Semantic Waves: A Strategy for Algorithmic Skills in K-12 Computer Science Education

Frauke Ritter
Department for Computer Science and Digital Education
Karlsruhe University of Education, Germany
Email: frauke.ritter@ph-karlsruhe.de

Bernhard Standl
Department for Computer Science and Digital Education
Karlsruhe University of Education, Germany
Email: standl@ph-karlsruhe.de

2024 IEEE Global Engineering Education Conference (EDUCON) | 979-8-3503-9402-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/EDUCON60312.2024.10578899

***Abstract*—The objective of this study is to develop and empirically evaluate an educational model that enhances *algorithmic thinking* — a key element of *computational literacy* — through the application with the concept of so called *semantic waves* for advancing K-12 students' digital proficiency. The concept of a *semantic wave* refers to the process of moving between abstract, theoretical knowledge and concrete, practical examples to create deeper understanding and learning. Considering this, our proposed model in the field of *algorithmic thinking* is intended to support pre-service computer science teachers and educators in designing instructional processes that are easy to implement and facilitate swift planning and reflection for K-12 computer science education. Initial results indicate promising outcomes but also suggested areas for enhancement. This research furthermore delves into refining the model through the incorporation of *notional machines* and the *computational action* approach for improving the training of future computer science teachers and students for the challenges of digital transformation.**

I. Introduction

Digital competencies refer to the skills needed to use digital technology effectively and safely in areas like education, work, and social activities. These skills are very important for students in school, especially in today's world where technology is constantly changing. They need to learn how to use technology in a smart and responsible way [1]. The skills involved in *computational thinking* and *digital competencies* overlap a lot [1]. Additionally, *algorithmic thinking*, which is a part of *computational thinking* [2], is key for helping these students understand *digital competencies* deeply. In the preparation of future computer science (CS) teachers, therefore it is also important to emphasize on teaching computational problem solving skills at school. Teachers should be able to integrate and apply these skills in their lesson-planning. Considering this, the concept of *semantic waves* [3] has been developed and tested in different subject areas to describe and reflect on planning instructional processes, but few, especially in unplugged settings such as [4] in a case study on so called *crazy characters* and [5] in two case studies (a *Teleporting Robot* and *Box Variables*), have been tested in computer science (CS) education. The study presented in this paper describes a teaching model that uses *semantic waves* to help improve students' *algorithmic thinking* and *digital skills*. In doing this, we run a *CS Teaching-Learning Lab*, where pre-service CS teachers can practice lessons in workshops with K-12 students. This supports pre-service CS teachers in designing and testing teaching methods for K-12 CS education. An initial online review suggested promising signs [6], but it also indicated the need for further development of our model. This study enhances our model with further concepts like *notional machines* [7] (flowcharts) and *computational action* [8]. The overall goal is to provide a effective framework for the training of future teachers, as discussed by [9], and to address students' competencies for a digital future.

II. Related Work

A. Computational and Algorithmic Thinking

Algorithmic thinking is part of *computational thinking*, which ultimately goes back to [10], as already noted by [2], who defined *computational thinking* as *thinking like a computer scientist*. In the following years, there have been many attempts to define the term, but they have not yet been brought together into one (e.g. [11]). In [1] many definitions are reviewed and finally they came up with eight component groups of *computational thinking* such as *data analysis and representation*, *computational artifacts*, *decomposition*, *abstraction*, *algorithms*, *communication and collaboration*, *computing and society*, and *evaluation*. Here in [1] also the high correspondence between *computational thinking* and the *digital competences* is emphasized.

Building upon computational thinking as initially proposed by [2] and further developed in for problem-solving processes by [12], this study positions *algorithmic thinking* within the broader domain of *computational thinking*. The conceptualization of algorithmic thinking is structured into three methodical phases: 1) Problem Understanding, encompassing description, abstraction, and decomposition (*UP*); 2) Problem Solving, through the design of algorithms (*SP*); and 3) Solution Analysis, involving the testing of the solution's effectiveness (*analyze*). Consequently, our research leverages a computational thinking framework for problem-solving, as evidenced in applications ranging e.g. from everyday challenges [13] to Python programming [14]. This approach underpins our methodology for educating students in problem-solving techniques, with the goal of enhancing their algorithmic thinking capabilities.

For teaching *algorithmic thinking* it is effective if students are personally engaged and activated by the problems they are asked to solve, especially if they are enabled to understand

and program and therefore manipulate digital phenomena that affect their daily lives, as required by *digital competences* [1]. According to [15], this is achieved by fostering learners' *computational identity* as well as *digital empowerment*, with *computational identity* being defined as *the recognition of a person's ability to design and implement computational solutions to self-identified problems or opportunities*. *Digital empowerment* is described as *learners' confidence that they can translate their computational identity into computational action - in contexts that truly impact their lives.* Earlier research of [8] shows the importance of covering topics in the curriculum that have a real impact on students. The *Dagstuhl Triangle* [16] characterizes digital artifacts from three different perspectives (*How does it work?, How is it used?, What is it's impact?)* can serve as idea do differentiate *digital competences*. In *How does it work?* competencies of *algorithmic thinking* are part of. A significant difference in students' self-perception can be assumed, when the topics are provided from the students' everyday digital world. That is why the concept of students workshops we design chosen as underlying idea to strengthen the *digital empowerment* and therefore the *digital identity* [8]. For instance, this is addressed by students developing an animation and a computer game on the one hand, and simulating a voice assistant on the other hand, which is part of the students' everyday life.

B. Semantic Waves

The term *semantic wave* comes from *Legitimation Code Theory*, which is a practical framework used to analyze a variety of issues, practices, and contexts in education and beyond [3]. Derived from this, semantics in this sense are *semantic gravity (SG)* and *semantic density (SD)* according to [3]:

- *Semantic gravity* (SG) refers to the degree to which meaning relates to its context. The stronger the *semantic gravity* (SG+), the more meaning is dependent on its context; the weaker the *semantic gravity* (SG-), the less dependent meaning is on its context.
- *Semantic density* (SD) refers to the degree of condensation of meaning within socio-cultural practices.(SD) The stronger the *semantic density* (SD+), the more meanings are condensed within practices; the weaker the *semantic density* (SD-), the less.

With the help of these two terms, it is possible to model a *semantic wave* (see Fig. 1).

For example, Fig. 1 shows the course of a *semantic wave* as indicated by [5] using crazy characters in an unplugged CS context. Time here is relative. At first, the content or topics to be learned are still introduced to the learner with low SG (abstract concept) and high SD (technical language), but then, for example, applied to a concrete example (SG+) with everyday language (SD-) and finally successively brought back to the original level. There is only a little work in the field of CS teaching that apply and investigate the concept of a *semantic wave* in unplugged settings, as in [4] a case study on so-called *crazy characters* investigates the concept

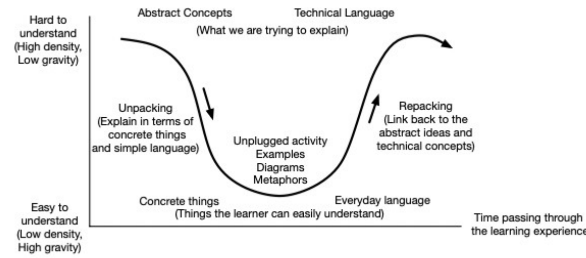


Fig. 1. Traversing a *semantic wave* by [5].

of a *semantic wave* and also in [5] using two case studies (a *Teleporting Robot activity* and a *Box Variable*). There are some applications of *semantic waves* for teaching English or even biology and history [3]. In this context, the approach of [17], which used *semantic waves* for teacher training of future English teachers by measuring and analyzing it with a translator, is notable. In [18], from the perspective of *semantic waves*, a model for the promotion of *English* teaching in higher education has been constructed. This provides a valuable reference for curricular reform and design, and a scientific basis for developing English teaching in Chinese higher education. The results of [6] indicate that a plugged approach for CS education could foster the acquisition of *algorithmic thinking*. This leads to a research gap in applying *semantic waves* to practical computing education, suggesting a promising direction for future studies.

