C·O·L
CEMCA

Virtual Labs for Data Structures: Search and Mergesort

(Final Report submitted to CEMCA)

Venkatesh Choppella
IIIT Hyderabad

[2021-06-15 Tue]

1 Introduction

The present report is a summary of the activities done under the Virtual Labs for Data Structures project supported by CEMCA, New Delhi for the period 15 Feb 2021 – 15 Jun 2021.

The objective of the project was to design, develop and temporarily host a collection of interactive experiments for the following topics:

1. **Search**: Carrying out search interactively in an array using a variety of strategies.
2. **Mergesort**: Carry out merge and mergesort interactively in an array using a variety of strategies.

The rest of the report is organised as follows. First, we list the timeline of the project along with links to relevant documents (Section 2). Then we briefly discuss the philosophy and pedagogy driving this particular effort of Virtual Labs and contrast it with other existing approaches (Section 3). We then list the experiments contained in each of the laboratory topics, Search (Section 4) and Mergesort (Section 5). We summarise the activities completed in 4th Phase of the project (Section 6). We also include feedback from a presentation, demo and discussion about the Virtual Labs in early Jun (Section 7) We conclude with some pointers to the next steps and future explorations related to Virtual Labs (Section 8).

2 Project Timeline and links to documents

The project officially commenced on 15th February 2021 with a duration of four months. The project was divided into four phases, with milestones for each phase. These are summarised in Table 1.

The final list of artefacts for delivery are listed in Table 2. These include hosted URLs and source codes of the experiments in each of the two topics.

Table 1: Activity timeline of the Project

No.	Activity	Document	Event Date
1	Submission of Phase 1 Requirements Gathering report	pdf (search) pdf (mergesort)	*[2021-03-16 Tue]*
2	Submission of Phase 2 Design report	pdf (search) pdf (merge)	*[2021-04-19 Mon]*
3	Phase 3 Prototype implementation (presentation)	pdf (presentation)	*[2021-05-27 Thu]*
4	Presentation to Open University Directors/Heads of Computer Science	pdf (presentation)	*[2021-06-09 Wed]*
5	Phase 4 Improvements and Final report	pdf (this document)	*[2021-06-15 Tue]*

Table 2: Hosted URLs and source code for the Virtual Labs Experiments

Topic	Hosted URL
Search	
Hosted URL	http://algodynamics.io/search/
Source code Repository	https://github.com/algodynamics-iiith/merge-sort/releases/tag/v1.0.1
Mergesort	
Hosted URL	http://algodynamics.io/mergesort/
Source code Repository	https://github.com/algodynamics-iiith/search/releases/tag/v1.0.0

3 Philosophy and Pedagogy

Virtual labs are a natural fit within the worldwide movement towards online learning, dramatically accelerated due to the recent and ongoing Covid-19 pandemic. The author of this report has already been involved in a large virtual labs effort sponsored by the Government of India[1].

The justification for building Virtual Labs for computer science subjects needs to be made carefully. Computing is a process that is already virtual and does not correspond to any physical reality. However, the representation of computing processes (as evident in algorithms, for example) via visual elements has essential value as demonstrated by several online animations of algorithms available.

However, insight gained through animation and tracing an algorithm is limited. Better insight comes with interaction ('Learning by doing'). Interaction implies taking control of the algorithm and steering it to the answer by employing a problem solving strategy not necessarily dictated by the algorithm. For example, an algorithm may employ a left-to-right traversal strategy to search for an element in an array, while a student may want to explore a right-to-left or a random strategy. Here we are faced with the following challenge. Algorithms are, by definition, non-interactive. All 'input' to an algorithm is used to initialise it. Once an algorithm starts, interaction is forbidden. By 'opening up' an algorithm for interaction, we automatically step outside the realm of algorithms and enter the space of interactive transition systems. This insight is the starting point of our approach to building the virtual labs for data structures. There are two benefits of using transition systems as a starting point. First, they provide a uniform high-level model for expressing the interactive systems. These models also guide the implementation of the experiments as web applications. Second, a sequence of successive refinements of these models demonstrate the gradual development of these models, an idea propounded by Niklaus Wirth, but in the context of program development [2]. Each transition system isolates and illus-

trates a specific aspect of the algorithm. Interacting with the experiment realising that system allows the student to discover a particular strategy for solving the given problem. At each stage, interactivity is traded for automation of the previous strategy.

Data Structures is an elementary course in Computer Science usually done in the 2nd year by students of Computer Science. In many unviersities, it is also very popular amongst non-computer science majors as it forms the first solid introduction to computer science after the basic programming course.

