An online platform for teaching upper secondary school computer science

Conference Paper · September 2021

DOI: 10.1145/3481282.3481287

Jane Waite
Queen Mary, University of London
23 PUBLICATIONS **368** CITATIONS

SEE PROFILE

Andrea Franceschini
University of Padova
8 PUBLICATIONS **8** CITATIONS

SEE PROFILE

Sue Sentance
King's College London
73 PUBLICATIONS **1,649** CITATIONS

SEE PROFILE

An online platform for teaching upper secondary school computer science (Authors' pre-print version)

Jane Waite
The Raspberry Pi Foundation
Cambridge, United Kingdom
jane.waite@raspberrypi.org

Sue Sentance
The Raspberry Pi Foundation
Cambridge, United Kingdom
sue@raspberrypi.org

Andrea Franceschini
Department of Computer Science and Technology
University of Cambridge
Cambridge, United Kingdom
andrea.franceschini@cl.cam.ac.uk

Matthew Patterson
Department of Computer Science and Technology
University of Cambridge
mbp36@cam.ac.uk

James Sharkey
Department of Computer Science and Technology
University of Cambridge
james.sharkey@cl.cam.ac.uk

ABSTRACT

The teaching of computing in schools is relatively new, with limited research informing what to teach and how in upper secondary contexts. However, computing education has spawned the development of many tools for use in such education settings.

Isaac Computer Science is a computer science (CS) learning platform aimed at school students in England aged 16 to 19 years old studying for formal A level CS qualifications. Over 34,000 students and over 2,400 teachers have registered on the platform to date, and over 1 million online questions have been attempted. The platform is pre-populated with CS content and questions. Feedback is tailored to respond to common mistakes. Hints and explanation videos accompany questions. Question sets can be assigned to students by teachers. Question types include Parsons problems, drag and drop, multiple-choice and text-matching answers, including Boolean Algebra responses. Students only see content, questions and notation pertinent to their course of study. Isaac CS has a centrally-organised ongoing provision of support, such as teacher professional development and student events.

This tools design paper outlines the development of Isaac CS through a review of design decisions and the effectiveness of its features. The review is informed by literature, platform usage data and teacher and student feedback. The discussion is framed in terms of online learning theories and a knowledge appropriation model. We suggest a new model, a Platform Pedagogy Matrix, which may be of use to other platform developers and researchers.

CCS CONCEPTS

• Social and professional topics → K-12 education; **Computer science education**; **• Applied computing → Computer-assisted instruction**.

KEYWORDS

computer science education, blended learning, platform

ACM Reference Format:
Jane Waite, Sue Sentance, Andrea Franceschini, Matthew Patterson, and James Sharkey. 2021. An online platform for teaching upper secondary school computer science (Authors' pre-print version). In *Proceedings of PLEASE SEE - United Kingdom and Ireland Computing Education Research conference. (UKICER '21).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

This tools design paper describes the computing education platform Isaac Computer Science (CS). Isaac CS is a free educational resource for students studying upper secondary school computer science (aged 16 to 19 years old) and their teachers. As a holistic offer, the resource comprises a content-rich educational website supported by centrally coordinated face-to-face student and teacher events, online student mentoring, and printed materials. Online resources include coverage of England's upper secondary A level[1] CS subject material and smart resources, such as embedded interactive questions with hints and tailored feedback, a tool for students to write Boolean algebra, and a Parsons problems question feature. The platform also incorporates more traditional online functionality of text, images, animations, videos and multiple-choice questions.

Although students could independently follow a suggested order to access Isaac CS material, students are expected to primarily work with the platform as guided by their teacher during class, as homework, or for revision. Teachers incorporate Isaac CS into their teaching. The resource is not intended to be a MOOC or an "empty" Learning Management System (LMS), but rather offers a solution that includes all the necessary subject content without prescribing

[1]A level is an examination-based 2 year course delivered in England and some other countries of the UK

how it is embedded into the teaching experience, with the aim of providing support to learners and reducing teacher workload.

Although existing A level teachers are likely to have more extensive subject knowledge than teachers of younger students, there is a shortage of experienced upper secondary teachers [39]. A contributing factor to the shortfall of teachers is that the number of students enrolling in CS A level courses in England has more than doubled from 4925 in 2015 to 10375 students in 2019 [13].

Isaac CS has been developed to address two main issues: firstly, the limited resources with curated, trusted online content and integrated activities to support the teaching and learning of A level CS in England [34, 37], and secondly, the shortfall of experienced upper secondary school CS teachers in England [39].

Isaac CS is a government-funded project created and run by the University of Cambridge and the Raspberry Pi Foundation. Isaac CS was launched in July 2019 and is free for anyone to use. Isaac CS teaching material is available under the Open Government Licence v3.0, which permits anyone to use and adapt the work [38], and the platform's software is open source.

In reviewing the design of Isaac CS, we have developed a model, the Platform Pedagogy Matrix, that relates the pedagogy that an online CS platform may afford.

2 LITERATURE REVIEW

To situate our study, we first outline research on online learning in general and then focus on research on education platforms designed to support blended learning in the teaching of computing.

