International Society of the Learning Sciences

"Jupyter-Notebook-as-Script": Investigating the Nature and Impact of Implicit Collaboration Scripts in Computational Notebooks

Zhenyu Cai, EPFL, zhenyu.cai@epfl.ch
Richard Lee Davis, KTH Royal Institute of Technology, rldavis@kth.se
Roland Tormey, EPFL, roland.tormey@epfl.ch
Pierre Dillenbourg, EPFL, pierre.dillenbourg@epfl.ch

Abstract: There are a variety of CSCL scripts that offer different levels of flexibility, ranging from high-structured guidance towards low-rigidity support. This study is grounded in the view that scripts should be designed with a certain degree of structuredness, and we aim to examine the presence of implicit scripts based on data collected from collaborative programming activities in authentic classroom settings. By extracting the task dependencies within Jupyter notebooks, we adopted Directed Acyclic Graphs (DAGs) to operationalize the concept of implicit scripts. Next, we analyzed the group interaction data and measured their collaboration strategy alignment with the implicit scripts. Notably, we found that introducing a planning phase significantly improved groups' adherence to the implicit scripts. Qualitative data were reported to triangulate our findings. We discuss pedagogical and technological implications for more general CSCL contexts.

Introduction and background

The topic of scripting has been widely investigated in the field of CSCL, with foundational efforts ranging from conceptual frameworks (Kollar et al., 2006; Fischer et al., 2013) to practical considerations (Dillenbourg & Jermann, 2007), as well as empirical work that integrate both (Vogel et al., 2021; Tchounikine, 2008).

Given the variations in their design and enactment across different contexts, prior work also offers insights into several taxonomies of script types. One common distinction is between micro- and macro-scripts: micro-scripts refer to dialogue models that guide specific interactions, such as prompting students to make and respond to arguments; macro-scripts, on the other hand, operate as pedagogical models that outline the group activities and processes (Dillenbourg & Hong, 2008). Here, granularity plays a key role: micro-scripts guide fine-grained interactions, while macro-scripts provide coarse-grained structures. Similarly, Dillenbourg (2002) classified scripts into five categories: induced scripts, instructed scripts, trained scripts, prompted scripts, and follow-me scripts, based on their levels of coercion degree. It is also argued that a low degree of coercion may be inefficient to influence the collaborative processes, while designers should be cautious that a high degree of coercion could lead to "over-scripting". The gap between the intended activities prescribed by scripts and the actual interactions is referred to the notion of "flexibility", which emphasizes that intrinsic and extrinsic constraints should be respected to maintain pedagogical integrity (Dillenbourg & Tchounikine, 2007).

Whereas typical collaboration scripts are externally represented to be followed and enacted by individuals and groups, Kollar et al. (2006) and Kollar et al. (2007) further introduced the concept of "internal scripts". In contrast to external ones, internal scripts refer to the mental representation students construct to reify the external ones, while there remains a complex interplay between them. Internalizing collaboration scripts involves cognitive and social processes through which students interpret, adapt, and integrate the external guidance into their own mental frameworks. Compared to experts, novices may find it harder to internalize an external script if it possesses higher cognitive load. This will exacerbate if the external scripts are designed with higher degree of coercion and enacted with more flexibility. Kollar et al., (2007) examined the interactions of internal and external scripts through a 2×2 factorial experiment. They found that a highly structured external script was effective in supporting acquisition of domain-general knowledge no matter how well learners internalized it. However, students who owned a high-structured internal script would gain more domain-specific knowledge than those with low-structured scripts.

Until now, the theorization of collaboration scripts, including the appropriation of scripts (Tchounikine, 2016), has primarily focused on the external-internal dimension. We propose to extend the theory by adding a second dimension: implicit-explicit. On the one hand, internal scripts function implicitly and cannot be directly observed, as they are essentially cognitive schemas according to the Script Theory of Guidance (SToG) (Vogel et al., 2021; Fischer et al., 2013). On the other hand, researchers have employed different methods to evaluate the internal scripts. For example, Kollar et al., (2007) used learners' test performance, where they were asked to assess the quality of argumentative moves, to distinguish the rather high- or low-level structure of individuals' internal scripts, which was then validated through a post-analysis of learner's argumentative discourses. From a

psychological perspective, Fischer et al. (2013), in their SToG, expanded the theory of internal scripts by representing them through four distinct components: play, scene, role, and scriptlet. In the meantime, the same components were reused in SToG to conceptualize external scripts that aligned with the same hierarchical structure but as scaffolds to activate and (re-)configure the internal ones. A reasonable inference is that using approaches such as think-aloud could enable a more explicit representation of these schemas. In the study of Stegmann et al. (2012), learners were asked to follow the think-aloud protocols, and the resulting segments were coded to measure the depth of cognitive elaboration (a prerequisite to internalize external scripts), although it could be affected by the learners' prior knowledge, types of learning task, etc.

