https://doi.org/10.15388/vu.thesis.938
https://orcid.org/0000-0002-7755-0428

VILNIUS UNIVERSITY

Vaida Masiulionytė-Dagienė

Modelling of Computational Thinking Automated Assessment

DOCTORAL DISSERTATION

Technological Sciences,
Informatics Engineering (T 007)

VILNIUS 2026

The dissertation was prepared between 2021 and 2025 at Vilnius University.

Academic Supervisor – Assoc. Prof. Dr. Tatjana Jevsikova (Vilnius University, Technological Sciences, Informatics Engineering – T 007).

This Doctoral Dissertation will be Defended in a Public Meeting of the Dissertation Defence Panel:

Chairman – Prof. Dr. Julius Žilinskas (Vilnius University, Technological Sciences, Informatics Engineering – T 007).

Members:
Prof. Dr. Tomas Blažauskas (Kaunas University of Technology, Technological Sciences, Informatics Engineering – T 007),
Prof. Habil. Dr. Gintautas Dzemyda (Vilnius University, Technological Sciences, Informatics Engineering – T 007),
Dr. Anita Juškevičienė (Vilnius University, Technological Sciences, Informatics Engineering – T 007),
Prof. Dr. Mart Laanpere (Tallinn University, Estonia, Natural Sciences, Informatics – N 009).

The dissertation will be defended at a public meeting of the Dissertation Defence Panel at 12 p.m. on 17th of June 2026 in meeting room 203 of the Vilnius University Institute of Data Science and Digital Technologies.
Address: Akademijos g. 4, LT-08412, Vilnius, Lithuania.
Tel. +370 5 210 9300; e-mail: info@mii.vu.lt

https://doi.org/10.15388/vu.thesis.938
https://orcid.org/0000-0002-7755-0428

VILNIAUS UNIVERSITETAS

Vaida Masiulionytė-Dagienė

Informatinio mąstymo automatizuoto vertinimo modeliavimas

DAKTARO DISERTACIJA

Technologijos mokslai,
Informatikos inžinerija (T 007)

VILNIUS 2026

Disertacija rengta 2021–2025 metais Vilniaus universitete.

Mokslinė vadovė – doc. dr. Tatjana Jevsikova (Vilniaus universitetas, technologijos mokslai, informatikos inžinerija – T 007).

Gynimo taryba:

Pirmininkas – prof. dr. Julius Žilinskas (Vilniaus universitetas, technologijos mokslai, informatikos inžinerija – T 007).

Nariai:
prof. dr. Tomas Blažauskas (Kauno technologijos universitetas, technologijos mokslai, informatikos inžinerija – T 007),
prof. habil. dr. Gintautas Dzemyda (Vilniaus universitetas, technologijos mokslai, informatikos inžinerija – T 007),
dr. Anita Juškevičienė (Vilniaus universitetas, technologijos mokslai, informatikos inžinerija – T 007),
prof. dr. Mart Laanpere (Talino universitetas, Estija, gamtos mokslai, informatika – N 009).

Disertacija ginama viešame Gynimo tarybos posėdyje 2026 m. birželio 17 d. 12 val. Vilniaus universiteto Duomenų mokslo ir skaitmeninių technologijų instituto 203 auditorijoje.
Adresas: Akademijos g. 4, LT-08412, Vilnius, Lietuva.
Tel. +370 5 210 9300; el. paštas info@mii.vu.lt

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my academic supervisor, Assoc. Prof. Dr. Tatjana Jevsikova, for her guidance, support, and valuable discussions throughout the preparation of this dissertation. Her expertise and feedback were important in shaping this work.

I am especially grateful to Prof. Dr. (HP) Valentina Dagienė for inspiring me to undertake doctoral studies and for her continuous support and motivation throughout this journey.

I would also like to thank the Bebras community, particularly Rimantas Žakauskas and Daumilas Ardickas, for providing access to valuable data and for implementing specific logging mechanisms essential for this research.

My sincere thanks go to the reviewers, Prof. Habil. Dr. Gintautas Dzemyda and Dr. Anita Juškevičienė, for their time, careful evaluation, and constructive comments that helped improve this dissertation.

I would also like to thank Danutė Rimeisienė for her support and assistance with administrative processes throughout the doctoral studies.

Finally, I am deeply grateful to my husband, Tautvydas, and our children for their patience, understanding, and unwavering support over these years. Their encouragement made it possible to complete this journey.

ABSTRACT

Computational thinking (CT) is considered an important competence in modern education. However, in practice, it is often assessed using tests and only by checking whether the final answer is correct. In initiatives such as the Bebras Challenge, interactive tasks generate large amounts of behavioural data that can be collected during task solving, but this information is usually not used for assessment. As a result, important aspects of how learners solve tasks remain unexamined.

This dissertation explores how interaction data can be used to assess CT in a process-oriented way. The study is based on real solution logs from Lithuanian Bebras Challenge tasks. Each solution attempt is transformed into a structured set of behavioural features, including total number of actions, time spent solving the task, object-level interactions (e.g., click counts and object click sequences).

Before clustering, raw solution logs are transformed into structured behavioural feature representations. All extracted features, including sequence-derived attributes, are encoded into a unified format and numerical features are normalised to a [0,1] range. Clustering is applied to individual solution attempts for each interactive task using the encoded and normalised feature vectors. The resulting behavioural clusters are subsequently analysed and generalised across tasks sharing the same interactivity group. For neural classification, interaction sequences are additionally separated and aligned to a common length (using padding or truncation) to ensure compatibility with the LSTM architecture.

A pilot study examined whether solution processes could be grouped according to behavioural similarity. In this pilot phase, k-means clustering was used to explore the structure of solution groups and the sensitivity to parameter settings. Based on these findings and the requirement to avoid manually fixing the number of clusters, Affinity Propagation was selected for the main experiments because it automatically determines the number of behavioural classes. In the empirical evaluation, incorrect (FALSE) solutions were grouped into seven behavioural classes and correct (TRUE) solutions into four classes.

For automatic classification of solution types, a dual-branch neural network was used. One branch analyses click sequences with an LSTM layer, the second branch analyses static behavioural features such as solution

duration, the total number of clicks, and the final binary outcome. The two outputs are merged to produce the final prediction. In the unweighted setting, the classifier achieved 85.9% accuracy for incorrect solutions and 93.0% for correct solutions. However, minority classes were more difficult to detect, with recall values of 0.464 and 0.321 for some incorrect solution types. After applying class weighting, recall for these minority classes increased to 0.844 and 0.728, while overall accuracy decreased slightly (to 83.4% and 90.6%, respectively). This shows that recall is particularly important in this assessment context, since rare solution strategies should not be ignored.

Additional experiments with new tasks of the same interaction type confirmed that the classifier can generalise beyond the training tasks. An expert evaluation with educators indicated that the identified behavioural solution groups are understandable and meaningful for assessment and feedback.

ABBREVIATIONS

AA	Academic Analytics
AI	Artificial Intelligence
AIED	Artificial Intelligence in Education
AP	Affinity Propagation
BCTt	Beginners Computational Thinking test
BDE	Big Data in Education
CCTt	Competent Computational Thinking test
CSV	Comma-Separated Values
CT	Computational Thinking
CTt	Computational Thinking test
DDE	Data-Driven Education
DDDM	Data-Driven Decision-Making in Education
EDM	Educational Data Mining
EDS	Educational Data Science
GUID	Globally Unique Identifier
IA	Institutional Analytics
JSON	JavaScript Object Notation
K–12	Primary and secondary education levels (Kindergarten to Grade 12)
LA	Learning Analytics
LSTM	Long Short-Term Memory neural network
ML	Machine Learning
PCNT	Programming-Comprehension and Non-Programming Test (for CT assessment)

ReLU	Rectified Linear Unit (activation function)
ROMT	Reasoning with Open-ended Multiple-choice Test
SAT	Scratch Analysis Tool
SoLAR	Society for Learning Analytics Research
TA	Teaching Analytics

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Computational thinking (CT) has increasingly been recognized as one of the most important competencies of the 21st century. It is considered a fundamental competence not only for computer scientists but for all learners who must navigate a world shaped by digital technologies. The concept was first introduced by Seymour Papert in the 1980s as a way of enhancing learning and thinking through computers. Later, Jeannette Wing (2006) reformulated CT as a universal problem-solving competence, stating that it "involves solving problems, designing systems, and understanding human behaviour, by drawing on the concepts fundamental to computer science." Since then, the concept has been widely discussed, refined, and adapted across various fields of education and research (Román-González et al., 2017; Tang et al., 2020; Tikva & Tambouris, 2021).

The importance of CT is widely acknowledged in curricula worldwide (Bocconi et al., 2022; Kampylis et al., 2023). However, one of the persistent challenges lies in how to assess CT effectively.

Research Problem

Although CT is often described through skills such as decomposition, pattern recognition, abstraction, and algorithmic thinking (Grover & Pea, 2013), in actual problem-solving these skills often overlap and are difficult to separate. Despite this complexity, CT is still commonly assessed based only on the outcome of a task, typically as correct or incorrect. In many existing approaches, learners are evaluated only on whether their answer is correct, while the way the solution was reached is not considered.

This becomes very visible in interactive CT tasks, like those used in the Bebras Challenge (Bebras, n.d.). Learners actively interact with task elements while solving the task. However, their performance is not evaluated. In multi-step tasks, a single mistake made at a later stage may lead to a completely incorrect result, even if earlier reasoning was appropriate. In contrast to mathematics, where some points are given for correct intermediate reasoning, this information is typically lost in CT assessment. Recent studies increasingly suggest that analysing solution processes can provide a more detailed view of learners' CT (Corrales-Álvarez et al., 2024; Luo & Zhang, 2024). Behavioural traces such as click sequences, time spent on different steps, or action patterns

have been shown to be informative in digital learning environments and educational games (Rowe et al., 2021; Liu, 2024).

At the same time, interactive Bebras tasks already generate large amounts of such behavioural data. Despite this, the data is rarely used for assessment purposes. As a result, much of the information about how learners solve tasks remains unused, and assessment provides only a limited picture of their CT competence. This highlights the need for assessment approaches that can make systematic use of solution process data in an automated way. In this study, computational thinking is therefore approached as a whole, focusing on the analysis of the problem-solving process rather than on the assessment of individual skills. Such an approach could transform assessment from a binary "correct/incorrect" into a process-aware evaluation of CT competence.

Research Focus

This research focuses on modelling an automated, process-oriented approach to the assessment of CT. The proposed system is not a fully implemented tool, but a conceptual model tested to demonstrate its feasibility. Instead of evaluating only the correctness of final answers, the model emphasizes the analysis of solution processes in interactive CT tasks.

The dataset used in this study was prepared from real Bebras Challenge logs. Interactive tasks were specifically designed to record behavioural traces of the solution process, which required preprocessing and unification to make heterogeneous data comparable across tasks.

By transforming behavioural traces such as click sequences, solution duration, number of actions, and final states into structured datasets, and applying machine learning techniques for clustering and neural network models for classification, the study aims to reveal meaningful patterns of problem-solving strategies. In this study, clustering is performed on individual solution attempts, each represented as a behavioural feature vector composed of task-level and object-level interaction attributes extracted from log data. The research highlights that CT can be assessed more comprehensively when attention is shifted from outcomes to processes and proposes a model to support scalable and automated CT assessment.

Research Aim and Objectives

The thesis aims to create a model of automated computational thinking assessment based on process data from solving interactive tasks.

Objectives:

1. To analyse and systematize existing methodologies in computational thinking assessment and identify their limitations.
2. To analyse, systematize, and empirically investigate learning analytics and machine learning methods for computational thinking assessment, using a pilot study based on behavioural process data from interactive tasks' solutions.
3. To develop an automated computational thinking assessment model based on the analysis of behavioural process data, applying clustering and classification methods.
4. To empirically evaluate the proposed computational thinking assessment model using interactive tasks' solutions data and assess its performance and applicability.

Research Methods

This research is based on the analysis of behavioural data generated while solving interactive CT tasks. The methodological approach combines literature analysis, learning analytics, and machine learning techniques.

A review of existing CT assessment methods was carried out to identify current limitations and to justify the need for a process-oriented assessment approach. Behavioural data from interactive Bebras tasks were collected and transformed into structured datasets suitable for further analysis.

To explore different solution strategies, a pilot study was conducted using clustering methods applied to behavioural task data. This analysis made it possible to identify recurring patterns in learners' problem-solving behaviour. The insights gained from the pilot study were then used to define the structure of a general assessment model, which combines clustering-based solution groups with neural network classification to support automated evaluation of solution processes.

The proposed model was evaluated using real task solution data. In addition, an expert-based survey was conducted to validate whether the

clustered solution groups and the resulting assessment outcomes are meaningful and understandable from an educational perspective.

Scientific Novelty

The novelty of this research lies in the development of a process-oriented model for assessing CT through interactive CT tasks. For the first time, heterogeneous logs from interactive Bebras tasks were transformed into a unified labelling and feature representation, enabling systematic comparison and analysis of solution strategies across tasks. The key result of the study is that clustering methods (k-means in the pilot phase and Affinity Propagation in the main experiments) successfully distinguished logical groups of solution processes, demonstrating that students' problem-solving approaches could be meaningfully separated without prior labelling.

The model further introduces a dual-branch neural network architecture for classification, combining sequential features (click sequences) with static features (solution duration, number of clicks, and the final binary answer). This integration demonstrates that behavioural data from non-programming Bebras tasks can be both clustered and classified automatically. In this way, the research expands the scope of CT assessment, showing that process-oriented data and algorithmic modelling can be applied effectively to educational tasks that have traditionally been evaluated only on final correctness.

Practical Value of the Research

The practical value of this study lies in demonstrating that automated, process-based assessment of CT can provide meaningful insights using behavioural data from interactive Bebras tasks. By analysing behavioural traces such as click sequences, solution durations, and final states, the model makes it possible to classify solution strategies and highlight different approaches students take when solving tasks. This allows educators to go beyond correct/incorrect outcomes and gain a deeper understanding of learners' reasoning, enabling more targeted feedback and support.

Beyond the educational context, the same methodology could also be applied in training environments or human resources, where evaluating how individuals approach and solve complex tasks is just as valuable as assessing whether their outcome is correct. In this way, the research opens opportunities

for practical applications that combine behavioural data analysis with automated assessment techniques.

Statements to be Defended

1. Features of solution processes in interactive computational thinking tasks enable the automated clustering of behavioural solution data into interpretable solution groups that reflect different problem-solving strategies.
2. Solution-process data clustering reveals the same behavioural solution patterns across different interactive tasks belonging to the same interactivity group. These patterns are suitable for unification into a common labelling scheme applicable across all tasks of the same interactivity group.
3. The proposed automated assessment model, based on the unified representation of solution processes, enables the classification of solutions with sufficient accuracy.

Publications and Conferences

Articles in international research journals with a citation index in the Clarivate Analytics Web of Science (CA WoS) database:

1. Masiulionytė-Dagienė, V., & Jevsikova, T. (2025). Towards a process-oriented assessment of computational thinking: behavioural data and machine learning approach. *Interactive Learning Environments*, 1–15. https://doi.org/10.1080/10494820.2025.2567601
2. Bilbao, J., Bravo, E., Garcia, O., Rebollar, C., Laakso, M.-J., Kaarto, H., Lehtonen, D., Parviainen, M., Jankauskienė, A., Pears, A., Guven, I., Gulbahar, Ozturk, T., Yenigun N.T., Pluhar, Z., Sarmasagi, P., Rumbus, A., Dagienė, V., & Masiulionytė-Dagienė, V. (2025). Analytical Methods and Tools for Evaluating the Development of Computational Thinking Abilities. *International Journal of Education and Information Technologies*, 19, 53-61. https://doi.org/10.46300/9109.2025.19.6

3. Bilbao, J., Bravo, E., Garcia, O., Rebollar, C., Laakso, M.-J., Kaarto, H., Lehtonen, D., Parviainen, M., Jankauskienė, A., Pears, A., Guven, I., Gulbahar, Y., Oncul, F. O., Yenigun, N. T., Pluhar, Z., Sarmasagi, P., Dagienė, V., & Masiulionytė-Dagienė, V. (2024). Introducing computational thinking and algebraic thinking in the European educational systems. *International journal of education and information technologies*. https://doi.org/10.46300/9109.2024.18.2

4. Kampylis, P., Dagienė, V., Bocconi, S., Chioccariello, A., Engelhardt, K., Stupurienė, G., Masiulionytė-Dagienė, V., Jasutė, E., Malagoli, C., Horvath, M., & Earp, J. (2023). Integrating Computational Thinking into Primary and Lower Secondary Education: A Systematic Review. *Educational Technology & Society*, 26(2), 99-117. https://doi.org/10.30191/ETS.202304_26(2).0008

International conference presentations:

1. Masiulionytė-Dagienė, V. (2021). "Gamification for developing computational thinking in blended-learning environment: students' motivation and assessment problems". *ISSEP 2021*, Netherlands.

2. Masiulionytė-Dagienė, V., and Jevsikova, T. (2022). „Assessing Computational Thinking: The Relation of Different Assessment Instruments and Learning Tools". *15th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP 2022*, Vienna, Austria.

3. Masiulionytė-Dagienė, V. (2023). "Modeling of the System for Computational Thinking Automatic Assessment". *28th annual ACM conference on Innovation and Technology in Computer Science Education (ITiCSE), ITiCSE 2023*, Turku, Finland.

4. Masiulionytė-Dagienė, V., & Jevsikova, T. (2025, June). Behavioural Data-Driven Approach for Computational Thinking Automatic Assessment Using Interactive Bebras Challenge Tasks. In *Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education* V. 2.

National conference presentations:

Masiulionytė-Dagienė V. (2025). „Interaktyvių Bebro užduočių sprendimų duomenų analizė ir vertinimo skalės modelis" *Kompiuterininkų dienos 2025*, Šiauliai, Lietuva.

Other publications:

1. Bilbao, J., Bravo, E., Garcia, O., Rebollar, C., Dagienė, V., Masiulionytė-Dagienė, V., Jankauskienė, A., Laakso, M-J., Kaarto, H., Lehtonen, D., Güven, I., Gulbahar, Y., Özdemir Öncül, F., Pluhár, Z., Sarmasági, P., Pears, A. (2023) Working the basis of computational thinking: definition and skills // *ICERI2023 Proceedings: 16th annual international conference of education, research and innovation*, Seville, Spain. 13-15 November 2023. València: IATED Academy, 2023. ISBN 9788409559428. p. 8410-8416. (ICERI Proceedings, ISSN 2340-1095). https://doi.org/10.21125/iceri.2023.2151.

2. Bilbao, J.; Bravo, E.; Garcia, O.; Rebollar, C.; Dagienė, Valentina; Masiulionytė-Dagienė, Vaida; Jankauskienė, A.; Laakso, M.J.; Kaarto, H.; Lehtonen, D.; Parviainen, M.; Güven, I.; Gulbahar, Y.; Öztürk, T.; Özdemir Öncül, F.; Tan Yenigün, N.; Pluhár, Z.; Sarmasági, P.; Pears, A. Computational thinking and problem solving in PISA era // *INTED2024: 18th international technology, education and development conference*, 4-6 March, 2024, Valencia, Spain: conference proceedings / edited by L. Gómez Chova, C. González Martínez, J. Lees. València: IATED Academy, 2024. ISBN 9788409592159. p. 7335-7342. (INTED Proceedings, ISSN 2340-1079). https://doi.org/10.21125/inted.2024.1922.

3. Bocconi, S., Chioccariello, A., Kampylis, P., Dagienė, V., Wastiau, P., Engelhardt, K., Earp, Jeffrey; Horvath, Milena; Jasutė, Eglė; Malagoli, Chiara; Masiulionytė-Dagienė, V., Stupurienė, G. (2022) Reviewing computational thinking in compulsory education: state of play and practices from computing education. *Luxembourg: Publications Office of the European Union*, 2022. 138 p. ISBN 9789276472087. https://doi.org/10.2760/126955.

Outline of the Thesis

The dissertation is divided into five chapters reflecting the development of conceptual foundations towards the proposed behaviour-based assessment model for interactive CT tasks and its empirical validation. The introductory chapter presents the context of the study, formulates the objectives and tasks, and sets out the scientific innovations and claims to be defended. Chapter 1

provides a critical review of existing methods for assessing and analysing CT, pointing out the methodological limitations of assessment and emphasizing the need for a common behavioural analysis in various interactive tasks.

Chapter 2 presents a pilot study demonstrating that different task solution variants can be distinguished using task solution behaviour data and clustering methods. Based on this knowledge, Chapter 3 formalizes a general assessment model, defining interactivity groups of tasks, unified feature and label spaces, and establishing a workflow for clustering historical solutions and classifying new solutions. Chapter 4 presents empirical studies with other Bebras tasks, evaluating the quality of clustering, the performance and generalizability of the classifier, and additionally presents an expert survey to assess comprehensibility. The conclusion of the work presents a synthesis of conclusions, contributions, limitations, and directions for future research.

1. LITERATURE REVIEW

1.1 Overview of CT Assessment Research Area

Research on the assessment of computational thinking has expanded considerably over the last decade, driven by the recognition of CT as a fundamental 21st-century competency and its integration into school and higher-education curricula. Across the reviewed literature, several dominant trajectories can be identified:

1. The use of tests and psychometric instruments to measure CT-related cognitive abilities.
2. The development of automated analysis tools, particularly those applied to students' programming artefacts.
3. The emergence of game-based and interactive environments for process-oriented assessment.
4. Broader attempts to conceptualise CT assessment frameworks and methodological approaches.
5. The increasing use of learning analytics, machine learning, and AI-supported systems for personalised assessment and feedback.

A substantial body of work relies on traditional CT tests, typically composed of multiple-choice or constructed-response items. One of the most extensively validated instruments is the Computational Thinking test (CTt) proposed by Román-González et al. (2017), who demonstrated that CTt reliably measures key CT constructs: abstraction, decomposition, algorithmic reasoning, and debugging, without relying on students' programming skills. CTt has subsequently been used in studies on CT development, predictive factors of CT growth, and instructional interventions (Guggemos, 2021). Adaptations of CTt for younger learners, such as the Beginners Computational Thinking test (BCTt) (Zapata-Cáceres et al., 2021), extend this tradition of psychometric assessment into primary education.

Other researchers developed instruments focusing on programming-related CT competencies. For example, the Test for Measuring Basic Programming Abilities designed by Mühling et al. (2015) evaluates students' understanding of control structures, while the Commutative Assessment proposed by Weintrop & Wilensky (2015, 2019) compares students'

conceptual understanding across block-based and text-based programming environments. Additional domain-specific instruments include CT scales based on CT frameworks (e.g., the Computational Thinking Scales (CTS) (Korkmaz et al., 2017; Tsai et al., 2020)), Likert-type measures of CT dispositions (Yağcı, 2019), or bespoke assessments used in intervention studies (e.g., gender differences in CT performance in Wu & Su, 2021).

More recent empirical work also emphasises the need for CT assessments that capture not only correctness but also strategic reasoning and metacognitive processes (Kovanović et al., 2024; Shute & Rahimi, 2023), further highlighting the limitations of purely test-based approaches.

Overall, these studies reveal strong interest in objective, scalable, test-based assessment methods, but also highlight recurring challenges: limitations in capturing CT as a process rather than an outcome, over-reliance on discrete question formats, and insufficient sensitivity to real-time problem-solving strategies.

Another prolific research line evaluates CT by examining digital artefacts, typically Scratch or similar block-based programming projects. Automated tools such as Dr. Scratch (Moreno-León & Robles, 2015; Moreno-León et al., 2015) analyse project code to infer CT-related competencies, including abstraction, logical reasoning, synchronization, parallelism, and data representation. Numerous studies have used Dr. Scratch to assess students' progress or to evaluate instructional interventions (Troiano et al., 2019; Fagerlund et al., 2020; Wei et al., 2021; Labusch & Eickelmann, 2020).

In response to methodological limitations of Dr. Scratch, such as inaccurate block recognition and inefficiencies, researchers developed alternative tools. The Scratch Analysis Tool (SAT) proposed by Chang et al. (2018) uses ANTLR for structured code parsing and aims to provide more reliable CT analytics.

Artefact-based approaches also extend beyond Scratch. For example, Metcalf et al. (2023) analysed snapshots of students' computational models to assess developing computational fluency, while Grgurina et al. (2018) proposed assessment tools for modelling and simulation competencies in secondary education. These studies highlight the value of code-based artefacts for assessing CT, particularly when coupled with automated analytics, although they remain limited to programming-centric contexts. Recent studies additionally explore how learning analytics and machine learning techniques can support more fine-grained CT assessment based on students' interaction

data with programming environments (Kurniawan et al., 2024; Li et al., 2023). These approaches demonstrate increasing interest in automated, data-driven evaluation of learners' computational reasoning processes.

Several studies explore non-programming-centred contexts where CT skills emerge through robotics, unplugged activities, and diverse learning environments. Grover (2011) and related work examine CT development through robotics tasks; Tonbuloğlu & Tonbuloğlu (2019) investigate CT through unplugged coding activities, supported by structured observation protocols; Statter & Armoni (2020) assess abstraction skills using a combination of tests, observations and qualitative field notes. Waite et al. (2020) introduce a new method that analyses students' drawings to evaluate early-stage design reasoning in programming education.

This group of studies underscores how CT competencies manifest beyond digital artefacts and evaluations, and how alternative modalities like robotics, physical artefacts, observational data can reveal aspects of CT not captured by conventional testing.

A rapidly growing strand of research investigates game-based learning environments as both CT learning and assessment tools. The Zoombinis game (Asbell-Clarke et al., 2021; Rowe et al., 2021) has been widely used to assess implicit CT development through logged behavioural data such as decision sequences, error patterns, difficulty progressions, and strategic reasoning. Other game-based CT assessment environments include student-designed games evaluated by Dr. Scratch (Troiano et al., 2019), serious games for STEM learning (Leonard et al., 2021), and simulation-based environments for modelling (Grgurina et al., 2018). Later research extends this trend through AI-augmented environments, for example, adaptive game-based platforms that use machine learning to classify solution strategies or to provide adaptive CT feedback (Chaim et al., 2024; Beltrán-Martín et al., 2023). Recent studies further extend this direction by integrating machine-learning-based models within gamified intelligent tutoring systems to analyse learners' behavioural traces and provide targeted feedback; however, these approaches typically rely on simplified representations of behavioural data and are applied in relatively constrained task environments (Cen et al., 2025).

Games are particularly valuable for CT assessment because they enable process-based evaluation, capturing sequences of actions rather than only outcomes. Although promising, many game-based assessments remain exploratory and are often tailored to specific tasks or age groups, which may

limit their broader applicability. This indicates that, despite extensive research activity, there is still a lack of consistent approaches to interpreting and using process-based data in CT assessment.

The Bebras Challenge, widely examined by Izu et al. (2017) and used in various CT research contexts (Djambong et al., 2018; Labusch & Eickelmann, 2020), is one of the most influential non-programming CT assessment resources. Bebras tasks target CT concepts through short problem-solving activities suitable for both national and international studies. Research consistently notes that Bebras tasks assess high-level analytical thinking and provide stable benchmarking across populations. However, all studies emphasise a major limitation: Bebras assessments capture only final answers, disregarding the solution process and intermediate decision-making steps.

This limitation is central to emerging critiques arguing that CT, as a problem-solving process, cannot be adequately measured through binary correctness alone (Lockwood & Mooney, 2017; Zhang et al., 2020). Recent studies (Corrales-Álvarez et al., 2024; Belletini et al., 2023) further highlight the untapped potential of analysing detailed behavioural traces within interactive tasks to capture solution strategies rather than just outcomes.

Several conceptual, theoretical, and systematic reviews have sought to consolidate the rapidly expanding field of CT assessment by offering overarching analytical frameworks. Curzon et al. (2019) provide one of the foundational distinctions in the literature, separating CT assessment approaches into those conducted through programming and those examining CT through broader problem-solving tasks. Building on this, Tang et al. (2020) synthesise empirical studies and categorise existing assessment methods into traditional tests, portfolio-based evaluations, interviews, and surveys, emphasising persistent methodological fragmentation and noting a particular lack of robust tools for assessing CT in higher-education contexts. A complementary perspective is offered by Tikva and Tambouris (2021), who map CT assessment within K-12 programming education and outline methodological clusters that have emerged across studies. Earlier reviews by Politis and Paulakis (2018) and Lockwood and Mooney (2017) similarly conclude that CT assessment remains underdeveloped, with many tools at early stages of maturity and substantial conceptual inconsistency across research traditions. Extending beyond method classification, Piatti et al. (2022) introduce the CT-Cube, a comprehensive framework for designing and assessing CT activities across the lifespan, highlighting the need for assessment systems that account for the situated, developmental, and

multifaceted nature of CT. More recent frameworks emphasise the integration of AI-enabled assessment (e.g., ChatGPT-supported scaffolding in CT tasks, Liao et al., 2024) and adaptive feedback mechanisms for CT competence development (Zhang et al., 2023; Rahimi et al., 2024). Together, these reviews indicate that despite clear progress, CT assessment research continues to face conceptual and methodological challenges, underscoring the need for more coherent models and evidence-based instruments.

Overall, the reviewed studies highlight several recurring patterns and limitations in current CT assessment research. General observations emerging from the literature:

1. CT assessment has historically been dominated by tests and programming artefact evaluation, while more holistic and process-oriented approaches, although increasingly explored in recent research, remain relatively underdeveloped.

2. Automated tools like Dr. Scratch, SAT, and log-based analytics are increasingly used but still capture only partial CT dimensions.

3. Game-based assessments offer rich behavioural data, but their generalisability across tasks, age groups, and learning contexts is not yet consistently demonstrated, as many existing studies focus on specific tasks, limited participant groups, and controlled experimental settings.

4. Bebras tasks are widely used yet methodologically limited due to binary scoring.

5. There is a field-wide call for CT assessments capturing solution processes, not just outcomes, and applicable beyond programming contexts.

6. Despite a broad research landscape, practically deployable, domain-independent, automated CT assessment tools remain relatively scarce.

7. Recent AI-enhanced and machine-learning-based systems show promise but are still in early exploratory stages.

This synthesis provides the conceptual foundation for the next section, which examines CT assessment methods and tools in greater depth.

1.2 Review of CT Assessment Methods and Tools

Two types of research studies concerning the assessment of CT are identified. The ones which focus on assessment methods and tools itself and another group of studies where assessment is used as a tool for conducting research in other areas.

As Curzon et al. (2019) mentions, "CT assessment is a critical research area. There can be two approaches of assessment: assessing CT through programming, assessing CT through problem-solving. The assessment must determine the progression, and the curriculum has to be age appropriate". However, many existing approaches still primarily evaluate the correctness of final answers, rather than the intermediate steps and strategies that characterise CT as a process (Grover & Pea, 2013; Corrales-Álvarez et al., 2024).

There are various methods used to assess CT competence; scientists look for the most effective ones and try new methods.

In literature review Tang et al. (2020) arrange the CT assessment methods into four groups:

1) The traditional test is composed of selected or constructed response questions;
2) Portfolio assessment;
3) Interviews;
4) Surveys.

Tikva, C., & Tambouris, E. (2021) suggest a slightly different arrangement of CT assessment methods:

1) Self-report methods;
2) Tests;
3) Artifact analysis;
4) Observations;
5) Frameworks.

The three groups of CT assessment methods were identified in this study and mentioned in several articles. These are:

- Tests (traditional tests which focus on CT concept assessment);
- Project/portfolio assessment (e.g., program code analysis;

problem-solving analysis; students' project assessment, etc.);

- Observations (e.g., the teacher's observation of how students perform while working on an assignment).

These methods are used separately; in some cases, for CT assessment, two or three CT assessment methods are used simultaneously. Table 1.1 provides examples of CT assessment methods used in the analysed literature.

Table 1.1: Examples of CT Assessment Methods

CT assessment methods	**Examples of methods in articles**
Tests	Djambong et al. (2018): 23 paper and pencil tasks-based assessment test (14 Bebras tasks and nine tasks designed by the research team). Leonard et al. (2021): pre-and post-CT test with 14 questions based on the CT concepts of sequences, loops, and variables. Román-González et al. (2017): have developed the Computational Thinking Test (CTt) based on the practical guide to validating CS knowledge assessments with application to middle school; the CTt can be administered collectively and online, via non-mobile or mobile e-devices. Weintrop & Wilensky (2019; also 2015): Commutative Assessment test designed for high-school students, grades 9-12; aimed at measuring students' understanding of different computational concepts, depending on whether they occur through scripts written in block-based or textual programming languages.

	Wu & Su (2021): design an assessment to evaluate the performance based on a set of multiple-choice questions to understand gender differences in the program learning environment and CT ability. El-Hamamsy et al. (2025): Competent Computational Thinking test (CCTt), a refined CT test for primary grades 3-6, with extensive validation datasets.
Project/portfolio assessment	Djambong et al. (2018): Using technology-rich learning environment Vizwik (developed by one of the authors, designed to teach coding of K-12), additionally, Scratch, and LEGO robotics tools for creating projects. Metcalf et al. (2023): Provide a methodology to assess students' CT through examining their programmed computational models, i.e., analyse program snapshots to assess developing fluency in CT concepts. Fagerlund et al. (2020): We adopted a comparatively inclusive view of what students can learn in CT through Scratch and, by revising a profound assessment framework, focused uniquely on individually instantiated coding patterns and their underlying code constructs, that is, relatively fine-grained evidence. Wei et al. (2021): The students' CT competence was assessed through Dr. Scratch, an open-source CT assessment tool that automatically analyses the

	programming projects submitted by students. Karakaş & Yöndem (2020): Machine learning based adaptive feedback system that analyses students' code solutions and provides targeted feedback grounded in CT concepts.
Observations	Statter & Armoni (2020): The field notes taken during the class observations were mostly used as a complementary qualitative tool for the interviews. Tonbuloğlu & Tonbuloğlu (2019): The observation form was prepared by the researcher to enable the teacher to monitor the classroom during the lesson, observe student behaviour within the process, and reflect on opinions.
Mixed methods	Wei et al. (2021): Mixed methods approach, including surveys, instruments, and interviews, was used to examine the effectiveness of partial pair programming on elementary school students' CT competence. Statter & Armoni (2020): Different methods were used for assessing: tests, projects, and observation. Ministry of Science and Education Croatia (2018): Methods and techniques for evaluating what has been learned in Informatics are possibly: oral tests; written tests and/or computer tests; e-portfolio – individual papers are evaluated according to the given learning outcomes and student progress during the school year; student projects –

	student participation, activity levels, communication and collaborations, project documentation and the final results of the project and their presentation; use of online checks that are part of internal or hybrid evaluation.

Many of these test-based and portfolio-based approaches, particularly non-programming CT quizzes (e.g., CTt), still operate with binary or coarse-grained scoring (correct/incorrect, or single global scores), which limits the extent to which problem-solving processes and intermediate strategies can be evaluated (Román-González et al., 2017; Sung, 2022). While programming tasks more naturally capture the solution process, programming is only one facet of CT (Cansu, 2019), leaving important parts of CT practice such as non-code problem-solving underassessed.

Besides these methods, computer games are used to teach and to assess students' CT competence. “Everyday game activities were logged and analysed” (Asbell-Clarke et al., 2021). There are more interesting methods mentioned for assessing, for example, analysis of students’ drawings (Waite et al., 2020). Even the dance activities are included in teaching and assessing CT competence: “While we specifically called upon dance to motivate and engage students, it is clear that these embodied experiences allowed them to engage in a variety of computational and choreographic practices*”* (Leonard et al., 2021). Recent research also explores extended-reality and collaborative environments, such as ARQuest and other XR-based tasks (Gardeli & Vosinakis, 2019; Hwang et al., 2023), as well as multi-agent or robotic platforms (Liu et al., 2017), positioning CT assessment within immersive or team-based problem-solving scenarios.

Even with many methods used for CT competence assessment, it is noted that “still some CT skills remained a challenge to be included in the assessment” (Zhang et al., 2020) or that CT assessment remains a thorny, unresolved issue (Román-González et al., 2019). According to Lockwood & Mooney (2017, p. 15):

> … overall, work in testing for computational thinking is in its infancy. Most of the examples are in the early stages of development. Tools do exist, such as Dr. Scratch and the tools developed by the

Scalable Design Group, but there is a need for more research into this area.

"Another problem regarding the assessment of CT is the progression of each CT practice" (Zhang et al., 2020). These critiques are echoed in later studies that highlight the gap between outcome-based scoring and the need to capture students' strategies, reasoning patterns, and decision-making processes over time (Luo & Zhang, 2024; Corrales-Álvarez et al., 2024).

Specific assessment tools are required for the use of any CT assessment method. The CT assessment tools that were identified during the analysis are described further.

Román-González, Moreno-León & Robles (2017) have classified the assessment tools into groups, and Djambong et al. (2018), analysing these groups, provided examples to several of them:

1) CT diagnostic tools;

2) CT summative tools (aptitudes-based Computational Thinking Test (Román-González et al., 2017), the Test For Measuring Basic Programming Abilities (Mühling et al., 2015), or the Commutative Assessment Test (Weintrop & Wilensky, 2015);

3) CT formative-iterative tools (which aim to provide students with immediate automatic feedback, helping them to improve their abilities. This category includes Dr. Scratch (Moreno-León & Robles, 2015) and the Computational thinking pattern graph (Koh et al.,2010);

4) CT data mining tools;

5) CT skill transfer tools (in this category, Djambong et al. (2018) include tasks of the International Bebras competitions, as well as the Computational Thinking Pattern Quizz (Izu et al., 2017; Basawapatna et al., 2011);

6) CT perceptions-attitudes scales (including Computational Thinking Scales (CTS) developed by Korkmaz, Çakir, and Özden (2017));

7) CT vocabulary assessment (tools which could help measure students' verbal skills when doing coding tasks, Grover (2011)).

Román-González et al. (2019) mention that tools coming from each group must be used for different assessment purposes, and they must be combined to obtain the most objective results. For example, Roman-Gonzalez et al. (2019) grouped CT assessment tools according to Bloom's taxonomy: CTt (Computational Thinking test) is used to evaluate understanding and memorizing; Bebras tasks are used to assess analysis and application skills; for creativity and self-evaluation, Dr. Scratch must be used.

An assessment tool, called CTt (Computational Thinking test), was developed by Román-González et al. (2017). This was used, validated, and mentioned in a few papers (Guggemos, 2021; Román-González et al., 2017). Now CTt consists of 28 questions, divided into seven groups (Román-González et al., 2017):

1. Basic directions and sequences (4 items);
2. For loop – repeat times (4 items);
3. While loop – repeat until (4 items);
4. If – simple conditional (4 items);
5. If/else – complex conditional (4 items);
6. While conditional (4 items);
7. Simple functions (4 items).

The test focuses on middle school children (mainly for 12 – 14 years old, but it can be used from 5th to 10th grade). Computational concepts used in the test are aligned with the CSTA Computer Science Standards for the seventh – eighth grades. Guggemos et al. (2023) mention the main CT concepts that CTt covers: abstraction, decomposition, algorithms, and debugging. The CTt has some strengths: it can be collectively administered in pure pre-test conditions, so it can be used in massive screenings and early detection of students with high abilities (or special needs) for programming tasks; and it can be utilized for collecting quantitative data in pre-post evaluations of the efficacy of curricula aimed at fostering CT (Margulieux, 2017). The CCTt (El-Hamamsy et al., 2025) further extends this line of development by refining item design and validation procedures for younger learners, providing richer diagnostic information on primary students' CT competence.

CTt was adapted by María Zapata Cáceres, Estefanía Martín-Barroso (Universidad Rey Juan Carlos, Spain), and Marcos Román-González (Universidad Nacional de Educación a Distancia, Spain) for primary school children (age 5-12), and was called Beginners Computational Thinking test

(BCTt). BCTt consists of 25 questions, which address these CT concepts: sequences, loops, and conditionals.

Besides CTt Román-González et al. (2017) in their work mention two other skills tests, one measuring basic programming abilities and the other measuring CT concepts:

a. Test for Measuring Basic Programming Abilities (Mühling et al., 2015): It is designed for Bavarian students from seventh to tenth grades. This test is aimed at measuring the students' ability to execute a given program based on the so-called 'flow control structures', which are considered at the core of the CT for this age group.

b. Commutative Assessment (Weintrop & Wilensky, 2015): It is designed for high-school students, from ninth to twelfth grade. This test is aimed at measuring students' understanding of different computational concepts.

Bebras tasks were used as another assessment tool for CT competence (Román-González et al., 2017; Labusch & Eickelmann, 2020; Djambong et al., 2018). "The Bebras challenge refers to the analytic and applied levels of the taxonomy – i.e., general analytical thinking" (Labusch & Eickelmann, 2020). More recently, interactive Bebras tasks have been discussed as a promising context for process-oriented CT assessment (Belletini et al., 2023; Zapata-Cáceres, 2024), since they naturally generate behavioural traces (click sequences, time-on-task, intermediate states). However, in current practice, these tasks are almost exclusively scored in a binary manner, based only on the outcome, which leads to the loss of rich information about students' problem-solving strategies (Luo & Zhang, 2024).

As a tool for assessing CT competence, Dr. Scratch or Scratch (Labusch & Eickelmann, 2020; Grover et al., 2019; Wei et al., 2021; Fagerlund et al., 2020) was used. Dr. Scratch automatically analyses Scratch programming projects and can also be used to develop CT (Wei et al., 2021; Troiano et al., 2019). Dr. Scratch analyses code based on these CT concepts (Moreno-León et. al, 2015):

- Abstraction and problem decomposition;
- Logical thinking;
- Synchronization;
- Parallelism;
- Algorithmic notions of flow control;
- User interactivity;

- Data representation.

Zoombinis (Rowe et al., 2015) is an award-winning learning game that was designed in the 1990s and re-released for current platforms. It is not only used for learning CT but also in recent years for CT assessment. In Zoombinis, all the players' actions are logged and then analysed for the purpose of learning or assessment. CT concepts that are assessed in Zoombinis (Rowe et al., 2021) are problem decomposition, pattern recognition, abstraction, and algorithm design. Subsequent work has shown that process data from Zoombinis (e.g., duration, accuracy, sequence of actions, puzzle difficulty) can be analysed using machine-learning techniques to distinguish students' CT stages and strategy profiles (Liu, 2024; Tan et al., 2024). Building on earlier studies that applied machine learning techniques to gameplay data in Zoombinis, process-oriented CT assessment has been increasingly operationalised through the analysis of behavioural logs and derived indicators (Rowe et al., 2021; Liu, 2024). These studies demonstrated that variables such as duration, accuracy, number of actions, and sequences of interactions can be used to identify patterns of learners' problem-solving behaviour and to infer aspects of computational thinking practices. However, the level at which these strategies or CT-related behaviours are defined often remains implicit or task-specific, making it difficult to determine how consistently such representations apply across different puzzles or contexts.

More recent work further extends this approach by introducing a structured assessment framework based on evidence identification (EI) at the puzzle level and evidence accumulation (EA) across puzzle levels (Rowe et al., 2025). In this framework, process data are operationalised through several basic observables derived from gameplay logs, including the percentage of successfully completed solutions, number of perfect rounds, average duration, number of rounds, and overall puzzle outcome. In addition, for selected puzzles, these indicators are complemented by machine-learning-based detectors, which estimate the probability that a learner demonstrates specific CT practices during gameplay (Rowe et al., 2025).

At the same time, this line of work also reveals important methodological limitations. The detector-based approach is applied only to a subset of puzzle types, while most tasks rely on more aggregated indicators derived from gameplay logs. Furthermore, the construction of such detectors depends on manually labelled gameplay data and puzzle-specific modelling decisions, which limits scalability and makes it difficult to apply the same analytical framework consistently across all tasks. As a result, even within the same

environment, the granularity of process analysis and the definition of strategies remain uneven across tasks (Rowe et al., 2025).

A similar pattern can be observed in other process-based CT assessment environments. While systems such as Blue Ant Code (Zapata-Cáceres et al., 2021), PhysGramming (Kanaki & Kalogiannakis, 2022), ctGameStudio (Werneburg, 2018), and Kodetu (Guenaga et al., 2021) demonstrate that interaction data can be used to analyse learners' behaviour. The types of process indicators vary considerably, including measures such as time on task, number of attempts, interaction logs, and action sequences. However, these indicators are typically defined in relation to specific environments and task designs, and studies often do not explicitly clarify how strategy representations are defined, at what level they operate, or how they can be generalised across different tasks or contexts (Liu, 2024; Tan et al., 2024). As a result, existing approaches tend to remain fragmented and context-dependent, highlighting a methodological gap for more unified and transferable models of process-based CT assessment.

