Automata Theory in Computing Education: A Systematic Review

Júlia Veiga da Silva[1], **Simone André da Costa Cavalheiro**[1], **Luciana Foss**[1]

[1]Technology Development Center – Federal University of Pelotas (UFPel)
CEP 96.010-610 – Pelotas – RS – Brazil

`{jvsilva,simone.costa,lfoss}@inf.ufpel.edu.br`

Abstract. *This paper presents a systematic literature review investigating approaches to teaching Automata Theory in Computing Education. Automata Theory, a foundational area of Computer Science essential for understanding formal languages and compilers, is increasingly recognized for its importance in educational contexts. This study examines how Automata Theory is approached, assessed, integrated, and taught in both higher education and K-12 settings. The review reveals several innovative and traditional methods used to enhance student engagement and comprehension, including educational tools and games. However, the findings highlight a lack of standardized assessment methods in current practices, as well as a gap in approaches in the K-12 context.*

1. Introduction

Automata Theory is the study of abstract computing devices, often referred to as “machines” [Hopcroft et al. 2001]. It facilitates the modeling and precise analysis of computational behaviors abstractly. This theory is a fundamental part of Computer Science, providing the theoretical basis for several areas, including formal languages [Hopcroft and Ullman 1969] and compilers [Mogensen 2024]. Despite its significance, theoretical Computer Science courses often pose considerable challenges for students, resulting in high failure rates in introductory classes. These challenges primarily stem from the abstract nature of the theoretical concepts introduced, which can be difficult to understand without appropriate instructional support.

Some approaches have been adopted to address these challenges and enhance the learning experience. Cognitive apprenticeship [Knobelsdorf et al. 2014], the Moore method [Coppin et al. 2009], and the integration of theoretical computing questions into high school curricula [del Vado Vírseda 2020] are among the strategies aimed at improving student comprehension and engagement. However, the effectiveness and implementation of these approaches vary, prompting a need for systematic exploration of how Automata Theory is approached, assessed, and integrated into Computing Education, especially at the K-12 level. This paper aims to address these aspects through a Systematic Literature Review (SLR). Specifically, it investigates three main research questions:

1. How is Automata Theory being approached in Computing Education?
2. How is Automata Theory being assessed in Computing Education?
3. How is Automata Theory being integrated and taught in K-12 Education?

The rest of this paper is organized as follows: Section 2 discusses related work, situating the paper within the context of Automata Theory in Computing Education; Section 3 describes the methods we followed in conducting the SLR; Section 4 presents our findings; Section 5 discusses the results; Section 6 concludes the paper with suggestions for future research.

DOI: 10.5753/sbie.2024.242472

2. Related Work

A similar study provides a comprehensive review of automata simulators used for educational purposes over the past five decades, emphasizing their role in teaching Automata Theory at universities [Chakraborty et al. 2011]. The study categorizes these simulators into two broad types based on their design paradigms: language-based and visualization-centric simulators.

The authors identified several trends in the development and use of automata simulators. Despite the wide range of existing tools, new simulators are continuously welcomed, each offering unique principles and utilities. The evolution of software development tools, such as JFLAP [Procopiuc et al. 1996], has significantly influenced simulator capabilities, particularly in graphics support. Developers often publish detailed descriptions of their tools in the refereed literature, ensuring long-term accessibility and usability for teaching and further tool development.

While web availability facilitates widespread access, it poses traceability challenges if web addresses change without corresponding literature documentation. Advanced graphics and animation techniques are frequently employed to enhance educational effectiveness, and student feedback is crucial for ongoing improvements. Although most simulators are in English, availability in other languages has expanded their reach to a broader user community.

Finally, the paper [Chakraborty et al. 2011] advocates for continued research and integration of automata simulators in education, recognizing their significant role in teaching and learning Automata Theory. This work is closely related to our study, as it comprehensively reviews automata simulators used in teaching over the past five decades. However, it was published in 2011 and focuses solely on automata simulation tools, not addressing other multidisciplinary approaches relevant to contemporary education in Automata Theory.

3. Methods

We followed a guideline proposed for SLR in Software Engineering [Kitchenham 2004], which suggests three main phases: Planning the Review, including the identification of the need for a review and the development of a review protocol; Conducting the Review, including the selection of studies, data extraction, and the synthesis; and Reporting the Review, for which we followed PRISMA [Page et al. 2021] guidelines.

The review protocol was built based on the research questions presented in the introduction. The chosen databases and search engines were selected to target those specialized in education (ERIC) and computing/STEM (IEEE Xplore, ACM, and SOL), in addition to the *Portal de Periódicos da CAPES*.

To account for the databases searched, including Brazilian ones, and to capture a larger number of studies without going out of the scope of the work, we developed the following search strings using words related to "automata" and "computing education":

- *("transition system" OR "state machine" OR "automaton" OR "automata" OR "automata theory") AND ("computer education" OR "computing education" OR "computer science education" OR "k-12")*;
- *("sistema de transição" OR "máquina de estado" OR "autômato" OR "autômatos" OR "teoria dos autômatos") AND ("educação em computação" OR "ensino de computação" OR "educação básica" OR "ensino fundamental" OR "ensino médio").*

We applied specific filters in each database according to available resources to ensure the relevance and quality of the articles included in this SLR. In the ERIC database, we restrict results to peer-reviewed articles only. At IEEE Xplore, we filter results by including only conferences and journals. In the ACM Digital Library, we limit the search to research articles published in the last five years due to the many articles returned. At SOL, we select event annals and journals published in English and Portuguese. Finally, in the CAPES database, we restricted the search to articles written in English and Portuguese.