C. Block-based programming language

Since the release of Scratch in 2007, a huge amount of practice and research has been done with block-based programming languages. In [19] and also in [20] were articles analyzed on the relationship between Scratch and *computational thinking* and both show that *computational thinking* can be taught with Scratch. Today, Scratch programmers can create stories, animations, games, music, and share their programs with the web. In his article *Programming for everyone*, [21] pointed out the main advantages of block-based languages for beginning programmers, since they do not present the syntax problems of text-based languages and, not to underestimate, students can set interesting programming tasks that go beyond prime calculations or similar. In [22] a broad overview of how Scratch has been used in different subjects to promote *computational thinking* is provided and conclude that these are promising approaches, but that quantitative data are still largely lacking at this time. From this we can conclude: Scratch is very well suited to teach *computational thinking* and therefore *algorithmic thinking* skills, but there is still a lack of quantitative data.

D. Notional machines

The idea of *notional machines* goes back to the 1970s. In [23] is a detailed overview of *notional machines* and it is defined a *notional machine* as a pedagogic device to assist the understanding of some aspect of programs or programming. In

[23] also examples of *notional machines* are examined and divided into the categories *Machine-Generated Representations* (e.g. Program visualization tools), which *usually show the state of the execution at any given step*, *Handmade Representations* (sketches, drawings - e.g. flowcharts, texts, actions) and finally *Analogies* (e.g. shoebox for a variable).

In [7] are some pedagogical recommendations for the use of *notional machines* given. They state that by using them, students become more aware of how and why a program really works, thus better understand program execution as a whole. In [23] is also the use of *notional machines* as explanatory aids to account for the learner's current level of knowledge described and to avoid unnecessary cognitive load. In particular, they increase *semantic gravity* and decrease *semantic density* (II-B) in the context of a *semantic wave* - thereby making a concept more understandable [23]. To the best of our knowledge, *notional machines* have not yet been used in the context of *semantic waves* in CS education.

E. Enhanced SWAT model

We synthesized the *semantic wave* framework [3] with the *algorithmic thinking* approach of [12] to design a model we named SWAT (Semantic Wave Algorithmic Thinking), a tool for educators to effectively plan and reflect on *algorithmic thinking* instruction within (CS) education. As it is illustrated in Figure 4, the model represents the Module flow, integrating *semantic wave* and *algorithmic thinking* across various phases. Workshops for school students, where we implement this model focus on algorithmic problems using block-based programming for K-12 learners, promoting a student-centered educational setting. Previous investigations [6] examined the efficacy of an online workshop modeled on SWAT, targeting the Pledge algorithm within a 90-minute session (see Figure 2). While the earlier study revealed no significant statistical improvement in K-12 students' *algorithmic thinking*, qualitative insights were more positive. Further, students recognized *semantic wave* phases quantitatively, but qualitative feedback was inconsistent. Based on these findings, we retained the core pedagogical strategy for this study, combining *semantic wave* with *algorithmic thinking* and using block-based programming languages (Scratch), but extended the duration to four 90-minute Modules (see Figure 2). In order to promote the students' competence in *algorithmic thinking* better than in the previous study, we refined the problem-solving steps: First, we included a *notional machine* concept [7] to promote a deeper understanding of problem solving and to encourage reflective thinking about program operations. Second, we situate all problems within the context of *computational action* [8] to strengthen students' *computational identity* and thus their *computational empowerment* through relatable, real-world scenarios.

III. Research Questions

Two research questions are addressed in this study: When students participate in a workshop designed after the so called SWAT model (Semantic Wave Algorithmic Thinking),

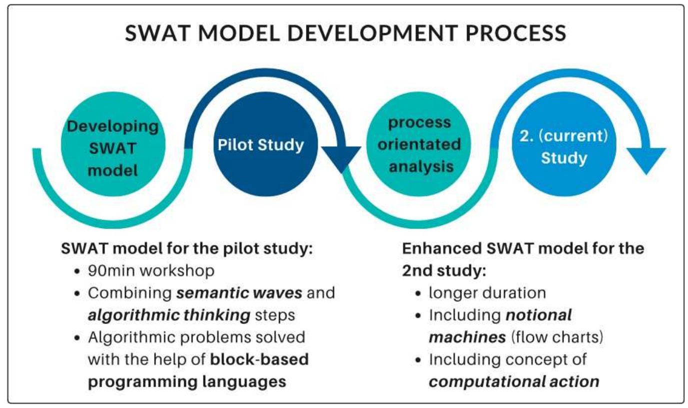


Fig. 2. Developing Process of the SWAT model.

- *RQ1: To what degree do K-12 students' competence in algorithmic thinking develop?*
- *RQ2: How are the phases of the semantic wave perceived by the students?*

RQ1 examines students' growth in *algorithmic thinking* skills, RQ2 tests whether our design using *semantic waves* is perceived as such. Operationalizing our research questions is done through a case study following a mixed methods approach [24] examining *Hypothesis 1: The SWAT model promotes K-12 students' algorithmic thinking.* and *Hypothesis 2: During each Module of the workshop, the phases of a semantic wave are perceived by the students*.

IV. Methods

A. Setting and Participants, Treatment fidelity

In this study, N=39 K-12 students aged 12-13 years, participated in an on-site workshop at our University. Before the workshop, the students had only had two computer science lessons at school in which they had learned simple coding sequences in Scratch - they were particularly unfamiliar with flowcharts in the run-up to the workshop. The control group consisted of N=38 high school students aged 13-14. The control group was one year older and had taken a small CS class the previous year learning algorithm structures (conditions, loops, sequences) in Scratch. Informed consent was obtained from the parents of the participating students in both the treatment and control groups. As it is shown in Figure 3 before and after the workshop, students in the treatment and students in a control group completed an *algorithmic thinking* test [25]. Since the prior knowledge of the control group was greater, the analysis (see section IV-C1) examined the learning gain rather than the absolute score achieved.

We also documented the *Treatment fidelity*, which identifies if an intervention is delivered as intended which is an important construct for educators when interpreting intervention research [26]. The study workshop is designed to address *treatment fidelity* categories of [27]: Each Module of the workshop records the steps students take using the worksheets and Scratch programs in phases 2-4 (*adherence*). All students

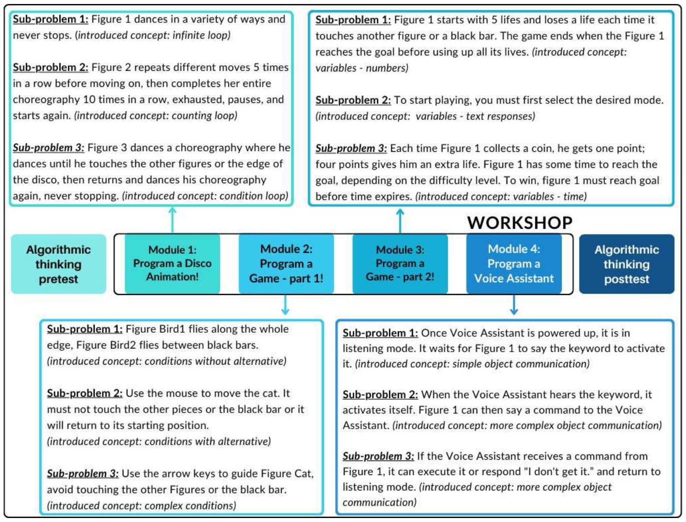


Fig. 3. Procedure of the workshop with *algorithmic thinking* pre- and posttest [25]. Each Module of the workshops is based on the SWAT model.

