An Interactive Learning Tool for Teaching Sorting Algorithms

Ahmad R. Qawasmeh, Zohair Obead, Mashal Tariq, Motaz Shamaileh, Ahmad Shafee

Department of Computer Science
The Hashemite University
Zarqa, Jordan
Email: ahmadr@hu.edu.jo, zjaj.zm2006@gmail.com, mashalalmomani@yahoo.com, motazsh81@gmail.com, shafii4ever@gmail.com

***Abstract*—Sorting is a crucial process for managing data and it is used in many scientific fields. This paper presents a learning interactive tool that helps students to efficiently understand the concepts, design, and techniques of sorting algorithms. We developed a user-friendly Web application, based on some multi-media elements such as graphics interchange format, to describe the working process of different sorting algorithms in a simulation-based way. Six common sorting algorithms that include Bubble sort, Quick sort, Merge sort, Radix sort, Insertion sort, and Selection sort were implemented, analyzed, and demonstrated in this application. Our application simplifies the learning process of algorithms by giving the user an interactive and animated way for analyzing and understanding the design of sorting algorithms in a detailed manner. Based on a conducted study, the results of using our tool demonstrated an average improvement of 40% in the grades of students in the course "Introduction to Algorithms" compared with the students, who did not use this tool in two previous semesters. The tool also increased the students' motivation and willingness to take this course.**

Keywords–Algorithm visualization; Performance analysis; E-Learning; Sorting algorithms; Simulation tools.

I. INTRODUCTION

In mathematics and computer science, an algorithm is a step-by-step set of operations to be performed on specific input(s) to achieve specific output(s). Algorithms preform calculations, data processing, (and/or) automated reasoning tasks.

Due to the massive growth of modern learning methods, it has become easier to simplify the difficulty and complexity of computer operations related to algorithms, where the ability to view items visually is available in the modern software. The visual explanation of algorithms speeds up the process of understanding certain things that might need to be explained in detail. Different previous studies conducted in the teaching methods field argue that some students in Computer Science face difficulties understanding the abstract concepts of programming [1][2].

In this work, we came up with an idea of designing an educational application that provides visual elements and detailed explanation for six different sorting algorithms commonly used in computer science. The application offers a suitable environment for users (students and instructors) interested in the sorting algorithms field and an ability to simplify the learning process of how these algorithms actually work. Our application provides an interactive way for face to face and distance learning, detailed visual representation and simulation about the covered algorithms, convenient and friendly user interface, and easy way to electronically update the material of the course.

The main motivation of this work lies in the following points:

- Instructors need to have an interactive tool to deliver information to students of higher educational levels.
- Students need, rather than traditional books whether they are paper or electronic, an interactive system, which allows them to interact with the material so as to achieve the educational goal and makes learning interesting and exciting.
- The algorithm courses are based on experimental visual explanation. This fact makes the usage of visible elements and tools for delivering information a necessity.
- Sorting algorithms are based on different design paradigms and techniques. It is important to develop a tool that can visually depict the similarities and differences between these techniques in order to understand the drawbacks of using one algorithm over the other.

Figure 1 shows the main components of the proposed tool.

This paper is organized as follows. Section 2 discusses the related work. Section 3 describes the methodology and implementation. Section 4 presents our results and finding. Finally, Section 5 concludes our contributions and mentions directions for the future work.

II. RELATED WORK

Different Web applications [3]-[4] perform a visual way to simulate the process of sorting inputs in sorting algorithms. They provide an interactive environment to improve the ability to deliver information. The user can interact with the Web page to see how the algorithm works and how the process of sorting is virtually done. ALGAE [5] allows C++ or Java code to produce a simulated version of that code. Marcelino et al. [6] developed a tool that can be used to support initial stages of programming learning using a procedural approach. Krushkov et al. [7] provides a tutoring system for programming. Some of these applications only cover numeric values and ignore the other types of data. In our application, we simplified the design to improve users interaction with the application and developed an easy to use application with explanation about every algorithm.