The studies were selected through an initial screening of titles and abstracts. The screening excluded all papers that met the following exclusion criteria:

1. Not being in an educational context, i.e., not related to teaching-learning;
2. Being in an educational context but not in the computing field;
3. Automata/Automata Theory not being one of the topics addressed in the work;
4. Just mentioning Automata/Automata Theory without featuring it as the main theme;
5. Being a duplicate (already analyzed from other search databases);
6. Not being an article (another type of document such as a report, text, book chapter, poster, etc.).

We sought to retrieve the full papers of all those passing the screening phase or whose titles and abstracts were insufficient to determine whether the study should be included. Finally, we repeated the analysis of the full texts for selecting the papers to be included in the review. The data extraction process, guided by answering our research questions, consisted of coding the selected studies according to their approach (using games, digital tools, theoretical classes, experiments, or teaching methods), tool (the name of the game, activity, or tool used, if any), concepts involved, audience (K-12 and/or higher education), country, and assessment (pre/post-test, surveys, or questionnaires).

4. Results

Our search returned 392 documents; 27 were removed due to being duplicates, 311 were excluded based on the exclusion criteria during the title and abstract screening, and another 15 were excluded in the full-text analysis. Additionally, 7 documents were removed because we could not access their full texts. Figure 1 details the search outcomes using the PRISMA [Page et al. 2021] flowchart. The selected studies are listed along with their respective approach, tools, concepts, audiences, countries, and assessment forms taken into consideration in Table 1.

To improve the clarity of the table for readability, the following abbreviations were used for the concepts presented: FM (Formal Languages); AT (Automata Theory); DFA (Deterministic Finite Automata); NFA (Nondeterministic Finite Automata); PDA (Pushdown Automata); NPDA (Nondeterministic Pushdown Automata); TM (Turing Machines); FSM (Finite State Machines); RegEx (Regular Expressions); and CFG (Context-Free Grammar).

Table 1: Studies approaching Automata Theory in Computing Education

Study	**Approach**	**Tool/Resource**	**Covered concepts**	**Audience**	**Country**	**Assessment**
[Isayama et al. 2016]	Digital Game	Automata Puzzle	DFA	K-12	Japan	Play logs analysis; Custom post-test
[Castro-Schez et al. 2009]	Educational Tool	SELFA	FL and AT	Higher Education	Spain	Custom questionnaire
[Cogliati et al. 2005]	Educational Tool	Hypertextbook on the Theory of Computing	FL and AT	Higher Education	United States	None

Continues on the next page

Study	**Approach**	**Tool/Resource**	**Covered concepts**	**Audience**	**Country**	**Assessment**
[Habiballa and Kmet' 2004]	Experiment	Constructivist and Traditional Teaching Methods Comparison	FL and AT	Higher Education	Czech Republic	Custom post-test
[Holly et al. 2022]	Digital Game	Chess-Puzzle-Simulation	State Machines	Higher Education	Austria	Tasks in tool; Custom pre, post and workload questionnaires
[Dengel 2018]	Educational Virtual Environment	Treasure Hunt VR	FSM	Higher Education and K-12	Germany	None
[Budiselic et al. 2007]	Educational Tool	RegExpert	RegEx and NFA-epsilon	Higher Education	Croatia	None
[Maurer 2021]	Theoretical Class	"Foundations of Computer Science"	FSM, PDA, and TM	Higher Education	United States	None
[Hartmann et al. 2001]	Educational Tool	Kara	FSM	Higher Education and K-12	Switzerland	Custom survey
[Dengel 2020]	Experiment	Computer Science Replugged	FSM	K-12	Germany	Custom pre and post-test
[Cogumbreiro and Blike 2022]	Educational Tool	Gidayu	NFA and (N)PDA	Higher Education	United States	None
[Bezáková et al. 2022]	Educational Tool	DAVID, a JFLAP Extension	DFA, NFA, RegEx, CFG, and PDAs	Higher Education	United States	Custom survey; Grades and homework submissions analysis
[Mohammed et al. 2021]	Educational Tool	eTextbook with Integrated Tools	DFA, NFA, PDA, and TM	Higher Education	United States	Custom post-test; Opinion survey
[Dey et al. 2020]	Experiment	Supporting Asynchronous Learning Environments with Multiple MT Representations	TM	Higher Education	United States	Custom optional opinion survey
[Neto and Terra 2016]	Mobile Application	LFApp	FL	Higher Education	Brazil	None
[Honda et al. 2023]	Digital Game	Automigos	DFA	Higher Education	Brazil	MEEGA+ questionnaire
[Jovanović et al. 2021]	Educational Tool	ComVis Finite Automata	Finite Automata	Higher Education	Serbia and India	Likert survey
[Cavalcante et al. 2004]	Educational Tool	JFLAP 4.0	Multi-tape TM, L-systems, Grammar Transformations, and Types of Parsing	Higher Education	United States	Custom evaluation questionnaire
[Hung and Rodger 2000]	Educational Tool / Experiment	JFLAP and Pâté new features	RegEx, NFA (JFLAP); Parse Tree for Unrestricted Grammars (Pâté)	Higher Education	United States	None
[Chesnevar et al. 2004]	Teaching Method	Didactic Strategies for Promoting Significant Learning	FL and AT	Higher Education	United States / Spain	None (empirical feedback)
[Gramond and Rodger 1999]	Educational Tool	JFLAP	LF and AT	Higher Education	United States	None (informal feedback)
[Rodger et al. 1997]	Educational Tool	JFLAP, Pâté, and PumpLemma	Finite Automata, PDA, and TM (JFLAP); Grammars Parsing Analysis, and CFGs to Chomsky Normal Form Transformation (Pâté); Proving Specific Languages are Not Regular (PumpLemma)	Higher Education	United States	None