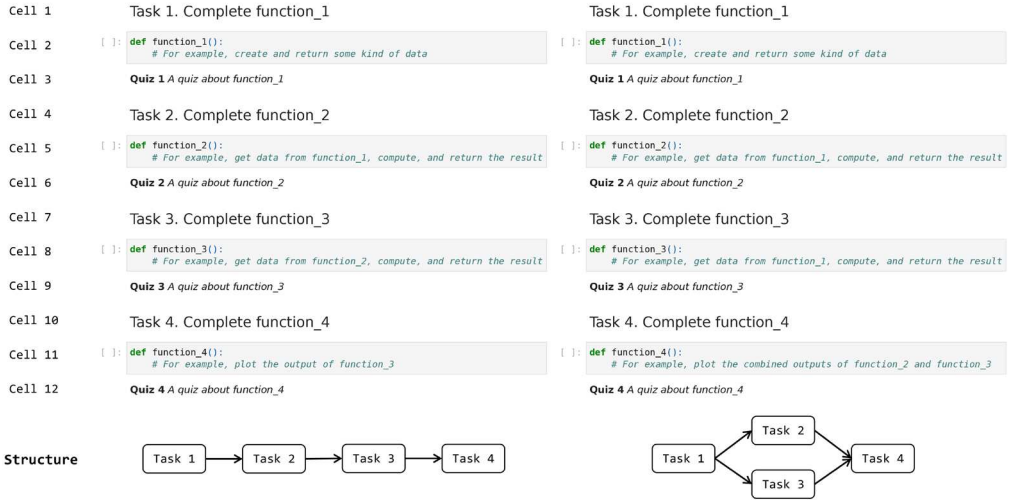
In contrast, external scripts are typically observable, and undoubtedly, the majority of script-based interventions fall within the quadrant of external and explicit scripts. However, as the degree of coercion decreases, an external script becomes less structured—essentially, less explicit in terms of conveying the expected interactions—until it reaches the threshold of "under-scripting" (Stegmann et al., 2011). Similar to the above discussion about internal scripts, we then introduce the implicit dimension to conceptualize this trend. For example, the interaction patterns would be shaped by the implicitly structured collaborative software (Lingnau & Bientzle, 2009) or the communication interface equipped with induced scripts (Dillenbourg, 2002). The explicit representation of such scripts is at a minimal level (e.g., the interactive elements, widget layouts, etc.), however, it is the implicit part that guides the expected collaborative processes. We thus hypothesize that, at the far end of low-structured scripts, they become practically implicit. Echoing the prior work that defined "implicit scripting" at the content level based on the concept of representational guidance (Runde et al., 2007), we term this type of scripts "*implicit scripts*" and define them as structural affordances embedded within learning materials that guide collaboration without explicit instruction. Such scripts demand cognitive effort to explicate and interpret, while their implicit nature allows learners to adaptively enact them. Although implicit scripts exhibit a high degree of flexibility, comparable to self-scripts (Dillenbourg & Tchounikine, 2007), it is important to clarify that they still possess certain intrinsic constraints that affect the ways how learners enact—such as the modes of interaction inherited from the technical environment—albeit with a lower degree of coercion than the latter.

Computational notebooks

Computational notebooks are platforms that allow users to integrate code, data, text, and interactive widgets within a single document. Such notebooks usually consist of a series of "cells" that can be executed in any sequence, supporting experimentation and rapid feedback. These features make them well-suited for use as interactive textbooks or programming exercises in educational settings. However, the flexibility to modify (e.g., moving, inserting, deleting, and editing) cells and to determine their execution order does not compromise the coherence of the notebook structure. The computational nature of code cells inherently establishes a clear routine for the navigation of learning materials or activities, as shown in Figure 1. In this regard, the underlying structure of computational notebooks aligns well the concept of "implicit scripts" discussed earlier—if working in group, the first example implicitly suggests completing the tasks one after another collectively; the script for the second example could be interpreted as: first working on Task 1, then handling Tasks 2 and 3 separately, and finally working on Task 4 together". Here, the collaboration strategies are implicitly scripted by the notebook structures.

