Teaching Parallel and Distributed Computing Concepts in Simulation with WRENCH

Ryan Tanaka¹  Rafael Ferreira da Silva¹
Henri Casanovas²

¹ Information Science Institute / USC, USA
² University of Hawaiʻi at Mānoa, USA
Teaching Challenges

- Teaching Parallel and Distributed Computing (PDC) and High Performance Computing (HPC) concepts in Computer Science curricula should be done more and earlier.
- Teaching everything “on the blackboard” is not effective, and students should learn in a **hands-on manner**.
- One option: provide students access with some hardware and software platform to learn/apply PDC and HPC concepts.
  - e.g., some on-campus cluster.

This comes with challenges!
REAL PLATFORMS: PARTICIPATION CHALLENGE

- An institution may not have an adequate platform
  - Or none readily available for teaching purposes
- There are several solutions:
  - Build a low-cost platform (e.g., raspberry pies, clusters of SoCs)
  - Use virtualization/container technology (e.g., locally, in some cloud)
- But all of these limit what can be done/learned because of their specs and scales
REAL PLATFORMS: PEDAGOGIC CHALLENGES

- Real-world stuff gets in the way of learning
  - Possibly intricate platform access mechanisms and policies
  - Platform downtimes (planned or unplanned)
  - Competition for the platform among students and with other users
  → Class and instructor time not spent on learning objectives

- Platform’s specifics get in the way of learning
  - “If we had more cores, then this would happen…”
  - “If the network was different, then this wouldn’t work as well…”
  - “If we had less RAM, then this would break…”
  → Many learning objectives cannot be achieved hands-on
SIMULATION AS AN ALTERNATIVE

- With **simulation**: no need for an actual platform, any arbitrary platform configuration, perfect repeatability, quick executions
- Used routinely for teaching in some areas of Computer Science (architecture, network)
- Time-and-again proposed and used for PDC/HPC education since the early 1990s
- Typically used with a “simulate and observe” strategy
  - Simulating the execution of code provided to students and that they cannot modify
  - Simulating the execution of code written by students, allowing them to develop/debug/run all in simulation
GOAL

- Develop a set of pedagogic modules that...
  1. Target standard HPC/PDC Student Learning Objectives
  2. Can be integrated piecemeal in existing courses starting at freshman levels
  3. Rely on simulation to provide students with hands-on, interactive learning opportunities without need for any hardware platform

- All developed as part of the WRENCH project...
To implement our pedagogic modules, we need to develop simulators. These simulators should be scalable and accurate. The SimGrid simulation framework has striven to provide both scalability and accuracy for more than a decade, so let’s build on it…
To implement our pedagogic modules, we need to develop simulators. These simulators should be scalable and accurate. The SimGrid simulation framework has striven to provide both scalability and accuracy for more than a decade, so let’s build on it...
To implement our pedagogic modules, we need to develop simulators. These simulators should be scalable and accurate. The SimGrid simulation framework has striven to provide both scalability and accuracy for more than a decade, so let’s build on it… But SimGrid provides low-level abstractions, and thus writing simulators can be labor-intensive.
WRENCH

- WRENCH builds on top of SimGrid to provide easy, high-level simulation abstractions.
- Therefore, we can now have simulators that are accurate, scalable, and easy to develop.
- Onward to “WRENCH Pedagogic Modules”

Simulated core CI services

Simulated low-level software / hardware stacks

WRENCH Developer API (C++)

SimGrid::S4U API (C++)
THE WRENCH PEDAGOGIC MODULES

- Each module has:
  - A set of learning objectives and a narrative
  - One or more simulators that students can execute
  - Guided, practice, and open-ended questions

- The simulators are used by students in various modes:
  - Run-and-observe
  - Run-to-verify-expectations
  - Run-to-discover-answers

- Students only need a browser and Docker
CURRENTLY AVAILABLE MODULES

Principles of Computing and Distributed Computing

- SINGLE-CORE COMPUTING
  speed, work, RAM

- MULTI-CORE COMPUTING
  speedup, efficiency, idle time

- I/O
  HDD/SSD, data rates, overlap with computation

- NETWORKING
  latencies, bandwidth, topologies, contention

Applying Principles To Workflows

- SCIENTIFIC WORKFLOWS
  Basic concepts

- WORKFLOWS AND DATA LOCALITY
  network proximity of data

- WORKFLOWS AND PARALLELISM
  multi-core, multi-node

- PROVISIONING RESOURCES
  meeting performance goals within budget
Currently Available Modules

Principles of Computing and Distributed Computing

**SINGLE-CORE COMPUTING**
speed, work, RAM

**MULTI-CORE COMPUTING**
speedup, efficiency idle time

**I/O**
HDD/SSD, data rates, overlap with computation

**NETWORKING**
latencies, bandwidth, topologies, contention

Applying Principles To Workflows

**SCIENTIFIC WORKFLOWS**
Basic concepts

**WORKFLOWS AND DATA LOCALITY**
network proximity of data

**WORKFLOWS AND PARALLELISM**
multi-core, multi-node

**PROVISIONING RESOURCES**
meeting performance goals within budget

Let’s look at one module…
SAMPLE MODULE: WORKFLOW AND PARALLELISM

Students are shown a platform and an application at a high level.
Students then learn about specifics
Students are able to try different specs and simulate application execution.
SAMPLE MODULE: WORKFLOW AND PARALLELISM

Execution Gantt chart for all tasks
SAMPLE MODULE: WORKFLOW AND PARALLELISM

Fig. 2. Simulated hardware specification of the cluster hosted by the Compute Service for Activity (IV).