Continues on the next page

Study	Approach	Tool/Resource	Covered concepts	Audience	Country	Assessment
[Korte et al. 2007]	Educational Tool	Game-building Educational Tool	Finite State Automata, RegEx, and TM	Higher Education	Scotland	None (informal observations)
[Wagenknecht and Friedman 1998]	Teaching Method	High-level programming language Scheme	Nondeterministic and Universal Automata	Higher Education	—	Custom individual interviews
[McDonald 2002]	Educational Tool	Interactive Pushdown Automata Animation	PDA	Higher Education	United States	None
[Grinder 2002]	Educational Tool	Animating Automata	(N)DFA	Higher Education	United States	None
[Grinder 2003]	Educational Tool	Finite State Automaton Animator	(N)DFN	Higher Education	United States	Custom tests (online and paper-and-pencil)
[Robinson et al. 1999]	Educational Tool	Java Computability Toolkit (JCT)	Finite Automata and TM	Higher Education	United States	Informal feedback
[Vieira et al. 2004]	Educational Tool	Language Emulator	RegEx, Regular Grammars, (N)DFA, and Moore and Mealy Machines	Higher Education	Brazil	Informal test and feedback
[Dershowitz and Dowek 2016]	Teaching Method	Two-dimensional Models of Computation for TM	TM	Higher Education and K-12	—	None

4.1. How is Automata Theory being approached in Computing Education?

Automata Theory is approached in Computing Education through several innovative and traditional methods to enhance student engagement and understanding. Interactive tools such as Gidayu [Cogumbreiro and Blike 2022], JFLAP [Gramond and Rodger 1999], and RegExpert [Budiselic et al. 2007] incorporate visualizations and simulations to help students grasp complex concepts such as Deterministic Finite Automata, Nondeterministic Finite Automata, Pushdown Automata, and Turing Machines. Educational games and mobile applications, including Automigos [Honda et al. 2023] and LFApp [Neto and Terra 2016], employ gamification to make the learning process more engaging. Virtual reality tools such as Treasure Hunt VR [Dengel 2018] offer immersive experiences for visualizing Finite State Machines and other automata concepts.

Traditional formal and didactic methods, such as lectures and structured teaching strategies, continue to promote the learning of formal languages and automata subjects substantially. Additionally, innovative teaching methods like two-dimensional models of computation are used to teach Turing Machines [Dershowitz and Dowek 2016], addressing both higher education and K-12 students.

4.2. How is Automata Theory being assessed in Computing Education?

The assessment of Automata Theory in computing education employs a set of methods to evaluate student learning and the effectiveness of teaching approaches. However, most studies either did not have formal evaluations or relied on self-developed/custom assessments. Some of the more formal assessment methods include surveys and questionnaires, such as MEEGA+ [Petri et al. 2016] and Likert-type [DeVellis 2003] surveys, which gather student feedback on their learning experiences. Pre- and post-tests are occasionally used to measure the impact of specific educational interventions. Traditional metrics, such as student grades and homework, provide a baseline for evaluation, while informal feedback from students offers additional perceptions of their learning experiences. Despite the diversity in assessment methods, there is a significant reliance on self-developed/custom evaluation approaches adapted to specific educational contexts, reflecting the need for more standardized assessment methods in this field.

