUSIONS

In conclusion, transformation and remixing are a part of CT and a part of research practice. To broaden students' learning, from generating fractals to designing for fractal thinking and to communicate form and function as the better performing students did with the pen fractals, multiple CT aspects need to be considered. We have touched on only the tip of the iceberg. Though CT is implicit in fractal development and fractal thinking, we hope that students would think deeper about computational practice and graduate progressively towards firmer mastery of CT concepts and better research skills.

ACKNOWLEDGEMENT

This project came about due to prior works by both authors in educational data mining and information systems, the first author in student modelling and creativity and the second author in Engineering and knowledge tracing. We wish to thank Zhejiang University of Technology, China, Sunway University, Malaysia, Universiti Tunku Abdul Rahman, Malaysia, and the Learning Sciences (especially Georgia Tech), Multimedia University, Malaysia, and Dr. K. Daniel Wong for past invaluable fundamental groundings. This work is partly supported by National Nature Science Foundation of China No.71704160.

REFERENCES

Aho, A. V. 2012. Computation and computational thinking. *Computer Journal, 55*, 832–835.

Cha, V. 2017. *Restorative Innovation Framework*. Online: https://www.restorativeinnovation.com/

Gibbs, M. https://study.com/academy/lesson/while-loops-in-java-example-syntax.html, last accessed 2018/07/29

Grover, S. 2011. Robotics and engineering for middle and high school students to develop computational thinking. Paper presented at the annual meeting of the *American Educational Research Association*, New Orleans, LA.

Grover, S., Pea, R. 2013. Computational Thinking in K–12. A Review of the State of the Field. *Educational Researcher, 42*(1), 38-43.

Hill, B. M. Monroy-Hernández, A. 2017. A longitudinal dataset of five years of public activity in the Scratch online community. *Sci. Data 4:*170002.

Holbert, N. R., Wilensky, U. 2011. Racing games for exploring kinematics: a computational thinking approach. Paper presented at the educational Researcher annual meeting of the *American Educational Research Association*, New Orleans, LA.

Jiang, B., Ye, Y., Zhang, H. 2018. Knowledge tracing within single programming exercise using process data. *International Conference on Computers in Education*. November 26-30, 2018, Manila, Philippines.

Lee C. S. 2010. Towards creative reasoning: Scaffolding systems thinking and decision-making. *International Conference on Computers in Education*, Kuala Lumpur, Malaysia, November 29-December 3, 2010, pp. 655-662.

Lee, C. S., Ooi, E. H. 2019. Identifying design factors to encourage reframing, reuse through granular coding-analogical thinking-storytelling. *International Conference on Engineering Technology*, July 6-7, 2019, Malaysia.

Lee, C. S., Wong K. D. 2015. Developing a disposition for social innovations: An affective-socio-cognitive co-design model. *International Conference on Cognition and Exploratory Learning in Digital Age*, October 24-26, 2015, Ireland, 180-186.

Lee, C. S., Wong K. D. 2017. An entrepreneurial narrative media-model framework for knowledge building and open co-design. *SAI Computing*, July 18-20, 2017, London, UK, 1169 - 1175.

Lee, C. S., Wong, K. D. 2017. Design - computational thinking, transfer and flavours of reuse: Scaffolds to Information and Data Science for sustainable systems in Smart Cities. *IEEE International Conference on Information Reuse and Integration*, IEEE Computer Society, Salt Lake City, Utah, July 7-9, 2018, pp. 225-228.

McCaffrey, T., Spector, L. 2012. Behind every innovative solution lies an obscure feature. *Knowledge Management & E-Learning. 4* (2).

Moreno-León, J., Robles, G. and Román-González, M. 2015. Dr. Scratch: Automatic Analysis of Scratch Projects to Assess and Foster Computational Thinking. *RED. Revista de Educación a Distancia*, 46, 1-23.

National Research Council. 2010. *Committee for the Workshops on Computational Thinking: Report of a workshop on the scope and nature of computational thinking*. Washington, DC: National Academies Press.

Nielsen, J. 1995. 10 Usability Heuristics for User Interface Design. Online:https://www.nngroup.com/articles/ten-usability-heuristics/

Norman, D. 1988. *The Design of Everyday Things.* New York: Basic Books.

Preece, J., Rogers, Y., and Sharp, H. 2002. *Interaction Design: Beyond Human-Computer Interaction*. New York: John Wiley & Sons.

PwC. 2018. Workforce of the future. *The competing forces shaping 2030. http://www.pwc.com/people*.

Resnick, M. 2005, 2016. Designing for Wide Walls. https://design.blog/2016/08/25/mitchel-resnick-designing-for-wide-walls/

Royal Society. 2012. *Shut down or restart: The way forward for computing in UK schools.* Retrieved from http://royalsociety.org/education/policy/computing-in-schools/report/

Shneiderman, B. 1998. *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* (3rd ed. ed.). Menlo Park, CA: Addison Wesley.

TUDelftScratchLab's Scratch dataset. 2017. A Dataset of Scratch Programs: Scraped, Shaped and Scored. *MSR Data Showcase*. GitHub.

Werner, L, Denner, J., Campe, S., and Kawamoto, D. C. 2012. The Fairy performance assessment: Measuring computational thinking in middle school. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education (SIGCSE '12)*, 215-220. New York.

Wing, J. 2006. Computational thinking. *Communications of the ACM, 49*(3), 33–36.