1. 630 TCT 1.0版本

一、测试资源

1.支撑端配置

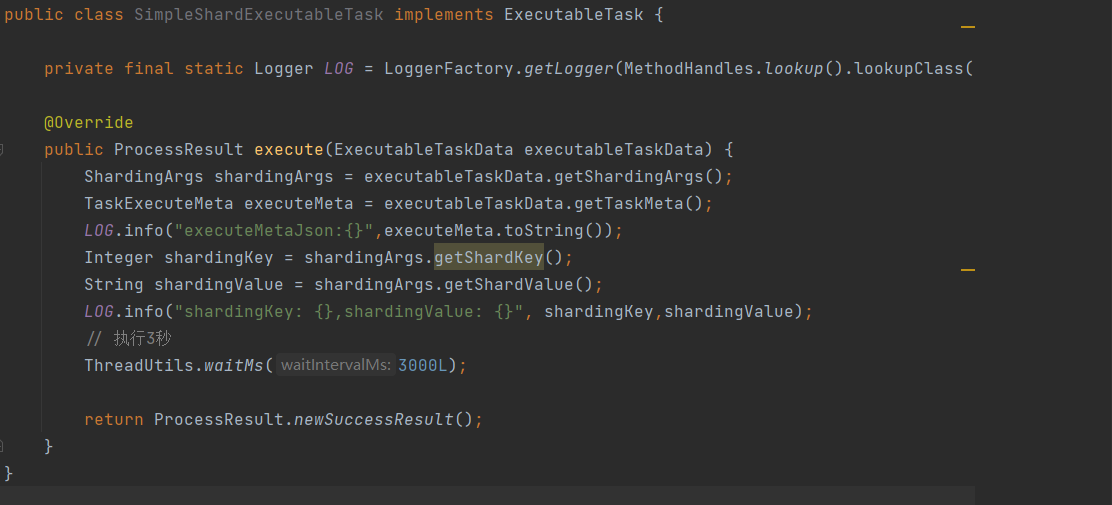
|  |  |  |  |
| --- | --- | --- | --- |
| 组件 | Cpu/核心数 | 内存/G | 数量 |
| task-trigger | 8 | 16 | 2 |
| task-server | 8 | 16 | 2 |
| task-access | 8 | 16 | 2 |
| TCT-mysql | 8 | 16 | 2 |
| task-console | 8 | 16 | 2 |
| task-ops | 8 | 16 | 2 |
| TCT-RABBITMQ | 8 | 16 | 2 |