Some other applications focused on analyzing how some of the most popular sorting algorithms work [8] [9]. These

 ISBN: 978-1-61208-599-9

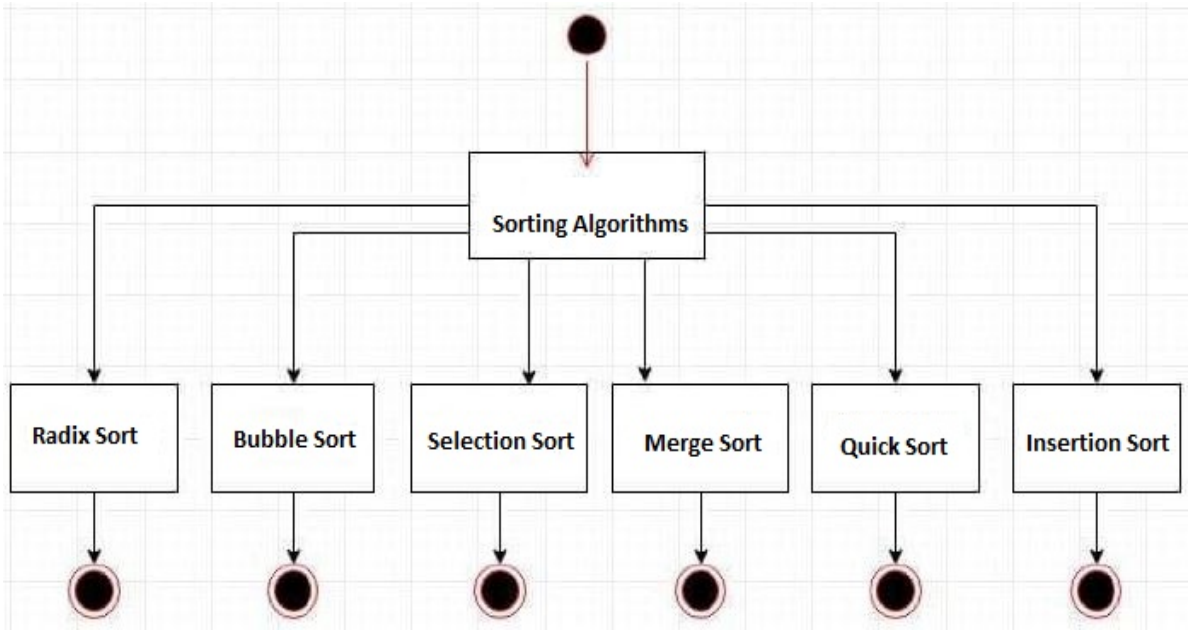

Figure 1. Components of the developed tool

applications provide two standpoints to look at algorithms: one is artistic, and the other is more analytical aiming to explain algorithms step by step. In contrast, our tool focuses on explaining how sorting algorithms work from a learning point of view.

In terms of teaching, Tuparov et al. [10] [11] developed and used an interactive simulation-based Learning Objects in an introductory course of programming focusing on sorting algorithms. Their study showed an increase in students' willingness and understanding. However, the main drawback of this work was the limited number of sorting algorithms, which were covered. TRAKLA2 [12] is a framework, which was developed for building interactive algorithm simulation exercises. Exercises constructed in TRAKLA2 are viewed as learning objects in which students manipulate conceptual visualizations of data structures in order to simulate the working of given algorithms. Grivokostopoulou et al. [13] presented an educational system that assists students in learning and tutors in teaching search algorithms. Automatic assessment was achieved in four stages, which constitute a general assessment framework. On the contrary, our tool focuses and covers a sufficient variety of sorting algorithms implemented via different design techniques.

