The active classroom: students and instructors parallel programming... in parallel

N. Giacaman, S. Kalra and O. Sinnen

Parallel and Reconfigurable Computing lab (PARC)
Department of Electrical and Computer Engineering
The University of Auckland
New Zealand
www.parallelit.org
https://acp.foe.auckland.ac.nz

5th NSF/TCPP Workshop on Parallel and Distributed Computing Education (EduPar-15)
Hyderabad, India, 25th May 2015.
Over the Pacific Ocean... New Zealand

From down under...
New Zealand

N. Giacaman, S. Kalra and O. Sinnen (ECE)
ACP-PDC
PARC, UoA
Zooming in... Auckland
Zooming in... The University of Auckland

- NZ's largest university
- 40,000 students
- Department of Electrical and Computer Engineering
- Parallel and Reconfigurable Computing group
Zooming in... Parallel and Reconfigurable Computing lab
Overview

- Programming, what is so difficult about it?
- The problem with traditional labs
- Desired features for programming with students
- Current tools worth considering
- Active learning environments
  - Traditional approaches
  - Improvements over traditional
- Active Classroom Programmer (ACP)
  - Context and benefits
  - Features (and current developments)
- ACP supporting the PDC Curriculum Initiative
- Student perceptions on their learning
- Conclusions
Programming... what is so difficult about it?

- Students mostly struggle with the higher level aspects of programming (rather than details such as syntax)
  - Understanding how to design the program to solve a problem
  - Debugging their own programs
- Requires development of cognitive skills that cannot be taught in the traditional classroom
  - Developing the programming strategy
  - Active learning is needed to motivate students
  - Debugging practice promotes cognitive programming skills
- Even more difficult for parallel programming
- Practice, practice, practice
  - Programming laboratory sessions (labs) are the typical opportunity in which students develop these cognitive programming skills
The problem with traditional labs

- **Typical model:**
  - Practical lab sessions are once a week, normally a few days following the classroom lesson pertaining to that topic

- **Delay between the lesson and associated practical lab is too long**
  - Short-term working memory only holds information for 20 minutes
  - Students need to interact with the new material almost immediately, otherwise it will be lost quickly

- **Frequency of the labs tend to be (at best) once a week. More material is thrown at the students before they have an opportunity to reinforce the earlier material**

- **It is essential to have a sufficient number of teaching assistants that circulate the room answering student questions, otherwise progress will be severely stalled**
Desired features for programming with students

- The instructor should have the ability to create code spontaneously.
- The examples should be transferable immediately to the student’s programming development environment seamlessly.
- Allow an exercise to be “progressed” in stages, with snapshots of the intermediate steps.
- The student and instructor should have independent copies of the code, so no one is “blocked” from coding.
- The tool should be non-intrusive and easy to use (no need for additional tools).
Current tools worth considering

- Interactive coding websites
  - Flexible, practice in own time
  - No need to download code
  - No IDE support, not really helpful for parallel programming
  - No guidance in programming strategy

- Collaboration plugins
  - Great for peer (and pair) programming
  - Cannot work independently
  - No versioning/snapshots

- Software version control
  - Versioning, powerful for students to replay in own time
  - Extra steps would be distraction in classroom
Active learning environments
The traditional approach

- Traditional classroom lessons
  - “Sage on the stage”
  - Minimal active learning
  - Not all bad:
    - Should not forget the expertise and knowledge of the teacher

- Traditional laboratory sessions
  - Active learning with predefined coding exercises
  - Weekly sessions, students complete exercises (alone?) at a workstation
  - Teaching assistants circulate the room answering (repeated?) queries
  - Too late? Students likely forgotten the material covered in lectures
Active learning environments
Improvements over traditional

- Pair programming laboratory sessions
  - Pairing students together in the lab exercises
  - Reduces pressure on teaching assistants, students solve queries together
  - Incompatible personality pairings
  - Students favour one of driver or observer roles without alternating

- Studio-based lessons
  - Replacing the standard lectures with lab sessions
  - Radical redesigned course, requiring an increase in the number of lecturers and teaching assistants to help conduct lessons
Active Classroom Programming (ACP)

https://acp.foe.auckland.ac.nz

N. Giacaman, S. Kalra and O. Sinnen (ECE)

ACP-PDC

PARC, UoA
Active programming as new material is presented

Only 1 instructor/teacher, “business as usual”
  - Cover theory or new material as normal (slides, etc)
  - But now slip in some coding snippets throughout the lesson

Exercises may be predefined or developed on-the-spot

Everything within the same IDE (only Eclipse support at the moment)

Snapshots/versions of the exercise are recorded and immediately available for students to download within the IDE

Students receive guidance on the *programming strategy*

The atmosphere of the classroom becomes *collegial*

*Pair programming* promoted, students (somewhat) *paired with* teacher

From institute’s financial point of view, cost-effective teaching model
ACP features

- User accounts
- Upload project (teacher only)
- Download project
- Sync project
- Version diff
- Admin controls (teacher only)
- Features currently being developed:
  - Generation of usage graphs from logs
  - Web-based version of ACP
ACP exercises for Parallel and Distributed Computing (PDC) Curriculum Initiative

Welcome to the homepage of the ACP exercises dedicated for the PDC Curriculum Initiative.