The data structures course is usually accompanied by a laboratory in most universities. However, the focus of the laboratory is usually programming, where the student is expected to code the algorithm given in the textbook or introduced in the lecture. There is no scope for the student to experiment with the algorithm, 'open' it up and interactively solve the algorithmic problem.

The approach taken in this project is motivated by the need to fill this gap between the lectures and programming. The interactive labs that we have built are expected to be done after the lectures but before embarking on the exercise of coding the algorithm. The interactive experiments help the students gain the insight and confidence with the strategies that she would then employ in the coding effort.

4 Experiments for Search

The suite of search experiments explore how a given number may be located within an array of elements (numbers). The number may or may not be present in the array. Four experiments have been built under this topic. They are listed in Table 3.

Table 3: List of Search Experiments

No.	Experiment	URL
1.	Random Search with replacement	https://algodynamics.io/search/RandomWithReplacement.html
2.	Random Search without replacement	https://algodynamics.io/search/RandomWithoutReplacement.html
3.	Linear Search	https://algodynamics.io/search/LinearSearch.html
4.	Binary Search	https://algodynamics.io/search/BinarySearch.html

5 Experiments for Mergesort

The suite of experiments for mergesort explore how a list of numbers may be sorted using mergesort. The first two (1 and 2) experiments introduce the basic operation in mergesort, the shuffle. The next two (3 and 4) show how the merge operation may be seen as a shuffle based on a particular strategy ('picking' the smaller element to shuffle). The next experiment (5) employs merge to sort a list of numbers by proceeding linearly along the list to look for sublists to merge, rather like TimSort, but simpler. Experiment 6 is an interactive rendition of the the recursive mergesort. Experiment 7 resembles 6 in that the same basic operations — split and merge — are used, but with the freedom to merge any two sorted sublists. They experiments are listed in Table 4.

Table 4: List of Mergesort Experiments

No.	Experiment	URL
1.	Shuffle	http://algodynamics.io/mergesort/merge/mergesystem.html
2.	Shuffle (sorted)	http://algodynamics.io/mergesort/merge/shuffleSorted.html
3.	Merge Strategy	https://algodynamics.io/mergesort/merge/msStrategy.html
4.	Merge Algorithm	https://algodynamics.io/mergesort/merge/msAlgo.html
5.	Merge sublists	https://algodynamics.io/mergesort/merge.html
6.	Mergesort (Recursive)	https://algodynamics.io/mergesort/recursive.html
7.	Mergesort (Arbitrary)	https://algodynamics.io/mergesort/msarbitrary.html

6 Improvements and Development done in Phase 4

By Phase 3, basic prototypes of all the search and mergesort experiments was completed. In Phase 4, the following work was accomplished:

1. Improvement in UI based on discussions accompanying Phase 3 demo.
2. Translation of code base from Elm to Javascript (for some experiments).
3. Initial prototype of experiment analytics (for the Search experiments).
4. Hosting and source code repository release.
5. Final Report writing (this report).

7 Presentation, Demo, Discussion and Feedback about Virtual Labs

A presentation of the Virtual Labs was made to Directors and Heads of Computer Science departments of some of the Open Universities in India on 9th Jun 2021. The presentation was followed by a vigorous discussion on how to improve the scope of Computer Science Virtual Labs in general, and also what steps should be taken to continue the present line of work. It was felt that the Virtual Labs should be designed to better cover the existing syllabi of contemporary computer science courses. The general opinion from the group that more experiments need to be built. Furthermore, some of the experiments involving programming should be

supported by cloud based compilers. In the pedagogy front, it was suggested that Transition Systems should be introduced in the syllabus and that teacher workshops should be run to train teachers in the transition system and the Algodynamics pedagogy.

8 Conclusion

Over the coming months and years, Virtual Labs are poised to play an increasingly central role in online education to alleviate the problems of both access and quality. The Data Structures Virtual Labs are but a small, yet refreshingly new beginning in the direction of making interactive experiments in computer science available to students worldwide. Many more experiments in Data Structures and Algorithms and other areas of computing need to be explored, designed and implemented. In addition, these experiments need to be accompanied by a set of workshops that introduce Algodynamics, the theoretical framework underlying the Virtual Labs. These workshops would also illuminate the connection between the theory, interactive experiments and the implementation of algorithms.

9 Bibliography

References

[1] Virtual labs for science and engineering. http://vlab.co.in, 2020. (accessed 01-01-2020).

[2] WIRTH, N. Program development by stepwise refinement. In *Pioneers and Their Contributions to Software Engineering*. Springer, 2001, pp. 545–569.