Meta reviews of research on online education reveal a lack of robust evidence for, and often conflicting views of, the effectiveness of online educational platforms [3, 16, 26, 29, 30]. Despite this, online platforms are used extensively in undergraduate courses and in some school settings [3, 26]. The common use of online resources, and a scarcity of robust evidence on their effectiveness to improve student outcomes, has led to advice to teachers that they should take much care when designing online learning activities, including thinking carefully about pedagogy, implementing change, and considering what is most useful for different phases of education, and different subject disciplines [9, 16, 24, 26]. Guidance has emerged as to why online resources are used [26], and on recommended uses [16] and processes to design blended teaching and learning, such as the Arena Blended Connected process (ABC) [42]. In their 2014 review of research on online learning, Means et al. [26] suggested six reasons why K-12 teachers turn to online resources to augment their classroom activities:

> *" broadening access to instruction, facilitating small-group and one-to-one teacher-led instruction, serving students with very diverse needs, providing more opportunity for productive practice, adding variety to instruction and enhancing student engagement and supporting learning of complex, abstract concepts." [26, p.101]*

In their 2019 review of the literature on the use of digital technology in schools, the Education Endowment Fund recommended three potential uses of online tools: i) that this technology can be used to improve the quality of explanations and modelling, ii) that it can offer ways to improve the impact of pupil practice, and iii) that it can play a role in improving assessment and feedback [16].

In online learning, the term "blended" is used to describe one learning scenario. Horn and Staker define the term as follows:

> *" Blended learning is any time a student learns at least in part at a supervised brick-and-mortar location away from home and at least in part through online delivery with some element of student control over time, place, path, and/or pace." [20, p.3]*

To transition from face-to-face to blended teaching, several universities use an online curriculum design process called Arena Blended Connected process (ABC) [42]. In the ABC process, educators review face-to-face learning activities and identify what student learning types are used, and design new sequences of blended teaching events that provide a balanced coverage of all learning types [24, 42]. Based on the conversational framework proposed by Laurillard [24], the ABC learning types are acquisition, investigation, discussion, collaboration, practice and production [42].

As well as research on how to design blended and online instructional activities in general, research looking at the use of online resources to support the teaching of CS has been undertaken. In their 2008 review of how to better support university CS education through learning management systems (LMS), Rößling et al. suggested a Computing Augmented Learning Management System (CALMS) [36]. CALMS is not a specific tool or toolset; instead it lists components, functionality, and pedagogy patterns suggested to support and improve CS LMS. Pedagogical patterns can include active learning, cooperative learning, and feedback patterns such as the *feedback loop model* where students use online resources for personal study contributing to in-class experiences [36].

Rößling et al. [36] suggest that functionality to support the online teaching and learning of CS should include:

- general pedagogy tools e.g. tools supporting authoring, collaboration, scaffolding, reflection, multiple representations
- augmented learning e.g. functionality to enable educators to change elements for student requirements
- specialised LMS e.g. systems that include specific adaptions, such as Interactive Development Environments (IDEs), functionality that checks knowledge in specific CS topics and often includes Computer Aided Assessment (CAA)
- algorithm and program visualisation tools

Computer Aided Assessment (CAA) refers to activities where computers are involved in the assessment process beyond storage and delivery [10]. An example of CAA would be a self-marking system with automated feedback that is dependent on answers provided or has questions adapted to the student [10]. CAA, IDE and other LMS are often standalone systems and calls have been made to improve their interoperability [22].

As well as algorithm and program tools that are integrated within an IDE or visualisation software, some tools handle pseudocode or code as text. An example of this are Parsons problems where students reorder blocks of text representing code statements [32]. A rich thread of research, predominately conducted in an undergraduate setting, comparing Parsons problems (and their many variants) to code tracing and other code writing activities, has been explored, with promising indication that students make more progress when using Parsons problems [17, 18]. Research indicates that Parsons problems are a particularly effective teaching approach

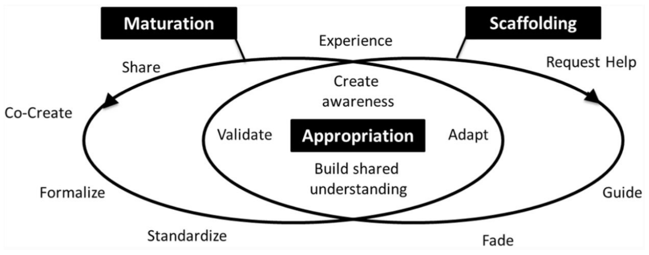


Figure 1: Ley et al.'s Knowledge Appropriation Model [25, p.107]

when dealing with problems with "non-novel" solutions [19], they help identify student difficulties [14], can be used to help students understand programming patterns [41], and increase student engagement [14]. However, such studies require replication to provide confidence about generalised benefits [14].

In their analysis of the use of an undergraduate CS e-book used in a blended course, Ericson et al. found that online activities with lower cognitive load, such as multiple choice questions and Parsons problems, were more popular with students than higher cognitive load activities, such as editing code [17, p.176].

Brusilovsky et al. reviewed university CS teaching and learning platforms and identified online systems as often including eTextbooks with smart content such as interactive examples, animations, and auto-assessment exercises [6]. These authors note that approaches for marking, feedback, and reporting of open ended coding solutions to complex problems are very different to the auto-assessment of multiple choice and other simple format questions. They highlight that, although functionality for learning programming is a common focus for many CS education systems, some systems focus on teaching and learning of abstract concepts [6].

