

Global scale hydrological modelling at 100 m, 1 h resolution, in Python

Kor de Jong^{1,2} Marc van Kreveld² Debabrata Panja² Oliver Schmitz¹
Derek Karssenberg¹

¹Department of Physical Geography
Faculty of Geosciences
Utrecht University, The Netherlands

²Department of Information and Computing Sciences
Faculty of Science
Utrecht University, The Netherlands

EGU21-7154, session HS2.5.1

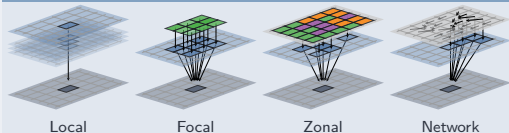


Utrecht University



Global scale hydrological modelling at 100 m, 1 h resolution, in Python

Rasters and map algebra operations



Asynchronous Many-Tasks



Fork-join



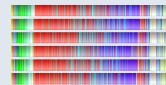
Asynchronous many-tasks

- Represent work by many interdependent tasks
- Allow work from multiple operations to be scheduled concurrently
- Overlap I/O latencies with computations
- LUE framework: C++ API, Python package
- +, *, and, or, sqrt, slope, focal_mean, zonal_max, flow_accumulation, ...

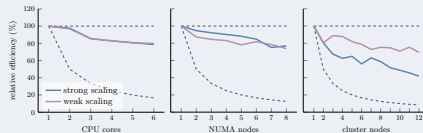
PyCatch + MERIT Hydro data



- 1 h time steps, 3 arc-second (≈ 100 m) raster cells (87.600×84.000 cells)
- Model is a regular Python script
- Model uses 12 nodes (1152 CPU cores)



Subset PyCatch trace



Scalability of wildfire model

Kor de Jong (k.dejong1@uu.nl), Marc van Kreveld, Debabrata Panja, Oliver Schmitz, Derek Karssenberg

Utrecht University, lue.computationalgeography.org

Base map ©OpenStreetMap contributors

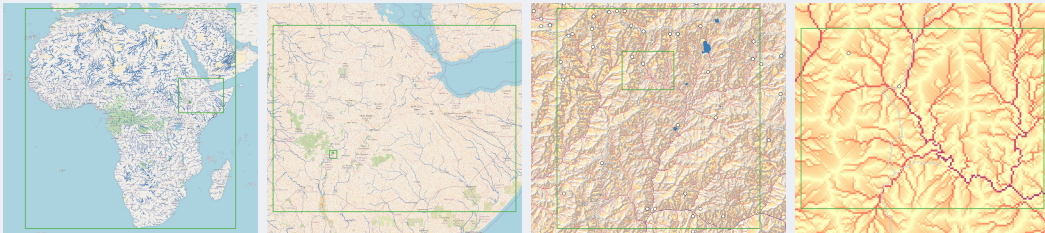
PyCatch

- Simulation of interception, evapotranspiration, surface storage, infiltration, subsurface and overland flow [Lana-Renault and Karssenberg, 2013]
- Model ported from PCRaster Python package to LUE Python package
- Time steps of 1 hour
- Runs on laptops (small datasets) and clusters (large datasets) now

Case-study: Africa at 100 m, 1 h resolution

- MERIT Hydro data, 3 arc-second resolution, ≈ 90 m at equator, [Yamazaki et al., 2019]
- 87.600×84.000 raster cells
- Proof of concept: unrealistic input data
- Ran on 12 nodes, containing 1152 (real) CPU cores in total
- Latency of a single time step (1 h) is currently about 2.5 minutes (compute only)

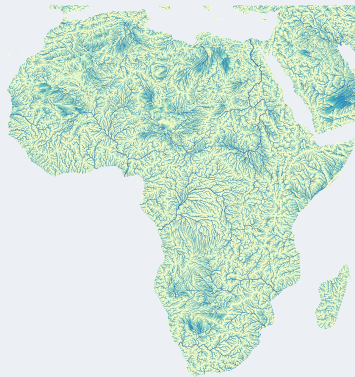
Case-study: Africa at 100 m, 1 h resolution



- Flow accumulation at 3 arc-second resolution

Case-study: Africa at 100 m, 1 h resolution

- Scalability of new flow accumulation operations and the whole model still need to be determined
- Scalability of other operations and another case-study model are good [De Jong et al., 2021]



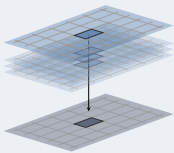
Runoff at 3 arc-second resolution

Environmental modelling

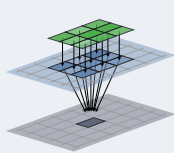
- Numerical simulation models
- Earth and environmental sciences
- Forward simulation: $\text{state}_{t+1} = f(\text{state}_t)$
- Example processes / systems: soil erosion, soil water content, plant seed dispersal, human exposome, crowd simulation, ...

