influxDB写入吞吐量测试报告

workload设计：

本测试基于官方对于influxDBv1.x版本的压力测试改编，旨在探究influxDB在单机场景下的最大写入吞吐量，并将所得经验应用到实际生产中。

实际场景下，每个传感器所采集的值构成一个时间序列，不同序列通过measurement和tag进行区别。一个树莓派可连接多个传感器，每隔定时将传感器的值发送到云数据库，于是可运用Batch的思想，将多个时间线的数据采集后打包发送，从而大量减少请求次数。

根据以上考虑，得到测试中的基本变量：

1. SeriesNumber 时间序列数：等于系统中的传感器数目；
2. WorkersNumber 并发协程数：等于树莓派数目，模拟多个树莓派完成定时发送，在测试中随PPS和BatchSize的设置而改变；
3. BatchSize 每个请求中所含数据点个数；
4. PPS：points per second，每秒发送的数据点个数；

代码实现思路：

首先保证每个Worker每秒发送一个Batch，那么如果设置BatchSize为1万，每秒发20万个点，就需要20个Workers并行。

如果设置SeriesNumber为10万，则每个Worker负责5000个Series，Worker在1秒内顺序生成这5000个Series的数据点，依次追加到send buffer中，当buffer长度达到BatchSize时（进行了两轮生成），就打包发送。若用时不足1秒，就等一个时间差补足1秒再进行接下来的数据生成。

每个Worker自注册时起，1分钟后停止，记录所有成功发送的数据点数目，将所有Worker工作量求和并计算每秒能够写入多少数据点。

测试环境：

机型：MacBook Pro16

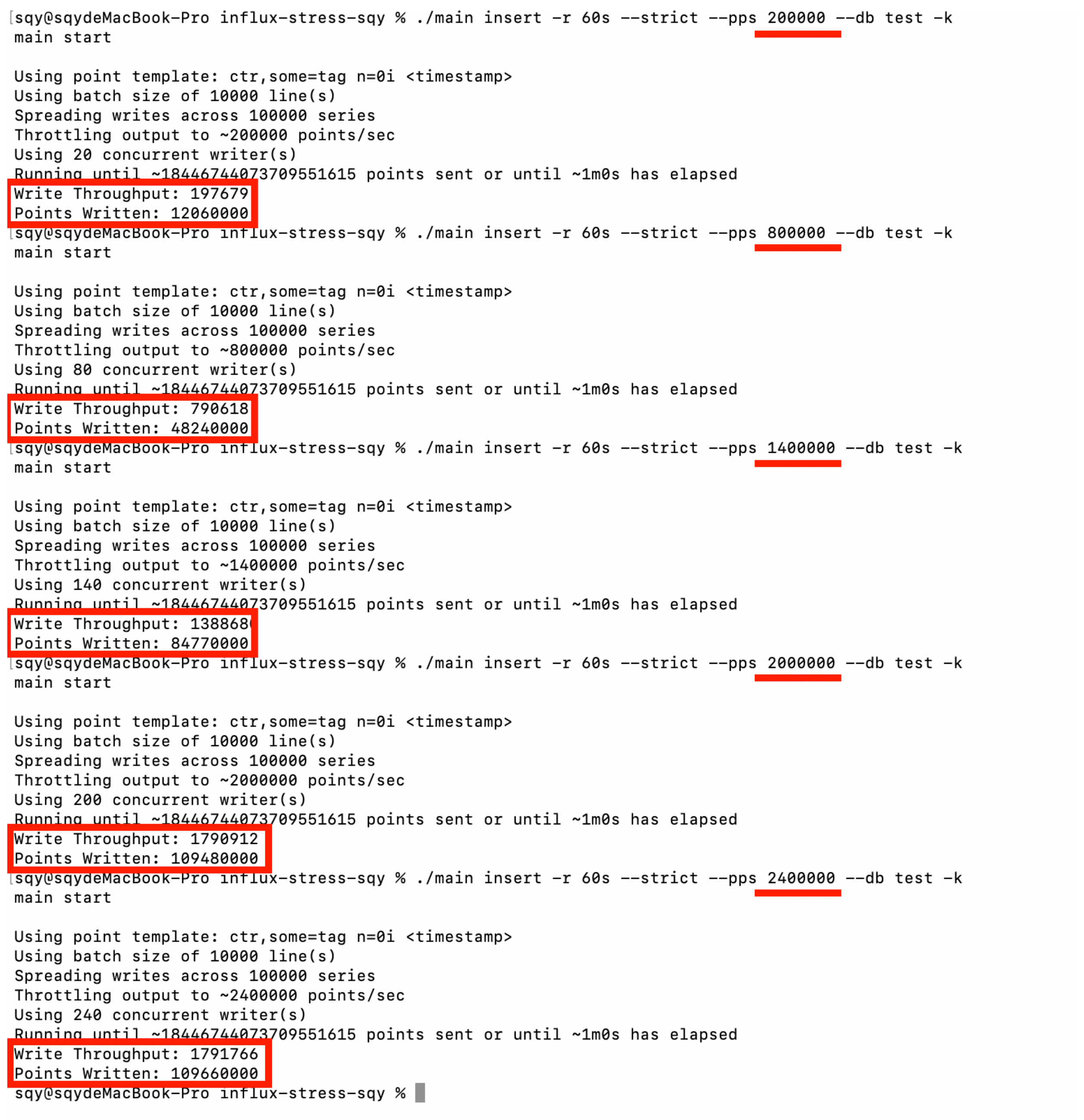
系统：macOS 10.15.4

influxDB：v2.0开源单机版本

测试过程及结果：

设置SeriesNumber为10万，BatchSize为1万，PPS分别为20万、80万、140万、200万、240万，所得Write Throughput 以及 Point Written 如下图所示：