2.用户端配置

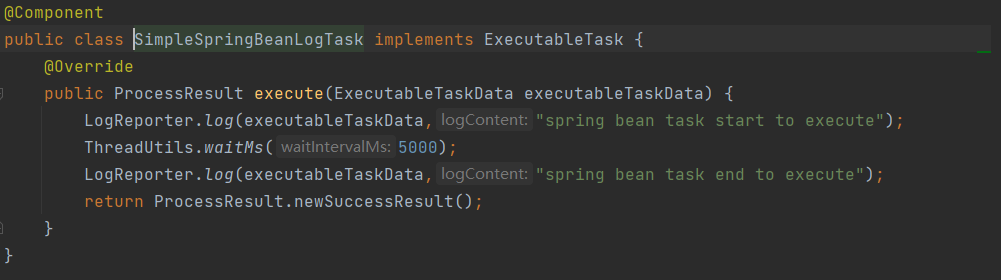
CPU/内存：1C/1G

3.应用侧测试demo

3.1分片任务代码：



3.2随机任务代码：



二、测试方法

通过python脚本自动化创建任务及工作流，通过循环创建任务的方式，配合多个任务参数的变更，其中包括任务执行类型、任务执行频率。

测试脚本：



同时，对支撑端组建的资源消耗情况，应用及组件日志是否异常等方面，进行观察。

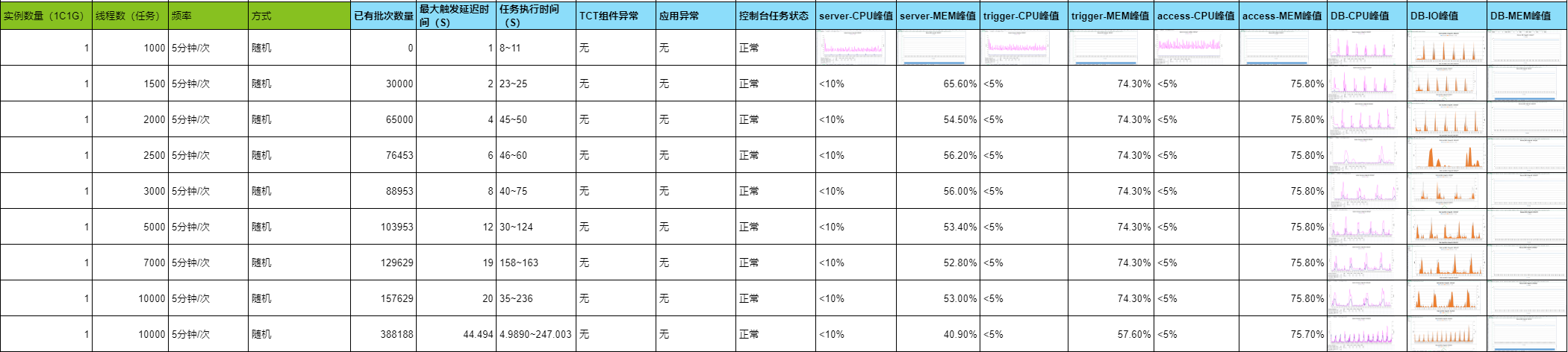
应用侧代码通过TCT的demo实现，其中模拟单任务执行耗时为5s；分片执行耗时为3s，分片数量为3。

三、测试用例

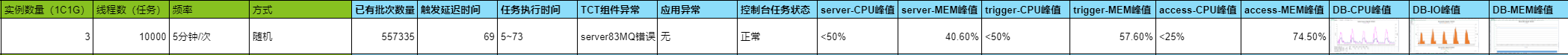
1.随机任务

下表所示为单任务随机方式执行的测试纪录，通过测试纪录可观察到随着任务数的增多，最大触发延迟时间、任务执行时间均随之增加，但TCT组件资源表现整体平稳，未出现较大波动，且组件及应用日志、控制台状态均未出现异常。