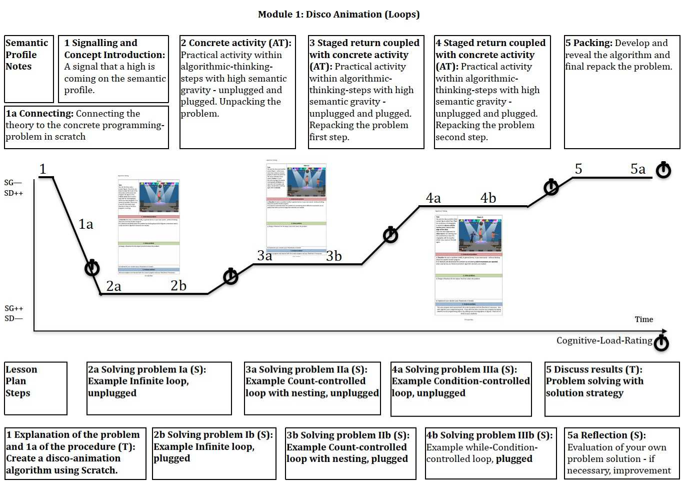


Fig. 4. The progression of a Module that follows a *semantic wave* [3] and in the phases 2-4 *algorithmic thinking* steps [12] (SWAT model). The problem is divided into three sub-problems, with students working on each sub-problem using worksheets (see Figure 5) in phases 2-4 in each Module.

complete the same tasks and have the same amount of time (*quantity*). *Process* is ensured by the whole workshop setting. During the workshop, students' cognitive load is measured IV-C2. *Quality* is addressed by an additional item where students rate their personal effort. The results are shown in Fig. 9c), 9f), 9i) and 9l), where the rating is high in all four workshop Modules, with a decrease towards the end, so the overall *quality* is very well met.

B. Materials

The workshop consists of four 90-minute Modules, all based on the SWAT model and increasing in difficulty (see Fig. 3). In each Module the students are guided in the problem-solving process by three sub-problems (*semantic wave* phases 2-4, see Fig. 3 and 4). This implements the *algorithmic thinking* steps unplugged (worksheets text answers and flowcharts) and

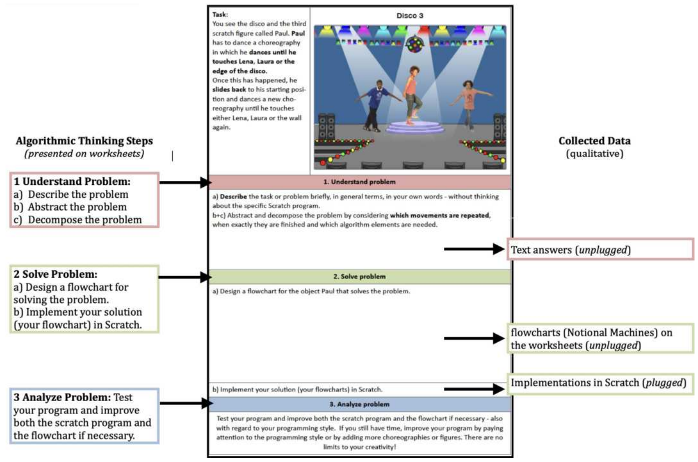


Fig. 5. Algorithmic thinking steps [12] presented on each worksheet for each sub-problem in all Modules and the qualitative data collected. For example: the worksheet for Module 1, phase 4.

plugged (block-based language programs) (see Fig. 5). Each topic of the Modules is chosen to motivate students to explore the concepts behind everyday applications in terms of [1] and *computational action* [8] (Module 1: animation (loops), Module 2: game (conditions), Module 3: game (variables), Module 4: voice assistant (object communication), see Fig. 3).

Each Module is structured according to a *semantic wave* as follows (see Fig. 4):

- Phase 1 and 1a (*signaling and concept introduction* phase): Here the teacher introduces the lesson problem and links it to a concrete problem in Scratch. This phase is designed to have a high *semantic density* (SD++) and low *semantic gravity* (SG–). For example, in Module 1: Programming a disco animation using loops.
- Phases 2, 3 and 4: The lesson problem is divided into 3 sub-problems. In each phase, using the three sub-problems, students here work individually through the steps of *algorithmic thinking* to solve the lesson problem (see example worksheet in Fig. 5). All sub-problems are shown in Figure 3. Tasks and contents are designed to correspond to the progression of *semantic density* and *semantic gravity* in terms of a *semantic wave*.
- Phase 5: The students are brought back to the starting level in the *semantic wave* (SG–,SD+). Solutions are discussed as a group and students have time to work individually.

C. Instruments and Analysis

1) Instruments and Analysis for RQ1: To what degree do K-12 students' competence in algorithmic thinking develop?: Quantitative data: Students in the treatment group (TG) and control groups (CG) completed an identical, reliable *algorithmic thinking* pre- and posttest (Cronbach's α = 0.8 good, [25]). Test items are translated, and distractors per task varied in order for posttest. The analysis examines the learning gain rather than the absolute score achieved, and is both descriptive and inferential. The informative hypothesis was $TG_{pretest} = CG_{pretest} = CG_{posttest} < TG_{posttest}$ (treatment group (TG) improved at posttest compared to pretest and control group (CG)), the null hypothesis was $TG_{pretest} = CG_{pretest} = CG_{posttest} = TG_{posttest}$ (neither group of students improved). This was evaluated with approximate adjusted fractional Bayes factors [28], using within repeated measures ANOVA as interpreted by [29]. As our sample size is N=39, we chose this statistical approach, which is robust to non-normally distributed data [30]. For effect sizes, we calculated the standardized Cohen's d and interpreted it according to [31]. The calculation used estimates for A [32]. The quantitative data was collected using a digital test. The data was analysed and the results presented using R-Studio software.

The qualitative data (full-filled worksheets and created Scratch programs, see Fig. 5) are analyzed through a qualitative content analysis according to [33], (interpretive paradigm to hypothesis 1 (III) and is differentiated in terms of *algorithmic thinking*, divided into units of analysis (phases 2, 3 and 4 of *semantic wave*). The worksheets are coded into *understand problem* (UP) and *solve problem* (SP) categories. Programs are differentiated based on both *algorithmic thinking* theory and individual categories of [34]. The developed category system uses the categories *flow control* (FC) and *logic* as they describe all used programming content. Characteristics weak, moderate and strong using for both developed category systems [35]. After coding, the team discussed and revised the data. A pre-service student was then trained to re-code the material. The Intercoder agreement in MAXQDA (segment overlap of 95%) was 88.30% (*understand problem* and *solve problem*) and 84.75% (*flow control* and *logic*). The following review included the addition of missing codes for the *understand problem* category and the importance of accurate and complete responses for strong coding. The *solve problem* category agreed that a flowchart that completely solves the problem but is not clearly drawn should be strongly coded. After revision, MAXQDA yielded an intercoder agreement of 99.25% (*understand problem* and *solve problem*) and 99.80% (*flow control* and *logic*) with 95% segment overlap. The qualitative data was analysed using MAXQDA software. The results were presented using a spreadsheet program.

2) Instruments and Analysis for RQ2: How are the phases of the semantic wave perceived by the students?: For the quantitative data the *cognitive load theory* (CL) is used to get a sense of how the phases of a *semantic wave* are perceived by students.