Across these studies, a common tendency is the increasing use of fine-grained behavioural data like mouse clicks, keystrokes, time intervals, gaze patterns, to gain insight into learners' cognitive and strategic processes (Hwan et al., 2023). When combined with advanced analytical methods such as machine learning or learning analytics, these data allow researchers to model how students decompose problems, recognise patterns, and abstract solutions, going beyond simple correctness-based scoring (Karakaş & Yöndem, 2020). Recent work also investigates AI-driven feedback and scaffolding systems that integrate CT assessment directly into the learning environment. For example, machine-learning-based adaptive feedback systems analyse students' code solutions and CT difficulties in real time to generate personalised hints and remediation paths, thereby linking assessment with instructional support (Kaleem et al., 2024). Similarly, studies on "Scaffolding Computational Thinking With ChatGPT" (Liao et al., 2024) demonstrate how large language models can be used to diagnose misconceptions, provide step-by-step guidance, and offer reflective prompts during CT tasks, effectively acting as an intelligent assessment and tutoring agent embedded in students' problem-solving workflows. Nevertheless, most of these tools are custom-built for specific age ranges or curricular contexts, which limits their generalisability and large-scale deployment.

In the analysed papers, additional tools for CT assessment are proposed. Kert, Kalelioğlu & Gülbahar (2019) analyse the holistic approach to teaching

CT and prepared their own test “to measure the students’ growth in academic achievement and proposed it as a sample assessment tool to computer science teachers”. Other authors developed assessment tests to analyse CT competence from various angles. “An assessment was designed and conducted to evaluate the performance based on a set of multiple-choice questions to understand the gender differences in the program learning environment and CT ability” (Wu & Su, 2021). Yağcı (2019), in his study, developed and validated a CT competence test. “The five-point Likert scale consists of 42 items under four factors. The scale was developed based on the skills in the definition of ISTE (2015)”. Grgurina et al. (2018) designed an assessment instrument for the assessment “of the intended learning outcomes for computational science for modelling and simulation in secondary computing science education”. To validate CT competence improvement using a learning methodology based on metaphors and Scratch, Pérez-Marín et al. (2020) used ROMT (a validated test for children to measure CT) and PCNT (a new test to measure CT based on the field's literature) tests. Similar to Dr. Scratch, the Scratch program analysis tool (SAT), based on ANTLR, is also used for CT assessment. The tool was developed to address some flaws (e.g., high failure rate and low efficiency) in Dr. Scratch (Chang et al., 2018). One more interesting approach to CT assessment is CT-cube, a framework for the design, realisation, analysis, and assessment of CT activities. The CT-cube allows for extending existing computational thinking models to consider the life-long development of CT competence in individuals, from childhood to adulthood, and to take into consideration the situated nature of CT activities (Piatti et al., 2022).

This review concludes that the most popular method of assessing CT remains the various versions of tests. One of the most commonly used tools for assessing CT without linking teaching and assessment to programming is the CTt or Bebras Tasks. The CT assessment review results also indicate that more CT assessments are needed for high school, college students, and teacher professional development programs. More reliability and validity evidence needs to be collected and reported in future studies (Tang et al., 2020). At the same time, there is a clear shift towards process-oriented, data-rich assessment approaches, which leverage behavioural traces and learning analytics to complement traditional tests. Of relevance to the present research, interactive Bebras tasks constitute a widely deployed, domain-independent environment that already generates such behavioural data, yet has not been systematically exploited for automated, process-based CT assessment.

1.3 Learning Analytics: Concepts, Methods, and Relevance for CT Assessment

Learning analytics (LA) has evolved into a mature interdisciplinary field concerned with extracting actionable insights from learner-generated data. The widely adopted definition by the Society for Learning Analytics Research (SoLAR) in the First Learning Analytics and Knowledge Conference in 2011 (LAK-11) describes LA as "the measurement, collection, analysis and reporting of data about learners and their contexts, for the purpose of understanding and optimising learning and the environments in which it occurs". Over the last decade, LA has expanded from higher education into K-12, professional learning, and broader digital ecosystems, driven by the increasing availability of behavioural, cognitive, and multimodal data. Recent research further emphasises evidence-based, feedback-oriented analytics, positioning LA as an active component of learning design rather than a post-hoc evaluation mechanism (Yan et al., 2024).

Another term often used together with or instead of Learning Analytics is Educational Data Mining (EDM). EDM is an emerging interdisciplinary research area that deals with the development of methods to explore data originating in an educational context. EDM is a field that exploits statistical, machine-learning, and data mining algorithms over different types of educational data (Romero & Ventura, 2010). So, what is the difference between Learning Analytics and Educational Data Mining? Both EDM and LA reflect the emergence of data-intensive approaches to education, and there are similarities between EDM and LA, which suggest several areas of overlap (Siemens & Baker, 2012). Siemens & Baker (2012) mention these main differences:

- EDM has a primary focus on automated discovery, whereas LA has a stronger focus on leveraging human judgment.
- EDM models are often used as the basis for automated adaptation, conducted by a computer system, whereas LA models are often developed to inform instructors and learners.
- EDM researchers use reductionist frameworks: they reduce phenomena to components and focus on the analysis of individual components and relationships between them. By contrast, LA researchers have a stronger focus on understanding complex systems as wholes (Viberg et al., 2018).

In addition to EDM and LA, there are also other related terms used as data analytics methods in the education sphere. Romero & Ventura (2020) distinguish the following terms:

- Academic Analytics (AA) and Institutional Analytics (IA) is concerned with the collection, analysis, and visualization of academic program activities such as courses, degree programs; research, revenue of students' fees, course evaluation, resource allocation, and management to generate institutional insight (Campbell, DeBlois, & Oblinger, 2007; Siemens and Long, 2011). So, it is focused on the political/economic challenge.
- Teaching Analytics (TA) refers to the analysis of teaching activities and performance data as well as the design, development, and evaluation of teaching activities (Prieto et al., 2016). It is focused on the educational challenge from the instructors' point of view. Data-Driven Education (DDE) and Data-Driven Decision-Making in Education (DDDM) refer to systematically collecting and analysing various types of educational data to guide a range of decisions to help improve the success of students and schools (Custer et al., 2018; Datnow & Hubbard, 2016). Big Data in Education (BDE) refers to applying big data (basic connotation summed up in volume, variety, value, and velocity) techniques to data from educational environment (Daniel, 2019). Educational Data Science (EDS) is defined as the use of data gathered from educational environments/settings for solving educational problems (Romero & Ventura, 2017). Data science is the concept unifying statistics, data analysis, machine learning, and related methods.

Recent work on human-centred learning analytics and AI in education further refines these distinctions by emphasising that data-intensive systems must be designed not only for prediction and optimisation, but also for transparency, safety and stakeholder agency (Alfredo et al., 2024). Human-centred LA/AIED approaches highlight the need to balance automation with human control and to explicitly consider ethical and pedagogical implications when deploying analytics at scale, which is particularly relevant for CT assessment scenarios described in Chapters 1.1 and 1.2.

1.3.1 Learning Analytics in the Learning Process

Learning analytics can improve learning practice by transforming the ways we support learning processes: improve learning outcomes, support learning and teaching (Viberg et al., 2018). The studies' results that provide some evidence in improvements of learning outcomes focus mainly on three areas (Viberg et al., 2018):

- knowledge acquisition, including improved assessment marks and better grades;
- skill development;
- cognitive gains.

Evidence that Learning Analytics improves learning support and teaching (e.g. retention, progress, and completion) was most prominent in Ferguson and Clow's (2017) and Viberg et al. (2018) review articles.

More recent reviews demonstrate that the impact of learning analytics increasingly depends on how feedback and reflection are designed. Yan et al. (2024) show that evidence-based multimodal learning analytics, which combine log data with interaction and sensor traces, can provide richer, more actionable feedback to learners and teachers. Their review indicates that analytics-driven feedback is most effective when it supports learners' self-reflection on strategies and processes rather than only reporting performance outcomes.

At the same time, a systematic review of human-centred LA/AIED systems by Alfredo et al. (2024) underscores that learning analytics and AI can only be sustainably integrated into educational practice if stakeholders, especially teachers and students, are actively involved in the design and deployment of these systems. Their findings show that many LA/AIED tools still offer limited human control and insufficient transparency, which may lead to mistrust or misalignment with pedagogical goals. This has important implications for CT assessment tools based on learning analytics: process-level CT analytics should be explainable and controllable by educators, not only technically accurate.

Finally, Mohammadi et al. (2025) analyse AI-enhanced multimodal learning analytics and conclude that AI is increasingly used to process complex behavioural data (e.g., logs, eye-tracking, video, speech) and to generate personalised feedback. However, they also highlight gaps related to theoretical grounding and ethical use of AI. These insights suggest that, for CT assessment based on rich process data (e.g., interactive Bebras tasks), LA

methods need to be theoretically aligned with CT constructs and designed in a human-centred way.

1.3.2 Computational Methods for Data Analysis

Merceron (2015) identifies the following computational methods used in learning analytics:

- Prediction: A major task tackled by prediction methods is to predict the students' performance. The most common methods include regression and classification.
- Clustering: Clustering techniques are used to group objects so that similar objects are in the same cluster and dissimilar objects are in different clusters.
- Relationship Mining: This category includes such methods as association rule mining, correlation mining, sequential pattern mining, and causal data mining.
- Distillation of data for human judgement: This category includes statistics and visualizations that help humans make sense of their findings and analyses.
- Discovery with models: This category encompasses approaches in which the model obtained in a previous study is included in the data to discover more patterns.

Historically, predictive methods dominated LA, but since 2016, their prevalence has decreased, while interest in relationship mining and interpretive analytics has grown (Viberg et al., 2018). The expansion of dashboards and visual analytics (Molenaar et al., 2020; Van Leeuwen & Rummel, 2020) reflects increasing demand for interpretable, user-centred analytic systems. Yet dashboard research shows persistent challenges: lack of theoretical grounding, limited validated instruments, and overemphasis on usability rather than learning impact (Jivet et al., 2018).

Contemporary LA extends these methodological categories into multimodal and AI-enhanced environments. Yan et al. (2024) demonstrate that prediction, clustering, and sequential pattern mining are increasingly applied to complex multimodal datasets, enabling more granular analysis of learning strategies. Mohammadi et al. (2025) show that AI now permeates the entire multimodal LA pipeline, from feature extraction and data fusion to

model learning and intervention design, supporting deep learning models capable of recognising behavioural patterns invisible to traditional analytics.

These methodological advances are directly relevant to CT assessment. For instance:

- Clustering enables identification of CT strategy profiles, addressing gaps noted in CT assessment research (Corrales-Álvarez et al., 2024; Luo & Zhang, 2024).
- Sequential pattern mining supports analysis of problem-solving processes is critical for tasks such as interactive Bebras puzzles, where solution behaviour is temporal and exploratory.
- Multimodal modelling parallels efforts in CT studies using log-based assessments, game data (e.g., Zoombinis), or automatic code-analysis tools (e.g., Dr. Scratch).

Thus, LA computational methods offer a robust methodological foundation for advancing CT assessment from final-answer scoring to behavioural, process-oriented evaluation.

1.4 Chapter Summary and Research Gaps

Across the reviewed literature, three interconnected research strands emerge: work on conceptualizing and measuring CT, studies developing or evaluating concrete assessment tools, and research on learning analytics and other data-driven approaches to understanding learning. Taken together, these strands depict a field that is expanding rapidly, driven by advances in educational technology and data availability, yet still marked by fragmentation in methods, limited theoretical alignment, and uneven integration of process-oriented evidence into assessment practices.

First, CT assessment is still dominated by psychometric tests and programming-based artefact analysis. Widely used instruments such as CTt, BCTt, and CCTt provide validated and scalable measurement of core CT constructs, while additional CT scales and questionnaires extend assessment into attitudinal and dispositional domains (Román-González et al., 2017; Zapata-Cáceres et al., 2021; El-Hamamsy et al., 2025). Artefact-based tools, including Dr. Scratch (Moreno-León & Robles, 2015), the Scratch Analysis Tool (SAT) (Chang et al., 2018), and game-based environments such as Zoombinis, Blue Ant Code, or modelling and simulation platforms (Asbell-

Clarke et al., 2021; Zapata-Cáceres et al., 2021; Grgurina et al., 2018), expand assessment into more authentic and interactive settings. However, most existing methods rely primarily on final outcomes, coarse scores, or static products, offering only partial insight into learners' solution strategies, intermediate choices, and process-level CT competencies such as decomposition, pattern recognition, algorithmic reasoning, and abstraction (Lockwood & Mooney, 2017; Corrales-Álvarez et al., 2024).

Second, although interactive tasks and games naturally produce rich behavioural traces such as action sequences, clickstreams, time-on-task and intermediate representations, few CT assessment systems systematically exploit these data. The Bebras Challenge illustrates this gap. Despite its widespread use as a CT benchmarking tool (Izu et al., 2017; Djambong et al., 2018; Labusch & Eickelmann, 2020), scoring remains almost exclusively binary (correct/incorrect), and the underlying problem-solving processes remain largely unexamined. Emerging work on interactive Bebras tasks (Belletini et al., 2023; Luo & Zhang, 2024) and process-rich games such as Zoombinis (Rowe et al., 2021) demonstrates the feasibility of behaviour-based, process-oriented CT assessment, yet these studies remain exploratory, context-specific, and methodologically unaligned. More detailed analysis of these approaches reveals several additional methodological limitations. First, the process indicators used across studies vary substantially in their level of granularity, ranging from aggregated measures such as time and number of actions to more complex detector-based representations derived from interaction logs. Second, strategy identification is often defined at different levels and remains task – or environment-specific, making it unclear how consistently such representations apply across different tasks or contexts (Liu, 2024; Tan et al., 2024). Third, even in more advanced approaches, such as Zoombinis-based assessment, machine-learning detectors are applied only to specific puzzle types and rely on manually labelled gameplay data, which limits scalability and reduces the generalisability of these methods across tasks (Rowe et al., 2025).

Third, research in learning analytics (LA) and educational data mining (EDM) provides a sophisticated methodological toolbox: prediction, clustering, relationship mining, sequential pattern mining, and multimodal modelling for analysing complex behavioural and multimodal data (Merceron, 2015; Romero & Ventura, 2020). Recent developments in multimodal LA and AI-enhanced analytics show how heterogeneous data streams (log data, video, sensor traces) can be transformed into interpretable learning indicators,

strategy profiles, and personalised feedback mechanisms (Yan et al., 2024; Mohammadi et al., 2025). Human-centred LA/AIED research further emphasises transparency, ethical use, and stakeholder agency in data-driven educational systems (Alfredo et al., 2024). Despite these advances, LA methods have only marginally been applied to CT assessment, and their application to non-programming CT contexts such as Bebras remains particularly rare, representing a substantial opportunity for methodological innovation.

Overall, the chapter points to a clear gap: there is no widely validated, domain-independent, automated CT assessment approach that leverages behavioural process data from non-programming tasks, uses LA different ML methods to identify solution strategies and CT profiles, and remains interpretable and usable for teachers.

The empirical study presented in the following chapters addresses this gap in the following ways:

- Using interactive Bebras tasks as a large-scale, domain-independent CT assessment context.
- Collecting and modelling fine-grained behavioural data (click sequences, action paths, time features) from students' problem-solving processes.
- Applying learning analytics and clustering methods to identify typical CT solution strategies and relate them to performance.
- Exploring how such process-based indicators could complement traditional correctness-based scoring in CT assessment.

In doing so, the study builds directly on the CT assessment and LA literature reviewed in this chapter and proposes a process-oriented, data-driven framework for assessing CT via interactive Bebras tasks.

2. PILOT STUDY AND INITIAL BEHAVIOUR ANALYSIS

2.1 Pilot Task Description

A variety of paper-based and digital tests are commonly used to assess computational thinking. However, unlike in mathematics, where partial solutions and intermediate reasoning steps are considered, most CT assessments evaluate only the final answer. In many widely used CT instruments, especially those based on multiple-choice questions or short-response formats, the problem-solving process remains invisible. Programming tasks are an exception because they naturally expose procedural steps, but programming represents only one domain of CT and is not equally accessible to all learners.

For this reason, an increasing number of CT assessments rely on interactive problem-solving tasks. One well-established example is the Bebras Challenge, where learners solve short informatics tasks by manipulating visual elements, exploring constraints, and making sequential decisions. Although learners engage in multiple actions while solving these tasks, the official scoring considers only whether the final answer is correct. As a result, essential diagnostic information is lost, learners who use different reasoning strategies may obtain the same score, and a single late-stage error in a multi-step solution may yield a completely incorrect result.

To examine whether process data can reveal meaningful patterns of reasoning, this pilot study employed an interactive version of the Coloring Page task from the 2022 International Bebras Challenge. Bebras tasks are published under a Creative Commons licence, which allows their use for research. The task consists of 12 distinct regions, and the learner must colour each region using one of three colours: yellow, green, or blue. The task is governed by a single constraint: no two adjacent regions may share the same colour. Although originally designed for younger pupils, the task contains sufficient structural complexity to elicit diverse problem-solving strategies among learners of various ages.

From a computer science perspective, the task is an instance of the classical graph colouring problem. Formally, it can be represented as a graph $G = (V, E)$, where:

- V is the set of 12 regions (vertices);

- E is the set of adjacent pairs between regions (edges);
- c is a colouring function assigning a colour to each vertex:

$$c: V \rightarrow \{\text{yellow,green,blue}\}.$$

The constraint

$$c(v_i) \neq c(v_j) \quad \forall (v_i, v_j) \in E$$

requires that adjacent regions must be assigned different colours.

In this notation, $v_i$and $v_j$denote two distinct regions of the task image. The subscripts i and $j$are indices used to refer to specific elements of the vertex set V. The function $c(v)$returns the colour assigned to a region v; therefore, $c(v_i)$ and $c(v_j)$represent the colours of neighbouring regions. The constraint $c(v_i) \neq c(v_j)$ensures that no pair of adjacent regions (i.e., regions connected by an edge in E) may share the same colour.

This formalisation aligns with the conceptual explanation provided in the original task description, which highlights the connection between colouring problems and real-world applications such as map colouring, scheduling, and conflict-avoidance tasks.

To illustrate the task structure and the constraints that learners must satisfy, the original Bebras task is presented in Figure 2.1.

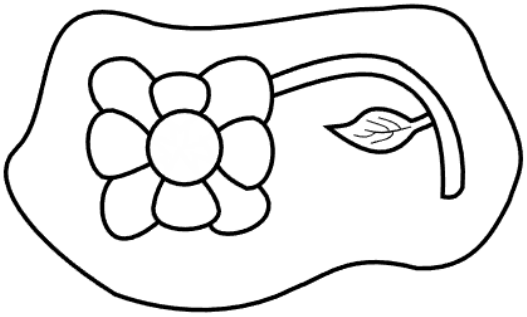


Figure 2.1: Original Bebras “Coloring page” task

Although the task appears simple, it exhibits a non-trivial combinatorial structure. With twelve regions and three colours, the unconstrained search space contains 3^{12} possible colour assignments. Finding a valid colouring requires recognising and managing adjacency constraints, revising earlier choices when conflicts occur, and maintaining consistency across multiple dependent decisions.

As a result, learners naturally exhibit different types of reasoning strategies, including:

- Systematic exploration (colouring regions in a deliberate order);
- Constraint-based reasoning (making decisions based on conflict detection);
- Heuristic backtracking (undoing incorrect assignments);
- Trial-and-error behaviour (repeated recolouring with limited forward planning).

The interactive version used in this study logged every colouring action, including region selections, recolouring attempts, conflict resolution steps, and resets. User interactions were collected within Bebras task-solving environment, where a dedicated event-logging mechanism was implemented to systematically record each user action as structured data. These logs produce a fine-grained behavioural trace, enabling the reconstruction of learners' solution paths and the identification of characteristic reasoning patterns that would not be observable in traditional final answer scoring systems.

An example of a correctly completed colouring configuration is presented in Figure 2.2.

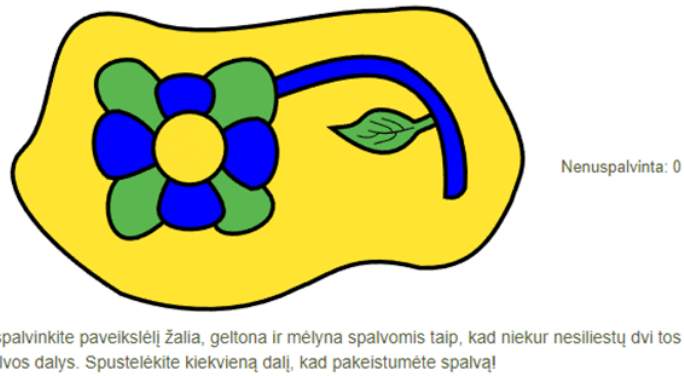


Figure 2.2: Example of a correctly completed solution

To enable the analysis of these reasoning behaviours and to evaluate how learners interacted with the task, all user actions were systematically captured through an event-logging system. The following section describes the data collection setup and the procedures used to record, store, and prepare the interaction data for analysis. This pilot study is partially based on the results published in Masiulionytė-Dagienė and Jevsikova (2025a).

2.2 Pilot Task Data Collection

The pilot experiment was conducted with learners from secondary and upper-secondary school levels (grades 7-12) as well as undergraduate university students. Since the purpose of this study was to analyse task-solving behaviour rather than to compare performance across different demographic groups, participant characteristics such as age or study level were not considered essential for interpretation. The analysis focuses exclusively on the structure and dynamics of the solution process, independent of who the solver is.

All collected data were fully anonymised. No personally identifiable information was stored, and no mechanism exists to link any solution attempt to a specific individual. This was intentional, as the study evaluates only problem-solving behaviour rather than personal characteristics of participants.

2.2.1 Data Logging Mechanisms

The interactive colouring task consisted of 12 clickable regions, representing the different elements of the task interface (petals, flower centre, stem, leaf, and background). For consistency with the event logs and subsequent data processing, these regions are hereafter referred to as objects. This terminology corresponds to the id values in the JSON structure used to store the logged data.

Each interaction produced a logged event represented as a triplet:

$$e_k = (o_k, c_k, t_k),$$

where o_k denotes the object identifier, c_k the resulting colour value after the click, and t_k the timestamp is measured in milliseconds since the Unix epoch.

A complete attempt at a solution, therefore, forms an ordered sequence of events:

$$E = (e_1, e_2, \dots, e_n).$$

Objects do not have a direct colour-selection mechanism. Instead, each click cycles the object's colour through a fixed sequence. Every object starts in the white (0) state, and each click advances the colour in the following cycle:

$$\text{white}(0) \rightarrow \text{yellow}(1) \rightarrow \text{blue}(2) \rightarrow \text{green}(3) \rightarrow \text{white}(0).$$

Formally, if s_k denotes the colour state after the k-th click on the same object, the update rule is:

$$s_k = (s_{k-1} + 1) mod 4.$$

Thus, multiple consecutive clicks on the same object typically represent attempts to reach the desired colour rather than distinct problem-solving steps.

2.2.2 Raw Data Format

All logs were stored in JSON format. Each solution record contains:

- user_ID – anonymised solver identifier;
- task_ID – identical for all entries;
- task_state – final colours and the complete sequence of click events;
- solved_flag – indicating whether the final colouring satisfied all constraints.

A shortened JSON example:


```
{
"task_id": 3661,
"task_state": "[[3,1,2,2,2,2,2,1,1,1,1,3],
[{\"id\":\"path1465\",\"value\":1,\"time\":167
7234620662},
{\"id\":\"path1465\",\"value\":2,\"time\":1677
234621141},
  ...
{\"id\":\"path1445\",\"value\":1,\"time\":1677
234637309}]]",
"solved_flag": "T"
```

},

The JSON structure corresponds directly to the formal event representation:

- "id" → object identifier o_k;
- "value" → colour value c_k;
- "time" → timestamp t_k.

The first array in task_state contains the final colours of all 12 objects. The final colour of each object v_i was determined by locating the last event in the chronological click sequence in which that object appeared and taking the colour value recorded at that moment.

During the initial design of the logging mechanism, it was intended to record not only click events but also mouse-hover data, capturing every instance when the cursor moved over a particular object. In this study, hover actions refer to cursor movements over interactive elements without executing a click, potentially indicating exploratory or hesitant behaviour. The rationale was that hover behaviour might reveal additional aspects of exploratory or hesitant reasoning. However, preliminary inspections of early test logs showed that hover events generated extremely large volumes of noisy data, defined here as incidental and non-informative interactions, dominated by rapid and unintended cursor movements that did not correspond to meaningful cognitive actions. The density and unpredictability of hover events made them difficult to interpret reliably and risked overshadowing the far more informative click-based interactions. For this reason, hover data were excluded from the final dataset, and only click events were retained for analysis.

In total, around 700 solutions were obtained, but after cleaning the transformed data, 336 remained, which are analysed further. Blank solutions, duplicates (which could have been influenced by the specific functionality of the task system), and outliers such as very few filled cells or an exceptionally high number of clicks were rejected. The data cleaning process was semi-automated: problematic cases, such as duplicates and outliers, were first identified through manual inspection of the transformed data and then were removed using query-based filtering according to the defined criteria.

2.2.3 Data Transformation

Raw logs were transformed into a structured dataset by extracting, for each object, the following attributes:

1. Number of clicks – the total count of events involving the object.
2. Final colour – the colour assigned during the object's last recorded click.
3. Reduced action sequence and position encoding. To capture the sequence of distinct object selections, consecutive repeated clicks on the same object were removed. This produced the reduced sequence:

   $$R' = (r'_1, r'_2, \dots, r'_m),$$

   where each r'_j denotes the object ID at the j-th distinct step in the solution process.

   A binary vector was then constructed to encode the positions where a given object v_i appears in R':

   $$b_j = \begin{cases} 1, & r'_j = v_i, \\ 0, & \text{otherwise}, \end{cases} \qquad j = 1, \dots, m.$$

   This binary sequence was further encoded into a decimal value:

   $$D(v_i) = \sum_{j=1}^{m} b_j \cdot 2^{m-j}.$$

   This compact representation provides a numerical signature of the participation pattern of the object v_i during the distinct action steps of the solution process.

4. Total number of clicks – the sum of all clicks across all objects.
5. Total solution time – the time difference between the timestamps of the first and last recorded events.

Figure 2.3 presents a processed data example.

lumn1.path1404_clickcount	Column1.path1404_colour	Column1.path1404_clicksequencebin	Column1.path1404_clicksequence
9	1	01000000001000000001000000001	268960769
3	3	00000000001000000000000000000010	33554434
9	1	00000000000100000000000001000000000	16778240
11	3	01000010000000010000000	4325504
5	1	10000000000010000000000000	134234112
5	1	00000000000010000000010101	8213
3	3	00001000010000000000000000000001	1107296257
3	3	00000000000000000010000000100000	16416
1	1	00000000000000000000000000000010	2
3	3	10000000000000000000	4194304
1	1	00000000000000000100000000000	2048
2	2	00000000000010000000100000000000	8404992
1	1	00000000000000000000000000000010	2

Figure 2.3: Snapshot of the processed data

This cleaned dataset forms the basis for the modelling and analysis presented in the subsequent sections.

2.3 Identification of Solution Behaviour Patterns

The structured interaction data prepared in the previous section were used in an exploratory clustering experiment aimed at determining whether distinct and meaningful solution behaviours could be identified.

Each solution attempt was represented using a set of attributes extracted from the processed interaction logs. These attributes reflect both global solving characteristics and detailed object-level behaviour. For clustering purposes, the following categories of attributes were used:

- Global task-level attributes are:
    - Total number of clicks performed during the attempt;
    - Total time spent solving the task.
- Per-object behavioural attributes. For each of the 12 objects, the following were included:
    - Number of clicks applied to the object;
    - Final colour after the last click;
    - Behavioural sequence descriptors such as:
        - The position in the overall interaction sequence where the object was first clicked.
        - The number of times the solver returned to the same object after moving to others.

- Whether the object appeared early or late in the solution process.

All attributes were combined into a single tabular dataset in which each row represents a solution attempt, and each column represents an attribute.

Because different behavioural attributes operate on different numerical scales, min-max normalisation was applied independently to each attribute *j* across all solution attempts. The normalised value of the attribute *j* for solution i is computed as:

$$z_{i,j} = \frac{x_{i,j} - min_i(x_{i,j})}{max_i(x_{i,j}) - min_i(x_{i,j})}.$$

Here, i indexes solution attempts, *j* indexes attributes, $x_{i,j}$ is the value of the attribute *j* for solution i, and $\min_i(x_{i,j})$ and $\max_i(x_{i,j})$ are the minimum and maximum values of that attribute across all attempts. Thus, each attribute is normalised independently, and every normalised column spans the interval *[0, 1]*.

An illustrative fragment of the normalised behavioural feature dataset is presented in Table 2.1, demonstrating how the extracted attributes (solution time, click counts, final states, and sequence descriptors) appear after preprocessing and min-max normalisation to the *[0, 1]* interval.

Table 2.1: Fragment of the normalised feature dataset

...	Feature_14	Feature_15	Feature_17	Feature_18	Feature_19	...
...	0.538462	1.000000	0.000004	0.230769	1.000000	...
...	0.153846	0.666667	0.499840	0.769231	0.666667	...
...	0.461538	0.666667	0.007811	0.461538	0.666667	...
...	...	...	...	...	...	...

2.3.1 Exploratory Clustering Using k-means

To investigate whether meaningful behavioural structure exists in the interaction data, the k-means (MacQueen, 1967) clustering algorithm was applied to the normalised dataset. Given the set of normalised solution representations

$$Z = \{z_1, z_2, \dots, z_N\},$$

the algorithm partitions these observations into k clusters by minimising the within-cluster sum of squared distances:

$$J = \sum_{i=1}^{N} \| z_i - \mu_{c(i)} \|^2,$$

where $c(i)$ denotes the index of the cluster assigned to the solution i, and $\mu_{c(i)}$ is the centroid of that cluster.

The k-means algorithm proceeds iteratively using two steps:

1. Assignment step:

   Each solution z_i is assigned to the closest cluster centroid according to Euclidean distance:

   $$c(i) = \arg\min_k \| z_i - \mu_k \|.$$

2. Update step:

   For each cluster k, the centroid is recomputed as the mean of all solutions currently assigned to that cluster:

   $$\mu_k = \frac{1}{| C_k |} \sum_{i \in C_k} z_i,$$

   where $C_k$is the set of indices of solutions belonging to the cluster k.

These two assignment and update steps are repeated until convergence, typically defined as the point at which centroid movement falls below a specified threshold or no longer changes.

Because the clustering stage was exploratory, no prior assumption was made regarding the number, stability, or interpretability of the resulting clusters. The primary objective was to determine whether the behavioural attributes exhibited a separable structure that could be interpreted as distinct solution strategies.

The clustering experiment was implemented using Microsoft Azure Machine Learning Studio (hereafter Azure ML Studio). The platform was used primarily as a workflow environment for executing the k-means

algorithm, uploading the dataset, and experimenting with different feature subsets and numbers of clusters.

Data cleaning, feature extraction, and categorical encoding were performed prior to uploading the dataset. During this stage, raw interaction logs were transformed into structured numerical attributes represented as decimal values. The resulting .csv file was then loaded into Azure ML Studio, where feature normalisation was applied using the built-in “Normalize Data” module.

In the Azure ML Studio experiment, the dataset was processed using the platform’s modular clustering workflow, where cluster centres were estimated using the “Train Clustering Model” component and subsequently applied using the “Assign Data to Clusters” component. The data flow reflects the technical configuration of the environment used at the time of the experiment. The process flow is presented in Figure 2.4.

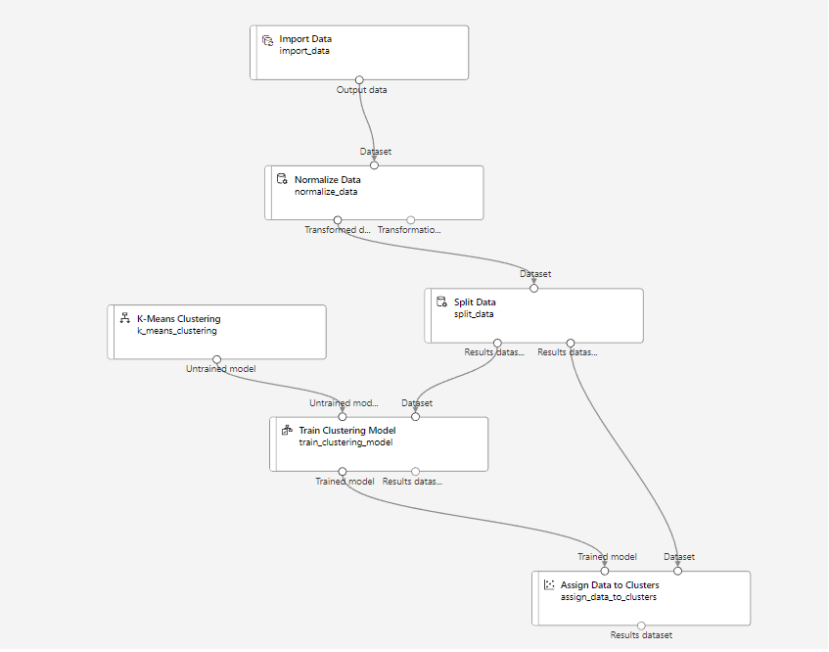


Figure 2.4: Data clustering process diagram

To determine whether the interaction data contained multiple distinct solution patterns, several feature-space configurations were evaluated. This step was essential for understanding which behavioural dimensions

contributed most to the emergence of meaningful clusters. Because the attributes encoded different aspects of problem-solving (click quantity, object order, colour selection, object revisits), reducing or expanding the feature space directly affected the level of detail available for differentiating solution strategies.

Four primary feature-set variants were examined:

1. Full behavioural set. Included all available attributes:
    - Global task-level features (total clicks, total time);
    - Per-object click counts;
    - Per-object final colours;
    - Sequence descriptors (first click order, revisit structure).
2. Object-only set. Excluded global time and total clicks, focusing solely on per-object behaviour.
3. Click-sequence set. Contained only click counts and sequence descriptors but omitted final colours.
4. Reduced click-count only set. Included only the total number of clicks and per-object click counts.

The clustering experiments demonstrated that the choice of behavioural attributes had a substantial impact on the structure and interpretability of the resulting clusters. When only reduced feature sets were used, such as click counts alone or simplified object-level descriptors, the algorithm tended to produce a small number of large, heterogeneous clusters and several degenerate clusters containing only one or two solutions. This indicated that such reduced representations did not capture enough behavioural variation to meaningfully distinguish different approaches to solving the task. In particular, the absence of sequence-related information made it impossible to separate solvers who followed systematic strategies from those who engaged in more exploratory or trial-and-error behaviours, even when their total number of clicks was similar.

Including sequence descriptors notably improved the clustering structure. These attributes reflect the temporal flow of interactions, such as the order in which objects were visited or how frequently solvers returned to previously

adjusted objects, and proved essential for revealing behavioural differences that were not evident from click counts alone. However, the most consistent and behaviourally interpretable results were obtained only when all available attributes were combined: global measures (total time and total clicks), per-object click counts, final colours, and sequence-related descriptors. This full behavioural representation preserved the richness of the interaction data and allowed the algorithm to identify clear patterns.

When testing different numbers of clusters, values below six led to overly broad groupings that merged distinct solving strategies into single clusters. As the number of clusters increased, more patterns became visible. The configuration with eight clusters provided the best balance between granularity and interpretability. The selection of eight clusters was based on empirical evaluation, prioritising the interpretability of cluster centres and the meaningful differentiation of behavioural patterns rather than relying on formal statistical criteria.

Overall, the results demonstrate that the interaction data contained sufficient internal structure to support the identification of multiple solution patterns. The emergence of eight distinct behavioural clusters shows that even a single interactive task can elicit a rich diversity of approaches, which becomes visible only when fine-grained interaction features are included in the analysis. These findings are analysed in the following chapter.

2.3.2 Representative Solutions and Behavioural Profile Reconstruction

To enable a qualitative interpretation of the clusters, representative solutions were selected from each cluster. For every cluster, the solution attempt whose normalised feature vector was closest to the cluster centroid was identified. When available, both a correct and an incorrect representative solution were chosen. This centroid-based selection ensures that the examples reflect the most characteristic behaviour within each cluster rather than outliers or marginal cases. The selected representative solutions are summarised in Figure 2.5.

Column3	Column4	Assignments	DistancesToClusterCenter no.0	DistancesToClusterCenter no.1	DistancesToClusterCenter no.2	DistancesToClusterCenter no.3	DistancesToClusterCenter no.4	DistancesToClusterCenter no.5	DistancesToClusterCenter no.6	DistancesToClusterCenter no.7
2e2e9e09-4917-49d4-a2de-faf35ccaccf6	TRUE	0	0.294314	1.747291	1.758517	0.987052	1.044711	1.495699	0.973555	1.858878
6def33f8-564f-4b93-9384-b4ae8227e1a7	FALSE	0	0.596445	1.633166	1.580286	0.837654	1.274395	1.20947	0.769026	1.960889
e39ede04-acf5-473a-af59-b397ee6a45d7	TRUE	1	1.687974	0.288456	1.4247	1.815556	1.505734	1.023738	1.298506	1.679056
4c7f69de-f5d6-43d5-9144-edb9ced684e9	FALSE	1	1.74101	0.551861	1.444704	1.905394	1.529497	1.200659	1.427671	1.603401
82b63f47-104d-44cf-b83b-291b23c56913	TRUE	2	1.663649	1.449765	0.824766	1.303534	2.007432	0.899276	1.736883	1.99046
9c52041f-1759-4c03-8e24-31308484bfbc	FALSE	3	1.416724	2.434408	1.711697	1.07333	2.126087	1.776903	1.818405	2.480374
8d9bf1bf-01ff-4d26-a444-6b85650fc722	TRUE	3	0.827932	1.81477	1.305312	0.37361	1.733772	0.988786	1.341971	2.198429
cdb70e6c-a7d1-47bb-9c2a-c85c480b3263	TRUE	4	1.120318	1.303775	2.061832	1.81506	0.37084	1.803146	0.926457	1.41964
48d885b7-3c97-4506-98e1-8493ac3769de	FALSE	4	1.026674	1.400744	2.044724	1.726286	0.627167	1.761719	0.858235	1.559469
b930d1ae-5626-4cb5-a224-bbebe14eff15	TRUE	5	1.309231	1.206515	0.904686	0.973441	1.800277	0.174308	1.348279	2.03987
7ecafa9f-76f9-4547-a41d-1db7b2cdabb7	FALSE	5	1.426326	1.565462	1.215875	1.262558	1.857529	1.098084	1.752094	2.066592
108ef147-5724-4297-ae9d-03f443f2f94e	FALSE	6	1.279938	1.703006	2.1119	1.686054	1.228167	1.770958	0.638654	2.000389
e707a9dd-92c4-4c1c-afed-accf9991ad00	TRUE	7	1.564119	1.727817	2.003637	1.998661	1.253885	2.047332	1.65269	1.007387
c4112108-7630-4dcc-8105-72bc7fdeebe9	FALSE	7	2.391047	2.292464	2.573877	2.720138	1.991611	2.694548	2.121639	1.52302

Figure 2.5: Solutions are closest to the cluster centre

Because the clustering was performed on transformed behavioural attributes rather than on the raw interaction traces, each solution preserved a unique internal identifier. These identifiers made it possible to map centroid-nearest cases back to their initial solutions, allowing reconstruction of the complete solving process. This reconstruction step was essential, as it grounded the interpretation of clusters in actual behavioural sequences rather than in aggregated numerical features alone.

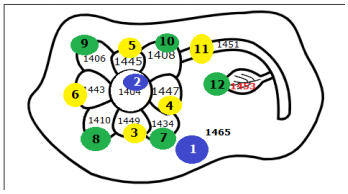
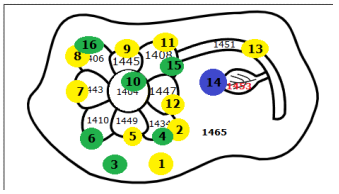
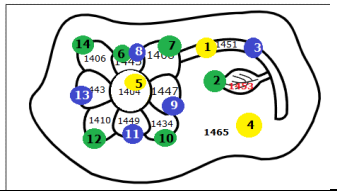
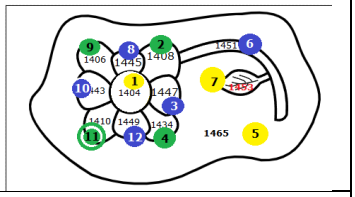
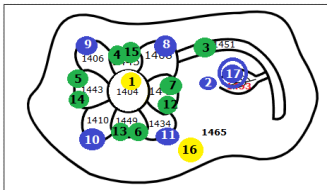
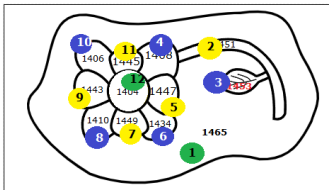
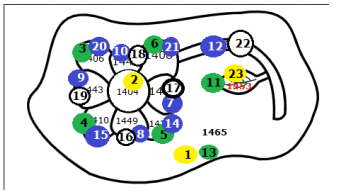
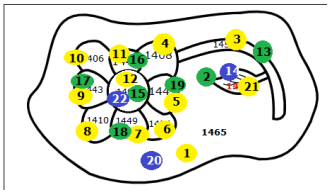
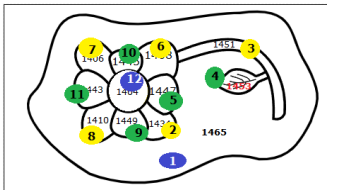
A fragment illustrating how representative cases were restored from the original dataset is presented in Figure 2.6.

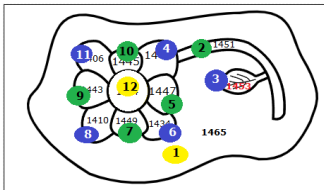
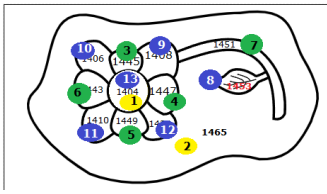
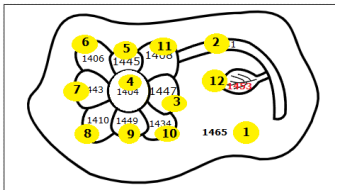
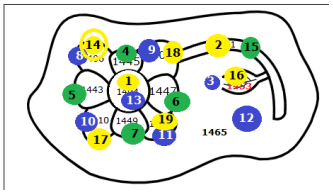
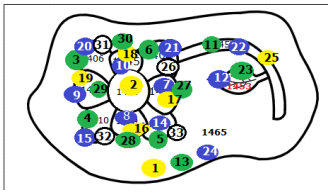
Column1.i	Column1.l	Column1.answer_id	Column1.s	Cluster	Colun	Column1.t	Colun	Colur	Column1.a	Column1.path	Column1.j	Column1.path1404_clicksequencebin	Column1.j	Co
360149	3662	2e2e9e09-4917-49d4-a2de-faf35ccaccf6	T	0	12	21	2023-(	2023-	31957	3	3	000000000010	2	
360179	3662	6def33f8-564f-4b93-9384-b4ae8227e1a7	F	0	12	20	2023-(	2023-	12612	2	2	000001000000000	512	
360189	3662	e39ede04-acf5-473a-af59-b397ee6a45d7	T	1	12	27	2023-(	2023-	19497	1	1	000000010000	16	
426563	3662	4c7f69de-f5d6-43d5-9144-edb9ced684e9	F	1	12	30	2023-(	2023-	29320	1	1	000000000001	1	
360179	3662	82b63f47-104d-44cf-b83b-291b23c56913	T	2	12	51	2023-(	2023-	63377	1	1	000000000000000001	1	
360182	3662	8d9bf1bf-01ff-4d26-a444-6b85650fc722	T	3	12	24	2023-(	2023-	34652	2	2	100000000000	2048	
360181	3662	9c52041f-1759-4c03-8e24-31308484bfbc	F	3	12	40	2023-(	2023-	74241	1	1	000000000000000000010	2	
360182	3662	cdb70e6c-a7d1-47bb-9c2a-c85c480b3263	T	4	12	25	2023-(	2023-	20262	3	3	100000100100000000000	2115584	
395739	3662	48d885b7-3c97-4506-98e1-8493ac3769de	F	4	12	21	2023-(	2023-	9815	3	3	100000000000	2048	
360181	3662	b930d1ae-5626-4cb5-a224-bbebe14eff15	T	5	12	27	2023-(	2023-	28752	1	1	100000000000	2048	
360183	3662	7ecafa9f-76f9-4547-a41d-1db7b2cdabb7	F	5	12	29	2023-(	2023-	137176	3	3	100000000001	4097	
360181	3662	108ef147-5724-4297-ae9d-03f443f2f94e	F	6	12	12	2023-(	2023-	6879	1	1	000000001000	8	
360183	3662	e707a9dd-92c4-4c1c-afed-accf9991ad00	T	7	12	49	2023-(	2023-	88478	3	3	000000100000000001	4097	
360183	3662	c4112108-7630-4dcc-8105-72bc7fdeebe9	F	7	12	55	2023-(	2023-	103624	1	1	000000000000000000000000000010	2	

Figure 2.6: Restored the initial solutions assigned to the clusters

A detailed summary of the reconstructed correct and incorrect solutions for each cluster is presented in Table 2.2. In this table, coloured circles indicate the final colour assigned to each object; numbers inside the circles represent the order in which objects were clicked; and an additional coloured ring denotes repeated colour changes applied consecutively to the same object; number inside each object indicates object ID.