Figure 1

Two Examples of Computational Notebooks with Different Structures



Research questions

To investigate the existence of implicit scripts, we seek to uncover how the predefined structures of computational notebooks can be strategically interpreted and utilized to guide collaborative learning. By analyzing the collaborative programming activities within computational notebook environments, our research aims to explore how the inherent structural potential can be transformed from an implicit script to an explicit collaborative learning strategy, as well as how such implicit scripts can be augmented to support learners in leveraging the underlying scaffolds. Specifically, following research questions drive the focus of our study:

- RQ1: Can we derive an implicit script from the cell dependencies within a Jupyter notebook to inform strategies for effective collaborations?
- RQ2: To what extent do students' actual collaboration behaviors align with these implicit scripts?
- RQ3: In what way can we support the explication of implicit scripts to structure collaborations?

Methods

Research context

The university at the center of our research is a European research institute specializing in STEM disciplines. Exercise sessions (lab sessions) are part of most courses, complementing lectures with practical activities. This study was conducted in a bachelor-level mathematics course. All exercises were Jupyter notebooks (a commonly used type of computational notebook) and needed to be completed in groups of three or four. Six out of thirteen exercise sessions were graded, while the best five scores of these assignments were counted into the final grade. Individual absence would be graded as zero.

Ungraded exercises started with basic Python programming, and progressively advanced to interpolation, differential equations, etc., which were for students to familiarize with implementing numerical analysis methods through Python. Then every two or three weeks, an assignment was released as a formative assessment on both their theoretical understanding (by open questions) and practical skills (through coding). 1 hour 45 minutes were allocated for all groups to complete the assignments, whose topics are summarized in Table 1.

Table 1

Topics of Six Assignments

Assignment	Covered Topic	Release Week
1	Representation of Numbers	Week 2
2	Non-linear Equations	Week 4
3	Lagrange Interpolation	Week 7
4*	Numerical Integration	Week 9
5*	Preconditioned Gradient Method and Direct Methods	Week 12
6	Fast Fourier transform (FFT)	Week 14

*Note: In weeks 9 and 12, to encourage students to plan their collaboration before coding, the instructor released only a non-interactive version of the notebook for the first 10 minutes.

Participants and data collection

71 undergraduates who enrolled in the course consented to have their data collected. Students formed their groups in Week 1, resulting in a total of 22 groups (excluding one student who worked alone), and remained in the same group for the rest of the semester. All of them participated in our study from Week 2 until Week 14. These groups were expected to practice with the ungraded exercises without time constraint and complete the graded ones during the exercise sessions.

A learning analytics system, "Jupyter Analytics" (Cai et al., 2025), was installed since the second week of the course. This enabled the collection of students' interactions with the notebooks. We observed these exercise sessions and took field notes. Additionally, we attended the weekly teaching-team meetings before each exercise session, discussed with the instructor and 5 TAs (3 females, 2 males; 3 PhD students, 2 MSc students), and documented their lesson plans.

In week 12 after the exercise session, we retrieved the data of three preselected groups and conducted an interview with them. Each participant was compensated for their time and effort. The interview questions focused on how they planned and organized their teamwork, e.g., "What was your plan? How did you distribute your teamwork?"; the interviewees were also asked to interpret the timeline plots generated from the assignment they just completed, to recall more details about their collaboration strategies. This study received ethical approval from our university.

Data analysis

Four assignments (1, 2, 3, and 5) were retained for further analysis due to partial data loss in the others during collection.

RQ1: Deriving implicit scripts from notebook cell dependencies to inform collaborative strategies

We extracted both the definitions and usage of all variables and functions, and then created Directed Acyclic Graphs (DAGs) showing the code cell dependencies of each notebook. In addition, we used different colors to highlight each section of the assignment that includes at least one code cell, as defined by the teacher in the notebook. It should be noted most of these sections were concluded with a graded markdown cell for students to address open questions.

