EuroSciPy 2018

Trento, Italy

# *The Hitchhiker’s Guide to Parallelism with Python*

Workshop/Tutorial

Wednesday 29th August

Declan Valters

University of Edinburgh, Scotland, UK

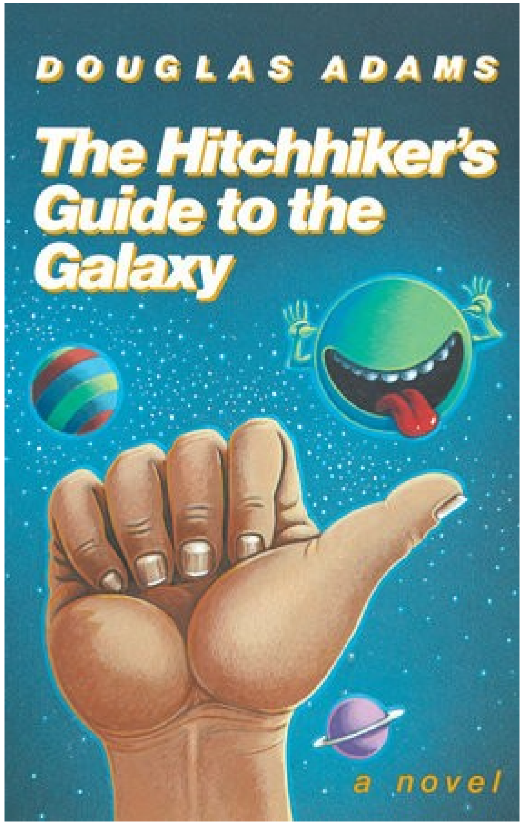
Research Software Engineer, School of GeoSciences