Table 2.2: Represents original Correct (TRUE) and Incorrect (FALSE) task solutions from each cluster

Number of the cluster	Correct (TRUE) solution	Incorrect (FALSE) solution
0		
1		
2		THERE WERE NO INCORRECT (FALSE) SOLUTIONS IN THIS CLUSTER
3		
4		
5		

Number of the cluster	Correct (TRUE) solution	Incorrect (FALSE) solution
		
6	THERE WERE NO CORRECT (TRUE) SOLUTIONS IN THIS CLUSTER	
7		

A targeted post-analysis allowed clusters to be reorganised according to meaningful behavioural indicators: how consistently the rule was applied, whether errors were corrected, how many errors remained at the end, and whether solvers demonstrated full or partial understanding of the task. This refinement produced clusters that better reflect actual solution types and are more appropriate for assessment purposes.

Clusters with correct (TRUE) solutions

Cluster A: Systematic correct solvers (clusters 0, 3, 5, and partly 2).

Clusters 0, 3, and 5 share identical problem-solving patterns: the same ordering of object interactions and the same overall structure of the solution, with colour differences that do not influence correctness. These clusters, therefore, represent a single behavioural type. Cluster 2 also contains correct solutions following the same strategy, though with additional unnecessary

clicks; since these do not reflect misconceptions, Cluster 2 can be merged with this group.

Cluster B: Correct solution with a corrected mistake (Cluster 1).

The representative correct solution in cluster 1 demonstrates a sound strategy: an error is made but quickly identified and corrected. This behaviour is not observed in other clusters, so it forms a distinct type reflecting successful error detection and repair.

Cluster C: Correct solutions achieved after misunderstandings or late corrections (Clusters 4 and 7).

Cluster 4 begins with a partially correct interpretation, then shifts to a period of colouring all objects the same colour, followed by numerous corrections until the correct final state is reached. Cluster 7 starts with a good plan but includes a late-stage error that prompts a full reassessment, especially involving the background object. Both clusters show repeated adjustments and a need to rethink earlier steps, suggesting a similar problem-solving dynamic.

Clusters with incorrect (FALSE) solutions

Cluster D: Partial understanding with incomplete application of colours (Cluster 0).

Incorrect representatives in cluster 0 show partial task understanding: some constraints are applied correctly, but not all colours are used appropriately. This indicates that the solver grasped part of the task but not the full set of requirements.

Cluster E: Nearly correct solutions with one uncorrected error (Clusters 1 and 4).

Clusters 1 and 4 each include an incorrect representative showing only a single uncorrected mistake. The overall solving strategy is sound, but attention lapses or incomplete checking result in leaving one object incorrectly coloured. These solutions are behaviourally similar and reflect minor execution errors rather than conceptual misunderstandings.

Cluster F: Excessive corrections and reliance on white colour (Clusters 3 and 7).

Incorrect solutions in clusters 3 and 7 reveal a pattern of frequent colour changes and occasional reversion to the initial white colour (which is not an

allowed final state). Solvers demonstrate partial understanding but struggle to stabilise a correct solution.

Cluster G: Correct strategy undermined by late-stage regression (Cluster 5).

The representative incorrect solution in Cluster 5 follows a generally correct solving pattern until the end, where a correctly coloured region is overwritten with an incorrect colour. This indicates a loss of control or confusion late in the process. Such solutions are partially correct but deserve lower credit than those in Cluster E.

Cluster H: Fully incorrect solutions with no understanding of the rule (Cluster 6).

Cluster 6 consists solely of solutions in which all objects are coloured with the same colour. This suggests a fundamental misunderstanding of the task and should be assessed as fully incorrect.

Analysis of correct (TRUE) and incorrect (FALSE) clusters shows that the solution space is far richer than a simple binary "correct/incorrect" classification. Instead of two possible outcomes, the data reveal 8-10 distinct solution types, each reflecting different aspects of CT:

- Strategic planning;
- Systematic work;
- Error detection and correction;
- Partial understanding;
- Misconceptions;
- Random or uninformed behaviour.

This demonstrates that interactive task logs allow for much more precise assessment.
The reconstructed solutions reveal:

- When errors were made;
- Whether they were corrected;
- Whether unnecessary steps were purposeful or random;
- Whether the solver understood task constraints;
- Whether the solving process was efficient or exploratory.

These insights confirm that interactive problem-solving data contain rich behavioural structure that cannot be captured by a simple correct/incorrect outcome.

To ensure that the behavioural patterns uncovered using k-means were not dependent on the choice of a predefined number of clusters, an additional clustering experiment was performed using an alternative method. Unlike k-means, the Affinity Propagation algorithm does not require specifying the number of clusters in advance. This makes it suitable for validating whether the solution groupings arise naturally from the data. The following section presents the Affinity Propagation experiment and demonstrates that it produced a highly similar cluster structure, further reinforcing the stability of the identified behavioural patterns.

2.3.3 Additional Clustering with Affinity Propagation

To verify that the behavioural patterns identified with k-means were not dependent on specifying the number of clusters in advance, an additional clustering experiment was conducted using the Affinity Propagation (AP) algorithm (Frey & Dueck, 2007). Unlike k-means, which requires a predefined value of k AP automatically determines the number of clusters and selects actual data points as cluster centres (exemplars). This makes it suitable for validating whether solution types emerge naturally from the structure of the behavioural dataset.

The same normalised feature vectors used in the previous clustering (denoted as z_i, where i indexes a solution attempt) were supplied to the AP model. Affinity Propagation begins by computing pairwise similarities between every pair of solutions i and potential exemplars k. In this study, similarity was defined in a standard way as the negative squared Euclidean distance:

$$s(i,k) = -\| z_i - z_k \|^2,$$

where z_i and z_k are the normalised behavioural feature vectors of solutions i and k, and $s(i,k)$ represents their similarity (less negative values indicate more similar behaviour).

The algorithm then iteratively exchanges two types of messages. The responsibility $r(i,k)$ measures how suitable solution k is as an exemplar for solution i, compared to all other candidate exemplars:

$$r(i,k) = s(i,k) - \max_{k' \neq k} (a(i,k') + s(i,k')),$$

where $a(i,k')$ is the availability of other candidates k'. The availability $a(i,k)$ reflects how appropriate it is for i to choose k as an exemplar, considering support from all other solutions:

$$a(i,k) = \min\left(0, r(k,k) + \sum_{i' \notin \{i,k\}} \max\,(0, r(i',k))\right).$$

Here, $r(k,k)$expresses how strongly $k$supports itself as an exemplar, while the sum over i' accumulates support from other solutions. The outer min operation ensures availability remains non-positive, preventing artificially inflated exemplar scores.

To avoid oscillation during updates, both responsibility and availability values are stabilised using a damping factor λ(with $\lambda \in [0,1)$):

$$m_{\text{new}} = \lambda m_{\text{old}} + (1-\lambda) m_{\text{update}}.$$

Higher damping values (often between 0.7 and 0.95) are recommended for real behavioural datasets to ensure stable convergence. In this experiment, adjusting the damping factor was essential for obtaining interpretable results.

Once the iterative message passing converges, each solution i selects as its exemplar the solution k that maximises the total evidence:

$$k^*(i) = \arg\max_k\,(a(i,k) + r(i,k)),$$

where $a(i,k) + r(i,k)$ represents the combined global and local support that k should act as the exemplar for i. All solutions that select the same exemplar k^* form a cluster.

Using a damping value of 0.7, the model produced roughly 60 clusters, indicating oversensitivity to small behaviour differences and unstable message dynamics. After increasing the damping factor to 0.9, the algorithm stabilised and yielded nine clusters, a structure closely aligned with the eight clusters obtained earlier using k-means. A fragment of the resulting cluster assignments is shown in Figure 2.7.

	A	B	C	D	E	F	G	H	I	J
1	AssignedCluster	Distance_to_Centroid_0	Distance_to_Centroid_1	ance_to_Cent	e_to_C	e_to_C	e_to_C	e_to_C	e_to_C	Distance_to_Centroid_8
2	4	2,832900424	3,387475875	3,150048105	1,686239	1,110537	2,406145	2,303392	1,605316	1,388705423
3	8	3,002219405	3,343097117	3,03264015	1,797166	1,585566	2,102886	1,973114	1,743022	1,570991665
4	4	3,045516296	3,358682173	2,996189406	1,737632	1,495564	2,033543	2,018163	1,64507	1,609942916
5	5	2,997870955	2,741149599	2,573174929	1,800496	2,001083	1,107413	1,285208	1,759806	1,875288855
6	4	2,933505493	3,33435664	2,892079435	1,561984	1,355183	1,997137	1,988063	1,459317	1,420115608
7	6	2,812626337	2,553022254	2,894984127	2,048621	1,814503	1,604333	1,531964	2,022237	1,997054427
8	6	2,698322065	2,688795638	2,666327648	2,214121	2,208026	1,447955	1,0616	2,222682	2,00017194
9	6	2,853895397	2,712281262	2,63162009	2,001071	1,931661	1,179228	0,45017	2,001801	1,68796686
10	5	2,96834397	2,565034881	2,626186836	1,77849	1,911733	0,93599	1,118618	1,742323	1,899416695
11	6	3,037044318	2,557794994	2,652288319	2,013535	2,222616	0,887336	0,835393	2,013535	1,995977277
12	5	2,974079035	2,641598847	2,56407479	1,778759	1,931228	1,133011	1,309562	1,74092	1,859149319
13	6	2,769172309	2,942038446	2,749899105	1,914558	1,767975	1,701907	1,097493	1,911503	1,394768913
14	6	2,904380273	2,921005176	2,831153195	1,824382	1,574782	1,444123	1,140483	1,790921	1,562592818

Figure 2.7: Clustered data fragment with distances to exemplars (centroids)

The clusters produced by Affinity Propagation closely matched the identified solution types: systematic solving, error correction, redundant steps, late mistakes, partial understanding, and incorrect solutions. Because AP selects exemplars from real solutions and does not require a predefined number of clusters, it provides a robust and interpretable basis for modelling assessment categories.

2.4 Conclusions of the Chapter

This chapter presented the pilot study conducted to explore whether learners' interactive solution behaviour contains sufficient structure to support computational thinking assessment beyond a simple correctness score. The analysis showed that even within a single Bebras task, learner actions form distinct behavioural patterns that can be identified from click-based solution process features.

Clustering using the k-means algorithm revealed several types of solution behaviour, including systematic solutions, corrected errors, redundant steps, partial understanding, and incorrect attempts. To verify that these clusters were not dependent on a predefined number of groups, an additional experiment using the Affinity Propagation algorithm was conducted. This method produced similar results while automatically determining the number of clusters.

These results show that interactive task-solving behaviour has a stable structure and can be analysed using unsupervised learning, supporting a process-based assessment approach. The insights from the pilot study are used to develop the assessment model presented in the next chapter and to validate it using a broader set of Bebras tasks.

3. THEORETICAL MODEL OF THE CT ASSESSMENT

This chapter presents the theoretical foundation for an automated assessment model designed for interactive CT tasks. The development of the model builds directly on the pilot behavioural analysis presented in Chapter 2, which demonstrated that learners' solution processes show recurring and interpretable patterns. Based on these findings, the proposed model transforms action-level solution data into behavioural categories and assigns assessment values that reflect both correctness and the quality of the problem-solving process. This chapter is partially based on the results presented in Masiulionytė-Dagienė (2023) and Masiulionytė-Dagienė and Jevsikova (2025b).

The complete workflow of the model is summarised in Figure 3.1, and each part of the workflow is described in detail in Sections 3.1 – 3.4. The practical implementation of this theoretical pipeline is presented in Chapter 4.

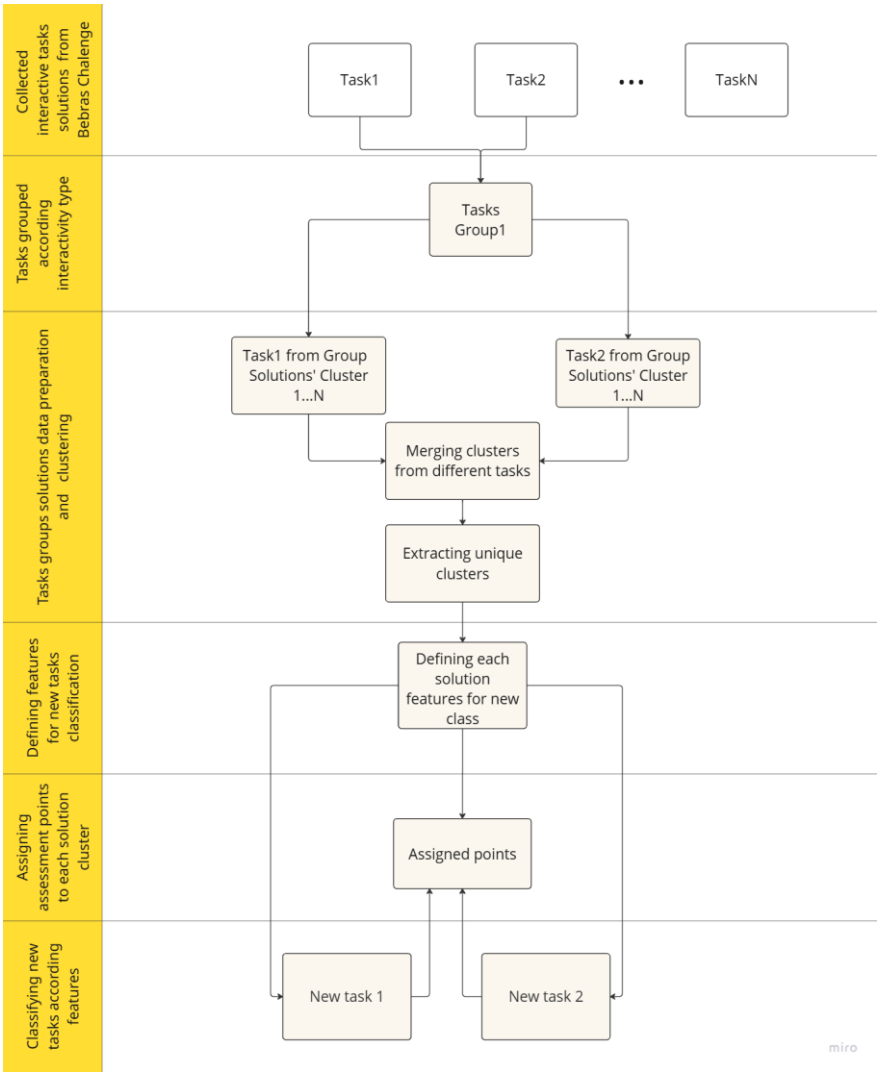


Figure 3.1: Theoretical automated CT assessment model

3.1 Conceptual Grouping of Interactive Tasks

Interactive CT tasks differ in how solvers interact with them, what types of actions are possible, and how errors can be corrected. To enable meaningful comparison and clustering of solution processes, tasks are conceptually grouped by their interactivity type.

This grouping relies on high-level characteristics, such as:

- Type of interaction (click, toggle, drag-and-drop);

- Number and structure of interactive objects;
- Whether actions accumulate or overwrite previous states;
- Whether partial correctness is detectable;
- Whether the task allows mid-solution corrections.

In this theoretical chapter, grouping is introduced only conceptually. A full operational grouping of interactive Bebras tasks, including all parameters and examples, is performed in Chapter 4, where the experimental methodology is presented.

3.2 Clustering Historical Solutions

After tasks are assigned to interactivity groups, historical solution logs are transformed into behavioural feature representations. These include:

- Number of actions;
- Object-level action counts;
- Time spent solving the task;
- Interactive tasks, objects status;
- Order of interactions.

Solutions are clustered together for correct and incorrect outcomes using Affinity Propagation, which automatically determines the number of behavioural classes and selects exemplar solutions.

Correct clusters typically identify:

- Fully correct, efficient solutions;
- Correct solutions requiring correction;
- Correct but redundant or exploratory behaviour.

Incorrect clusters typically reveal:

- One-mistake solutions;
- Partially correct reasoning;
- Multiple structural misunderstandings;
- Random or uninformed behaviour.

These clusters represent discrete behavioural solution classes applicable to tasks within a given interactivity group.

3.3 Classification of New Tasks and New Solutions

Before a new interactive task can be assessed automatically, it must be assigned to one of the predefined interactivity groups. These groups represent families of tasks that share similar interaction mechanics and therefore produce comparable behavioural patterns.

In the theoretical model presented here, each new task is matched to its appropriate interactivity group based on its overall interaction characteristics (for example, whether actions overwrite or accumulate, whether corrections are allowed, and what kind of object manipulations are possible).

The detailed criteria and the complete construction of all interactivity groups are presented in Chapter 5, where real Bebras tasks are analysed systematically. Once the task is assigned to a group, all behavioural classes, fingerprints, and assessment rules associated with that group become applicable to the task.

After the task is assigned to its interactivity group, the next step is to classify each new solution attempt. The system uses the behavioural classes derived from clustering historical solution data as predefined solution models. Each class represents a distinct way of solving the task type, such as:

- Fully correct and efficient reasoning;
- Correct with corrections;
- Partially correct reasoning;
- Systematic misunderstanding;
- Random or exploratory behaviour.

When a new solution attempt is recorded:

1. The interaction log is encoded into behavioural features using the same method as for historical data.
2. These features are compared with the behavioural fingerprints of all solution classes in the assigned interactivity group.
3. The solution is assigned to the class whose fingerprint it most closely resembles.

This allows the system to automatically recognise whether the learner has followed one of the known behavioural strategies.

3.4 Assessment Rules Derived from Behavioural Classes

The behavioural classes identified during the analysis of historical solutions form the basis for a refined scoring model for interactive CT tasks. Unlike the traditional Bebras scoring system, which distinguishes only between correct and incorrect solutions, this behaviour-based model acknowledges that interactive tasks reveal a detailed problem-solving process. These processes include error correction, partial understanding, systematic exploration, misunderstanding, or random behaviour. The goal of the assessment model is to reward deeper cognitive engagement and to distinguish between qualitatively different solving behaviours that would otherwise receive the same score under the traditional approach.

Each behavioural class derived during clustering is assigned a score in the interval [0, 1]. This score reflects both the correctness of the final answer and the quality of the solution process.

Fully correct solutions completed without any errors or corrections receive the maximum score:

$$p = 1.$$

Solutions that reach a correct result but involve one or more corrected errors receive a slightly lower score. Such behaviour demonstrates successful problem-solving but with intermediate inaccuracies. These solutions are assigned a value in the interval:

$$0.95 \leq p < 1.$$

The lower bound of 0.95 ensures that these scores align with traditional rounded competition scoring (for example, 9.5 rounding to 10). This distinction allows the assessment to differentiate between perfect and error-corrected solutions even when the final outcome is the same.

Solutions that end with an incorrect final result but show partially correct reasoning or meaningful progression toward the goal are scored within the range:

$$0 < p \leq 0.90.$$

Within this interval, scores must reflect the degree of partial correctness. If there are $k$distinct partially correct behavioural classes, then the points can be distributed systematically across this interval using the following scheme:

$$p_j = 0.9 \times \left(1 - \frac{j-1}{k}\right), \qquad j = 1, \ldots, k.$$

This ensures equal spacing between categories and guarantees that the most accurate partially correct behaviour receives the highest score within the partial-correctness band.

For example, when five partially correct behavioural types are identified, the resulting scores could be:

$$p \in \{\, 0.90,\ 0.72,\ 0.54,\ 0.36,\ 0.18 \,\}.$$

This progression captures decreasing levels of understanding and increasing numbers of errors in the problem-solving process.

Finally, solutions that demonstrate no understanding, such as colouring all objects the same colour, performing random actions, or ignoring the constraints of the task, are assigned:

$$p = 0.$$

These represent behaviour patterns where no meaningful progress toward the correct solution is observed. The listed behavioural examples listed illustrate possible solution classes identified within a given task type. Table 3.1 presents a generalised scoring scale in which several behavioural classes may be mapped to the same broader assessment category. The p values represent the proportion of correctly completed solution steps. Values of 0.95 and above indicate correct solutions, while values of 0.9 and below indicate incorrect solutions.

Table 3.1: Scale of task assessment

Behaviour type	**Score (p)**
Correct result without any corrections	$p = 1.0$
Correct final result with corrected errors	$0.95 \leq p < 1$
Incorrect final result, but part of the solution is correct	$0 < p \leq 0.90$
Totally incorrect solution	$p = 0$

This scoring model is directly connected to the behavioural clustering described in Chapter 3. Once behavioural classes are established, each new solution is assigned to one of these classes, and the corresponding score is applied automatically. Thus, the assessment process becomes transparent, reproducible, and aligned with the actual cognitive processes demonstrated by learners solving interactive CT tasks.

3.5 Conclusions of the Chapter

This chapter presented a theoretical model for automated assessment of interactive CT tasks. It defined how new tasks are first assigned to predefined interactivity groups, after which learners' solution behaviours are classified using behavioural classes derived from historical data. These behavioural fingerprints form the basis for a continuous scoring scale that evaluates not only the correctness of the final result but also the underlying problem-solving process. In this way, assessment shifts from binary outcomes toward a richer, process-based interpretation of CT performance.

The next chapter applies this model to real Bebras data, demonstrating how interactivity groups are constructed, how behavioural fingerprints are extracted, and how the assessment model operates in practice.

4. EXPERIMENTAL VALIDATION OF THE CT ASSESSMENT MODEL

This chapter transitions from the theoretical foundations presented earlier to their practical application. Using behavioural data collected from Bebras Challenge tasks, the proposed automated assessment model is examined under real conditions. The objective of this chapter is to demonstrate how the model operates when applied to empirical behavioural data and to explore what types of insights can be extracted from students' solution processes.

To clarify the overall structure of the empirical part of the dissertation and its relation to the research objectives, Figure 4.1 presents a general scheme of the experimental workflow.

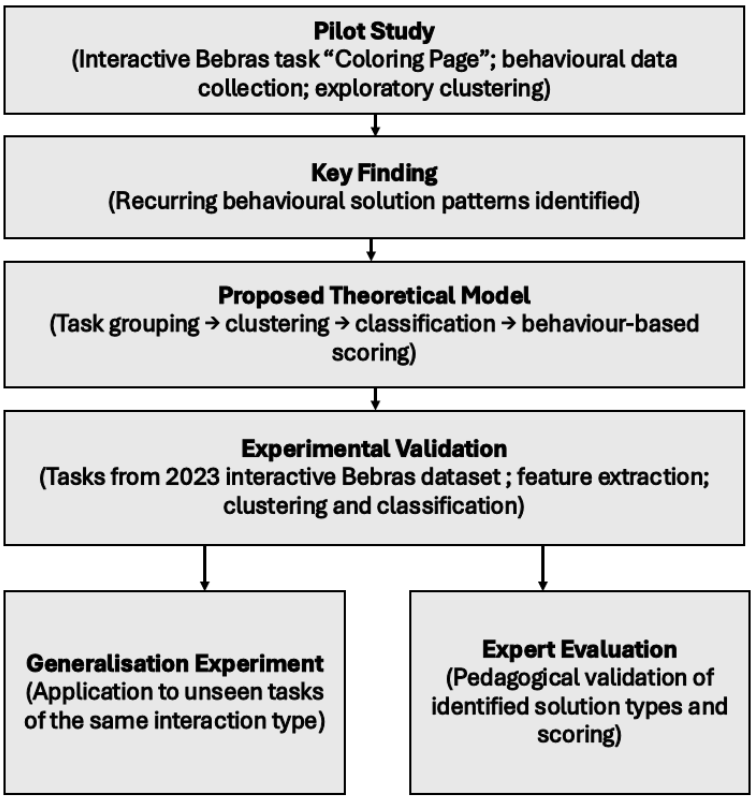


Figure 4.1: Experimental process steps

As shown in the scheme, the pilot study and the development of the theoretical model were presented in the previous chapters. This chapter

focuses on the subsequent stages, including the experimental validation of the model, its application to real behavioural data, and additional evaluation through generalisation experiments and expert assessment.

The chapter begins with an overview of the dataset and its preparation. It then proceeds to the analysis of interactive tasks, highlighting the properties relevant for further modelling.

4.1 Dataset Overview

Following the initial analysis of a single interactive task, all interactive tasks included in the 2023 Bebras Challenge in Lithuania were subjected to a systematic preparation process enabling detailed logging of student actions. Throughout the competition, extensive interaction data were collected, capturing complete action sequences for 29 distinct interactive Bebras tasks. All data used in this research were fully anonymised before analysis, ensuring that no participant could be identified from the stored records. Participation in the Bebras Challenge requires explicit consent: students or their legal guardians provide permission for the use of anonymised competition data for research and educational purposes. The consent form used in the competition is provided in Appendix 1.

A total of 974,864 solutions to the interacting Bebras tasks were obtained. As the Bebras Challenge was attended by students of grades 1 to 12, participation was not evenly distributed across the grades, and consequently, the number of solutions to some Bebras tasks was much higher than others. The minimum number of solutions per task was 4,340, while the maximum number of solutions per task was 88,226. There are also large differences because the same task is solved by several different grades. Figure 4.2 presents the diagram with the number of solutions for all tasks.

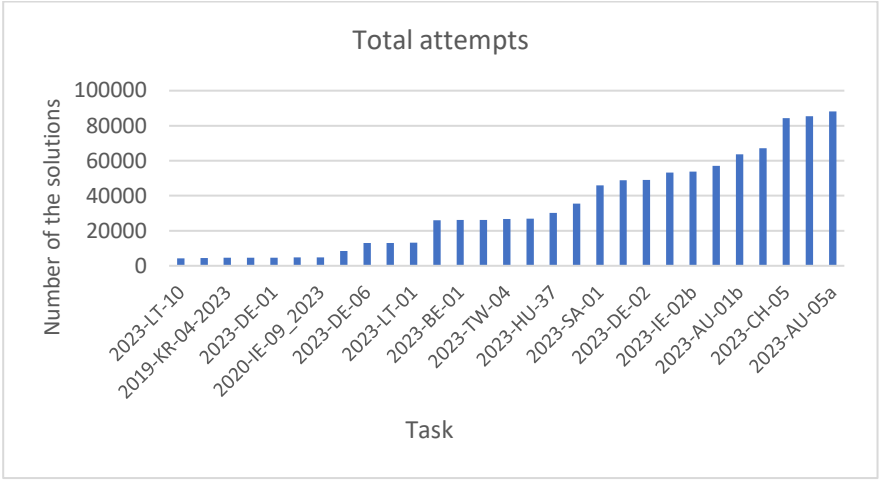


Figure 4.2: Number of solutions for each task

Table 4.1 presents a more detailed overview of the solution data for each interactive task. The table includes the following indicators: F Count – the number of solutions submitted with an incorrect final result; T Count – the number of solutions marked as correct; N Count – the number of tasks that were opened but not solved; Empty Count – solutions in which the task was skipped and no actions were recorded; Unique User Count – the number of distinct participants who attempted the task; and All Classes – the school grades in which the task was assigned during the Bebras Challenge.

Table 4.1: Bebras tasks solutions data

Task Id	F Count	T Count	N Count	Empty Count	Unique user Count	All Classes
2017-RU-05-2023	3,261	173	1,287	29	2,909	9;10;11;12
2019-KR-04-2023	2,995	553	1,093	22	2,906	1;9;10;11;12
2020-IE-09 2023	3,944	349	474	478	2,335	1;2;9
2023-AT-01	30,665	8,250	9,985	242	24,430	1;2;3;4;5;6;7;8;9;12
2023-AU-01b	48,924	6,426	8,323	394	30,637	5;6;7;8;9;12
2023-AU-05a	66,606	1,813	19,807	20,809	43,238	5;6;7;8;9;10;11;12
2023-BE-01	18,357	801	7,071	7,218	15,407	1;9;10;11;12
2023-BR-05	22,733	16,350	14,166	272	28,928	1;6;7;8;9;10;11;12
2023-CH-05	38,137	29,244	16,901	544	43,188	5;6;7;8;9;10;11;12
2023-CZ-01	40,214	4,016	12,821	13,016	29,011	1;6;7;8;9;10;11;12
2023-CZ-03	5,255	6,453	1,396	64	6,784	1;2;3;4;9
2023-DE-01	2,906	164	1,630	30	2,908	1;9;10;11;12
2023-DE-02	35,003	5,158	8,932	210	24,443	1;2;3;4;5;6;7;8;9;12
2023-DE-04	25,194	29,107	12,819	290	35,016	3;4;5;6;7;8;9;12
2023-DE-06	6,043	5,925	993	97	6,782	1;2;3;4;9
2023-DE-08	9,773	10,459	6,014	6,108	15,374	1;9;10;11;12
2023-DE-09	3,461	52	1,162	30	2,906	1;9;10;11;12
2023-HU-37	20,143	4,354	5,753	5,845	16,684	1;9;10;11;12
2023-IE-02b	36,029	6,838	10,886	289	28,957	1;6;7;8;9;10;11;12
2023-IE-05	1,608	6,255	685	46	4,503	3;4
2023-LT-01	8,372	3,919	870	98	6,785	1;2;3;4;9
2023-LT-08	2,648	1,479	383	29	2,323	1;2;9
2023-LT-10	1,285	2,603	452	15	2,334	1;2;9
2023-NZ-01	19,438	72	6,550	6,700	15,352	1;9;10;11;12
2023-SA-01	8,505	32,453	4,942	227	24,420	1;2;3;4;5;6;7;8;9;12
2023-SK-07	51,655	10,435	23,347	499	43,203	5;6;7;8;9;10;11;12
2023-TW-04	15,607	417	10,696	10,924	15,394	1;9;10;11;12
2023-UA-01	17,831	792	8,383	160	15,387	1;9;10;11;12
2023-US-03	24,407	4,347	6,787	34,692	17,514	3;4;6;7;8;9

The actions collected from the task solutions were stored in a single JSON file. The first step before the analysis was to extract the solutions for each task into separate JSON documents. Also, drawing on the expertise of the previous task analysis, the longest solution for each task was selected as an example. The longest solution is used to develop a data coding algorithm for each task group or unique task.

4.2 Classification of Interactive Tasks by Interactivity Type

Building on the dataset described in the previous section, this part of the chapter examines the interactive tasks themselves, focusing on the characteristics relevant for modelling and behavioural analysis. Since interactive tasks differ substantially in their mechanics, constraints, and action-logging structure, it is necessary to understand how these tasks operate before defining coding rules or extracting features for clustering. This chapter is partially based on the results presented Masiulionytė-Dagienė and Jevsikova (2025b).

To illustrate the diversity of interactive designs, Figure 4.3 and Figure 4.4 present examples of tasks for which detailed solution logs were collected. In each case, the original task interface is shown first, followed by a corresponding example of a recorded solution. These examples highlight how specific interactions, such as clicking, dragging, selecting, or switching object states, generate distinct behavioural patterns that later become essential for modelling and assessment.

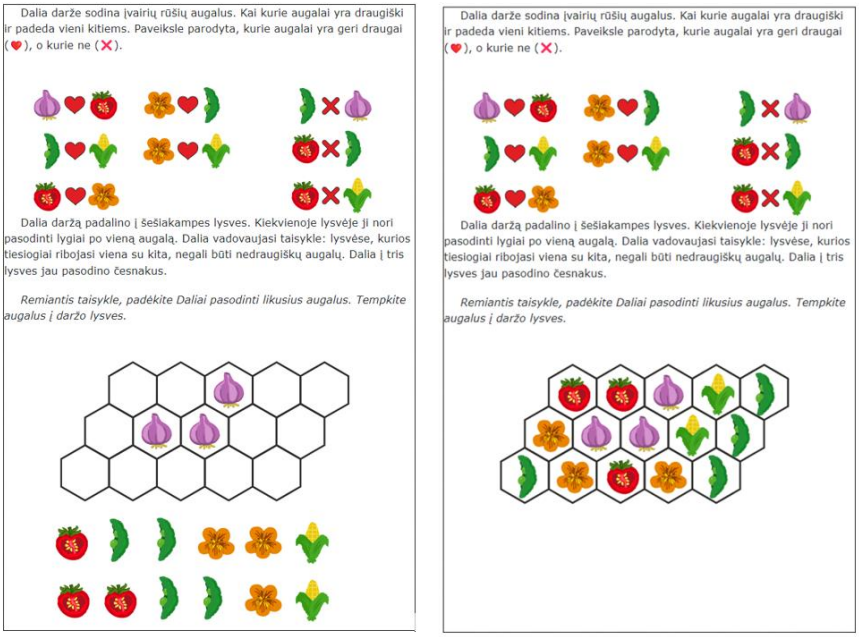


Figure 4.3: Image of the task 2023-AU-01b “Companion Planting” and its solution

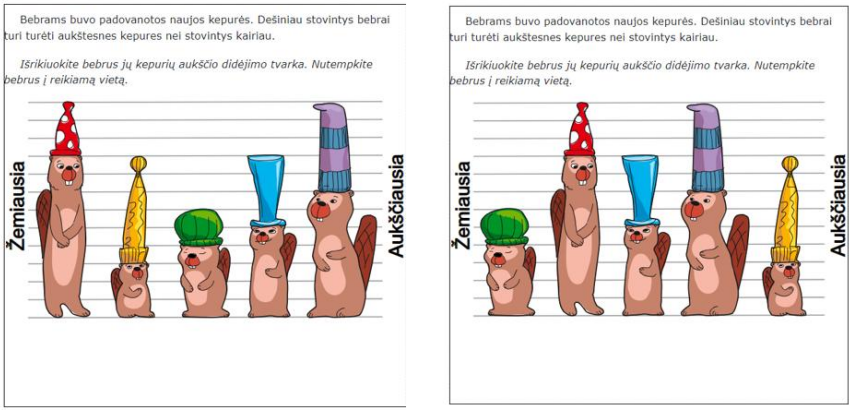


Figure 4.4: Image of the task 2023-LT-01 Sort the beavers by hats, and its solution

Each of the 29 interactive tasks was analysed on the following parameters:

- Interaction – what type of interactivity is used in the task, e.g., drag-and-drop, click-to-select, or some other special type of interactivity.

- Tracking – what tracking parameters were used. Used some patterns as for e.g. in drag-and-drop, or the tracked data is a special, unique solution.
- Can the solution be revised during task solving by the student? This means that is it possible for a student when he started to solve the task to change the first step, or the task is designed so that if you started the wrong way, there is no option to fix in the middle of the solution, and you have to restart the task.
- What is evaluated in the task? For example, is the final combination of the objects evaluated, or is it only some final result, the smallest number, or something like that? The main idea is that in the final evaluation, we have several objects to evaluate, only one number, or something in between.
- Are there any other ways to reach the solution? Is it possible to come to the same solution in different ways? For example, there is one main way, but you can also reach from the solution from the other side, or you can randomly reach it.
- Is there a partially correct answer? This means if you performed some correct actions and some wrong ones, how does it affect the solution? Is it totally wrong, or can it be considered partially correct?
- Mechanics: How technically the task interactivity works. What can be done with the task, and how many objects are involved?
- What could be encoded – what logically can be encoded from the tracked actions of the solution. E.g., the sequence of steps, the total number of actions, etc.

Figure 4.5 presents a snapshot of the interactive Bebras tasks analysis using the mentioned parameters:

Task ID	Title	Interaction	Tracking	Student can fix	What is evaluated?	Are there different ways to reach correct solution?	Is there partialy correct answe	Mechanics	What could be encoded
2017-RU-05-2023	Breaking the Cipher	Click-to-select	switches	Yes	Final combination	No	No	10 Separators, which, could be switched on or off. Sequence of clicking does not matter	clicking sequence. potentially, clicking on same separator could be aggregated if result stays the same, clickings could be removed if it's opposite then it could be aggregated
2019-KR-04-2023	Glass Slipper Buying Shoes	Open integer	special	No	Max number of steps you make to reach randomly hidden result (everytime different) - effectiveness	Yes, There is main strategy, but also factor of randomness	Yes, you can find object, but in too many steps. Correct, but not effective	49 boxes, which could be clicked in a sequence, one at a time	clicking sequence. Just problem, that final box is always different. Probably it would be different for each person
2020-IE-09_2023	Dubenėliai	Click-to-select	click (special)	No	Fastest way to reach one of possible combination - effectiveness	No	Yes, there are 4 possible combinations, but only 1 fastest	2 arrows could be clicked, one at a time	clicking sequence

Figure 4.5: Example of the interactive tasks analysis according to different parameters

The solutions to the interactive tasks were grouped according to the earlier listed parameters.

The tasks were first divided into two groups:

- Single attempt tasks (it means that you can't change the chosen solution algorithm in the middle).
- Trial-and-error tasks (it means that you can change all your steps in the solution without resetting).

After this step, the tasks were grouped according to other parameters, keeping in mind how the solutions could be grouped for clustering. In Figure 4.6, it is shown how single-attempt tasks were grouped:

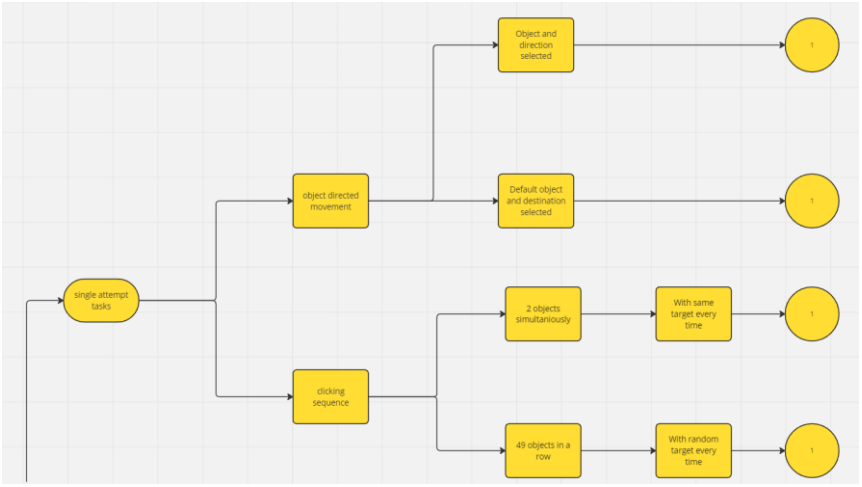


Figure 4.6: Single-attempt tasks grouping

In this grouping diagram, single-attempt tasks were grouped as follows:

- Object-directed movement – Mechanics parameter:
    - Object and direction selected (Mechanics parameter, more detailed) – 1 task in this group.
    - Default object and direction selected (Mechanics parameter, more detailed) – 1 task in this group.
- Clicking sequence – Encoding parameter:
    - Two objects simultaneously – The Mechanics parameter is more detailed.
        - With the same target every time (Encoding parameter more detailed) – 1 task in this group.
    - Forty-nine objects in a row – Mechanics parameter more detailed
        - With a random target every time (Encoding parameter more detailed) – 1 task in this group

In the Single attempt tasks group, all tasks' interactive realization was unique. Consequently, there is only one task at the end of the grouping tree.

The trial-and-error tasks were also divided into two groups:

- Drag-and-drop;
- Switches (object state changed).

In Figure 4.7, it is shown how the tasks from the drag-and-drop group were arranged:

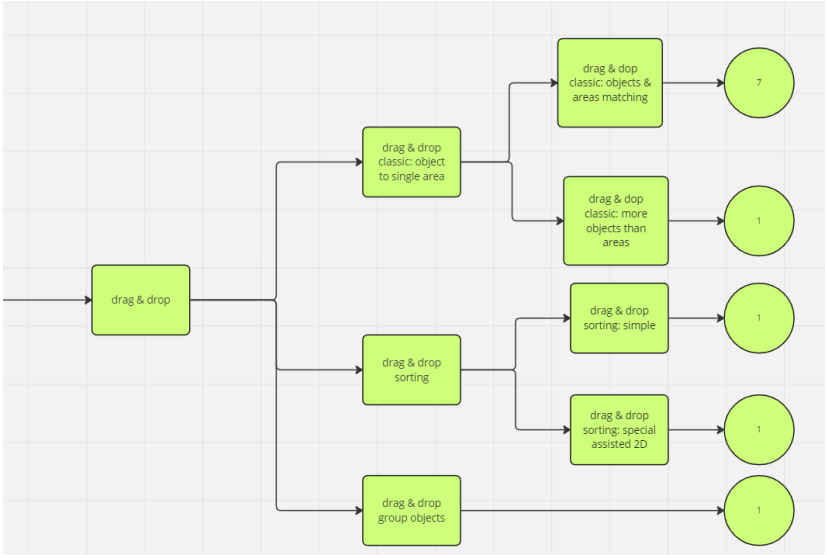


Figure 4.7: Grouping drag-and-drop-type tasks

The drag-and-drop tasks were divided into groups based on mechanics and encoding parameters, just like the previous group (single-attempt tasks). These tasks were not all unique, and as many as seven tasks were placed in one group. In addition to this group, there were four more groups with only one task in each group.

Figure 4.8 shows how the Switches type tasks were grouped:

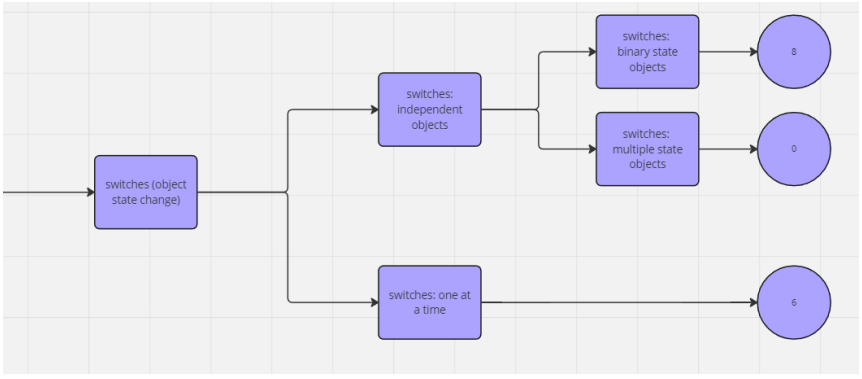


Figure 4.8: Grouping Switches type tasks

According to the mechanics and encoding parameters, the tasks in the switches group were grouped into smaller groups. In total, there are three groups, two of which contain eight and six tasks respectively, and one is empty.

The group is included because the pilot task would have been placed in this group.

A total of 12 groups were obtained. Only three of the groups have more than one task; the other groups are for the unique implementation of interactive tasks.

The data for each task must then be prepared for clustering. The actions performed must be coded accordingly. Tasks that are in the same cluster can use the same data preparation and coding algorithm, but tasks that are unique need a separate data preparation and coding algorithm.

4.3 Bebras Tasks' Solutions Extraction, Preparation, and Clustering

Based on the interactivity-based classification presented in Section 4.2, three major groups of interactive tasks were identified:

1. Drag-and-drop tasks involving object-area matching;
2. Switch-based tasks with binary-state objects;
3. Switch-based tasks where only one object may be changed at a time.

Among these groups, the binary-state switches group is particularly suitable for initial model implementation. Tasks in this category share structural similarities with the pilot task analysed in Chapter 2, including:

- A fixed set of objects;
- A limited number of possible states for each object;
- Repeated interaction with the same object during the solution process;
- A clearly defined, state-dependent correctness condition.