On the other hand, Hundhausen et al. [14] presented a systematic meta-study of 24 experimental studies to examine the effectiveness of algorithm visualization in education. Their most significant finding was that how students use algorithm visualization technology has a greater impact on effectiveness than what algorithm visualization technology shows them. We worked on this finding in our tool by showing students how to use the tool efficiently to gain the most outcome. More details about the use of our tool are given later in the paper.

As can be obtained from the aforementioned discussion, our tool provides a sufficient variety of sorting algorithms, which covers both comparison and non-comparison based algorithms. In contrast, most of the current tools discuss comparison-based sorting algorithms only, especially when the tool is related to an introductory algorithms course. Moreover, our tool provides an easy interactive way, which motivates students and users to become more engaged with the topic of sorting algorithms.

There are other tools and applications that have been developed and used in the context of algorithms. However, we tried to focus on the most recent and relevant researches.

III. Methodology and Implementation

Because of the importance of design in any Web application, we tried to create an attractive and flexible interface that provides a convenient way of interaction for students/instructors. In other words, user can easily access our application and choose the type of sorting he/she wants to explore. Our system responds by giving the user an animated way to learn how the chosen algorithm sorts the input data, while displaying the algorithm's pseudo-code and observing the current variable values.

In our application, we used Hyper Text Markup Langaue (HTML), Cascade Style Sheet (CSS) and Java-script to design the front-end environment, while using Personal Home Page (PHP) for implementing the back-end functions and methods. For animation, we used the canvas library for the process of moving objects on the screen, then we added the corresponding algorithms to the animation code to create a simulation environment.

A. User Interface

The home page of our tool is divided into separate frames reachable easily by the user. By clicking on the "ALGORITHMS" tab, the algorithms page, shown in Figure 2, will be displayed. This page consists of six different algorithms.

Each sorting algorithm page has an explanation paragraph that gives the user an overview of the algorithm. We also inserted a graphic multimedia element, as shown in Figure 3 to give the user a visual explanation about the algorithm. At

 ISBN: 978-1-61208-599-9

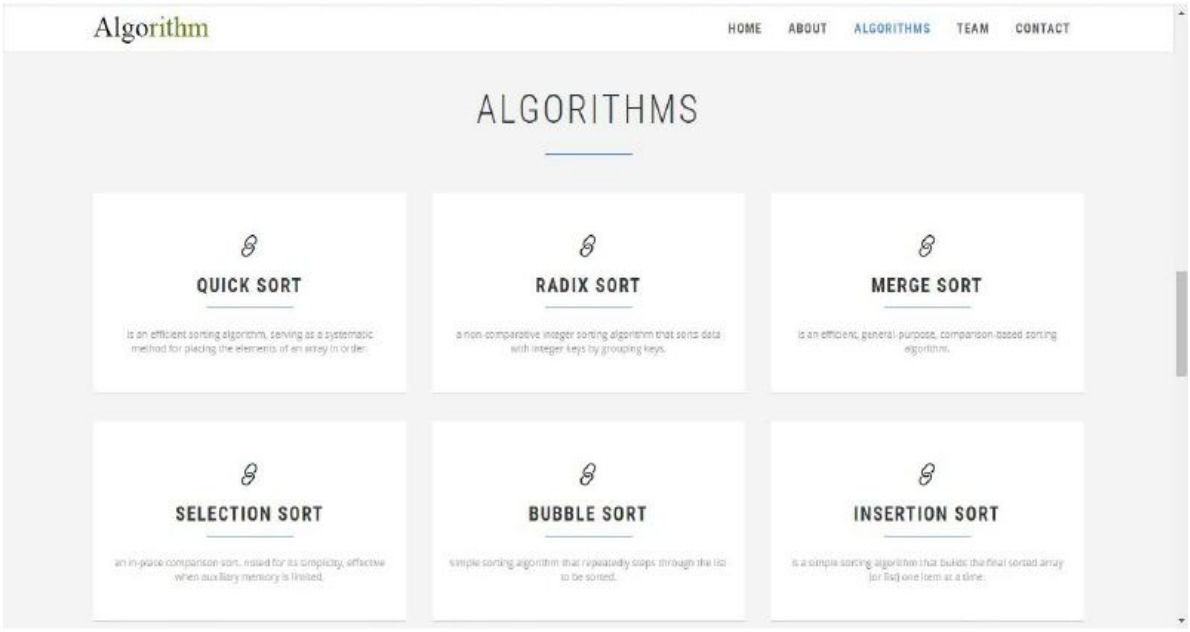

Figure 2. Sorting algorithms main page

the bottom of each page, we created a link, which transfers the user to the animation page.