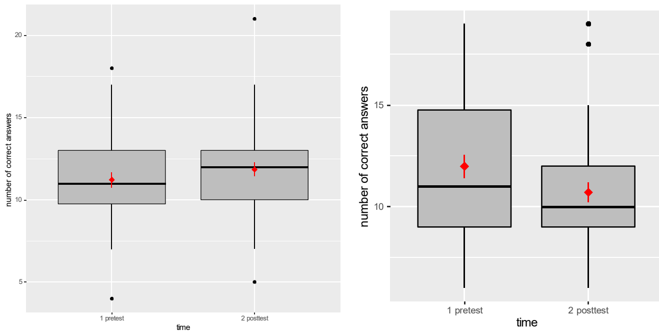
The subjective cognitive load approach assumes that working memory is limited and long-term memory nearly infinite [36]. These basic assumptions about human cognitive architecture have implications for successful teaching and learning. The *intrinsic cognitive load* (ICL) is the load that results from the inherent complexity of the learning task, and the *extraneous cognitive load* ECL is the load that results from the instructional design of the learning content [37], [38]. Based on our workshop design (see IV-B), it suggests that high *semantic density (SD)* (technical language formulations) and low *semantic gravity (SG)* (abstract concepts) require higher ICL than low SD (everyday language) and high SG (real-world examples). The survey developed by [39] is particularly suitable for learning environments that use digital interactive learning media [40]. Thus, since Scratch is an interactive learning environment, this questionnaire (paper pencil) is used to collect data (Cronbach's $\alpha = 0.81$ (good) for ICL and 0.86 (good) for ECL [39]). Figure 4 shows the *cognitive load* rating times for each Module five times.

In the test of [39], two items ask about ICL and three items ask about ECL, each on a 7-point Likert scale, where a 1 means ICL/ECL is low and a 7 means ICL/ECL is high.

The adopted informative hypothesis for ICL was $ICL1 > ICL2 < ICL3 < ICL4 < ICL5$ (students' ratings of their ICL correspond to a wave-like arrangement), the null hypothesis was $ICL1 = ICL2 = ICL3 = ICL4 = ICL5$ (no differences in students' ICL ratings) with ICL1 the ICL measurement at time 1 (Means of both ICL items in the survey) and correspondingly ICL2 to ICL5.

In addition, students assessed their personal ECL. The tasks and worksheet design are such that ECL should decrease over the course of the Modules IV-B. Thus, the adopted informative hypothesis ECL was: $ECL1 > ECL2 > ECL3 > ECL4 > ECL5$, where ECL1 means ECL measurement at time 1 (means of three ECL questionnaire items) and correspondingly ECL2 to ECL5. The point-zero hypothesis was: $ECL1 = ECL2 = ECL3 = ECL4 = ECL5$ (ECL remains the same). The quantitative data for ICL and ECL in each Module was analyzed with the approximate adjusted fractional Bayes factors [28], using within repeated measures ANOVA as interpreted by [29]. We chose this statistical approach, which is robust to non-normally distributed data [30]. For effect sizes, we calculated the standardized Cohen's d and interpreted it according to [31]. The calculation used estimates for A [32]. The quantitative data were analysed and the results illustrated using R-Studio software.

For the qualitative data, (textual responses on the worksheets in the *algorithmic thinking* step *understand problem* (UP)), a summary qualitative content analysis following the [33] interpretive paradigm is used. The analysis focuses on hypothesis 2 (III). The question is structured according to the theory of a *semantic wave*, the units analyzed for each Module are phases 2-4 of the *semantic wave*. A category system is developed [35], the material is coded with weak, medium, and strong characteristics. In terms of a *semantic wave*, the students' responses in the developed category system should be weak in phase 2 (SD–, SG++), medium in phase 3 (SD-, SG+), and strong in phase 4 (SD+, SG-). The team discussed and revised the coded material, then trained a pre-service student to code according to the category system. The intercoder agreement in MAXQDA was 90.10% (segment overlap 95%). The coding was then revised by adding text passages forgotten by both coders and harmonizing synonyms in the category system. In order to pay more attention to the use of technical terms, the category system was revised. After the revision, the intercoder



(a) Treatment group (b) Control group

Fig. 6. Results of the *algorithmic thinking* pre- and posttests [25] with Mean and Standard Deviation.

Fig. 7. Cumulative relative frequencies of the *algorithmic thinking* categories *understand problem* (UP) and *solve problem* (SP) in the characteristics weak, medium and strong, phases 2-4 in Modules 1-4.

agreement in MAXQDA is 99.34% (95% segment overlap). The qualitative data was analysed and the results illustrated using MAXQDA software.

V. Results

A. Results for RQ1: To what degree do K-12 students' competence in algorithmic thinking develop?

The **quantitative data** evaluation (*algorithmic thinking* pre- and posttest) revealed a slight decrease in Mean, Median and Standard Deviation for the control group (CG) (see Fig. 6, $Mean_{pretest}$ = 12.0, $Mean_{posttest}$ = 10.7, $Standard\ Deviation_{pretest}$ = 3.5, $Standard\ Deviation_{posttest}$ = 3.0, $Median_{pretest} = 11$, $Median_{posttest} = 10$).

In contrast, the treatment group (TG) displayed stable Standard Deviation and marginal increases in Mean and Median (see Fig. 6, $Mean_{pretest}$ = 11.1, $Mean_{posttest}$ = 11.8, $Standard\ Deviation_{pretest}$ = 2.92, $Standard\ Deviation_{posttest}$ = 2.67, $Median_{pretest} = 11$, $Median_{posttest} = 12$).

The effect sizes showed a weak effect for an improvement in the treatment group (TG) and a weak effect for worsening in the control group (CG) between pre- and posttest ($d = -0.3$ for TG, $d = 0.4$ for CG, [31]).

Inferential statistics showed no significant findings for the informative hypothesis (1.4 add that and more times more likely, [29]).

The results of the **qualitative** content analysis using the developed category system are shown in Figures 7 and 8 and Table I. Figure 7 shows that the textual responses on the worksheets in the *understand problem* category were predominantly weakly categorized, with the exception of Module 2.

The *solve problem* category, where students had to create flowcharts (notional machines), shows a more differentiated picture. In Module 1 and Module 4, the proportion of moderately and strongly categorized flowcharts increases compared to weakly categorized flowcharts. However, the proportion of flowcharts strongly classified in the *solve problem* category is predominantly greater than the percentage of text answers in the *understand problem* category, especially in Modules 2 and 4. The only exceptions are Module 2, Phase 4 and Module 3, Phase 2.

The absolute responses (see Table I) in Module 3, but also in Module 2, Phase 4, are low in both categories, but especially in the *solve problem* category.

The programming solutions (see Fig. 8 and Tab.I) in Module 1 start predominantly with the strong category in phase 2 (*flow control* (FC) 96% and *logic* 89.7%), but then deteriorate in phases 3 and 4.

In Module 2 and 3, the solutions are also predominantly in the strong category in phases 2 and 3 (2: *logic* and *flow control* 90% and more; 3: *logic* and *flow control* between 75% and 90%), in Module 3 the category increases in phase 3 (see Fig. 8 and Tab.I).

In Module 4, the strongly coded portions of the *logic* and *flow control* decrease only slightly over the course of phases 2 to 4 (see Fig. 8 and Tab.I).

In each Module the third sub-problem (phase 4) was worked on less (absolute numbers see Tab. I) and solved worse (see Fig. 8) - especially visible in Module 3.

In summary, it can be said that the students improved in *algorithmic thinking*, especially in *problem solving*, *logic* and *flow control* in Module 1, as well as in Module 2 in phases 2 and 3. In the *logic* and *flow control* categories, students maintained their high performance in Module 3, but dropped slightly in Module 4, although they improved here in *problem solving*.