Because of these commonalities, the behavioural patterns in binary-state tasks can be encoded in a consistent manner, making the group an appropriate starting point for testing the generalisability and robustness of the assessment model. This group contains eight interactive tasks.

4.3.1 Preparation and Clustering of the "Breaking the Cipher" Task Solutions

The first task in this group was **"**Breaking the Cipher". There was a total of 4,721 attempts to solve this task in the original data. Of these, 3,261 were incorrect or generated an empty solution (29 cases), 173 were correct, and

1,287 were not solved. In the Bebras competition, points are awarded for correct solutions, a certain percentage of points is deducted for incorrect solutions, and zero points are awarded for unsolved tasks. Figure 4.9 and Figure 4.10 show the task before and after the solution.

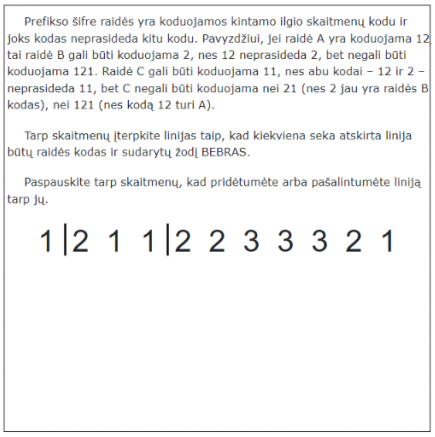


Figure 4.9: Task "Breaking the Cipher"

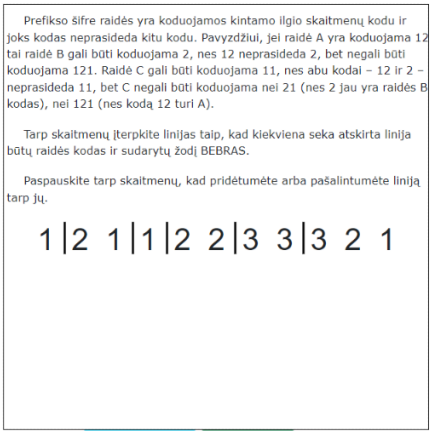


Figure 4.10: Solved task "Breaking the Cipher"

The initial solutions to the task were saved in a JSON file format. Figure 4.11 shows one example of the task solution.


```
{
  "user_id": 218676,
  "task_international_id": "2017-RU-05-2023",
  "answer_state": "[[{\"name\":\"task_text\",\"state\":{\"x\":4,\"y\":10,\"width\":500,\"height\":249,\"value\":\"\"}},{\"name\":\"b1\",\"state\":
{\"x\":52,\"y\":250,\"width\":40,\"height\":50,\"value\":1}},{\"name\":\"b2\",\"state\":{\"x\":92,\"y\":250,\"width\":40,\"height\":50,\"value\":0}},{\"name\":\"b3\",\"state\":
{\"x\":132,\"y\":250,\"width\":40,\"height\":50,\"value\":1}},{\"name\":\"b4\",\"state\":{\"x\":172,\"y\":250,\"width\":40,\"height\":50,\"value\":1}},{\"name\":\"b5\",\"state\":
{\"x\":212,\"y\":250,\"width\":40,\"height\":50,\"value\":0}},{\"name\":\"b6\",\"state\":{\"x\":252,\"y\":250,\"width\":40,\"height\":50,\"value\":1}},{\"name\":\"b7\",\"state\":
{\"x\":292,\"y\":250,\"width\":40,\"height\":50,\"value\":0}},{\"name\":\"b8\",\"state\":{\"x\":332,\"y\":250,\"width\":40,\"height\":50,\"value\":1}},{\"name\":\"b9\",\"state\":
{\"x\":372,\"y\":250,\"width\":40,\"height\":50,\"value\":0}},{\"name\":\"b10\",\"state\":{\"x\":412,\"y\":250,\"width\":40,\"height\":50,\"value\":0}},{\"name\":\"t0\",\"state\":
{\"x\":40,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t1\",\"state\":{\"x\":80,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t2\",\"state\":
{\"x\":120,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t3\",\"state\":{\"x\":160,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t4\",\"state\":
{\"x\":200,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t5\",\"state\":{\"x\":240,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t6\",\"state\":
{\"x\":280,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t7\",\"state\":{\"x\":320,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t8\",\"state\":
{\"x\":360,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t9\",\"state\":{\"x\":400,\"y\":250,\"height\":\"auto\",\"value\":null}},{\"name\":\"t10\",\"state\":
{\"x\":440,\"y\":250,\"height\":\"auto\",\"value\":null}}],[{\"event\":\"click\",\"switch\":\"b1\",\"value\":1,\"time\":1699858027104},
{\"event\":\"click\",\"switch\":\"b3\",\"value\":1,\"time\":1699858027983},{\"event\":\"click\",\"switch\":\"b4\",\"value\":1,\"time\":1699858032054},
{\"event\":\"click\",\"switch\":\"b6\",\"value\":1,\"time\":1699858034719},{\"event\":\"click\",\"switch\":\"b8\",\"value\":1,\"time\":1699858038184}]]",
  "start_time": "2023-11-13 08:46:05",
  "end_time": "2023-11-13 08:47:24",
  "solved_flag": "T",
  "user_class": 11
 },
```


Figure 4.11: Example of the task solution in JSON format

The solution to the task consists of the following parts:

- user_id: unique ID of the user who solved the task.
- task_international_id: unique task ID.

- answer_state: consists of the final answer and user actions during the solution.
- start_time and end_time: when the user starts and finishes solving the task.
- solved_flag: marks if the task was solved correctly, incorrectly, or hasn't been solved at all.
- user_class: marks the grade at which the user who solves the task is.

This task contains ten interactive objects (the gaps between the given numbers), each of which can take one of two possible states: with a dash or without a dash. During the solving process, participants may change the state of any object multiple times. The solution log records precisely these interactions that is, the sequence in which each object was modified and the total number of state changes applied to each one.

However, there were errors in the data, which made the data difficult to process. Errors in the data included the assignment of a solution to a completely different task from the task ID. This appears to be a bug in the problem-solving system and should be investigated and corrected in the future to fully automate the assessment. There have also been cases where a correctly solved task does not have a record of the user's actions, only the final status of the solution. There were cases where the part of a wrongly solved task (answer_state) was empty.

After preprocessing the data, excluding erroneous or empty cases (9 cases with incorrect solution JSON data and 28 cases with empty JSON data, 1,817 cases where user actions were not provided or the task was not solved), a total of 2,867 solutions remained for further analysis.

For each recorded solution, the following parameters were extracted or calculated from the raw data:

- Task solution duration (in seconds).
- Total number of clicks performed during the task.
- Final answer representation, indicating which objects remained selected in a binary format.
- Full solution click sequence, capturing the order of interactions.

Since the task involved 10 interactive objects, which could be toggled on or off in response to the task statement, additional object-specific parameters were recorded:

- Total number of clicks on each specific object.

- Binary sequence representing the exact order in which each object was clicked.

The dataset also included metadata related to task outcomes and participant details:

- Final solution correctness (TRUE/FALSE).
- Target grade level for which the task was designed.
- Solution ID for tracking unique responses.
- Participant ID, linking the solution to the individual who completed the task.

Figure 4.12 shows the fragment of the JSON file in which the task's data was stored:


```
{
    "user_id": 218676,
    "task_international_id": "2017-RU-05-2023",
    "answer_state": "[[{\"name\":\"task_text\",\"state\":{\"x\":4,\"y\":10,\"width\":500,\"height\":249,\"value\":\"\"}},
    "start_time": "2023-11-13 08:46:05",
    "end_time": "2023-11-13 08:47:24",
    "solved_flag": "T",
    "user_class": 11,
    "_solved_flag_dec": 1,
    "_iteration": 22,
    "_unique_guid": "0e98acd4-0257-4a54-803f-a1e2fe5ac130",
    "_duration_sec": 79.0,
    "_number_of_clicks": 5,
    "_number_of_restarts": 0,
    "_final_answer_bin": "1011010100",
    "_final_answer_dec": 724,
    "_final_answer_outofclicks_bin": "1011010100",
    "_final_answer_outofclicks_dec": 724,
    "_click_sequence_full": "b1;b3;b4;b6;b8",
    "_click_sequence_fullwrestarts": "b1;b3;b4;b6;b8",
    "_click_sequence_afterlastreatart": "b1;b3;b4;b6;b8",
    "_answer_jsonvalidation_state": "Y",
    "_answer_jsonvalidation_message": "Valid JSON",
    "_answer_datavalidation_state": "Y",
    "_answer_datavalidation_message": "Valid Data",
    "_b1_clickcount": 1,
    "_b1_clicksequence_bin": "10000",
    "_b1_clicksequence_dec": 16,
    "_b1_clicksequence_folded_bin": "001",
    "_b1_clicksequence_folded_dec": 1,
    "_b2_clickcount": 0,
    "_b2_clicksequence_bin": "00000",
    "_b2_clicksequence_dec": 0,
    "_b2_clicksequence_folded_bin": "000",
    "_b2_clicksequence_folded_dec": 0,
    "_b3_clickcount": 1,
    "_b3_clicksequence_bin": "01000",
    "_b3_clicksequence_dec": 8,
    "_b3_clicksequence_folded_bin": "010",
    "_b3_clicksequence_folded_dec": 2,
```


Figure 4.12: Fragment of already encoded solutions data stored in JSON file

The clustering analysis of the pilot task, "Coloring Page"revealed multiple solution clusters that, while identical in terms of the underlying problem-solving strategy, differ in variations in execution. Specifically, the primary distinction among clusters arose from the order in which participants initiated the colouring process. In some cases, the sequence began with the first object, whereas in others, it started from the last object.

A challenge emerged when encoding these sequences in binary format. If an object was clicked first or last, the resulting binary representations (e.g., 00000001 and 10000000) differed a lot when converted into decimal values.

Although these sequences represent functionally equivalent solutions, their numerical transformations yield vastly different values. This discrepancy poses a challenge for clustering algorithms that rely on numerical representations, as they may misinterpret structurally similar solutions as distinct due to variations in encoding.

In the context of the "Breaking the Cipher" task, an encoding method was developed to address the issue of positional bias in each task object click sequences. New approach was proposed to mitigate the effects of whether an object was clicked first or last in the sequence. The core idea behind this method is to fold the binary-encoded sequence of clicks in half and sum the overlapping bits together. This transformation ensures that structurally equivalent sequences yield the same encoded representation, regardless of the initial position of the clicks.

The issue in sequence-based task analysis is that binary representations of click sequences can produce vastly different numerical values when converted to decimal format, even when functionally identical. For example, consider two sequences: 10000000 and 00000001. Both sequences contain eight digits, but in standard binary encoding, their decimal representations would be 128 and 1, respectively, two different values. However, in terms of task execution, these sequences are equivalent, as they both indicate that a single object was clicked once, either at the beginning or end of the sequence. And in the “Breaking the Cipher” task, there were not only eight clicks on one object, but far more. The longest binary click sequences of one solution were 72 digits.

The proposed method works as follows:

Let a binary click sequence be denoted

$$b = (b_\ell, \dots, b_2, b_1), \qquad b_i \in \{0,1\}.$$

To remove positional dependence, the sequence is folded in half, and overlapping bits are added using standard binary addition.

If the sequence length ℓ is even, it is split into:

$$h_L = \left(b_\ell, \dots, b_{\frac{\ell}{2}+1}\right),\ h_R = \left(b_{\frac{\ell}{2}}, \dots, b_1\right).$$

The left half is reversed:

$$h_L^{\text{rev}} = \left(b_{\frac{\ell}{2}+1}, \dots, b_\ell \right).$$

The folded representation is obtained via element-wise binary addition:

$$f_i = h_{R,i} + h_{L,i}^{\text{rev}}, \qquad i = 1, \dots, \ell/2.$$

If $\ell = 2k + 1$ is odd, the right half receives the extra bit:

$$h_L = (b_\ell, \dots, b_{k+2}),\ h_R = (b_{k+1}, \dots, b_1).$$

The shorter half (left) is reversed:

$$h_L^{\text{rev}} = (b_{\text{k+2}}, \dots, b_\ell).$$

Because h_L^{rev}contains only k bits, it is right aligned by padding a zero on the left:

$$\tilde{h}_L = (0, h_L^{\text{rev}}).$$

The folded representation is obtained by element-wise binary addition:

$$f_i = h_{R,i} + \tilde{h}_{L,i}, \qquad i = 1, \dots, k + 1.$$

This yields a folded sequence of length $k + 1$, consistent with the structure of the original interaction pattern.

Example (sequence 1101010):

$$\tilde{h}_L = 0011,\ h_R = 1010,$$

$$1010 + 0011 = 1101.$$

As previously discussed, this encoding approach serves a dual purpose. Firstly, it ensures that sequences remain consistent regardless of whether an object is clicked at the beginning or the end of the solution sequence. Secondly, this method effectively compresses the numerical range when converting binary sequences into their decimal counterparts. By folding and summing the sequence, the disparity between the highest and lowest possible decimal values is reduced. This helps mitigate potential skewness in data distribution. Figure 4.13 shows an example of the unfolded binary and decimal encoding, and also of the folded binary and decimal encoding of one object of the task with different solutions:

_b1_clickcount	_b1_clicksequence_bin	_b1_clicksequence_dec	_b1_clicksequence_folded_bin	_b1_clicksequence_folded_dec
8	11000001111010001000000000000000000	1,04104E+11	0010001011110000011	71555
3	0001100001000000	6208	01011000	88
3	0001100001000000	6208	01011000	88
0	0000000000000000	0	000000000	0
2	0000001000000000001	8193	0001000001	65
2	1000000010000000	16512	10000001	129
0	0000000000000000	0	0000000000	0
2	1010000000000000	40960	00000101	5
4	10000100000001000000100000	69214240	10000001000001	8257
4	10000100000001000000100000	69214240	10000001000001	8257
2	0110000000000000000	98304	000000110	6
1	000000000001000	8	00001000	8
4	00011000001000010000	197664	10000111000	1080
4	1000101000010000	70688	001110001	113
4	1000101000010000	70688	001110001	113
4	00110000000000001001000000	25166976	00010010001100	1164
4	00110000000000001001000000	25166976	00010010001100	1164
2	1010000000000000000000	2621440	00000000101	5

Figure 4.13: Shows object b1 binary and decimal non-folded and binary and decimal folded click sequence examples

An alternative approach considered for reducing the complexity of click sequences while preserving essential task-solving patterns involved extracting a unique sequence representation for each task. Instead of storing every click event, including repetitions on the same object, this method retains only the first occurrence of each click in the order they appear. Subsequently, individual object click sequences can be encoded relative to this unique sequence, thereby reducing redundancy while maintaining task-specific interaction patterns.

For instance, the original sequence of interactions:

> Initial sequence: *b1; b1; b1; b1; b2; b3; b4; b5; b6; b6; b5; b4; b2; b3; b3; b1; b1; b2; b3; b3*
>
> By removing repeated clicks on the same object while preserving their first occurrence within the task sequence, the transformed representation becomes:
>
> Optimized unique sequence: *b1; b2; b3; b4; b5; b6; b5; b4; b2; b3; b1; b2; b3*
>
> From the given sequences, it can be observed how a specific object, e.g., b1, can be encoded using two different approaches: original sequence encoding, which preserves all repeated interactions, and unique sequence encoding, which removes redundant clicks while maintaining order.
>
> In the original sequence: *b1; b1; b1; b1; b2; b3; b4; b5; b6; b6; b5; b4; b2; b3; b3; b1; b1; b2; b3; b3*

The object b1 appears at multiple positions. In the binary representation, every occurrence of b1 is denoted as 1, while all other positions are represented as 0: 11110000000000011000.

In the unique sequence *b1; b2; b3; b4; b5; b6; b5; b4; b2; b3; b1; b2; b3* applying the same binary encoding principle, where b1 is represented by 1, and all other interactions by 0, the resulting sequence of b1 object is: 1000000000100.

In the process of clustering solutions for the "Breaking the Cipher" task, various combinations of encoded data representations were explored to assess their impact on clustering performance and task solutions pattern recognition.

To cluster the solutions of the "Breaking the Cipher" task, the Affinity Propagation (AP) clustering algorithm was selected. The main motivation for choosing this algorithm lies in its ability to determine the number of clusters dynamically, eliminating the need to specify this parameter beforehand. Given that the objective of the task is to identify different possible solution patterns, AP provides a suitable method for uncovering naturally emerging templates in the data.

This algorithm has already been successfully applied in clustering solutions of the pilot task. The results from that analysis demonstrated its capability to group solutions based on shared characteristics, confirming its suitability for other tasks clustering solutions.

Several iterations of the clustering process were conducted to fine-tune the parameters and achieve an optimal cluster distribution. After testing, the following parameter values were selected: Damping Factor (0.9) and Preference Value (-10).

These parameters were selected based on empirical evaluations to extract a meaningful and interpretable set of clusters from the solutions data.

Prior to applying clustering algorithms, the dataset underwent normalization using the min-max normalization technique (the same as for the pilot task) to ensure that all variables operated within a standardized range.

To explore different clustering configurations, multiple iterations were conducted with varying sets of parameters. However, a core dataset was consistently maintained across all iterations to ensure comparability. This core dataset included the following features for each solution attempt:

- Task solution duration: The total time taken to complete the task, measured in seconds.
- Total number of clicks: The cumulative count of clicks across all objects.
- Final task solution code: A binary representation indicating which objects were switched on at the end of the task. Converted to a decimal number.
- Total number of clicks per object: The frequency of clicks for each object.

To analyse the impact of different data representations on clustering outcomes, multiple configurations were tested by altering the way the sequence of clicks per object was encoded. The following variations were tried in different clustering iterations:

1. Non-unique and non-folded sequences: the complete sequence of clicks for each object was retained without modification.
2. Non-unique and folded sequences: the sequence of clicks for each object was folded using a predefined encoding strategy to reduce positional dependencies.
3. Unique and non-folded sequences: only the first occurrence of each click pattern was retained for each object, eliminating redundant clicks but maintaining their original sequence order.
4. Unique and folded sequences: Redundant clicks were removed, and the remaining sequence was further transformed using the folding method to reduce length and emphasize structural patterns.

Each of these clustering iterations was analysed to assess which representation provided the most meaningful and interpretable solution groups. The goal of this methodological variation was to determine whether modifications in sequence encoding influenced the clustering results and to identify the most suitable representation for distinguishing different problem-solving strategies.

Provided are the results of clustering iterations using different variants of encoded data for each object click sequence:

1. Unique click sequences (non-folded and folded):

- Regardless of whether non-folded or folded click sequences were used, the clustering process produced an identical set of nine clusters.

2. Non-Unique Folded Sequences:
    - When using folded sequences with non-unique click patterns, the clustering algorithm identified ten distinct clusters.
    - Some of these clusters are identical or very similar to unique click sequences' clusters.
3. Non-Unique Non-Folded Sequences:
    - In this configuration, ten clusters were also obtained. However, the distribution of solutions across these clusters was highly uneven.
    - A notable proportion of solutions were concentrated in just a few clusters, while some clusters contained only one or a very small number of solutions.
    - Further analysis of solutions within the same clusters revealed substantial variations in problem-solving approaches, indicating that this clustering configuration failed to effectively group similar strategies. Due to its inability to generate meaningful and interpretable clusters, this representation was deemed unsuitable and was discarded from further consideration.

In deciding between unique and non-unique sequences, non-unique sequences were ultimately chosen. The primary reason for this selection was that removing duplicate clicks to create unique sequences resulted in data loss. Since the logic of the task involved sequences of clicks, maintaining the full sequence, including repeated clicks, was considered more representative of participants' problem-solving strategies.

Beyond the initial clustering experiments, two additional iterations of clustering were conducted, this time distinguishing between correct (TRUE) and incorrect (FALSE) solutions. The objective was to determine whether clustering correct and incorrect solutions separately would yield meaningful insights into problem-solving strategies.

For these clustering iterations, the previously selected dataset configuration was utilized, where each object's non-unique folded click sequence was used for clustering. The clustering results:

- Correct Solutions: Formed three distinct clusters;
- Incorrect Solutions: Formed nine distinct clusters.

Upon closer examination, it was observed that some of the resulting clusters showed overlaps with the clusters obtained in previous experiments, where correct and incorrect solutions were not explicitly separated. However, an important observation was made: the number of clusters formed for correct solutions was three, and it was fewer than the number of clusters formed when correct (TRUE) and incorrect (FALSE) solutions were not separated (there were five clusters of correct solutions).

The decision was made not to separate correct and incorrect solutions into distinct clustering processes. The rationale behind this decision stemmed from an in-depth analysis of the solution logic present in the mixed (unseparated) clusters:

- When correct and incorrect solutions were clustered together, distinct patterns of solution strategies emerged, revealing variations in approaches taken by participants.
- In contrast, when correct solutions were clustered separately, the resulting clusters lacked some of the logical structures observed in the mixed clustering process.

Figure 4.13 shows the “Breaking the Cipher” task correct solution with marked places of each object:

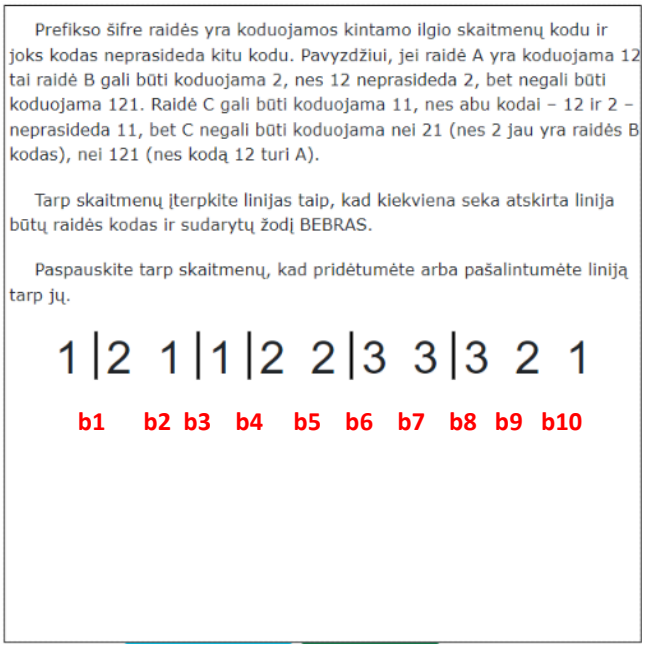


Figure 4.14: Correct solution of the task “Breaking the Cipher” with marked object IDs and places where to click

Following the clustering process using the selected datasets, the representative solutions within each cluster were identified. This approach was similar to the one used in the pilot task, "Coloring Page", where solutions closest to the cluster centroid were selected for further analysis. The rationale behind this selection was to determine the most prototypical solution within each cluster, which best represented the problem-solving patterns of that group.

For each cluster, both a correct (TRUE) and an incorrect (FALSE) solution closest to the centroid were identified whenever possible because not all clusters contained correct solutions. The results are presented in Figure 4.15.

Assigned Cluster	Distance to Centroid	Cluster size TRUE	Cluster size FALSE	Solve dflag	Iteration	Unique GUID	Duration (s)	Number of clicks	Final answer binary	Click sequence full
0	0.4175633380592804	8	0	T	976	a3954df5-4aaf-4171-aee7-a268090799e1	151.0	23	1011010100	b1*b1*b1*b2*b2*b1*b2*b3*b5*b6*b7*b9*b9*b3*b4*b5*b7*b8*b4*b2*b1*b3*b4
0	1.0536712127723509e-08	0	224	F	732	94d2f29c-1987-4e38-9173-7968bdcfc40f	108.0	19	101011010	b2*b1*b2*b1*b1*b3*b4*b5*b1*b2*b3*b3*b4*b4*b3*b5*b6*b7*b9
1	0.3960898503986166	22	0	T	642	b8bc060d-025f-40b9-8fa2-af4c8302722d	101.0	27	1011010100	b6*b7*b6*b7*b1*b3*b4*b4*b3*b1*b2*b5*b3*b3*b2*b5*b9*b6*b9*b8*b10*b1*b4*b4*b3*b4*b10
1	2.1073424255447017e-08	0	108	F	1602	c4224026-b6e1-438a-91fa-3fe5ebf2e626	172.0	31	1011101011	b2*b2*b2*b3*b3*b2*b1*b2*b4*b6*b6*b5*b7*b9*b1*b2*b4*b5*b7*b9*b10*b1*b3*b4*b6*b8*b8*b6*b5*b7*b9
2	0.0	0	2	F	2181	b444dfdf-447b-446d-b776-b41f0edd787f	4.0	72	0000000000	b1*b3*b3*b1*b2*b3*b5*b7*b7*b7*b7*b6*b3*b2*b1*b3*b5*b6*b3*b1*b2*b4*b4*b3*b3*b2*b1*b2*b3*b3*b2*b3*b1*b2*b5*b7*b8*b9*b8*b7*b5*b3*b4*b4*b2*b1*b3*b1*b2*b3*b3*b5*b5*b5*b9*b9*b3*b4*b5*b4*b2*b1*b3*b3*b1*b2*b2*b1*b3*b3*b1*b9
3	0.0	0	312	F	2487	87f08263-e437-4d24-8a88-d39b02d96824	50.0	6	1000010	b2*b2*b1*b1*b4*b9
4	6.452392069879462e-09	0	331	F	2542	dc08ced1-8d71-4cb4-9759-18ba852958a6	5.0	5	1111100	b4*b5*b6*b7*b8
5	1.0536712127723509e-08	0	293	F	3105	8e02d5d5-17bf-4ec2-87cd-785f34a855a9	109.0	11	101110111	b2*b4*b5*b7*b9*b3*b3*b10*b6*b7*b8
6	0.4466261914093038	4	0	T	3134	cad4d738-b7b4-4fba-b469-293ad4750594	258.0	41	1011010100	b1*b2*b1*b3*b5*b3*b4*b5*b2*b4*b1*b2*b3*b1*b3*b3*b3*b9*b7*b5*b3*b3*b1*b1*b4*b5*b3*b2*b3*b2*b4*b5*b3*b2*b1*b4*b5*b9*b8*b7*b6
6	0.0	0	59	F	3799	fa5332fc-3800-442d-a72f-c42b9740378	212.0	43	110101010	b1*b2*b4*b4*b3*b2*b2*b3*b2*b1*b1*b1*b3*b6*b9*b3*b6*b9*b2*b3*b5*b2*b3*b5*b1*b1*b1*b3*b4*b5*b5*b4*b3*b1*b1*b1*b2*b3*b5*b8*b9*b8*b7
7	0.0	0	372	F	3862	bd3f6ff0-64bf-4886-94ee-66dd26a1dd3b	50.0	10	110101110	b1*b2*b2*b1*b2*b5*b7*b8*b9*b3
8	2.1073424255447017e-08	59	0	T	4568	ad950506-bb2a-4d9e-bcb9-b079267821 0b	96.0	9	1011010100	b1*b1*b1*b2*b2*b3*b4*b6*b8
8	0.15971918303351573	0	214	F	88	c2bd2496-8d4c-4704-ad24-9cb020eba46e	13.0	10	1011000100	b3*b2*b1*b1*b3*b4*b3*b2*b8*b1
9	0.2955684097562174 5	26	0	T	4317	2e21c630-b40a-4b4e-91e0-c72a3e6c7924	132.0	13	1011010100	b2*b3*b5*b6*b6*b5*b3*b2*b1*b3*b4*b6*b8
9	1.0536712127723509e-08	0	234	F	4702	7db98877-85f4-4318-9198-cb52eb84915c	33.0	14	1101101111	b3*b2*b2*b3*b1*b2*b4*b6*b5*b6*b7*b8*b9*b10

Figure 4.15: Represented are the solutions correct (TRUE) and incorrect (FALSE), which are closest to the cluster exemplar (centroid) for each cluster

After a thorough examination of the solutions within each cluster, a systematic grouping approach was applied to categorize the solutions based on two primary factors: (1) the correctness of the selected objects, i.e., whether the key objects required for a correct solution were included, and (2) the sequence and timing of the interactions, which provided insights into whether the participant demonstrated a structured approach to problem-solving or engaged in random clicking behaviour.

Each cluster was assigned a label reflecting its defining characteristics, ensuring a clear differentiation between solution strategies. These labels were based on observed patterns in the data, such as solutions that were almost correct but contained additional, unnecessary clicks, solutions that missed key elements, and those that appeared to involve trial-and-error, rather than a logical progression toward a solution, but got the correct answer and solutions with the trial-and-error method, but the final answer was incorrect.

Table 4.2 presents a structured overview of how the reviewed clusters were categorized for assessment. The table includes the assigned cluster labels, ID of examples of representative solutions, and comments highlighting common errors and potential misconceptions.

Table 4.2: Categorized clusters with comments

Solution label (T – True (correct); F-False (incorrect))	Cluster number	Solution ID	Comment
T1	8	ad950506-bb2a-4d9e-bcb9-b0792678210b	Completely correct solution, or at the beginning, there are some extra clicks on the button.
T2	9	2e21c630-b40a-4b4e-91e0-c72a3e6c7924	The solution starts from the incorrect sequence, but then it is noticed and fixed, and the correct solution is marked.
T3	0	a3954df5-4aaf-4171-aee7-a268090799e1	Using the trial-and-error method but got the correct solution.
	1	b8bc060d-025f-40b9-8fa2-af4c8302722d	
T4	6	cad4d738-b7b4-4fba-b469-293ad4750594	Using the trial-and-error method but takes much longer and more clicks than in T3.
F1	8	c2bd2496-8d4c-4704-ad24-9cb020eba46e	Only one mistake is made.
F2	1	c4224026-b6e1-438a-91fa-3fe5ebf2e626	First half of the solution is correct
F3	0	94d2f29c-1987-4e38-9173-7968bdcfc40f	Almost half of the solution is correct, but not in the row, normal solution time, and there is some logic in sequence, few random clicks and logical solution timing.
	5	8e02d5d5-17bf-4ec2-87cd-785f34a855a9	
F4	3	87f08263-e437-4d24-8a88-d39b02d96824	A small part of the solution is correct (~3 objects), also saw some logic in sequence, and few random clicks and moderate solution timing.
	7	bd3f6ff0-64bf-4886-94ee-66dd26a1dd3b	
	9	7db98877-85f4-4318-9198-cb52eb84915c	
F5	2	b444dfdf-447b-446d-b776-b41f0edd787f	Incorrect solution, lots of random clicks, or, in a very short time, just a few random clicks.
	4	dc08ced1-8d71-4cb4-9759-18ba852958a6	
	6	fa5332fc-3800-442d-a72f-c42b974037f8	

The categorized clusters, along with their assigned labels, will serve as a foundational framework for the classification and assessment of solutions in future tasks. By utilizing these predefined clusters, new task solutions can be systematically analysed and mapped to existing solution patterns, allowing for an efficient and standardized assessment process.

4.3.2 Preparation and Clustering of the "A Day at the Zoo" Task Solutions

The next task selected for analysis from the same group (switches: binary state objects) of interactive tasks is titled "A Day at the Zoo". This task is designed to assess participants' ability to optimize their selection of activities based on a given timetable. The objective is to maximize the number of activities attended while considering time constraints, as certain activities have overlapping time slots.

The task requires participants to demonstrate strategic decision-making and time management skills by identifying an optimal schedule that minimizes conflicts and ensures the highest possible engagement in available activities. Figure 4.16 shows the task before and after the solution.

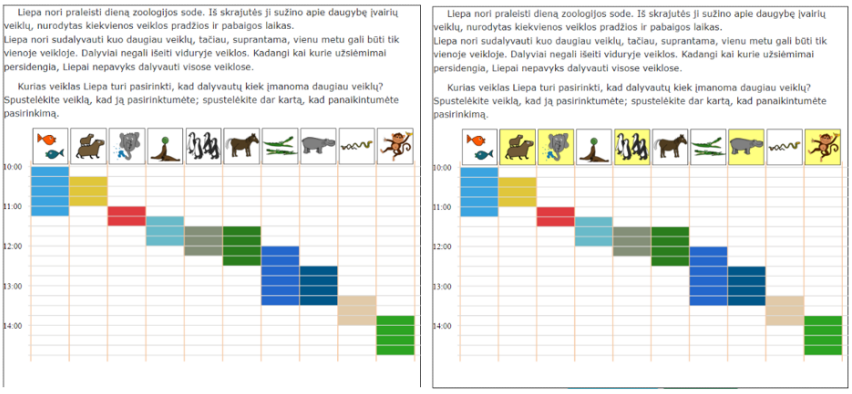


Figure 4.16: "Day at the Zoo" task and solved task

A total of 48,900 recorded attempts were made to solve this task. However, following a rigorous data validation process, the dataset was refined to 25,579 valid solutions suitable for clustering and further analysis. The task was included in the Bebras Challenge and was assigned to students across all grade levels, ranging from the first to the twelfth grade. The task comprises

ten interactive objects, each of which can be toggled on or off through user interaction. The solution process is recorded based on the sequence of clicks performed by the participant. Figure 4.17 presents a fragment of the dataset in JSON format, illustrating an example of an original recorded solution.


```
{
   "user_id": 445346,
   "task_international_id": "2023-AT-01",
   "answer_state": "[
   [
   {\"name\":\"txt3\",\"state\":{\"x\":1,\"y\":107,\"width\":600,\"height\":71,\"value\":\"\"}},
   {\"name\":\"img1\",\"state\":{\"x\":1,\"y\":183,\"width\":599,\"height\":340,\"value\":\"\"}},
   {\"name\":\"ats1\",\"state\":{\"x\":42,\"y\":177,\"width\":50,\"height\":50,\"value\":0}},
   {\"name\":\"ats2\",\"state\":{\"x\":97,\"y\":177,\"width\":50,\"height\":50,\"value\":1}},
   {\"name\":\"ats3\",\"state\":{\"x\":152,\"y\":177,\"width\":50,\"height\":50,\"value\":1}},
   {\"name\":\"ats4\",\"state\":{\"x\":207,\"y\":177,\"width\":50,\"height\":50,\"value\":0}},
   {\"name\":\"ats5\",\"state\":{\"x\":262,\"y\":177,\"width\":50,\"height\":50,\"value\":1}},
   {\"name\":\"ats6\",\"state\":{\"x\":317,\"y\":177,\"width\":50,\"height\":50,\"value\":0}},
   {\"name\":\"ats7\",\"state\":{\"x\":372,\"y\":177,\"width\":50,\"height\":50,\"value\":0}},
   {\"name\":\"ats8\",\"state\":{\"x\":427,\"y\":177,\"width\":50,\"height\":50,\"value\":1}},
   {\"name\":\"ats19\",\"state\":{\"x\":482,\"y\":177,\"width\":50,\"height\":50,\"value\":0}},
   {\"name\":\"ats10\",\"state\":{\"x\":537,\"y\":177,\"width\":50,\"height\":50,\"value\":1}}],
   [
   {\"event\":\"click\",\"switch\":\"ats5\",\"value\":1,\"time\":1700463277724},
   {\"event\":\"click\",\"switch\":\"ats5\",\"value\":0,\"time\":1700463329460},
   {\"event\":\"click\",\"switch\":\"ats2\",\"value\":1,\"time\":1700463343931},
   {\"event\":\"click\",\"switch\":\"ats3\",\"value\":1,\"time\":1700463346140},
   {\"event\":\"click\",\"switch\":\"ats10\",\"value\":1,\"time\":1700463357277},
   {\"event\":\"click\",\"switch\":\"ats5\",\"value\":1,\"time\":1700463367572},
   {\"event\":\"click\",\"switch\":\"ats5\",\"value\":0,\"time\":1700463367796},
   {\"event\":\"click\",\"switch\":\"ats5\",\"value\":1,\"time\":1700463368484},
   {\"event\":\"click\",\"switch\":\"ats8\",\"value\":1,\"time\":1700463372933},
   {\"event\":\"click\",\"switch\":\"ats8\",\"value\":0,\"time\":1700463373213},
   {\"event\":\"click\",\"switch\":\"ats8\",\"value\":1,\"time\":1700463374557}
   ]]",
   "start_time": "2023-11-20 08:53:07",
   "end_time": "2023-11-20 08:56:25",
   "solved_flag": "T",
   "user_class": 1
}
```


Figure 4.17: Original recorded task “Day at the Zoo” solution fragment in JSON format

For each recorded solution, the following parameters were extracted or calculated from the raw data:

- Task solution duration (in seconds);
- Total number of clicks performed during the task;
- Final answer representation, indicating which objects remained selected in a binary format;
- Full solution click sequence, capturing the order of interactions.

Since the task involved ten interactive objects, which could be toggled on or off in response to the task statement, additional object-specific parameters were recorded:

- Total number of clicks on each specific object;

- Binary sequence representing the exact order in which each object was clicked.

Metadata related to the task outcome and participant details were stored, including:

- Final solution correctness (TRUE/FALSE);
- Target grade level for which the task was designed;
- Solution ID for tracking unique responses;
- Participant ID, linking the solution to the individual who completed the task.

Figure 4.18 represents a JSON file of prepared and encoded data for clustering:


```
{
    "user_id": 445346,
    "task_international_id": "2023-AT-01",
    "answer_state": "[[{\"name\":\"txt3\",\"state\":{\"x\":1,\"y\":107,\"width\":600,\"height\":71,\"value\":\"\"}},
    "start_time": "2023-11-20 08:53:07",
    "end_time": "2023-11-20 08:56:25",
    "solved_flag": "T",
    "user_class": 1,
    "_solved_flag_dec": 1,
    "_iteration": 133,
    "_unique_guid": "65187529-cfe7-47e5-b69a-77afa9902b61",
    "_duration_sec": 198.0,
    "_number_of_clicks": 11,
    "_number_of_restarts": 0,
    "_final_answer_bin": "0110100101",
    "_final_answer_dec": 421,
    "_final_answer_outofclicks_bin": "0110100101",
    "_final_answer_outofclicks_dec": 421,
    "_click_sequence_full": "ats5;ats5;ats2;ats3;ats10;ats5;ats5;ats5;ats8;ats8;ats8",
    "_click_sequence_fullwrestarts": "ats5;ats5;ats2;ats3;ats10;ats5;ats5;ats5;ats8;ats8;ats8",
    "_click_sequence_afterlastreatart": "ats5;ats5;ats2;ats3;ats10;ats5;ats5;ats5;ats8;ats8;ats8",
    "_answer_jsonvalidation_state": "Y",
    "_answer_jsonvalidation_message": "Valid JSON",
    "_answer_datavalidation_state": "Y",
    "_answer_datavalidation_message": "Valid Data",
    "_ats1_clickcount": 0,
    "_ats1_clicksequence_bin": "00000000000",
    "_ats1_clicksequence_dec": 0,
    "_ats1_clicksequence_folded_bin": "000000",
    "_ats1_clicksequence_folded_dec": 0,
    "_ats2_clickcount": 1,
    "_ats2_clicksequence_bin": "00100000000",
    "_ats2_clicksequence_dec": 256,
    "_ats2_clicksequence_folded_bin": "000100",
    "_ats2_clicksequence_folded_dec": 4,
    "_ats3_clickcount": 1,
    "_ats3_clicksequence_bin": "00010000000",
    "_ats3_clicksequence_dec": 128,
    "_ats3_clicksequence_folded_bin": "001000",
    "_ats3_clicksequence_folded_dec": 8,
```


Figure 4.18: “Day at the Zoo” task solution, prepared and encoded data fragment

The Affinity Propagation clustering algorithm, which was previously utilized for clustering solutions in the "Breaking the Cipher" task, was also applied to this dataset. To adapt the clustering process to the characteristics of the new task, certain parameter adjustments were made. The damping factor was maintained at 0.9, consistent with the previous clustering experiments. However, the preference value was lowered to -35 to reduce the number of

clusters, thereby refining the grouping process and ensuring that only meaningful and distinct clusters were formed.

Building on the insights gained from the analysis of the previous task, a consistent encoding and dataset structure were applied to this task to ensure comparability in clustering results. The following data parameters were selected for clustering:

- Task solution duration: the total time taken to complete the task, measured in seconds.
- Total number of clicks: the overall number of interactions recorded across all objects.
- Final task solution code: a binary representation capturing which objects remained switched on at the end of the task, subsequently converted into a decimal value for clustering.
- Total number of clicks per object: the count of interactions for each specific object, reflecting individual engagement with different elements of the task.
- Non-unique and folded sequences: the click sequence for each object was processed using a predefined encoding technique that folded the sequence. This method aimed to reduce positional dependencies while preserving essential patterns in the solution process.

Figure 4.19 presents the task interface, emphasizing the titles of clickable objects. The task allows for two possible correct solutions, which are indicated in the figure. A solution is considered correct if either the object *ats5* or *ats6* is selected.

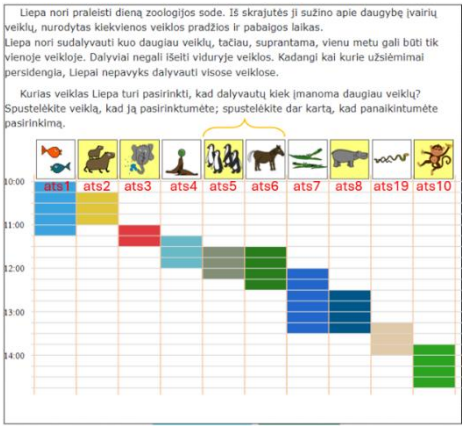


Figure 4.19: Represents objects of the task “Day at the Zoo” which could be clicked

The clustering process resulted in the identification of six distinct solution clusters. Like the previous task, for this analysis, the solutions closest to the centroid were extracted from each cluster, including both correct (TRUE) and incorrect (FALSE) solutions. Figure 4.20 presents the results obtained.

Assigned Cluster ID	Distance to Centroid	Cluster size TRUE	Cluster size FALSE	Solved flag	Iteration	Unique GUID	Duration sec	Number of clicks	Final answer binary	Click sequence full
0	0,132060958	931	0	T	46084	355e66ef-00bf-4b35-bdf3-9c72b31f06be	326	21	110010101	ats1*ats1*ats1*ats2*ats3*ats4*ats5*ats7*ats19*ats8*ats19*ats8*ats7*ats1*ats3*ats3*ats5*ats4*ats6*ats8*ats10
0	7,45E-09	0	2540	F	1660	647d9f25-8508-4c7d-80f2-287b901e6b99	232	21	1000000000	ats1*ats2*ats4*ats5*ats6*ats8*ats10*ats19*ats7*ats3*ats3*ats2*ats1*ats4*ats6*ats8*ats7*ats5*ats19*ats10*ats1
1	1,86E-09	0	4629	F	5427	7e0fd743-71d5-48a7-ace4-adc47e154957	83	3	1001001	ats4*ats7*ats10
2	0,078825019	3935	0	T	16696	28e57acd-7001-4672-8968-36a6bf94b729	140	7	110100101	ats1*ats1*ats2*ats3*ats5*ats8*ats10
2	0	0	8539	F	19159	8ee683ec-e4e2-43fe-96f8-78d3f82745ca	140	8	111100101	ats1*ats4*ats8*ats1*ats2*ats3*ats5*ats10
3	3,73E-09	0	979	F	25370	0e274a4e-d90a-453b-b47f-c0a00a9aa5c9	98	20	110000	ats5*ats6*ats5*ats5*ats4*ats3*ats2*ats1*ats7*ats19*ats10*ats8*ats7*ats8*ats10*ats19*ats4*ats3*ats2*ats1
4	1,49E-08	0	2947	F	29941	66d21d81-35b2-4a80-8545-a3ffdbd3ac2e	209	11	1101111111	ats1*ats2*ats3*ats3*ats4*ats5*ats6*ats7*ats8*ats19*ats10
5	7,45E-09	651	0	T	33592	a5d787e4-4833-48bb-9d97-223be3d78b05	125	39	110100101	ats2*ats2*ats1*ats2*ats3*ats4*ats5*ats6*ats7*ats8*ats19*ats10*ats1*ats2*ats3*ats4*ats5*ats5*ats6*ats7*ats8*ats19*ats10*ats1*ats4*ats5*ats7*ats10*ats10*ats7*ats3*ats4*ats3*ats1*ats2*ats3*ats5*ats8*ats10
5	0,108332901	0	428	F	45225	ba040206-914e-48dd-9ec5-dc05959cc620	109	35	110101001	ats5*ats5*ats3*ats4*ats5*ats6*ats7*ats8*ats19*ats10*ats2*ats1*ats1*ats2*ats3*ats5*ats6*ats7*ats8*ats19*ats10*ats4*ats4*ats3*ats3*ats4*ats1*ats1*ats3*ats5*ats7*ats7*ats7*ats10*ats2

Figure 4.20: Closest to exemplars (centroids), correct (TRUE) and incorrect (FALSE) solutions

As in the previous task, solutions within each cluster were systematically grouped based on two key factors: (1) the correctness of the selected objects and (2) the sequence and timing of interactions, indicating whether the approach was structured or involved random clicking.