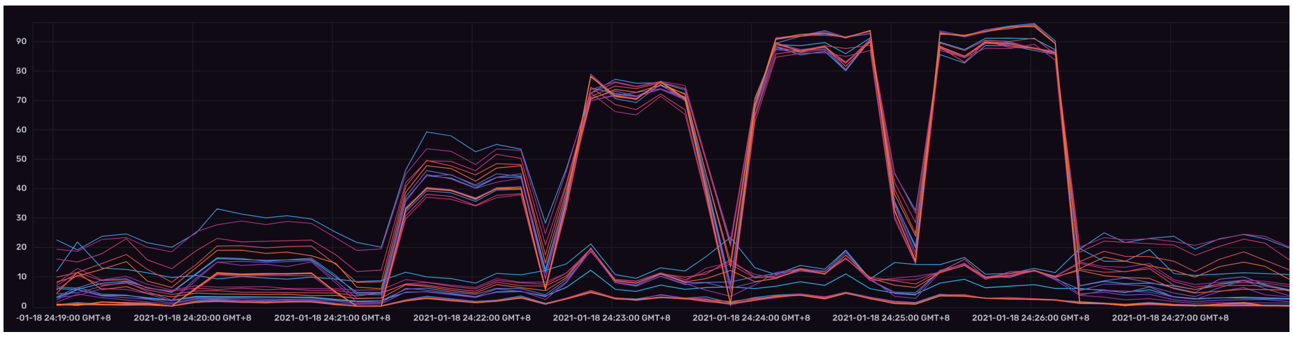
可知在此条件下写入吞吐量最大值为180万每秒。此时系统性能瓶颈为influxDB。



```
[sqy@sqydeMacBook-Pro influx-stress-sqy % ./main insert -r 60s --strict --pps 200000 --db test -k
main start

Using point template: ctr,some=tag n=0i <timestamp>
Using batch size of 10000 line(s)
Spreading writes across 100000 series
Throttling output to ~200000 points/sec
Using 20 concurrent writer(s)
Running until ~18446744073709551615 points sent or until ~1m0s has elapsed
Write Throughput: 197679
Points Written: 12060000
[sqy@sqydeMacBook-Pro influx-stress-sqy % ./main insert -r 60s --strict --pps 800000 --db test -k
main start

Using point template: ctr,some=tag n=0i <timestamp>
Using batch size of 10000 line(s)
Spreading writes across 100000 series
Throttling output to ~800000 points/sec
Using 80 concurrent writer(s)
Running until ~18446744073709551615 points sent or until ~1m0s has elapsed
Write Throughput: 790618
Points Written: 48240000
[sqy@sqydeMacBook-Pro influx-stress-sqy % ./main insert -r 60s --strict --pps 1400000 --db test -k
main start

Using point template: ctr,some=tag n=0i <timestamp>
Using batch size of 10000 line(s)
Spreading writes across 100000 series
Throttling output to ~1400000 points/sec
Using 140 concurrent writer(s)
Running until ~18446744073709551615 points sent or until ~1m0s has elapsed
Write Throughput: 138868
Points Written: 84770000
[sqy@sqydeMacBook-Pro influx-stress-sqy % ./main insert -r 60s --strict --pps 2000000 --db test -k
main start

Using point template: ctr,some=tag n=0i <timestamp>
Using batch size of 10000 line(s)
Spreading writes across 100000 series
Throttling output to ~2000000 points/sec
Using 200 concurrent writer(s)
Running until ~18446744073709551615 points sent or until ~1m0s has elapsed
Write Throughput: 1790912
Points Written: 109480000
[sqy@sqydeMacBook-Pro influx-stress-sqy % ./main insert -r 60s --strict --pps 2400000 --db test -k
main start

Using point template: ctr,some=tag n=0i <timestamp>
Using batch size of 10000 line(s)
Spreading writes across 100000 series
Throttling output to ~2400000 points/sec
Using 240 concurrent writer(s)
Running until ~18446744073709551615 points sent or until ~1m0s has elapsed
Write Throughput: 1791766
Points Written: 109660000
sqy@sqydeMacBook-Pro influx-stress-sqy %
```

同时我们用influxDB的监控功能实时测量了系统的CPU利用率，如图所示：

可见用户进程CPU占用率在五次测试中分别约为：30%、50%、75%、93%、93%。后两次占用率没有明显变化。


90
80
70
60
50
40
30
20
10
0
-01-18 24:19:00 GMT+8
2021-01-18 24:20:00 GMT+8
2021-01-18 24:21:00 GMT+8
2021-01-18 24:22:00 GMT+8
2021-01-18 24:23:00 GMT+8
2021-01-18 24:24:00 GMT+8
2021-01-18 24:25:00 GMT+8
2021-01-18 24:26:00 GMT+8
2021-01-18 24:27:00 GMT+8