Parallel Python - Tutorial

Paul Springer

Aachen Institute for Advanced Study in
Computational Engineering Science

Aachen, 17.01.13

Global Interpreter Lock (GIL)

- Python interpreter not entirely thread-safe
- Thread must acquire GIL to access python objects
- Only one thread can access hold the GIL at a time
  - Limits thread-level parallelism

- NumPy/ SciPy
  - Precompiled algorithms
  - Efficient multi-dimensional arrays
- Numexpr
  - JIT compiles expressions like: x**3 + 2. * y + 1.
- Shedskin
  - Python to C++ converter
  - Little effort
- PyPy
  - JIT compiler
  - Little effort

Live Demo

- Threading
  - Multi-threading for python
  - Limited by GIL
- Multiprocessing
  - Spawns multiple processes (overcomes GIL)
  - Limited to one machine
- Parallel Python
  - Parallelism is expressed in jobs
  - Parallel across one or more SMP nodes
- Cython
  - Annotate python code with data types
  - Generate C code
  - Parallel support (via prange)

- MPI4Py
  - Interface to MPI 1/2
- NumbaPro
  - Translates python code to LLVM
  - Multicore and GPU support
- pyCUDA
  - CUDA wrapper for python
- pyOpenCL
  - OpenCL wrapper for python

Case Study

#Threads	Numexpr	Cython	Shedskin	NumbaPro	C
1	4.2	5.8	7.5	7.6	8.6
2	8.3	9.3	-	12.5	-

Table: Speedups summary of the Live Demo using the improved NumPy version as reference.

	Original	Improved	Cython	PyPy	Shedskin@1	Shedskin@8
Part 1	1	1.4	31.4	40.9	37.2	272.4
Part 2	1	8.9	-	68.9	289.0	389.6

Table: Speedups of a Molecular Dynamics (MD) simulation.

Summary

- Improve serial code first
- Parallel python is possible

Single Node

- NumPy/ SciPy
- Numexpr
- Cython (prange)
- Shedskin & OpenMP
- NumbaPro
- Parallel Python
- pyCUDA
- pyOpenCL

Multiple Nodes

- MPI4Py
- Parallel Python

- High-Performance Python Tutorial
  - http://ianozsvald.com/2012/03/18/high-performance-python-1-from-pycon-2012-slides-video-src/

Thank you for your attention.

Additional Resources

- Discussion on GIL
  - http://www.artima.com/weblogs/viewpost.jsp?thread=214235
- Threading and MultiProcessing
  - http://www.scipy.org/ParallelProgramming
- NumPy
  - http://www.NumPy.org/
- SciPy
  - http://www.scipy.org
- Numexpr
  - http://code.google.com/p/numexpr/
- Cython
  - http://www.cython.org
- PyPy
  - http://www.pypy.org

Additional Resources

- Shedskin
  - http://code.google.com/p/shedskin/
- Numba
  - https://github.com/Numba/Numba
- NumbaPro
  - https://store.continuum.io/cshop/NumbaPro
- Parallel Python
  - http://www.parallelpython.com/
- pyCuda
  - http://mathema.tician.de/software/pycuda
- pyOpenCL
  - http://mathema.tician.de/software/pyopencl
- MPI4Py
  - http://code.google.com/p/mpi4py/

- Threading
  - Standard python module
  - Limited by GIL
  - Can still be useful if GIL is released (e.g. NumPy)
- Multiprocessing
  - Standard python module
  - API similar to threading but uses processes instead of threads
  - Not limited by GIL
  - Limited to one node
  - Uses pickle

- NumPy
  - Convenient array notations (like Matlab)
  - Efficient data containers (multi dim arrays)
  - Releases the GIL
  - Parallel, if liked against multi-threaded BLAS
  - Precompiled algorithms (very fast)
    - Linear algebra
    - Fourier transform
    - Random number capabilities
- SciPy
  - Based on NumPy
  - Precompiled algorithms (very fast)
    - Numerical integration
    - Interpolation
    - Optimization
    - Sparse eigensolver problems
    - ...

- Provides MPI-1 and MPI-2 functionality
- Very similar to C++ interface
- Possible to send any picklable python object
  - Very convenient
  - Slow
- Optimized data transfers for NumPy arrays
  - Fast

- Evaluates expressions (e.g. x**3 + 2. * x * y + 1.)
- Compiles the expression JIT
- Uses blocked algorithms
- Releases the GIL
- Supports multi-threading

- Based on processes
- Automatic work balancing
- Uses inter process communication (IPC)
- Complexity of IPC completely hidden from the user
  - User submits jobs and receives results
- Supports SMP nodes and clusters
- Warning: Tasks longer than 30sec will time out
  - Change TRANSPORT_SOCKET_TIMEOUT in pptransport.py on every host

- Annotate python code with types
- Generates C code
- Offers multi-threading support (i.e. prange)

1. cython -a <file>.pyx
2. Look at generated <file>.html file
   - As few yellow lines as possible!
3. create setup.py file
4. python setup.py build_ext –inplace
   - Compiles code/ generates .so file

Shedskin

- Python to C++ converter
- Generates need .cpp files
- Possible to use OpenMP within generated .cpp files
- Restrictions
  - no NumPy support!
  - Python code may not use dynamic typing

1. shedskin <file>.py
   - Generate C files
2. make
   - Generates executable

- Numba
  - OpenSource
  - Dynamically compile python code
  - Based on LLVM
  - NumPy and SciPy support
- NumbaPro
  - Commercial, but offers academic license
  - Dynamically compile python code
  - Based on LLVM
  - NumPy and SciPy support
  - Targets: Multicore CPUs and NVIDIA GPUs

- JIT Compiler
- Can yield huge speedups without any programming effort
  - Always worth a try
- Limited NumPy support
  - import NumPypy instead of NumPy
  - Continuously improved
  - Status: http://buildbot.pypy.org/numpy-status/latest.html