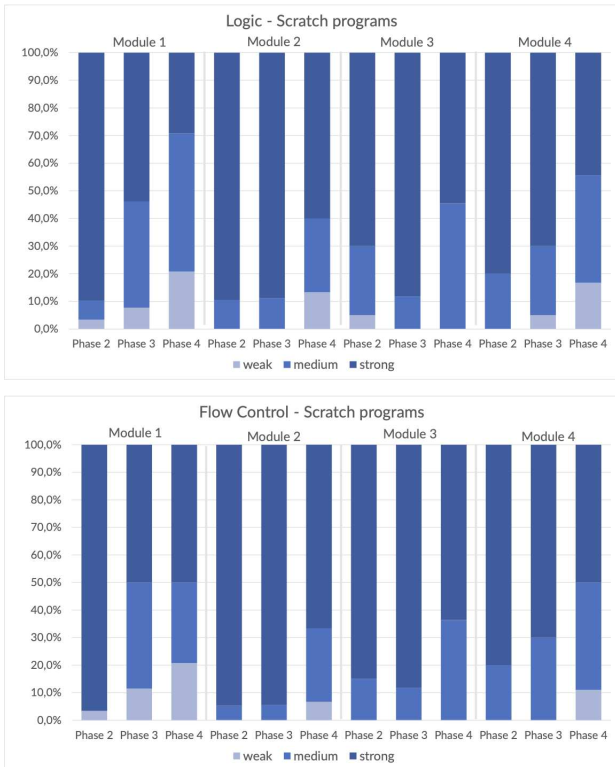


Fig. 8. Cumulative relative frequencies of the *algorithmic thinking* categories *logic* and *flow control* in the characteristics weak, medium and strong, phases 2-4 in Modules 1-4.

B. Results for RQ2: How are the phases of the semantic wave perceived by the students?

The evaluated **quantitative** data show a wave-like illustrative progression of **ICL** measurements for all four Modules (Fig. 9). Analyzing the students' ratings yields statistically relatively strong significance for the informative hypothesis in each Module (Module 1:$1.3 \cdot 10^{34}$; Module 2: $1.9 \cdot 10^{78}$; Module 3:$3.4 \cdot 10^{110}$; Module 4: $6.6 \cdot 10^{47}$ add that and more times more likely, [29]).

The effect size values (standardized Cohan's d) underscore this result of the ICL ratings, as there are strong differences [31] between the mean values in each Module from time 1 to time 2 (Module 1: $d = 2.4$; Module 2: $d > 4$; Module 3: $d = 2.7$; Module 4: $d = 2.8$) and 3 (Module 1: $d = 1.6$; Module 2: $d = 2.3$; Module 3: $d = 2.6$; Module 4: $d = 1.7$), and from time 2 to 3 (Module 1: $d = -1.7$; Module 2: $d = -3.7$; Module 3: $d = -1.9$; Module 4: $d = -1.7$) and 4 (Module 1: $d = -2.4$; Module 2: $d < -4$; Module 3: $d = -2.6$; Module 4: $d = -2.6$). The decrease in ICL measurement at time point 5 (Fig. 9), is also reflected in the effect size values, as between the mean values of time points 4 and 5, the effect sizes are each positive and moderate to strong (Module 1: $d = 0.7$; Module 2: $d = 1.3$; Module 3: $d = 0.8$; Module 4: $d = 0.9$).

For all four Modules of the workshop, Fig. 9 shows a decreasing course of **ECL** measurements in phases 1-4 and a smaller or larger increase in phase 5, depending on the

Codes	Module 1.1	Module 1.2	Module 1.3	Module 2.1	Module 2.2	Module 2.3
UP strong	0 (0.0%)	0 (0.0%)	0 (0.0%)	7 (22.6%)	10 (32.3%)	2 (11.1%)
UP medium	23 (62.2%)	11 (39.3%)	9 (52.9%)	11 (35.5%)	15 (48.4%)	13 (72.2%)
UP weak	14 (37.8%)	17 (60.7%)	8 (47.1%)	13 (41.9%)	6 (19.4%)	3 (16.7%)
Sum UP	37 (100.0%)	28 100.0%	17 (100.0%)	31 (100.0 %)	31 (100.0%)	18 (100.0%)
SP strong	7 (18.4%)	0 (0.0%)	2 (11.1%)	18 (58.1%)	13 (41.9%)	0 (0.0%)
SP medium	8 (21.1%)	15 (48.4%)	10 (55.6%)	8 (25.8%)	10 (32.3%)	2 (28.6%)
SP weak	23 (60.5%)	16 (51.6%)	6 (33.3%)	5 (16.1%)	8 (25.8%)	5 (71.4%)
Sum SP	38 (100.0 %)	31 (100.0%)	18 (100.0%)	31 (100.0 %)	31 (100.0%)	7 (100.0%)
Logic strong	26 (89.7%)	14 (53.8%)	7 (29.2%)	17 (89.5%)	16 (88.9%)	9 (60.0%)
Logic medium	2 (6.9%)	10 (38.5%)	12 (50.0%)	2 (10.5%)	2 (11.1%)	4 (26.7%)
Logic weak	1 (3.4%)	2 (7.7%)	5 (20.8%)	0 (0.0%)	0 (0.0%)	2 (13.3%)
Sum Logic	29 (100.0 %)	26 (100.0%)	24 (100.0%)	19 (100.0 %)	18 (100.0%)	15 (100.0%)
FC strong	28 (96.6%)	13 (50.0%)	12 (50.0%)	18 (94.7%)	17 (94.4%)	10 (66.7%)
FC medium	0 (0.0%)	10 (38.5%)	7 (29.2%)	1 (5.3%)	1 (5.6%)	4 (26.7%)
FC weak	1 (3.4%)	3 (11.5%)	5 (20.8%)	0 (0.0%)	0 (0.0%)	1 (6.7%)
Sum FC	29 (100.0 %)	26 (100.0%)	24 (100.0%)	19 (100.0%)	18 (100.0%)	18 (100.0%)
Codes	**Module 3.1**	**Module 3.2**	**Module 3.3**	**Module 4.1**	**Module 4.2**	**Module 4.2**
UP strong	1 (3.7%)	0 (0.0%)	0 (0.0%)	1 (3.1%)	2 (7.7%)	0 (0.0%)
UP medium	11 (40.7%)	10 (37.0%)	6 (35.3%)	12 (37.5%)	6 (23.1%)	4 (30.8%)
UP weak	15 (55.6%)	17 (63.0%)	11 (64.7%)	19 (59.4%)	16 (69.2%)	9 (69.2%)
Sum UP	27 (100.0%)	27 (100.0%)	17 (100.0%)	32 (100.0%)	26 (100.0%)	13 (100.0%)
SP strong	0 (0.0%)	0 (0.0%)	0 (0.0%)	9 (26.5%)	10 (50.0%	2 (33.3%)
SP medium	1 (25.0%)	3 (75.0%)	1 (50.0%)	15 (44.1%)	5 (25.0%)	3 (50.0%)
SP weak	3 (75.0%)	1 (25.0%)	1 (50.0%)	10 (29.4%)	5 (25.0%)	1 (16.7%)
Sum SP	4 (100.0%)	4 (100.0%)	2 (100.0%)	34 (100.0%)	20 (100.0%)	6 (100.0%)
Logic strong	14 (70.0%)	15 (88.2%)	6 (54.5%)	16 (80.0%)	14 (70.0%)	8 (44.4%)
Logic medium	5 (25.0%)	2 (11.8%)	5 (45.5%)	4 (20.0%)	5 25.0%	7 (38.9%)
Logic weak	1 (5.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	1 (5.0%)	3 (16.7%)
Sum Logic	20 (100.0%)	17 (100.0%)	11 (100.0%)	20 (100.0%)	20 (100.0%)	18 (100.0%)
FC strong	17 (85.0%)	15 (88.2%)	7 (63.6%)	16 (80.0%)	14 (70.0%)	9 (50.0%)
FC medium	3 (15.0%)	2 (11.8%)	4 (36.4%)	4 (20.0%)	6 (30.0%)	7 (38.9%)
FC weak	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	2 (11.1%)
Sum FC	20 (100.0%)	17 (100.0%)	11 (100.0%)	20 (100.0%)	20 (100.0%)	18 (100.0%)

TABLE I
ABSOLUTE AND RELATIVE FREQUENCIES OF THE CODE CATEGORIES OCCURRING WITH REGARD TO RQ1 IN THE QUALITATIVE DATA OF PHASES 2-4 (MODULE X.1-3).