For animation, we designed attractive simulation pages for each implemented algorithm. From a technical point of view, user can insert his own values or try the random numeric values given by the tool. After inserting the values and clicking on the sort button, the tool will start sorting the values according to the chosen algorithm. The tool gives the user a code tracker (pseudo-code) to make the process of understanding more efficient, while displaying the current variables values. The pseudo-code also describes the design paradigm used.

Figure 4 demonstrates the simulation process of the Quick sort algorithm. Each color assigned to the displayed bars has its own meaning depending on the used algorithm, and this is thoroughly explained in every algorithm page. The same animation format was used in all comparison-based sorting algorithms, demonstrated in our tool, that include bubble sort, selection sort, insertion sort, and merge sort. However, we developed another animation format for the non-comparison Radix sort algorithm, as shown in Figure 5. We included this type of sorting to help instructors explain the difference between comparison-based and non-comparison based sorting algorithms to their students more attractively.

The tool also has an input/output page that can track a flat file containing unsorted data. The tool returns the values sorted. This page can track any value type (numeric, character, string, etc.) and produce a file with sorted data. The output file can be downloaded to the local directory of the users computer.

New algorithms can easily be added to our learning tool, as the simulation environment has already been developed. We first need to modify the home page to add the corresponding new algorithm. Second, we need to add a graphic multimedia element page to give the user a visual explanation about the new algorithm. For animation, we just need to add the corresponding algorithm to the existing animation code.

B. A study for observing the efficiency of the tool

Our tool consists of the most popular algorithms: Bubble sort, Selection sort, Merge sort, Radix sort, Quick sort, and Insertion sort. A study was conducted in the undergraduate course “Introduction to Algorithms” in the college of Information Technology at the Hashemite University in Jordan on a sample of 100 students. This sample includes students in their second, third, and fourth academic year. The students of our sample also have different high school background, where two high school sections (1- Science 2- Information Technology) were considered. This variety in our sample ensures the correctness and accuracy of our findings. The main idea of this study was to compare the average grades of these students with the grades of students from two previous semesters, in which this tool was not available. We compared the results with two previous semesters, rather than one, to increase the accuracy and reliability of our study. The tool was available to students for self-learning and was also used during the face to face lectures and office hours. The instructor demonstrated the simulation with different input data to show students how to obtain the best case, average case, and worst case of each algorithm, while explaining the changes of colors and values of variables.

IV. RESULTS

Our observations through the conducted study performed in the “Introduction to Algorithms” course demonstrated promising results. The use of our simulation-based tool as an alternative to the traditional way of explaining algorithms was really efficient and reliable. First of all, the students were impressed by the use of the interactive tool. The element that was the most interesting to them was the ability to observe the behavior of every algorithm, while changing the input data. The students had the chance to understand the impact of input data on the complexity of algorithms. The students also managed to implement the pseudo-code of these