Turning to younger students learning CS with online platforms, Anohah investigated which pedagogy is best to support online teaching of CS for high school students (ages 14 to 18) [2]. The author concluded that no single pedagogy fits all, and that multiple pedagogical approaches should be considered [2]. Nevertheless, Anohah suggested that automatic feedback on programming exercises, visualization of algorithms and representation of concepts in animations or through physical activities are extremely important to consider [2]. Physical activities are not conducted online, rather they are done "offline" and are sometimes called unplugged learning activities [4].

Simply creating a new platform does not guarantee practical adoption by teachers or learners. As shown in Figure 1, Ley et al. have proposed a Knowledge Appropriation Model to help understand the phases of becoming aware of (engagement with), scaffolding use of, and maturing the use of new tools and working practices that lead to appropriation [25]. This model has been used to analyse the introduction of tools in educational settings [35] and the adoption of programming pedagogy in industry [40].

3 AIMS AND APPROACH

The overarching aim of this study is to contribute to the body of research on school-focused teaching and learning CS platforms, specifically on their features and how they relate to pedagogical approaches. Our approach to meet these aims is to: i) reflect on the design decisions of the key features of Isaac CS; ii) review the testing and evaluation of the platform; and iii) using the literature of the field, reflect on lessons learned that emerged during the development of the platform to suggest overarching models to support similar platform development.

4 PLATFORM DESCRIPTION

Rather than developing the platform from scratch, Isaac CS built upon an existing educational platform called Isaac Physics, a learning platform aimed at improving students' physics and associated mathematics problem-solving skills. Isaac Physics has been successful as a unique product that is simple to introduce, low maintenance, trusted, and with carefully considered pedagogy focused on meeting the needs of one group of teachers and their students [15, 21].

Isaac CS benefits from the features of Isaac Physics but for CS communities of learners. Building upon an existing platform reduced development costs and provided a trusted brand that was likely to be known to students and teachers. Issac CS, like Isaac Physics, is a web-based, single-unit application with some internal services, such as checking symbolic maths, but currently has no integration with other software like school LMS. Users sign up individually to Isaac CS, and teachers can view and manually download student assessment data.

The major features of Isaac CS can be grouped into themes of Pedagogy, Content, Appropriation (including engagement) and General Functionality. We describe each in turn and then outline the evaluation of the platform.

Pedagogy - A content-rich platform. A key feature of Isaac CS is that it should support students to learn and acquire the content of the syllabus through presentation of core material in text and video format that students can read and watch as well as practising questions about the content. To make the content more interactive, "quick quiz" activities are used. These activities do not require students to enter an answer; rather the answer is hidden, and students click a button to reveal it.

Pedagogy - Question types for CS.. An early activity in the development of Isaac CS was a review of the question types needed for computer science. The Isaac Physics resource provides multiple choice questions, single value numerical answer questions, and drag and drop algebra questions. Through an analysis of previous A level CS examinations, computer science questions were mapped to the types of available question types, and gaps identified. For example, most CS examination questions required paragraph-length answers and many required students to draw diagrams. As a result, three significant changes to question types were made to accommodate CS requirements: i) The equation editor was adapted to support Boolean Logic drag and drop questions; ii) A new Parsons problems question type was added; iii) Text matching questions using string matching and natural language processing were developed.

As discussed in Section 2, Parsons problems have been identified as being of promise in improving student progress and engagement when learning to program [14, 17, 18]. This format of question was added to the platform, including reordering and indentation of blocks of text. As well as using this question type for code, pseudocode, and algorithm problems, the question type has also been used in topics such as networks, architecture, and ethics.

Long text answers are difficult to mark automatically [6]. However, text matching questions with short answers of one or two words were added to the platform by developing a new string matching and natural language processing question type. The text matching functionality built upon and adapted OpenMark, a free-text auto-marking system [7] developed at The Open University. To use this question type, content designers must define rules related to expected answers.

Pedagogy - Automated scaffolding through episode-related hints which students choose to see or not. Before students attempt to answer a question, they can choose to reveal a hint or not. As students become more proficient, they can choose to fade the scaffolding provided by the hint. There are levels of hints, providing help with the content needed (declarative knowledge) and how to tackle the problem (procedural knowledge). Research into the hint facility within Isaac Physics indicated episode-related self-selected hints enhanced problem-solving skills [11, 33].

Pedagogy - A practice pedagogy of student's multiple tries to answer questions (as many times as they like) and automated tailored feedback. When attempting questions, if the student gets the answer wrong, feedback is provided instead of the correct answer. If the answer is associated with a common error, then tailored feedback is provided. Students can attempt a question as many times as they like. Being able to practise problem-solving with immediate and relevant feedback scaffolds learning and affords students a form of low-stakes formative self-assessment [5], prompting increased learning [8], self-efficacy, and self regulation [27].

Pedagogy - Blended teaching and learning. Teachers are expected, and are supported through professional development, to blend Isaac activities into their lessons, homework and to recommend it for student independent study. Many questions, as well as being presented on the platform, are also distributed as physical books. The books are sold at cost price to schools and students. It is anticipated that students will work in their books offline. Teachers may mark student's books for assessment purposes, helping teachers to identify misconceptions.