Each cluster was assigned a descriptive comment reflecting its defining characteristics, such as nearly correct solutions with extra clicks, missing key elements, trial-and-error approaches that resulted in either correct or incorrect answers.

Table 4.3 provides an overview of the categorized clusters, including assigned labels, representative solution IDs, and comments on errors and misconceptions.

Table 4.3: Represents grouped clusters according to solution logic with comments

Solution label (T – True (correct); F-False (incorrect))	Cluster number	Solution ID	Comment
T1	2	28e57acd-7001-4672-8968-36a6bf94b729	Totally correct, or there are some extra clicks on the object at the beginning.
T2	0	355e66ef-00bf-4b35-bdf3-9c72b31f06be	Trial-and-error method but finally correct.
T3	5	a5d787e4-4833-48bb-9d97-223be3d78b05	Trial-and-error method, finally correct, but many more clicks.
F1	2	8ee683ec-e4e2-43fe-96f8-78d3f82745ca	One mistake, or alternative solution with maximum one less available choice.
F2	5	ba040206-914e-48dd-9ec5-dc05959cc620	Two mistakes.
F3	1	7e0fd743-71d5-48a7-ace4-adc47e154957	More than two mistakes, or an alternative partial solution.
F4	0	647d9f25-8508-4c7d-80f2-287b901e6b99	Incorrect solution, random clicks.
	3	0e274a4e-d90a-453b-b47f-c0a00a9aa5c9	
	4	66d21d81-35b2-4a80-8545-a3ffdbd3ac2e	

As in the previous task, the categorized clusters, along with their assigned labels, will serve as a foundational framework for the classification and assessment of solutions in future tasks.

4.3.3 Preparation and Clustering of the “Disagreement Detector” Task Solutions

The subsequent task selected for analysis from the same category (binary-state switchable objects) of interactive tasks is “Disagreement Detector”. The objective of this task is to manipulate switches on a series of interconnected

wires to achieve the correct logical outcome. The underlying principle of the task is based on the XOR logical operation, where the final state of the system depends on the parity of activated switches. Participants must strategically toggle the switches to produce the correct result, reinforcing their understanding of fundamental logical operations. Figure 4.21 provides a visual representation of the task interface along with its correct solution.

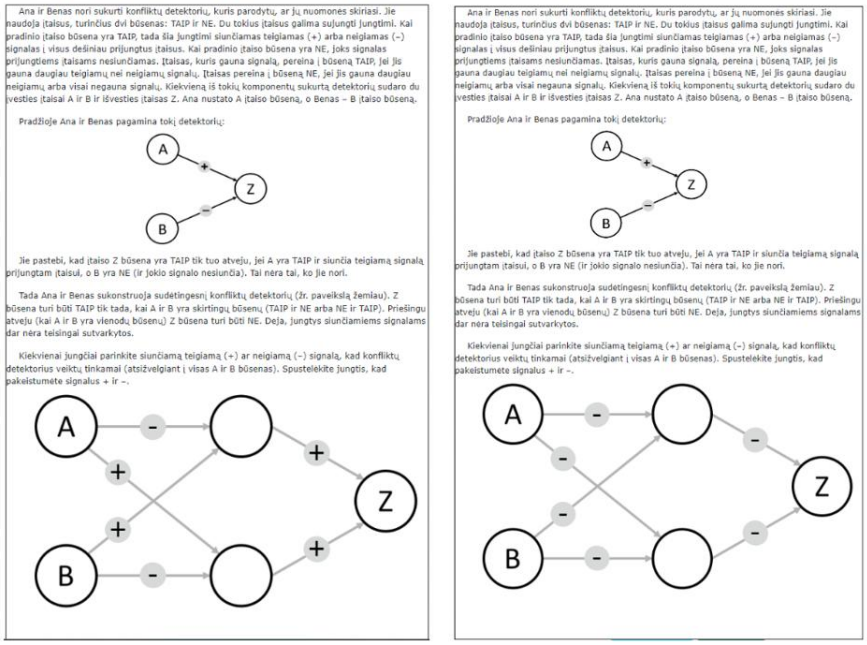


Figure 4.21: Representation of task “Disagreement Detector” and its solution

A total of 4,700 attempts were recorded for this task. However, after a thorough data validation process, the dataset was refined to 2,184 valid solutions deemed suitable for clustering and further analysis. This task, as part of the Bebras Challenge, was assigned to students in grades 9 through 12. The task consists of six interactive objects, each capable of being toggled on or off through user interaction. Figure 4.22 provides a JSON data fragment showcasing an example of an original recorded solution.

```
{
   "user_id": 309841,
   "task_international_id": "2023-DE-01",
   "answer_state": "[
   [
   {\"name\":\"main\",\"state\":{\"x\":20,\"y\":593,\"width\":\"640\",\"height\":\"360\",\"value\":\"\"}},
   {\"name\":\"s1\",\"state\":{\"x\":210,\"y\":638,\"value\":0}},
   {\"name\":\"s2\",\"state\":{\"x\":155,\"y\":708,\"value\":1}},
   {\"name\":\"s3\",\"state\":{\"x\":155,\"y\":798,\"value\":1}},
   {\"name\":\"s4\",\"state\":{\"x\":210,\"y\":868,\"value\":0}},
   {\"name\":\"s5\",\"state\":{\"x\":465,\"y\":678,\"value\":1}},
   {\"name\":\"s6\",\"state\":{\"x\":465,\"y\":828,\"value\":1}}
   ],[
   {\"event\":\"click\",\"switch\":\"s5\",\"value\":1,\"time\":1699868364600},
   {\"event\":\"click\",\"switch\":\"s4\",\"value\":1,\"time\":1699868366936},
   {\"event\":\"click\",\"switch\":\"s1\",\"value\":1,\"time\":1699868368095},
   {\"event\":\"click\",\"switch\":\"s2\",\"value\":1,\"time\":1699868371424},
   {\"event\":\"click\",\"switch\":\"s1\",\"value\":0,\"time\":1699868372591},
   {\"event\":\"click\",\"switch\":\"s4\",\"value\":0,\"time\":1699868386760},
   {\"event\":\"click\",\"switch\":\"s4\",\"value\":1,\"time\":1699868388088},
   {\"event\":\"click\",\"switch\":\"s2\",\"value\":0,\"time\":1699868388759},
   {\"event\":\"click\",\"switch\":\"s2\",\"value\":1,\"time\":1699868390824},
   {\"event\":\"click\",\"switch\":\"s4\",\"value\":0,\"time\":1699868391424},
   {\"event\":\"click\",\"switch\":\"s6\",\"value\":1,\"time\":1699868393169},
   {\"event\":\"click\",\"switch\":\"s6\",\"value\":0,\"time\":1699868393713},
   {\"event\":\"click\",\"switch\":\"s6\",\"value\":1,\"time\":1699868399344},
   {\"event\":\"click\",\"switch\":\"s3\",\"value\":1,\"time\":1699868441952}
   ]]",
   "start_time": "2023-11-13 11:39:13",
   "end_time": "2023-11-13 11:40:45",
   "solved_flag": "T",
   "user_class": 11
},
```


Figure 4.22: Task's "Disagreement Detector" fragment of raw data collected in JSON format

For each recorded solution, key parameters were extracted or derived from the raw data to facilitate clustering and analysis. These parameters include:

- Task solution duration (in seconds);
- Total number of clicks performed throughout the task;
- Final answer representation, capturing which objects remained selected in a binary format;
- Full solution click sequence, recording the precise order of interactions.

Since the task consisted of six interactive objects that could be toggled on or off in response to the task statement, additional object-specific data were collected:

- Total number of clicks per object, indicating the interaction frequency for each individual object.
- Binary click sequence per object, preserving the exact order in which each object was toggled.

Additionally, metadata related to the task outcome and participant information were recorded:

- Final solution correctness (TRUE/FALSE);

- Target grade level, specifying the intended student group for the task;
- Solution ID, serving as a unique identifier for each recorded attempt;
- Participant ID, linking the solution to the corresponding individual.

Figure 4.23 presents a JSON file containing the prepared and encoded dataset used for clustering.


```
{
    "user_id": 309841,
    "task_international_id": "2023-DE-01",
    "answer_state": "[[{\"name\":\"main\",\"state\":{\"x\":20,\"y\":593,\"width\":\"640\",\"height\":\"360\",\"value\":\"\"}},
    "start_time": "2023-11-13 11:39:13",
    "end_time": "2023-11-13 11:40:45",
    "solved_flag": "T",
    "user_class": 11,
    "_solved_flag_dec": 1,
    "_iteration": 153,
    "_unique_guid": "27852345-0cab-4ec1-94c2-bb4bb0d8efce",
    "_duration_sec": 92.0,
    "_number_of_clicks": 14,
    "_number_of_restarts": 0,
    "_final_answer_bin": "011011",
    "_final_answer_dec": 27,
    "_final_answer_outofclicks_bin": "011011",
    "_final_answer_outofclicks_dec": 27,
    "_click_sequence_full": "s5;s4;s1;s2;s1;s4;s4;s2;s2;s4;s6;s6;s6;s3",
    "_click_sequence_fullwrestarts": "s5;s4;s1;s2;s1;s4;s4;s2;s2;s4;s6;s6;s6;s3",
    "_click_sequence_afterlastreatart": "s5;s4;s1;s2;s1;s4;s4;s2;s2;s4;s6;s6;s6;s3",
    "_answer_jsonvalidation_state": "Y",
    "_answer_jsonvalidation_message": "Valid JSON",
    "_answer_datavalidation_state": "Y",
    "_answer_datavalidation_message": "Valid Data",
    "_s1_clickcount": 2,
    "_s1_clicksequence_bin": "00101000000000",
    "_s1_clicksequence_dec": 2560,
    "_s1_clicksequence_folded_bin": "0010100",
    "_s1_clicksequence_folded_dec": 20,
    "_s2_clickcount": 3,
    "_s2_clicksequence_bin": "00010001100000",
    "_s2_clicksequence_dec": 1120,
    "_s2_clicksequence_folded_bin": "1101000",
    "_s2_clicksequence_folded_dec": 104,
    "_s3_clickcount": 1,
    "_s3_clicksequence_bin": "00000000000001",
    "_s3_clicksequence_dec": 1,
    "_s3_clicksequence_folded_bin": "0000001",
    "_s3_clicksequence_folded_dec": 1,
```


Figure 4.23: Fragment of the task's "Disagreement Detector" prepared and encoded data

The Affinity Propagation clustering algorithm, previously employed for analysing solutions in other tasks, was also utilized for clustering the solutions of this task. To tailor the clustering process to the specific characteristics of this dataset, certain parameter adjustments were implemented. The damping factor was kept at 0.9, while the preference value was adjusted to −10. To maintain consistency and comparability with the previous clustering experiments, the same data encoding and dataset structure were applied to this task. The selected parameters for clustering included:

- Task solution duration;
- Total number of clicks;
- Final task solution code;
- Total number of clicks per object;
- Non-unique and folded sequences.

The IDs of the clickable objects are embedded in the task interface, presented in Figure 4.24. The same as in the previous task, this task has two

possible correct answers: either s1 or s2 could be switched; the same with s3 and s4.

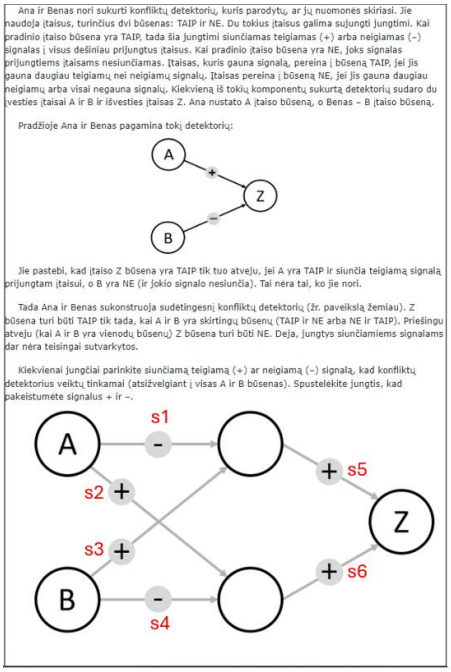


Figure 4.24: Represents a task with marked clickable objects' IDs

The clustering process identified six distinct solution clusters. As in the previous task, the solutions nearest to the centroid, both correct (TRUE) and incorrect (FALSE), were extracted for analysis. Figure 4.25 presents the results.

Assigned Cluster ID	Distance to Centroid	Cluster size T	Cluster size F	Solved flag	Iteration	Unique GUID	Duration sec	Number of clicks	Final answer binary	Click sequence full
0	0,137491133	22	0	T	1515	bd0bb229-c6a4-470d-8521-b552be201822	14	10	100111	s1*s2*s3*s3*s2*s1*s4*s5*s1*s6
0	1,05E-08	0	381	F	1574	99d0aa94-4e2a-4622-ac84-021f54f2e397	38	10	101101	s1*s1*s1*s2*s2*s3*s4*s6*s5*s5
1	0,157212278	44	0	T	1297	8cc73829-0c23-4eeb-a4a0-d8ea28fa045b	27	4	11011	s5*s6*s3*s2
1	5,27E-09	0	508	F	1838	cfc19c5c-3e42-4b37-b973-2a664414a923	36	4	10111	s2*s6*s5*s4
2	0	0	281	F	2109	3a2eef49-efb1-4795-b544-d2e4974fe01e	3	3	1	s1*s1*s6
3	0,223191423	14	0	T	1688	9e7e8c8e-a8d0-471f-9ba4-68780b80d865	46	16	100111	s2*s1*s5*s4*s1*s2*s5*s3*s4*s3*s2*s2*s1*s5*s6*s4
3	0	0	182	F	2119	90180834-f871-4836-947d-755335b9776a	139	18	11110	s1*s5*s6*s4*s3*s2*s1*s2*s3*s4*s6*s5*s1*s4*s2*s1*s3*s5
4	0,135772086	29	0	T	2914	cd7e5b90-decb-4436-a12f-9ea5670e277c	47	4	100111	s1*s4*s5*s6
4	1,49E-08	0	719	F	3242	8cef7ea2-c87f-43f6-b9fc-b642f7987fee	55	4	101101	s1*s4*s6*s3
5	1,186968495	1	0	T	265	2d55a889-6eb8-4c46-b132-d101fe01fd47	137	52	100111	s1*s3*s6*s5*s6*s6*s2*s2*s4*s1*s1*s3*s1*s2*s4*s3*s4*s2*s6*s5*s5*s6*s5*s1*s3*s6*s5*s6*s4*s6*s5*s6*s5*s3*s6*s5*s6*s5*s2*s6*s5*s6*s5*s1*s6*s5*s6*s2*s1*s3*s4*s5
5	0	0	3	F	3758	fecf9eff-0326-4287-8c75-fb772034bb03	134	55	101111	s6*s5*s5*s5*s1*s6*s6*s3*s2*s2*s1*s1*s1*s1*s4*s4*s2*s2*s1*s2*s3*s4*s5*s5*s5*s5*s6*s2*s5*s5*s3*s4*s2*s2*s1*s6*s5*s3*s4*s2*s1*s1*s3*s2*s3*s5*s1*s2*s5*s1*s2*s3*s2*s5*s2

Figure 4.25: Extracted solutions which are nearest to the cluster centre (correct (TRUE) and incorrect (FALSE))

Table 4.4 provides an overview of the categorized clusters, including assigned labels, representative solution IDs, and comments on errors and misconceptions.

Table 4.4: Represents categorized clusters and comments

Solution label (T – True (correct); F-False (incorrect))	Cluster number	Solution ID	Comment
T1	1	8cc73829-0c23-4eeb-a4a0-d8ea28fa045b	Correct solution from the first attempt, both variants.
	4	cd7e5b90-decb-4436-a12f-9ea5670e277c	
T2	0	bd0bb229-c6a4-470d-8521-b552be201822	Trial-and-error method.
	3	9e7e8c8e-a8d0-471f-9ba4-68780b80d865	
T3	5	2d55a889-6eb8-4c46-b132-d101fe01fd47	Trial-and-error method with lots of clicks and long timing.
F1	5	fecf9eff-0326-4287-8c75-fb772034bb03	One mistake but a lot of extra clicks.
F2	0	99d0aa94-4e2a-4622-ac84-021f54f2e397	Two mistakes and extra clicks.
	1	cfc19c5c-3e42-4b37-b973-2a664414a923	
	4	8cef7ea2-c87f-43f6-b9fc-b642f7987fee	
F3	3	90180834-f871-4836-947d-755335b9776a	Half correct with the one strategy at the beginning, then erased with a new solution provided.
F4	2	3a2eef49-efb1-4795-b544-d2e4974fe01e	Totally incorrect, a couple of random clicks with very short timing.

As in the previous tasks, the categorized clusters and labels will serve for the future solution classification and assessment.

4.3.4 Preparation and Clustering of the “Domino” Task Solutions

The next task selected for analysis from the same category (binary state objects, or "switches") of interactive challenges is titled “Domino”. This task is based on the classical domino tile game and requires participants to engage in pattern recognition and logical reasoning.
In this task, participants are presented with a set of eight domino tiles and are required to determine which tiles can be used at the beginning and end of a correctly ordered sequence.

A critical component of the challenge involves analysing constraints: participants must determine which tiles cannot be placed at the endpoints due to an inability to match with other tiles in a valid sequence. This step requires deductive reasoning, problem decomposition, and strategic decision-making. Figure 4.26 shows the task before and after the solution.

Figure 4.26: “Domino” task and solved task

A total of 4,675 recorded attempts were made to solve this task. However, after a data validation process, the dataset was refined to 2,126 valid solutions, which were deemed suitable for clustering and further analysis. The task is in the Bebras Challenge and was assigned to students in grades 11 and 12.

The task consists of eight interactive objects, each of which can be toggled on or off through user interaction. The solution process is recorded based on the sequence of clicks performed by the participant. Figure 4.27 presents a fragment of the dataset in JSON format, illustrating an example of an original recorded solution.

```
{
  "user_id": 383918,
  "task_international_id": "2023-DB-09",
  "answer_state": "[
  [
  {\"name\":\"t1\",\"state\":{\"x\":24,\"y\":380,\"width\":48,\"height\":48,\"value\":0}},
  {\"name\":\"t2\",\"state\":{\"x\":80,\"y\":380,\"width\":48,\"height\":48,\"value\":1}},
  {\"name\":\"t3\",\"state\":{\"x\":136,\"y\":380,\"width\":48,\"height\":48,\"value\":1}},
  {\"name\":\"t4\",\"state\":{\"x\":192,\"y\":380,\"width\":48,\"height\":48,\"value\":1}},
  {\"name\":\"t5\",\"state\":{\"x\":248,\"y\":380,\"width\":48,\"height\":48,\"value\":1}},
  {\"name\":\"t6\",\"state\":{\"x\":304,\"y\":380,\"width\":48,\"height\":48,\"value\":1}},
  {\"name\":\"t7\",\"state\":{\"x\":360,\"y\":380,\"width\":48,\"height\":48,\"value\":0}},
  {\"name\":\"t8\",\"state\":{\"x\":416,\"y\":380,\"width\":48,\"height\":48,\"value\":0}}
  ],[
  {\"event\":\"click\",\"switch\":\"t2\",\"value\":1,\"time\":1700572627359},
  {\"event\":\"click\",\"switch\":\"t2\",\"value\":0,\"time\":1700572627776},
  {\"event\":\"click\",\"switch\":\"t3\",\"value\":1,\"time\":1700572681405},
  {\"event\":\"click\",\"switch\":\"t5\",\"value\":1,\"time\":1700572691758},
  {\"event\":\"click\",\"switch\":\"t6\",\"value\":1,\"time\":1700572698590},
  {\"event\":\"click\",\"switch\":\"t4\",\"value\":1,\"time\":1700572699213},
  {\"event\":\"click\",\"switch\":\"t2\",\"value\":1,\"time\":1700572700228}
  ]]",
  "start_time": "2023-11-21 15:16:12",
  "end_time": "2023-11-21 15:18:21",
  "solved_flag": "T",
  "user_class": 11
},
```


Figure 4.27: Original recorded task “Domino” solution fragment in JSON format

For each recorded solution, several key parameters were extracted or derived from the raw data:

- Task completion time (in seconds).
- Total number of clicks performed during the task.
- Final answer representation, indicating which objects remained selected in a binary format.
- Full solution click sequence, capturing the exact order of interactions.

Since the task involved eight interactive objects, which could be toggled on or off in response to the task statement, additional object-specific parameters were recorded:

- Total number of clicks per object, tracking user interaction intensity.
- Binary sequence representing the exact order in which each object was clicked.

Furthermore, metadata related to the task outcome and participant details were stored, including:

- Final solution correctness (TRUE/FALSE);
- Student’s grade level (academic year);
- Solution ID, uniquely identifying each recorded response;

- Participant ID, linking the solution to the individual who completed the task.

Figure 4.*28* presents the JSON representation of the pre-processed and encoded dataset, prepared for clustering and further computational analysis.

Figure 4.28: “Domino” task solution, prepared and encoded data fragment

The Affinity Propagation clustering algorithm, which had previously been employed for clustering solutions in earlier tasks, was also applied to this dataset. Since this task contains fewer solutions compared to the “A Day at the Zoo” task, the main Affinity Propagation clustering parameters, damping factor, and preference were adjusted to align with tasks that had a similar number of solutions, such as “Disagreement Detector”. Specifically, the damping factor was set to 0.9, and the preference was set to -10, resulting in the formation of 11 clusters.

Building on insights gained from the analysis of previous tasks, a consistent encoding scheme and dataset structure were applied to ensure comparability in clustering results. The following data parameters were selected for clustering:

- Task solution duration;
- Total number of clicks;
- Final task solution code;
- Total number of clicks per object;
- Non-unique and folded sequences.

Figure 4.29 presents the task interface, labelling the titles of clickable objects. This task features a single correct solution, which is visually indicated in the figure.

Figure 4.29: Represents objects of the task “Domino” which could be clicked

The clustering process identified eleven distinct solution clusters. As in the previous tasks, the solutions nearest to the centroid, both correct (TRUE) and incorrect (FALSE), were extracted for analysis. Figure 4.30 presents the results.

Assigned Cluster ID	Distance to Centroid	Cluster size True	Cluster size False	Solved flag	Iteration	unique GUID	Duration sec	Number of clicks	Final answer binary	Click sequence full
0	1,49E-08	0	57	F	97	8a5c8057-c0b4-448a-a63c-bcbbd678aeda	148	51	10001001	t2*t4*t5*t3*t1*t7*t8*t6*t5*t4*t3*t2*t6*t8*t1*t7*t7*t7*t1*t1*t7*t7*t1*t2*t3*t4*t5*t6*t8*t7*t7*t8*t6*t5*t4*t3*t2*t1*t2*t1*t4*t5*t3*t1*3*t2*t1*t4*t5*t8*t1*t5
1	0	0	79	F	1468	d125a275-d002-424a-b36b-2a3dc1047a95	81	42	11	t5*t8*t6*t7*t4*t3*t2*t1*t6*t7*t7*t8*t6*t5*t4*t3*t1*t1*t1*t2*t2*t3*t3*t4*t5*t3*t8*t8*t5*t6*t3*t4*t2*t7*t7*t7*t7*t7*t8*t8*t8*t7
2	0,169390873	27	0	T	4273	d3f510af-abad-4115-80d5-bd62422ee092	115	9	1111100	t1*t1*t1*t1*t6*t5*t4*t3*t2
2	0	0	338	F	1682	851005ce-63ce-45c0-a788-e47c60697bff	70	11	10011101	t1*t1*t1*t8*t2*t3*t2*t3*t6*t5*t4
3	0,146479839	8	0	T	1129	83bc3545-b54d-4b88-870c-abb786e21cb5	130	21	1111100	t5*t5*t5*t6*t7*t8*t4*t3*t2*t1*t7*t7*t7*t6*t4*t2*t1*t8*t2*t4*t6
3	7,45E-09	0	158	F	2016	15a2041c-377b-4444-9ec2-91745e030565	76	21	1101110	t3*t4*t5*t7*t7*t7*t8*t6*t2*t1*t1*t2*t3*t4*t5*t6*t8*t3*t5*t2*t6
4	1,49E-08	0	175	F	2106	b4b2df4c-9f03-4058-9d37-2eb9ab40370d	82	28	11111111	t1*t2*t3*t5*t4*t4*t5*t6*t7*t7*t7*t8*t8*t6*t1*t3*t2*t7*t7*t7*t7*t1*t3*t2*t4*t5*t8*t6
5	0	0	410	F	2784	e3ffa79a-b7e2-4469-9618-32bd9ee9ef44	67	6	1100	t5*t1*t2*t1*t2*t6
6	0	0	1	F	2968	cc36999a-3adf-4ee1-8996-4464326933e9	151	98	10000010	t4*t4*t4*t4*t4*t2*t3*t1*t2*t1*t3*t4*t6*t6*t6*t5*t3*t4*t5*t7*t7*t5*t5*t4*t3*t6*t2*t2*t2*t2*t2*t2*t2*t3*t3*t3*t2*t1*t2*t2*t1*t7*t7*t7*t1*7*t8*t8*t6*t8*t4*t5*t4*t5*t7*t6*t7*t6*t8*t8*t6*t6*t5*t5*t4*t4*t3*t3*t2*t2*t1*t1*t7*t8*t8*t6*t6*t5*t5*t4*t4*t3*t3*t2*t2*t1*t1*t8*t6*t5*t2*t5*t2*t6*t7*t7*t7*t7*t8*t1
7	0	0	1	F	2969	909386b2-fe42-46b0-ab9a-51dfe73503d1	6	98	10000010	t4*t4*t4*t4*t4*t4*t2*t3*t1*t2*t1*t3*t4*t6*t6*t6*t5*t3*t4*t5*t7*t7*t5*t5*t4*t3*t6*t2*t2*t2*t2*t2*t2*t2*t3*t3*t2*t1*t2*t2*t1*t7*t7*t7*t1*7*t8*t8*t6*t6*t4*t5*t4*t5*t7*t6*t7*t6*t8*t8*t6*t6*t5*t5*t4*t4*t3*t3*t2*t2*t1*t1*t7*t8*t8*t6*t6*t5*t5*t4*t4*t3*t3*t2*t2*t1*t1*t8*t6*t5*t2*t5*t2*t6*t7*t7*t7*t7*t8*t1
8	3,73E-09	0	438	F	3799	45a61956-200d-4e31-9d87-c9e08c377a87	22	6	111100	t2*t2*t3*t4*t5*t6
9	7,45E-09	0	161	F	3851	e8d32ffe-70ef-459f-889a-d25596d9e509	100	24	10010	t1*t2*t2*t1*t7*t3*t3*t3*t4*t4*t5*t5*t6*t6*t8*t8*t1*t1*t2*t2*t7*t7*t4
10	0	0	273	F	4057	3c6ac7be-0dd9-4efc-a5c9-62133a14b8f9	13	12	11111111	t2*t2*t1*t3*t5*t6*t7*t8*t7*t7*t4*t2

Figure 4.30: Presents the extracted clustered solutions, nearest to the exemplar (centroid) centre

Table 4.5 outlines the categorized clusters, detailing their assigned labels, example solution IDs, and key observations on solution strategies.

Table 4.5: Presented categorized task solution clusters with comments on solution strategies

Solution label (T – True (correct); F – False (incorrect))	Cluster number	Solution ID	Comment
T1	2	d3f510af-abad-4115-80d5-bd62422ee092	Totally correct.
T2	3	83bc3545-b54d-4b88-870c-abb786e21cb5	Trial-and-error method, but finally correct, not so many clicks.
F1	8	45a61956-200d-4e31-9d87-c9e08c377a87	One mistake.
F2	4	b4b2df4c-9f03-4058-9d37-2eb9ab40370d	Three mistakes, and extra clicks.
	5	e3ffa79a-b7e2-4469-9618-32bd9ee9ef44	
F3	2	851005ce-63ce-45c0-a788-e47c60697bff	Half solution correct, not so many extra clicks.
	3	15a2041c-377b-4444-9ec2-91745e030565	
F4	9	e8d32ffe-70ef-459f-889a-d25596d9e509	Three objects correct, not so many extra clicks.
F5	0	8a5c8057-c0b4-448a-a63c-bcbbd678aeda	Only two objects correct, long timing, and lots of extra clicks.

F6	1	d125a275-d002-424a-b36b-2a3dc1047a95	Incorrect, very short timing with random clicks, or very long timing with a lot of random clicks.
	6	cc36999a-3adf-4ee1-8996-4464326933e9	
	7	909386b2-fe42-46b0-ab9a-51dfe73503d1	
	10	3c6ac7be-0dd9-4efc-a5c9-62133a14b8f9	

As with the previous tasks, the categorized clusters and their assigned labels will be utilized as a reference framework for future classification and assessment of interactive Bebras task solutions.

4.4 Bebras Task Solution Classification

The purpose of this classification step is to evaluate how well the behavioural solution types, previously identified through clustering, can be recognised automatically from encoded process data. By assigning unified labels to all solutions, the manually derived cluster structure becomes a reference dataset that enables supervised learning. This allows us to verify whether the discovered solution patterns are stable and distinguishable enough to be classified algorithmically, thus demonstrating the practical applicability of the proposed assessment model.

To support the classification of problem-solving behaviour, solution clusters across four distinct tasks were analysed and manually merged to form a unified labelling scheme. These newly assigned labels capture key qualitative patterns within the participants' solutions, ranging from fully correct to entirely incorrect approaches. Table 4.6 presents the mapping between these new unified labels and the original task-specific clusters from four different problem-solving tasks ("Breaking the Cipher", “Disagreement Detector”, “Domino”, and “A Day at the Zoo”). Each row describes a new label (e.g., 101, 102, …, 207), its interpretive description, and how specific clusters from each task were assigned to it. For example, label 101 represents “totally correct” solutions, possibly with a few extraneous initial clicks, while label 207 represents entirely incorrect attempts.

Table 4.6: Merged solutions with new labelling

New label for classification	Comment	Current clustering " Breaking the Cipher" (RU_01)	Original clusters " Breaking the Cipher" (RU_01)	Current clustering " Disagreement-Detector" (DE_01)	Original clusters " Disagreement-Detector" (DE_01)	Current clustering " Domino" (DE_09)	Original clusters " Domino" (DE_09)	Current clustering " A day at the zoo" (AT_01)	Original clusters " A day at the zoo" (AT_01)
101	Totally correct, or there are some extra clicks on the object at the beginning.	T1	8	T1	1;4	T1	2	T1	2
102	The solution starts from the incorrect sequence, but then it is noticed and fixed, and the resulting solution is correct.	T2	9	_	_	_	_	_	_
103	Using the trial-and-error method but got the correct solution.	T3	0;1	T2	0;3	T2	3	T2	0
104	Using the trial-and-error method but takes much longer and more clicks.	T4	6	T3	5	_	_	T3	5

New label for classification	Comment	Current clustering “ Breaking the Cipher” (RU_01)	Original clusters “ Breaking the Cipher” (RU_01)	Current clustering “ Disagreement-Detector” (DE_01)	Original clusters “ Disagreement-Detector” (DE_01)	Current clustering “ Domino” (DE_09)	Original clusters “ Domino” (DE_09)	Current clustering “ A day at the zoo” (AT_01)	Original clusters “ A day at the zoo” (AT_01)
201	Only one mistake is made.	F1	8	F1	5	F1	8	F1	2
202	Two mistakes.	–	–	F2	0;1; 4	–	–	F2	5
203	More than two mistakes, or not a full solution.	–	–	–	–	F2	4;5	F3	1
204	Half of the solution is correct.	F2	1	F3	3	F3	2;3	–	–
205	Almost half of the solution is correct.	F3	0;5	–	–	F4	9	–	–
206	A small part of the solution is correct.	F4	3;7;9	–	–	F5	0	–	–
207	Incorrect solution.	F5	2;4;6	F4	2	F6	1;6; 7;10	F4	0;3; 4

The objective of this mapping is to harmonize varied behavioural patterns across tasks into a consistent labelling framework for further classification. This unified labelling allows for the training of machine learning models that generalize across tasks.

Once the unified labelling scheme had been established, the next methodological step was to prepare the previous tasks’ solutions data for machine learning classification model training. The transformations applied ensured that all data from the four different interactive “Bebras” tasks were expressed in a consistent, structured, and numerical format, while also

preserving both the problem-solving process and the static characteristics of the solution outcome.

From the complete solution logs of the previously analysed interactive tasks (used for clustering), only the features relevant for classification were retained. While clustering relied on a richer set of features, including object-level interaction data specific to each task, classification required a unified representation based on task-level features. This adjustment was necessary because different tasks contain varying numbers of objects, and the classification model is designed to operate across multiple tasks. The retained features therefore capture general properties of the solution process that are consistent across tasks. These included:

- Click sequence – the exact ordered list of objects clicked during the problem-solving attempt, currently recorded as a semicolon-separated string (e.g., "*s5;s4;s1;s2;s1;...").*
- Solution duration – the total time taken from the first click to the completion of the attempt, measured in seconds.
- Number of clicks – the total count of click actions in the attempt, regardless of whether they were correct or incorrect.
- Final answer binary – a binary string representing the final state of the solution (e.g., "101011"), where each position corresponds to an object in the interactive task and the value indicates if the object is marked ("1") or left unmarked ("0").
- Unified solution label – the label assigned in all tasks' solutions mapping process (e.g., 101, 104, 201).
- Unique attempt identifier (GUID) – a persistent reference allowing the classified output to be linked back to the original attempt for validation or further analysis.

All other metadata fields, timestamps, and intermediate validation states were removed to reduce noise and avoid task-specific dependencies.

The next step involved the normalization of object identifiers across all tasks, ensuring that the click sequence data could be meaningfully compared and processed in a task-independent manner. Click sequences across the four tasks used different naming conventions (e.g., *"t1", "ats3", "b5"*) tied to the task context. To allow a single classification model to process data from all tasks, these identifiers were mapped to a generic naming scheme (*obj1, obj2,*

obj3, ...). This transformation preserved the relative identity of objects within a task but eliminated task-specific symbolic differences that would otherwise bias the model.

The normalized click sequences were tokenized using a vocabulary index, converting each object into an integer representation (e.g., *"obj1 obj2 obj3 obj1" → [1, 2, 3, 1]*). Since recurrent neural architectures are specifically designed to model temporal dependencies, the Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) network was selected for processing the sequential click-stream data. LSTM is well-established as a robust method for capturing long-range dependencies and order-sensitive patterns in behavioural or event-based sequences (Yu et al., 2019; Ullah et al., 2020). To enable batch processing, all sequences were padded with a special zero-token up to a fixed maximum length, ensuring consistent input dimensionality required by LSTM-based models.

The maximum length was determined empirically by measuring the longest observed click sequence across all training files and then increasing it slightly to account for potentially longer sequences in test data (in the current situation, the length is 100). Padding was applied at the end of sequences.

This step preserved the ordering of actions, allowing the model to learn sequential dependencies between task solution steps.

The final answer binary field was converted from a string representation (e.g., "*101011*") to a numerical binary vector ([*1, 0, 1, 0, 1, 1*]). All binary vectors were standardized to the same length by padding with zeros.

For each solution, three numeric elements were combined into a single static feature vector: solution duration, number of clicks, binary vector of the final answer.

Solution duration and number of clicks values were normalized using z-score standardization (subtracting the mean and dividing by the standard deviation) to ensure that differences in scale did not bias the model toward features with larger numeric ranges.

Although the GUID was not used as an input feature, it was retained in the processed dataset as a metadata column. This allowed post-classification results to be mapped back to the original solution records for traceability and error analysis.

After the unified labelling and feature preparation steps, the next step was classification of the solution attempts. In this setting, the classification task can be viewed as learning a mapping from the interaction sequence and static solution features to a unified solution-type label, where the LSTM branch is used to model sequential dependencies in click sequences, while the dense branch processes non-sequential static features. The purpose of this step was to train a model capable of automatically predicting the solution type label for each attempt by simultaneously leveraging two complementary sources of information:

1. Process-oriented sequential data – the click sequence of actions taken during task solving.
2. Static outcome features – the duration, the number of clicks, and the binary representation of the final state.

To integrate both aspects, a dual-branch neural network architecture was implemented. Figure 4.31: Task solutions classification scheme represents the architecture scheme.

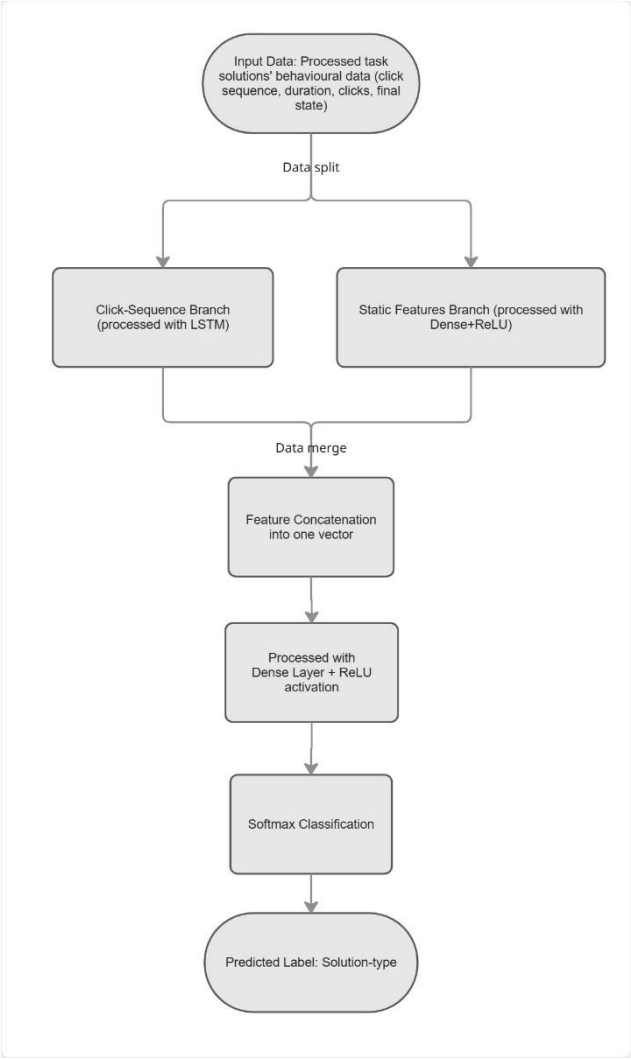


Figure 4.31: Task solutions classification scheme

Sequential branch

Each token in converted click sequences was mapped to a dense vector through an embedding layer. The embedded sequences were processed by a Long Short-Term Memory (LSTM) layer with 64 hidden units. LSTM networks were chosen because they are specifically designed to capture temporal dependencies in sequences, retaining relevant information from earlier steps while discarding irrelevant parts. In this context, the order of clicks is essential, since identical sets of clicks executed in different orders often correspond to different solution strategies and outcomes. To make this

explicit, the LSTM cell computes its hidden representation using the following standard equations (Hochreiter & Schmidhuber, 1997):

Forget gate:

$$f_t = \sigma\big(W_f x_t + U_f h_{t-1} + b_f\big).$$

Input gate:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i).$$

Candidate memory update:

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c).$$

Memory state:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t.$$

Output gate and hidden state:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), h_t = o_t \odot \tanh(c_t).$$

Where:

x_t – input token at step *t* (embedded click);
h_t – hidden state output of the LSTM;
c_t – internal memory state;
σ – sigmoid activation;
W, U, b – trainable parameters.
These equations allow the model to remember or forget parts of the click sequence, making them suitable for modelling interaction traces.

Static branch

The non-sequential features – solution duration (seconds, z-score normalized), number of clicks (z-score normalized), and the binary vector representing the final answer state after they were concatenated into a single fixed-length vector were passed through a fully connected Dense layer with 64 units and ReLU activation (Nair & Hinton, 2010). A Dense layer applies the transformation:

$$h = \text{ReLU}(Wx + b).$$

Where:

x – input feature vector (duration, clicks, final binary state);

W – weight matrix;
b – bias vector;
ReLU is defined as.

$$\mathrm{ReLU}(z) = \max(0, z).$$

Merging and classification. The outputs of the sequential and static branches were concatenated to form a joint representation of each attempt. The merging step produces a combined vector:

$$z = \left[h^{(seq)}; h^{(static)}\right].$$

This merged vector was further processed by a fully connected Dense layer with 64 units and ReLU activation, which integrated sequential and static information. Finally, the output was passed to a SoftMax classification layer, producing a probability distribution across all unified solution labels:

$$\hat{y}_k = \frac{\exp\,(z_k)}{\sum_{j=1}^{K} \exp\,(z_j)}.$$

Where:

K – number of unified solution classes;
$\hat{y}_k$ – predicted probability of class k.
Training was carried out with the Adam optimizer (Kingma & Ba, 2014) and sparse categorical cross-entropy loss. The data were split into 70% training and 30% test.

In this study, a 70/30 hold-out validation strategy was applied for model evaluation. While k-fold cross-validation is frequently discussed in machine learning literature, some methodological sources classify the hold-out method as a non-exhaustive form of cross-validation, since it involves partitioning the dataset into separate training and testing subsets (Seraj et al., 2023). In the present research, this approach was considered sufficient given the dataset size and the presence of strongly imbalanced classes, where repeated partitioning could lead to unstable representation of rare solution types.

The distribution of solution labels in the dataset was not uniform. Some solution types (e.g., dominant correct or frequent incorrect strategies) appeared in large numbers (the biggest class had 2,707 solutions in test and 6,398 in training from the incorrect (FALSE) solution group), while other solution types occurred only rarely (the smallest class had 224 solutions in test and 471 in training). This imbalance created a risk that the classifier would

achieve high global accuracy by focusing on the majority classes, while systematically neglecting the minority classes.

To address this, two variants of model training were performed:

- Unweighted model. In this baseline setup, all training examples contributed equally to the loss function. This variant reflects the "natural" class distribution and maximizes overall accuracy. It is useful as a benchmark, since it shows how well the model performs without any correction for imbalance.

- Weighted model. In this setup, each class was assigned a weight inversely proportional to its frequency in the training set:

$$w_k = \frac{1}{n_k}.$$

  Where n_k is the number of training samples in class k. As a result, misclassifying a rare solution type incurred a larger penalty than misclassifying a common one. This encouraged the network to learn to recognize underrepresented solution types, even though they appear less often.

The performance of each variant was first analysed through confusion matrices (Table 4.7, Table 4.8, Table 4.9, Table 4.10). Rows represent the true labels of solution attempts, while columns represent the predicted labels. The diagonal entries correspond to correctly classified attempts, and off-diagonal values indicate misclassifications.

Table 4.7: Confusion Matrix – Incorrect Solutions
(Unweighted, Accuracy = 85.85%)

True \ Predicted	201	202	203	204	205	206	207
201	2,569	29	18	5	3	13	70
202	0	532	20	6	2	0	44
203	3	27	1,478	8	2	4	74
204	4	23	36	143	4	6	33
205	24	15	4	20	104	27	30
206	55	3	35	16	31	97	65
207	115	51	140	36	13	6	1,870

Table 4.8: Confusion Matrix – Correct Solutions
(Unweighted, Accuracy = 92.97%)

True \ Predicted	101	102	103	104
101	1,217	1	17	0
102	7	0	2	0
103	36	0	234	30
104	0	0	29	162

Table 4.9: Confusion Matrix – Incorrect Solutions
(Weighted, Accuracy = 83.38%)

True \ Predicted	201	202	203	204	205	206	207
201	2,409	35	15	30	56	101	61
202	0	562	10	10	2	0	20
203	5	53	1,360	90	11	51	26
204	0	15	5	209	9	5	6
205	3	0	1	16	189	9	6
206	10	0	14	10	40	217	11
207	92	96	102	96	47	146	1,652

Table 4.10: Confusion Matrix – Correct Solutions
(Weighted, Accuracy = 90.61%)

True \ Prediction	101	102	103	104
101	1,175	23	37	0
102	0	9	0	0
103	30	2	216	52
104	0	0	19	172

From the confusion matrices, performance metrics (Sathyanarayanan, S., & Tantri, 2024) were computed using the following definitions:

$$Precision = \frac{TP}{TP + FP}.$$

$$Recall = \frac{TP}{TP + FN}.$$

$$F1 = 2 \cdot \frac{Precission \cdot Recall}{Precission + Recall}.$$

$$Accuracy = \frac{\sum TP}{\sum (TP + FP + FN + TN)}.$$

Where *TP, TN, FP, and FN* denote the true positives, true negatives, false positives, and false negatives. These definitions are computed separately for each class in the confusion matrix. The following tables present the detailed evaluation of classification results for both incorrect (FALSE) and correct (TRUE) solutions under unweighted and weighted training conditions.