Module. This upward trend is smaller in Modules 2 and 3, as the significance for the hypothesis could be shown here, but there is weak evidence for hypothesis 1 and no evidence for hypothesis 4. (Module 2: 287642.4; Module 3: 131183716; Module 1: 35.62; Module 4: 0.21; add that and more times more likely, [29]).

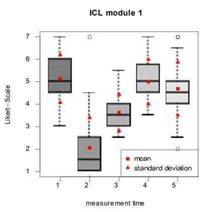
The effect size values (standardized Cohan's d) illustrate this, as between the mean values of time points 1, 2, 3 and 4 at the respective time points 2, 3 and 4 are positive and predominantly moderate to strong, also the increase in the ECL measurement at time 5 (Fig. 9), is also reflected in the effect size values (mean values time 4 and 5 negative, moderate to strong) [31].

The **qualitative data** analysis for RQ2, using a content analysis system, revealed varied student responses across Modules. Figure 10 shows a document comparison chart of the categorized text responses on the worksheets between sub-problems 1-3 (phases 2-4 of the *semantic wave*) in each Module (Modules 1-4) and individual students (numbers 1-39). In Module 2, responses improved from weak in phase 2 to medium in phase 3, matching the *semantic wave* phases, but no further increase was seen in phase 4 with some responses missing.

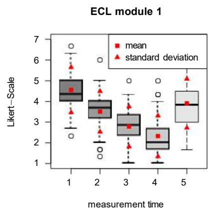
Conversely, Modules 1, 3, and 4 did not align clearly with *semantic wave* phases. Module 1 showed a decline from medium/strong to medium/weak responses. Module 3 had predominantly weak responses, and Module 4, especially in phase 4, had few strong responses.

VI. Discussion

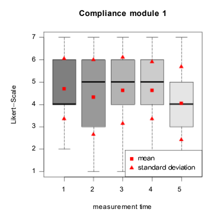
In this study, we presented a model for designing computer science lessons at school promoting *algorithmic thinking*. We integrated and refined it with the concept of the *semantic*



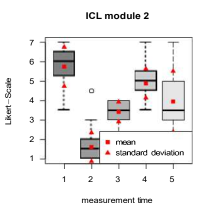
(a) ICL Module 1



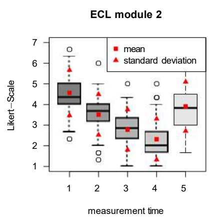
(b) ECL Module 1



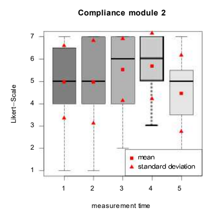
(c) CP Module 1



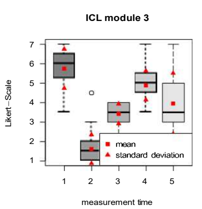
(d) ICL Module 2



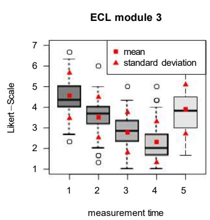
(e) ECL Module 2



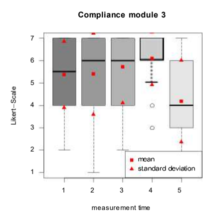
(f) CP Module 2



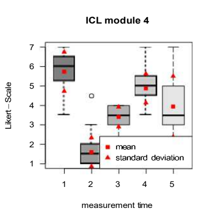
(g) ICL Module 3



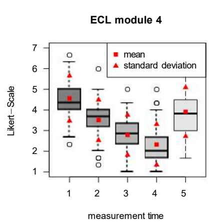
(h) ECL Module 3



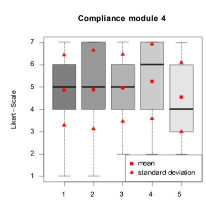
(i) CP Module 3



(j) ICL Module 4



(k) ECL Module 4



(l) CP Module 4

Fig. 9. Intrinsic Cognitive Load (ICL), Extraneous Cognitive Load (ECL) and Compliance (CP) with Mean and Standard Deviation.

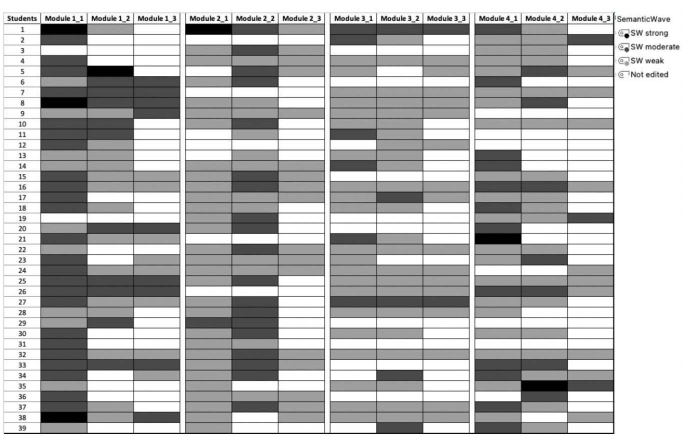


Fig. 10. Occurrence of the *semantic wave* categories in the characteristics weak (color light grey), medium (color dark grey) and strong (color black), phases 2-4 per student.

wave. The context of the study is a *plugged* CS educational environment using a block-based programming language. To the best of our knowledge, there is little research only in unplugged settings in the area of *semantic wave* in the design and reflection of instructions in CS education in the case studies of [4] and [5]. Our proposed model in the area of *algorithmic thinking* is also intended to support pre-service computer science teachers in designing instructional processes that are easy to implement and facilitate rapid planning and reflection for K-12 computer science (CS) education. This study also deals with the refinement of the model through a longer duration (4 Modules 90 min each instead of 1 Module), of the incorporation of *notional machines* and the *computational action* approach to improve the training of future CS teachers and students for the challenges of digital transformation (see Fig. 2).

The analysis of the qualitative data for RQ1 (To what degree do K-12 students' competence in *algorithmic thinking* develop?) shows that the students solved the Modules 1, 2 and 4 in the categories *solve problem*, using *notional machines* predominantly in the characteristics medium and strong compared to the text response tasks (category *understand problem*). In the categories *flow control* and *logic* (programming solutions), the student solutions (Scratch programs) were even predominantly strongly and moderately mapped in all four Modules. Since the difficulty of the Modules in the workshop increases, this suggests that the students' *algorithmic thinking* competence has been promoted, while no significant statistical increase in *algorithmic thinking* is observed for RQ1. These results of the quantitative analysis could be due to a number of reasons, such as the number of participants, as the sample size of N=39 is not too large, and e.g. in [41] an improvement in problem solving was shown with a Scratch intervention with 113 participants and a duration of one month. The duration of the intervention could also be a reason, as in [42] an improvement in problem solving was shown with a Scratch intervention with only 28 participants but a duration of 2 semesters. Of course, both the number of participants and the duration could be the reason for the non-significant statistical results, as in [43] an improvement in problem solving was shown with a Scratch intervention of 139 participants and a duration of 1 semester. Certainly, it also remains to be investigated whether a statistically significant increase in *algorithmic thinking* occurs when the difficulty of the task or the context, such as physical computing [44], is varied. Finally, it should be mentioned that the treatment and control group is not fully randomized. All this should be taken into account in a follow-up study.

