Virtual Labs for Data Structures: An Algodynamic Approach

Venkatesh Choppella
9th Jun, 2021
IIIT Hyderabad
Dedication

Virtual Labs

Algodynamics: Algorithms as Systems

Bubblesort

Search Interactive Demo

Mergesort Virtual Lab

Conclusion
Contents

Dedication

Virtual Labs

Algodynamics: Algorithms as Systems

Bubblesort

Search Interactive Demo

Mergesort Virtual Lab

Conclusion
Dedication: Kesav V Nori

13th December 1945 – 29th May 2021

- IIIT Hyderabad (Distinguished Professor), IIT Hyderabad, IIT Kanpur, Pune Univ.
- Executive Vice President TCS
- Founding Head, Tata Research Design and Development Centre (TRDDC) Pune
Contents

Dedication

Virtual Labs

Algodynamics: Algorithms as Systems

Bubblesort

Search Interactive Demo

Mergesort Virtual Lab

Conclusion
What are Virtual Labs?

1. Online simulation experiments

2. Over 1100 experiments in a dozen disciplines

3. Developed over the last 10 years. Sponsored by Ministry of Education.

4. http://vlab.co.in
Virtual Labs Usage since 1st Jan 2020

1. **Online Analytics**

2. 42 Million+ Views.

3. 6 Million Users

4. **Data Structures-I** and **Data Structures-II** have a combined cumulative views of 1 Million+. 
• How do Virtual Labs Enhance Learning?

• Can the act of designing and doing experiments affect the way we think about a subject?
1. Bubblesort on Wikipedia

2. Bubblesort on Youtube
Bridging the gap between Understanding and Implementing

1. Theory
2. Virtual Labs
3. Code

1. Modeling
2. Interacting
3. Implementing
Contents

Dedication

Virtual Labs

Algodynamics: Algorithms as Systems

Bubblesort

Search Interactive Demo

Mergesort Virtual Lab

Conclusion
The fundamental question in any branch of engineering is “how does it work?” The fundamental question in any branch of any engineering science is “why does it work?” These are exactly the questions of software engineering and computer science. We must therefore, inspire our students with curiosity about these questions, and then satisfy it. ...

... Perhaps they don’t even need to know how these systems work, but in a university they should be taught all the same, because one day it might help.

— C. A. R. Hoare

Quoted in Sec 2.2.3

Grand Challenges in Computing — Education

British Computer Society, 2004
<table>
<thead>
<tr>
<th>Engine</th>
<th>Flow</th>
<th>Engineering Science</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heat Engines (Locomotives)</td>
<td>Heat flow</td>
<td><em>Thermodynamics</em></td>
</tr>
<tr>
<td>Jet Engines (Turbines)</td>
<td>Fluid flow</td>
<td><em>Fluid Dynamics</em></td>
</tr>
<tr>
<td>Electromagnetic Engines</td>
<td>Current flow</td>
<td><em>Electrodynamics</em></td>
</tr>
<tr>
<td>(Generators/Motors)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computational Engines</td>
<td>Information flow</td>
<td><em>Algodynamics</em></td>
</tr>
<tr>
<td>(Algorithms/Apps)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. **Computer** Organization

2. **Computer** Networks

3. **Computer** Graphics

But also

1. Operating **Systems**

2. Database **Systems**

3. Information **Systems**

Not to mention other disciplines

1. Eco**Systems**

2. Transportation **Systems**

3. Education **Systems**

4. etc.
What is a system?

1. **Observations:** Quantities that may be sensed/seen

2. **Behaviour:** Traces of observations

3. **State:** Internal machinery

4. **Display:** Dashboard

5. **Actions:** Controls that affect state

6. **Dynamics:** Laws that decide how an action affects a state
## Continuous vs. Discrete Systems

<table>
<thead>
<tr>
<th></th>
<th>Continuous</th>
<th>Discrete</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time</strong> $t$</td>
<td>$\mathbb{R}$ or $\mathbb{R}^+$</td>
<td>$\mathbb{Z}$ or $\mathbb{N}$</td>
</tr>
<tr>
<td><strong>State</strong> $x$</td>
<td>$x(t) : \mathbb{R}$</td>
<td>$x_n : X$</td>
</tr>
<tr>
<td><strong>Change of state</strong></td>
<td>Rate</td>
<td>Next state</td>
</tr>
<tr>
<td></td>
<td>$\dot{x} \ (dx/dt)$</td>
<td>$x' \ (x_{n+1})$</td>
</tr>
<tr>
<td><strong>Dynamics</strong></td>
<td>$\dot{x} = F(x)$</td>
<td>$x' = F(x)$</td>
</tr>
<tr>
<td><strong>Differential Equations</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Transition Equations</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Discipline</strong></td>
<td>Science</td>
<td>Computing</td>
</tr>
</tbody>
</table>
## Closed vs. Open Systems