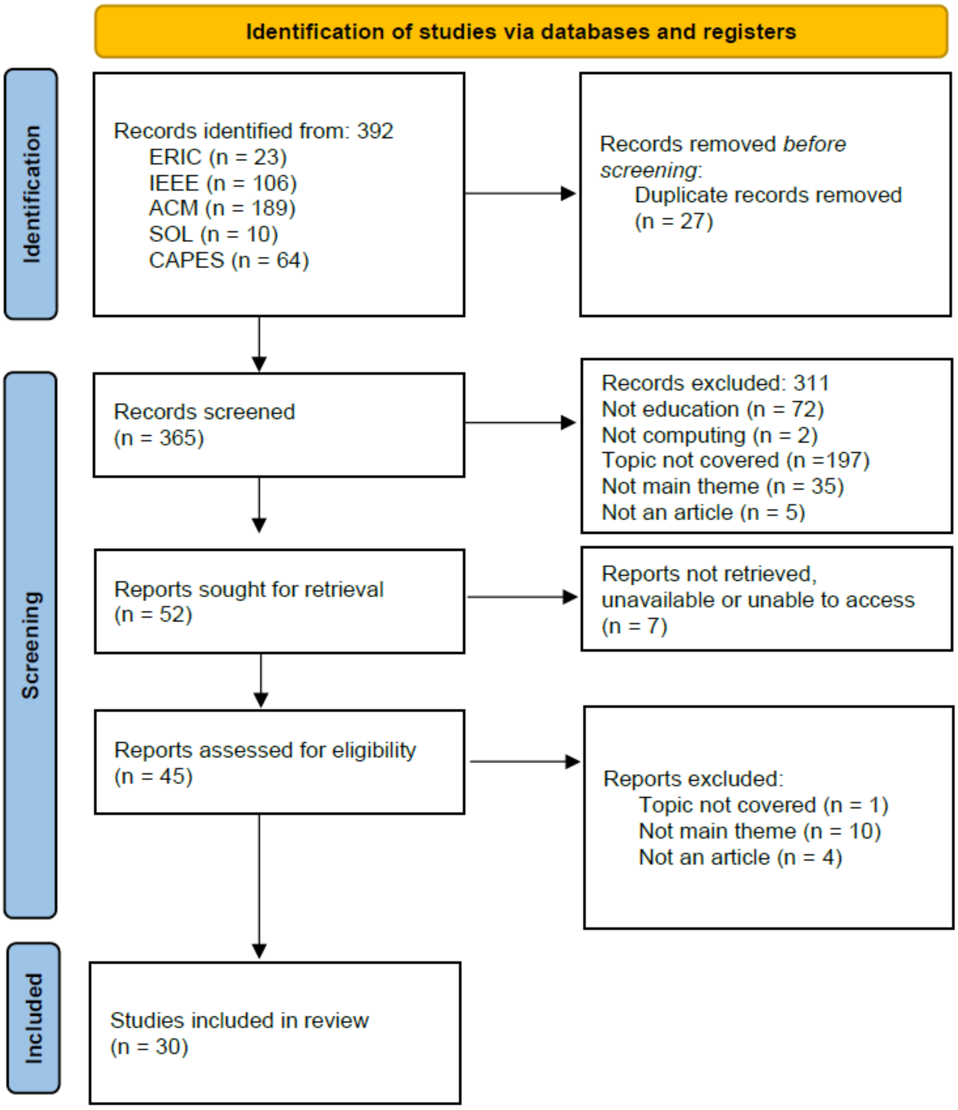


Figure 1. PRISMA flowchart of the review

4.3. How is Automata Theory being integrated and taught in K-12 Education?

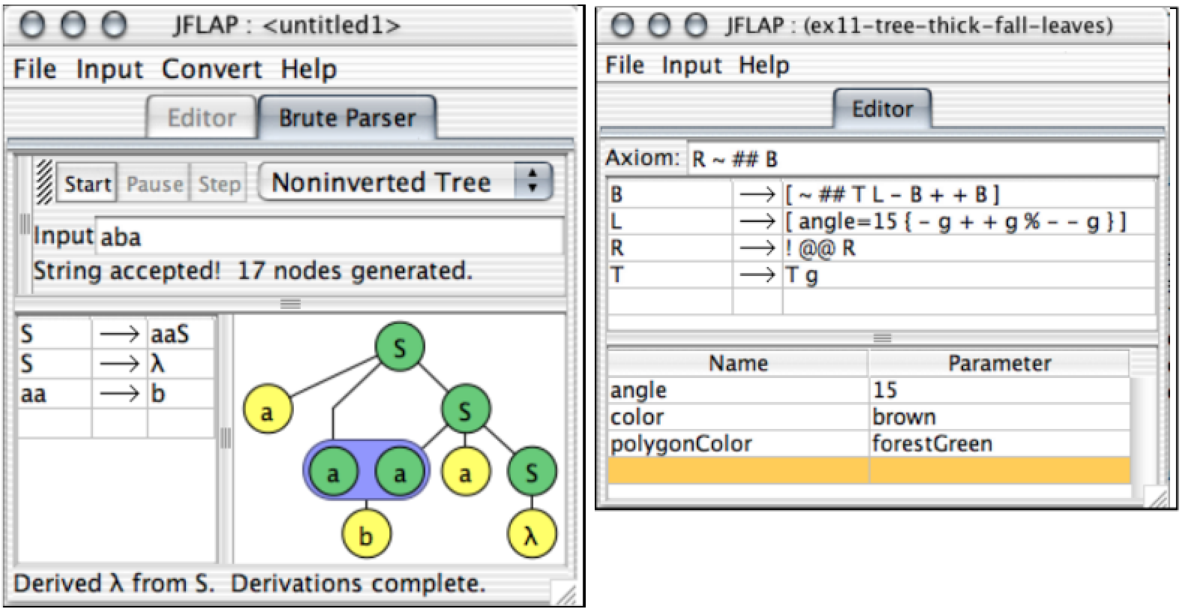
Automata Theory is being integrated and taught in K-12 Education mainly through digital games and interactive tools designed to make abstract concepts more accessible to younger students. For example, in Japan, the Automata Puzzle game [Isayama et al. 2016] is specifically designed for K-12 students (9-12 years old), focusing on teaching Deterministic Finite Automata. In Switzerland, the educational tool Kara [Hartmann et al. 2001] is used to teach Finite State Machines across different educational levels, including K-12. In Germany, the Treasure Hunt VR [Dengel 2018] provides an innovative, immersive virtual reality environment where K-12 students can explore and learn about Finite State Machines. Additionally, Computer Science Replugged [Dengel 2020], a collection of free learning activities, teaches Computer Science concepts, including Automata Theory, through engaging activities. These methods highlight the importance of interactive and immersive experiences in making complex subjects such as Automata Theory approachable for younger students.

4.4. Some Examples

This subsection presents some of the most relevant works identified during the SLR. The works were selected based on criteria such as innovation in the educational approaches, reported impact on student learning, and the use of interactive tools and advanced technologies. The selected works provide a comprehensive view of the diverse strategies used to teach Automata Theory in Computing Education.