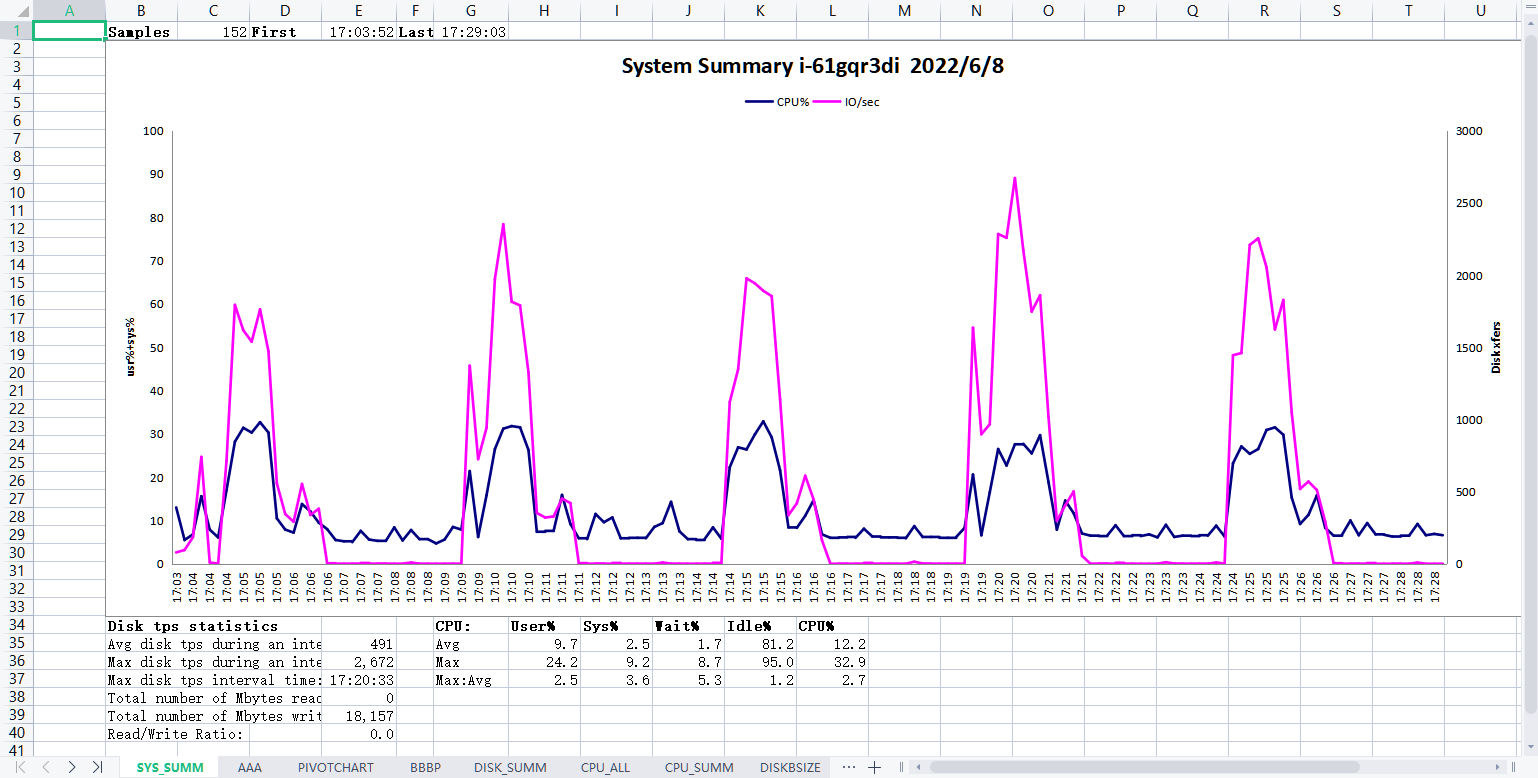
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 实例数量 （1C1G） | 线程数 （任务） | 批次数量 | 频率 (分钟/次) | 方式 | **已有批次数量** | **最大触发延迟时间（S）** | **任务执行时间 （S）** |
| 1 | 1000 | 5 | 5 | 随机 | 0 | 1 | 8~11 |
| 1 | 1500 | 5 | 5 | 随机 | 30000 | 2 | 23~25 |
| 1 | 2000 | 5 | 5 | 随机 | 65000 | 4 | 45~50 |
| 1 | 2500 | 5 | 5 | 随机 | 76453 | 6 | 46~60 |
| 1 | 3000 | 5 | 5 | 随机 | 88953 | 8 | 40~75 |
| 1 | 5000 | 5 | 5 | 随机 | 103953 | 12 | 30~124 |
| 1 | 7000 | 5 | 5 | 随机 | 129629 | 19 | 158~163 |
| 1 | 10000 | 5 | 5 | 随机 | 157629 | 20 | 35~236 |
| 1 | 10000 | 5 | 5 | 随机 | 388188 | 44.494 | 4.9890~247.003 |