<table>
<thead>
<tr>
<th></th>
<th>Closed</th>
<th>Open</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td>Autonomous</td>
<td>Interactive</td>
</tr>
<tr>
<td><strong>Action set</strong></td>
<td>Singleton</td>
<td>No restriction</td>
</tr>
<tr>
<td><strong>Dynamics</strong></td>
<td>( x' = F(x) )</td>
<td>( x \overset{u}{\rightarrow} x' )</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>Algorithms</td>
<td>Apps</td>
</tr>
<tr>
<td><strong>Discipline</strong></td>
<td>(Computer) Science</td>
<td>(Software) Engineering</td>
</tr>
</tbody>
</table>
1. Iterative System: $\langle X, F : X \to X \rangle$

2. Trajectory: $x_0, F(x_0), F^2(x_0), \ldots$

3. Fixed point: $x = F(x)$
def loop(x, F):
    while True:
        next_x = F(x)
        if next_x = x:  # fixed point!
            return x
        else:
            x = next_x
1. Specify $X$: State Space

2. Specify $F$: Transition function

3. Specify $x_0$: initial state
Bubblesort as an initial value problem.

1. **State space**: \( x : X = (i, b, a) : (\mathbb{N}, \mathbb{N}, (\text{Array}(\mathbb{N}))) \)

2. **Initial state**: \((a^0, 0, |a^0|)\)

3. **Transition Equation**: \((a', i', b') = F(a, i, b)\)
Bubblesort as an initial value problem.

1. **Transition Function**:

   \[ F(i, 1, a) = (i, b, a) \]
   \[ F(i, b, a) = (i, b, a) \quad i \geq b \]
   \[ F(b - 1, b, a) = (0, b - 1, a) \]
   \[ F(i, b, a) = (i + 1, b, a) \quad a_i \leq a_{i+1} \]
   \[ F(i, b, a) = (i + 1, b, \text{swap}(a, i, i + 1)) \quad a_i > a_{i+1} \]
Bubblesort Example Run

\[
\begin{align*}
[6 \ 8 \ 7 \ 4 \ | \ ] & \rightarrow \\
[6 \ 8 \ 7 \ 4 \ | \ ] & \rightarrow \\
[6 \ 7 \ 8 \ 4 \ | \ ] & \rightarrow \\
[6 \ 7 \ 4 \ 8 \ | \ ] & \rightarrow \\
[6 \ 7 \ 4 \ | \ 8 ] & \rightarrow
\end{align*}
\]

\[
\begin{align*}
[6 \ 7 \ 4 \ | \ 8 ] & \rightarrow \\
[6 \ 4 \ 7 \ | \ 8 ] & \rightarrow \\
[4 \ 6 \ | \ 7 \ 8 ] & \rightarrow \\
[4 \ | \ 6 \ 7 \ 8 ] & \text{fixed point}
\end{align*}
\]
Bubblesort as an initial value problem.

1. **State space:** $X = Array(\mathbb{N})$

2. **Initial state:** $a^0$

3. **Actions:** $\text{swap}(i, j)$,

4. **Transition Relation:**

   $$a \xrightarrow{\text{swap}(i, j)} a' \quad \text{iff} \quad 0 \leq i, j < |a|$$

   and $a' = \text{swap}(a, i, j)$
Interactive run of Bubblesort

\[ \begin{array}{c}
\text{[8 6 4 7]} \xrightarrow{\text{swap(0,2)}} \\
\text{[4 6 8 7]} \xrightarrow{\text{swap(2,3)}} \\
\text{[4 6 7 8]} \end{array} \]
Bubblesort Demo
Search Demo
Mergesort Demo
Contents

Dedication

Virtual Labs

Algodynamics: Algorithms as Systems

Bubblesort

Search Interactive Demo

Mergesort Virtual Lab

Conclusion
1. Learning by Doing is a well-known principle in Learning Science.

2. Virtual Labs encourage learning by doing

3. Virtual Labs correspond to Open Systems

4. The systems view of Computer Science is Algodynamics

5. Focus on Modelling, Interaction along with coding.