The first study, presented by Cavalcante et al. (2004) , introduces the JFLAP 4.0 tool, developed to create a visual and interactive course on Automata Theory. JFLAP 4.0 allows students to interact with automata, grammars, and Turing Machines visually, making complex concepts easier to understand. This new version covers topics from eleven chapters of a Formal Languages and Automata course, including new features such as multi-tape Turing Machines, L-systems, grammar transformations, and different types of parsing. JFLAP 4.0 also offers new approaches for converting between Nondeterministic Finite Automata and regular expressions and for matching and comparing automata. Used in courses at Duke University, the tool proved to be effective, providing a practical and interactive means for teaching Automata Theory.

The evaluation of the JFLAP 4.0 tool at Duke University, specifically in the CPS 140 course during Spring 2003, revealed positive student feedback. Out of 33 students, all found JFLAP easy to use. Of the 27 students who did not consult the help documentation, the 6 who did found it useful. When asked about their preference between creating Finite Automata using JFLAP or drawing them on paper, 17 students preferred using JFLAP, 12 preferred drawing on paper first and then testing on JFLAP, and 2 preferred paper only. Figure 2 shows an example of JFLAP 4.0.



Input aba

String accepted! 17 nodes generated.

S → aaS
S → λ
aa → b

Derived λ from S. Derivations complete.

Axiom: R ~ ## B

B → [~ ## T L - B + + B]
L → [angle=15 { - g + + g % - - g }]
R → ! @@ R
T → T g

Name	Parameter
angle	15
color	brown
polygonColor	forestGreen

Figure 2. JFLAP 4.0 educational tool. Source: [Cavalcante et al. 2004]

The second study introduces the educational software system Kara [Hartmann et al. 2001], which allows students to program a virtual ladybug using Finite State Machines. Students sequence commands for each state of the ladybug, transitioning to the next state at the end of each sequence. They can also branch execution based on inputs from “sensors” on the ladybug, such as detecting trees or leaves. Kara includes extensions such as TuringKara for Turing Machines and JavaKara to facilitate the transition to real-world programming languages such as Java.

Widely used in Germany, Austria, and Switzerland, Kara is employed from primary to higher education levels to teach programming in a motivating way. Beginners appreciate its simplicity and find it conducive to learning programming concepts before advancing to more traditional languages. Figure 3 shows an example of Kara educational tool.

