Concepts and Models of Parallel and Data-centric Programming

Parallel Algorithms IV

Lecture, Summer 2020

Dr. Christian Terboven <terboven@itc.rwth-aachen.de>

i12 High Performance Computing

RWTH AACHEN UNIVERSITY

Outline

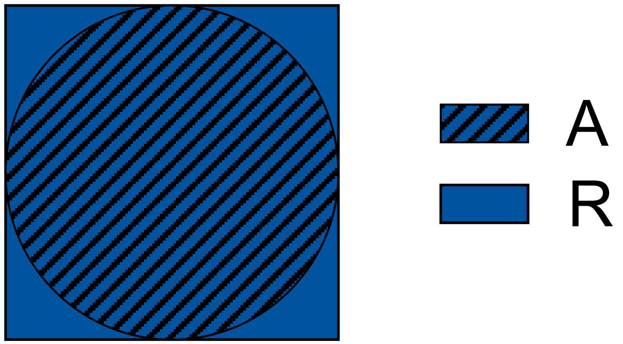
0. Organization
1. Foundations
2. Shared Memory
3. GPU Programming
4. Bulk-Synchronous Parallelism
5. Message Passing
6. Distributed Shared Memory
7. **Parallel Algorithms**
8. Parallel I/O
9. MapReduce
10. Apache Spark

a. Berkeley DWARFS
b. Dense Linear Algebra
c. Sparse Linear Algebra
d. Monte Carlo Methods
e. Graph Traversal

Monte Carlo

PI / 1

- $\frac{\pi}{4}$ = ratio of
  - area of the circle
  - area of the square



- Monte Carlo method
  - Statistical simulation
  - The ratio of the number of points that fall inside **A** to the total number of points tried (all within **R**) is equal to the ratio of the two areas (or volume in 3d)
    - Randomly throw darts inside a 2x2 square area
    - Count darts that hit the radius 1 circle
    - PI = 4* hits/total

PI / 2

- What is the intuitive approach to parallelize this problem?

- Nearly embarrassingly parallel solver
  - Parallel generation of random numbers
  - Parallel counting of hits
  - Linear collection of hit count at rank 0 (by design)
  - Final calculation and printout at rank 0

- Scaling expectation
  - Scales linear until sequential part starts to dominate (Amdahl's law)

PI / 3

- Perfect network, i.e., 0 latency and ∞ bandwidth
  - Eventually, the applications runs out of work
  - Increasing the number of iter-ations pushes the scalability limit