Pedagogy - Providing examination-specific content. Through consultation with teachers, it was decided that a feature should be added to enable students to see only content and questions pertinent to their Awarding Body (AB) in order that students would not be confused by material that was not pertinent to the examinations they were studying for. An AB is an accredited organisation recognised by government-appointed regulatory bodies that sets and accredits examinations [12]. In England, there are numerous ABs, including currently AQA and OCR [31].

Although one might expect the content of the curriculum specifications for an examination to be identical across ABs this is not the case for A level CS in England. Differences include the specification of pseudocode, Boolean Algebra notation, and assembly language commands. By enabling students and teachers to select their preferred AB, the platform presents the appropriate content and notation.

Pedagogy - Provision of an IDE.. An interactive development environment (IDE) for programming activities has been cited as an

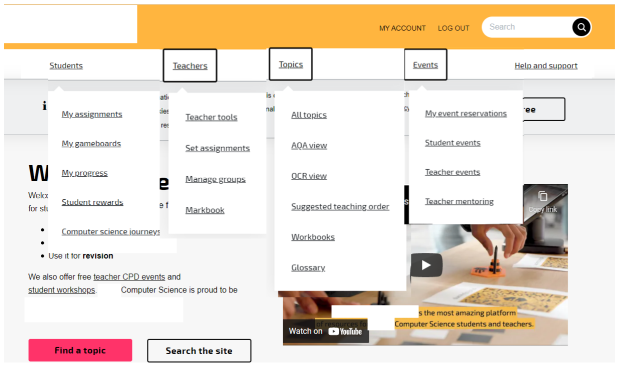


Figure 2: Isaac CS Menu (with menus expanded)

important feature of CS platforms [6, 17, 23]. However, for Isaac CS, it was decided that an integrated IDE was not required at this stage. The rationale for this decision was: i) there were many IDEs already in use by teachers; ii) developing a new IDE was not practical in the timescales and funding provided; iii) programming practice through Parsons problems and other question types was to be provided.

Content. A team of experienced computer science educators is responsible for the development of the content for the platform. Content topics include Computer Networks, Computer Systems, Cybersecurity, Data & information, Data structures & algorithms, Impacts of digital technology, Maths for CS, Programming fundamentals, Programming paradigms, Software Engineering, and Theory of computation. Further content covering other AB specifications is scheduled to be added to the platform over the project's lifetime.

Appropriation (including engagement). To raise awareness about the platform, an Isaac CS team devises and runs marketing campaigns and events such as student masterclasses, teacher professional development, and student and teacher mentoring sessions. To help educators and students learn how to use and adapt the platform, to appropriate it to their teaching and learning, the platform is used in events. Student events include industry sessions relating to careers and university-led sessions on CS topics and higher education opportunities.

General Functionality. To adapt Isaac Physics, a CS-specific user interface was designed for Isaac CS; the menu structure for Isaac CS is shown in Figure 2. Isaac CS is a standalone platform accessed through a web interface. The sign-up process has been designed to be very simple and easy to use. Teachers create student groups and invite students to sign up using a QR code or an invitation link. Students must choose whether or not they want to share their data about questions completed with their teacher. In other words, students can choose to use the platform and not share their data. Individuals signed up to Isaac CS can request a teacher account. The upgrade is verified by the platform team using a manual process. All users on the site (students, teachers and platform developers) may assemble "gameboards". A gameboard is a set of up to 10 questions selected from existing questions on the platform. To help students monitor their progress, a "progress page" is provided showing a breakdown by topic of the total number of questions they attempted and answered correctly. To support teaching and assessment, teachers are provided with a summary page, called a "markbook". The markbook shows the number of correct and incorrect submitted answers per student per question assigned. This page does not show the number of attempts or details on the students' incorrect answers.

5 TESTING AND EVALUATION

The platform has been tested and evaluated through numerous processes that can be categorised into four themes: Functionality testing, Content testing, Pedagogy testing and Platform evaluation covering functionality, content, pedagogy and appropriation.

Functionality testing. The new user interface was tested during an iterative process of prototyping and testing with a user group of nine students and three teachers. In the testing period, participants completed realistic tasks such as setting and completing homework, tracking progress, and finding curriculum content. Changes to the Isaac Physics platform such as the AB specific functionality, handling Boolean Algebra, new Parsons problem questions and text matching questions were tested by the platform team. Manual and automated tests were performed, as well as regression testing.

Content testing. Before going live, each unit of content follows a quality assurance process to ensure accuracy and trustworthiness. The process includes an internal and external review, pilot of content with teachers, and documented incorporation of feedback. Each unit of content is then reviewed annually. In addition users can report content and technical issues. Issues raised by users are recorded in a ticketing system that is regularly monitored by the platform team.

Pedagogy testing. It is difficult to test how teachers and students use the platform and how content and questions support knowledge building and overcoming misconceptions. One indication of what learning is taking place is provided by students' incorrect answers. Teachers can view the results of students' attempts of assigned gameboards, but they can only see how many answers were correct or incorrect. The Isaac CS team receives reports detailing what incorrect responses were given, and the number of such attempts. The team uses this information to reflect on potential misconceptions and improve content, questions, hints and tailored feedback. For example, a misconception related to queue abstract data types and dequeued elements was discovered. To help address this misconception, a tailored feedback response was added, and content changes made.

