性能测试报告示例

性能测试

汪琳仙 | 4天前发布 | 5 | 0

本文是后端性能测试报告的示例，大家编写性能测试报告时可以参考。

性能测试报告模板请参考：后端性能测试报告模板

JMeter 使用文档请参考：JMeter 官方用户手册

1. 概述

1.1 测试目的

本次测试是针对 ThingMap 系统文件库相关功能进行的性能测试，目的是对系统的性能进行评估，并对性能测试过程中发现的缺陷进行修复。

1.2 测试范围

通过对实际业务的分析，ThingMap 文件库相关的性能测试点如下：

编号	性能场景	相关接口
1	上传效果模板	/api/renderer/add /group1/mapRenderers/${renderer_ref}/map.json /group1/mapRenderers/${renderer_ref}/preview.png
3	分享项目	/api/map/share/createMapJson
4	导出项目	/api/map/export /api/map/getMapTaskState /group1/mapZipResult/${task_export_id}/map.zip

1.3 测试目标

此次性能测试的目标是测试出满足以下性能指标的最大并发用户数。

测试类型	性能指标
负载测试	• 业务平均响应时间小于或等于5秒； • 测试过程中服务器CPU最大占用率小于80%； • 不存在内存泄露，且在性能测试结束后1小时内内存占用恢复到初始值（由于测试服务器空载情况下内存用率在80%左右，本次性能测试暂不对内存最大占用率做要求）； • 系统平均负载小于服务器核数； • 事务错误率为0。

2. 测试工具及环境

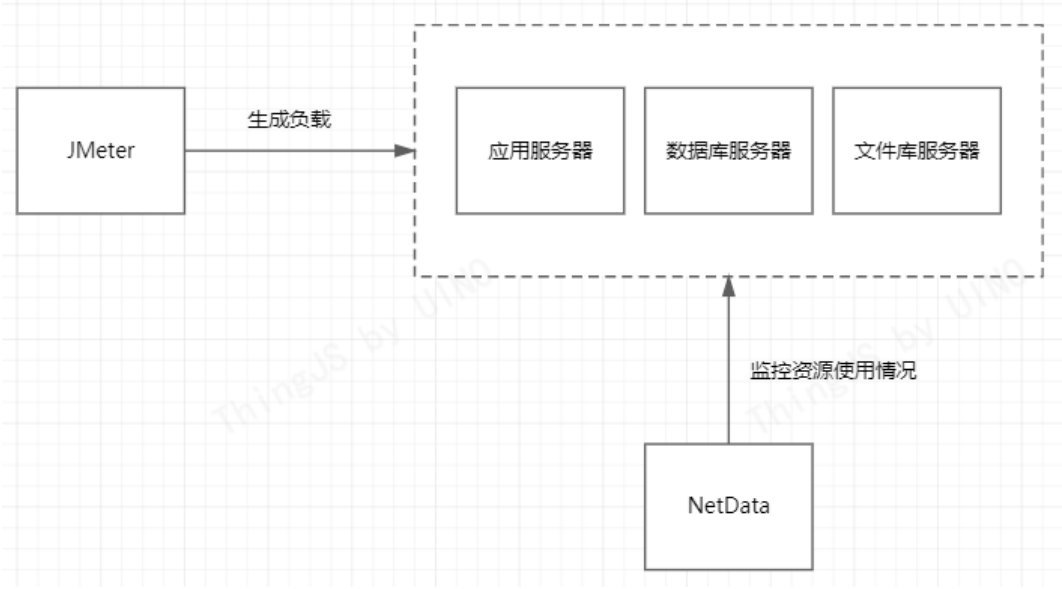
2.1 测试工具

编号	工具名称	用途及说明	厂商	版本
1	Jmeter	负载生成	Apache	5.1.1
2	NetData	监控服务器资源		1.25.0

2.2 测试环境

2.2.1 测试环境架构

测试环境的应用服务器、数据库服务器、文件库服务器部署在同一台 Linux 虚拟机上，由于本次性能测试的并发用户数比较少，使用单个压力机即可。



2.2.2 配置信息

软 硬 件 配 置 信 息	
硬件配置	**应用服务器/数据库服务器/文件服务器（10.209.1.206）** CPU(12C): Intel(R) Core(TM) i5-10400 CPU @ 2.90GHz 内存：32G 硬盘：1TB **负载机** CPU(12C):Intel(R) Core(TM) i5-10400F CPU @ 2.90GHz 内存：16GB 硬盘：1.2TB
软件配置	**应用服务器**