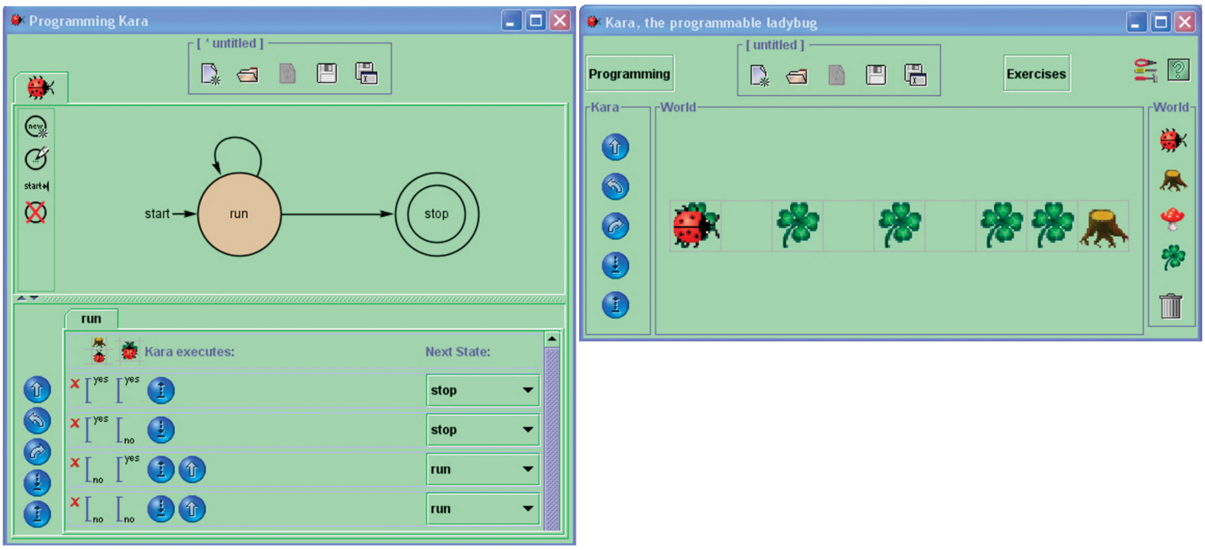


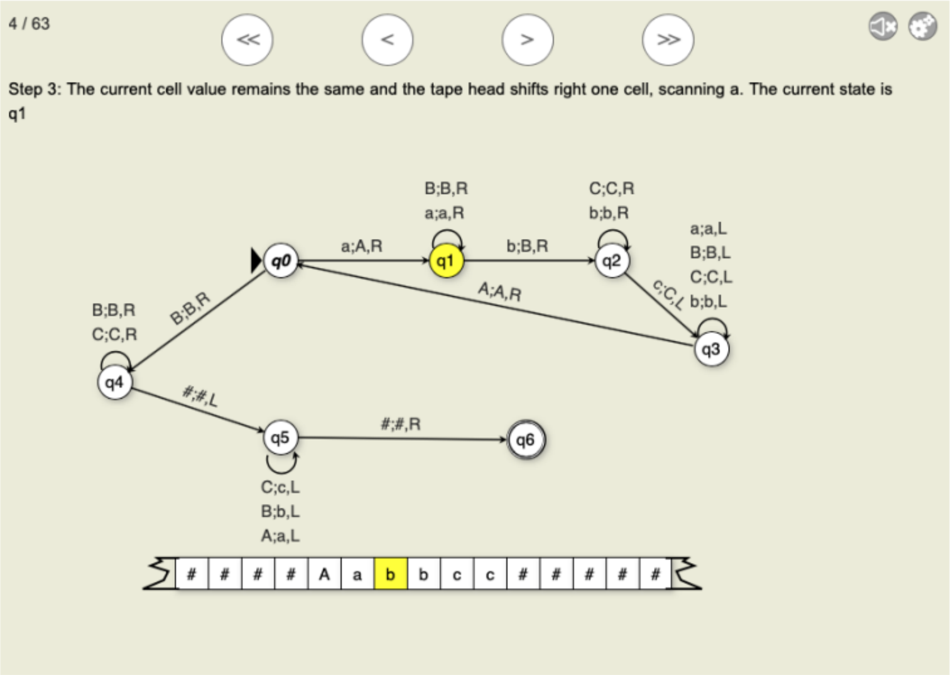
Figure 3. Kara programming environment (left) and board (right). Source: [Kiesmüller 2009]

The third study proposes an integrated eTextbook for teaching Formal Languages and Automata [Mohammed et al. 2021]. Utilizing the OpenDSA platform, the eTextbook combines text, visualizations, simulators, and many interactive exercises through the JSAV library. The tool, called OpenFLAP, reimplements and expands parts of JFLAP, offering auto-graded exercises that test students' ability to create Deterministic Finite Automata, Nondeterministic Finite Automata, Pushdown Automata, and Turing Machines, as well as proficiency exercises that require the application of algorithms.

The tool's evaluation included comparative studies to test the hypothesis that the eTextbook increases student interaction and provides more practice opportunities. The results indicated that students using the integrated eTextbook performed better than the control group using a traditional textbook. Student feedback was positive, highlighting the usefulness of the auto-graded exercises. Thus, the article concludes that the eTextbook with visualizations and auto-graded exercises effectively improves students' understanding of Formal Language and Automata, proposing future expansions and enhancements to the functionalities of OpenFLAP and the eTextbook. Figure 4 shows an example of the resource.

The fourth work presents Automigos, an educational game designed to facilitate learning complex computing topics, specifically Deterministic Finite Automata [Honda et al. 2023]. The game employs Jean Piaget's Constructivist learning theory and aims to enhance students' understanding through gamification. The design process involves a meticulous sequence of steps, including conception, gamification, and evaluation, to ensure that the educational content is effectively integrated into the game mechanics.

Automigos was positively received by its target audience, with approving feedback on the game's aesthetics, mechanics, and overall enjoyment. However, some concerns were noted



Step 3: The current cell value remains the same and the tape head shifts right one cell, scanning a. The current state is q1

Figure 4. Example of a Turing Machine accepting a string. Source: [Mohammed et al. 2021]

regarding the clarity of rules and content understanding. The study highlights the challenges of creating educational games that effectively teach complex topics, emphasizing the importance of thorough evaluation and iterative design to balance educational value and engagement. Automigos represents a significant effort to make abstract computational concepts more accessible and engaging for students through innovative educational tools. Figure 5 shows an example of the game.

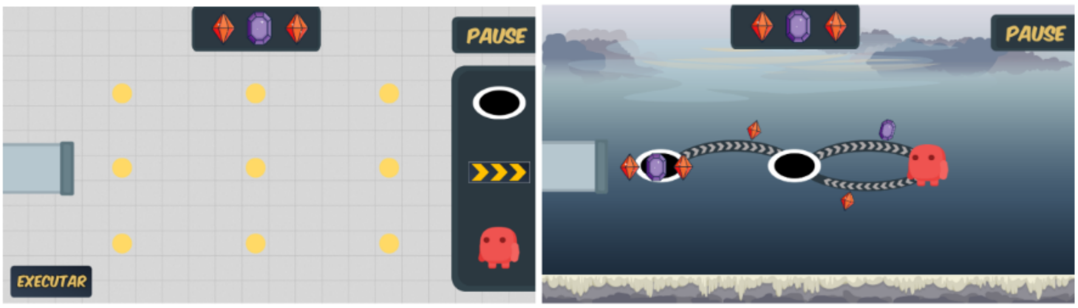


Figure 5. Automigos educational game. Source: [Honda et al. 2023]

5. Discussion

The SLR results highlight using interactive tools and gamification to teach Automata Theory concepts. Educators can effectively enhance student engagement and understanding by integrating interactive simulations and game-like elements into the curriculum. Such approaches make abstract theoretical concepts more tangible and foster active learning environments where students can experiment, visualize, and intuitively grasp complex ideas. Moreover, the gamification of learning activities encourages competition, collaboration, and inherent motivation among students, thereby potentially improving learning outcomes and retention rates in this foundational area of Computer Science.

On the other hand, the assessment of Automata Theory education reveals some challenges. While several methods are employed to evaluate student learning and the effectiveness of teaching approaches, many studies either lack formal evaluations or rely heavily on self-developed/custom assessments. It indicates a significant need to develop more standardized and universally applicable assessment methods to ensure consistent and comprehensive evaluation of student learning outcomes.

