Teaching CS with and through other forms of knowledge

Paul Curzon
Queen Mary University of London
London, UK
p.curzon@qmul.ac.uk

Jane Waite
University of Cambridge and
Raspberry PI Foundation
Cambridge, UK
jw2251@cam.ac.uk

Karl Maton
University of Sydney
Sydney, Australia
karl.maton@sydney.edu.au

ABSTRACT

Computer Science (CS) is often taught in K-5 education with and through other forms of knowledge, such as CS with Maths, Science, or Art. How two bodies of knowledge and skills are interleaved in a single learning event can be complex to analyse. The sociological framework Legitimation Code Theory (LCT) includes a set of concepts called 'Autonomy' for exploring how different knowledge practices are brought together and with what effects. To explore the value of Autonomy in CS education, we analysed a lesson plan of an activity that teaches CS (algorithms) through magic, visualising the findings on an *autonomy plane*. This revealed ways to improve learning such as by creating *autonomy tours*. Autonomy analysis has use in reflective CS K-5 lesson design as CS is often taught with other subjects to overcome timetabling constraints, build on other subjects, or to raise interest in equitable learning experiences.

CCS CONCEPTS

• **Social and professional topics** → **Computer science education**; **K-12 education**; *Computational thinking*; *Computing education programs*.

KEYWORDS

Unplugged computing, Autonomy, Legitimation Code Theory

ACM Reference Format:
Paul Curzon, Jane Waite, and Karl Maton. 2024. Teaching CS with and through other forms of knowledge. In *The 19th WiPSCE Conference on Primary and Secondary Computing Education Research (WiPSCE '24), September 16–18, 2024, Munich, Germany*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3677619.3678104

1 INTRODUCTION AND RELATED WORK

A variety of pedagogic approaches have been suggested as ways to teach Computer Science (CS) effectively, including using a second set of knowledge practices (such as a different subject area) to situate learning. The rationale for using a second subject may be to overcome the lack of time to teach each subject separately (particularly in K-5 classrooms); because one (e.g. Maths) is the foundation for the other; or to increase student motivation by employing knowledge viewed as more relevant or fun to learners. An example of using a second form of knowledge to increase motivation is to use magic tricks to teach CS. This combination has proven a successful approach, where children and teachers are inspired about CS topics through its links to magic. [2]. An aim is to teach both areas which risks teaching neither well, a general issue that has been raised both specifically in CS and more generally [5, 11]. Why this is hard is yet to be resolved [15, 16]. Similar issues arise when teaching a variety of combinations of knowledges, such as the Maths underpinning computing [19]. How, then, can such combinations be taught successfully?

Legitimation Code Theory (LCT) is a widely used sociological framework that reveals the organising principles underlying knowledge practices (e.g. [7]). It has been used to explore curriculum, teaching, learning and assessment practices across the disciplinary and institutional maps of education (e.g., [7][1]). LCT offers a toolkit of concepts that have had major impact, both shaping research and empowering teachers and students to master the basis of achievement (e.g., [13]). LCT concepts are organised into *dimensions*. CS education studies have previously analysed classroom practices using the LCT dimension called 'Semantics' [6]. These concepts reveal the context-dependence and complexity of knowledge practices. The tools help educators improve learning activities by, e.g., optimising the sequence that knowledge practices are taught / learned. Such analysis has been done for teaching CS with unplugged activities [18], including with magic tricks using a heuristic analysis method [3]. The Autonomy dimension [8] provides a second set of tools valuable for revealing how different kinds of knowledge practices can be brought together. They show how distinctive **contents** and **purposes** are related in different ways and with what effects. These different practices could be different forms of knowledge, different disciplines, technology, classrooms, etc. [8]. For example, studies have explored what patterns of teaching support / inhibit the teaching of Maths in Science lessons [9], building on student experiences of COVID-19 to teach Biology [12], using Harry Potter to teach English literary writing [4] and using everyday knowledge in History lessons [8]. The contribution of this paper is to demonstrate how autonomy analysis can apply to teaching CS with other forms of knowledge, and illustrate its potenital utility to support CS teachers, asking: **What is revealed by an autonomy analysis of a CS lesson plan in which magic is used to help teach CS?**

2 METHOD: A CASE STUDY

We used a specific CS via magic lesson plan (Invisible Palming) as a case study. The magic trick involves invisibly moving a card from one pile of cards to another. The trick introduces the core UK K-5 concept of an algorithm. The lesson is a concrete and popular activity aimed at primary school children. The 7 steps of the lesson are shown in Table 1. In this context, the aim of the activity is to teach both what an algorithm is and the trick. It is the success of

WiPSCE '24, September 16–18, 2024, Munich, Germany

[1]For work using LCT, see the publications database: www.legitimationcodetheory.com

Table 1: Analysis of the outline plan used for Invisible Palming, when the lesson target is CS.