In RQ2 (How are the phases of the *semantic wave* perceived by the students?), analysis of the quantitative data from all four workshop Modules revealed a wave-like course of ICL measurements similar to the course of a *semantic wave* and a declining trend in ECL measurements across all Modules. This suggests that the phases of the *semantic wave* of the workshop were appropriately perceived by the students and that the design of the workshop was basically in line with the planning of a *semantic wave*. Since according to [38] the ICL depends on two different factors, the interactivity of the element and the learner's prior knowledge, this should of course be investigated in a subsequent study with different participants or intervention settings, e.g. using a constructionism versus a *semantic wave* approach [45]. Following the work of [46] and [47], who investigated both the relationship between cognitive load and learning programming with block-based programming languages or simulation games, the relationship between cognitive load and a *semantic wave* could also be investigated in more detail. Qualitatively, no *semantic wave* was detectable in the textual responses of the students. The first aim of this study was to explore and test the SWAT model as a whole, so the next interesting step would be to examine the individual phases of the *semantic wave* more in detail. This could be done, for example, using think-aloud approaches or interviews to examine the *semantic wave* flow more closely, as described in [5] for unplugged settings. In this context, the relationship between *notional machines* (flowcharts) and *semantic waves* could be further investigated. The use of flowcharts (*notional machines*) in the problem-solving phase (category *solve problem*) was new in this second study of the SWAT model. The flowcharts almost consistently provided better categorical results in terms of *algorithmic thinking* than the text responses in the *understand problem* step. This could be an indication of a *notional machine* effect mentioned in [23], that the use of a *notional machine* increases *semantic gravity* and decreases *semantic density* (see Figure 1 and section II-D) and thus makes a concept or here a problem more understandable. This should be explored in more detail, perhaps by comparing exclusively textual responses in the problem-solving step with exclusively flowcharts in this step, and provides a great opportunity for further research into the relationship between a *semantic wave* and the use of *notional machines*.

The study workshop is also designed to address *treatment fidelity* categories of [27] (see sec. IV). *Quality* is addressed by an additional item where students rate their personal effort (see Fig. 9c), 9f), 9i) and 9l)). The rating is high in all four workshop Modules, with a decrease towards the end, so the overall *quality* is very well met although the level of difficulty of the workshop increased from Module to Module and also within a Module along the *semantic wave*. This is perhaps an indication that the requirement for *computational action*, which was incorporated into the design of this workshop, could have a positive effect on compliance. However, this would need to be investigated further - for example, through qualitative interviews that examine precisely this question.

The qualitative results for RQ1 showed that in all Modules *algorithmic thinking* competence decreases in phase 4 (see Fig. 7 and 8, Tab.I), while both students' compliance and ICL scores were high (Fig.9). The results of the qualitative analysis in terms of *semantic density* and *semantic gravity* were also mostly weak characterised in this phase (Fig.10). This suggests that *semantic density* may have been higher and *semantic gravity* lower than predicted in phase 4 in all Modules, especially in Module 3, since the analyzed data show weaker results here than in the other Modules.

VII. Conclusions

Overall, it is important to note that the focus of this study is on the experience of using a *semantic wave* in CS education to promote *algorithmic thinking*. The results show, on the one hand, that the K-12 students seem to have experienced the phases of the *semantic wave* accordingly and, on the other hand, that in all Modules the phases 4 in terms of the *semantic wave* have more *semantic density* (technical jargon) and less *semantic gravity* (complexity of the task) than originally intended. In addition, the qualitative data show a promotion of *algorithmic thinking* - both in the use of *notional machine* and in block-based programming. However, the statistical results were limited due to sample size and study design. In conclusion, the strategy of combining *semantic wave* with *algorithmic thinking* steps in a model with the integration of *notional machines* and *computational action*-oriented content is a promising approach that provides a strategy for prospective CS teachers to accurately plan instructional steps and reflect on them after the fact, as is done in the discussion here, but should be pursued further. The present study therefore opens up exciting new directions for research in K-12 CS education, e.g., the connections between *semantic wave* and *notional machine* in more detail or to further enhance the competence of *algorithmic thinking* by integrating *productive failure* approaches into the model.

Future research on the SWAT model should therefore perhaps investigate shortening the *semantic wave* phases by one, e.g. phase 4, to give students more time for problem solving and thus for *algorithmic thinking*. The comparison with the misconceptions studied by [48] could also be helpful in the design of the SWAT model - this needs to be investigated. To increase the competence of *algorithmic thinking*, the *productive failure* approach of [49] could also be a possible extension of the model to further enhance *algorithmic thinking* and thus *digital competences*. Next steps will focus on revising the SWAT model to better promote students' *algorithmic thinking* competences by intensifying the problem-solving process.

References

[1] A. Juskeviciene and V. Dagiene, "Computational thinking relationship with digital competence," *Informatics in Education*, vol. 17, no. 2, pp. 265–284, 2018.

[2] J. M. Wing, "Computational thinking and thinking about computing," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1881, pp. 3717–3725, 2008.

[3] K. Maton, "Making semantic waves: A key to cumulative knowledge-building," *Linguistics and Education*, vol. 24, no. 1, pp. 8–22, 2013.

[4] P. Curzon, J. Waite, K. Maton, and J. Donohue, "Using semantic waves to analyse the effectiveness of unplugged computing activities," *ACM International Conference Proceeding Series*, 2020.

[5] J. Waite, K. Maton, P. Curzon, and L. Tuttiett, "Unplugged computing and semantic waves: Analysing crazy characters," *ACM International Conference Proceeding Series*, no. September, 2019.

[6] F. Ritter and B. Standl, "Promoting Student Competencies in Informatics Education by Combining Semantic Waves and Algorithmic Thinking," *Informatics in Education*, 2022.

[7] J. Sorva, "Notional machines and introductory programming education," *ACM Transactions on Computing Education*, vol. 13, no. 2, 2013.

[8] M. Tissenbaum and A. Ottenbreit-Leftwich, “A vision of K—,” *Communications of the ACM*, vol. 63, no. 5, pp. 42–44, 2020.

[9] B. Standl, B. Standl, J. Rosato, A. Leftwich, and V. Fragapane, “(How) Do They Bring Computer Science to Everyone? Pre-Service Computer Science Teacher Programs in Germany and in the USA,” *Society for Information Technology Teacher Education International Conference*, vol. 2020, no. 1, pp. 631–637, 2020.

[10] S. Papert, *Mindstorms*. Birkhäuser Basel, 1982.

[11] M. Lodi, “Informatical Thinking,” *Olympiads in Informatics*, vol. 14, pp. 113–132, 2020.

[12] B. Standl, “Solving everyday challenges in a computational way of thinking,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10696 LNCS, no. November, pp. 180–191, 2017.

[13] N. O. Ezeamuzie, J. S. Leung, R. C. Garcia, and F. S. Ting, “Discovering computational thinking in everyday problem solving: A multiple case study of route planning,” *Journal of Computer Assisted Learning*, vol. 38, no. 6, pp. 1779–1796, 2022.

[14] B. Standl, “A case study on cooperative problem solving processes in small 9th grade student groups,” in *2016 IEEE Global Engineering Education Conference (EDUCON)*, pp. 961–967, IEEE, 2016.