The exercises are currently under construction, and all information to set up will be available here around May 2015. In the meantime, please contact us if you have any questions.
ACP supporting the PDC Curriculum Initiative

Task/thread spawning

Table of Contents

1. Learn the Basics
   1. Some work with sequential code execution
   2. Write multithreaded program by "extends thread"
   3. Write multithreaded program by "implements runnable"
2. Thread methods
   1. Common Pitfall: Calling run() instead of start()
   2. Using join() to wait for threads to finish
   3. What is Daemon thread in Java and Difference to Non daemon thread
3. Life Cycle of Thread
   1. New
   2. Running
   3. Waiting
   4. TimeWaiting
   5. Termintate
   6. Understand the exceptions
2. Thread methods

2.1 Common Pitfall: Calling run() instead of start()

You should not call run() instead of start() to start the thread. If you call run(), you just run the run() method, which runs this statement “System.out.println("Hello from a thread!");” in current main thread. And the new thread not start yet. If you call start() method, the new thread will start, and then the new thread will call run() inside itself thread.

<table>
<thead>
<tr>
<th>run()</th>
<th>run some work.</th>
<th>call run() = run the codes inside the method(). It still run in the same thread and run in sequential.</th>
</tr>
</thead>
<tbody>
<tr>
<td>start()</td>
<td>start the Thread.</td>
<td>call start() = start the new thread, then the new thread will call run(). It will run in parallel.</td>
</tr>
</tbody>
</table>

We do not modify the last code, and put some comments to indicate which thread it runs inside.

Figure 4: Sample code with start() and run() - (ACP --- Task_thread_spawning --- Version 4 and Version 5)

```java
Console.println("Thread " + Thread.currentThread().getName() + ": Program starts at \$s", timer.Start());
Create ... 
Thread.start() ...

Console.println("Thread " + Thread.currentThread().getName() + ": Program stop at \$s", timer.Stop());
Console.println("Thread " + Thread.currentThread().getName() + ": Total used \$2f seconds", duration);

// Inside the thread worker:
public void run() {
    Console.println("Thread " + Thread.currentThread().getName() + ": \$s starts", name);
    Some work ...
    Console.println("Thread " + Thread.currentThread().getName() + ": \$s finished, took \$2f seconds", name, durations);
}
```

The output with start():

Thread 1 : Program starts at 21:10:14
Thread 10 : SomeWork 1 starts
Thread 11 : SomeWork 2 starts
Thread 12 : SomeWork 3 starts
Thread 13 : SomeWork 4 starts
Thread 14 : SomeWork 5 starts
Thread 12 : SomeWork 3 finished, took 2.70 seconds
What was most helpful for your learning?

- “The live examples. The practical examples of using the Eclipse/Android IDE in class, as well as the kinds of explanations and discussions held in class. **Errors and mistakes made were helpful** in the sense that we can not only see how they were caused, but **the thought process** behind the coding that caused the error, the consequence, and the workaround/fix to correct the error”

- “I enjoyed the coding sessions in class. It is really helpful because a lot of **the content we learn is only helpful when we actually code it ourselves** but **seeing the lecturer code in class** gives us a better understanding of what you are talking about and how to actually implement it”
In what way, if any, do you feel the tool assisted your learning?

- “Helped deepen my understanding and grasping of concepts”
- “Better understanding of theory that was taught. Exercise examples are really good, especially step by step versioning is helpful”
- “Look through parts and fiddle/modify the code to see differences and understand what happened”
- “Being able to run through examples in my own time outside lectures so I can grasp/understand concepts better”
- “It helped me understand theory through a practical approach”
Conclusions

- In programming courses, both lectures and labs are equally as important to the learning process
  - Students need the lecturer to explain new material
  - Students also need a lab-like environment for immediate practice
- A collection of projects are being developed to target core topics in the PDC Curriculum Initiative
- Available for instructors (and students) of PDC courses
- Instructors may create their own projects in the ACP framework and use it for their own courses – just ask for your own account!

**Acknowledgement**  Special thanks to the NSF/TCPP CDER Center Early Adopter Awards for their continued support