AN APPROACH TO PARALLEL GEOPROCESSING WITH PYTHON

Chris Jochem

University of Colorado Boulder
warren.jochem@colorado.edu
Spatial Data Processing Challenges

• Growing size of available datasets
  • LiDAR point clouds
  • High spatial/temporal resolution remote sensing products

• Computer processing power is growing… but desktop GIS software is not able to fully utilize new hardware (yet).

• Motivation for this project:
  • Site-suitability model for power generation
  • 100m resolution for the lower 48 states
  • Over 2 dozen input layers (vector and raster) needing processing
Parallel Processing?

• The Checkout Line Analogy:
  • When you have a lot of customers, open more registers rather than speeding up cashiers.
  • When you have a large computational task, divide the work over multiple processing cores rather than trying to build a single, faster processor.
Research Objectives

1. Improve performance for geoprocessing tasks and make use of very large datasets
   - Examples: Map Algebra, Reclassification, Conversion, Moving Windows, Buffering, Spatial Neighbors, etc.
   - “Embarrassingly Parallel” problems for raster AND vector data

2. Bring together already available resources
   - Parallel processing common in other disciplines
   - Apply parallel processing to spatial datasets with open-source libraries

3. Implement generic functions for multiple platforms in a familiar language for “non-expert” programmers
Spatial Data and Parallel Processing

- **GRASS GIS**
  - [http://grass.osgeo.org/wiki/Parallel_GRASS_jobs](http://grass.osgeo.org/wiki/Parallel_GRASS_jobs)

- **Manifold GIS on GPGPU**
  - [http://www.manifold.net/info/pr_gpu_record.shtml](http://www.manifold.net/info/pr_gpu_record.shtml)

- **Parallel Raster Processing Library (pRPL)**
  - [http://www.geog.ucsb.edu/~guan/pRPL/index.htm](http://www.geog.ucsb.edu/~guan/pRPL/index.htm)

- **Other examples of specialized applications for image processing, parallel agent simulations, and genetic algorithm implementations**
Why (and How) to Use Python?

- Fast, efficient, powerful language for scientific computing
- Available spatial data libraries
- Recent options expand parallel capabilities
- Familiar to many GIS analysts
Parallel Python

- http://www.parallelpython.com/
  - created by Vitalii Vanovschi

- Open-source module for multi-core/multi-processors and networked cluster computers

- What about the GIL? (Global Interpreter Lock)
  - Only 1 thread can operate Python bytecode at a time
  - PP uses subprocess to perform multi-processing, avoiding lock

- Other options include MPI-based (e.g. PyMPI), IPython, spawning, and several others
  - ParallelPython found to be user-friendly and stable
Example: Moving Window Filter

- Objective: Calculate “focal” or “neighborhood” sum
- Using 33 x 33 pixel window on a 100m input raster
  - motivating work was to calculate a 1 to 20 mile radius densities
- Parallelize by dividing input, calculating in parallel, merging output…

![Input Raster](image)

![Output Raster](image)
Example: Parallel Moving Window Filter

1. Input Raster with Sub-Raster Divisions

2. Sub-Rasters sent to “Job Server” queue for available processing node

3. Processing Step – return sub-raster

4. Processed Sub-Rasters merged to create output raster

overlapping edge pixels included for adjacent areas
Moving Window Algorithm

- Sum all pixels in local moving window
  - Need efficiency → recursion
- Solution: only add all pixels in the first position

\[
\text{Output}_{ij} = \text{Sum}_{i-1,j} - \text{Sum}_{i-\text{offset}} + \text{Sum}_{i+\text{offset}}
\]
Output Results

Parallel Python raster processing, 32 pixel local sum

Test Systems:
Server with four, quad-core (2.7 GHz) processors, 32 Gb RAM, Python 2.6 ArcGIS 9.3.1 on dual-core (2.2 GHz), 4 Gb RAM

ArcGIS processing time = 72.12 sec.

Processing Time (seconds)

# of Processors
Results and Conclusions

- Successfully used Python and available modules

- Significant processing time improvement over ArcGIS using parallel processing methods

- Parallel processing allows for larger datasets to be processed → avoiding memory limits of desktop GIS

- Code provides opportunities for reusability/expansion to multiple processing steps and other applications
Future Directions and Next Steps

• Need further testing:
  • For limits and benchmarks to establish scalability/speed-up
  • Optimal way to balance task loads/sub-divide data?
  • Test other hardware architectures: clusters, etc.

• Exploration and comparison with ArcGIS 10 and ArcPy
  • Also compare with other GIS software packages

• Expansion to other geoprocessing algorithms and creation of programming interfaces
  • Open-Source project ( Longer term goal )

• Spread the word to Geographers!
AN APPROACH TO PARALLEL GEOPROCESSING WITH PYTHON

Chris Jochem

University of Colorado Boulder
warren.jochem@colorado.edu