RQ2: Examining the alignment between student collaboration and the implicit scripts

Based on the data from student's interactions with the notebooks, we grouped the code cells and the corresponding markdown cells according to the generated DAGs. Next, we extracted all groups' cell execution events, labeled them with assigned section numbers, and excluded consecutive executions within a section, leaving only between-section transitions. For instance, if a group executed a code cell in Section 1 (S1), followed by a subsequent code execution in Section 2 (S2), this would constitute a valid transition to be analyzed as "S1 → S2".

Further, we used the proportion of "ideal transitions" as a proxy to measure how well these transitions aligned with the collaboration strategies derived from RQ1. The "ideal transitions" between sections could be obtained from DAGs and notebook structures. If sections are in sequence, the ideal transition that aligns with the preferred collaboration strategy is deterministic: from one section to another; no backward transition. Conversely, if students work on two or more sections simultaneously, there could be bidirected transitions between these sections. Taking Assignment 1 as an example, seven ideal transitions for the preferred strategy should be observed within a group: S0 (library importing) → S1, S0 → S2, S1 → S2, S2 → S1, S1 → S3, S2 → S3, S3 → S4. Similarly, for Assignments 2, 3, and 5, there would be 7, 4, and 6 ideal transitions, respectively. In addition, we triangulated the results by reporting interview transcripts and observation notes collected from three groups: A4, B3, and C1.

RQ3: A strategy for facilitating structured collaboration informed by implicit scripts

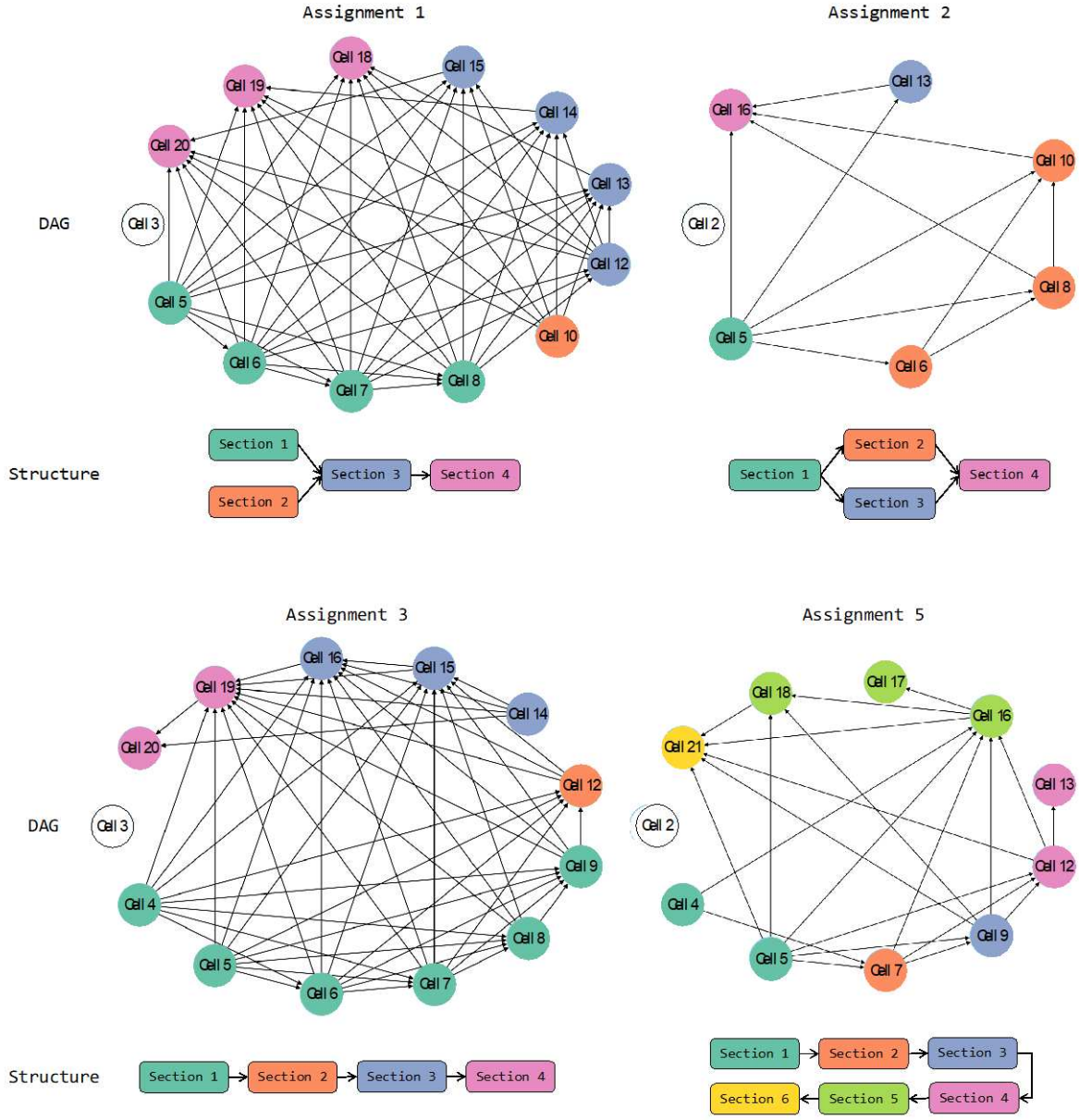
Because an intended planning phase was introduced by the instructor in Assignments 4 and 5, here we treated Assignments 3 and 5 as pre- and post-intervention condition, respectively. A paired t-test was conducted to evaluate within the same groups whether there was a significant difference in the alignment of collaboration strategies with those derived from RQ1.

