Kinesthetic Learning Activities in an Upper-Division Computer Science Course

Paul Sivilotti
The Ohio State University

Terms

Kinesthetic Learning: Students learn by carrying out a physical activity rather than passively listening to a lecture or watching a demonstration.

KLA: Kinesthetic Learning Activity

Distributed Computing: Many programs running concurrently, communicating over a network.

Assertional Thinking: Understanding an algorithm in terms of invariants, rather than as a step-by-step sequence.

Abstract

We have developed a set of KLAs for teaching concepts in distributed computing at the graduate and senior undergrad level.

Topic Area: Active learning in the classroom

Objectives

1) Improve student understanding of distributed algorithms
2) Encourage assertional thinking about these algorithms.

Level: Senior undergrads and grads

Developmental History of Innovation

Many KLAs already exist for introductory computer science courses.

(1) People as processors: Students act out algorithms (eg sorting, merging)

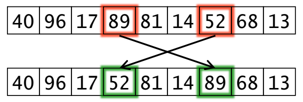
(2) People as memory: Students form data structures (eg trees, stacks)

But these KLAs often encourage an operational view (i.e., algorithms as sequences of actions).

In 2002, we developed a KLA for middle school girls attending a computer science workshop. Subsequently, we designed many new KLAs, focusing on assertional reasoning for students in an upper-division course in distributed computing.

Example: Nondeterministic Sorting

Algorithm: Randomly select a pair to compare and swap if out of order. Repeat.



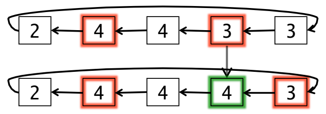
Each student is an element in the array. Each has a sign (their position in array) and an index card (their value). They mill around, randomly comparing with others and swapping index cards.

Learning Objectives:

1) Sequential sort is an over-specification. Deterministic order is not required.
2) Termination detection is difficult. Global quiescence is not locally detectable.
3) Proving termination requires a metric. Convincing argument needed for progress.

Example: Self-stabilizing Mutual Exclusion

Algorithm: Continuously compare value with left neighbor. If they differ, use the resource. To release the resource, change one's value to equal left neighbor.



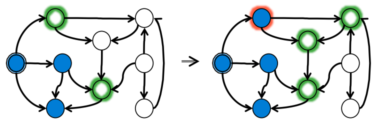
Each student has a paper indicating their value. Students are arranged in a circle so each can see their left neighbor's value. To indicate they are using the resource, they play a musical note. Initially there is a cacophony of sound, but as the algorithm stabilizes, a melody emerges.

Learning Objectives:

1) Self-stabilization = convergence + stability. Once the tune emerges, it continues.
2) Process 0 plays an important role in algorithm. Noise is eliminated by this process.

Example: Parallel Garbage Collection

Algorithm: Mark root as reachable. Mark any node that is reachable from a marked node. Repeat.



Learning Ojbectives:

1) Operational reasoning is dangerous
   The activity is most effective in illustrating an *incorrect* algorithm! The incorrect algorithm appears to give the right answer, but a subtle error exists.

Myths vs Reality

Myth: KLAs take too much class time

Reality: KLAs can be short and effective. They have an excellent return on investment!

Myth: KLAs are too disruptive

Reality: KLAs are controlled and directed student engagement. Energizing the class is a positive outcome!

Myth: Engineers are not kinesthetic learners

Reality: Research shows all learning styles represented in significant proportion.

Myth: KLAs work only in small classes

Reality: Some KLAs actually benefit from greater numbers of participants!

Major Issues to Resolve

1) Development of new KLAs
   How can these patterns be applied to other courses in computer science, including service-level courses on programming? Is this even a good idea?
2) Assessment of effectiveness
   How can we assess the effectiveness of these techniques in student learning? How can the results of such assessments be used to improve the activities?

Tips for Success

1) Design KLA to achieve a *specific* learning objective. Avoid KLAs that only illustrate.
2) Plan KLA carefully ahead of time.
3) Use KLAs throughout the semester. Introduce them early and include often.
4) Be sensitive to physical disabilities (e.g., restricted mobility, blindness, deafness) and differences in cultural norms.

Discussion

KLAs can be effective tools in teaching, so long as they are carefully designed to target specific learning objectives.

Designing good KLAs is a significant effort. By sharing successful KLAs, we can amortize this effort over many people.

These KLAs leverage the specific nature of distributed computing. Can KLAs be equally effective in other disciplines too?

Acknowledgments

These KLAs were developed in cooperation with Scott Pike, Murat Demirbas, Nigamanth Sridhar, Hilary Pike, Greg Buehrer, and Matt Lang.

This work was supported in part by the National Science Foundation under Grant No. 0931669.

NSF

2010 Frontiers of Engineering Education Symposium

Irvine, California
December 13 - 16

Sponsored by:

The National Academy of Engineering and
The O' Donnell Foundation

THE OHIO STATE UNIVERSITY DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NATIONAL ACADEMY OF ENGINEERING OF THE NATIONAL ACADEMIES