Platform evaluation including functionality, content, pedagogy and appropriation. Evaluation of the platform is obtained in three main forms: from an advisory group of teachers, through student and teacher surveys (of events and the platform in general), and from the review of platform usage statistics.

To support the development of the platform, a panel of representative teachers was appointed, meeting four times a year. Before each meeting, the practitioners complete a survey to capture issues and ideas for improvement of the content and pedagogy of Isaac CS. An example of feedback from the panel was that static pictures and text may not help students to learn about procedural aspects of topics such as sorting and searching algorithms. Teachers suggested the inclusion of an animation or video that steps students through each stage; as a result of this, animations are being added.

Surveys are used to capture platform data. 94 teachers and 253 students responded to the 2020 annual survey. Reporting about the benefits of the platform, 64% of the teachers said that using the Isaac CS helped them save an average of 3 hours teacher workload hours per week [28]. Also, in the 2020 annual survey, teachers and students were asked about the platform and quality of the content. 94% of teachers and students rated the platform as easy to use. 88% of teachers and 84% of students rated the platform as having good quality written material. 90% of teachers and 79% of students rated the platform with good quality questions.

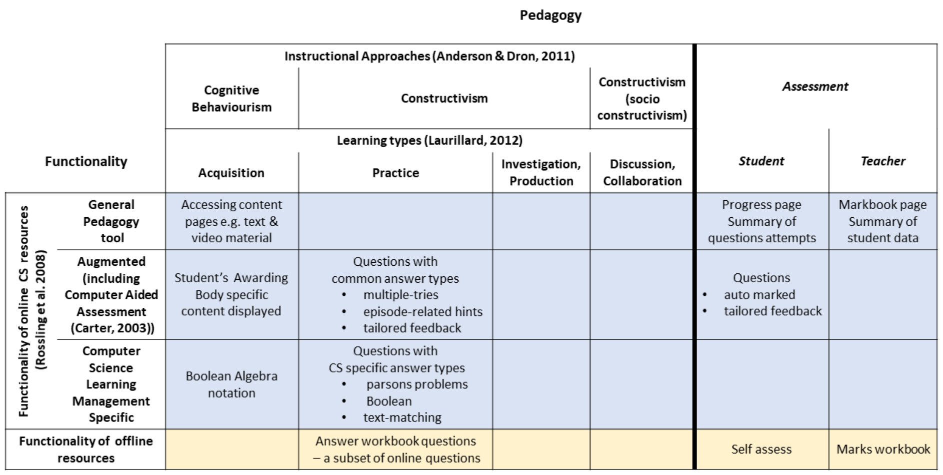
Platform usage statistics provide data on how many users are using the system. As at the end of May 2021, over 26,000 students and 2000 teachers from England had registered and over 1 million question attempts have been made.

6 DISCUSSION

We review the features of Isaac CS in two ways. First, as shown in Figure 3, we consider the pedagogy of the platform using the instructional approaches suggested by Anderson and Dron [1], which we have correlated to the learning types proposed by Laurillard [24], and functionality of online CS resources of Rößling et al. [36]. Secondly, we evaluate the coverage of functionality using the six purposes of blending instruction proposed by Means et al. [26].

As shown in Figure 3, the pedagogy of Isaac CS focuses on a cognitive behaviourist acquisition learning type as well as constructivist practice. Functionality that only shows students their AB content augments the learning for student needs [36] and supports learning through acquisition. Question types, including Boolean logic, Parsons problems [32], and text matching, can be viewed as LMS specific adaptations for CS [36] providing practice learning opportunities. Included in practice learning is immediate marking, context-specific feedback, and scaffolding hints which can be classified again as augmented functionality [36], a form of computer aided assessment [10]. Referring to the final row of Figure 3, there are opportunities to investigate whether the offline workbook is being used to exploit the learning types of investigation, production, discussion and collaboration within a more socio-constructivist instructional approach [1] as suggested by Anohah [2] for teaching this age group of learners of CS. However, analysing Isaac CS against the matrix, there are some gaps, with an apparent lack of system-specific opportunities for investigation, production, discussion and collaboration. These are addressed through professional development sessions where classroom discussions, investigation and collaborative production activities are suggested and demonstrated. To what extent these modelled activities are being appropriated into practice requires further study.

To reflect upon the effectiveness of the features of the Isaac CS we use the **six purposes of blending instruction** proposed by Means et al. [26] (see Section 2). Firstly, with tens of thousands of registered students who have made more than 1 million question attempts in the last two years, Isaac CS may be providing students in schools with **broadening access to instruction**. This may be in one of two ways: first, students with less experienced teachers may find their teachers blend the platform into class activities to



Pedagogy

		Instructional Approaches (Anderson & Dron, 2011)				Assessment	
		Cognitive Behaviourism	Constructivism		Constructivism (socio constructivism)		
		Learning types (Laurillard, 2012)					
Functionality		Acquisition	Practice	Investigation, Production	Discussion, Collaboration	Student	Teacher
Functionality of online CS resources (Rossling et al. 2008)	General Pedagogy tool	Accessing content pages e.g. text & video material				Progress page Summary of questions attempts	Markbook page Summary of student data
	Augmented (including Computer Aided Assessment (Carter, 2003))	Student's Awarding Body specific content displayed	Questions with common answer types • multiple-tries • episode-related hints • tailored feedback			Questions • auto marked • tailored feedback	
	Computer Science Learning Management Specific	Boolean Algebra notation	Questions with CS specific answer types • parsons problems • Boolean • text-matching				
Functionality of offline resources			Answer workbook questions – a subset of online questions			Self assess	Marks workbook

Figure 3: Isaac CS Pedagogy Matrix

strengthen instruction. Second, students who have suitable online home access may use the platform independently, but this needs to be evidenced.