Step	Description of step	Description of PA and RA	Position on plane
1	Presenter does the trick (Invisible Palming)	Using magic content (PA– –) for the purposes of learning magic (RA– –).	(PA– –, RA– –)
2	Audience does the trick (to ensure they can do it).	Using magic content (PA– –) for the purposes of learning magic (RA– –).	(PA– –, RA– –)
3	Audience tries to work out how the trick works by experiment and problem solving.	Using general CS related STEM content (PA+) for the purposes of understanding the magic (RA– –).	(PA+, RA– –)
4	Presenter explains how the trick works.	Using STEM and magic principles underpinning the trick content (PA–) for the purposes of understanding magic. (RA– –)	(PA–, RA– –)
5	Presenter explains how the principles behind the trick relate to CS topics (self-working tricks and algorithms are the same)	Using magic principles (self working tricks) content (PA–) for the purposes of understanding CS (RA++).	(PA–, RA++)
6	Presenter summarises the CS (what an algorithm is)	Using CS content (PA++) for the purposes of understanding CS (RA++).	(PA++, RA++)
7	Presenter summarises the underlying lesson about magic (self-working tricks are algorithms)	Using CS content (PA++) for the purposes of understanding principles of magic (RA–).	(PA++, RA–)

Table 2: Translation device for the purpose of teaching Computer Science as part of the Invisible Palming activity

1st level	This study	2nd level	This study
target	CS	core (PA++/RA++)	CS: Algorithms and computation
		ancillary (PA+/RA+)	Science and maths techniques underpinning CS
non-target	Non-CS	associated (PA–/RA–)	Science, maths, magic underpinning trick
		unassociated (PA– –/RA– –)	Everything else (including how to do trick)

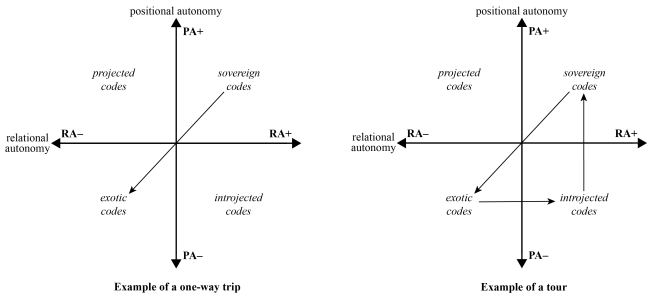

Figure 1: Examples of autonomy planes and related quadrant names (codes) and autonomy pathways [8] ©Karl Maton

this dual aim that we studied exploring whether the aim of teaching the trick undermines that of teaching what an algorithm is.

As the primary aim of the activity is teaching CS, we analysed the lesson using Autonomy from the perspective of CS being the target. We first created a translation device with CS as the target (Table 2). It guided the allocation of codes (giving strengths of content or purpose) to steps in the lesson plan as given in Table 1. We used consensual coding to do the analysis (both analysts were experienced CS educators with one also an experienced amateur magician). The first author did an initial analysis. The results were then discussed with the second author step by step. The first author explained the coding, with potential issues or disagreements raised by the second author. Final codes were then agreed. This process led to some changes to the translation device and to the coding.

Once the coding was agreed, the results were plotted on the autonomy plane. It gives a visual version of the lesson that can aid reflection and help suggest changes to improve the plan e.g., whether a one-way trip or tour is followed (Figure 1). The y-axis represents strengths of positional autonomy (PA, content) and the x-axis represents strengths of relational autonomy (RA, purpose), each charting a continuum between stronger and weaker. The autonomy plane represents a topology of infinite possible positions – we can locate practices anywhere within the plane. The four principal autonomy codes are named (from top right clockwise) as sovereign codes, introjected codes, exotic codes and projected codes (Figure 1).

3 ANALYSIS AND RESULTS

Step 1 (see Table 1) is the presenter does the magic trick, followed by the audience attempting the trick in pairs (Step 2). Both steps are about doing magic content for the immediate purpose of learning the trick. From the translation device (Table 2), both steps are exotic codes (PA– –, RA– –). Both use magic content (PA– –) for the purpose of teaching magic (RA– –). Both are plotted in the bottom left-hand corner of the plane (Figure 2).