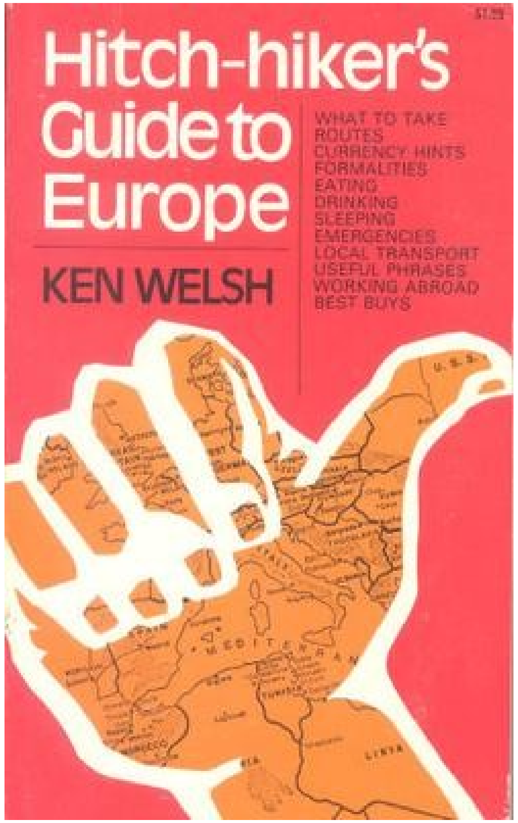
@dvalts

## Workshop aims

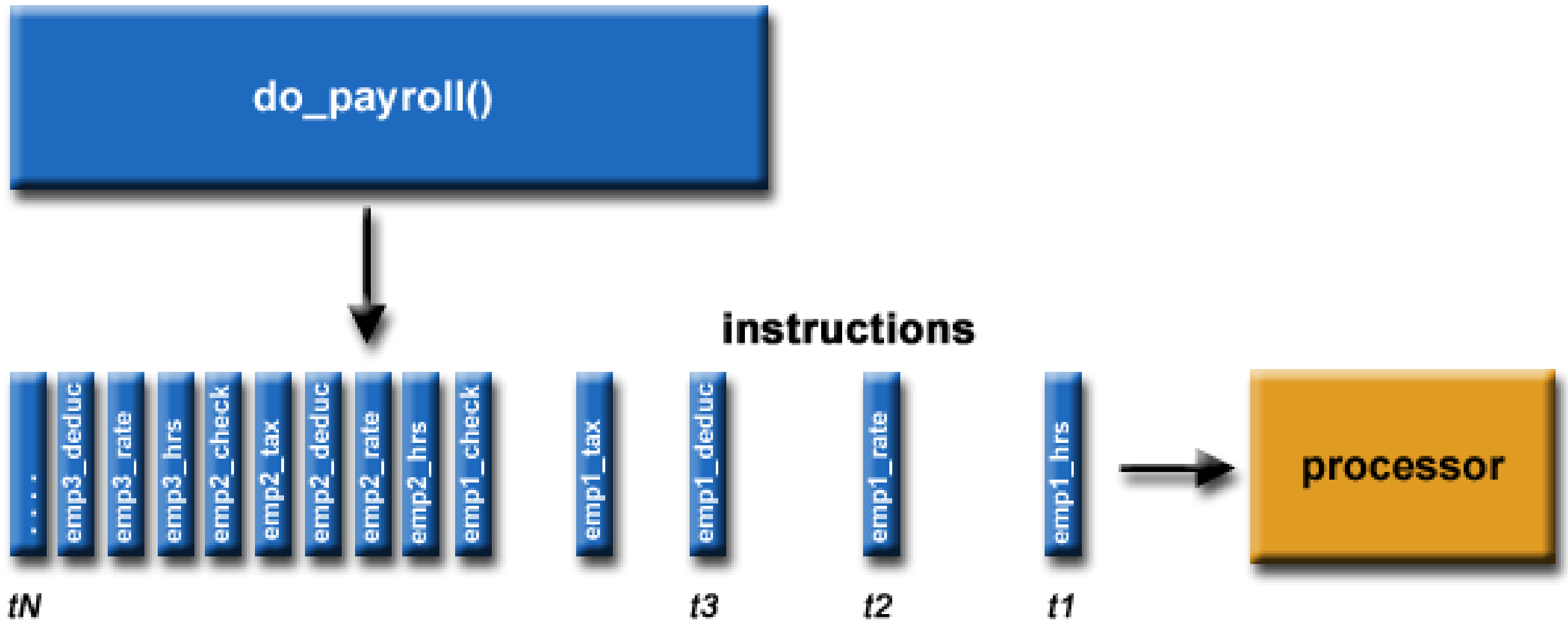
- Gentle introduction to parallel programming
- Python parallel programming through 4 mini-tutorials
  - Each covering a different library for parallelism
- Not a ‘super-advanced’ parallelism tutorial
- Based on own learning and topics interested
- Doesn’t cover every parallel library in Python
- **Opportunity to share your own experiences**

