Executing code on columnar data: the translation problem and formats that help

Jim Pivarski

Princeton – DIANA

February 8, 2017

Scope of computability

Three types of data transformations:

Flat: apply N-argument function to each element of N aligned arrays.

Known in the Numpy community as a "ufunc."

Explode: emulate (nested) for loops by replicating data in one array so that it becomes aligned with another array.

Reduce: emulate reducer functions (sum, mean, max...) by combining elements of an array so that it becomes aligned with an outer level of structure.

Scope of computability

diana hep

Three types of data transformations:

Flat: apply N-argument function to each element of N aligned arrays.

Known in the Numpy community as a "ufunc."

Explode: emulate (nested) for loops by replicating data in one array so that it becomes aligned with another array.

Reduce: emulate reducer functions (sum, mean, max...) by combining elements of an array so that it becomes aligned with an outer level of structure.

What's missing?

"Repeat until convergence." Whatever determines "convergence" may be different for each element of the array: they'd all wait for the last one, anyway.

Flat transformations

```
import ctypes
import numpy
import numba

libMathCore = ctypes.cdll.LoadLibrary("/opt/root_v6.06.08/lib/libMathC

ChisquareQuantile_ctypes = libMathCore._ZN5TMath17ChisquareQuantileEdd
ChisquareQuantile_ctypes.argtypes = (ctypes.c_double, ctypes.c_double)
ChisquareQuantile_ctypes.restype = ctypes.c_double

@numba.vectorize(["f8(f8, f8)"], nopython=True)
def ChisquareQuantile_Numpy(p, ndf):
    return ChisquareQuantile_ctypes(p, ndf)

p = numpy.random.uniform(0, 1, int(1e6))

result = [ChisquareQuantile_ctypes(pi, 100) for pi in p]
# 10.30 seconds

result = ChisquareQuantile_Numpy(p, 100)
# 3.22 seconds
```