Table 4.11: Results of Incorrect solutions unweighted
(overall accuracy 85.85%)

Class	Precision	Recall	F1	Support
201	0.927	0.949	0.938	2,707
202	0.782	0.881	0.829	604
203	0.854	0.926	0.888	1,596
204	0.611	0.574	0.592	249
205	0.654	0.464	0.543	224
206	0.634	0.321	0.426	302
207	0.855	0.838	0.847	2,231

The model performs strongly on the majority classes (201, 203, 207). Minority classes (205, 206) show lower recall, indicating that rare incorrect solution types are more difficult to detect under unweighted training.

Table 4.12: Results of Correct solutions unweighted
(overall accuracy 92.97%)

Class	Precision	Recall	F1	Support
101	0.966	0.985	0.976	1,235
102	0.000	0.000	0.000	9
103	0.830	0.780	0.804	300
104	0.844	0.848	0.846	191

High accuracy is achieved on classes 101, 103, and 104, with precision and recall above 0.80. However, the rare class 102 is not recognized at all, reflecting the challenge of training with very few examples in an unbalanced dataset.

Table 4.13: Results of Incorrect solutions weighted
(overall accuracy 83.38%)

Class	Precision	Recall	F1	Support
201	0.957	0.890	0.922	2,707
202	0.737	0.931	0.823	604
203	0.902	0.852	0.876	1,596
204	0.450	0.871	0.592	249
205	0.531	0.844	0.652	224
206	0.406	0.728	0.522	302
207	0.929	0.747	0.829	2,231

Applying class weights improves recall in underrepresented classes (205, 206), but this comes at the cost of reduced precision and overall accuracy. This demonstrates the expected trade-off: better balance across classes but slightly weaker global performance.

Table 4.14: Results of Correct solutions weighted
(overall accuracy 90.61%)

Class	Precision	Recall	F1	Support
101	0.975	0.951	0.963	1,235
102	0.265	1.000	0.419	9
103	0.794	0.720	0.755	300
104	0.768	0.900	0.829	191

Weighting fully recovers the rare class 102, achieving a recall of 1.0, though with limited precision (0.265). While overall accuracy decreases compared to the unweighted case, the weighted model provides a more balanced view across all classes, which is particularly valuable for diagnostic applications where rare solution types of matter.

Overall, the results demonstrate that the proposed classification model can distinguish between different solution types with relatively high accuracy. The unweighted variant consistently achieves higher overall accuracy (85.9% for incorrect solutions and 93.0% for correct solutions), reflecting strong

performance when the majority classes dominate. However, this comes at the cost of neglecting minority classes.

In the unweighted correct (TRUE) solution setting, although overall accuracy reached 92.97%, the rare class 102 (support = 9) obtained a recall of 0.000, meaning that none of its instances were correctly recognised. Similarly, in incorrect (FALSE) solutions, minority classes 205 and 206 achieved recall values of 0.464 and 0.321, respectively. These results indicate that high global accuracy may conceal weak recognition of underrepresented but potentially meaningful behavioural solution types. For this reason, recall is a particularly important evaluation metric in the context of the proposed assessment model, as it reflects the ability to correctly identify each solution type, including rare classes.

To address this limitation, a weighted training variant was additionally evaluated. The weighted variant reduces overall accuracy (83.4% for incorrect solutions and 90.6% for correct solutions) but substantially improves the recognition of underrepresented classes. For example, recall for class 102 increases from 0.000 to 1.000, and for classes 205 and 206 from 0.464 and 0.321 to 0.844 and 0.728, respectively. Thus, weighting ensures that rare solution types are not overlooked, even though this comes at the cost of slightly reduced global accuracy.

In the current experimental setting, where some solution types are represented by only a small number of data items, weighting is an effective way to ensure that rare classes are not overlooked. At the same time, it is reasonable to expect that with a larger dataset, especially for the rarest classes, the model could be retrained without weights and still achieve balanced recognition across all categories. Thus, weighting should be seen as a practical interim solution to compensate for data scarcity, while the unweighted model remains the preferable long-term approach once sufficient data are accumulated.

Across the main set of Bebras tasks used for training and evaluation, the model demonstrated strong performance in both correct (TRUE) and incorrect (FALSE) classification. The confusion matrices showed that the model was able to reliably separate both correct and incorrect solution strategies, achieving high accuracy and meaningful clustering of error types. This confirms that the methodological approach, transforming behavioural data into structured features and applying clustering and neural classification, is effective for modelling CT processes.

4.5 Generalization Experiment with New Bebras Tasks

To further test the robustness of the proposed model, two additional new Bebras tasks were introduced into the classification pipeline. In this study, new tasks refer to interactive tasks of the same interaction type whose solution data were not used during the training phase of the classifier. Two such tasks were used for evaluation: the task “Weekend at the Harbour”, which involves assigning docking places over several days (Figure 4.32), and the task “Age Codes”, based on combining coloured cards to represent numerical values (Figure 4.33). Both tasks differed from the training tasks in terms of their specific configuration and number of interactive objects, while belonging to the same interactivity group. This provided a practical validation of the model’s ability to generalize beyond the data it had been trained on.

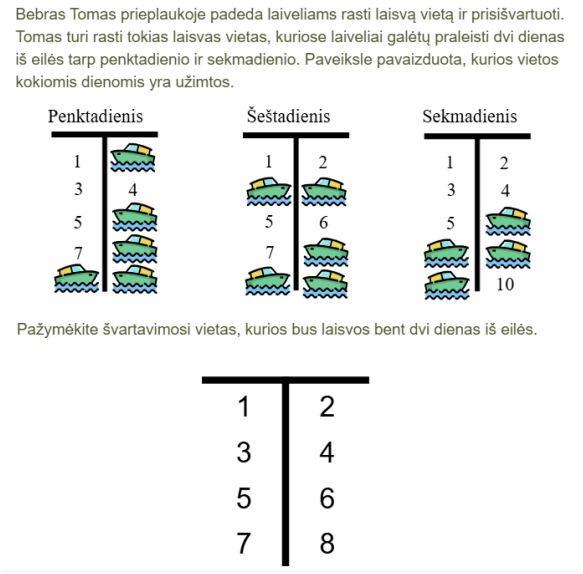


Figure 4.32: Interactive task “Weekend at the Harbour”

The first task (“Weekend at the Harbour”) contained ten interactive objects and included 14,750 correct (TRUE) and 47,843 incorrect (FALSE) solution attempts.

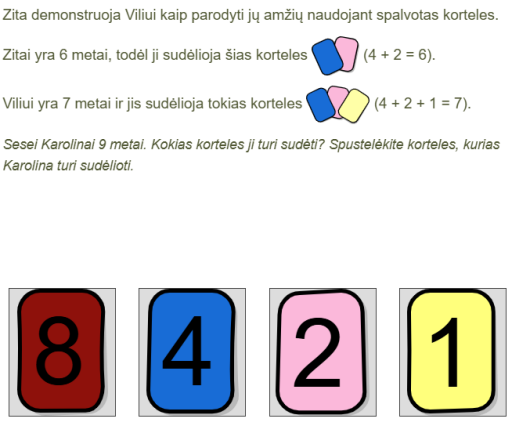


Figure 4.33: Interactive task “Age Codes”

The second task was shorter, containing only four objects, with 5,357 correct (TRUE) and 1,080 incorrect (FALSE) attempts.

The classification of correct solutions demonstrated stable performance across both tasks, with solutions consistently assigned into meaningful categories that aligned with the expected logical solution groups. This confirmed that the model can recognize correct problem-solving patterns even when applied to new tasks. For incorrect solutions, however, results were more mixed. In the first task, manual inspection of the predictions revealed that a large proportion of class 202 attempts were misclassified as class 201, while only about half of the actual 201 solutions were correctly identified. Such neighbour-class confusions indicate that, although the model detects broad tendencies in error patterns, it struggles to fully distinguish between them when faced with new data. These shortcomings highlight the importance of extending the training dataset with a wider variety of error examples in order to strengthen generalization.

In the second task, which contained fewer interactive objects, the model performed better: random guessing attempts were reliably grouped into the fully incorrect class 207, while partially reasoned solutions were more evenly distributed into logic-based error categories. The weighted version of the model also improved the balance of classifications by giving more attention to smaller classes. Overall, these experiments confirm that while the model already provides meaningful insights into students’ problem-solving behaviour, especially for correct solutions, its reliability on incorrect solutions can be further enhanced through additional training data. This supports the

broader conclusion that process-oriented classification is a promising direction for automated CT assessment but requires continued refinement for handling new error types.

The experiments with new tasks further validated the robustness of the model. While classification of correct solutions remained stable, the results for incorrect solutions revealed partial misclassifications, particularly when the new task involved more objects and complex error patterns. Nevertheless, the model was still able to assign random guessing attempts and partially reasoned incorrect solutions to different behavioural classes, indicating that the classification captures broad differences in whole-solution behaviour even when applied to tasks outside its training set.

To complement the quantitative evaluation of the model, including both performance on known tasks and generalisation to new tasks, a qualitative assessment involving domain experts was conducted. This expert validation aimed to determine whether the automatically derived solution groups and their labels meaningfully correspond to how educators themselves interpret and evaluate students' problem-solving behaviours.

4.6 Expert Validation of the Practical Part of the Model

To evaluate how the proposed assessment model aligns with practical educational perspectives, an expert survey was conducted. The goal of the survey was to examine how teachers and university lecturers interpret partially correct, corrected, or incorrect student solutions, and whether these interpretations correspond to the scoring principles defined in the proposed model.

The expert evaluation was based on the pilot interactive Bebras task "Coloring Page", in which students had to colour 12 regions using three colours so that no adjacent regions had the same colour. The task was scored on a scale from 0 to 1, allowing intermediate values. Rather than analysing individual solution paths, the experts were presented with generalised solution scenarios derived from this task and were asked to indicate what scoring principle should be applied in each case.

To clarify the evaluation procedure, examples of the questionnaire items are provided below. The experts were asked to evaluate different types of solution scenarios and indicate how such solutions should be assessed from a pedagogical perspective.

Example 1:

A student coloured the picture correctly but made one mistake during the process and corrected it independently. How would you evaluate this solution?

- Award the full score.
- Reduce the score symbolically, because the solution process was not error-free.
- Other (please specify).

Example 2:

A student coloured the picture correctly but made several mistakes during the process and corrected them independently. How would you evaluate this solution?

- Award the full score.
- Reduce the score in the same symbolic way as for a single mistake.
- Reduce the score more than in the case of a single mistake.
- Other (please specify).

Additional items asked experts to evaluate incorrect final solutions caused by one mistake, several mistakes, or many mistakes, using response options such as proportional scoring, minimal symbolic credit, zero score, or an alternative proposed by the respondent. A more detailed version of the questionnaire is provided in APPENDIX 2.

A total of 35 respondents participated in the study: 20 schoolteachers, 12 university or college lecturers, 2 individuals teaching in both sectors, and 1 instructor of extracurricular informatics activities. All participants held a bachelor's degree; 18 held a master's degree, and 8 had completed a PhD. Among schoolteachers, five were certified expert teachers (out of 37 nationwide), eight were methodologists, and six were senior teachers. Among the higher education staff, two were professors and seven were associate professors. Figure 4.34 presents the distribution of respondents' pedagogical experience, which is predominantly above 20 years.

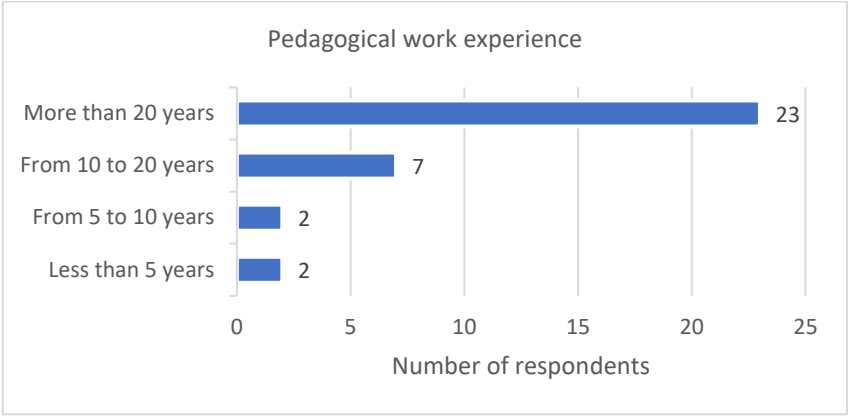


Figure 4.34: Chart represents the pedagogical experience of the participants

The first set of survey questions asked respondents to evaluate cases where a student makes one or more errors during the solution process but successfully corrects them and ultimately reaches the correct final answer.

According to the proposed assessment model, such solutions should receive a slightly reduced score, reflecting that the student's thinking process involved missteps that distinguish these attempts from flawless solutions. However, survey responses indicate that most teachers would still award full credit, with only four respondents choosing to reduce the score.

This result suggests that in real classroom practice, final correctness tends to outweigh the process, even when that process reveals notable differences in reasoning. It also indicates a divergence between pedagogical intuition and process-oriented assessment models, highlighting different priorities in educational evaluation compared to fields requiring strict error minimization, such as medicine or engineering.

The next two survey questions examined cases where a student solves most of the task correctly but provides an incorrect final answer due to one or several mistakes. Here, most respondents supported proportional scoring, which aligns with the logic of the proposed model. Nevertheless, some teachers indicated that they would assign a score of zero even if only a single mistake led to an incorrect final response.

Figure 4.35 and Figure 4.36 illustrate the preferred scoring strategies for single-error cases and for solutions containing several errors (2-4).

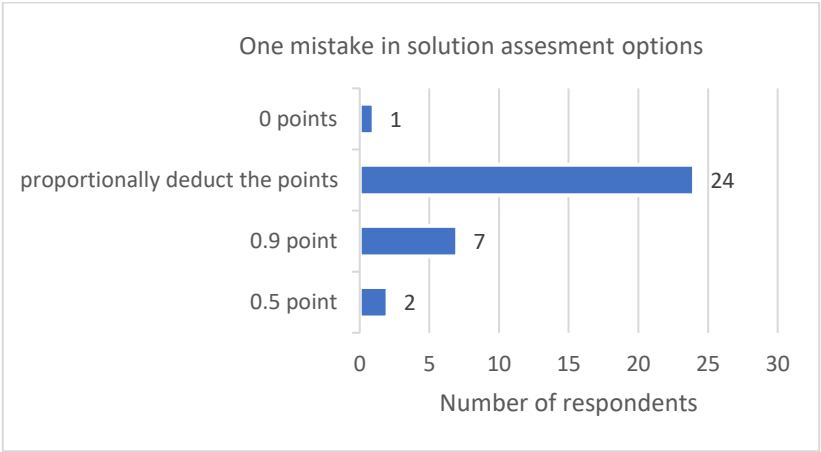


Figure 4.35: Chart representing assessment options for one mistake

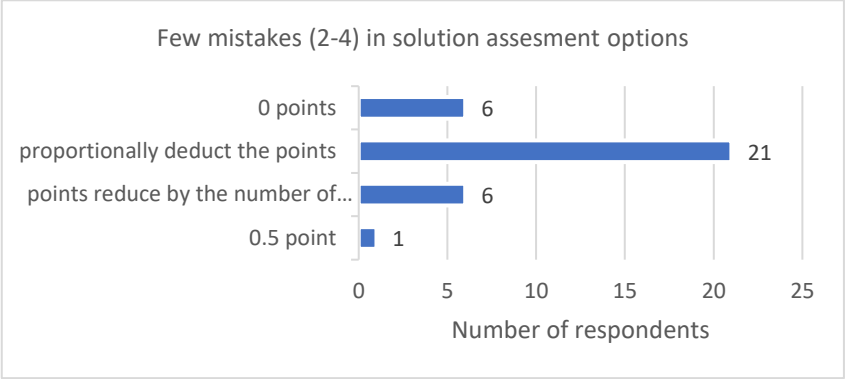


Figure 4.36: Chart representing assessment options for a few mistakes (2-4)

The most diverse opinions emerged when evaluating solutions containing many errors (e.g., eight or more), while still demonstrating some minimal correct reasoning. Although proportional scoring remained the most frequent choice, fewer than half of the respondents selected it. A substantial group preferred assigning symbolic minimal points (e.g., 0.1 or 0.2), and an increased proportion supported awarding a score of zero.

Respondents noted several challenges:

- Uncertainty about whether minimal correctness is intentional or the result of chance.

- The need for a clearer threshold defining when a solution is “too incorrect” to receive credit.
- The difficulty of distinguishing random clicking from genuine reasoning.

As demonstrated in the pilot task (“Coloring Page”), the clustering procedure successfully isolates random attempts into a separate group. Moreover, access to behavioural trace data (such as click sequences) provides a basis for distinguishing between random and thoughtful actions, an advantage not available in traditional testing.

Figure 4.37 shows the distribution of preferred scoring options for solutions involving many errors.

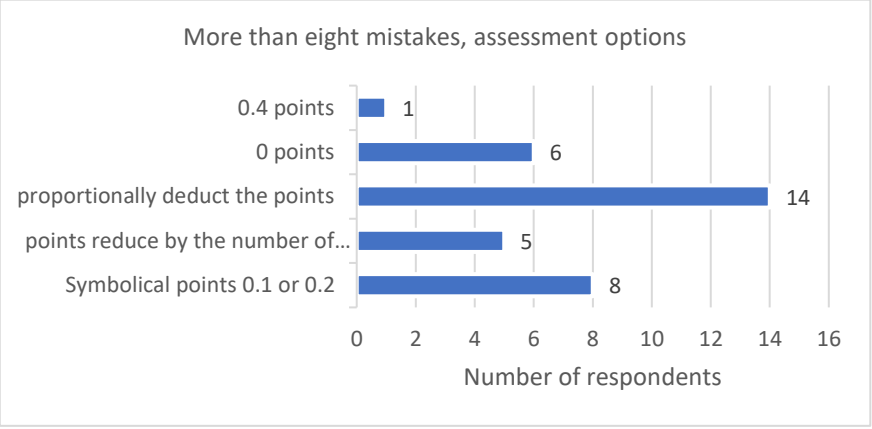


Figure 4.37: Chart representing assessment options for more than eight mistakes

Teachers and lecturers were also asked about the need to assess the results of a test, which requires solving problems rather than just answering questions, not only with a one or a zero, but also to differentiate the marking according to the solutions. The overwhelming majority agreed that such a model, which considers the solution and the mistakes made in it, is useful and shows what the student does not know and where attention should be paid. It was also mentioned that in such situations, it is very useful for the teacher to have the possibility to use a learning analysis. Figure 4.38: Chart representing attitude to the differentiated assessment model shows a diagram of teachers' attitudes towards a differentiated assessment model.

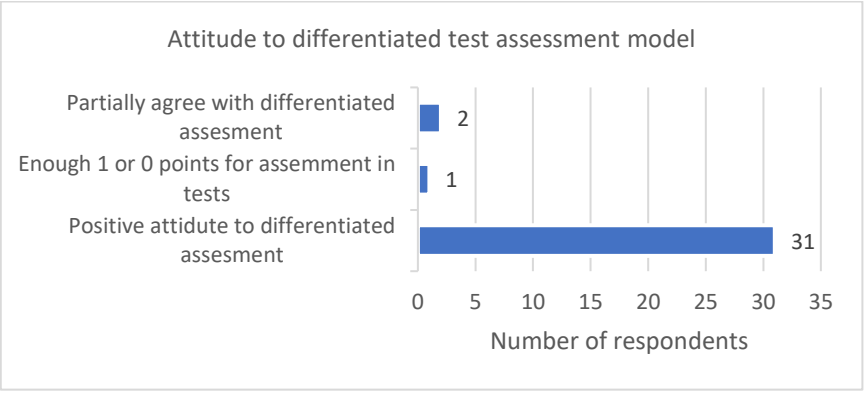


Figure 4.38: Chart representing attitude to the differentiated assessment model

Participants also provided open responses about the benefits and drawbacks of such assessment systems.

Perceived benefits**:**

- A more informative view of student understanding;
- Clearer diagnostic insight into errors;
- Improved opportunities for targeted teaching interventions.

Main challenges noted:

- Differentiated assessment demands more time and teacher involvement compared to binary scoring.
- Existing testing systems do not support detailed solution-level analytics.
- Potential subjectivity unless assessment rules are clearly defined.
- A need for a shared framework to ensure consistent scoring across teachers.

Respondents emphasized that students should be informed about the assessment criteria in advance to avoid perceived unfairness. Many also highlighted that automated assessment systems could substantially reduce teacher workload while improving feedback quality.

Overall, the survey results indicate that teachers and lecturers support differentiated scoring and recognise its pedagogical value. At the same time, they emphasise the need for clear rules, common standards, and automated tools to handle the complexity of analysing students' solution processes.

The findings confirm that the proposed process-oriented assessment model aligns with expert expectations and could serve as a solid foundation for future implementation and refinement.

4.7 Conclusions of the Chapter

The experimental validation presented in this chapter demonstrates that process-based behavioural data from interactive Bebras tasks can be systematically transformed, clustered, and classified to reveal meaningful patterns of problem-solving behaviour. The clustering experiments confirmed that distinct solution strategies, both correct and incorrect, can be identified from click sequences, timing data, and object-level interactions, enabling a unified labelling scheme transferable across tasks. In the experimental analysis, clustering produced interpretable structures, including seven classes of incorrect solutions and four classes of correct solutions. The dual-branch neural network classifier further showed that these behavioural representations support accurate automated recognition of solution types, achieving 93.0% accuracy for correct solutions and 85.9% for incorrect solutions in the unweighted setting. After applying class weighting, the accuracy slightly decreased to 90.6% and 83.4%, respectively, while improving the recognition of less frequent solution types. In particular, recall for minority classes increased from 0.464 and 0.321 to 0.844 and 0.728. These results provide strong empirical support for the feasibility of an automated assessment model grounded in learners' solving processes rather than outcomes alone.

Generalisation experiments with new tasks, together with expert validation, reinforced both the robustness and practical relevance of the proposed approach. While classification of correct solutions remained stable across new task contexts, incorrect solutions revealed subtler forms of misclassification, emphasising the need for more diverse error data to strengthen model generalisation. Expert educators positively evaluated the idea of differentiated, process-aware assessment and recognised its potential to provide richer diagnostic information than traditional binary scoring. In the expert evaluation, 94% of participants fully or partially agreed with the need for differentiated assessment, and 80% indicated that they would apply differentiated grading, reduce scores based on the number of errors, or assign partial credit when multiple errors (e.g., 2–4 errors) are present. At the same time, they highlighted practical constraints, such as the complexity of interpreting solution processes and the need for clear evaluation rules,

suggesting that automated models could serve as valuable support tools in real assessment settings.

GENERAL CONCLUSIONS

Computational thinking (CT) is often assessed in a simplified way that does not reflect how learners solve tasks. Most existing approaches rely on binary correctness, focusing only on the final answer and ignoring the solution process. The literature review conducted in this thesis revealed a clear gap: behavioural process data generated in interactive tasks are rarely used for assessment. As a result, important aspects of CT related to problem-solving remain unobserved.

This research addresses this gap by proposing and empirically validating a process-oriented CT assessment approach based on behavioural data from interactive Bebras tasks. The research objectives were achieved, including the analysis of existing methods, the exploration of behavioural data, the development of the model, and its validation using real data.

The thesis empirically demonstrates that process-oriented analysis of interactive problem-solving provides a more informative alternative to traditional correctness-based assessment and establishes a validated foundation for the use of behavioural data in CT assessment.

The main conclusions of this research, based on the results obtained in this study, are presented as follows:

1. The results confirm that behavioural solution data from interactive CT tasks can be automatically clustered into meaningful solution groups. Clustering revealed distinct patterns of solution behaviour, including systematic correct solutions, solutions with corrected errors, trial-and-error strategies, partially correct solutions, and incorrect approaches. In the pilot study, an interpretable structure emerged with approximately eight clusters. In the main experiments, Affinity Propagation identified seven classes of incorrect solutions and four classes of correct solutions, demonstrating that distinct solution types can be derived directly from behavioural data.

2. For interactive tasks belonging to the same interactivity group, clustering results revealed recurring behavioural solution patterns that are not task-specific but repeat across different tasks. Based on this, a common labelling scheme was developed and applied across tasks of the same interactivity group. This shows that behavioural solution

patterns can be consistently represented across tasks within the same interactivity group.

3. The proposed automated assessment model, based on a unified representation of solution processes, enables the classification of solutions with high accuracy. The model achieved 85.9% accuracy for incorrect solutions and 93.0% for correct solutions in the unweighted setting. After applying class weighting, accuracy slightly decreased to 83.4% and 90.6%, but the recognition of less frequent solution types improved. Recall for minority classes increased from 0.464 and 0.321 to 0.844 and 0.728, demonstrating that the model is capable of reliably identifying both dominant and less frequent behavioural patterns.

4. In addition, the practical relevance of the approach was supported by expert evaluation. A total of 94% of experts fully or partially agreed with the need for more differentiated assessment. Also, 80% indicated that they would either apply differentiated grading, reduce scores based on the number of errors, or assign partial credit when solutions contain multiple errors (e.g., 2–4 errors). This suggests that process-based assessment is consistent with existing educational practices.

5. At the same time, the results highlight certain limitations. The model performs best on tasks that generate structured behavioural data and belong to clearly defined interactivity groups. The classification of less frequent or more complex incorrect solutions depends on the size and diversity of the training data.

From a practical perspective, the proposed approach can be applied in various contexts where interactive problem-solving tasks are used, as it is not tied to a specific age group or educational level. It can support more detailed and process-oriented assessment, as well as provide richer feedback based on how solutions are constructed rather than only on final outcomes. At the same time, further research is needed to validate the model across a broader range of task types and learning environments.

BIBLIOGRAPHY

1. Alfredo, R., Echeverria, V., Jin, Y., Yan, L., Swiecki, Z., Gašević, D., & Martinez-Maldonado, R. (2024). Human-centred learning analytics and AI in education: A systematic literature review. *Computers and Education: Artificial Intelligence*, 6, 100215. https://doi.org/10.1016/j.caeai.2024.100215.

2. Asbell-Clarke, J., Rowe, E., Almeda, V., Edwards, T., Bardar, E., Gasca, S., Baker, R. S., & Scruggs, R. (2021). The development of students' computational thinking practices in elementary – and middle-school classes using the learning game, Zoombinis. *Computers in Human Behavior*, 115. https://doi.org/10.1016/j.chb.2020.106587

3. Banihashem, S. K., Aliabadi, K., Pourroostaei Ardakani, S., Delaver, A., & Nili Ahmadabadi, M. (2018). Learning analytics: A systematic literature review. *Interdisciplinary Journal of Virtual Learning in Medical Sciences*, *9*(2). https://doi.org/10.5812/ijvlms.63024

4. Bebras (n.d.). International Challenge on Informatics and Computational Thinking. https://bebras.org

5. Bellettini, C., Lonati, V., Monga, M., Morpurgo, A. (2023). All Green: How Different Age Groups Solved the Same Bebras Task. In: Pellet, JP., Parriaux, G. (eds) *Informatics in Schools. Beyond Bits and Bytes: Nurturing Informatics Intelligence in Education. ISSEP 2023. Lecture Notes in Computer Science*, vol 14296. Springer, Cham. https://doi.org/10.1007/978-3-031-44900-0_5

6. Bhatt, S., Verbert, K., & Van Den Noortgate, W. (2024). A Method for Developing Process-Based Assessments for Computational Thinking Tasks. *Journal of Learning Analytics*, *11*(2), 157-173. https://doi.org/10.18608/jla.2024.8291

7. Basawapatna, A., Koh, K. H., Repenning, A., Webb, D. C., & Marshall, K. S. (2011). Recognizing computational thinking patterns. In *Proceedings of the 42nd ACM technical symposium on computer science education* (pp. 245–250). New York, NY: ACM. https://doi.org/10.1145/1953163.1953241

8. Cansu, F. K., & Cansu, S. K. (2019). An Overview of Computational Thinking. *International Journal of Computer Science Education in Schools*, *3*(1), 17–30. https://doi.org/10.21585/ijcses.v3i1.53

9. Cen, Z., Zheng, L., & Zhan, Z. (2025). Design and performance validation of a computational thinking gamified intelligent tutoring system focusing on thinking process. In *2025 14th International Conference on Educational and Information Technology (ICEIT)* (pp. 181-187). IEEE. https://doi.org/10.1109/ICEIT64364.2025.10975898

10. Chang Z., Sun Y., Wu T. -Y. and Guizani M. (2018) "Scratch Analysis Tool (SAT): A Modern Scratch Project Analysis Tool based on ANTLR to Assess Computational Thinking Skills," *In 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pp. 950-955, https://doi.org/10.1109/IWCMC.2018.8450296.

11. Corrales-Álvarez, M., Ocampo, L. M., & Cardona Torres, S. A. (2024). Instruments for Evaluating Computational Thinking: A Systematic Review. *TecnoLógicas*, *27*(59).

12. CSTA.: K12 computer science standards (2017). Retrieved from https://www.csteachers.org/page/about-csta-s-k-12-nbsp-standards

13. Curzon, P., Bell, T., Waite, J., & Dorling, M. (2019). Computational Thinking. In A. V. Robins & S. A. Fincher (Eds.), *The Cambridge Handbook of Computing Education Research* (pp. 513–546). Cambridge University Press. https://doi.org/10.1017/9781108654555.018

14. Custer, S., King, E. M., Atinc, T. M., Read, L., & Sethi, T. (2018). Toward Data-Driven Education Systems: Insights into Using Information to Measure Results and Manage Change. *Center for Universal Education at The Brookings Institution*.

15. Dagienė, V., Sentance, S. (2016). It's Computational Thinking! Bebras Tasks in the Curriculum. In: Brodnik, A., Tort, F. (eds) *Informatics in Schools: Improvement of Informatics Knowledge and Perception. ISSEP 2016. Lecture Notes in Computer Science()*, vol 9973. Springer, Cham. https://doi.org/10.1007/978-3-319-46747-4_3

16. Dagienė, V., Stupurienė, G., & Vinikienė, L. (2017). Implementation of Dynamic Tasks on Informatics and Computational Thinking. *Baltic*

Journal of Modern Computing, 5(3), 306-316. https://doi.org/10.22364/bjmc.2017.5.3.05

17. Datnow, A., & Hubbard, L. (2016). Teacher capacity for and beliefs about data-driven decision making: A literature review of international research. *Journal of Educational Change*, *17*(1), 7-28. https://doi.org/10.1007/s10833-015-9264-2

18. Djambong, T., Freiman, V., Gauvin, S., Paquet, M., Chiasson, M. (2018). Measurement of Computational Thinking in K-12 Education: The Need for Innovative Practices. In: Sampson, D., Ifenthaler, D., Spector, J., Isaías, P. (eds) *Digital Technologies: Sustainable Innovations for Improving Teaching and Learning*. Springer, Cham. https://doi.org/10.1007/978-3-319-73417-0_12

19. El-Hamamsy, L., Zapata-Cáceres, M., Martín-Barroso, E., Mondada, F., Zufferey, J. D., Bruno, B., & Román-González, M. (2025). The Competent Computational Thinking Test (cCTt): A Valid, Reliable and Gender-Fair Test for Longitudinal CT Studies in Grades 3–6: L. El-Hamamsy et al. *Technology, Knowledge and Learning*, 30(3), 1607-1661. https://doi.org/10.1007/s10758-024-09777-8

20. Fagerlund, J., Häkkinen, P., Vesisenaho, M., & Viiri, J. (2020). Assessing 4th Grade Students' Computational Thinking through Scratch Programming Projects. *Informatics in Education*, 19(4), 611–640. https://doi.org/10.15388/INFEDU.2020.27

21. Ferguson, R., & Clow, D. (2017, March). Where is the evidence? A call to action for learning analytics. In *Proceedings of the seventh international learning analytics & knowledge conference* (pp. 56-65).https://doi.org/10.1145/3027385.3027396

22. Frey, B. J., & Dueck, D. (2007). Clustering by passing messages between data points. *Science*, *315*(5814), 972-976. https://doi.org/10.1126/science.1136800

23. Gardeli, A., & Vosinakis, S. (2019, September). ARQuest: A tangible augmented reality approach to developing computational thinking skills. In *2019 11th International Conference on Virtual Worlds and Games for Serious Applications (VS-Games)* (pp. 1-8). IEEE. https://doi.org/10.1109/VS-Games.2019.8864603

24. Grgurina, N., Barendsen, E., Suhre, C., Zwaneveld, B., & Van Veen, K. (2018). Assessment of modeling and simulation in secondary computing science education. In Cutts Q. & Muhling A. (Eds.), *ACM Int. Conf. Proc. Ser. Association for Computing Machinery*. https://doi.org/10.1145/3265757.3265764

25. Groher, I., Sabitzer, B., Demarle-Meusel, H., Kuka, L., & Hofer, A. (2021, April). Work-in-progress: Closing the gaps: Diversity in programming education. *In 2021 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1449-1453). IEEE. https://doi.org/10.1109/EDUCON46332.2021.9454035

26. Grover, S. (2011). Robotics and engineering for middle and high school students to develop computational thinking. In *Annual meeting of the American Educational Research Association*, New Orleans, LA

27. Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher*, *42*(1), 38-43. https://doi.org/10.3102/0013189X12463051

28. Grover, S., Jackiw, N., & Lundh, P. (2019). Concepts before coding: Non-programming interactives to advance learning of introductory programming concepts in middle school. *Computer Science Education,* 29(2–3), 106–135. https://doi.org/10.1080/08993408.2019.1568955

29. Guenaga, M., Eguíluz, A., Garaizar, P., & Gibaja, J. (2021). How do students develop computational thinking? Assessing early programmers in a maze-based online game. *Computer Science Education*, *31*(2), 259–289. https://doi.org/10.1080/08993408.2021.1903248

30. Guggemos, J. (2021). On the predictors of computational thinking and its growth at the high-school level. *Computers and Education*, 161. https://doi.org/10.1016/j.compedu.2020.104060

31. Guggemos, J., Seufert, S., & Román-González, M. (2023). Computational thinking assessment–towards more vivid interpretations. *Technology, Knowledge and learning*, 28(2), 539-568. https://doi.org/10.1007/s10758-021-09587-2

32. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

33. Hwang, J., Lee, S., Kim, Y., Zaman, M. (2023). Evaluating Young Children's Computational Thinking Skills Using a Mixed-Reality Environment. In: Stephanidis, C., Antona, M., Ntoa, S., Salvendy, G. (eds) *HCI International 2023 Posters. HCII 2023. Communications in Computer and Information Science*, vol 1834. Springer, Cham. https://doi.org/10.1007/978-3-031-35998-9_35

34. Jivet, I., Scheffel, M., Specht, M., & Drachsler, H. (2018). License to evaluate: Preparing learning analytics dashboards for educational practice. In *Proceedings of the 8th international conference on learning analytics and knowledge* (pp. 31-40). https://doi.org/10.1145/3170358.3170421

35. ISTE.: *Computational thinking: leadership toolkit* (2015). https:// www.iste.org/ computational-thinking

36. Izu, C., Mirolo, C., Settle, A., Mannila, L., & Stupurienė, G. (2017). Exploring Bebras tasks content and performance: A multinational study. *Informatics in Education*, 16(1), 39–59. https://doi.org/10.15388/infedu.2017.03

37. Kaleem, M., Hassan, M. A., & Khurshid, S. K. (2024). A machine learning-based adaptive feedback system to enhance programming skill using computational thinking. *IEEE Access*, *12*, 59431-59440. https://doi.org/10.1109/ACCESS.2024.3391873

38. Kanaki, K., & Kalogiannakis, M., (2022). Assessing Algorithmic Thinking Skills in Relation to Gender in Early Childhood, *EDUPIJ*, vol. 11, no. 2, 2022. https://doi.org/10.22521/edupij.2022.112.3.

39. Karakaş, E., & Yöndem, M. T. (2020). Performance-based evaluation of computational thinking skills using machine learning. In *2020 Turkish National Software Engineering Symposium (UYMS)* (pp. 1-5). IEEE. https://doi.org/10.1109/UYMS50627.2020.9247066

40. Kert, S. B., Kalelioğlu, F., & Gülbahar, Y. (2019). A holistic approach for computer science education in secondary schools. *Informatics in*

Education, 18(1), 131–150. Scopus. https://doi.org/10.15388/infedu.2019.06

41. Kia, F. S., Teasley, S. D., Hatala, M., Karabenick, S. A., & Kay, M. (2020). How patterns of students dashboard use are related to their achievement and self-regulatory engagement. *In Proceedings of the tenth international conference on learning analytics & knowledge* (pp. 340-349). https://doi.org/10.1145/3375462.3375472

42. Kingma, D. P., and Ba J. (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:*1412.6980,

43. Koh, K. H., Basawapatna, A., Bennett, V., & Repenning, A. (2010, September). Towards the automatic recognition of computational thinking for adaptive visual language learning. *In 2010 IEEE symposium on visual languages and human-centric computing* (pp. 59-66). IEEE. https://doi.org/10.1109/VLHCC.2010.17

44. Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in Human Behavior*, 72, 558–569. https://doi.org/10.1016/j.chb.2017.01.005

45. Labusch, A., Eickelmann, B., & Vennemann, M. (2019). Computational thinking processes and their congruence with problem-solving and information processing. *In Computational thinking education* (pp. 65-78). Singapore: Springer Singapore. https://doi.org/10.1007/978-981-13-6528-7_5

46. Labusch, A., & Eickelmann, B. (2020). Computational Thinking Competences in Countries from Three Different Continents in the Mirror of Students' Characteristics and School Learning. In S. C. Kong, H. U. Hoppe, T. C. Hsu, R. H. Huang, B. C. Kuo, K. Y. Li, C. K. Looi, M. Milrad, J. L. Shih, K. F. Sin, K. S. Song, M. Specht, F. Sullivan, & J. Vahrenhold (Eds.), *Proceedings of International Conference on Computational Thinking Education* 2020 (pp. 2–7). The Education University of Hong Kong.

47. Leonard, A. E., Daily, S. B., Jörg, S., & Babu, S. V. (2021). Coding moves: Design and research of teaching computational thinking through dance choreography and virtual interactions. *Journal of Research on*

Technology in Education, 53(2), 159–177. https://doi.org/10.1080/15391523.2020.1760754

48. Liao, J., Zhong, L., Zhe, L., Xu, H., Liu, M., & Xie, T. (2024). Scaffolding computational thinking with ChatGPT. *IEEE Transactions on Learning Technologies*, *17*, 1628-1642. https://doi.org/10.1109/TLT.2024.3392896

49. Liu, T. (2024). Assessing implicit computational thinking in game‐based learning: A logical puzzle game study. *British Journal of Educational Technology*, 55(5), 2357-2382. https://doi.org/10.1111/bjet.13443

50. Liu, Z., Zhi, R., Hicks, A., and Barnes, T., (2017). Understanding problem solving behavior of 6–8 graders in a debugging game, *Computer Science Education*, vol. 27, no. 1, pp. 1–29, https://doi.org/10.1080/08993408.2017.1308651.

51. Lockwood, J., Mooney, A. (2017). Computational Thinking in Education: Where does it fit? A systematic literary review. *Department of Computer Science, Maynooth University*, Maynooth, Co. Kildare, Ireland, Retrieved from https://arxiv.org/pdf/1703.07659.pd

52. Luo, Q., & Zhang, S. (2024). Game-Based Assessment for Computational Thinking: A Systematic Review. In *2024 IEEE Integrated STEM Education Conference (ISEC)* (pp. 01-08). IEEE. https://doi.org/10.1109/ISEC61299.2024.10664854

53. Ma, H., Zhao, M., Wang, H., Wan, X., Cavanaugh, T. W., & Liu, J. (2021). Promoting pupils' computational thinking skills and self-efficacy: A problem-solving instructional approach. *Educational Technology Research and Development*, 69(3), 1599-1616. https://doi.org/10.1007/s11423-021-10016-5

54. MacQueen, J. (1967). Multivariate observations. In Proceedings of the *5th Berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 281-297). Oakland, CA, USA: University of California press.

55. Merceron, A. (2015, September). Educational Data Mining/Learning Analytics: Methods, Tasks and Current Trends. In *DeLFI WOrkshops* (pp. 101-109).

56. Metcalf, S. J., Reilly, J. M., Jeon, S., Wang, A., Pyers, A., Brennan, K., & Dede, C. (2023). Assessing computational thinking through the lenses of functionality and computational fluency. In *Assessing Computational Thinking* (pp. 87-111). Routledge. https://doi.org/10.1080/08993408.2020.1866932

57. Mohammadi, M., Tajik, E., Martinez-Maldonado, R., Sadiq, S., Tomaszewski, W., & Khosravi, H. (2025). Artificial intelligence in multimodal learning analytics: a systematic literature review. *Computers and Education: Artificial Intelligence*, 8, 100426. https://doi.org/10.1016/j.caeai.2025.100426

58. Molenaar, I., Horvers, A., Dijkstra, R., & Baker, R. S. (2020, March). Personalized visualizations to promote young learners' SRL: The learning path app. In Proceedings of the tenth international conference on learning analytics & knowledge (pp. 330-339). https://doi.org/10.1145/3375462.337546

59. Moreno-León, J., & Robles, G. (2015). Analyze your Scratch projects with Dr. Scratch and assess your computational thinking skills. In *Scratch conference* (pp. 12–15).

60. Moreno-León, J., Robles, G., & Román-González, M. (2015). Dr. Scratch: Automatic analysis of scratch projects to assess and foster computational thinking. RED. Revista de Educación a Distancia, (46), 1-23.

61. Mühling, A., Ruf, A., & Hubwieser, P. (2015, November). Design and first results of a psychometric test for measuring basic programming abilities. In *Proceedings of the workshop in primary and secondary computing education* (pp. 2-10). https://doi.org/10.1145/2818314.2818320

62. Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (pp. 807-814).

63. Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. New York: Basic Books.

64. Pérez-Marín, D., Hijón-Neira, R., Bacelo, A., & Pizarro, C. (2020). Can computational thinking be improved by using a methodology based on metaphors and scratch to teach computer programming to children? *Computers in Human Behavior*, 105. https://doi.org/10.1016/j.chb.2018.12.027

65. Piatti, A., Adorni, G., El-Hamamsy, L., Negrini, L., Assaf, D., Gambardella, L., & Mondada, F. (2022). The CT-cube: A framework for the design and the assessment of computational thinking activities. *Computers in Human Behavior Reports*, 5, 100166. https://doi.org/10.1016/j.chbr.2021.100166

66. Poulakis, E., & Politis, P. (2021). Computational thinking assessment: Literature review. Research on e-learning and ICT in education: *Technological, pedagogical and instructional perspectives*, 111-128. https://doi.org/10.1007/978-3-030-64363-8_7

67. Prieto, L. P., Sharma, K., Dillenbourg, P., & Jesús, M. (2016). Teaching analytics: towards automatic extraction of orchestration graphs using wearable sensors. In *Proceedings of the sixth international conference on learning analytics & knowledge* (pp. 148-157). https://doi.org/10.1145/2883851.2883927

68. Román-González, M., Pérez-González, J.-C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior*, 72, 678–691. https://doi.org/10.1016/j.chb.2016.08.047

69. Román-González, M., Moreno-León, J., & Robles, G. (2019). Combining assessment tools for a comprehensive evaluation of computational thinking interventions. In Computational thinking education (pp. 79-98). Singapore: Springer Singapore. https://doi.org/10.1007/978-981-13-6528-7_6

70. Romero, C., & Ventura, S. (2010). Educational data mining: a review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (applications and reviews)*, 40(6), 601-618. https://doi.org/10.1109/TSMCC.2010.2053532

71. Romero, C., & Ventura, S. (2020). Educational data mining and learning analytics: An updated survey. *Wiley interdisciplinary reviews: Data mining and knowledge discovery*, 10(3), e1355. https://doi.org/10.1002/widm.1355

72. Rowe E., Almeda V., Asbell-Clarke J., Scruggs R., Baker R., Bardar E., Gasca S. (2021), Assessing implicit computational thinking in Zoombinis puzzle gameplay, *Computers in Human Behavior*, Volume 120, https://doi.org/10.1016/j.chb.2021.106707.

