CSSE 220

Performance with Threads

Checkout *SumArrayInParallel* project from SVN

Uses for Threads

- Multiple code paths running at once
  - Animation
  - Responding to user input at the same time we’re doing a calculation
  - etc.
- What about performance?
  - Could we not get better performance by creating enough threads to divide work among them on different processor cores?

Conceptually

- The concept is pretty straightforward:
  - Existing Problem: A large task that runs on one core, doing one thing at a time
  - Running a program in one core on our machines would be roughly as “fast” as running the same program on a processor from 12 years ago! (2004 was the last time Rose had single-core machines)
  - Modern processors have multiple cores
    - HOW DO WE TAKE ADVANTAGE OF MULTIPLE CORES??

Modern Operating Systems

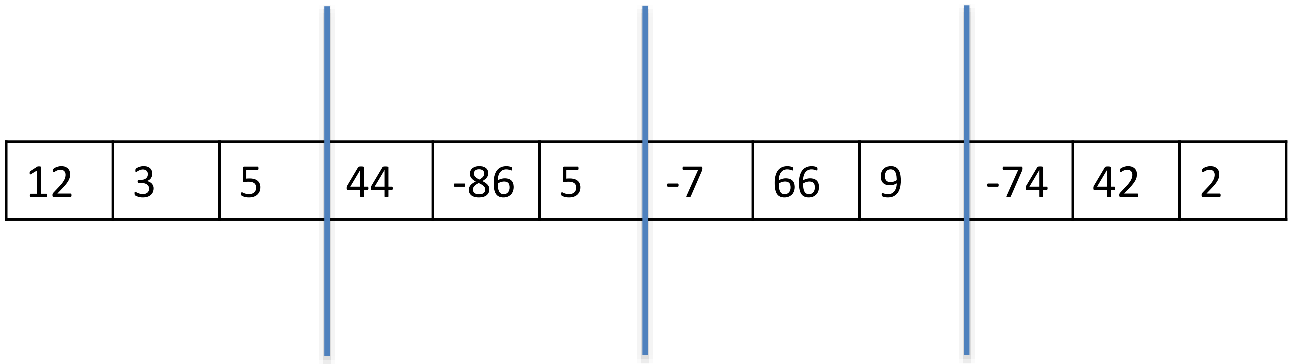
- Woo Hoo!
- Modern operating systems automatically (more-or-less) send waiting threads to a processor core that is waiting for work
- If we write the program to allow the operating system to assign threads to separate cores, then our task (in this class) is just splitting up the work into different threads!

Our Task Today

- We want to sum a huge array of integers
- Serially, we just add each array element to the current sum and then return the sum when finished
- With threads, we can split up the work very easily because of the associative law of addition

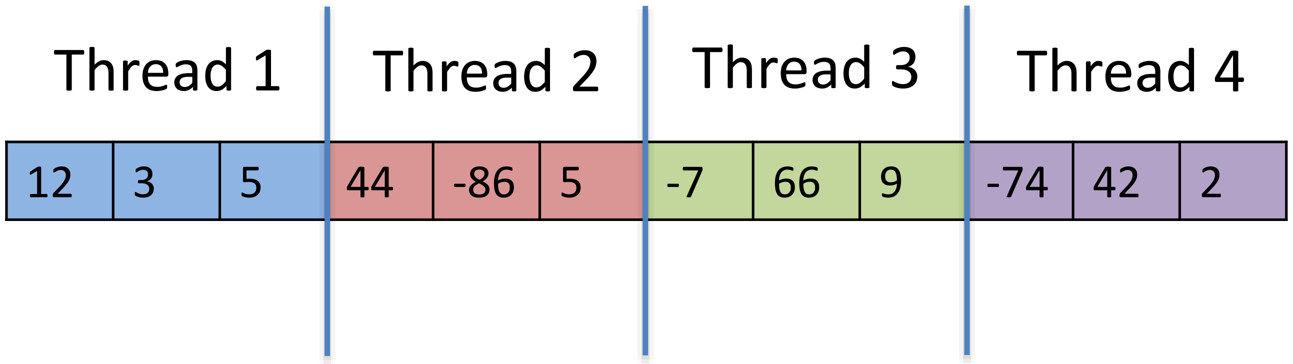
The idea

- When a very large task can be split into pieces
  - Assign a thread to one piece and let that thread return its result



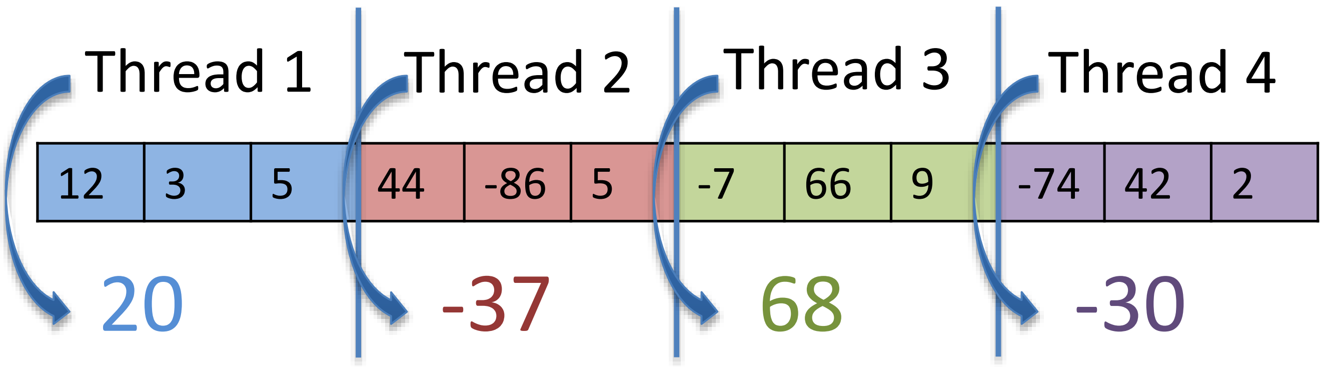
The idea

- When a very large task can be split into pieces
  – Assign a thread to one piece and let that thread return its result



The idea

- When a very large task can be split into pieces
  - Assign a thread to one piece and let that thread return its result



Add individual portions and return result: **21**

The Difference

- Conceptually, one core adding 12 numbers serially will “take longer” than 4 cores adding 3 numbers in parallel, then adding those 4 together.
- IN REALITY, we need to sum a very large array to see the performance gains in Java since the threads are so heavyweight
  - We’ll use about 200,000,000 integers in an array!

Student Evaluations

- The school treats these very seriously
- I treat them very seriously too...usually reading them multiple times
- Try to be as honest and detailed as you can
- If you liked the course, don’t say nice things about me (though I like that of course) tell me what topics you thought were most exciting or useful

Presentations

- 8 minutes long
- ~3 minutes showing off your extra features
- ~5 minutes explaining JUST ONE technical decision in detail
  - Could be a design decision, feature implementation, or tricky bug
  - Be careful about code examples (good – but in moderation)
  - Should include prepared slides usually including diagrams

Work time

PRESENTATION IS FRIDAY!!!

TEAM PROJECT