Subject: Re: Speed does matter
Posted by on Mon, 17 Oct 2016 07:17:20 GMT
View Forum Message <> Reply to Message

Well, just saw a very ill-informed blog article from harrisgeospatial, where the anonymous author pits IDL against non-MKL Python.
While the recommended solution is to use Anaconda which is now MKL-compiled by default !
The article pretends that IDL is quicker than Python ...
Let us state the obvious : Among its category IDL is right now one the slowest "serious" software, I'd say roughly at least one order of magnitude.

Some have been pestering for years Exelis and Harris to simply recompile their software with Intel-MKL.
Right now I am enjoying a 40 speedup on SVD over IDL by using the IDL-Python bridge.

here is the culprit :

"The Amazing Race!

Wednesday, September 28, 2016
It wasn't so long ago that the IDL-Python bridge was introduced to IDL 8.5. It was with this new version, that I got my first experience programming with Python and testing the IDL-Python bridge. Through the past year it has been exciting to see the new changes and improvements that have become a part of the bridge.

Some of these new features include:

-Better error catching with the IDL-Python bridge

-Enhanced Jupyter notebook that allows for the development of full IDL programs and Python code in the same environment

-Improved support for variables passing back and forth

With all the time I have spent with Python, I have always wondered what some of the advantages are between Python and IDL. One thing that I have commonly heard several engineers say was that IDL was much faster than Python. For this blog, I decided to put that to the test and see how Python and IDL really compared to one another.

Before talking about the test, I do just want to explain things a bit about how it was set up and

some potential caveats about the processing times that will be shown. With the tests I created, I did my best to choose tests that were comparable between IDL and Python. Since I'm no expert at Python, there very well may have been other methods that could be faster than what I will show. Most of the pieces I included in the test are things I found easily by doing a web search - meaning that most of the approaches I used were the most common programming methods that people are likely using. This shows how much faster IDL might be than a general program than something that someone might write in Python.

The test:

Here is what was actually tested between IDL and Python with an array of [10000,10000] or 10000*10000 elements

-Array creation time

-Type conversion times

-Index array creation times (i.e. [0,1,2,3,4...,n-1])

-Incrementing array values of all elements by 1

-Complex math expression with array (exact equation: sqrt(sin(arr*arr)))

-Single threaded for IDL and multithreaded

-Array element access times (i.e. setting y = arr[i])

-Simple image processing filter times (filters: sobel, roberts, prewitt)

The results:

Average array creation time (seconds):

Python : 0.213000 +/- 0.00953933

IDL : 0.0936666 +/- 0.0155028

Total time (seconds):

Python : 0.639000

IDL : 0.281000

Python/IDL time ratio: 2.27402

Average array data type conversion time (seconds):

Python : 0.171333 +/- 0.0155028

IDL : 0.0730000 +/- 0.00866031

Total time (seconds):

Python : 0.514000

IDL : 0.219000

Python/IDL time ratio: 2.34703

Average index array creation time (seconds):

Python : 0.229000 +/- 0.00866031

IDL : 0.124667 +/- 0.0160104

Total time (seconds):

Python : 0.687000

IDL : 0.374000

Python/IDL time ratio: 1.83690

Average increasing array value time (seconds):

Python : 0.0933333 +/- 0.000577446

IDL : 0.0313334 +/- 0.000577377

Total time (seconds):

Python : 0.280000

IDL : 0.0940001

Python/IDL time ratio: 2.97872

Average complex math statements (1 thread) time (seconds):

Python : 6.36967 +/- 0.0645319

IDL : 8.34667 +/- 0.0155028

Total time (seconds):

Python : 19.1090

IDL : 25.0400

Python/IDL time ratio: 0.763139

Average complex math statements (8 thread) time (seconds):

Python : 6.34400 +/- 0.0321871

IDL : 1.93933 +/- 0.00923762

Total time (seconds):

Python : 19.0320

IDL : 5.81800

Python/IDL time ratio: 3.27123

Average loop through array element time (seconds):

Python : 11.5290 +/- NaN

IDL : 3.29100 +/- NaN

Total time (seconds):

Python : 11.5290

IDL : 3.29100

Python/IDL time ratio: 3.50319

Average image processing routines time (seconds):

Python : 15.3660 +/- 0.0829635

IDL : 1.39900 +/- 0.0238955

Total time (seconds):

Python : 46.0980

IDL : 4.19700

Python/IDL time ratio: 10.9836

Conclusion:

In short, IDL significantly outperformed Python will all the speed tests apart from the complex math statement. However, IDL has access to built in multithreading for large arrays and, with multithreading enabled, IDL outperforms Python significantly when using all available cores.

Below is the IDL code used to compare the processing speed of IDL and Python. To use it you will

need a few Python modules which can be found at the beginning of the procedure "python_test". "