Furthermore, integrating Automata Theory in K-12 Education underscores the importance of adapting educational tools and methods to younger students' cognitive levels and learning styles. The success of digital games and virtual reality tools in this context suggests that interactive and immersive learning experiences are particularly effective for K-12 students.

For example, in Brazil, recent updates to the Brazilian National Curricular Common Base (BNCC) [Brazil 2022] have included an appendix dedicated to Computing Education, outlining essential content for K-12 Education. Despite these advancements, there remains a significant lack of research and educational initiatives especially addressing Automata Theory for K-12 students in Brazil. Issues such as the availability of adequate resources and teacher training need to be addressed to integrate these teaching methods effectively.

6. Conclusion

This work reviewed how Automata Theory is being taught in Computing Education. The current approaches to teaching Automata Theory in Computing Education, especially within K-12, demonstrate a significant shift towards more interactive and engaging methods. While these innovations have shown promising results in enhancing student understanding and engagement, the field still faces challenges in standardizing assessment methods. Future research should focus on developing and validating more standardized assessment tools and exploring the long-term impact of these innovative teaching methods on student learning outcomes in Automata Theory.

Furthermore, even with the appendix that deals with the inclusion of computing in the BNCC, few works specifically address this topic in the Brazilian context. This gap in literature underscores the need for more research and educational initiatives focused on effectively integrating computing concepts and skills into the national curriculum. Developing resources that align with local educational needs and challenges can facilitate the implementation of Computing Education nationwide, fostering digital literacy and preparing students for the demands of a technology-driven society. Efforts in this direction are crucial for advancing educational equity and empowering future generations with essential computational competencies.

References

Bezáková, I., Fluet, K., Hemaspaandra, E., Miller, H., and Narváez, D. E. (2022). Effective Succinct Feedback for Intro CS Theory: A JFLAP Extension. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1*, SIGCSE 2022, page 976–982, New York, NY, USA. Association for Computing Machinery.

Brazil (2022). Normas sobre Computação na Educação Básica. `http://portal.mec.gov.br/index.php?option=com_docman&view=download&alias=182481-texto-referencia-normas-sobre-computacao-na-educacao-basica&category_slug=abril-2021-pdf&Itemid=30192`. Online. Accessed on March 2024.

Budiselic, I., Srbljic, S., and Popovic, M. (2007). RegExpert: A Tool for Visualization of Regular Expressions. In *EUROCON 2007 - The International Conference on "Computer as a Tool"*, pages 2387–2389.

Castro-Schez, J. J., del Castillo, E., Hortolano, J., and Rodriguez, A. (2009). Designing and Using Software Tools for Educational Purposes: FLAT, a Case Study. *IEEE Transactions on Education*, 52(1):66–74.

Cavalcante, R., Finley, T., and Rodger, S. H. (2004). A Visual and Interactive Automata Theory Course with JFLAP 4.0. In *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education*, pages 140–144.

Chakraborty, P., Saxena, P. C., and Katti, C. P. (2011). Fifty Years of Automata Simulation: A Review. *ACM Inroads*, 2(4):59–70.

Chesnevar, C. I., González, M. P., and Maguitman, A. G. (2004). Didactic Strategies for Promoting Significant Learning in Formal Languages and Automata Theory. *ACM SIGCSE Bulletin*, 36(3):7–11.

Cogliati, J. J., Goosey, F. W., Grinder, M. T., Pascoe, B. A., ROSS, R. J., and Williams, C. J. (2005). Realizing the Promise of Visualization in the Theory of Computing. *J. Educ. Resour. Comput.*, 5(2):5–es.

Cogumbreiro, T. and Blike, G. (2022). Gidayu: Visualizing Automaton and Their Computations. In *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 1*, ITiCSE '22, page 110–116, New York, NY, USA. Association for Computing Machinery.

Coppin, C. A., Mahavier, W. T., May, E. L., and Parker, E. (2009). *The Moore Method: A Pathway to Learner-Centered Instruction.* Number 75. MAA.

del Vado Vírseda, R. (2020). From the Mathematical Impossibility Results of the High School Curriculum to Theoretical Computer Science. In *Proceedings of the 20th Koli Calling International Conference on Computing Education Research*, pages 1–5.

Dengel, A. (2018). Seeking the Treasures of Theoretical Computer Science Education: Towards Educational Virtual Reality for the Visualization of Finite State Machines. In *2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pages 1107–1112.

Dengel, A. (2020). How Important is Immersion for Learning in Computer Science Replugged Games? In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, SIGCSE '20, page 1165–1171, New York, NY, USA. Association for Computing Machinery.

Dershowitz, N. and Dowek, G. (2016). Universality in Two Dimensions. *Journal of Logic and Computation*, 26(1):143–167.

DeVellis, R. F. (2003). Scale Development: Theory and Applications. *Sage London*, 13:0–176.

Dey, P. P., Sinha, B. R., and Amin, M. (2020). Supporting Asynchronous Learners with Multiple Representations. *Journal of Computing Sciences in Colleges*, 36(2):108–116.

Gramond, E. and Rodger, S. H. (1999). Using JFLAP to Interact with Theorems in Automata Theory. In *The Proceedings of the Thirtieth SIGCSE Technical Symposium on Computer Science Education*, pages 336–340.

Grinder, M. T. (2002). Animating Automata: A Cross-Platform Program for Teaching Finite Automata. In *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education*, pages 63–67.

Grinder, M. T. (2003). A Preliminary Empirical Evaluation of the Effectiveness of a Finite State Automaton Animator. *ACM Sigcse Bulletin*, 35(1):157–161.