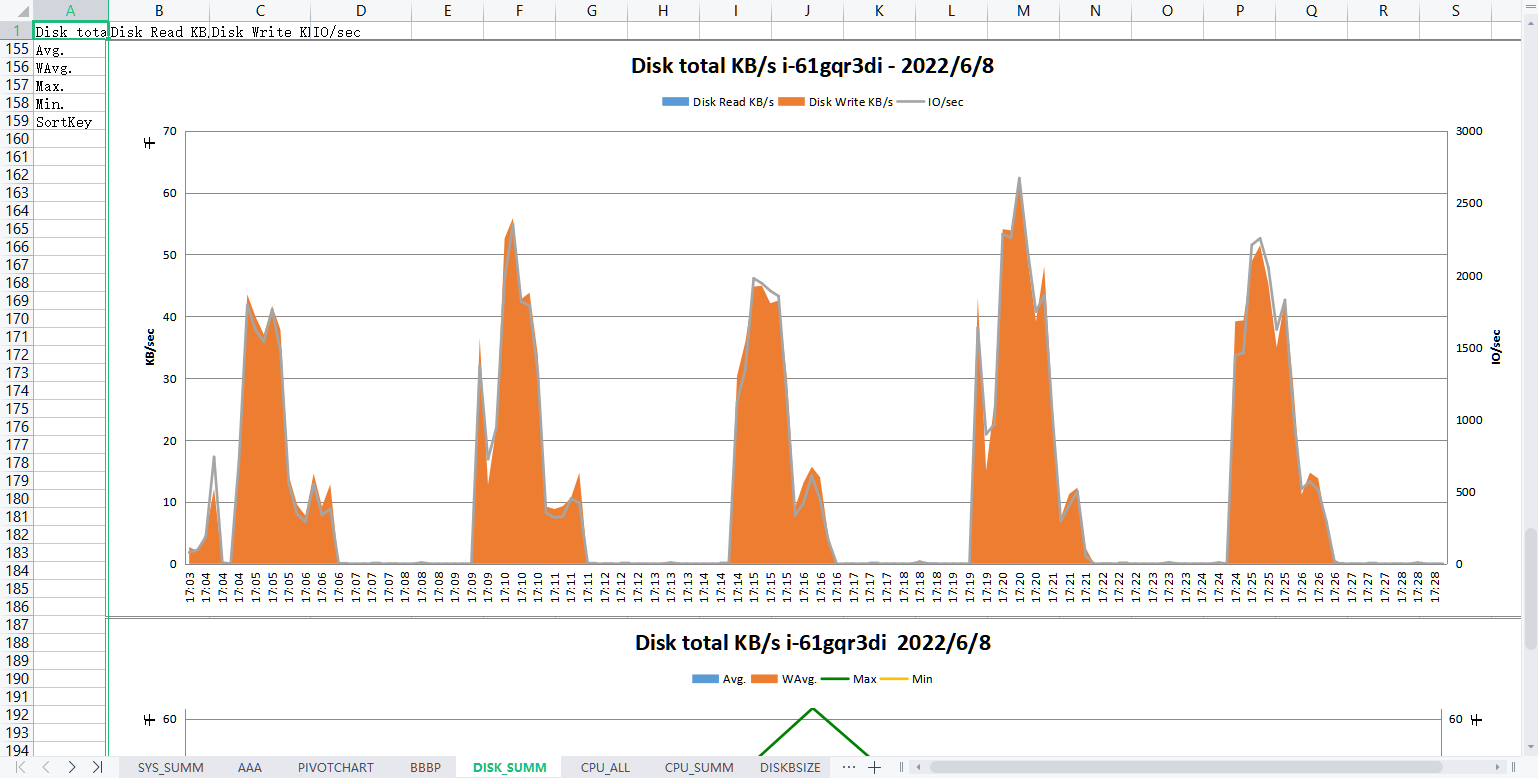
同时为了测试应用实例个数对数据的影响，将应用实例个数增加为3个，发现实例数量主要影响任务执行的平均耗时，其它部分影响不明显。