73. Rowe, E., Almond, R. G., & Almeda, M. V. (2025). Validating game-based learning assessment of students' computational thinking practices using Bayesian networks and machine-learning based detectors. *Journal of Research on Technology in Education*, 1-18. https://doi.org/10.1080/15391523.2025.2590018

74. Sathyanarayanan, S., & Tantri, B. R. (2024). Confusion matrix-based performance evaluation metrics. *African Journal of Biomedical Research*, *27*(4S), 4023-4031. https://doi.org/10.53555/AJBR.v27i4S.4345

75. Seraj, A., Mohammadi-Khanaposhtani, M., Daneshfar, R., Naseri, M., Esmaeili, M., Baghban, A., ... & Eslamian, S. (2023). Cross-validation. In *Handbook of hydroinformatics* (pp. 89-105). Elsevier. https://doi.org/10.1016/B978-0-12-821285-1.00021-X

76. Siemens, G., & Baker, R. S. D. (2012, April). Learning analytics and educational data mining: towards communication and collaboration. In Proceedings of the 2nd international conference on learning analytics and knowledge (pp. 252-254). https://doi.org/10.1145/2330601.2330661

77. Statter, D., & Armoni, M. (2020). Teaching abstraction in computer science to 7th grade students. *ACM Transactions on Computing Education*, 20(1), 8–837. https://doi.org/10.1145/3372143

78. Sung, J. (2022). Assessing young Korean children's computational thinking: A validation study of two measurements. *Education and Information Technologies*, *27*(9), 12969-12997. https://doi.org/10.1007/s10639-022-11137-x

79. Tan, B., Jin, H. Y., & Cutumisu, M. (2024). The applications of machine learning in computational thinking assessments: a scoping review. *Computer Science Education*, *34*(2), 193-221. https://doi.org/10.1080/08993408.2023.2245687

80. Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers and Education*, 148. https://doi.org/10.1016/j.compedu.2019.103798

81. Tikva, C., & Tambouris, E. (2021). Mapping computational thinking through programming in K-12 education: A conceptual model based on a systematic literature Review. *Computers and Education*, 162. https://doi.org/10.1016/j.compedu.2020.104083

82. Tonbuloğlu, B., & Tonbuloğlu, I. (2019). The effect of unplugged coding activities on computational thinking skills of middle school students. *Informatics in Education*, 18(2), 403–426. https://doi.org/10.15388/infedu.2019.19

83. Troiano G., Snodgrass S., Argımak E., Robles G., Smith G., Cassidy M., Tucker-Raymond E., Puttick G., and Harteveld C. (2019). Is My Game OK Dr. Scratch? Exploring Programming and Computational Thinking Development via Metrics in Student-Designed Serious Games for STEM. In *Proceedings of the 18th ACM International Conference on Interaction Design and Children Association for Computing Machinery*, New York, NY, USA, 208–219. https://doi.org/10.1145/3311927.3323152

84. Tsai, M. J., Liang, J. C., & Hsu, C. Y. (2021). The computational thinking scale for computer literacy education. *Journal of Educational Computing Research*, 59(4), 579-602. https://doi.org/10.1177/0735633120972356

85. Ullah, A., Muhammad, K., Hussain, T., Lee, M., & Baik, S. W. (2020). Deep LSTM-based sequence learning approaches for action and activity recognition. In *Deep Learning in Computer Vision* (pp. 127-150). CRC Press.

86. Van Leeuwen, A., & Rummel, N. (2020, March). Comparing teachers' use of mirroring and advising dashboards. *In Proceedings of the tenth international conference on learning analytics & knowledge* (pp. 26-34). https://doi.org/10.1145/3375462.337547

87. Viberg, O., Hatakka, M., Bälter, O., & Mavroudi, A. (2018). The current landscape of learning analytics in higher education. *Computers in Human Behavior*, *89*, 98-110. https://doi.org/10.1016/j.chb.2018.07.027

88. Waite, J., Curzon, P., Marsh, W., & Sentance, S. (2020). Difficulties with design: The challenges of teaching design in K-5 programming. *Computers and Education*, 150. https://doi.org/10.1016/j.compedu.2020.103838

89. Wei, X., Lin, L., Meng, N., Tan, W., Kong, S.-C., & Kinshuk. (2021). The effectiveness of partial pair programming on elementary school students' Computational Thinking skills and self-efficacy. *Computers and Education*, 160. https://doi.org/10.1016/j.compedu.2020.104023

90. Weintrop, D., & Wilensky, U. (2015). Using commutative assessments to compare conceptual understanding in blocks-based and text-based programs. *In Proceedings of the eleventh annual international conference on international computing education research, ICER15* (pp. 101e110). http://dx.doi.org/10.1145/ 2787622.2787721

91. Weintrop, D., & Wilensky, U. (2019). Transitioning from introductory block-based and text-based environments to professional programming languages in high school computer science classrooms. *Computers and Education*, 142. Scopus. https://doi.org/10.1016/j.compedu.2019.103646

92. Werneburg, S., Manske, S., & Hoppe, H. U. (2018). ctGameStudio – A game-based learning environment to foster computational thinking. In *26th international conference on computers in education, Philippines*.

93. Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.

94. Wu, S. Y., & Su, Y. S. (2021). Visual programming environments and computational thinking performance of fifth-and sixth-grade students. Journal of Educational Computing Research, 59(6), 1075-1092. https://doi.org/10.1177/0735633120988807

95. Yan, L., Echeverria, V., Jin, Y., Fernandez-Nieto, G., Zhao, L., Li, X., Alfredo, R., Swiecki, Z., Gašević, D., & Martinez-Maldonado, R. (2024). Evidence-based multimodal learning analytics for feedback and

reflection in collaborative learning. *British Journal of Educational Technology*, 55, 1900–1925. https://doi.org/10.1111/bjet.13498

96. Yağcı, M. (2019). A valid and reliable tool for examining computational thinking skills. *Education and Information Technologies*, 24(1), 929–951. Scopus. https://doi.org/10.1007/s10639-018-9801-8

97. Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation*, *31*(7), 1235-1270. https://doi.org/10.1162/neco_a_01199

98. Zapata-Cáceres, M., Martín-Barroso, E., & Román-González, M. (2021). Collaborative game-based environment and assessment tool for learning computational thinking in primary school: A case study. *IEEE Transactions on Learning Technologies*, 14(5), 576-589. https://doi.org/10.1109/TLT.2021.3111108

99. Zapata-Cáceres, M., Marcelino, P., El-Hamamsy, L., & Martín-Barroso, E. (2024). A Bebras Computational Thinking (ABC-Thinking) program for primary school: Evaluation using the competent computational thinking test. *Education and Information Technologies*, 29(12), 14969-14998. https://doi.org/10.1007/s10639-023-12441-w

100. Zhang, L. C., Nouri, J., & Rolandsson, L. (2020). Progression of Computational Thinking Skills in Swedish Compulsory Schools with Block-based Programming. ACE – *Proc. Australas. Comput. Educ. Conf., Held Conjunction Australas. Comput. Sci. Week*, 66–75. https://doi.org/10.1145/3373165.3373173.

APPENDIX 1 BEBRAS Challenge Privacy Policy

Informatinio mąstymo konkurso „BEBRAS“ privatumo politika

1. „Lietuvos kompiuterininkų sąjunga“ (toliau — LIKS) yra svetainių www.bebras.lt ir lt.bebras.lt tvarkomų asmens duomenų valdytojas ir informatinio mąstymo konkurso „BEBRAS“ organizatorius.

2. Atkreipiame dėmesį, jog šios Privatumo politikos nuostatos galioja visiems klientų (toliau — Vartotojas), apsilankiusių bebras.lt ir lt.bebras.lt internetinėje Svetainėje (toliau — Svetainė), veiksmams, kuriuos jie atlieka Svetainėje, įskaitant registraciją, dalyvavimą konkurse, skelbiamos informacijos skaitymą, bet kokio pobūdžio informacijos ir (arba) duomenų pateikimą bei gavimą (toliau — Paslaugos).

3. Naudodamasis Paslaugomis Vartotojas sutinka su Privatumo politikos nuostatomis. Jeigu Vartotojas nesutinka laikytis LIKS Privatumo politikos nuostatų, Vartotojas negali toliau naudotis jokiomis Svetainės Paslaugomis.

4. Vartotojų pateikiamus asmeninius duomenis saugo ir administruoja LIKS. Tiek LIKS, tiek jokios kitos trečiosios šalies sprendimu, išskyrus Lietuvos Respublikos įstatymų numatytus atvejus, Vartotojų asmeniniai duomenys negali būti perduoti trečiosioms šalims be atskiro ir aiškaus Vartotojo sutikimo.

5. LIKS turi teisę vienašališkai savo nuožiūra bet kuriuo metu keisti Privatumo politikos nuostatas. Bet koks Privatumo politikos nuostatų pakeitimas įsigalioja nuo jų paskelbimo Svetainėje, o prisiregistravusiems Vartotojams papildomai yra pranešama el. paštu. Jei perskaitęs šias Privatumo politikos nuostatas Vartotojas nusprendžia toliau naudotis Svetainės suteikiamomis galimybėmis, tai yra laikoma patvirtinimu, kad su nuostatomis Vartotojas sutinka bei jas priima.

6. Pažymime, kad LIKS jokiomis aplinkybėmis neatsako už galimus padarinius, sukeltus nenugalimų jėgų (force majeure) įtakos, kurių negalėjo niekaip paveikti ar numatyti.

7. LIKS turi teisę bet kokiu, Lietuvos Respublikos teisės aktų neuždraustu, būdu naudotis visa sukaupta informacija apie Vartotoją, siekiant svetainės Vartotojams suteikti aukščiausios kokybės paslaugas.

Vartotojų registracija

8. Tam, kad Vartotojai gautų prieigą visoms Svetainės paslaugoms, jie turi prisiregistruoti ir pateikti asmens duomenis Svetainėje. Vartotojai teikiantys

savo kontaktinę informaciją suvokia, sutinka ir pageidauja, kad LIKS komanda galėtų su jais susisiekti.

9. LIKS Vartotojų asmeniniai duomenys pateikti Svetainės sistemoje yra apdorojami tiek rankiniu, tiek automatiniu būdu. Vartotojas turi teisę nesutikti, kad jo asmeniniai duomenys būtų tvarkomi LIKS, nepateikti savo asmeninių duomenų ir nesinaudoti Svetaine.

10. Be duomenų rinkimo per Svetainę, Vartotojas bet kuriuo metu gali būti paprašytas pateikti savo asmeninius duomenis per visus kitus bendravimo kanalus, pavyzdžiui, telefonu ar laiškais.

11. Vartotojo kontaktiniams duomenims pasikeitus, jis privalo papildyti ir (arba) pakeisti juos Svetainėje.

12. Vartotojas yra visiškai atsakingas už visų prisijungimo duomenų slaptumą.

13. Naudojantis Svetaine draudžiama pateikti klaidinančią, kitus asmenis ar subjektus kompromituojančią ar įžeidžiančią informaciją, kuri galėtų padaryti tiesioginę ar netiesioginę žalą LIKS ir jo vartotojams. Asmenys, teikiantys klaidingą (žalingą) informaciją, prisiima visišką atsakomybę už galimus padarinius, atsirandančius dėl tokios informacijos pateikimo ir tiesiogiai atsako pagal Lietuvos Respublikos įstatymus.

14. LIKS turi teisę pašalinti Vartotojo registraciją ir bet kokius asmens padarytus įrašus iš savo Svetainės duomenų bazės, jeigu Vartotojas pažeidė svetainės taisykles. Taip pat LIKS turi teisę kreiptis į atitinkamas institucijas dėl asmens patraukimo atsakomybėn dėl jo neleidžiamais veiksmais padarytos materialinės ir nematerialinės žalos LIKS, jo partneriams, klientams ar kitiems asmenims bei subjektams.

Renkami duomenys

15. Vartotojas sutinka, kad jo vardas, pavardė, klasė, mokyklos pavadinimas, apskritis, savivaldybė, rezultatas, atsakymai į klausimus ir kita su dalyvavimu konkurse susijusi informacija būtų saugoma LIKS bei matoma Vartotojui ir jo mokytojui. Viešai nėra skelbiama jokia vartotojų informacija.

16. Vartotojas sutinka, kad jo duomenys gali būti perduoti savivaldybėms apdovanojimų tikslais. Jokioms kitoms trečiosioms šalims duomenys negali būti teikiami.

17. Vartotojas sutinka, kad LIKS naudoja slapukus ir jų dėka renka tiek užsiregistravusių, tiek neužsiregistravusių Vartotojų informaciją, kuri yra

panaudojama statistikos ir svetainės gerinimo tikslais. Šis duomenų tvarkymas neleidžia nustatyti Vartotojo tapatybės.

Registracijos pašalinimas

18. Prisiregistravę Vartotojai LIKS Svetainės sistemoje gali bet kada paprašyti pašalinti savo visus asmeninius duomenis iš LIKS duomenų bazės, parašydami prašymą į info@bebras.lt, ir jie bus pašalinti.

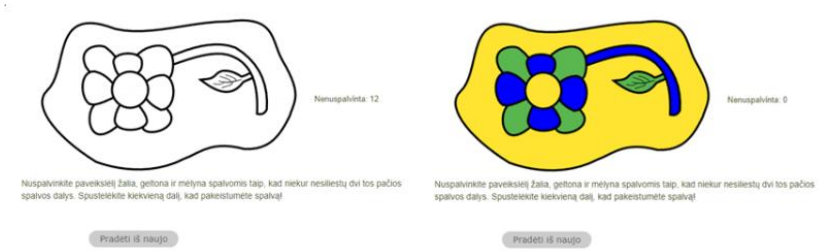
Asmens duomenų saugojimo terminai

19. LIKS tvarkomų asmens duomenų saugojimo terminai, atsižvelgiant į asmens duomenų tvarkymo tikslus, yra šie:

- Vartotojo asmeniniai duomenys, tvarkomi Paslaugų teikimo tikslu, yra saugomi 5 metus nuo paskutinio Vartotojo prisijungimo.
- Vartotojo asmeniniai duomenys yra saugomi ne ilgiau kaip vienerius metus po to, kai registruotas Vartotojas raštu ar automatiniu būdu atsisako LIKS teikiamų paslaugų.
- Sąžiningo dalyvavimo konkurse tikslu duomenys yra tvarkomi ir saugomi 25 metus.
- Mokslinių tyrimų tikslu duomenys tvarkomi neribotą laiką, tačiau jie saugomi anonimizuoti.

APPENDIX 2 Questionnaire about Interactive Bebras Tasks Assessment

Bebro uždavinio su sprendimu pavyzdys:



Užduotis: Nuspalvinkite paveikslėlį žalia, geltona ir mėlyna spalvomis taip, kad niekur nesiliestų dvi tos pačios spalvos dalys. Iš viso reikia nuspalvinti 12 sričių. Vertinama 1 balu. Maksimalus įvertinimas 1 balas, minimalus – 0 balų. Taip pat galimi tarpiniai vertinimai dešimtainės trupmenos pavidalu tarp 0 ir 1.

1 klausimas
Mokinys nuspalvino paveikslėlį teisingai, tačiau spalvinimo metu padarė vieną klaidą, ją pastebėjo ir pataisė. Kaip vertintumėte?

- 1 balu.
- Simboliškai mažiau, negu vienu balu, nes 1 balu būtų įvertintas užduoties sprendimas, kurio sprendimo eigoje nebuvo klaidų. Šiuo atveju galima būtų vertinti pavyzdžiui, 0,98.
- Kita (įrašykite).

2 klausimas
Mokinys nuspalvino paveikslėlį teisingai, tačiau spalvinimo metu padarė keletą klaidų ir pats ištaisė jas sprendimo eigoje. Kaip vertintumėte?

- 1 balu.
- Simboliškai mažiau, taip pat, kaip ir vieną klaidą padariusio mokinio sprendimą (pvz., 0,98 balo).
- Vertinčiau truputį mažesniu balu, negu sprendimą to mokinio, kuris padarė vieną klaidą (pvz., 0,95 balo).
- Kita (įrašykite).

3 klausimas

Mokinys nuspalvino paveikslėlį, tačiau spalvinimo metu padarė vieną klaidą, jos nepastebėjo ir nepataisė. Kaip vertintumėte?

- 0,95 balo, nes viena klaida
- Išskaičiuočiau proporcingai, koks būtų balas (jei reikia nuspalvinti 12 sričių, o nuspalvinti 11 nepasilietų gerai, tai vertinčiau 11/12 = 0,92 balo)
- Kita (įrašykite)

4 klausimas

Mokinys nuspalvino paveikslėlį, tačiau padarė daugiau klaidų ir jų nepataisė. Kaip vertintumėte?

- Kiek padarė klaidų, tiek ir sumažėja balas, pvz., jei tris klaidas, tai 0,7 balo
- Balą išskaičiuočiau proporcingai (pvz. jei reikia nuspalvinti 12 sričių, o nuspalvinta gerai 9/12 = 0,75 balo)
- Kita (įrašykite)

5 klausimas

Mokinys padarė labai daug klaidų (pvz., daugiau kaip 8), tačiau mažą dalį užduoties vis tik atlikta teisingai. Kaip vertintumėte?

- Prisiminčiau simbolinį teisingą dalį įvertinčiau pavyzdžiui 0,1 ar 0,2 balo, nes mokinys nesuprato pritaikė užduoties.
- Kiek padarė klaidų, tiek ir mažėja balas, tai jei 9 klaidos, tai 0,1 balo.
- Proporciškai skaičiuočiau: jei 9 klaidos, 3 iš 12 = 0,25 balo
- Kita (įrašykite)

6 klausimas

Kaip manote, ar tokia sistema, kuri diferencijuotų testų (kuriuose reikia spręsti užduotis) vertinimą padeda mokiniams ir mokytojams geriau suprasti, kur yra žinių ar mąstymo spragos ir ateityje labiau į jas atkreipti dėmesį? Pakomentuokite.

- Taip, nes padarius vieną ar daugiau klaidų užduotinio sprendimas nėra vertinamas nulis ir mokinys mato, kur jam reikia pasitempti.
- Ne, užtenka, kad testai yra vertinami 0 arba 1.
- Kita (įrašykite)

7 klausimas

Pakomentuokite papildomai testų ir užduočių, kurioms reikalingas sprendimas, vertinimo trūkumus ir galimybes. (Atviras atsakymas – be pasirenkamų variantų.)

8 klausimas
Jūsų pedagoginė patirtis:

- Iki 5 metų
- Nuo 5 iki 10 metų
- Nuo 10 iki 20 metų
- Daugiau nei 20 metų

9 klausimas
Jūsų išsilavinimas:

- Bakalauro laipsnis
- Magistro laipsnis
- Daktaro laipsnis
- Kita (įrašykite)

10 klausimas
Pedagoginį darbą dirbate (galite pasirinkti kelis):

- Mokykloje
- Universitete
- Kolegijoje
- Kita (įrašykite)

11 klausimas
Jei esate mokytojas, kokią kvalifikacinę kategoriją turite?

- Mokytojas
- Vyresnysis mokytojas
- Metodininkas
- Ekspertas
- Nesu mokytojas

12 klausimas
Jei dėstote universitete (ar kolegijoje), kokias pareigas susijusias su dėstymu užimate?

- Lektorius
- Jaunesnysis asistentas

- Asistentas
- Docentas
- Profesorius
- Nedėstau universitete

SUMMARY IN LITHUANIAN

Įvadas

Informatinis mąstymas (IM) vis dažniau pripažįstamas kaip viena iš svarbiausių XXI amžiaus kompetencijų. Jis laikomas pagrindine ne tik informatikos specialistų, bet ir visų besimokančių, kurie turi gebėti orientuotis skaitmeninių technologijų formuojamame pasaulyje, kompetencija. Informatinio mąstymo sąvoką 1980 m. pirmą kartą. pristatė Seymour'as Papert'as kaip būdą gerinti mokymąsi ir mąstymą naudojant kompiuterius. Vėliau Jeannette Wing (2006) performulavo IM kaip universalią problemų sprendimo kompetenciją, teigdama, kad ši „apima problemų sprendimą, sistemų projektavimą ir žmogaus elgesio supratimą, remiantis pamatinėmis informatikos sąvokomis". Nuo tada ši sąvoka plačiai aptariama, tobulinama ir taikoma įvairiose švietimo ir mokslinių tyrimų srityse (Román-González et al., 2017; Tang et al., 2020; Tikva & Tambouris, 2021).

IM svarba yra plačiai pripažįstama visame pasaulyje (Bocconi et al., 2022; Kampylis et al., 2023). Tačiau vienas iš nuolat keliamų klausimų yra tas, kaip veiksmingai vertinti informatinį mąstymą.

Tyrimo problema

Nors dažnai vertinamas kaip visuma, informatinis mąstymas taip pat apibūdinamas kaip gebėjimai skaidyti, atpažinti modelius, abstrahuoti ir mąstyti algoritmiškai (Grover & Pea, 2013). Tačiau realiame problemų sprendimo procese šie gebėjimai dažnai sutampa ir nėra lengvai atskiriami. Tad informatinio mąstymo vertinimas vis dar daugiausia grindžiamas galutiniu uždavinio rezultatu. Daugelyje esamų metodų mokiniai vertinami tik pagal tai, ar jų atsakymas teisingas, o į tai, kaip pasiektas sprendimas, neatsižvelgiama.

Tai labai aiškiai matoma interaktyviuose informatinio mąstymo uždaviniuose, pavyzdžiui, naudojamuose konkurse „Bebras". Spręsdami uždavinį, mokiniai aktyviai dirba su uždavinio elementais, tačiau jų pasiekimai nėra vertinami. Sudėtinguose uždaviniuose vėlesniame etape padaryta klaida, gali lemti visiškai neteisingą rezultatą, net jei iš pradžių mąstymas buvo teisingas. Kitaip nei matematikoje, kur už teisingą tarpinį sprendimą skiriami tam tikri balai, informatinio mąstymo vertinime ši informacija paprastai prarandama. Naujausi tyrimai vis dažniau rodo, kad

sprendimų procesų analizė gali suteikti išsamesnį vaizdą apie mokinių informatinį mąstymą (Corrales-Álvarez et al., 2024; Luo & Zhang, 2024). Elgsenos žymės, tokios kaip paspaudimų seka, skirtingiems veiksmams sugaištas laikas ar veiksmų šablonai, pasirodė informatyvios skaitmeninėse mokymosi aplinkose ir edukaciniuose žaidimuose (Rowe et al., 2021; Liu, 2024).

Interaktyvūs „Bebro“ uždaviniai generuoja didelius tokių elgsenos duomenų kiekius. Nepaisant to, duomenys retai naudojami vertinimo tikslais. Todėl didžioji dalis informacijos apie tai, kaip mokiniai sprendžia uždavinius, lieka nepanaudota, o vertinimas suteikia tik ribotą vaizdą apie jų informatinio mąstymo kompetenciją. Pastebimas vertinimo metodų, kurie gali sistemingai ir automatizuotai naudoti sprendimo proceso duomenis, poreikis. Šiame darbe informatinis mąstymas vertinamas kaip visuma, akcentuojant sprendimo proceso analizę, o ne atskirus gebėjimus. Toks metodas galėtų pakeisti vertinimą iš dvejetainio „teisingas / neteisingas“ į procesą atspindintį informatinio mąstymo kompetencijos vertinimą.

Tyrimo kryptis

Šis tyrimas skirtas automatizuoto, procesais orientuoto informatinio mąstymo vertinimo metodui modeliuoti. Siūloma sistema yra ne visiškai įgyvendintas įrankis, o konceptualus modelis, išbandytas siekiant įrodyti jo įgyvendinamumą. Užuot vertindamas tik galutinių atsakymų teisingumą, modelis akcentuoja sprendimų procesų analizę interaktyviuose informatinio mąstymo uždaviniuose.

Šiame tyrime naudotas duomenų rinkinys parengtas remiantis tikrais konkurso „Bebras“ sprendimo proceso duomenimis. Interaktyvūs uždaviniai specialiai pritaikyti, kad būtų galima įrašyti sprendimo elgsenos žymes, tad siekiant lyginti skirtingus uždavinius reikėjo atlikti išankstinį apdorojimą ir suvienodinti duomenis.

Transformuojant elgsenos žymes, pvz., paspaudimų sekas, sprendimo trukmę, veiksmų skaičių ir galutinę būseną, į struktūrizuotus duomenų rinkinius ir taikant mašininio mokymosi technikas klasterizavimui bei neuroninių tinklų modelius prižiūrimam klasifikavimui, tyrime siekiama atskleisti reikšmingus uždavinių sprendimo strategijų modelius. Klasterizuojami yra interaktyvaus uždavinio sprendimo bandymai, aprašomi bendraisiais uždavinio ir atskirų uždavinio objektų sąveikos požymiais. Tyrimas pabrėžia, kad informatinis

mąstymas gali būti vertinamas išsamiau, kai dėmesys perkeliamas nuo rezultatų prie procesų, o siūlomas modelis padėtų vykdyti išplėstą ir automatizuotą informatinio mąstymo vertinimą.

Tyrimo tikslas ir uždaviniai

Disertacijos tikslas – sukurti informatinio mąstymo automatizuoto vertinimo modelį, pagrįstą interaktyvių uždavinių sprendimo proceso duomenimis.

Uždaviniai:

1. Išanalizuoti ir susisteminti esamas informatinio mąstymo vertinimo metodikas bei išskirti jų ribotumus.
2. Atlikus bandomąjį tyrimą, paremtą interaktyvių uždavinių sprendimo elgsenos duomenimis, išanalizuoti, susisteminti ir empiriškai ištirti mokymosi analitikos ir mašininio mokymosi metodus, taikomus informatinio mąstymo vertinimui,
3. Sukurti automatizuotą informatinio mąstymo vertinimo modelį, pagrįstą sprendimo proceso elgsenos duomenų analize, taikant klasterizavimo ir klasifikavimo metodus.
4. Empiriškai įvertinti siūlomą informatinio mąstymo vertinimo modelį, naudojant interaktyvių uždavinių duomenis ir įvertinant jo veikimą bei pritaikomumą.

Tyrimo metodai

Šis tyrimas paremtas elgsenos duomenų, sukauptų sprendžiant interaktyvius informatinio mąstymo uždavinius duomenų analize. Tyrimo metodika apima literatūros analizę, mokymosi analitiką ir mašininio mokymosi metodus.

Siekiant nustatyti esamus informatinio mąstymo vertinimo metodų trūkumus ir pagrįsti procesais orientuoto vertinimo metodo poreikį, atlikta esamų metodų apžvalga. Surinkti elgsenos duomenys, gauti sprendžiant interaktyvius „Bebro“ uždavinius, ir paversti tolesnei analizei tinkamais struktūrizuotais duomenų rinkiniais.

Siekiant ištirti įvairias sprendimo strategijas, atliktas bandomasis tyrimas, taikyti klasterizavimo metodai elgsenos duomenims. Analizė leido

nustatyti pasikartojančius mokinių uždavinių sprendimo elgsenos modelius. Siekiant palengvinti automatizuotą sprendimų procesų vertinimą, per bandomąjį tyrimą sukauptos įžvalgos panaudotos bendro vertinimo modelio struktūrai, kuri sujungia klasterizuojant sudarytas sprendimų grupes su prižiūrimu neuroninio tinklo klasifikavimu, apibrėžti.

Siūlomas modelis išbandytas naudojant realius uždavinių sprendimo duomenis. Be to, atlikta ekspertų apklausa, siekiant patvirtinti, ar išskirtos uždavinių sprendimų grupės ir jų vertinimo rezultatai yra prasmingi ir suprantami edukaciniu požiūriu.

Mokslinis naujumas

Šio tyrimo naujumas orientuotas į sprendimo proceso modelio, skirto vertinti informatinį mąstymą, atliekant interaktyvius informatinio mąstymo uždavinius, sukūrimu. Pirmą kartą įvairių interaktyvių konkurso „Bebras“ uždavinių sprendimų proceso duomenys pertvarkyti į vientisą žymėjimų ir požymių atvaizdavimą, leidžiantį sistemingai lyginti ir analizuoti uždavinių sprendimo strategijas. Svarbus tyrimo rezultatas yra tas, kad taikant klasterizavimo metodus (pvz., k-vidurkių metodą bandomajame etape ir „Affinity Propagation“ pagrindiniuose eksperimentuose) sėkmingai išskirtos loginės sprendimų grupės, parodant, kad mokinių uždavinių sprendimo metodai gali būti prasmingai atskirti be išankstinio ženklinimo.

Modelis taip pat pristato dvišakio neuroninio tinklo architektūrą klasifikavimui, derindamas eiliškumo požymius (paspaudimų sekas) su statiškais požymiais (sprendimo trukmė, paspaudimų skaičius ir galutinis dvejetainis atsakymas). Ši integracija rodo, kad sprendžiant „Bebro“ interaktyvius uždavinius gauti elgsenos duomenys gali būti tiek klasterizuojami, tiek automatiškai klasifikuojami. Taip tyrimas išplečia informatinio mąstymo vertinimo apimtį: matyti, kad sprendimo proceso duomenys ir algoritminis modeliavimas gali būti veiksmingai pritaikyti švietimo uždaviniams, kurie tradiciškai buvo vertinami tik pagal galutinį rezultatą.

Tyrimo praktinė vertė

Šio tyrimo praktinė vertė yra tai, kad jis parodo, jog automatizuotas, procesais pagrįstas informatinio mąstymo vertinimas, remiantis interaktyvių Bebro

uždavinių sprendimų elgsenos duomenimis, gali suteikti reikšmingų įžvalgų. Analizuodamas elgsenos žymes, pavyzdžiui, paspaudimų sekas, sprendimų trukmę ir galutinius rezultatus, modelis leidžia klasifikuoti sprendimų strategijas ir išskirti skirtingus požiūrius, kuriuos mokiniai taiko sprendžiant uždavinius. Tai padeda pedagogams neapsiriboti rezultatu „teisinga / neteisinga“ ir geriau suprasti mokinių mąstymą, sykiu teikti tikslesnį grįžtamąjį ryšį ir pagalbą.

Ta pati metodika taip pat galėtų būti taikoma ne tik švietimo kontekste, bet ir profesinio mokymo ar žmogiškųjų išteklių srityse, nes vertinti, kaip asmenys mąsto spręsdami sudėtingus uždavinius, yra lygiai taip pat svarbu, kaip vertinti, ar jų galutinis rezultatas yra teisingas. Taigi tyrimas atveria galimybes praktiniam pritaikymui, derinant elgsenos duomenų analizę su automatizuotomis vertinimo technikomis.

Ginamieji teiginiai

1. Interaktyvių informatinio mąstymo uždavinių sprendimo procesų požymiai leidžia sprendimo elgsenos duomenis automatizuotai grupuoti į interpretuojamas sprendimų grupes, atspindinčias skirtingas uždavinių sprendimo strategijas.
2. Sprendimo procesų duomenų grupavimas atskleidžia tuos pačius sprendimo elgsenos dėsningumus skirtinguose interaktyviuose uždaviniuose, priklausančiuose tai pačiai interaktyvumo grupei. Šie dėsningumai yra tinkami suvienodinimui į bendrą žymėjimo schemą, taikomą visiems tai pačiai interaktyvumo grupei priklausantiems uždaviniams.
3. Siūlomas automatizuotas vertinimo modelis, pagrįstas suvienodintu sprendimų procesų pateikimu, leidžia pakankamai tiksliai klasifikuoti sprendimus.

Publikacijos ir konferencijos

Straipsniai tarptautiniuose moksliniuose žurnaluose, įtraukti į *Clarivate Analytics Web of Science (CA WoS)* duomenų bazę su citavimo indeksu:

1. Masiulionytė-Dagienė, V., & Jevsikova, T. (2025). Towards a process-oriented assessment of computational thinking: behavioural data and machine learning approach. *Interactive Learning Environments*, 1–15. https://doi.org/10.1080/10494820.2025.2567601

2. Bilbao, J., Bravo, E., Garcia, O., Rebollar, C., Laakso, M.-J., Kaarto, H., Lehtonen, D., Parviainen, M., Jankauskienė, A., Pears, A., Guven, I., Gulbahar, Ozturk, T., Yenigun N.T., Pluhar, Z., Sarmasagi, P., Rumbus, A., Dagienė, V., & Masiulionytė-Dagienė, V. (2025). Analytical Methods and Tools for Evaluating the Development of Computational Thinking Abilities. *International Journal of Education and Information Technologies,* 19, 53-61. https://doi.org/10.46300/9109.2025.19.6

3. Bilbao, J., Bravo, E., Garcia, O., Rebollar, C., Laakso, M.-J., Kaarto, H., Lehtonen, D., Parviainen, M., Jankauskienė, A., Pears, A., Guven, I., Gulbahar, Y., Oncul, F. O., Yenigun, N. T., Pluhar, Z., Sarmasagi, P., Dagienė, V., & Masiulionytė-Dagienė, V. (2024). Introducing computational thinking and algebraic thinking in the European educational systems. In *International Journal of Education and Information technologies*. https://doi.org/10.46300/9109.2024.18.2

4. Kampylis, P., Dagienė, V., Bocconi, S., Chioccariello, A., Engelhardt, K., Stupurienė, G., Masiulionytė-Dagienė, V., Jasutė, E., Malagoli, C., Horvath, M., & Earp, J. (2023). Integrating Computational Thinking into Primary and Lower Secondary Education: A Systematic Review. *Educational Technology & Society*, 26(2), 99-117. https://doi.org/10.30191/ETS.202304_26(2).0008

Pranešimai tarptautinėse konferencijose:

1. Masiulionytė-Dagienė, V. (2021). „Gamification for developing computational thinking in blended-learning environment: students' motivation and assessment problems". *ISSEP 2021*, Netherlands

2. Masiulionytė-Dagienė, V., and Jevsikova, T. (2022). „Assessing Computational Thinking: The Relation of Different Assessment Instruments and Learning Tools". *15th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP 2022*, Vienna, Austria.

3. Masiulionytė-Dagienė, V. (2023). „Modeling of the System for Computational Thinking Automatic Assessment". *28th annual ACM conference on Innovation and Technology in Computer Science Education (ITiCSE), ITiCSE 2023*, Turku, Finland.

4. Masiulionytė-Dagienė, V., & Jevsikova, T. (2025, June). „Behavioural Data-Driven Approach for Computational Thinking Automatic Assessment Using Interactive Bebras Challenge Tasks". In *Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education* V. 2.

Pranešimai nacionalinėse konferencijose:

Masiulionytė-Dagienė V. (2025). „Interaktyvių Bebro užduočių sprendimų duomenų analizė ir vertinimo skalės modelis". *Kompiuterininkų dienos* 2025, Šiauliai, Lietuva.

Kitos publikacijos:

1 Bilbao, J., Bravo, E., Garcia, O., Rebollar, C., Dagienė, V., Masiulionytė-Dagienė, V., Jankauskienė, A., Laakso, M-J., Kaarto, H., Lehtonen, D., Güven, I., Gulbahar, Y., Özdemir Öncül, F., Pluhár, Z., Sarmasági, P., Pears, A. (2023) Working the basis of computational thinking: definition and skills // *ICERI2023 Proceedings: 16th annual international conference of education, research and innovation*, Seville, Spain. 13-15 November, 2023. València: IATED Academy, 2023. ISBN 9788409559428. p. 8410-8416. (ICERI Proceedings, ISSN 2340-1095). https://doi.org/10.21125/iceri.2023.2151.

2 Bilbao, J.; Bravo, E.; Garcia, O.; Rebollar, C.; Dagienė, Valentina; Masiulionytė-Dagienė, Vaida; Jankauskienė, A.; Laakso, M.J.; Kaarto, H.; Lehtonen, D.; Parviainen, M.; Güven, I.; Gulbahar, Y.; Öztürk, T.; Özdemir Öncül, F.; Tan Yenigün, N.; Pluhár, Z.; Sarmasági, P.; Pears, A. Computational thinking and problem solving in PISA era // *INTED2024: 18th international technology, education and development conference*, 4-6 March, 2024, Valencia, Spain : conference proceedings / edited by L. Gómez Chova, C. González Martínez, J. Lees. València: IATED Academy, 2024. ISBN 9788409592159. p. 7335-7342. (INTED Proceedings, ISSN 2340-1079). https://doi.org/10.21125/inted.2024.1922.

3 Bocconi, S., Chioccariello, A., Kampylis, P., Dagienė, V., Wastiau, P., Engelhardt, K., Earp, Jeffrey; Horvath, Milena; Jasutė, Eglė; Malagoli, Chiara; Masiulionytė-Dagienė, V., Stupurienė, G. (2022) Reviewing computational thinking in compulsory education: state of play and practices from computing education. Luxembourg: Publications Office of the European Union, 2022. 138 p. ISBN 9789276472087. https://doi.org/10.2760/126955.

Disertacijos struktūra

Disertacija suskirstyta į penkis skyrius, apimančius teorinių pagrindą ir siūlomo elgsenos duomenimis pagrįsto vertinimo modelio, skirto interaktyviems informatinio mąstymo uždaviniams, empirinį patvirtinimą. Įvadiniame skyriuje pateikiamas tyrimo kontekstas, suformuluojami tikslai ir uždaviniai, pristatomas mokslinis tyrimo naujumas ir ginamieji teiginiai. 1 skyriuje pateikiama esamų informatinio mąstymo vertinimo ir mokymosi analizės metodų kritinė apžvalga, nurodant metodologinius vertinimo apribojimus ir pabrėžiant bendros elgsenos analizės poreikį įvairiuose interaktyviuose uždaviniuose.

2 skyriuje pateikiamas bandomasis tyrimas, įrodantis, kad galima išskirti skirtingus uždavinio sprendimo variantus naudojant uždavinio sprendimo elgsenos duomenis ir klasterizavimo metodus. Remiantis šiomis žiniomis, 3 skyriuje formalizuojamas bendras vertinimo modelis, apibrėžiant sąveikos struktūras, uždavinių grupes, bendras savybių ir žymių erdves bei nustatant darbo eigą, skirtą seniems sprendimams klasterizuoti ir naujiems sprendimams klasifikuoti. 4 skyriuje pateikiami empiriniai tyrimai su kitais „Bebro“ uždaviniais, kuriuose vertinama klasterizavimo kokybė, klasifikatoriaus veikimas ir apibendrinamumas, papildomai pateikiama ekspertų apklausa, suvokiamumui įvertinti. Darbo pabaigoje pateikiama išvadų, indėlio, apribojimų ir ateities tyrimų krypčių sintezė.

S.1. Literatūros apžvalga

Per pastarąjį dešimtmetį informatinio mąstymo (IM, angl. *computational thinking*) vertinimo tyrimai smarkiai išsiplėtė – IM vis plačiau pripažįstamas esmine XXI a. kompetencija ir yra įtraukiama į bendrojo ugdymo bei aukštojo mokslo programas. Literatūros analizė leidžia išskirti kelias dominuojančias kryptis: testais ir psichometriniais instrumentais paremtą IM vertinimą, į programavimo artefaktus, žaidimais ir interaktyviomis aplinkomis grindžiamą procesinį vertinimą orientuotus automatizuotus analizės įrankius, IM vertinimo konceptualių karkasų ir sisteminių apžvalgų plėtrą ir mokymosi analitikos bei dirbtiniu intelektu grįstas asmenines vertinimo sistemas.

Didelė dalis darbų remiasi tradiciniais IM testais – uždarų ar atvirų klausimų rinkiniais. Vienas iš plačiausiai validuotų instrumentų yra *Computational Thinking test* (CTt), kurį pasiūlė Román-González ir bendraautoriai (2017). Šis testas leidžia vertinti pagrindines IM konstrukcijas – abstrakciją, dekompoziciją, algoritminį mąstymą ir derinimą – nereikalaujant mokinių programavimo įgūdžių, ir plačiai taikomas tyrimuose apie IM raidą bei intervencijų veiksmingumą (pvz., Guggemos, 2021). CTt pritaikytas ir jaunesniems mokiniams – BCTt versija skirta pradinei mokyklai (Zapata-Cáceres et al., 2021), o CCTt – pradinių klasių mokiniams pateikia dar detalesnę diagnostinę informaciją (El-Hamamsy et al., 2025). Kiti instrumentai orientuojasi į programavimo su IM susijusius gebėjimus, pavyzdžiui, *Test for Measuring Basic Programming Abilities* (Mühling et al., 2015) ar *Commutative Assessment* (Weintrop & Wilensky, 2015, 2019). Be to, kuriamos IM skalės ir klausimynai, vertinantys nuostatas bei polinkius (Korkmaz et al., 2017; Yağcı, 2019). Vis dėlto naujesniuose darbuose pabrėžiama, kad vien testais paremtos priemonės dažnai nepakankamai atskleidžia strateginį mąstymą ir metakognityvinius procesus, kurie yra svarbūs IM praktikai (Shute & Rahimi, 2023; Kovanović et al., 2024).

Kita gausi tyrimų kryptis IM vertinimą sieja su programavimo artefaktų – dažniausiai „Scratch“ ar panašių blokinio programavimo projektų – analize. Įrankis *Dr. Scratch* (Moreno-León & Robles, 2015; Moreno-León et al., 2015) automatiškai analizuoja mokinių programas ir iš jų sprendžia apie IM gebėjimus (abstrakciją, loginį mąstymą, duomenų išraišką ir kt.). Jis plačiai taikomas vertinant mokinių pažangą ar ugdymo intervencijų poveikį (pvz., Troiano et al., 2019; Fagerlund et al., 2020; Wei et al., 2021). Atsakant į metodologinius „Dr. Scratch“ ribotumus (blokų atpažinimo netikslumus, veiksmingumo problemas), sukurta alternatyvių įrankių, tokių kaip *Scratch*

Analysis Tool (SAT) (Chang et al., 2018). Artefaktų analizė taip pat taikoma modeliavimo ir simuliacijų aplinkose (Grgurina et al., 2018; Metcalf et al., 2023). Ši kryptis atskleidžia didelį potencialą automatizuotai vertinti IM iš mokinių sukurtų skaitmeninių produktų, tačiau dažniausiai apsiriboja programavimo kontekstu.

Vis daugiau dėmesio skiriama neprogramavimo kontekstams, kuriuose IM kompetencija atsiskleidžiama per robotiką, veiklas be kompiuterio, fizinius artefaktus ar vizualinius produktus. Pavyzdžiui, Grover (2011) nagrinėja robotikos užduotis, Tonbuloğlu ir Tonbuloğlu (2019) – be kompiuterio vykdomas programavimo veiklas, Statter ir Armoni (2020) IM vertinti jungia testus, stebėjimus ir kokybinius užrašus, o Waite et al. (2020) siūlo mokinių piešinių analizę kaip ankstyvojo projektavimo mąstymo indikatorius. Šie darbai parodo, kad IM kompetencijos pasireiškia ne tik programavimo užduotyse, o daug platesniame veiklų spektre.

Sparčiai auga tyrimų sritis, nagrinėjanti žaidimais grįstą IM mokymą ir vertinimą. Žaidimas *Zoombinis* (Asbell-Clarke et al., 2021; Rowe et al., 2021) pasitelkiamas implicitiniam IM ugdymui ir vertinimui: analizuojami žaidimo žurnalai – veiksmų seka, klaidų modeliai, sunkumo progresija, strategijos. Kitos aplinkos – mokinių kuriami žaidimai, STEM srities veiklos, modeliavimo simuliacijos – taip pat generuoja elgsenos duomenis, leidžiančius vertinti IM procesiniu požiūriu (Leonard et al., 2021; Grgurina et al., 2018). Naujesniuose darbuose į IM žaidybines aplinkas įtraukiamas mašininis mokymąsis ir kiti dirbtinio intelekto metodai, klasifikuojant sprendimo strategijas ar teikiant adaptyvų grįžtamąjį ryšį (Beltrán-Martín et al., 2023; Chaim et al., 2024).