	CentOS 64位、MySQL 8.0.32-0ubuntu0.22.04.2 最大连接数 100 **文件库服务器** CentOS 64位、go-fastdfs **负载机** Windows10 64位、apache-jmeter-5.1.1、JDK 1.8.0
网络环境	负载机和服务器在同一个局域网中，可忽略网络的影响

2.3 测试人员

测试人员	职责
李四	构造测试数据、搭建测试环境、编写测试脚本
王五	执行测试、收集测试结果、输出测试报告

3. 测试方法

3.1 测试数据

性能场景	测试数据
上传效果模板	只包含基本图层效果的模板文件，大小为133KB
分享项目	效果模板应用太空旅客Day，只包含芜湖市的图层数据（建筑数量为30566）
导出项目	效果模板应用太空旅客Day，只包含芜湖市的图层数据（建筑数量为30566），导出地图包的大小为7.78MB

3.2 测试场景设计

本次性能测试只做了单场景的测试，即单独对上传效果模板、分享项目、下载项目的场景进行性能测试。为了更真实的模拟用户并发，测试脚本中使用了20个用户的token。由于用户登录接口会影响到单场景的测试结果，测试脚本中对token采用的是参数化方式而非关联方式，即先用脚本跑用户登录接口，将接口返回的token写入CSV文件，再用CSV文件对性能测试脚本进行参数化。

3.3 测试执行

本次共进行了一轮性能测试，先采用逐步增加负载的方式进行测试，并记录不同并发下的测试结果，包括聚合报告截图、资源监控截图。随着并发用户数的增加，当不满足性能测试指标时，再采用缓慢减少负载的方式测试出满足性能指标时可以支持的最大并发用户数。为了测试结果更加真实有效，对于相同的并发数采用多次（6次）测试取平均值的方式记录测试结果。

4. 测试结果

4.1 上传模板

并发用户数	响应时间平均值(秒)	响应时间最大值（秒）	90%业务的响应时间	错误率	CPU占用率	内存占用率

100	0.947	2.088	1.646	0	33.9%	81.3%
200	2.146	4.229	3.755	0	42.5%	81.7%
250	5.257	10.39	8.307	0	73.3%	81%
300	5.84	10.963	9.735	0	90.8%	80.7%

4.2 分享项目

并发用户数	响应时间平均值(秒)	响应时间最大值（秒）	90%业务的响应时间（秒）	错误率	CPU占用率	内存占用率
5	0.204	0.255	0.22	0	43.9%	83.3%
10	1.020	1.8	1.703	0	73.1%	82.7%
12	1.96	3.04	2.92	0	82.7%	83.4%
15	2.27	3.5	3.43	0	99.9%	82.7%

4.3 导出项目

并发用户数	响应时间平均值(秒)	响应时间最大值（秒）	90%业务的响应时间（秒）	错误率	CPU占用率	内存占用率
5	0.256	0.346	0.246	0	46.2%	79.2%
10	1.59	1.97	1.96	0	79.8%	79.1%
15	1.249	2.788	2.526	0	87%	79.2%
20	2.743	3.554	3.295	0	99.8%	79.5%
30	3.363	4.215	3.848	0	100%	80%

5. 测试结论

在满足性能指标的情况下，各个场景支持的最大并发数、90%业务的响应时间、CPU占用率如下：

性能场景	并发用户数(个)	90%业务响应时间(s)	CPU占用率
上传模板	250	8.307	73.3%
分享项目	10	1.703	73.1%
下载项目	10	1.96	79.8%