Habiballa, H. and Kmet', T. (2004). Theoretical Branches in Teaching Computer Science. *International Journal of Mathematical Education in Science and Technology*, 35(6):829–841.

Hartmann, W., Nievergelt, J., and Reichert, R. (2001). Kara, Finite State Machines, and the Case for Programming as Part of General Education. In *Proceedings IEEE Symposia on Human-Centric Computing Languages and Environments (Cat. No.01TH8587)*, pages 135–141.

Holly, M., Tscherko, J.-H., and Pirker, J. (2022). An Interactive Chess-Puzzle-Simulation for Computer Science education. In *2022 8th International Conference of the Immersive Learning Research Network (iLRN)*, pages 1–5.

Honda, F., Pires, F., Pessoa, M., and Oliveira, E. (2023). Automigos: Llearning Design para Ludificação de Autômatos Finitos Determinísticos. In *Anais do XXXI Workshop sobre Educação em Computação*, pages 545–556, Porto Alegre, RS, Brasil. SBC.

Hopcroft, J. E., Motwani, R., and Ullman, J. D. (2001). Introduction to Automata Theory, Languages, and Computation. *ACM Sigact News*, 32(1):60–65.

Hopcroft, J. E. and Ullman, J. D. (1969). *Formal Languages and their Relation to Automata*. Addison-Wesley Longman Publishing Co., Inc., USA.

Hung, T. and Rodger, S. H. (2000). Increasing Visualization and Interaction in the Automata Theory Course. *ACM SIGCSE Bulletin*, 32(1):6–10.

Isayama, D., Ishiyama, M., Relator, R., and Yamazaki, K. (2016). Computer Science Education for Primary and Lower Secondary School Students: Teaching the Concept of Automata. *ACM Trans. Comput. Educ.*, 17(1).

Jovanović, N., Miljković, D., Stamenković, S., Jovanović, Z., and Chakraborty, P. (2021). Teaching Concepts Related to Finite Automata Using ComVis. *Computer Applications in Engineering Education*, 29(5):994–1006.

Kiesmüller, U. (2009). Diagnosing Learners' Problem-solving Strategies Using Learning Environments with Algorithmic Problems in Secondary Education. *ACM Transactions on Computing Education (TOCE)*, 9(3):1–26.

Kitchenham, B. (2004). Procedures for Performing Systematic Reviews. *Keele, UK, Keele University*, 33(2004):1–26.

Knobelsdorf, M., Kreitz, C., and Böhne, S. (2014). Teaching Theoretical Computer Science Using a Cognitive Apprenticeship Approach. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, pages 67–72.

Korte, L., Anderson, S., Pain, H., and Good, J. (2007). Learning by Game-Building: A Novel Approach to Theoretical Computer Science Education. In *Proceedings of the 12th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, pages 53–57.

Maurer, P. M. (2021). Computer Science Theory from a Practical Point of View. In *2021 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 920–924.

McDonald, J. (2002). Interactive Pushdown Automata Animation. In *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education*, pages 376–380.

Mogensen, T. Æ. (2024). *Introduction to Compiler Design*. Springer Nature.

Mohammed, M., Shaffer, C. A., and Rodger, S. H. (2021). Teaching Formal Languages with Visualizations and Auto-Graded Exercises. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, SIGCSE '21, page 569–575, New York, NY, USA. Association for Computing Machinery.

Neto, J. and Terra, R. (2016). LFApp: Um Aplicativo Móvel para o Ensino de Linguagens Formais e Autômatos. In *Anais do XXIV Workshop sobre Educação em Computação*, pages 2196–2205, Porto Alegre, RS, Brasil. SBC.

Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., et al. (2021). The PRISMA 2020 Statement: An Updated Guideline for Reporting Systematic Reviews. *Bmj*, 372.

Petri, G., von Wangenheim, C. G., and Borgatto, A. F. (2016). MEEGA+: An Evolution of a Model for the Evaluation of Educational Games. *INCoD/GQS*, 3:1–40.

Procopiuc, M., Procopiuc, O., and Rodger, S. H. (1996). Visualization and Interaction in the Computer Science Formal Languages Course with JFLAP. In *Technology-Based Re-Engineering Engineering Education Proceedings of Frontiers in Education FIE'96 26th Annual Conference*, volume 1, pages 121–125. IEEE.

Robinson, M. B., Hamshar, J. A., Novillo, J. E., and Duchowski, A. T. (1999). A Java-based Tool for Reasoning About Models of Computation Through Simulating Finite Automata and Turing Machines. In *The Proceedings of the Thirtieth SIGCSE Technical Symposium on Computer Science Education*, pages 105–109.

Rodger, S. H., Bilska, A. O., Leider, K. H., Procopiuc, M., Procopiuc, O., Salemme, J. R., and Tsang, E. (1997). A Collection of Tools for Making Automata Theory and Formal Languages Come Alive. In *Proceedings of the Twenty-Eighth SIGCSE Technical Symposium on Computer Science Education*, pages 15–19.

Vieira, L. F. M., Vieira, M. A. M., and Vieira, N. J. (2004). Language Emulator, a Helpful Toolkit in the Learning Process of Computer Theory. *ACM Sigcse Bulletin*, 36(1):135–139.

Wagenknecht, C. and Friedman, P. D. (1998). Teaching Nondeterministic and Universal Automata Using Scheme. *Computer Science Education*, 8(3):197–227.