Results

RQ1

Based on the cell dependencies, the extracted DAGs of four assignments are shown in Figure 2. The isolated vertices are the first code cell of each notebook given by the teachers, which handles library imports. Notably, these vertices should be included and usually connected to all the other ones if strictly analyzing the code cell dependencies—here we remove all these existing edges to facilitate readability.

Figure 2
Extracted DAGs and Structures of Four Assignments



At the section level, we could determine whether working on multiple sections simultaneously is possible by checking if they have in-edges from the preceding one. For instance, in Assignment 1: Section 2 is consisted of only cell 10 whose in-degree is zero (i.e., no in-edge), therefore learners can work on Sections 1 and 2 in parallel. However, both Sections 3 and 4 can only be started after the completion of their preceding sections, thus it would be hard to keep using the same collaboration strategy. The collaboration strategies that are implicitly scripted by the above graphs can be summarized as follows (after importing the libraries):

- Assignment 1: students can split the task and work on Sections 1 and 2 in parallel, and then the whole group complete Sections 3 and 4 together.
- Assignment 2: the whole group should first work on Section 1, and then Sections 2 and 3 can be done in parallel, while at the end they should converge on Section 4.
- Assignment 3: the whole group must work sequentially for the entire notebook – there is no way to skip any proceeding sections.

- Assignment 5: although this notebook has more sections than the others, the collaboration strategy should be the same as Assignment 3 because it also follows a serial form.

RQ2

In total, after excluding groups that did not execute all the sections (e.g., unconsented student), there were 9624 cell execution records and 3487 transitions in the dataset being analyzed for these assignments, as summarized in Table 2.

Table 2

Total Number of Transitions and Cell Execution Records Included for Analysis

	Assignment 1 (N=20)	Assignment 2 (N=19)	Assignment 3 (N=19)	Assignment 5 (N=20)
No. of Transitions	928	722	833	1004
No. of Executions	3038	2078	2279	2229

The proportion of observed ideal transitions for each group was calculated and presented in the boxplots, as illustrated in Figure 3. In addition, Table 3 shows the descriptive statistics for the same measurement. These results indicate that overall, Assignments 1, 2, and 5 had higher proportion of ideal transitions compared to Assignment 3, suggesting potential better alignment with the implicit script. Meanwhile, Assignment 5 deviated the most, which exhibited a wider range of values and several outliers. Notably, a gradual decrease in alignment from Assignment 1 to Assignment 3 could be seen from the plots, and this was in line with the instructor's observations, which led to the intervention to encourage planning in Assignments 4 and 5.

