Subject: float function unexpectedly slow
Posted by timothyja123 on Wed, 12 Mar 2014 23:56:44 GMT
View Forum Message <> Reply to Message

Hi guys,

I've spent the last day or so squeezing every last big of performance out of one of my code paths. I've speant a large amount of time staring at IDL's profiler (with is great by the way) and I'm starting to get to a point were most time is speant inside IDL's own procedures which means there isn't much more room for me to tweak anything further.

Anyway to get to the point one thing I have discovered this morning is that the built in float() function seems to be unexpectedly slow.
For example my profiler shows the following:

float() calls (mostly stright string to float conversions)
86,640 = 117.13ms

at first glance this seems acceptable however when I then comapre this to strmatch calls things start to look like there is room for improvement in the float() function

strmatch() calls
85,215 = 34.46ms

So strmatch() is around 4x faster on average in my use cases.
Ofcourse the speed of strmatch() is dependend on the complexity of the regular expresion and the length of the string its searching but one would still assume it would always be slower than a float() call. Is this a reasonable assumption?
Is there any reason float() would be slower than something like strmatch()?

I'd like to get some opinions before I consider sending a support request about this.

Thanks,
Tim

Extra Notes:

I did a comparison between the IDL and Python function to see if my assumptions are resonable.

Results
Python - 86,000 calls = 31ms
IDL - 86,000 calls = 62ms (about 2x faster than what I'm seeing in my real program but still 1/2 the speed of Python)

Code used for comparison

Python

```
from datetime import datetime

tstart = datetime.now()

range_count = range(0, 86000)
float_str = '363.491'

for x in range_count:
 float_val = float(float_str)

tend = datetime.now()
print(tend - tstart)
```

IDL

```
-------
pro float_speed_test

  startTime = systime(/seconds)

  float_str = '363.491'

  for i=0, 86000 do float_val = float(float_str)

  finishTime = systime(/seconds)
  timeSpent = finishTime - startTime
  print, 'Time: ', timeSpent

end
```