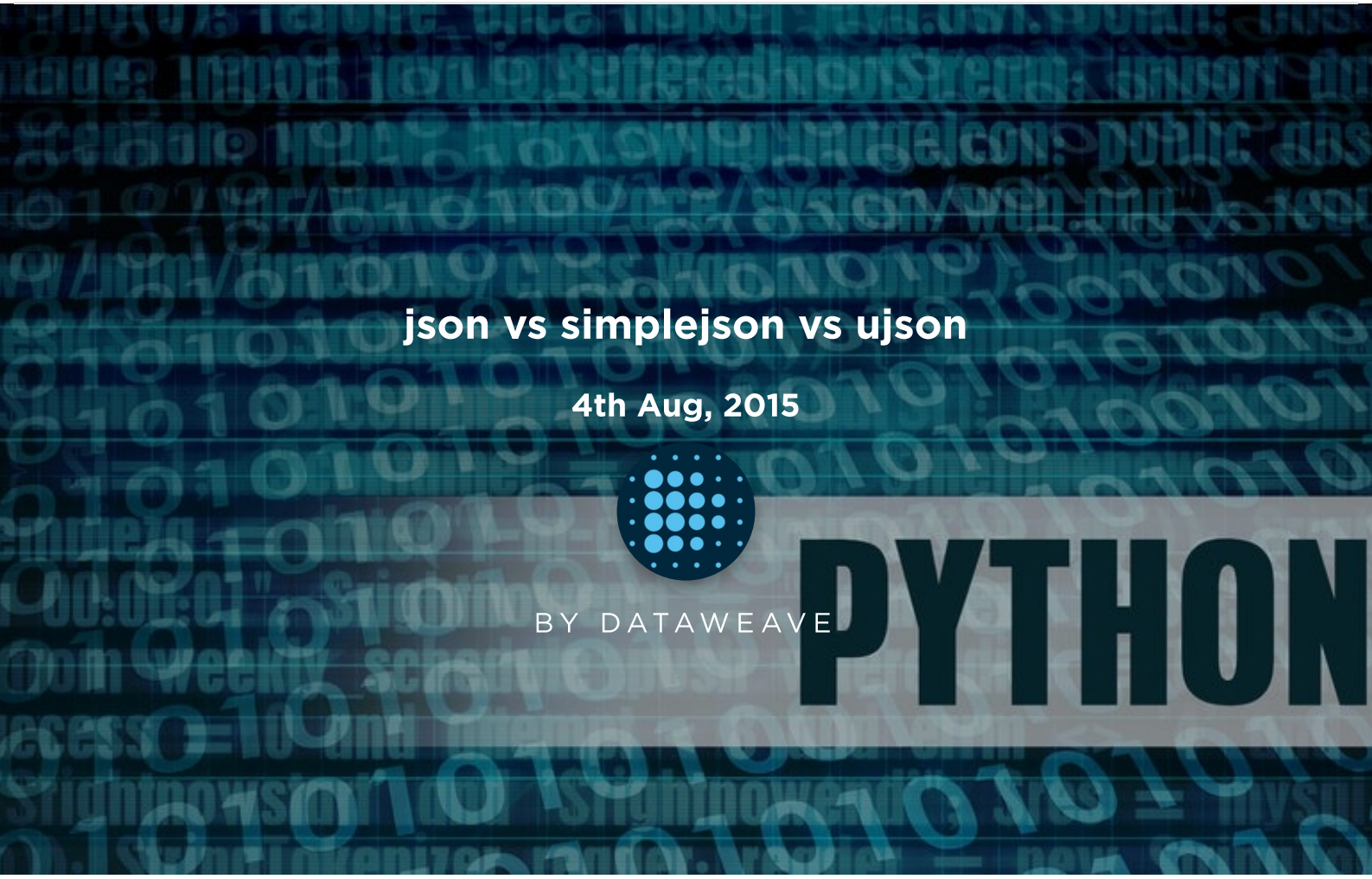

json vs simplejson vs ujson

4th Aug, 2015

Without argument, one of the most common used data model is JSON. There are two popular packages used for handling json — first is the stockjsonpackage that comes with default installation of Python, the other one issimplejson which is an optimized and maintained package for Python. The goal of this blog post is to introduce *ultrajson* or *Ultra JSON*, a JSON library written mostly in C and built to be extremely fast.