 ISBN: 978-1-61208-599-9

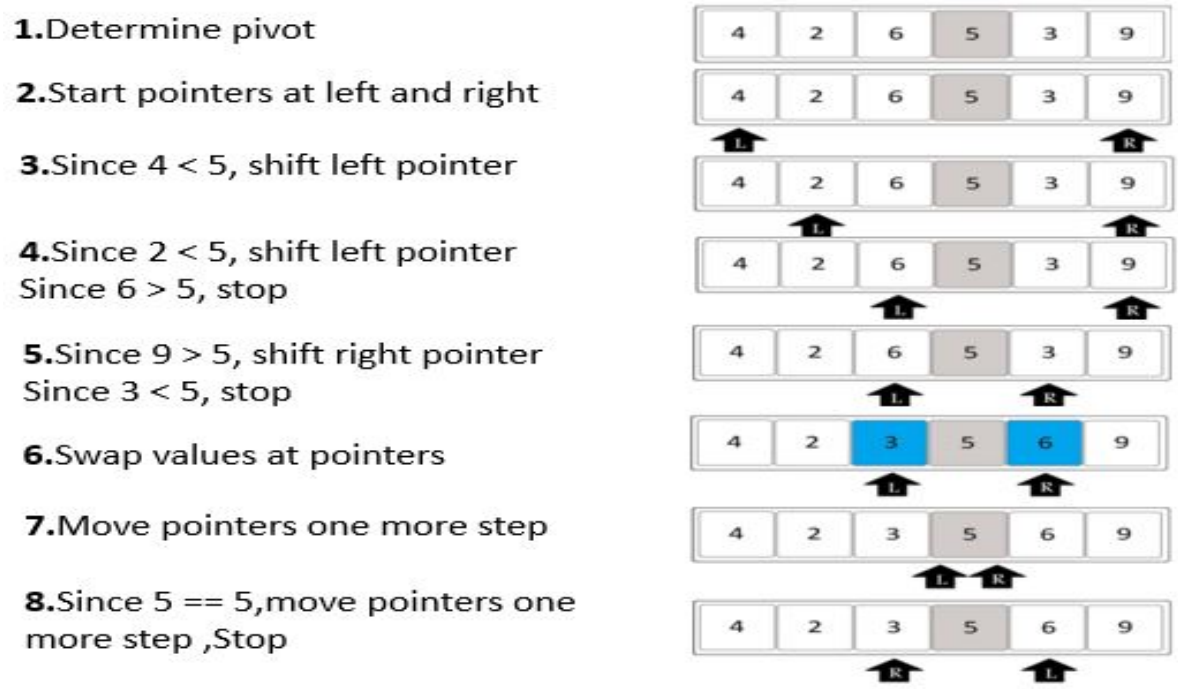

Figure 3. Quick Sorting multimedia page

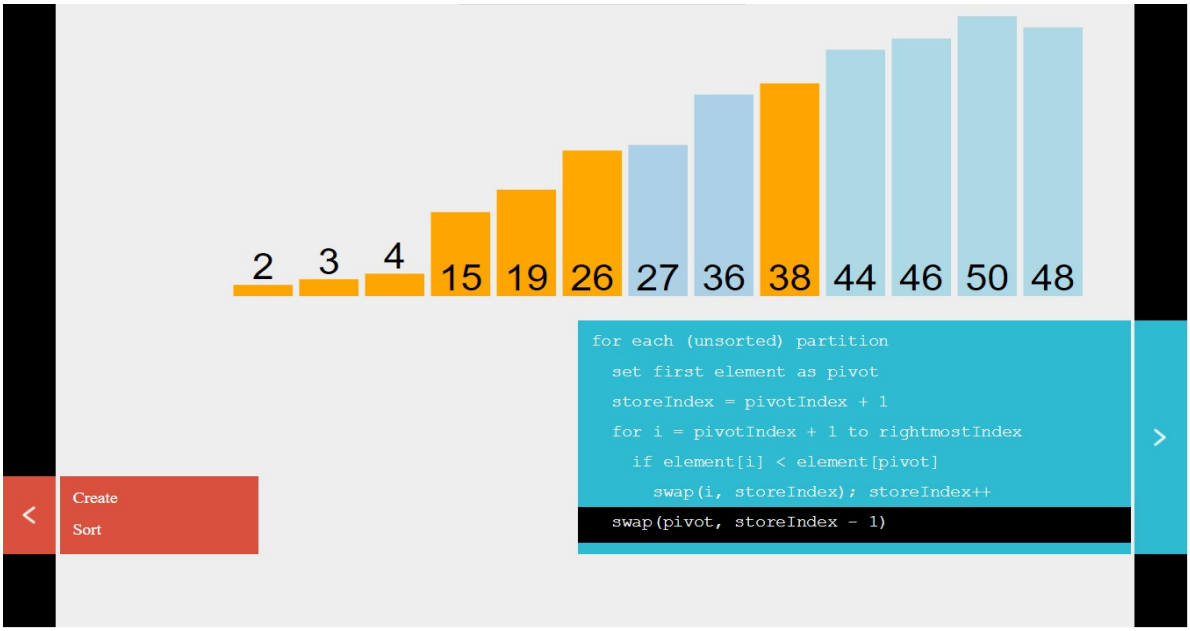

Figure 4. Quick Sorting animation page

algorithms more easily using different programming languages such as java or C++. The simulations of algorithms also helped the students to understand the different design paradigms used to implement the covered algorithms and how they can be adopted to implement other algorithms. The percentage of students participating in the discussions during lectures significantly increased compared with the previous semesters, in which the simulation-based tool was not used.

Furthermore, the grades of the current students were significantly better than the students who took the course previously without using the tool. The