Article 3 Reference Critic

Nikolai Perlt

December 2020

Intro

In this report I will try to further improve on the excellent report Quick-Sort Performance written by Michael Ejdal Lundsgaard[1]. I will do this by finding references that I believe would help improve the article, and further validate the point the original author wanted to illuminate.

C# versus Python runtime operations

In the original article the author concluded that C# is 14x faster than Python. However, this is only when comparing C# and Python on quick sort algorithm. Is this still applicable when compared to other algorithms, or other forms of computer operations. If the article had references another article also concluding this, it would further strengthen the statement that C# is a significantly faster language than Python. To this I have found an article that I believe accomplishes this purpose[2].

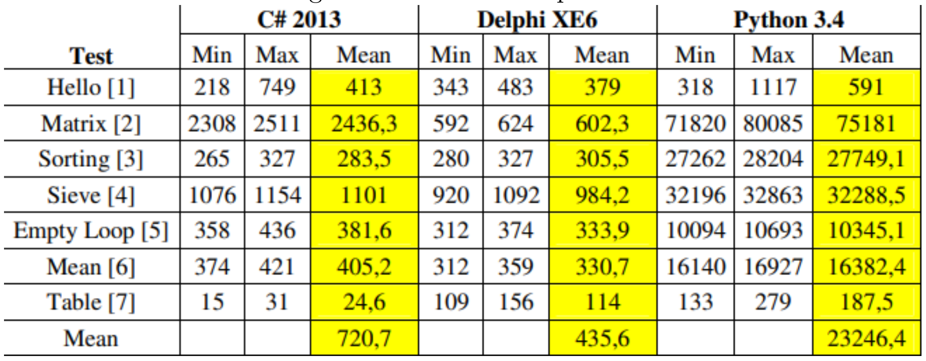
The article measures on multiple different metrics runtimes of C# and python. The ones I will discuss and compare to the original article are.

- Sorting 10000 elements with selection sort.
- Multiplication of two matrices.
- Printing Hello World on the screen 5000 times.

The main aspect I would focus on in the reference article, will be sorting since it is the driving force of the original article. We can see from table1 taken from the reference article, that the reference article as well can conclude that sorting is magnitudes faster using C# instead of Python. This would have further solidified the main point of the original article, that C# is faster than Python when comparing sorting algorithms. Not only that, but also that this is not only true for quick sort, but other sorting algorithms as well.

I also believe the article could benefit from quickly mentioning the remaining two metrics. These are still measurably faster when using C#, however still not on the same scale as sorting. By showing the reader this aspect, it would

Figure 1: Runtime comparison



	C# 2013			Delphi XE6			Python 3.4		
Test	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Hello [1]	218	749	413	343	483	379	318	1117	591
Matrix [2]	2308	2511	2436,3	592	624	602,3	71820	80085	75181
Sorting [3]	265	327	283,5	280	327	305,5	27262	28204	27749,1
Sieve [4]	1076	1154	1101	920	1092	984,2	32196	32863	32288,5
Empty Loop [5]	358	436	381,6	312	374	333,9	10094	10693	10345,1
Mean [6]	374	421	405,2	312	359	330,7	16140	16927	16382,4
Table [7]	15	31	24,6	109	156	114	133	279	187,5
Mean			720,7			435,6			23246,4

supplementary show C# speed advantage is not only applicable to sorting but also other forms of operations. This would in turn make the article more usable to more readers. There are some aspects to keep in mind when comparing the original article with the reference article. The original article never mentions what version of Python nor C# it is running. Since the Original article was written in 2020, and the reference article written in 2015, it is a very real possibility that we are comparing two different versions of the languages. This could mean that Python has lowered the gap between the two languages, thereby making the comparison between the two articles not as precise.

Pypy

The author not only compares python to C#, but also pypy, and throughout the article some statements about pypy are made that i believe could use sources. Some of these statements are

- Pypy being a JIT compiler.
- That pypy does not have any benefits on the first run through of the code.

These statements are never backed by any sources, but by using a source such as[3], the reader would be left with no doubt as to the validity of the claims. None of the statements are false, however by having external sources backing up the claims it would further cement the article as reputable.

Outro

Overall I think this is an incredible article written by Michael Ejdal Lundsgaard. It's an interesting topic that is well explained and well documented, by the author. However, I believe that if the two references mentioned in this critic could be incorporated, it would even further improve the quality of the article.

References

[1] Michael Ejdal Lundsgaard. Quick-sort performance c# vs python. 2020. `https://cphbusiness.mrooms.net/pluginfile.php/491289/mod_resource/content/1/Artikel%203%20Quick-Sort%20Performance%2C%20C%23%20vs%20Python.pdf`.

[2] Abdulkadir KARACI. A performance comparison of c# 2013, delphi xe6, and python 3.4 languages. *Int J Progr Lang Appl*, 2015. `https://wireilla.com/papers/ijpla/V5N3/5315ijpla01.pdf`.

[3] The PyPy team. Pypy features. `https://www.pypy.org/features.html`.