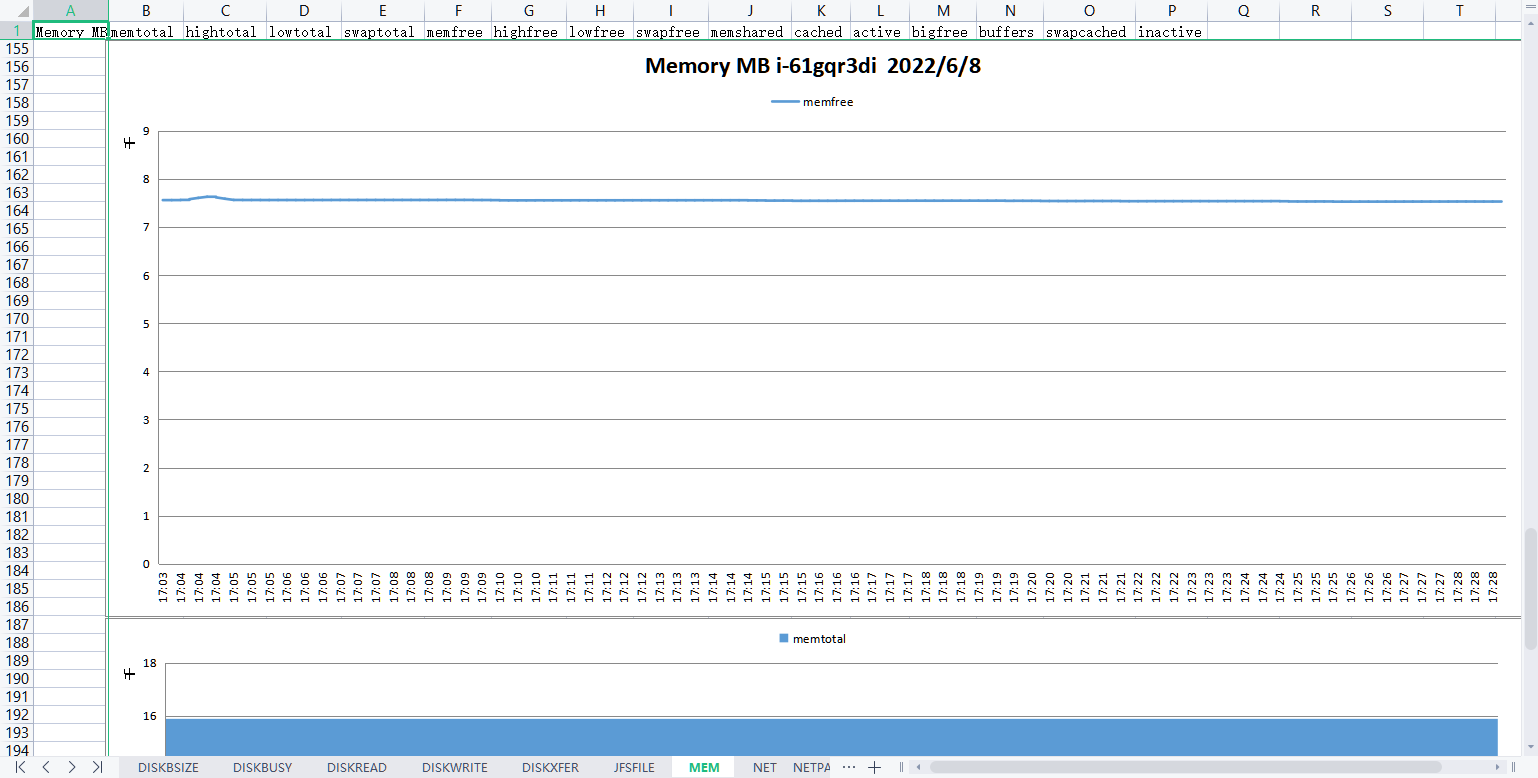
**DB-CPU峰值**



**DB-IO峰值**



**DB-MEM峰值**



分片任务

分片任务+工作流

四、测试问题

MQ问题

1.任务并发数高导致MQ机器文件标识符不足

根据用例进行测试时发现，当任务并发数量达到1w时，rabbitmq发生消息堆积并报错，错误日志如下：

//补充错误日志截图



通过分析发现，原因为rabbitmq所在机器的文件描述符为默认1000，目前已修改为65535，并将此动作优化到部署步骤中。

1. 任务批次数据量大导致任务延迟触发

测试随机任务时发现随着任务执行批次不断增加，任务被延迟触发的时间也在不断增加，当达到批次数据达到15w条左右，任务最大的被延迟触发时间达到20秒左右，并且导致部分任务状态停留在waiting、dispatching。另外，分片任务的批次数量=任务数\*分片数量，所以触发延迟及状态流转延迟的现象更为明显。此问题在TCT2.0版本中已通过将批次数据迁移到ES存储解决。

五、测试结论

建议的任务数、执行频率等、分片数量（分不同类型）

运维方面：TCT巡检脚本