6. 性能瓶颈分析

6.1 上传效果模板场景分析

从图中可以看到，服务器内存的波动不大（空载情况下，内存的使用率在80%左右），而 CPU 的使用率从30.4%上升到了73.3%，其中iowait占48.7%。

再次执行脚本，查看服务器磁盘使用情况（见下图）。

```
Device:            rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda                  0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
sdb                  0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-0                 0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-1                 0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-2                 0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

Device:            rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda                  0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
sdb                  0.00     0.00    0.00    1.00     0.00     4.00     8.00     0.01    7.50    0.00    7.50   7.50   0.75
dm-0                 0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-1                 0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-2                 0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

Device:            rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda                  0.00     0.00    0.00    6.00     0.00    93.50    31.17     0.01    2.25    0.00    2.25   0.50   0.30
sdb                  0.00   188.50    2.50  176.50    10.00 13298.00   148.69     3.67   17.30   17.60   17.29   1.43  25.60
dm-0                 0.00     0.00    0.00    3.00     0.00    93.50    62.33     0.01    4.50    0.00    4.50   1.00   0.30
dm-1                 0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-2                 0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

Device:            rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda                  0.00     6.50    0.00   30.00     0.00   340.75    22.72     0.07    2.18    0.00    2.18   0.28   0.85
sdb                  0.00   138.00    6.00  384.00    24.00 16344.00    83.94     5.00   14.26   24.17   14.11   2.37  92.50
dm-0                 0.00     0.00    0.00   22.50     0.00   298.75    26.56     0.07    2.91    0.00    2.91   0.29   0.65
dm-1                 0.00     0.00    0.00   10.50     0.00    42.00     8.00     0.00    0.43    0.00    0.43   0.19   0.20
dm-2                 0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
```

从图中可以看到，sdb 磁盘设备的 %util 在测试过程中达到了92.5%，说明此时系统I/O已经满负荷了，磁盘I/O可能是性能的瓶颈。

6.2 导出项目场景分析

在15个并发测试过程中，服务器资源使用情况如下图所示（来自 NetData 资源监控）。

从图中可以看到，服务器内存的波动不大（空载情况下，内存的使用率在80%左右），而 CPU 的使用率上升到了87%，其中 user 占85.1%。

再次执行脚本，查看服务器中容器对资源的使用情况（见下图），可以看到 thingmap-cloud-api 对 CPU 的使用为861.57%（12核）。

```
d7ae1419376a   fastdfs                      0.01%   644MiB / 31.16GiB   2.02%   0B / 0B          991MB / 1.59GB   30
10966a3774a9   tiles-server-mbtiles-terrain 0.00%   2.9GiB / 31.16GiB   9.31%   22.6MB / 167MB   51.1MB / 458kB   11
da5d06b5c1fe   jinchang-web                 0.00%   7.652MiB / 31.16GiB 0.02%   146kB / 0B       0B / 23kB        13
919ba34e7265   jinchang-server              0.05%   1.705GiB / 31.16GiB 5.47%   381MB / 11.5MB   199MB / 9.73kB   50
fc61c4515905   nginxlantu                   0.00%   7.805MiB / 31.16GiB 0.02%   860kB / 108MB    53.2MB / 8.19kB  13
58f7b015ad31   tiles-server-file            0.00%   24.43MiB / 31.16GiB 0.08%   202MB / 263MB    137MB / 587kB    11
633a48cdd9ff   thingmap-cloud-glb           0.00%   36.82MiB / 31.16GiB 0.12%   154kB / 7.76kB   87.8MB / 0B      10
0e356eef388e   thingmap-rabbit              0.14%   67.9MiB / 31.16GiB  0.21%   25.4MB / 24.8MB  136MB / 24MB     45
62445d799610   thingmap-cloud-glb-golang    0.00%   1.93MiB / 31.16GiB  0.01%   174kB / 13.2kB   25MB / 0B        7
d33b6ea3b2ab   tiles-server-mbtiles-vecotr  0.00%   135.3MiB / 31.16GiB 0.42%   85.7MB / 1.2GB   710MB / 318kB    11
dce8f6bd153b   tiles-server-mbtiles1.0      0.00%   1.901GiB / 31.16GiB 6.10%   6MB / 81.2MB     205MB / 300kB    11
```

