The Suitability of KLAs for Teaching Distributed Algorithms

Paul Sivilotti
Ohio State University
paulo@cse.ohio-state.edu

Scott Pike
Texas A&M
pike@cs.tamu.edu

What is Kinesthetic Learning?
1) Engages students through a physical activity
   - Students are walking, waving, pointing, shouting, clapping, dancing, hopping ...
   - Watching the instructor is not kinesthetic
2) The activity supports a specific learning objective
   - Designed to make a point
   - Not just fun for the sake of a break

KLA = "kinesthetic learning activity"

Typical KLAs in CS
1) People as processors
   - Students acting out an algorithm
   - eg flow chart hopscotch, network routing
2) People as data structures
   - Students as memory cells/data, with pointers
   - eg human binary tree, sort the students
   - Generally used in intro courses
   - Gentle introduction for the non-expert

Why is it a Good Thing?
- Active engagement
- Benefits extend to time spent in "traditional" lecturing
- Multimodal
  - Audio, visual, physical
- Visceral
  - Memorable examplar internalized as a powerful hook for an important concept

Problems...

Common Concerns re: KLAs
- They take too much class time
  - Covering course material vs teaching course content
- I will lose control of my class
  - Takes some confidence, but not so scary
- CS students are more technical than kinesthetic: they won't participate
  - They will
- They take too much time to design
  - Ok, I'll give you that one
The REAL Problems...

Problem #1: Concurrency
- Concurrency and nondeterminism
  - Every student has a brain
  - They insist on using them
- Result: rampant parallelism
  - eg every node in the tree is computing
  - Often this parallelism is not the point
  - Activity must be carefully designed to accommodate the parallelism, or maybe even leverage it

Problem #2: Broken Metaphor
- Encourages the *wrong* mental model
  - Rampant anthropomorphisms
    - "This guy sends a message to that gal..."
  - Operational view
    - "First this flag is set, then that pointer..."
- Famous EWD warnings on the dangers of metaphor

Our Work
- Present a collection of KLAs
  - Senior/graduate level
  - Class in distributed algorithms
  - Enrollment: ~30
- Describe designs in detail
  - Lessons learned (Ohio State and Texas A&M)
  - Tips for success
- Sensitive to both "real problems"

KLAs for Teaching Dist Alg's
- Problem #1: Concurrency
  - Not a problem, it's the whole point!
  - Fundamental to course material
    - Multiple threads and interleavings
    - Distinction between local and global view
    - Appreciate limitation of having only local knowledge

KLAs for Teaching Dist Alg's
- Problem #2: Broken metaphor
  - This is a real danger, especially for dist alg's
  - Must go beyond "act out the algorithm"
- Design to directly support *assertional/view*
  - Help build intuition about key algorithmic invariants, metrics, properties
- Design to expose inadequacy of *operational/view*
  - Use KLA to present an algorithm that is simple, elegant, compelling, and wrong
Demo: Sorting
- Classic "sort the students"
- But with a twist...

Initialization
- Place green "rank" sign around your neck
  - Number should be visible to everyone
  - Do not remove sign until the end
- Write the last 4 digits of your phone number on the small white "data card"
  - Should not be visible to anyone
  - This data card will change hands
- Note:
  - Rank is not physical position (you will need to move around!)

The Algorithm
Repeat:
  Choose a random partner
  Compare data values
  If they are out of order (wrt rank), then switch data cards
  ie smaller rank gets smaller data
Until you all are "sorted"

Note: comparisons are pair-wise

Questions to Ponder
- If you had time to plan out a strategy, what would you do?
- Is this stable: Pos(n) = Rank(n)?
- Variant:
  - Use other participants as "comparison processors"
  - Only see other data value after swap
- How do participants know they are done?

Reasoning about Safety
- Strawmen for invariants:
  - True
  - The sky is blue
  - My number (5835) is somewhere in the list
  - ...
  - The list is a permutation of the original

Reasoning About Progress
- How do we know this will terminate?
- Strawmen for metrics
  - (For all n: |Pos(n) - Rank(n)|) is decreasing
  - (Sum n: |Pos(n) - Rank(n)|) is decreasing
- Key observation
  - Frequency of swapping decreases
  - Earlier comparisons more likely to result in a swap
  - Earlier, there are more out-of-order pairs
Metric: # of Out-of-Order Pairs

- Metric is bounded below (0)
- Does it decrease with every swap?

\[
\begin{array}{cccccc}
  a & a & x & b & b & y & c & c \\
  a & a & y & b & b & x & c & c \\
\end{array}
\]

- For every pair in order before, there exists a pair in order after the swap
- a,x doesn’t change (x,c same for a,y & y,c)
- For b: x,b in order before \(\implies y,b \) in order after

KLAs We Have Designed

- SIGCSE 2007
  - Nondeterministic sorting
  - Metric and termination
  - Parallel garbage collection
  - Operational view: simple, natural, and wrong
  - Stabilizing leader election
  - Nondeterminism and convergence
- Earlier work (SIGCSE 2003, SIGACT News, Distributed Computing Column)
  - Coffee can problem
  - Self-stabilizing token ring

Take-home Messages

- The real challenges in KLAs for CS
  - Concurrency and promoting an operational view
- KLAs can be used in advanced courses
  - Good response from grad (\& 8th graders too!)
- KLAs may actually be particularly well suited for some advanced topics
  - Embrace the concurrency
- KLAs can be designed to support assertional reasoning
  - Must be done deliberately

The Suitability of KLAs for Teaching Distributed Algorithms

Paul Sivilotti  
Ohio State University  
poolo@cse.ohio-state.edu

Scott Pike  
Texas A&M  
pike@cs.tamu.edu