results demonstrate an average improvement of 40% in the grades of students in the exam covering the sorting topic in the course “Introduction to Algorithms” compared with the students who did not use this tool in two previous semesters. More students are also interested in taking this course in the summer semester based on the observations from the initial registration period.

At the end of next semester, we plan to conduct a survey,

 ISBN: 978-1-61208-599-9

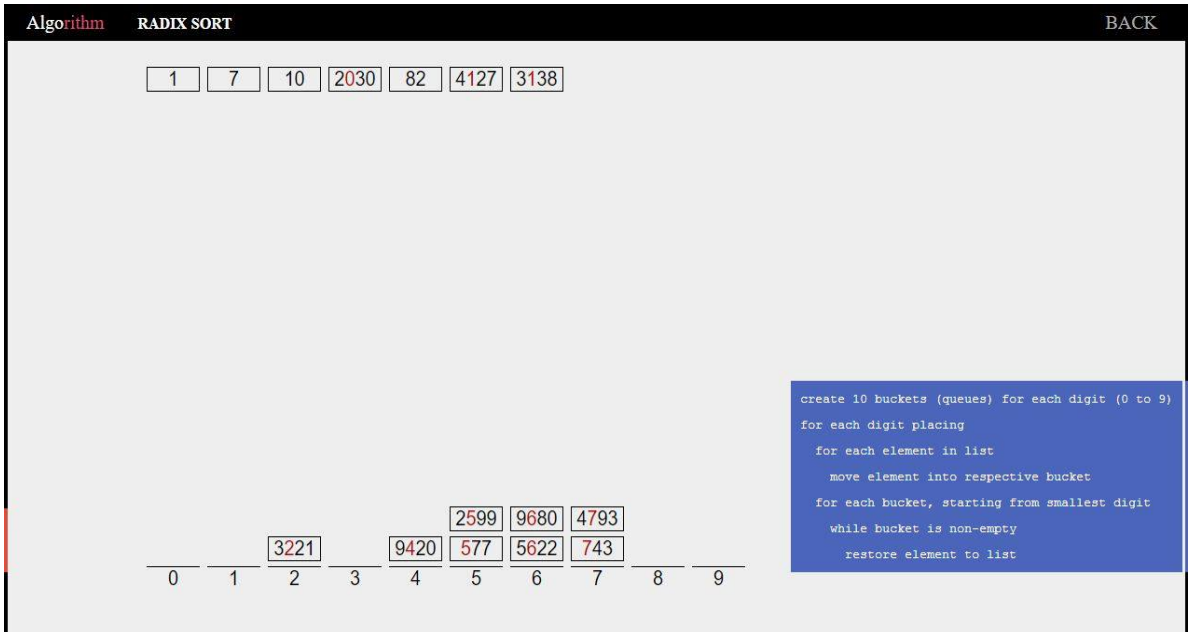

Figure 5. Radix Sorting animation page

directed to students, to have a better idea about using the simulation-based tool in teaching. This shall be an important step to get students' feedback about how to improve the tool and if it is useful to simulate other algorithms that solve different problems.