Fig. 3. Simulator input panel for Activity (IV).

Fig. 4. Sample Gantt chart of task executions for Activity (IV) given the input shown in Figure 3.

Fig. 5. Sample core utilization time-line for Activity (IV) given the input shown in Figure 3.

for participating was a desire to learn more about a crucial topic that is often not (sufficiently) taught in the standard curriculum. Students were given a reading assignment in which they had to perform activities (I) and (II) on their own. A week later, in a 1-hour session, the pedagogic team (the first and last author of this paper) together with the students did activity (III), answering questions based on group discussion. A week later, in another 1-hour session, the participants did as many questions as possible in activity (IV), with the pedagogic team providing necessary scaffolding.

Students were given pre and post knowledge tests, as well as questionnaires about their experience. The purpose was mostly to identify potential improvements in the tests and questionnaires themselves, and in the pedagogic content. Based on participant feedback we made several content changes: (i) we removed content that proved too detailed and detrimental to learning; (ii) we added text and figures to clarify points of confusion; (iii) we improved simulated execution visualizations to include more information. In spite of these students being presented with a preliminary version of our activities, feedback was overall very positive and students felt that they acquired new knowledge easily.

B. Classroom Evaluation

The last author of this paper regularly teaches the undergraduate Operating Systems (ICS 332) course at UHM. A PDC module was added to the course syllabus for the Spring 2019 semester, which consists of the following steps:

1) A 30-minute lecture on PDC to motivate the topic;
2) A reading assignment in which students performed activities (I) and (II) on their own;
3) A 75-minute in-class interactive session during which the pedagogic team did activity (III), soliciting participation from students and fielding questions;
**Sample Module: Workflow and Parallelism**

- **Sample Questions #1:** Assuming the cluster has 4 8-core compute nodes, what can we expect the execution time of the workflow to be? Write a simple formula. Run the simulation and check your results against the simulator.

- **Sample Question #2:** Assuming that you can add an arbitrary number of 5-core nodes, with the same per-core compute speed, is it possible to decrease the workflow execution time? Why or why not?
These modules were used in the ICS332 course at UH Manoa in Spring 2019
• And will be used next week again!

Students were given:
• A 30-minute lecture on PDC
• A reading assignment in which students did foundational modules on their own
• Two 75-minute in-class interactive sessions, going through modules with instructor scaffolding
• A homework that consisted in completing the 2nd half of one of the workflow modules
• Three final exam questions on these topics (10% of the exam grade)
**In-Class Evaluation (2)**

In the evaluation we gathered:
- Anonymous post questionnaire about the modules and about perceived learning
- Anonymous pre and post knowledge tests
- Non-anonymous grades for homework and exam questions
- Non-anonymous time-stamps of simulation activities

What we don’t have: a control group that does not use simulation
- Unclear how that would be feasible/fair
WHAT WE LEARNED (1)

- Students are using the simulation
- 45 out of 55 students ran simulations (22 times on average)
- 40% of simulations were for input settings not suggested to them
WHAT WE LEARNED (2)

- Students are learning the material (thanks to simulation?)
- Students who never ran a simulation did poorly on the exam (but perhaps they were just unengaged)
- Pre to post knowledge tests: ~20% success rate to ~80% rate
- Interesting correlation between grades and number of simulation runs:

**CORRELATION BETWEEN NUMBER OF SIMULATIONS EXECUTED AND AVERAGE GRADE ON PDC-FOCUSED FINAL EXAM QUESTIONS.**

<table>
<thead>
<tr>
<th># of simulations</th>
<th># of students</th>
<th>grade average</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>10</td>
<td>67.6</td>
</tr>
<tr>
<td>1-10</td>
<td>14</td>
<td>88.8</td>
</tr>
<tr>
<td>11-20</td>
<td>13</td>
<td>99.8</td>
</tr>
<tr>
<td>21-30</td>
<td>6</td>
<td>81.0</td>
</tr>
<tr>
<td>31+</td>
<td>12</td>
<td>75.5</td>
</tr>
</tbody>
</table>
WHAT WE LEARNED (3)

- Students had a positive experience
- Students appeared engaged during in-class sessions
- Perceived difficult level:
  - 60% “just right”, 23% “too difficult but useful”, 10% “too hard to be useful”, 7% “too easy to be useful”
- Written-in comments in course evaluation were very positive
- Two students since then have joined the WRENCH project as undergraduate researchers
- One technical issue: Docker on Windows 10 Home
CONCLUSION

- The modules are publicly available
- Many more are being developed as part of an NSF Cybertraining award
- Please contact us if you want to use these modules, or have feedback, or want to contribute

http://wrench-project.org/wrench-pedagogic-modules/