Figure 3

Distribution of Observed Ideal Transition Proportion

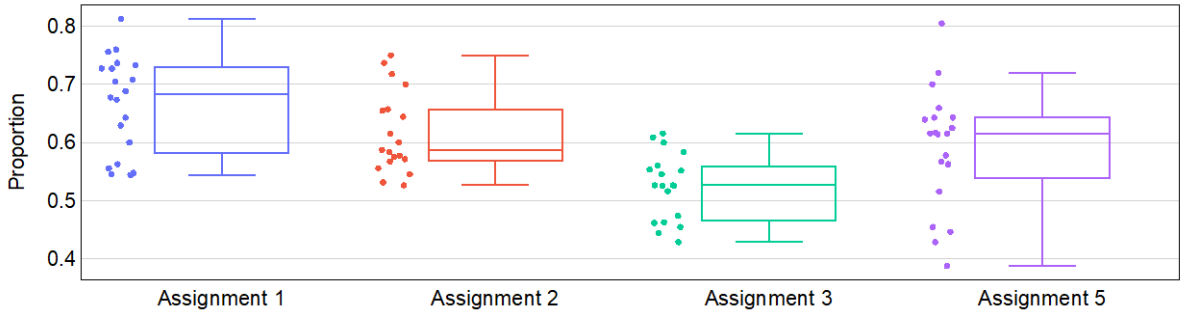


Table 3

Descriptive Statistics of Observed Ideal Transition Proportion

	Assignment 1 (N=20)	Assignment 2 (N=19)	Assignment 3 (N=19)	Assignment 5 (N=20)
Mean ± SD	0.67 ± 0.08	0.62 ± 0.07	0.52 ± 0.06	0.59 ± 0.10
Minimum	0.54	0.53	0.43	0.39
Maximum	0.81	0.75	0.62	0.80

To validate and enrich our quantitative findings, we also incorporated qualitative data that provides additional insight into how students approached their teamwork. Among all that were included into the above analysis, Group B3 in Assignment 5 received the lowest proportion score (0.39). During the interview, one student from this group described their strategy: "we don't have like a precise plan. We just go with the flow." Although the group named this "improvisation", they "usually do two teams of two people, and every team starts with a different part." In this assignment, they thought "exercise five [i.e., the last section] was kind of independent… then we start with the five… the two of them started with exercise one." However, this caused some problems that made them stuck, as noted by the student: "we took more time on exercise five because we had some problems. They continued going on, we stayed on. And so we just went with whatever was happening at the moment." Such collaboration strategies somehow meant "suboptimal" to them, as concluded by the same student. Indeed, as revealed by Figure 2, the last section heavily relied on the preceding ones. This might probably explain why Group B3's strategy did not align well with the implicit script.

By contrast, the other two groups that we interviewed had much clearer strategies. Group C1 worked in pairs, while one student handled the programming part: "I'm good at coding. So usually I do the coding heavy stuff." Group A4 adopted another strategy—they collaborated with only one notebook opened, one programmed while "the other two usually look at the lesson and the PDFs, etc." For the same assignment, their interaction data resulted in proportion scores of 0.61 (C1) and 0.64 (A4), which were above the average value.

RQ3

A paired t test was performed on the proportion scores to compare the within-subject differences between Assignments 3 and 5. We included 18 groups, whose transitions were observed in all sections of both assignments, into analysis. The test revealed a statistically significant increase in proportion scores, $t\,(17) = 2.6$, $p = .019 < .05$. The mean difference was 0.073 (95% CI [0.01, 0.13]), with a moderate to large effect size (Hedges' $g = 0.86$). These results suggest that the intervention had a significant positive effect on the collaboration strategy alignment with the implicit script.

In addition, we also observed that the numbers of groups in which only one notebook was opened were different. Based on the group interaction data, there were six other groups employed the same strategy as Group A4 in Assignment 5 (i.e., seven in total), while the numbers for the others were: zero (Assignment 1), one (Assignment 2), and one (Assignment 3). The strategy was seen as the optimal way to collaborate within a notebook that owned a sequential implicit script, like Assignment 5. As stated by Group A4, "because each exercise [i.e., section] depends on the one before it. And we can't split the code. It doesn't make sense." Interestingly, we found that in the previous assignments there were two notebooks opened for the same group. This could imply that, because of the introduction of a 10-minute planning phase, they have learned and adapted to the implicit script accordingly.

Moreover, the above qualitative observations might help explain the high proportion scores in both Assignments 1 and 2, since working in parallel matched part of the preferred strategies for these assignments. However, the same strategy could not be implemented for Assignment 3, which resulted in a decrease of scores.

In summary, the above findings suggested that the instructor's intervention achieved its intended effect on improving students' planning behavior, and thus led to better correspondence with the implicit script of the notebook.