Framework approach

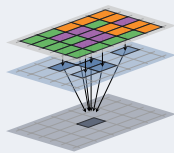
- A target user is the **domain expert**
- Model development interface as simple as possible, e.g.: map algebra in field based modelling: `a_raster = sqrt(another_raster + a_scalar)`
- Data models for representing state + modelling operations for translating state
- To the framework developer **the model is not known!**



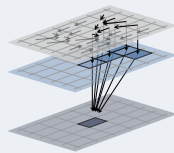
(a) Local operation



(b) Focal operation



(c) Zonal operation



(d) Network operation

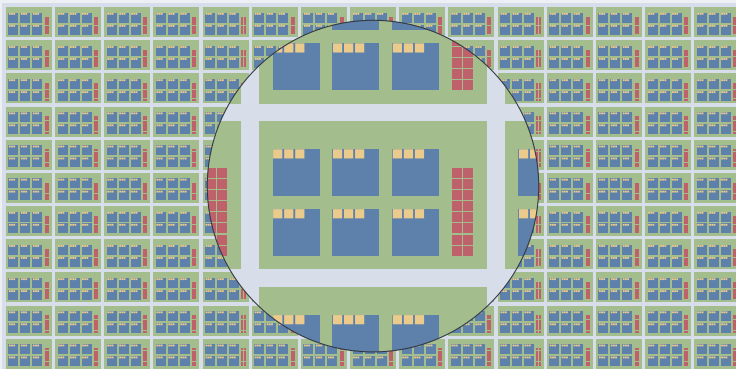
Asynchronous Many-Tasks (AMT)

- Spawn asynchronous tasks on all workers, that together perform all work
- Also define dependencies between the tasks
- Let AMT runtime schedule tasks
- We use the HPX C++ library implementing the AMT programming model and runtime [Kaiser et al., 2020]

Advantages to other approaches we seek:

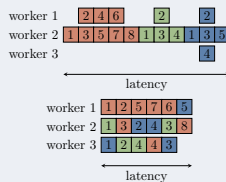
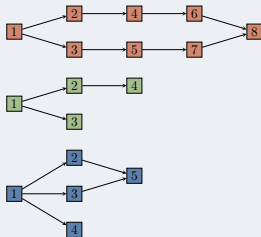
- Better programmability: single API, allow to focus on modelling operations
- Similar / better use of all workers (scalability)

Asynchronous Many-Tasks (AMT)



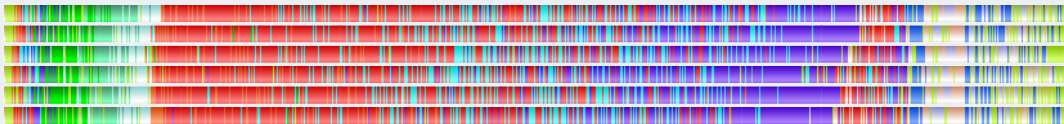
Develop for a single  abstract machine containing multiple  processes and the resulting software runs on platforms ranging from laptops to clusters

Asynchronous Many-Tasks (AMT)