[15] M. Tissenbaum, J. Sheldon, and H. Abelson, “Viewpoint from computational thinking to computational action,” *Communications of the ACM*, vol. 62, no. 3, pp. 34–36, 2019.

[16] GI, “Dagstuhl-Erklärung: Bildung in der digitalen vernetzten Welt. Eine gemeinsame Erklärung der Teilnehmerinnen und Teilnehmer des Seminarsauf Schloss Dagstuhl –Leibniz-Zentrum für Informatik GmbH,” 2016.

[17] Z. Jina Asvat, “Semantic waves and their affordances for teaching scaffolding to pre-service teachers,” *Reading Writing*, vol. 13, no. 1, pp. 1–10, 2022.

[18] S. Sheng, “Research on College English Classroom Construction Based on Semantic Wave,” *Journal of Contemporary Educational Research*, vol. 7, no. 3, pp. 49–54, 2023.

[19] H. Montiel and M. G. Gomez-Zermeño, “Educational challenges for computational thinking in k–12 education: A systematic literature review of “scratch” as an innovative programming tool,” *Computers*, vol. 10, no. 6, 2021.

[20] L. C. Zhang and J. Nouri, “A systematic review of learning computational thinking through Scratch in K-9,” *Computers and Education*, vol. 141, no. June, p. 103607, 2019.

[21] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, and Y. Kafai, “Scratch: Programming for all,” *Communications of the ACM*, vol. 52, no. 11, pp. 60–67, 2009.

[22] J. Moreno-Leon and G. Robles, “Code to learn with Scratch ?,” *2016 IEEE Global Engineering Education Conference (EDUCON)*, no. April, pp. 150–156, 2016.

[23] S. Fincher, J. Jeuring, C. S. Miller, P. Donaldson, B. Du Boulay, M. Hauswirth, A. Hellas, F. Hermans, C. Lewis, A. Mühling, J. L. Pearce, and A. Petersen, “Notional Machines in Computing Education: The Education of Attention,” *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, pp. 21–50, 2020.

[24] Robert K. Yin, *Case Study Research and Applications Design and Methods Sixth Edition*. 2018.

[25] M. Román-González, “Computational Thinking Test: Design Guidelines and Content Validation,” *Proceedings of the 7th Annual International Conference on Education and New Learning Technologies (EDULEARN 2015)*, no. July 2015, pp. 2436–2444, 2015.

[26] L. M. Sanetti, B. G. Cook, and L. Cook, “Treatment Fidelity: What It Is and Why It Matters,” *Learning Disabilities Research and Practice*, vol. 36, no. 1, 2021.

[27] S. W. Smith, A. P. Daunic, and G. G. Taylor, “Treatment fidelity in applied educational research: Expanding the adoption and application of measures to ensure evidence-based practice,” *Education and Treatment of Children*, vol. 30, no. 4, pp. 121–134, 2007.

[28] H. Hoijtink, J. Mulder, C. van Lissa, and X. Gu, “A Tutorial on Testing Hypotheses Using the Bayes Factor,” *Psychological Methods*, 2019.

[29] E. J. Wagenmakers, R. Wetzels, D. Borsboom, and H. L. van der Maas, “Why Psychologists Must Change the Way They Analyze Their Data: The Case of Psi: Comment on Bem (2011),” *Journal of Personality and Social Psychology*, vol. 100, no. 3, 2011.

[30] M. Bosman, *Robust Bayes factors for Bayesian ANOVA: overcoming adverse effects of non-normality and outliers*. Msc thesis, Utrecht University, 2018.

[31] P. A. Lachenbruch and J. Cohen, “Statistical Power Analysis for the Behavioral Sciences (2nd ed.).,” *Journal of the American Statistical Association*, vol. 84, no. 408, 1989.

[32] A. Vargha and H. D. Delaney, “A critique and improvement of the CL common language effect size statistics of McGraw and Wong,” *Journal of Educational and Behavioral Statistics*, vol. 25, no. 2, 2000.

[33] P. Mayring and T. Fenzl, *Handbuch Methoden der empirischen Sozialforschung*. Springer Fachmedien Wiesbaden, 2019.

[34] J. Moreno-León and G. Robles, “Dr. Scratch: A web tool to automatically evaluate scratch projects,” *ACM International Conference Proceeding Series*, vol. 09-11-Nove, no. October, pp. 132–133, 2015.

[35] F. Ritter, “Category system for all modules.” https://cloudfiles.mpgg.de/index.php/s/oAjnHfG2mkttPS4, 2024. Accessed: March 1, 2024.

[36] J. Sweller, J. J. Van Merrienboer, and F. G. Paas, “Cognitive Architecture and Instructional Design,” *Educational Psychology Review*, vol. 10, no. 3, pp. 251–296, 1998.

[37] J. Sweller and P. Chandler, “Why Some Material is Difficult to Learn,” *Cognition and Instruction*, vol. 12, no. 3, pp. 185–233, 1994.

[38] J. Sweller, J. J. van Merriënboer, and F. Paas, “Cognitive Architecture and Instructional Design: 20 Years Later,” *Educational Psychology Review*, vol. 31, no. 2, pp. 261–292, 2019.

[39] M. Klepsch, F. Schmitz, and T. Seufert, “Development and validation of two instruments measuring intrinsic, extraneous, and germane cognitive load,” *Frontiers in Psychology*, vol. 8, no. NOV, 2017.

[40] A. Skulmowski and G. D. Rey, “Subjective cognitive load surveys lead to divergent results for interactive learning media,” *Human Behavior and Emerging Technologies*, vol. 2, no. 2, pp. 149–157, 2020.

[41] Q. Brown, W. Mongan, D. Kusic, E. Garbarine, E. Fromm, and A. Fontecchio, “Computer Aided Instruction as a Vehicle for Problem Solving: Scratch Programming Environment in the Middle Years Classroom,” *ASEE Annual Conference and Exposition, Conference Proceedings*, 2008.

[42] D. Giordano and F. Maiorana, “Use of cutting edge educational tools for an initial programming course,” in *IEEE Global Engineering Education Conference, EDUCON*, pp. 556–563, 2014.

[43] S.-c. Kong, H. Abelson, and M. Lai, *Computational Thinking Education*. Springer Singapore, 2019.

[44] F. Ritter and B. Standl, “Promoting Computational Thinking in Teacher Education - Combining Semantic Waves and Algorithmic Thinking,” in *Proceedings of the 2022 ACM Conference on International Computing Education Research - Volume 2*, ICER ’22, (New York, NY, USA), pp. 48–49, Association for Computing Machinery, 2022.

[45] F. Ritter, N. Schlomske-Bodenstein, and B. Standl, “Integration of an Informatics Teaching-Learning Laboratory into Pre-Service Informatics Teacher Education,” 2023.

[46] Ü. Çakiroğlu, S. S. Suiçmez, Y. B. Kurtoğlu, A. Sari, S. Yildiz, and M. Öztürk, “Exploring perceived cognitive load in learning programming via scratch,” *Research in Learning Technology*, vol. 26, no. 1063519, pp. 1–19, 2018.

[47] N. Pellas, “Exploring relationships among students’ computational thinking skills, emotions, and cognitive load using simulation games in primary education,” *Journal of Computer Assisted Learning*, vol. n/a, no. n/a.

[48] Y. Qian, S. Hambrusch, A. Yadav, S. Gretter, and Y. Li, “Teachers’ Perceptions of Student Misconceptions in Introductory Programming,” *Journal of Educational Computing Research*, vol. 58, no. 2, pp. 364–397, 2020.

[49] M. Kapur, “Productive failure,” *Cognition and Instruction*, vol. 26, no. 3, pp. 379–424, 2008.