Discussion

Our findings contribute to the field of CSCL by introducing the implicit–explicit dimension and advancing the concept of implicit scripts, thereby extending the existing theoretical understanding of scripting. In this study, computational notebooks served as a prime example: while students' teamwork was not explicitly scripted, the implicit scripts for collaborative programming strategies were revealed through an analysis of dependencies among sections in the notebooks' underlying structure. This suggests that educators could potentially shift their focus from designing explicit scripts to emphasizing learning design itself—particularly when the structure of the content inherently informs and supports the intended problem-solving approaches and, by extension, the learning objectives, provided these approaches are pedagogically meaningful.

Moreover, in response to RQ1 and RQ2, we developed a DAG-based method and corresponding metric to evaluate students' adherence to a script (whether implicit or explicit). Both quantitative and qualitative results indicated diverse collaboration strategies employed by students, which could be influenced by the changes in the task structure. These changes—from a parallel-included structure to a solely sequential format—resulted in greater deviations from the implicit script, highlighting the dynamic interplay between task design and group behavior during collaborative activities.

While implicit scripting leaves enough flexibility for students, challenges arose in some groups that adopted ineffective collaboration strategies, leading to issues such as idle time and disorganized efforts. These were particularly evident when a single strategy was uniformly applied across varied tasks. Importantly, our study found that providing students with dedicated planning time could address such problems by improving their adherence to implicit scripts, addressing RQ3. This also aligns with the Self-Regulated Learning (SRL) theories.

Implications

The findings of this study highlight several pedagogical implications for improving collaborative programming and broader learning activities:

First, while scripting could be done implicitly, its effectiveness might be enhanced by introducing a dedicated planning phase. This SRL-based approach could encourage more effective collaboration strategies and reduce inefficient interactions or unnecessary waiting time while remaining the implicit nature of such scripts. Additionally, explicating implicit scripts by extracting and visually presenting its structure—for example, through

a DAG of a pre-designed computational notebook—may further improve learner comprehension and group coordination. Such visualizations could serve as cognitive scaffolds, making the task structure more transparent.

Second, the implication above is not limited to collaborative learning activities using computational notebooks. We suggest that it could be a more general pedagogical approach for those using implicit scripts in their work. For instance, collaborative academic writing tasks inherently follow an implicit workflow or script, in which learners cannot draft a conclusion without first reporting or discussing their results. By identifying and designing around these implicit structures, educators can enhance collaboration and learning outcomes across diverse activities.

Finally, the technique we used to extract and represent the implicit scripts might have the potential to inform further development of adaptive collaboration scripts. This would be important to support those who needs more scaffold to internalize such scripts according to the Zone of Proximal Development (Vygotsky, 1978), which has been one of the foundational theories for research on collaboration scripts. Moreover, the study points to the potential for designing orchestration tools that enable teachers to intervene and adapt implicit scripts in real time. Such tools could provide educators with the ability to dynamically adjust tasks, clarify structures, and address challenges as they arise during collaborative learning activities. This capability would not only improve task effectiveness but also empower teachers to tailor activities to learners' needs.

Limitations and future work

This study has several limitations. First, the limited number of assignments available for comparison might restrict the generalizability of our findings; in particular, the data loss in Assignments 4 and 6 hindered further testing of the intervention effects when first introducing and after fading out the scaffold. Technical constraints also impacted the way how students collaborated: the computational notebooks platform in our study lacked a collaborative version, which prevented synchronized work like Google Docs and possibly influenced students' collaboration strategies. The findings could have been enriched if verbal communication or behavioral observations had been recorded, although we already conducted interviews with some students to validate our interpretations.

Future research could expand on the examined effectiveness of implicit scripts combined with a planning phase in supporting collaborative programming. Specifically, it would be valuable to explore whether introducing subsequent SRL phases, such as performance and reflection, could yield comparable learning benefits in the same context. Moreover, it is worth conducting experiments to explore the relationship between learner's domain-specific prior knowledge and their ability to explicate implicit scripts, drawing a parallel to the internalization of external collaboration scripts observed in prior research (Kollar et al., 2007; Stegmann et al., 2012).