In terms of **adding variety to instructions and enhancing learner engagement**, Isaac CS provides free access to online material to read and videos to watch, a variety of question types to answer with episode-related self-selected hints and tailored feedback. From the 2020 student survey there is indication that most learners think the resources are of good quality. For **productive practice**, there is indication of this through the 1 million question attempts and the 2020 survey results of reported teacher satisfaction with the quality of questions. But, investigation is needed to be sure of how productive this practice is.

The provision of content, questions and videos on CS affords the opportunity to **support the learning of complex, abstract concepts** but how effective this learning is requires study. Similarly, whether teachers have appropriated [25] the use of Isaac CS to effectively **serve students with diverse needs** and whether and how they have adapted and matured their use of Isaac CS in class to **facilitate small group and 1:1 teacher led instruction** requires further investigation.

7 CONCLUSION, LESSONS LEARNED AND NEXT STEPS

As in other countries, there is a shortage of teachers of computer science at upper secondary level in England [39]. Teachers new to the subject are being employed while still needing support in the underlying content [39], and there are few context-specific content-trusted resources with embedded activities available to them [34, 37]. To address this shortfall of resources and to support teachers and students, Isaac CS has been developed as a holistic content-rich A level CS online platform with wraparound support to help teachers blend the platform into their teaching. The open source nature of the content and platform means, although designed for this specific audience, the tool can be adapted to other CS contexts.

In this paper, we outlined the key design decisions made in developing Isaac CS and reflected on the testing and evaluation of these, and have drawn out four themes of Content, Pedagogy, Appropriation and General Functionality.

Lessons learned from the development and early use of Isaac CS include: i) General functionality: for platform developers, not all functionality is easy to apply, or of the same cost: questions involving natural language processing were found to be difficult to set up, and videos can be expensive to create and replace; ii) Pedagogical: student practice through answering questions can be augmented with multiple-tries, episode-related self-selected hints and tailored feedback functionality; iii) Appropriation: to ensure that teachers and students use an online platform, careful and sustained effort is needed; iv) Content: what upper secondary school CS content should include or look like has not yet been universally agreed but teachers are looking for content targeted to their requirements.

Future research could explore the Pedagogy Matrix, including theoretical foundations, providing examples of each dimension and comparing different tools and approaches with Isaac CS. Research could be undertaken to investigate how Issac CS impacts teachers' subject knowledge, knowledge of misconceptions and self-efficacy. Another interesting line of inquiry would be to study how each feature of the platform is blended with other online tools (such as IDEs) and face to face teaching activities to create pedagogy patterns [2, 6, 24]. Research could also focus on question type usage, such as whether Parsons problems are preferred by students to multiple choice questions, and what rate of multiple-attempts is most useful for learning, where we can compare to preferences and consider desirable difficulty [17]. In addition, more studies could investigate platform use by students and the effect on knowledge, progress, misconceptions, self-efficacy, and student communities of practice; such studies could inform our understanding of the ways that students in this age group learn CS.

Our review of Isaac CS has led us to the development of a Platform Pedagogy Matrix which we will continue to investigate. We hope this model may be useful for others to evaluate and build upon when creating or studying similar resources.

ACKNOWLEDGMENTS

This work was funded by the Department for Education as part of the Isaac Computer Science Project. The authors thank the developers & content writers who supported the writing of this paper.

REFERENCES

[1] Terry Anderson and Jon Dron. 2011. Three Generations of Distance Education Pedagogy. *The International Review of Research in Open and Distributed Learning* 3 (2011), 80–97. https://doi.org/10.19173/irrodl.v12i3.890

[2] Ebenezer Anohah. 2016. Pedagogy and Design of Online Learning Environment in CS Education for High Schools:. *International Journal of Online Pedagogy and Course Design* 6, 3 (2016), 39–51. https://doi.org/10.4018/IJOPCD.2016070104

[3] Michael K Barbour. 2019. The Landscape of K-12 Online Learning: Examining the state of the field. In *Online Learning* (4th ed.), Michael Moore and William Diehl (Eds.). Routledge, 521–524.