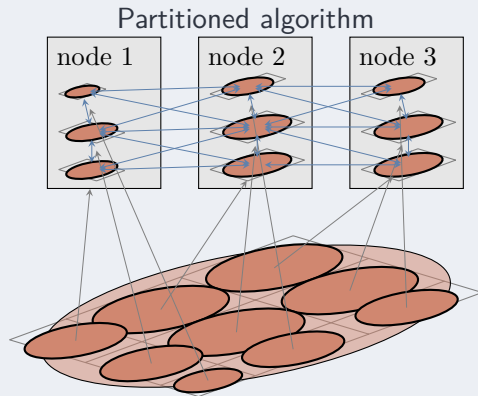
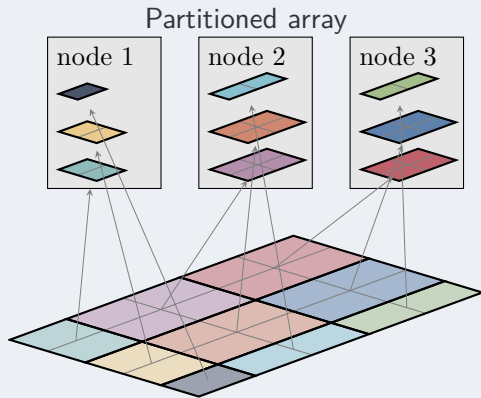
- Artificial example with same-sized tasks and their interdependencies from three operations
- Tasks from multiple operations can run in parallel(!)
- More concurrent tasks $\implies$ more opportunity for the runtime to turn concurrency into parallelism $\implies$ lower model latency

Asynchronous Many-Tasks (AMT)



- Trace shows tasks of different modelling operations using in the PyCatch model running in parallel, on 6 CPU cores (vertical), through time (horizontal)
- No cores are idling during this subset of the trace
- Task size is in the order of μs / ms

Asynchronous Many-Tasks (AMT)

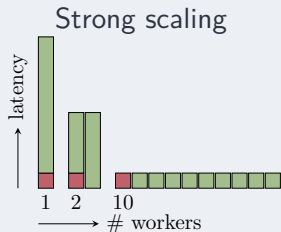


- Array partitions and computations are distributed over processes
- Task size is dependent on partition size

Scalability

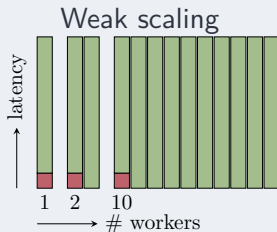
Design, implement and asses scalability of:

- Local, focal, zonal, and network operations
- Example models



Linear scaling:

$$n \text{ workers} \Rightarrow \frac{\text{latency}}{n} \Rightarrow \text{speedup} == n \Rightarrow \text{efficiency} == 100\%$$



Linear scaling:

$$n \text{ workers} \Rightarrow \text{latency} \times 1 \Rightarrow \text{efficiency} == 100\%$$

LUE software

- All mentioned modelling operations are part of the LUE framework software
- LUE provides APIs for C++ and Python
- LUE is open source licensed and free to use
- lue.computationalgeography.org

Some of the remaining challenges

- Scalable I/O: parallel, and latencies hidden behind computations
- Additional modelling operations: `kinematic_wave`, `dynamic_wave`, `spread`, ...

References I

- 
- De Bakker, M. P., De Jong, K., Schmitz, O., and Karssenber, D. (2017).
Design and demonstration of a data model to integrate agent-based and field-based modelling.
Environmental Modelling & Software, 89:172–189.
- 
- De Jong, K. and Karssenber, D. (2019).
A physical data model for spatio-temporal objects.
Environmental Modelling & Software, 122:104553.
- 
- De Jong, K., Panja, D., Van Kreveld, M., and Karssenber, D. (2021).
An environmental modelling framework based on asynchronous many-tasks: Scalability and usability.
Environmental Modelling & Software, 139:104998.
- 
- Kaiser, H., Diehl, P., Lemoine, A. S., Lebach, B. A., Amini, P., Berge, A., Biddiscombe, J., Brandt, S. R., Gupta, N., Heller, T., Huck, K., Khatami, Z., Kheirhahan, A., Reverdell, A., Shirzad, S., Simberg, M., Wagle, B., Wei, W., and Zhang, T. (2020).
HPX - The C++ standard library for parallelism and concurrency.
Journal of Open Source Software, 5(53):2352.
- 
- Lana-Renault, N. and Karssenber, D. (2013).
PyCatch: component based hydrological catchment modelling.
Cuadernos de Investigación Geográfica, 39(2):315–333.

References II



Yamazaki, D., Ikeshima, D., Sosa, J., Bates, P. D., Allen, G. H., and Pavelsky, T. M. (2019). MERIT Hydro: A high-resolution global hydrography map based on latest topography dataset. *Water Resources Research*, 55(6):5053–5073.