Ypatingą vietą IM vertinimo literatūroje užima tarptautinis konkursas „Bebras“ – vienas iš neprogramavimo IM vertinimo išteklių (Izu et al., 2017; Djambong et al., 2018; Labusch & Eickelmann, 2020). „Bebro“ uždaviniai leidžia patikimai vertinti aukštesnio lygmens analitinį mąstymą įvairiose šalyse, tačiau įprastai vertinami dvejetainiu būdu – tik pagal galutinį atsakymą. Tai reiškia, kad sprendimo procesas ir tarpiniai veiksmai visiškai neturi įtakos vertinimui, nors naujausiuose darbuose pabrėžiamas poreikis IM laikyti būtent problemų sprendimo procesu, o ne vien rezultatu (Lockwood & Mooney, 2017; Zhang et al., 2020; Corrales-Álvarez et al., 2024; Belletini et al., 2023).

Apibendrinant galima teigti, kad IM vertinime šiuo metu vis dar dominuoja testai ir programavimo artefaktų analizė, o procesinės, sprendimo eigą fiksuojančios prieigos yra nepakankamai išplėtotos. Interaktyvios

užduotys ir žaidimai natūraliai generuoja gausius elgsenos duomenis, tačiau dauguma sistemų išnaudoja tik galutinį rezultatą, o ne sprendimo seką ar strategijas. „Bebro“ uždaviniai tipinis šios spragos pavyzdys.

Kita svarbi teorinė atrama IM vertinimui yra mokymosi analitika (MA) ir edukacinių duomenų tyryba (EDT). SoLAR apibrėžimu, MA – tai duomenų apie besimokančiuosius ir jų kontekstą matavimas, rinkimas, analizė ir pateikimas siekiant geriau suprasti mokymąsi ir optimizuoti mokymosi aplinkas. EDT apibrėžiama kaip statistikos, mašininio mokymosi ir duomenų gavybos metodų taikymas edukaciniams duomenims (Romero & Ventura, 2010). Nors abi sritys susijusios, Siemens‘as ir Baker‘is (2012) išskiria kelis skirtumus: EDT labiau orientuota į automatinį atradimą ir prisitaikančias sistemas, o MA – į žmogaus sprendimus ir praktinį pritaikymą; EDT dažniau remiasi redukciniais modeliais, MA – holistiniu, sisteminiu požiūriu. Paskesni darbai papildomai išskiria susijusias sąvokas – akademinę ir institucinę analitiką, mokymosi analitiką dėstymo lygiu, duomenimis grindžiamą ugdymą, didžiuosius duomenis švietime ar edukacinius duomenų mokslus (Romero & Ventura, 2020).

Mokymosi analitika plačiai tiriama kaip priemonė gerinti mokymosi procesą ir rezultatus. Apžvalgos rodo, kad analitiniai sprendimai gali prisidėti prie žinių įsisąmoninimo, įgūdžių plėtojimo ir kognityvinių pasiekimų, taip pat prie mokymosi palaikymo ir studijų tęstinumo (Ferguson & Clow, 2017; Viberg et al., 2018). Naujesniuose darbuose pabrėžiama, kad MA poveikis vis labiau priklauso nuo grįžtamojo ryšio dizaino: multimodalinė MA, derinanti veiksmų žurnalus su sąveikos ar jutiklių duomenimis, sudaro prielaidas teikti išsamesnį, strategijomis grįstą grįžtamąjį ryšį (Yan et al., 2024). Sykiu žmogui palankios MA ir DI sistemos tyrimai (Alfredo et al., 2024) pabrėžia skaidrumo, etikos ir naudotojų įgalinimo svarbą – analitinės priemonės turi būti ne tik techniškai tikslios, bet ir suprantamos, kontroliuojamos mokytojų bei mokinių.

Metodiniu požiūriu MA remiasi įvairiais skaičiavimo metodais. Merceron‘as (2015) išskiria prognozavimo, klasterizavimo, ryšių paieškos (asociacijų, koreliacijų, sekų), vizualizacijų ir modeliais grįsto atradimo užduotis. Nors anksčiau dominavo prognozavimas, pastaraisiais metais išaugo susidomėjimas interpretacinėmis, ryšių ir sekų analizę taikančiomis prieigomis bei vizualinėmis analitinėmis priemonėmis (Viberg et al., 2018; Molenaar et al., 2020; Van Leeuwen & Rummel, 2020). Naujausiuose darbuose šios kategorijos išplečiamos į multimodalines ir DI stiprinamas aplinkas: Yan et al. (2024) ir Mohammadi et al. (2025) rodo, kad prognozavimas, klasterizavimas ir sekų analizė vis dažniau taikomi

sudėtingiems elgsenos ir multimodaliniams duomenims, o DI naudojamas visoje analitinėje grandinėje – nuo požymių gavimo iki intervencijų planavimo.

Šie metodiniai pasiekimai yra tiesiogiai aktualūs IM vertinimui. Klasterizavimo metodai leidžia identifikuoti tipinius sprendimo strategijų profilius – taip atsakant į literatūroje išsakytą poreikį vertinti ne tik rezultatą, bet ir IM praktikas (Corrales-Álvarez et al., 2024; Luo & Zhang, 2024). Sekų analizė ir sekų gavyba suteikia priemonių tirti sprendimo eigą, o tai ypač svarbu interaktyviems „Bebro“ uždaviniams, kuriuose sprendimo elgsena yra nuosekli ir tyrinėjamojo pobūdžio. Multimodaliniai modeliai tinka ir kitose IM studijose taikomoms žaidimų ar programavimo aplinkų žurnalų analizėms (Zoombinis, Dr. Scratch ir pan.).

Apibendrinant literatūros apžvalgą, išryškėja trys panašios, bet dar nepakankamai susietos tyrimų kryptys: IM konceptualizavimas ir matavimo instrumentų kūrimas, konkrečių IM vertinimo įrankių – testų, žaidimų, programavimo analizės priemonių – plėtra ir mokymosi analitikos bei kitų duomenimis grįstų metodų taikymas mokymuisi tirti. Nors pažanga didelė, sritį vis dar riboja metodų fragmentacija, nepakankamas procesinių duomenų panaudojimas ir menkas teorinis suderinamumas. Šiuo metu nėra plačiai validuoto, domeno atžvilgiu nepriklausomo, automatizuoto IM vertinimo metodo, kuris: remtųsi sprendimo procesu neprogramavimo užduotyse, naudotų mokymosi analitikos ir mašininio mokymosi metodus sprendimo strategijoms identifikuoti ir būtų pedagogiškai interpretuojamas ir praktiškai naudingas mokytojams.

Šią spragą siekiama užpildyti tolesniuose disertacijos skyriuose pristatomu empiriniu tyrimu, kuriame: interaktyvūs Bebro uždaviniai naudojami kaip didelio masto, domeno atžvilgiu neutrali IM vertinimo aplinka; renkami ir modeliuojami detalią sprendimo elgseną atspindintys duomenys (paspaudimų sekos, veiksmų keliai, laiko charakteristikos); taikomi mokymosi analitikos ir klasterizavimo metodai tipinėms sprendimo strategijoms identifikuoti ir susieti su pasiekimais; ir nagrinėjama, kaip tokie procesiniai rodikliai galėtų papildyti tradicinį, tik galutinio rezultato pagrindu grindžiamą IM vertinimą.

S.2. Bandomasis tyrimas ir pradinė elgsenos analizė

Informatiniam mąstymui (IM) vertinti neretai taikomi popieriniai ir skaitmeniniai testai, tačiau dažniausiai vertinamas tik galutinis atsakymas. Kitaip nei matematikoje, kur pripažįstami tarpiniai sprendimo žingsniai ir dalinis teisingumas, IM testuose problemos sprendimo eiga įprastai yra „nematoma“. Programavimo uždaviniai išimtis, nes procesas atsispindi kode, tačiau programavimas apima tik dalį IM ir nėra vienodai prieinamas visiems besimokantiesiems. Todėl vis didesnį potencialą IM vertinimui turi interaktyvūs problemų sprendimo uždaviniai, kurie leidžia fiksuoti sprendimo eigą ir elgseną.

Bandomajame tyrime pasitelkiama interaktyvi 2022 m. tarptautinio konkurso „Bebras“ uždavinio „Coloring Page“ versija. „Bebro“ uždaviniai tyrime naudojami pagal „Creative Commons“ licenciją. Uždavinyje yra 12 sričių, kurias reikia nuspalvinti trimis spalvomis (geltona, mėlyna, žalia), laikantis vienos taisyklės: gretimos sritys negali būti tos pačios spalvos. Uždavinys formaliai atitinka klasikinį grafų spalvinimo uždavinį ($G=(V,E)$, $c:V\rightarrow\{yellow, green, blue\}$, *su apribojimu* $c(vi)\neq c(vj)$ *visoms* $(vi,vj)\in E$). Nors uždavinys vizualiai paprastas (S.2.1. pav.), jis yra netrivialios kombinatorinės struktūros: be apribojimų galimų priskyrimų erdvė būtų 3^{12}, tad teisingam sprendimui reikalingas nuoseklus apribojimų valdymas, konflikto aptikimas ir sprendimo koregavimas. Tikėtina, kad skirtingi sprendėjai taikys nevienodas strategijas: sistemingą eiliškumą, apribojimais grįstą planavimą, grįžimą atgal (*backtracking*), bandymų ir klaidų elgseną.

S.2.1 pav.: Dar neišspręstas ir išspręstas uždavinys

Interaktyvi sistemos versija registravo kiekvieną vartotojo veiksmą (paspaudimus), todėl buvo galima atkurti sprendimo kelią. Buvo renkami 7–12 klasių mokinių ir bakalauro studentų duomenys; demografiniai kintamieji šiame etape neanalizuoti, nes tikslas buvo elgsenos struktūra, o ne grupių lyginimas. Visi duomenys anonimizuoti, nerenkant asmens identifikatorių ir nepaliekant galimybės susieti bandymą su konkrečiu asmeniu.

Kiekvienas paspaudimas įrašytas kaip įvykis

$$e_k = (o_k, c_k, t_k),$$

kur o_k – objekto (srities) identifikatorius, c_k – spalvos reikšmė po paspaudimo, t_k – laiko žyma. Sprendimo bandymas sudaro įvykių seką:

$$E = (e_1, e_2, \dots, e_n).$$

Spalvų pasirinkimas įgyvendintas ciklišk ai:

balta(0)→geltona(1)→mėlyna(2)→žalia(3)→balta(0),

todėl keli paspaudimai iš eilės ant to paties objekto dažnai reiškia norimos būsenos pasiekimą, o ne naują sprendimo žingsnį. Pradiniame etape svarstyta rinkti ir „hover“ įvykius, tačiau jie sugeneravo didelį, triukšmingą ir sunkiai interpretuojamą duomenų srautą, tad galutiniam rinkiniui palikti tik paspaudimai.

Iš pradžių surinkta apie 700 sprendimo bandymų, tačiau po valymo (tušti, pasikartojantys įrašai, ekstremalūs atvejai su neįprastai mažu ar dideliu paspaudimų skaičiumi) tolesnei analizei palikti 336 kokybiški sprendimai. Pradiniai *JSON* formato įrašai transformuoti į struktūrinį duomenų rinkinį, išskiriant objekto lygmens ir globalius požymius: paspaudimų skaičių kiekvienam objektui, galutinę spalvą, sprendimo trukmę ir bendrą paspaudimų skaičių. Siekiant modeliuoti prasmingą veiksmų eigą, iš paspaudimų sekos pašalinti iš eilės pasikartojantys paspaudimai ant to paties objekto, sudarant redukuotą objektų pasirinkimų seką. Objekto dalyvavimas sekoje užkoduotas dvejetainiu vektoriumi pagal padėtis ir papildomai suspaustas į dešimtainį kodą.

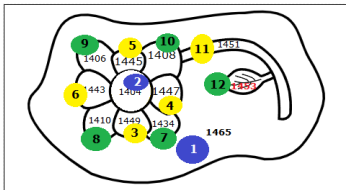
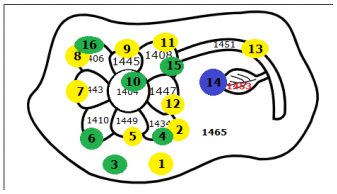
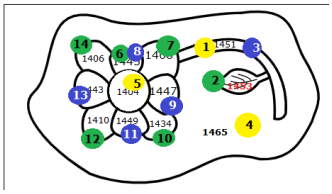
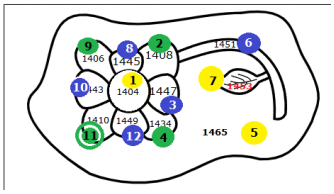
Paruošti duomenys panaudoti tiriamajam klasterizavimui, siekiant nustatyti, ar sprendimo elgsena turi atpažįstamus tipus. Kiekvienas bandymas aprašytas požymių vektoriumi, apimančiu globalias charakteristikas (laiką, paspaudimų kiekį) ir objekto lygmens elgseną (paspaudimai, galutinės spalvos, pirmo paspaudimo padėtis, grįžimai prie objektų). Dėl skirtingų mastelių visi požymiai normalizuoti *min–max* metodu į [0;1] intervalą.

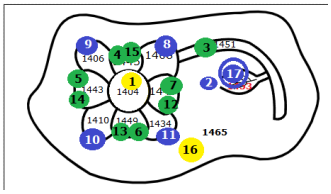
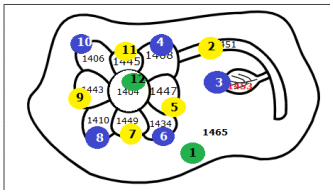
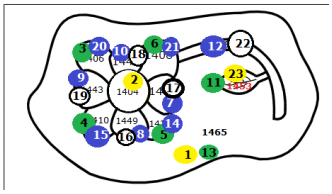
K-vidurkių klasterizavimas taikytas kaip pirminė, tiriamoji priemonė. Lyginti keli požymių rinkiniai (pilnas, tik objekto lygmens, be spalvų, tik paspaudimų skaičiai), ir nustatyta, kad per daug redukuotos išraiškos suformuoja didelius heterogeniškus klasterius bei degeneracinius klasterius su

labai mažu sprendimų skaičiumi. Esminė išvada – sekos požymiai (objektų lankymo tvarka, grįžimai) yra būtini norint atskirti sistemingą sprendimo strategiją nuo bandymų ir klaidų elgsenos. Interpretuojamiausia pusiausvyra tarp detalumo ir aiškumo pasiekta su 8 klasteriais.

Klasterių interpretacijai iš kiekvieno klasterio atrinkti būdingi pavyzdžiai (sprendimai arčiausiai centroido). Naudojant sprendimų identifikatorius, buvo galima grįžti prie pirminių sprendimų duomenų įrašų ir atkurti realias sprendimo sekas. Rekonstrukcija leido pergrupuoti klasterius į prasmingus elgsenos tipus pagal taisyklės taikymo nuoseklumą, klaidų taisymą, klaidų likutį pabaigoje ir uždavinio supratimo lygį. Išryškėjo, kad sprendimų erdvė yra gerokai turtingesnė už dvejetainį „teisinga / neteisinga“: identifikuojama apie 8–10 tipinių profilių, apimančių sistemingą teisingą sprendimą, teisingą sprendimą su greitu klaidos pataisymu, teisingą sprendimą po vėlyvo taisymo, beveik teisingą sprendimą su viena nepataisyta klaida, dalinį supratimą, perteklinį taisymą ir visišką taisyklės nesupratimą. S.2.1 lentelėje pateikiami, atkurti sprendimai iš pirmų keturių klasterių:

S.2.1 lentelė: Pateikiami pirmų keturių klasterių atkurti sprendimai

Klasterio numeris	Teisingas sprendimas	Klaidingas sprendimas
0		
1		

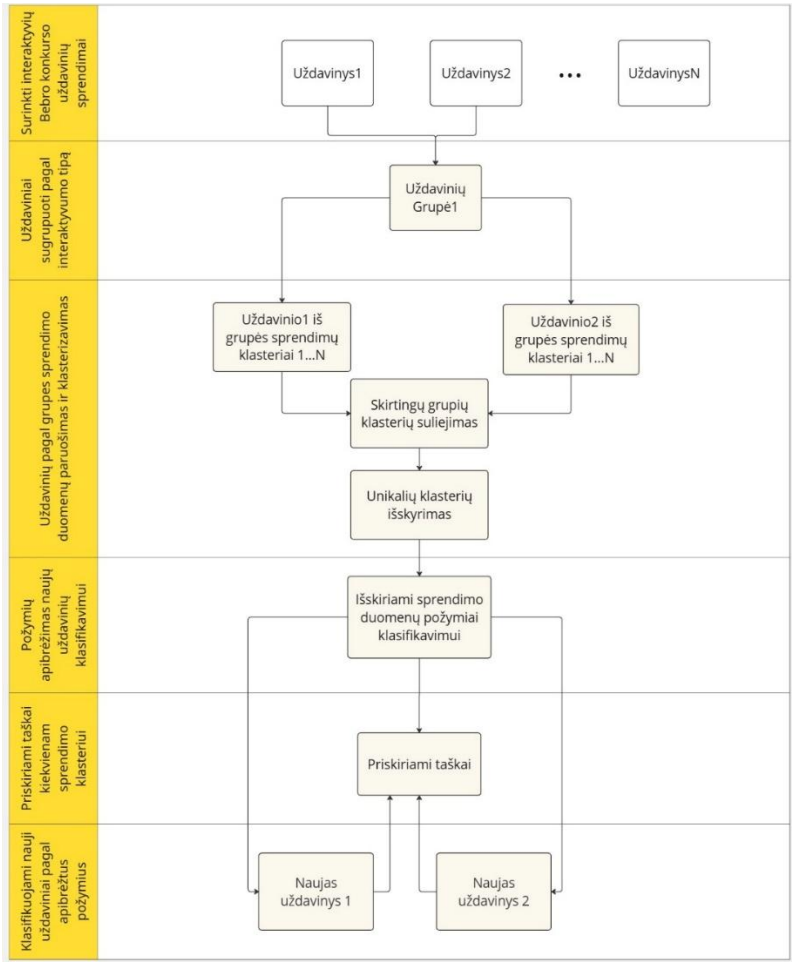
Klasterio numeris	Teisingas sprendimas	Klaidingas sprendimas
2		NEBUVO NETEISINGŲ SPRENDIMŲ ŠIAME KLASTERYJE
3		

Siekiant patikrinti, ar gauti klasteriai nėra tik k-vidurkių metodo ir iš anksto parinkto klasterių skaičiaus pasekmė, papildomai taikytas *Affinity Propagation* algoritmas. Jis automatiškai parenka klasterių skaičių ir naudoja realius sprendimus kaip egzempliorius (angl. *exemplars*). Nustatyta, kad mažesnis slopinimo koeficientas (pvz., 0,7) sukuria pernelyg daug klasterių (apie 60), o padidinus slopinimą iki 0,9 gaunama stabili struktūra (apie 9 klasteriai), artima k-vidurkių rezultatui. Tai patvirtino, kad sprendimo elgsenos tipai kyla iš duomenų struktūros ir yra gana stabilūs.

Skyriaus išvada: net vieno interaktyvaus uždavinio paspaudimų žurnalai turi pakankamai informacijos, kad būtų galima diferencijuoti sprendimo strategijas ir klaidų taisymo dinamiką, t.y. sudaryti prielaidas procesiniam, diagnostiniam IM vertinimui. Bandomojo tyrimo rezultatai pagrindžia tolesnį bendresnio modelio kūrimą ir validaciją su platesniu interaktyvių „Bebro“ uždavinių rinkiniu.

S.3. Teorinis IM vertinimo modelis

Šiame skyriuje pateikiamas teorinis automatizuoto informatinio mąstymo (IM) vertinimo modelis, skirtas interaktyviems Bebro uždaviniams vertinti. Modelis remiasi atlikta bandomojo sprendimo proceso analize ir siekia transformuoti detalius veiksmų lygmens duomenis į interpretuojamas elgsenos kategorijas, leidžiančias vertinti ne tik galutinio atsakymo teisingumą, bet ir sprendimo proceso kokybę. Bendras modelio veikimo principas pavaizduotas S.3.1. paveiksle.



S.3.1 pav.: Teorinis automatizuotas IM interaktyvių uždavinių vertinimo modelis

Pirmasis modelio etapas – interaktyvių uždavinių konceptualus grupavimas. Interaktyvūs IM uždaviniai skiriasi savo sąveikos mechanika:

veiksmų tipais (paspaudimai, perjungimai, vilkimas), objektų struktūra, veiksmų kaupimu ar perrašymu, klaidų taisymo galimybėmis ir dalinio teisingumo atpažinimu. Kad sprendimo elgsenos būtų palyginamos, uždaviniai grupuojami į interaktyvumo tipus, kuriuose sprendimo veiksmai generuoja struktūriškai panašius elgsenos duomenis. Šiame skyriuje grupavimas aprašomas konceptualiai, o išsamus operacinis interaktyvių „Bebro“ uždavinių suskirstymas pateikiamas empirinio tyrimo skyriuje.

Antrajame etape atliekamas senų sprendimų klasterizavimas kiekvienoje interaktyvumo grupėje. Sprendimų žurnalai transformuojami į elgsenos požymių vektorius, apimančius veiksmų skaičių, objektų lygmens veiksmų pasiskirstymą, sprendimo laiką, objektų būsenas ir sąveikos seką. Naudojant *Affinity Propagation* metodą, sprendimai grupuojami į elgsenos klases be iš anksto nustatyto klasterių skaičiaus. Gautos klasės atspindi tipinius sprendimo būdus: nuo visiškai teisingų ir veiksmingų sprendimų iki sprendimų su pataisytomis klaidomis, daliniu supratimu, struktūriniais nesusipratimais ar atsitiktiniu elgesiu. Šios klasės apibrėžia diskrečius elgsenos sprendimo tipus, būdingus konkrečiam uždavinių interaktyvumo tipui.

Trečiasis etapas – naujų uždavinių ir naujų sprendimų klasifikavimas. Prieš automatinį vertinimą naujas interaktyvus uždavinys priskiriamas vienai iš egzistuojančių interaktyvumo grupių pagal jos sąveikos savybes. Priskyrus uždavinį grupei, jai tampa taikomos tos grupės elgsenos klasės ir vertinimo taisyklės. Kiekvienas naujas sprendimas užkoduojamas į požymių vektorių analogiškai seniems duomenims ir palyginamas su elgsenos klasių „pirštų atspaudais“. Sprendimas priskiriamas tai klasei, kurios elgsenos profilis yra artimiausias, taip automatiškai atpažįstant, kokia sprendimo strategija taikyta.

Ketvirtasis etapas – elgsena grįstos vertinimo taisyklės. Kitaip nei tradicinis „Bebro“ vertinimas, kuris remiasi tik dvejetaine teisingumo logika, siūlomas modelis naudoja tęstinę vertinimo skalę [0;1]. Visiškai teisingi sprendimai be klaidų vertinami maksimaliu balu *(p = 1)*. Teisingi sprendimai su ištaisytomis klaidomis vertinami šiek tiek mažesniu balu $(0{,}95 \leq p < 1)$, išlaikant suderinamumą su įprastomis apvalinimo taisyklėmis. Iš dalies teisingi sprendimai, rodantys prasmingą pažangą, vertinami intervale $0 < p \leq 0{,}90$, o konkretūs balai paskirstomi pagal identifikuotų dalinio teisingumo klasių skaičių. Visiškai neteisingi sprendimai, rodantys taisyklės nesupratimą ar atsitiktinį elgesį, vertinami $p = 0$. Taip vertinimas tampa tiesiogiai susietas su sprendimo elgsena ir atspindi IM uždavinių sprendimo kokybę.

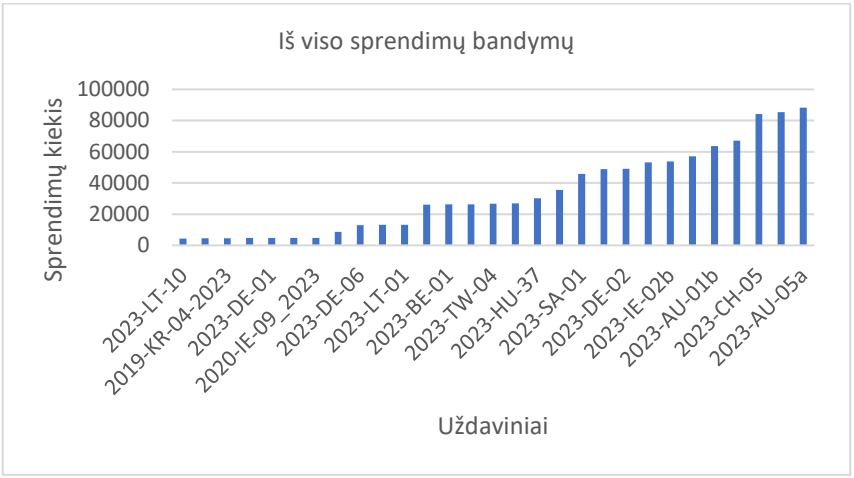
Apibendrinant šiame skyriuje pristatytas teorinis modelis apibrėžia nuoseklų automatizuoto IM vertinimo procesą: nuo uždavinių grupavimo pagal interaktyvumą, per sprendimų elgsenos klasterizavimą, iki naujų sprendimų klasifikavimo ir procesinio vertinimo. Modelis leidžia pereiti nuo dvejetainio rezultatų vertinimo prie interpretuojamo, duomenimis grįsto sprendimo proceso įvertinimo. Kitame skyriuje šis modelis taikomas realiems „Bebro“ duomenims ir empiriškai patikrinamas praktiniame kontekste.

S.4. Vertinimo modelio eksperimentinė validacija

Šiame skyriuje pristatomas teorinio informatinio mąstymo vertinimo modelio empirinio patikrinimo tyrimas, kuriame modelis taikomas realiems interaktyvių konkurso „Bebras“ uždavinių sprendimų duomenims. Pagrindinis tyrimo tikslas – įvertinti, ar elgsena grįstas vertinimo modelis leidžia patikimai identifikuoti skirtingus sprendimo strategijų tipus, automatiškai juos klasifikuoti ir apibendrinti skirtinguose uždaviniuose, taip pereinant nuo dvejetainio rezultatų vertinimo prie procesinio informatinio mąstymo analizės.

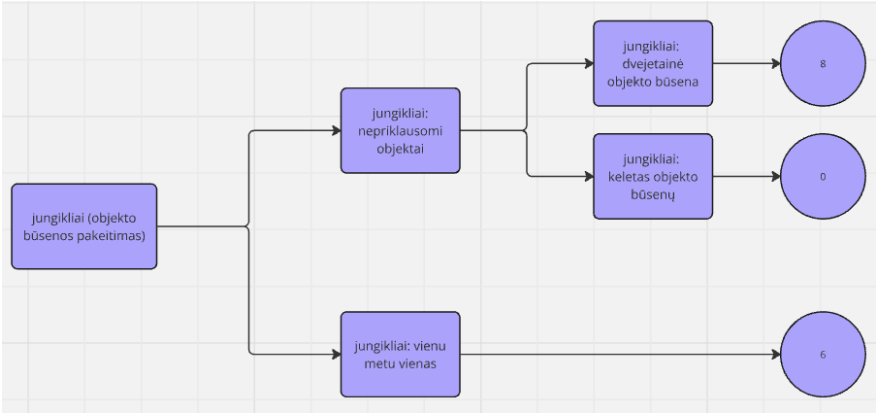
Eksperimentui naudoti 2023 m. Lietuvos konkurso „Bebras“ interaktyvių uždavinių duomenys. Iš viso analizuoti 974 864 sprendimų įrašai, surinkti iš 29 skirtingų interaktyvių uždavinių, kuriuos sprendė 1–12 klasių mokiniai. Kadangi tie patys uždaviniai buvo pateikiami kelioms amžiaus grupėms, uždavinių sprendimų skaičius reikšmingai skyrėsi – nuo kelių tūkstančių iki daugiau kaip aštuoniasdešimt tūkstančių sprendimų. Visi duomenys buvo visiškai anonimizuoti ir apėmė sprendimo veiksmų sekas, trukmę, paspaudimų skaičių, galutinę uždavinio būseną bei galutinio rezultato teisingumą.

Uždavinių sprendimų pasiskirstymas pateiktas diagramoje (S.4.1. pav.)



S.4.1 pav.: Konkurso „Bebras“ uždavinių sprendimų kiekiai

Siekiant užtikrinti elgsenos duomenų lyginamumą, pirmiausia visi interaktyvūs uždaviniai suklasifikuoti pagal interaktyvumo tipą. Klasifikacija rėmėsi sąveikos mechanikos analize, atsižvelgiant į tai, ar uždavinio sprendimą leidžiama koreguoti eigoje, ar veiksmai kaupia ar perrašo būseną, kokio tipo objektai naudojami ir ar galima identifikuoti dalinį teisingumą. Pradiniame lygmenyje uždaviniai skirstyti į tokias grupes, kuriose sprendimas atliekamas vienu bandymu, ir tokias, kuriose leidžiami keli bandymai ir klaidų taisymas. Tolesnė analizė leido išskirti dvylika interaktyvumo grupių, nors dauguma jų pasirodė struktūriškai unikalios. Tik trys grupės apėmė daugiau kaip vieną uždavinį, o tai pabrėžia interaktyvių uždavinių įvairovę ir kartu pagrindžia grupavimo būtinybę. S.4.2 paveiksle pateikiamas „switches“ uždavinių tipo grupavimo schema.



S.4.2 pav.: „Switches“ (jungiklių) tipo uždavinių grupavimo schema

Tolesnei elgsenos analizei pasirinkta „binary-switch“ tipo uždavinių grupė, nes jos struktūra artimiausia bandomajame tyrime analizuotam uždaviniui ir leido nuosekliai taikyti tą pačią elgsenos kodavimo schemą. Detaliai nagrinėti keturi uždaviniai: „Šifro nulaužimas“ (angl. *Breaking the Cipher)*, „Diena zoologijos sode“ (angl. *A Day at the Zoo*), „Konfliktų detektorius“ (angl. *Disagreement Detector*) ir „Domino“ (angl. *Domino*). Kiekvieno uždavinio sprendimų duomenys išvalyti, transformuoti į bendrą elgsenos požymių rinkinį ir analizuojami taikant *Affinity Propagation* klasterizavimo algoritmą. Požymiai apėmė sprendimo trukmę, paspaudimų skaičių, objektų būsenas bei sulankstytas veiksmų sekas, leidžiančias sumažinti padėties šališkumą ir išryškinti sprendimo struktūrą.

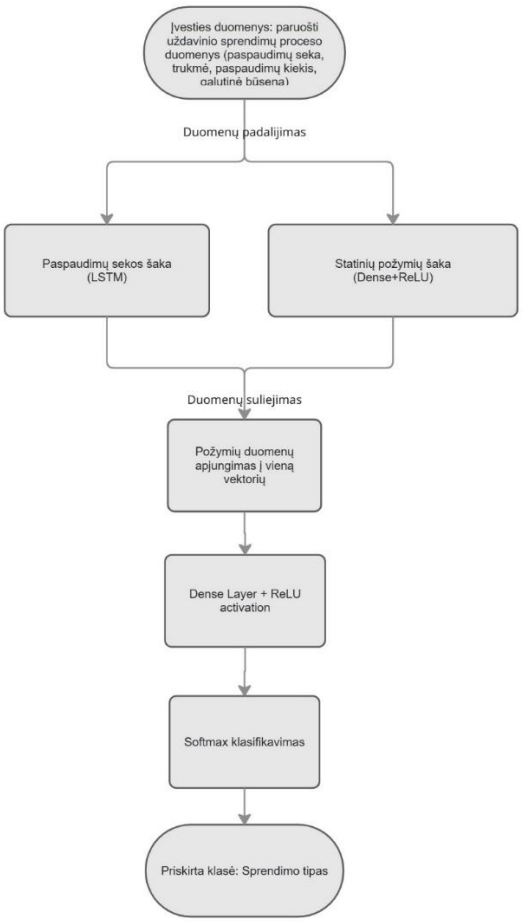
Klasterizavimo rezultatai parodė, kad net ir skirtinguose uždaviniuose kartojasi panašūs sprendimų elgsenos tipai. Identifikuoti visiškai teisingi ir veiksmingi sprendimai, teisingi sprendimai su pataisytomis klaidomis, bandymų ir klaidų strategijos, sprendimai, dalinį uždavinio supratimą rodantys sprendimai, bei sprendimai, kuriuose dominuoja atsitiktinis elgesys ar taisyklės nesupratimas. Šie elgsenos tipai sudarė pagrindą vieningai sprendimų tipų žymėjimo sistemai (S.4.1 lentelė), kuri leido apibendrinti skirtingų uždavinių klasterius ir naudoti juos tolesniam automatiniam klasifikavimui.

S.4.1 lentelė: Bendra skirtingų uždavinių sprendimų žymėjimo sistema

Nauja žyma klasifikavimui	Komentaras	Dabartiniai klasteriai „Šifro nulaužimas“ (RU_01)	Originalūs klasteriai „Šifro nulaužimas“ (RU_01)	Dabartiniai klasteriai „Konfliktų detektorius“ (DE_01)	Originalūs klasteriai „Konfliktų detektorius“ (DE_01)	Dabartiniai klasteriai „Domino“ (DE_09)	Originalūs klasteriai „Domino“ (DE_09)	Dabartiniai klasteriai „Diena zoologijos sode“ (AT_01)	Originalūs klasteriai „Diena zoologijos sode“ (AT_01)
101	Visiškai teisingas sprendimas	T1	8	T1	1;4	T1	2	T1	2
102	Uždavinys pradėtas spręsti neteisingai, bet vėliau pataisomas	T2	9	–	–	–	–	–	–
103	Sprendžiama klaidų ir bandymų metodu	T3	0;1	T2	0;3	T2	3	T2	0
104	Sprendžiama klaidų ir bandymų metodu, bet atliekama žymiai daugiau veiksmų	T4	6	T3	5	_	_	T3	5
201	Padaryta tik viena klaida	F1	8	F1	5	F1	8	F1	2
202	Dvi klaidos	_	_	F2	0;1;4	_	_	F2	5
203	Daugiau negu dvi klaidos	_	_	_	_	F2	4;5	F3	1
204	Pusė sprendimo teisingas	F2	1	F3	3	F3	2;3	_	_
205	Ne visa pusė sprendimo teisinga	F3	0;5	_	_	F4	9	_	_

Nauja žyma klasifikavimui	Komentaras	Dabartiniai klasteriai „Šifro nulaužimas“ (RU_01)	Originalūs klasteriai „Šifro nulaužimas“ (RU_01)	Dabartiniai klasteriai „Konfliktų detektorius“ (DE_01)	Originalūs klasteriai „Konfliktų detektorius“ (DE_01)	Dabartiniai klasteriai „Domino“ (DE_09)	Originalūs klasteriai „Domino“ (DE_09)	Dabartiniai klasteriai „Diena zoologijos sode“ (AT_01)	Originalūs klasteriai „Diena zoologijos sode“ (AT_01)
206	Nedidelė dalis sprendimo teisinga	F4	3;7;9	_	_	F5	0	_	_
207	Visiškai neteisingas sprendimas	F5	2;4;6	F4	2	F6	1;6; 7;10	F4	0;3;4

Remiantis šia sistema, buvo sukurta klasifikavimo procedūra. Sprendimų duomenys paruošti neuroninio tinklo mokymui, sujungiant sekų analizę ir statinius sprendimo požymius. Taikyta dvišakė architektūra, leidžianti atskirai apdoroti veiksmų sekas ir globalius sprendimo parametrus (S.4.3. pav.).



S.4.3 pav.: Uždavinių sprendimų klasifikavimo proceso schema

Modelio tikslumas buvo aukštas: teisingų sprendimų atpažinimas siekė apie 93 %, o klaidingų – apie 86 %, papildomai pagerintas mažų klasių atpažinimas, taikant klasių svorius, nors bendras tikslumas sumažėjo. Šie rezultatai rodo, kad elgsena grįsti sprendimų tipai yra pakankamai stabilūs automatinei klasifikacijai.

Modelio generalizavimo gebėjimai patikrinti taikant jį dviem naujiems to paties interaktyvumo tipo „Bebro“ uždaviniams. Nors uždaviniai skyrėsi objektų skaičiumi ir struktūra, modelis patikimai atpažino teisingus sprendimus ir iš dalies gebėjo klasifikuoti klaidingus sprendimus, nors pastariesiems reikalingas didesnis duomenų kiekis ir platesnė elgsenos tipų aprėptis. Tai rodo, kad modelis gali būti taikomas skirtingiems uždaviniams, tačiau jo tikslumas priklauso nuo sukauptų duomenų įvairovės.

Papildomai atliktas ekspertinis vertinimas, kuriame dalyvavo 35 pedagogai ir dėstytojai, parodė, kad nors praktikoje dažnai dominuoja galutinio atsakymo vertinimas, elgsena grįsta gradacinė sistema leidžia atskleisti sprendimo kokybinius skirtumus, kurie paprastai lieka nepastebėti. Tai patvirtina siūlomo modelio pedagoginį pagrįstumą ir jo potencialą taikyti mokymo bei grįžtamojo ryšio kontekstuose.

Apibendrinant galima teigti, kad eksperimentiniai rezultatai patvirtina teorinio modelio tinkamumą: interaktyvių uždavinių sprendimų elgsena turi stabilią vidinę struktūrą, kurią galima automatiškai modeliuoti ir interpretuoti. Siūlomas metodas sudaro prielaidas pereiti nuo dvejetainio informatinio mąstymo vertinimo prie procesinio, duomenimis grįsto sprendimų analizės modelio.

Informatinis mąstymas dažnai vertinamas supaprastintai neatsižvelgiama į tai, kaip mokiniai sprendžia uždavinius. Dauguma esamų vertinimo metodų remiasi dvejetaine teisingumo logika, vertindami tik galutinį atsakymą ir ignoruodami sprendimo procesą. Disertacijoje atlikta literatūros analizė atskleidė aiškią spragą: interaktyviuose uždaviniuose generuojami sprendimo proceso elgsenos duomenys vertinimo tikslais naudojami retai. Todėl svarbūs informatinio mąstymo aspektai, susiję su uždavinių sprendimu, lieka neatskleisti.

Šiame darbe ši spraga sprendžiama siūlant ir empiriškai pagrindžiant procesais grįstą informatinio mąstymo vertinimo metodą, paremtą interaktyvių „Bebro“ uždavinių sprendimo elgsenos duomenimis. Tyrimo tikslas ir iškelti uždaviniai įgyvendinti: išanalizuoti esami vertinimo metodai, ištirti elgsenos duomenys, sukurtas vertinimo modelis ir patikrintas jo veikimas naudojant realius duomenis.

Disertacijoje empiriškai parodyta, kad procesais grįsta interaktyvių uždavinių sprendimo analizė suteikia daugiau informacijos už įprastą tik galutiniu rezultatu paremtą vertinimą, ir sudaro pagrindą elgsenos duomenų taikymui informatinio mąstymo vertinime.

Pagrindinės šio tyrimo išvados, pagrįstos gautais rezultatais, yra šios:

1. Gauti rezultatai rodo, kad interaktyvių uždavinių sprendimo elgsenos duomenys gali būti automatizuotai klasterizuojami į prasmingas sprendimų grupes. Klasterizavimo rezultatai atskleidė aiškias sprendimo elgsenos struktūras, apimančias nuoseklius teisingus sprendimus, sprendimus su pataisytomis klaidomis, bandymų ir klaidų strategijas, iš dalies teisingus sprendimus ir neteisingus sprendimus. Bandomajame tyrime išryškėjo interpretuojama struktūra su maždaug aštuoniais klasteriais, o pagrindiniuose eksperimentuose „Affinity Propagation“ metodu identifikuotos septynios klaidingų sprendimų klasės ir keturios teisingų sprendimų klasės. Taigi skirtingi sprendimo tipai gali būti išskirti tiesiogiai iš elgsenos duomenų.

2. Analizuojant skirtingus uždavinius nustatyta, kad tai pačiai interaktyvumo grupei priklausančiuose uždaviniuose pasikartoja panašūs sprendimo elgsenos šablonai. Šie šablonai nėra specifiniai konkrečiam uždaviniui, bet pastebimi skirtinguose uždaviniuose.

Remiantis tuo, sukurta ir pritaikyta bendra sprendimo tipų žymėjimo schema, leidžianti nuosekliai interpretuoti ir palyginti sprendimo strategijas skirtinguose tos pačios interaktyvumo grupės uždaviniuose.

3. Sukurtas automatizuotas vertinimo modelis, paremtas suvienodintu sprendimo procesų atvaizdavimu, leidžia klasifikuoti sprendimus gana tiksliai. Modelis pasiekė 85,9 % tikslumą klaidingų sprendimų atveju ir 93,0 % teisingų sprendimų atveju be svorių taikymo. Pritaikius klasių svorius, tikslumas šiek tiek sumažėjo (atitinkamai iki 83,4 % ir 90,6 %), tačiau pagerėjo retesnių sprendimų tipų atpažinimas. Mažesnių klasių „recall“ rodikliai padidėjo nuo 0,464 ir 0,321 iki 0,844 ir 0,728. Tai rodo, kad modelis geba patikimai atpažinti tiek dažniausius, tiek retesnius sprendimo elgsenos tipus.

4. Siūlomo modelio praktinį pagrįstumą papildomai patvirtino ekspertinis vertinimas. 94 % ekspertų visiškai arba iš dalies pritarė detalesnio, diferencijuoto vertinimo poreikiui, o 80 % nurodė, kad vertindami taikytų diferencijuotą vertinimą, mažintų balus pagal klaidų skaičių arba skirtų dalinius balus, jei sprendime padarytos kelios klaidos (pvz., 2–4 klaidos). Tai rodo, kad procesais grįstas vertinimas dera su realia pedagogine praktika.

5. Sykiu rezultatai atskleidžia ir tam tikrus apribojimus. Modelis geriausiai veikia su uždaviniais, kuriuose generuojami struktūruoti elgsenos duomenys ir kurie gali būti priskirti aiškiai apibrėžtoms interaktyvumo grupėms. Sudėtingesnių ar retesnių klaidingų sprendimų klasifikavimas priklauso nuo turimų duomenų kiekio ir įvairovės.

Praktiniu požiūriu siūlomas metodas gali būti taikomas įvairiuose kontekstuose, kuriuose naudojamos interaktyvios problemų sprendimo užduotys, nes jis nėra susietas su konkrečia amžiaus grupe ar ugdymo lygiu. Šis metodas leidžia taikyti detalesnį, į sprendimo procesą orientuotą vertinimą bei suteikti išsamesnį grįžtamąjį ryšį, pagrįstą ne tik galutiniu rezultatu, bet ir sprendimo konstravimo eiga. Sykiu reikalingi tolesni tyrimai, siekiant patikrinti modelio taikymą platesniame uždavinių tipų ir mokymosi kontekstų spektre.

CURRICULUM VITAE

Vaida Masiulionytė-Dagienė completed her bachelor's studies in Informatics in 2004 and obtained a master's degree in Computer Modelling in 2006 at the Faculty of Mathematics and Informatics of Vilnius University. After her studies, she worked in the IT industry in various roles, including software engineer, analyst, and project manager, gaining extensive practical experience in software development and project delivery. She was also a founding member of the Lithuanian Agile Association. In parallel with her professional career in industry, she was actively involved in academic work, teaching at the Faculty of Mathematics and Informatics of Vilnius University. In 2021, she started her doctoral studies in Informatics Engineering at Vilnius University. She currently works there as an assistant and researcher, while also teaching at Vilnius Waldorf school. Her research focuses on computational thinking and its automated assessment, as well as the application of educational technologies in computer science education, with particular emphasis on the Bebras Challenge and its tasks. During her doctoral studies, she has published several scientific papers in international conferences and journals.

In addition to her teaching and research activities, she is actively involved in international research and educational projects and contributes to international academic collaboration activities. She is also actively engaged in the international Bebras Challenge community. Since 2026, she has been coordinating the Bebras Challenge in Lithuania. Her academic and pedagogical activities are closely connected, with a strong emphasis on the development of computational thinking skills and innovative approaches to teaching informatics.

Vaida Masiulionytė-Dagienė

Modelling of Computational Thinking Automated Assessment

Doctoral Dissertation

Technological Sciences

Informatics Engineering (T 007)

Thesis Editor: Zuzana Šiušaitė

Informatinio mąstymo automatizuoto vertinimo modeliavimas

Daktaro disertacija

Technologijos mokslai

Informatikos inžinerija (T 007)

Santraukos redaktorė: Jorūnė Rimeisytė-Nekrašienė

Vilniaus universiteto leidykla

Saulėtekio al. 9, III rūmai, LT-10222 Vilnius

El. p. info@leidykla.vu.lt, www.leidykla.vu.lt

bookshop.vu.lt, journals.vu.lt

Tiražas 20 egz.