Step 3 asks the audience to work out how the trick works. To do this they use general STEM content relevant to CS — doing experiments and problem-solving (PA+). However, they are doing it for the purposes of understanding the magic (RA– –). We plot this step on the plane, therefore, as a projected code (PA+, RA– –).

In Step 4, the magician talks through the trick, explaining the general magic principle of a self-working trick and the maths of odd and even numbers (PA–). The immediate purpose is for the

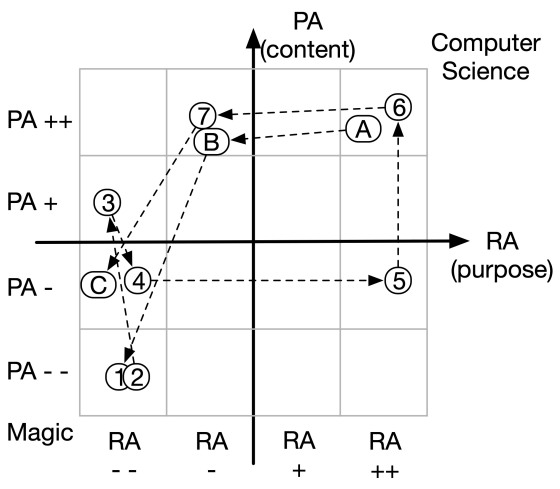

Figure 2: Autonomy plane for the initial plan (Steps 1-7) and potential lesson changes (Steps A-C).

audience to understand the magic trick (RA– –). This is, therefore, a move back to an exotic code.

The final stage of the lesson involves the presenter using what has been learned about magic to explain the CS that is the target of the lesson. First (Step 5), they explain how the principles behind the trick, relate to CS topics (self-working tricks are just what a computer scientist calls an algorithm). This is using the magic content of what a self-working trick is (PA–) for the purpose now of understanding the CS of what an algorithm is (RA+). This step is plotted, therefore, in the introjected code quadrant (PA–, RA+). Step 6 solidifies this link as CS, by now explaining the CS content of what an algorithm is (PA++) now for the purpose of the audience understanding the CS of what an algorithm is (RA++). This step is plotted in the top right corner of the autonomy plane diagram.

There is one final step (Step 7). The activity concludes by the presenter making a link back to magic – they summarise the underlying lesson about magic (self-working tricks are algorithms). They are using CS content (PA++) for the purposes of helping the audience understand the principles of magic (RA–). This final step is plotted on the plane at (PA++, RA–).

The final analysis of the original version of the lesson (Steps 1–7) shows that with the aim of teaching CS, the lesson follows a one-way trip (if a meandering one) from the exotic codes (using magic for the purposes of teaching magic) eventually to the sovereign codes (using CS for the purposes of teaching CS), passing through both other codes (using one to teach the other) but finishing in the projected codes (summarising with the purpose of magic).

A positive of this pathway's structure is that it moves across the plane, including using the introjection and projection codes to strengthen core understanding of both CS and magic with paths from those codes into the sovereign and exotic codes. The pathway does not just jump between sovereign (CS for the purpose of teaching CS) and exotic (magic for the purpose of teaching magic) codes. This would indicate little linkage between the subjects – that they were being taught separately with limited scaffolding to the students to build understanding from one into the other. There are weaknesses in the pathway, however. For example, the path neither starts nor ends in a sovereign code. If that is the main topic of the lesson then that is not necessarily being made clear. Students could easily leave thinking it was *all* about magic. The lack of a tour also misses the opportunity to fully strengthen the core CS learning.

4 LESSON CHANGES AND DISCUSSION

The lesson plan follows a meandering path, where the end is not back at the origin, so it is not a tour. A way to turn this into a tour is to add an initial lesson step: using CS content for the purpose of teaching CS (PA++, RA++) (Step A in Figure 2), explaining the learning outcome. This would also mean the activity starts in a sovereign code and so emphasises that CS is the point. Coming back to that in a summary at the end emphasises this learning.

Context matters, however. LCT is not giving definitive answers, it is just a tool to aid reflection. The reason for going straight into the magic was to grab attention and for the students to work out for themselves the link between a self-working trick and an algorithm, encouraging active learning. If we do anything, it must not spoil the delight of the magic. A brief introduction to what an algorithm is, is not likely to cause any problems with respect to this. It also gives the students the chance to gain the insight of the algorithm link before being told. If they know nothing about algorithms at the outset, then that is not possible. Step A seems, therefore, both a safe, and positive step to add.