V. Conclusions and Future Work

The application proposed in this paper was designed as a learning tool that can be used by instructors/students to facilitate the process of teaching sorting algorithms. The tool provides an interactive environment and meets the requirements of modern learning. The tool has an animated explanation for the process of sorting via different sorting algorithms that vary in terms of usage, design technique, best/worst case running time complexity, and concept. The tool is compatible with different browsers and can handle different input data types. Last but not least, the tool demonstrated promising results based on a conducted study performed on a sufficient sample of students.

Path forward, we plan to simulate the working process of other algorithms depending on a survey that will be conducted soon. We also think that making this tool available on mobile phones will improve learning.

Acknowledgment

The authors would like to thank the Hashemite University for providing the required resources.

References

[1] C. M. Areias, A. J. Mendes, and A. J. Gomes, "Learning to program with proguide," in Proc. Int. Conf. Engineering Education, 2007.

[2] G. Rößling, "A family of tools for supporting the learning of programming," algorithms, vol. 3, no. 2, 2010, pp. 168–182.

[3] visualgo.com, "Algorithm Visualisation," http://visualgo.com, 2017, accessed: 01-01-2017.

[4] University of San Francisco, "Sorting visualization, comparison sorting algorithm," http://cs.usfca.edu, accessed: 13-11-2016.

[5] Steven J. Zeil, "AlgAE (Algorithm Animation Engine)," http://www.cs.odu.edu/~zeil/AlgAE/referenceManual.pdf, 2011, accessed: 13-11-2016.

[6] M. Marcelino, T. Mihaylov, and A. Mendes, "H-sicas, a handheld algorithm animation and simulation tool to support initial programming learning," in Frontiers in Education Conference, 2008. FIE 2008. 38th Annual. IEEE, 2008, pp. T4A–7.

[7] H. Krushkov, M. Krushkova, V. Atanasov, and M. Krushkova, "A computer-based tutoring system for programming," Mathematics and Mathematical Education, 2009.

[8] Aldo Cortesi, "Sorting Algorithm Visualisation," http://sortvis.org, 2010, accessed: 13-11-2016.

[9] Carlo Zapponi, "Sorting," http://soting.at, 2014, accessed: 13-11-2016.

[10] G. Tuparov, D. Tuparova, and V. Jordanov, "Teaching sorting and searching algorithms through simulation-based learning objects in an introductory programming course," Procedia-Social and Behavioral Sciences, vol. 116, 2014, pp. 2962–2966.

[11] G. Tuparov, D. Tuparova, and A. Tsarnakova, "Using interactive simulation-based learning objects in introductory course of programming," Procedia-Social and Behavioral Sciences, vol. 46, 2012, pp. 2276–2280.

[12] L. Malmi et al., "Visual algorithm simulation exercise system with automatic assessment: Trakla2," Informatics in education, vol. 3, no. 2, 2004, p. 267.

[13] F. Grivokostopoulou, I. Perikos, and I. Hatzilygeroudis, "An educational system for learning search algorithms and automatically assessing student performance," International Journal of Artificial Intelligence in Education, vol. 27, no. 1, 2017, pp. 207–240.

[14] C. D. Hundhausen, S. A. Douglas, and J. T. Stasko, "A meta-study of algorithm visualization effectiveness," Journal of Visual Languages & Computing, vol. 13, no. 3, 2002, pp. 259–290.

 ISBN: 978-1-61208-599-9