[4] Tim Bell, Jason Alexander, Isaac Freeman, and Mick Grimley. 2009. Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology* 13, 1 (2009), 20–29. https://www.citrenz.ac.nz/jacit/

[5] Paul Black and Dylan Wiliam. 2010. Inside the Black Box: Raising Standards through Classroom Assessment. *Phi Delta Kappan* 92, 1 (2010), 81–90. https://doi.org/10.1177/003172171009200119

[6] Peter Brusilovsky, Ken Koedinger, David A. Joyner, and Thomas W. Price. 2020. Building an Infrastructure for Computer Science Education Research and Practice at Scale. In *Proceedings of the Seventh ACM Conference on Learning @ Scale* (Virtual Event USA, 2020-08-12). ACM, 211–213. https://doi.org/10.1145/3386527.3405936

[7] Philip G Butcher and Sally E Jordan. 2010. A comparison of human and computer marking of short free-text student responses. *Computers & Education* 55, 2 (2010), 489–499. https://doi.org/10.1016/j.compedu.2010.02.012

[8] Andrew C. Butler. 2018. Multiple-Choice Testing in Education: Are the Best Practices for Assessment Also Good for Learning? *Journal of Applied Research in Memory and Cognition* 7, 3 (2018), 323–331. https://doi.org/10.1016/j.jarmac.2018.07.002

[9] Carmen Carrillo and Maria Assunção Flores. 2020. COVID-19 and teacher education: a literature review of online teaching and learning practices. *European Journal of Teacher Education* 43, 4 (2020), 466–487. https://doi.org/10.1080/02619768.2020.1821184 00063.

[10] Janet Carter, Kirsti Ala-Mutka, Ursula Fuller, Martin Dick, John English, William Fone, and Judy Sheard. 2003. How Shall We Assess This? *SIGCSE Bull.* 35, 4 (June 2003), 107–123. https://doi.org/10.1145/960492.960539

[11] Stephen Cummins, Alistair Stead, Lisa Jardine-Wright, Ian Davies, Alastair R. Beresford, and Andrew Rice. 2016. Investigating the Use of Hints in Online Problem Solving. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale* (Edinburgh Scotland UK). ACM, 105–108. https://doi.org/10.1145/2876034.2893379

[12] Department for Education. 2020. Apply to have your qualifications regulated. https://www.gov.uk/guidance/apply-to-have-your-qualifications-regulated

[13] Department for Education. 2020. A level and other 16 to 18 results 2019 to 2020. https://analytics.ofqual.gov.uk/apps/Alevel/Outcomes/

[14] Yuemeng Du, Andrew Luxton-Reilly, and Paul Denny. 2021. A Review of Research on Parsons Problems. In *Proceedings of the Twenty-Second Australasian Computing Education Conference* (Melbourne VIC Australia, 2020-02-03). ACM, 195–202. https://doi.org/10.1145/3373165.3373187

[15] Jessie Durk, Ally Davies, Robin Hughes, and Lisa Jardine-Wright. 2020. Impact of an active learning physics workshop on secondary school students' self-efficacy and ability. *Physical Review Physics Education Research* 16, 2 (2020), 020126. https://doi.org/10.1103/PhysRevPhysEducRes.16.020126

[16] Education Endowment Fund. 2019. Using Digital Technology to Improve Learning – Guidance Report. https://educationendowmentfoundation.org.uk/public/files/Publications/digitalTech/EEF_Digital_Technology_Guidance_Report.pdf

[17] Barbara J. Ericson, Mark J. Guzdial, and Briana B. Morrison. 2015. Analysis of Interactive Features Designed to Enhance Learning in an Ebook. In *Proceedings of the eleventh annual International Conference on International Computing Education Research* (Omaha Nebraska USA). ACM, 169–178. https://doi.org/10.1145/2787622.2787731

[18] Barbara J. Ericson, Lauren E. Margulieux, and Jochen Rick. 2017. Solving parsons problems versus fixing and writing code. In *Proceedings of the 17th Koli Calling International Conference on Computing Education Research* (Koli Finland, 2017-11-16). ACM, 20–29. https://doi.org/10.1145/3141880.3141895

[19] Carl C. Haynes and Barbara J. Ericson. 2021. Problem-Solving Efficiency & Cognitive Load for Adaptive Parsons Problems vs. Writing the Equivalent Code. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama Japan, 2021-05-06). ACM, 1–15. https://doi.org/10.1145/3411764.3445292

[20] Michael B Horn and Heather Staker. 2011. *The Rise of K–12 Blended Learning.* Innosight Institute, Inc. 18 pages. https://aurora-institute.org/resource/the-rise-of-k-12-blended-learning/

[21] Lisa Jardine-Wright. 2018. *Isaac Physics Impact and Engagement Summary V6.* University of Cambridge. https://cdn.isaacphysics.org/isaac/publications/impact_summary_201804_v6.pdf

[22] Ville Karavirta, Petri Ihantola, and Teemu Koskinen. 2013. Service-Oriented Approach to Improve Interoperability of E-Learning Systems. In *2013 IEEE 13th International Conference on Advanced Learning Technologies* (Beijing, China). IEEE, 341–345. https://doi.org/10.1109/ICALT.2013.105

[23] Ari Korhonen, Rocky Ross, Clifford A. Shaffer, Thomas Naps, Charles Boisvert, Pilu Crescenzi, Ville Karavirta, Linda Mannila, Bradley Miller, Briana Morrison, and Susan H. Rodger. 2013. Requirements and design strategies for open source interactive computer science eBooks. In *Proceedings of the ITiCSE working group reports conference on Innovation and technology in computer science education-ITiCSE -WGR '13* (Canterbury, England, United Kingdom). ACM Press, 53–72. https://doi.org/10.1145/2543882.2543886