We have done the benchmark on three popular operations — **load**, **loadsanddumps**. We have a dictionary with 3 keys — id, name and address. We will dump this dictionary using **json.dumps()** and store it in a file. Then we will use **json.loads()** and **json.load()** separately to load the dictionaries from the file. We have performed this experiment on 10000, 50000, 100000,200000, 1000000 dictionaries and observed how much time it takes to perform the operation by each library.

DUMPS OPERATION LINE BY LINE

Here is the result we received using the json.dumps() operations. We have dumped the content dictionary by dictionary.

Number of dicts	json (in ms)	simplejson (in ms)	ujson (in ms)
10000	84.106	115.876	25.891
50000	395.163	579.576	122.626
100000	820.280	1147.962	246.721
200000	1620.239	2277.786	487.402
500000	3998.736	5682.641	1218.653
1000000	7847.999	11501.038	2530.791

We notice that json performs better than simplejson but ultrajson wins the game with almost 4 times speedup than stock json.

DUMPS OPERATION (ALL DICTIONARIES AT ONCE)

In this experiment, we have stored all the dictionaries in a list and dumped the list using **json.dumps().**

Number of dicts	json (in ms)	simplejson (in ms)	ujson (in ms)
10000	21.674	21.941	14.484
50000	102.739	118.851	67.215
100000	199.454	240.849	138.830
200000	401.376	476.392	270.667
500000	1210.664	1376.511	834.013
1000000	2729.538	2983.563	1830.707

simplejson is almost as good as stock json, but again ultrajson outperforms them by more than 60% speedup. Now lets see how they perform for load and loads operation.

LOAD OPERATION ON A LIST OF DICTIONARIES

Now we do the load operation on a list of dictionaries and compare the results.

Number of dicts	json (in ms)	simplejson (in ms)	ujson (in ms)
10000	47.040	8.932	10.706
50000	165.877	44.065	45.629
100000	356.405	97.277	105.948
200000	718.873	185.917	205.120
500000	1699.623	461.605	503.203
1000000	3441.075	949.905	1055.966

Surprisingly, simplejson beats other two, with ultrajson being almost close to simplejson. Here, we observe that simplejson is almost 4 times faster than stock json, same with ultrajson.

LOADS OPERATION ON DICTIONARIES

In this experiment, we load dictionaries from the file one by one and pass them to the **json.loads()** function.

Number of dicts	json (in ms)	simplejson (in ms)	ujson (in ms)
10000	99.501	69.517	15.917
50000	407.873	246.794	76.369
100000	893.989	526.257	157.272
200000	1746.961	1025.824	318.306
500000	8446.053	2497.081	791.374
1000000	8281.018	4939.498	1642.416

Number of dicts	json (in ms)	simplejson (in ms)	ujson (in ms)
10000	99.501	69.517	15.917
50000	407.873	246.794	76.369
100000	893.989	526.257	157.272
200000	1746.961	1025.824	318.306
500000	8446.053	2497.081	791.374
1000000	8281.018	4939.498	1642.416

Again ultrajson steals the show, being almost 6 times faster than stock json and 4 times faster than simplejson.

That is all the benchmarks we have here. The verdict is pretty clear. Use simplejson instead of stock json in any case, since simplejson is well maintained repository. If you really want something extremely fast, then go for ultrajson. In that case, keep in mind that ultrajson only works with well defined collections and will not work for un-serializable collections. But if you are dealing with texts, this should not be a problem.

This post originally appeared **here.**

- DataWeave Marketing

4th Aug, 2015

DATA ENGINEERING