# The Hitchhiker's Guide to Parallelism with Python

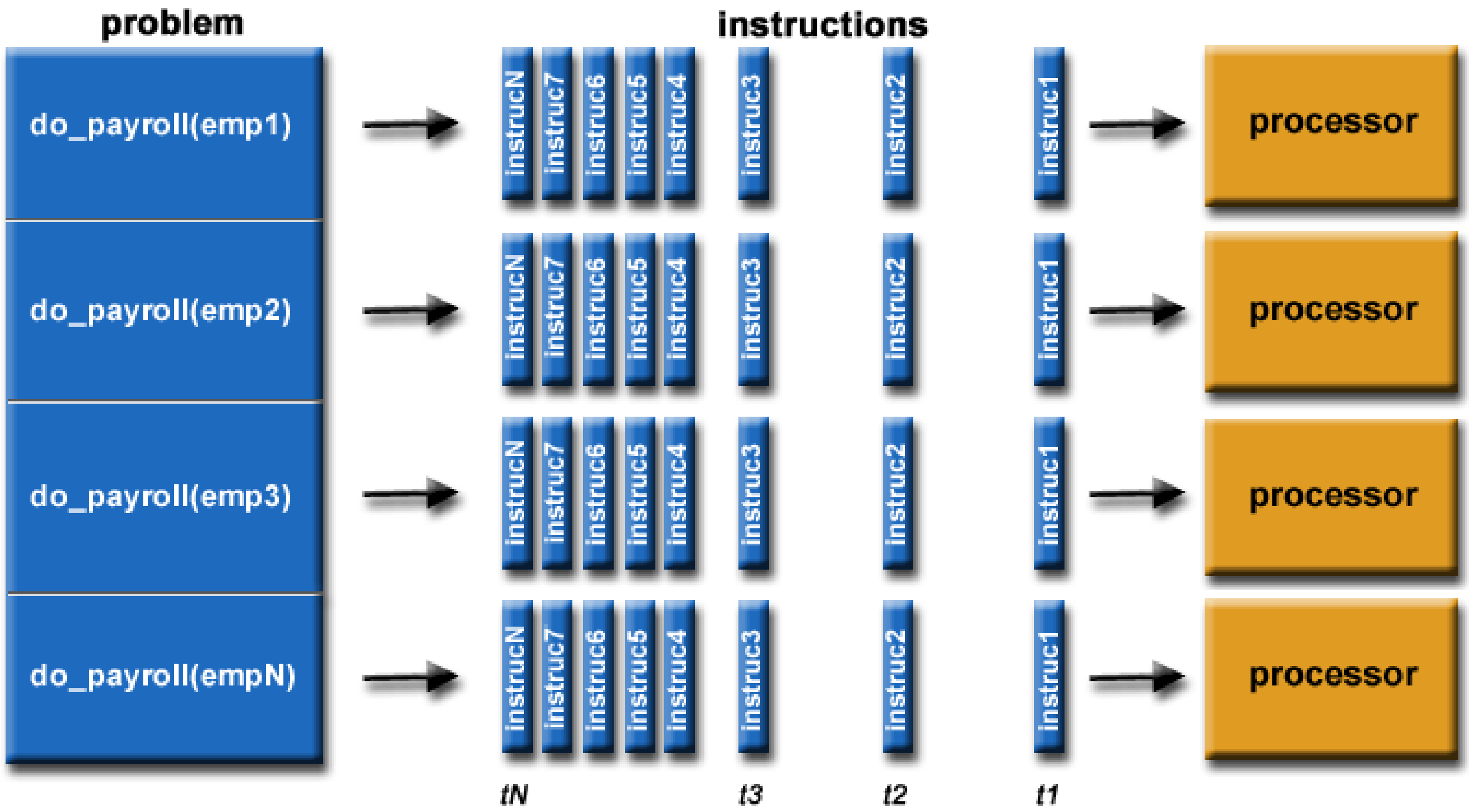




## Serial programming



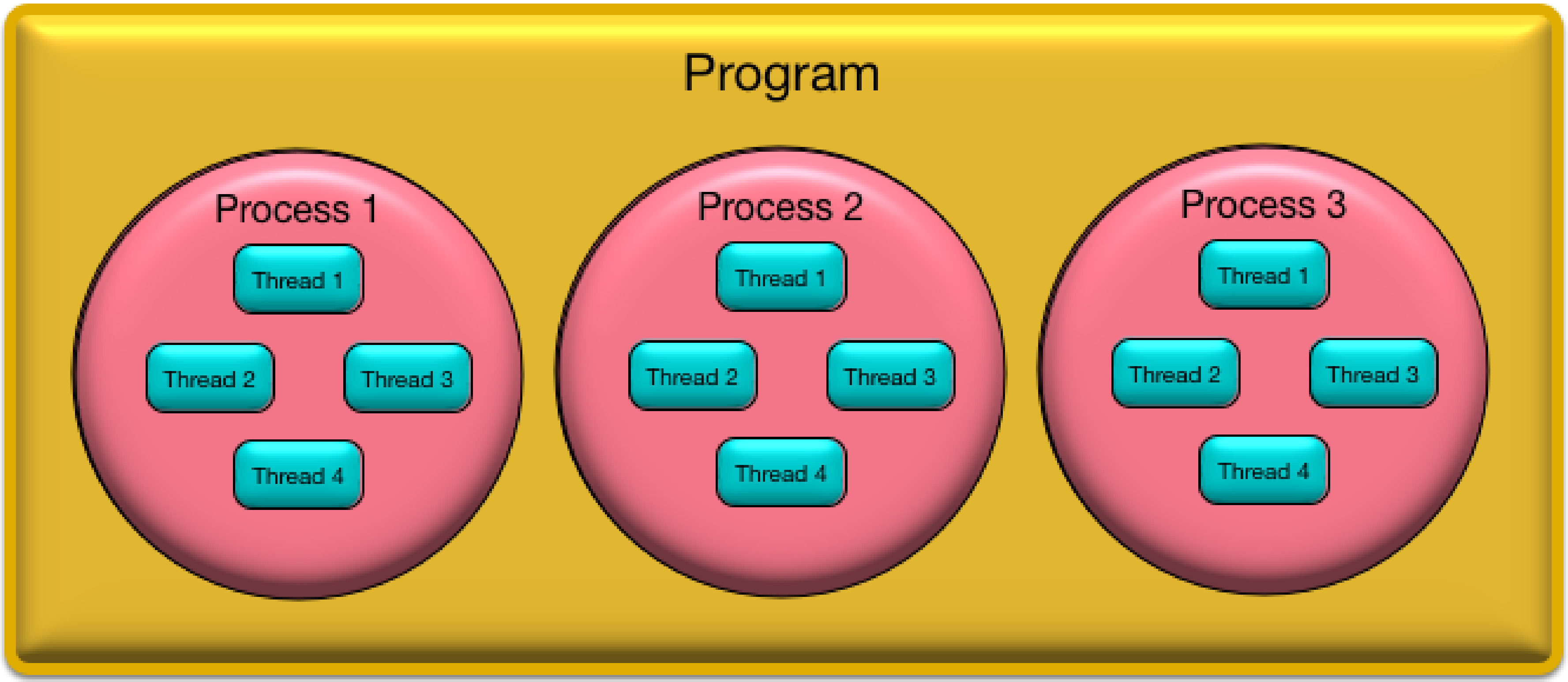
# Parallel programming



## Parallelism models

- CPU **multi-processing** / Distributed-memory parallelism
  - Create multiple OS/system processes
  - Execute them in parallel
  - Python MPI (**mpi4py**), **multiprocessing** module
- CPU **multi-threading** (shared memory parallelism)
  - Threads share the same portion of memory assigned to their parent process
  - OpenMP (C/Fortran/C++) (**Cython, Numba**)

# The Hitchhiker’s Guide to Parallel Programming



## Parallelism and Python

- CPython implementation – the *de facto* standard Python
- The Global Interpreter Lock (GIL)
  - ~~Native Python multithreading?~~
  - But not the only type of parallelism
    - Process / task based parallelism
    - Cheat and use C + OpenMP
    - MPI (message passing interface)

## Parallelism and Python

- Where to begin?
  - What problem are you trying to solve?
    - Big data?
    - Big computation?
    - Is it “Embarrassingly parallel”?
- Four ‘taster tutorials’
  - **Part 1: Multiprocessing**
  - **Part 2: Numba**
  - **Part 3: Mpi4py**
  - **Part 4: Cython (+OpenMP)**

https://github.com/dvalters/RSE18-Python-Parallel-workshop

## Let’s go!

- Tutorials in Jupyter Notebook format
  - Introduction + Four Mini Tutorials
- Links to the GitHub repository
- Use a Jupyter notebook or Ipython or editor
- **Experiment! Break the examples!**
  - **Discussion session at end?**
  - **Share your other Python parallel programming experiences!**
  - **Blog/write up?**

https://github.com/dvalters/RSE18-Python-Parallel-workshop

## Part 1: Multiprocessing

- Python’s built in multi-processing library
- https://docs.python.org/3.4/library/multiprocessing.html

## Part 2: Numba

- Numpy optimising library
- Uses Python decorators
- Auto-parallelisation features
- http://numba.pydata.org/

## Part 3: mpi4py

- Interface to the *Message-passing Interface*
- Distributed memory parallelism
- Cluster computers
- https://mpi4py.readthedocs.io/en/stable/

## Part 4: Cython + parallelism

- Cython – superset of Python with C-like type features
- Also compiler that compiles Python/Cython code to binaries
- Parallelism with OpenMP (shared-memory parallelism)
- http://cython.org/