进入thingmap-cloud-api 容器，查看进程情况（见下图），可以看到 pid 为1的 java 进程占用 CUP 最多。

```
top - 16:25:01 up 28 days,  8:23,  0 users,  load average: 2.09, 2.08, 1.44
Tasks:   3 total,   1 running,   2 sleeping,   0 stopped,   0 zombie
%Cpu(s): 97.4 us,  2.1 sy,  0.0 ni,  0.3 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0 st
KiB Mem:  32668676 total, 32392872 used,   275804 free,   445680 buffers
KiB Swap: 16449532 total,  3008512 used, 13441020 free.  5030576 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
    1 root      20   0 26.048g 8.156g   8216 S 813.0 26.2 138:08.25 java
 5447 root      20   0   21932   2020   1568 S   0.0  0.0   0:00.02 bash
 5453 root      20   0   23652   1532   1132 R   0.0  0.0   0:00.01 top
```

查看该进程的堆栈信息，发现有大量的线程处于等待状态（见下图）。

```
"http-nio-9003-exec-4191" #5275 daemon prio=5 os_prio=0 tid=0x00007fd7ac0f9000 nid=0x155b waiting on condition [0x00007fd6e09ca000]
   java.lang.Thread.State: TIMED_WAITING (parking)
        at sun.misc.Unsafe.park(Native Method)
        - parking to wait for  <0x000000034d330378> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject)
        at java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215)
        at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.awaitNanos(AbstractQueuedSynchronizer.java:2078)
        at java.util.concurrent.LinkedBlockingQueue.poll(LinkedBlockingQueue.java:467)
        at org.apache.tomcat.util.threads.TaskQueue.poll(TaskQueue.java:89)
        at org.apache.tomcat.util.threads.TaskQueue.poll(TaskQueue.java:33)
        at java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor.java:1066)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1127)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
        at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
        at java.lang.Thread.run(Thread.java:745)

"http-nio-9003-exec-4190" #5274 daemon prio=5 os_prio=0 tid=0x00007fd7ac13d000 nid=0x155a waiting on condition [0x00007fd7feaf1000]
   java.lang.Thread.State: TIMED_WAITING (parking)
        at sun.misc.Unsafe.park(Native Method)
        - parking to wait for  <0x000000034d330378> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject)
        at java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215)
        at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.awaitNanos(AbstractQueuedSynchronizer.java:2078)
        at java.util.concurrent.LinkedBlockingQueue.poll(LinkedBlockingQueue.java:467)
        at org.apache.tomcat.util.threads.TaskQueue.poll(TaskQueue.java:89)
        at org.apache.tomcat.util.threads.TaskQueue.poll(TaskQueue.java:33)
        at java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor.java:1066)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1127)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
        at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
        at java.lang.Thread.run(Thread.java:745)

"http-nio-9003-exec-4189" #5273 daemon prio=5 os_prio=0 tid=0x00007fd7ac0e0000 nid=0x1559 waiting on condition [0x00007fd6e8e4e000]
   java.lang.Thread.State: TIMED_WAITING (parking)
        at sun.misc.Unsafe.park(Native Method)
        - parking to wait for  <0x000000034d330378> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject)
        at java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215)
        at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.awaitNanos(AbstractQueuedSynchronizer.java:2078)
        at java.util.concurrent.LinkedBlockingQueue.poll(LinkedBlockingQueue.java:467)
        at org.apache.tomcat.util.threads.TaskQueue.poll(TaskQueue.java:89)
        at org.apache.tomcat.util.threads.TaskQueue.poll(TaskQueue.java:33)
        at java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor.java:1066)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1127)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
        at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
        at java.lang.Thread.run(Thread.java:745)
```

7. 优化建议

1）可以通过提高磁盘的性能来提升上传效果模板业务的性能，例如更换成固态硬盘、分布式部署等。

2）需要研发对导出项目的进程进行进一步的分析，定位导致导出项目性能差的具体原因。

3）建议不要将所有的服务部署在同一台设备上，合理的利用服务器资源，提高系统的可用性。