LCT concepts (and practice in other subjects) suggests that paths that go via projected / introjected codes can be more powerful for learning. Here, an initial trip passing through the projected codes (using CS content to explain magic) may help. This would involve making a link directly from the introductory CS we just explained to its purpose with respect to the magic. This has the potential to strengthen the understanding of the magic as well as strengthening the link between the two (more so than just jumping straight from one to the other). On the plane, this would be coded as Step B (PA++, RA–) using the STEM underpinning tricks to introduce a general principle of magic. This makes the step a trip from sovereign to exotic, but via the projected quarter, suggesting it would strengthen the understanding of the magic as well as the link.

Looking at the end of the path, we finish on an extra dog-leg after the tour. However, the endpoint is in the projected codes. This is the step that is essentially a copy of the one we have just added at the start. Similar reasoning applies though, if that step's purpose is to use CS to underpin the understanding of magic, why not finish on the magic itself. This part of the activity could, therefore, be strengthened for the purposes of teaching magic by making the very last step a summary explanation of the magic just in terms of magic concepts, but building on the understanding of the link we just gave. This would be using the CS introduced (algorithms) for the purpose of teaching magic as the basis for explaining the final magic lesson, rather than as an endpoint in itself. By ending with magic this would further strengthen the lesson around the magic concepts learned. This gives a new endpoint C. This would be coded as a point (PA–, RA– –) because it would be describing

STEM principles underpinning magic (rather than details of the specific trick) for the purpose of understanding that specific trick.

Do these changes improve the activity in practice? A full empirical study would be needed to confirm this, which is beyond the scope of this paper. We did, however, integrate some of these suggestions into a new version of the activity that was then delivered. Notably, we added Steps A and B to workshops for two groups of primary school children (one group aged 7-8, the other 9-11) as an initial pilot. It was noticeable that some students made the connection themselves, with prompting, that the magic trick was just an algorithm. In the younger group, there was a clear disadvantage to this change, though. The pace needed to be fast to keep their attention. The extra steps slowed this down. Getting to the concrete grab more quickly was more important. In this group, perhaps keeping the original structure or just giving a very brief statement of what an algorithm is would be preferable. Starting with concrete context - a magic trick - and then gradually building to the abstract theory was the main aim (later linking back to the trick). This may well be preferable as they are less able to understand or find interesting the abstract concepts without something to pin them too [14].

Our research question was: **What is revealed by an autonomy analysis of a CS lesson plan in which magic is used to help teach CS?** The autonomy analysis reveals the underlying structure of how the two topics are taught. It shows how insulated or otherwise they are from each other: is one being actively used to support understanding in the other, or are they being taught separately in a way that makes it hard for learners to make any connections between them. It also reveals potential lesson plan improvements to reinforce the links. Overall, it gave insight into whether the main target topic was likely to be learnt well or whether the subservient topic (here magic) was more likely to be remembered.

An autonomy analysis does not give precise solutions. It is a tool for reflection. LCT Autonomy could be introduced in CS teacher professional development to meet the calls to help educators better understand how to design and analyse learning activities that teach with and through other subjects (e.g., [5, 10, 19]), which is increasingly required or suggested in CS education (e.g., [1, 17]).

We have focused on just one example of two knowledge practices being combined. CS is taught with other subjects for many reasons: programming has to be set within a context domain and often that domain is some other topic (such as programming a Biology quiz); metaphor and analogy drawn from another discipline or everyday life can be useful; some topics build directly on other areas (e.g.,. much CS is founded on Maths, and CS is used as an application area to illustrate the Maths). Autonomy analysis can cast light on all these uses in a lesson plan.

5 CONCLUSIONS AND FURTHER WORK

The use of LCT Autonomy to analyse the Invisible Palming activity illustrated the way different knowledge practices were integrated. Autonomy plane diagrams gave a practical way to visualise learning pathways. The analysis suggested that changes to: introduce explicit tours; make better use of projected / introjected codes, and alter start and end points of the pathway could improve the lesson for the purpose of teaching CS. However, empirical studies are needed to confirm that lesson plan changes will lead to improved practice.

LCT Autonomy helps highlight why the teaching of multiple topics might be successful or not. It provides a simple and powerful tool for reflection that can improve lesson plans. LCT Autonomy could be introduced in CS teacher professional development, thereby filling a gap in the need to support the teaching of multiple topics in an integrated but effective way.

Acknowledgements: Funded in part by EPSRC on funding agreement EP/W033615/1.

