



Performance Optimization

Programming for Data Science

Org Stuff

Deadline to hand in your coding solutions (PDS)

- 07.02.2020, for everyone!

Colloquium (KPD)

- Presentations will be on 21. + 28.01.2020
- Topics will be assigned on 14.01.2020

Please also tell your fellow students.

Performance, Performance, Performance

Parallelize

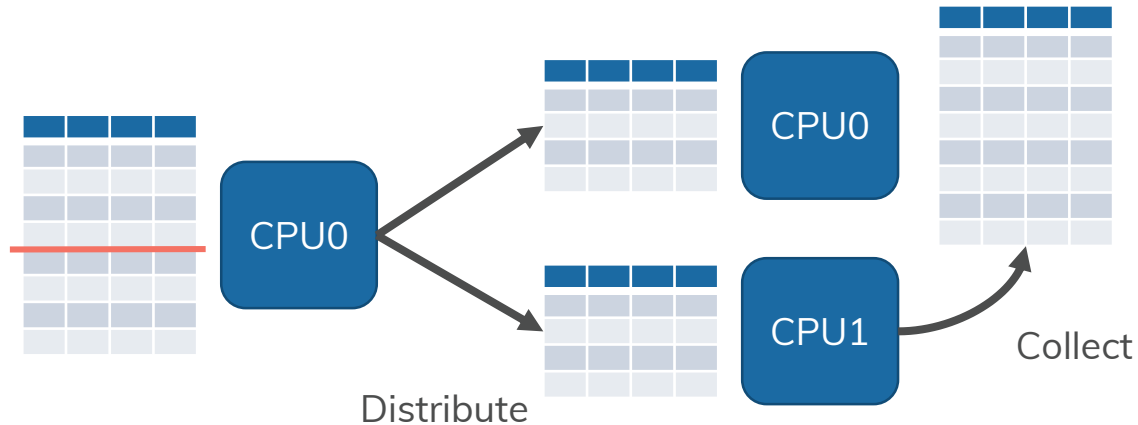
Push to C

Provide Vectorization

Parallelize

Most Data Science tasks are highly parallel.

- Horizontal data parallelism



Distribute + Collect
can be slow!

- R: parallel, python: multiprocessing

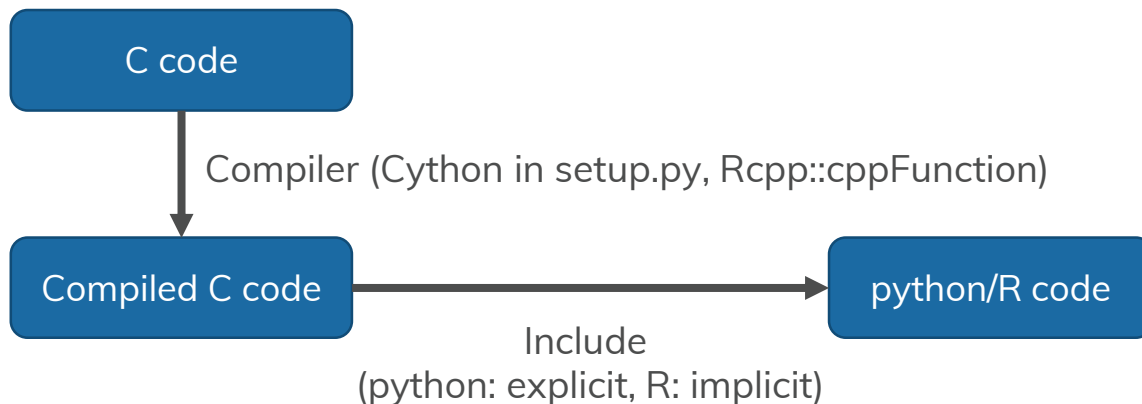
Push to C

R and python are generally slow

- Interpreted languages
- Weakly typed

C is fast

- Compiled language
- Strongly typed
- Convenient data types from R/python need to be converted. (Done in provided libs)



Provide Vectorization

A for loop can be replaced by a vectorized approach

- Most of the time: matrix solution
- F.e. Euclidean distance

```
for x in X:  
    for y in Y:  
         $\forall i \text{ in } \text{len}(x): \text{sqrt}(\sum (x[i]-y[i])^2)$ 
```

```
M = combinatorial stack columns (X,Y)  
 $\text{sqrt}(\sum (M[\text{columns from } x] - M[\text{columns from } y])^2)$ 
```

- R: , python: numpy (Not numpy.vectorize!)

Task

Step 0

- Generate two datasets with 2000 two-dimensional points each. (python: `numpy.random.rand`, R: `rnorm`)
- Implement a function* for calculating the Euclidean distances between the two data sets. Only use out-of-the-box functionalities. Measure its run time.

Step 1

- Parallelize* the function from step 0 (python: `multiprocessing.starmap`, R: `parLapply`). Measure its run time.

Step 2

- Implement a C function* (python: edit `euclid.pyx` → run `python3 setup.py build_ext --inplace` → from `euclid` import `euclid_c`, R: inline code) for calculating the Euclidean distances. Measure its run time.

Step 3

- Implement a vectorized version* of the Euclidean distance. Measure its run time.

*use your own implementation https://cython.readthedocs.io/en/latest/src/tutorial/cython_tutorial.html#primes

Package suggestions

R

- Rcpp
- parallel
- microbenchmark

python3

- numpy
- Cython
- setup.py (provided)
- multiprocessing
- timeit

Exercise Appointment

We compare and discuss the results

- Tuesday, 07.01.2020,
- Consultation: 19.12.2019,
- Please prepare your solutions! Send us your code!

If you have questions, please mail us:

claudio.hartmann@tu-dresden.de Orga + Code

lucas.woltmann@tu-dresden.de Tasks + Python

lars.kegel@tu-dresden.de Tasks + R