Conclusion

The study explores the presence of implicit scripts embedded in the computational notebook structure and examines whether such scripts affect students' collaboration strategies using mixed methods. The findings indicate that the implicit script of a computational notebook can be uncovered using a graph-based approach, and an effective collaboration strategy can be informed by the DAG representation. We compared the task transitions with the DAGs to measure 22 groups' adherence to the implicit scripts, and it was found that most groups followed the scripts closely for Assignments 1 and 2. while some misalignment between their actual strategies and the optimal one was observed in certain groups. This misalignment became more pronounced in Assignment 3, resulting in unproductive collaboration behaviors. This was consistent with the instructor's observation of such unexpected dynamics, which prompted the decision to add a 10-minute planning phase before Assignment 4. Importantly, we found a significant difference in strategy adherence to the implicit scripts between Assignments 3 and 5, and this highlighted the effectiveness of the SRL-based intervention. Follow-up interviews with three groups triangulated these finding and enriched our understanding of the group collaboration strategies identified from their interaction data. Overall, the study provides evidence to operationalize the concept of implicit scripts and demonstrates how such scripts can be strategically enacted to scaffold effective collaborations.

References

Cai, Z., Davis, R. L., Mariétan, R., Tormey, R., & Dillenbourg, P. (2025). Jupyter Analytics: A Toolkit for Collecting, Analyzing, and Visualizing Distributed Student Activity in Jupyter Notebooks. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*, 172-178.

Dillenbourg, P. (2002). Over-scripting CSCL: The risks of blending collaborative learning with instructional design. *Three worlds of CSCL. Can we support CSCL/Open Universiteit Nederland*.

Dillenbourg, P., & Hong, F. (2008). The mechanics of CSCL macro scripts. *International Journal of Computer-Supported Collaborative Learning*, *3*, 5-23.

Dillenbourg, P., & Jermann, P. (2007). Designing Integrative Scripts. In Scripting computer-supported collaborative learning, pages 275–301. Springer US.

Dillenbourg, P., & Tchounikine, P. (2007). Flexibility in macro-scripts for computer-supported collaborative learning. *Journal of computer assisted learning*, *23*(1), 1-13.

Fischer, F., Kollar, I., Stegmann, K., & Wecker, C. (2013). Toward a script theory of guidance in computer-supported collaborative learning. *Educational psychologist*, *48*(1), 56-66.

Kollar, I., Fischer, F., & Hesse, F. W. (2006). Collaboration scripts–a conceptual analysis. *Educational Psychology Review*, *18*, 159-185.

Kollar, I., Fischer, F., & Slotta, J. D. (2007). Internal and external scripts in computer-supported collaborative inquiry learning. *Learning and Instruction*, *17*(6), 708-721.

Lingnau, A. & Bientzle, M. (2009). A technical framework to support implicit structured collaboration. In *Computer Supported Collaborative Learning Practices: CSCL2009 Conference Proceedings* (pp. 527-531).

Runde, A., Bromme, R., & Jucks, R. (2007). Scripting in net-based medical consultation: The impact of external representations on giving advice and explanations. *Scripting computer-supported collaborative learning: Cognitive, computational and educational perspectives*, 57-72.

Stegmann, K., Mu, J., Gehlen-Baum, V., & Fischer, F. (2011). The Myth of Over-scripting: Can Novices be Supported too much?. In *Connecting Computer-Supported Collaborative Learning to Policy and Practice: CSCL2011 Conference Proceedings. Volume I — Long Papers* (pp. 406-413)

Stegmann, K., Wecker, C., Weinberger, A., & Fischer, F. (2012). Collaborative argumentation and cognitive elaboration in a computer-supported collaborative learning environment. *Instructional Science*, *40*, 297-323.

Tchounikine, P. (2008). Operationalizing macro-scripts in CSCL technological settings. *International Journal of Computer-Supported Collaborative Learning*, *3*, 193-233.

Tchounikine, P. (2016). Contribution to a theory of CSCL scripts: Taking into account the appropriation of scripts by learners. *International Journal of Computer-Supported Collaborative Learning*, *11*, 349-369.

Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes* (Vol. 86). Harvard university press.

Vogel, F., Weinberger, A., & Fischer, F. (2021). Collaboration scripts: Guiding, internalizing, and adapting. *International handbook of computer-supported collaborative learning*, 335-352.

Acknowledgments

The project was funded by Swiss National Science Foundation (SNSF) in the National Research Program “Digital Transformation” (NRP 77) under grant number 407740_187534. We also acknowledge the continuous support of the course instructors, teaching assistants, and students throughout the project.