REFERENCES

[1] CSforCA. 2022. *Integrating CS into other subject areas.* Technical Report. CS FOR California, California, USA. https://csforca.org/wp-content/uploads/2022/01/CSforCA_integration_v3-1.pdf

[2] Paul Curzon and Peter W. McOwan. 2008. Engaging with computer science through magic shows. In *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education* (Madrid, Spain) *(ITiCSE '08)*. ACM, New York, NY, USA, 179–183. https://doi.org/10.1145/1384271.1384320

[3] Paul Curzon, Jane Waite, Karl Maton, and James Donohue. 2020. Using semantic waves to analyse the effectiveness of unplugged computing activities. In *Proceedings of the 15th Workshop on Primary and Secondary Computing Education* (Virtual Event, Germany) *(WiPSCE '20)*. ACM, New York, USA, 1–10. https://doi.org/10.1145/3421590.3421606

[4] G. Jackson. 2021. Harry Potter and the critical gaze: Autonomy pathways in literary response writing. *Journal of Education* 83 (2021), 69–86.

[5] Alaina Mabie, Monica McGill, and Brenda Huerta. 2023. A Systematic Literature Review Examining the Impacts of Integrating Computer Science in K-5 Settings. In *2023 ASEE Annual Conference & Exposition Proceedings*. ASEE Conferences, Baltimore, Maryland. https://doi.org/10.18260/1-2--42523

[6] Karl Maton. 2013. Making semantic waves: a key to cumulative knowledge-building. *Linguistics and Education* 24, 8-22 (2013).

[7] Karl Maton. 2014. *Knowledge and Knowers: Towards a realist sociology of education.* Routledge, New York.

[8] Karl Maton. 2018. *Taking autonomy tours: A key to integrative knowledge-building.* LCT Centre for Knowledge-Building, University of Sydney, Sydney, Australia. LCT Centre Occasional Paper 1.

[9] Karl Maton. 2021. Targeting science: Successfully integrating mathematics into science teaching. In *Teaching Science: Knowledge, Language, Pedagogy* (1st ed.). Routledge, New York, 23–48. https://doi.org/10.4324/9781351129282

[10] Monica M. McGill, Laycee Thigpen, and Alaina Mabie. 2023. Emerging Practices for Integrating Computer Science into Existing K-5 Subjects in the United States. In *Proceedings of the 18th WiPSCE Conference on Primary and Secondary Computing Education Research* (Cambridge, United Kingdom) *(WiPSCE '23)*. ACM, New York, USA, 1–10. https://doi.org/10.1145/3605468.3609759

[11] Graham McPhail. 2018. Curriculum integration in the senior secondary school: a case study in a national assessment context. *Journal of Curriculum Studies* 50, 1 (2018), 56–76. https://doi.org/10.1080/00220272.2017.1386234

[12] M. Mouton. 2021. Harnessing the Beast – COVID-19: Integrative Knowledge-Building with LCT Autonomy. *Eurasian Journal of Science and Environmental Education* 1, 1 (2021), 27–42.

[13] Clarence Sherran. 2021. *Turning Access into Success: Improving university education with Legitimation Code Theory.* Routledge, New York.

[14] Anna-Vera Meidell Sigsgaard. 2024. Session 14 (B46): Semantic waves for helping teachers teach science to second language students. In *Programme of the 5th International Legitmation Code Theory Conference* (Johannesberg, South Africa) *(LCT 5)*. University of Witwatersrand, 43–44.

[15] Serena Thoma, Elise Deitrick, and Michelle Hoda Wilkerson. 2018. "It Didn't Really Go Very Well": Epistemological Framing and the Complexity of Interdisciplinary Computing Activities. *Proceedings of International Conference of the Learning Sciences, ICLS* (2018), 1121–1124.

[16] Benoît Tonnetti and Vanessa Lentillon-Kaestner. 2023. Teaching interdisciplinary in secondary school: a systematic review. *Cogent education* 10 (2023), 2216038. https://doi.org/10.1080/2331186X.2023.2216038

[17] Emiliana Vegas, Michael Hansen, and Brian Fowler. 2021. *Building Skills for Life (Brookings report).* Technical Report. The Brookings Institution.

[18] Jane Waite, Karl Maton, Paul Curzon, and Lucinda Tuttiett. 2019. Unplugged Computing and Semantic Waves: Analysing Crazy Characters. In *Proceedings of the 1st UK & Ireland Computing Education Research Conference on - UKICER*. ACM Press, Canterbury, UK, 1–7. https://doi.org/10.1145/3351287.3351291

[19] David Weintrop, Elham Beheshti, Michael Horn, Kai Orton, Kemi Jona, Laura Trouille, and Uri Wilensky. 2016. Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology* 25, 1 (Feb. 2016), 127–147. https://doi.org/10.1007/s10956-015-9581-5