

Early Adopter – PDC At Washington and Lee

Four-year Liberal Arts

- >CS2/DS (Python): 7 students
- >Graphics (C++): 7 students

Dr. Joshua Stough (stoughj@wlu.edu)

Washington and Lee University, Virginia



Our department

- Four Faculty:
 - Lambert, Levy, Sprenkle, Stough
 - Areas: programming languages, computational linguistics, robotics, AI, web application testing, image analysis and graphics
- 5-10 majors, 1-5 minors per year
- OS by prevalence: Linux, OS X, XP

Proposal Goals

- CS2/DS

Topic	Bloom	Hours	Outcome
Divide & conquer	A	3	Mergesort, quicksort, sum of primes, Monte Carlo simulation (compute pi), numerical integration
Performance Metrics	C	1	Basic definitions (speedup, efficiency, Amdahl), scalability with processes versus cores, load balance
Tasks and Threads	C	1	Process/Pipe versus Pool/map (Python multiprocessing), task/data parallelism

- Graphics

Topic	Bloom	Hours	Outcome
Divide & conquer	A	1	Raytracing/radiosity applications
Misc.	C, K	1	Parallelism in the pipeline stages, GPGPU, SIMD, student presentations

CS2/DS

- Course goal: efficient interaction with and manipulation of collections of stuff.
- PDC applications: sorting, recursive tree structures, image processing,...
- Used Python multiprocessing module:
 - Process/Pipe: spawn Processes for smaller problems, communicate through Pipe.
 - Pool/map: pool of worker processes if limited communication required
 - Synchronization primitives

CS2/DS Examples

- All code includes timing versus sequential
- Using Pool/map:
 - Shown: parallel integration
 - Developed in class: sum of primes $\leq n$
 - Assigned: Monte Carlo Simulation to compute pi
- Using Process/Pipe:
 - Shown: parallel quicksort
 - Assigned: parallel mergesort



CS2/DS Evaluation

- Assignment, Test questions, Student Evaluation Question
- Positive:
 - Little syntactic complexity
 - Students quickly see impressive speedup
- More to do: better integration
 - Recursive tree functions
 - Client/Server app
 - Parallel image processing
 - Synchronization

Graphics

- Only got to 'Know the term'
 - Non-Graphics topics already covered:
 - Linear Algebra
 - C++/OpenGL
 - Places to save:
 - Ray tracing
 - Low-level (clipping, rasterizing, texture-mapping) could be replaced in part with parallelism at a higher level (e.g., game design).
- Suggestions welcome!



PDC at W&L

- Course offerings:
 - Parallel & Distributed Systems (Sprenkle)
 - Parallel Computing (Lambert)
- Curriculum changes:
 - Modeling: non-major, 20 students (Levy)
 - Survey: non-major, 20 (Stough)
 - Scientific Computing: 20 (Levy)
 - CS1: 40 students (Stough)
 - CS2/DS: 12 students (Lambert)
 - Upcoming: Graphics, Image Processing, Networks, Architecture

Upcoming

- CS1 in Python (F '11)
 - Client/Server programming (Bloom A)
 - Data-parallel, Integration, Monte Carlo (C)
 - O-notation, Amdahl's Law, speedup (A)
- Survey of Computer Science (W '12):
 - Multicore, SIMD (using Intel Ticker Tape) (Bloom K)
 - Cloud/grid/distributed computing (K)

Appendix: Pool/map

- Monte Carlo pi computation:
 - Generate N points in the unit square
 - Fraction of points that are also within the unit disk approaches $\pi/4$ as N grows.
- Parallel Solution:
 - Instantiate pool of P processes.
 - `pool = Pool(Processes = P)`
 - Have each Process estimate pi on N/P samples
 - `estimates = pool.map(f, [N/P, N/P, ...])`
 - Return the average of the estimates.

Appendix: Process/Pipe

- Mergesort:
 - Mergesort the left half-list
 - Mergesort the right half-list
 - Merge the two sorted half-lists
- Parallel mergesort:
 - Mergesort the left and right half-lists
concurrently: spawn new Processes
 - Merge the results of the subprocesses
 - Use Pipes for return values.