[24] Diana Laurillard. 2012. *Teaching as a design science: building pedagogical patterns for learning and technology.* Routledge. https://doi.org/10.4324/9780203125083

[25] Tobias Ley, Ronald Maier, Stefan Thalmann, Lena Waizenegger, Kai Pata, and Adolfo Ruiz-Calleja. 2020. A Knowledge Appropriation Model to Connect Scaffolded Learning & Knowledge Maturation in Workplace Learning Settings. *Vocations and Learning* 13, 1 (2020), 91–112. https://doi.org/10.1007/s12186-019-09231-2

[26] Barbara Means, Marianne Bakia, and Robert Murphy. 2014. *Learning online : what research tells us about whether, when and how.* Routledge, Taylor & Francis Group, New York.

[27] James Moore. 2020. Mark it yourself, it's good for you! A case study examining the impact of self-assessment on student learning and motivation in a year 12 physics class studying energy. *Journal of Trainee Teacher Education Research* 11 (2020), 38.

[28] National Centre for Computing Education. 2021. *Blog Post: Isaac Computer Science learning resources save teachers three hours a week.* Retrieved June 2, 2021 from https://blog.teachcomputing.org/isaac-computer-science-learning-resources-save-teachers-3-hours-a-week/

[29] Organization for Economic Cooperation \and Development (OECD). 2015. *Students, Computers and Learning: Making the Connection.* OECD. https://doi.org/10.1787/9789264239555-en

[30] Organization for Economic Cooperation \and Development (OECD). 2020. Education responses to covid-19: Embracing digital learning and online collaboration. https://read.oecd-ilibrary.org/view/?ref=120_120544-8ksud7oaj2&title=Education_responses_to_Covid-19_Embracing_digital_learning_and_online_collaboration&_ga=2.52549928.1475366060.1620910560-2096753378.1620234231

[31] Ofqual. 2021. Register of Regulated Qualifications. https://register.ofqual.gov.uk/

[32] Dale Parsons and Patricia Haden. 2006. Parson's Programming Puzzles: A Fun and Effective Learning Tool for First Programming Courses. In *Proceedings of the 8th Australasian Conference on Computing Education - Volume 52* (Hobart, Australia) *(ACE '06)*. Australian Computer Society, Inc., AUS, 157–163.

[33] Henk J. Pol, Egbert G. Harskamp, Cor J. M. Suhre, and Martin J. Goedhart. 2008. The Effect of Hints and Model Answers in a Student-Controlled Problem-Solving Program for Secondary Physics Education. *Journal of Science Education and Technology* 17, 4 (2008), 410–425. https://doi.org/10.1007/s10956-008-9110-x

[34] Pye Tait Consulting. 2017. *After the Reboot: The State of Computing Education in UK Schools and Colleges Final Report September 2017.* Technical Report. The Royal Society. https://royalsociety.org/~/media/policy/projects/computing-education/pye-tait-teacher-survey-report.pdf

[35] María Jesús Rodríguez-Triana, Luis P. Prieto, Tobias Ley, Ton de Jong, and Denis Gillet. 2020. Social practices in teacher knowledge creation & innovation adoption: a large-scale study in an online instructional design community for inquiry learning. *International Journal of Computer-Supported Collaborative Learning* 15, 4 (01 Dec 2020), 445–467. https://doi.org/10.1007/s11412-020-09331-5

[36] Guido Rößling, Mike Joy, Andrés Moreno, Atanas Radenski, Lauri Malmi, Andreas Kerren, Thomas Naps, Rockford J. Ross, Michael Clancy, Ari Korhonen, Rainer Oechsle, and J. Ángel Velázquez Iturbide. 2008. Enhancing Learning Management Systems to Better Support Computer Science Education. *SIGCSE Bull.* 40, 4 (Nov. 2008), 142–166. https://doi.org/10.1145/1473195.1473239

[37] Sue Sentance and Andrew Csizmadia. 2017. Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies* 22, 2 (2017), 469–495. https://doi.org/10.1007/s10639-016-9482-0

[38] The National Archives. 2014. Launch of Open Government Licence 3.0. https://webarchive.nationalarchives.gov.uk/ukgwa/+/http://www.nationalarchives.gov.uk/news/970.htm

[39] The Royal Society. 2019. Policy briefing on teachers of computing. https://royalsociety.org/-/media/policy/Publications/2019/21-08-19-policy-briefing-on-teachers-of-computing.pdf

[40] Jane Waite, Paul Curzon, and Jo Brodie. 2021. Sharing research-informed programming pedagogy with IT Professionals - submitted. (2021).

[41] Nathaniel Weinman, Armando Fox, and Marti A. Hearst. 2021. Improving Instruction of Programming Patterns with Faded Parsons Problems. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama Japan, 2021-05-06). ACM, 1–4. https://doi.org/10.1145/3411764.3445228

[42] Clive Young and Nataša Perović. 2016. Rapid and Creative Course Design: As Easy as ABC? *Procedia - Social and Behavioral Sciences* 228 (2016), 390–395. https://doi.org/10.1016/j.sbspro.2016.07.058