基于云计算的大规模性能测试服务平台

陈铁南 唐 震 王晓冉 任 凯 支孟轩
(中国科学院软件研究所软件工程技术研发中心 北京 100080)

摘 要 性能测试是通过自动化的测试工具模拟多种正常、峰值以及异常负载条件来对系统的各项性能指标进行测试。测试对象分为基准测试和非基准测试两种。大规模的性能测试受到所需的大量软硬件资源以及与此规模匹配的管理维护代价的限制,传统的性能测试采用一种1:20的微缩仿真模拟,但这种微缩仿真测试不充分,会带来严重的后果。采用云计算技术,使用其承诺的按需的廉价软硬件资源服务,来构建大规模性能测试服务平台,提供按需定制的测试服务。

关键词 云计算,测试即服务,性能测试,负载测试,QoS
中图法分类号 TP319 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2014.09.010

Large Scale Performance Test Service Platform Based on Cloud

CHEN Tie-nan TANG Zhen WANG Xiao-ran REN Kai ZHI Meng-xuan
(Technology Center of Software Engineering Institute of Software Chinese Academy of Science, Beijing 100080, China)

Abstract In software engineering, performance test is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. The object of performance test is divided to two group. One is benchmark, and the other is non-benchmark. As performance test needs significant input of software resowrces, hardware resources and corresponding input of management and maintenance, traditional performance test takes a measure of miniature simulation. But this kind of miniature simulation causes inadequate test, and brings with serious consequences. Based on cloud computing, with its promise of on-demand, low-cost software and hardware resource, we builded the large scale performance test service platform to provide test service that is on-demand and customized.

Keywords Cloud computing, Test as a service, Performance test, Load test, QoS

1 研究背景

自从有软件以来,测试就是必不可少的一部分。而性能测试更是对软件验收、评测必不可少的步骤,其重要性不言而喻。尤其是进入了互联网时代,互联网应用的开放性和用户行为以及规模的不确定性,使得对于性能测试的需求更为迫切。然而性能测试是一种典型的资源密集型,或者说是计算密集型的工作。性能测试所需的规模越大,所投入的软硬件资源以及与之相关的管理维护费用增长越快。然而大多数网络应用对性能测试的迫切需求却又只能望而却步。举一个例子来说明此问题:Myspace公司执行一次10万量级的性能测试,如果Myspace选择成熟的商业化性能测试软件Load runner进行此次测试,那么在每台pc模拟1000并发用户(每台pc模拟1000并发用户,几乎是业界标准)的前提下,在硬件投入上,Myspace需要购置1000台pc以及2台以上的应用服务器作为测试控制器和结果数据库;在软件投入上,Load runner每1000虚拟用户收费和每种负载协议收费均在30万美元以上,检测诊断等服务则需额外支付7万美元以上,其软件资源的总投入将超过3亿美金;与此同时Myspace还需要管理维护这1000台pc和2台以上的服务器以及Load runner软件。总的来看,性能需要投入的代价过于昂贵,使得性能测试成为了一种奢侈品。然而在互联网时代对于性能测试的需求又是如此迫切,所以人们采用了一种1∶20的微缩仿真模拟的方式进行性能测试。

这种微缩仿真模拟导致测试不充分而引发了许多严重的问题。例如一旦系统性能测试不充分,将导致系统上线后出现服务崩溃或服务不可用的状态,而这种系统状态恰恰是性能测试的用武之地。

随着云计算技术不断的发展成熟,其为性能测试带来了全新的解决方案。

云计算强调将所有的资源,无论软件资源、硬件资源,都作为服务提供,用户可以按时按需按量进行使用,然而这恰巧

到稿日期:2013-12-25 返修日期:2014-01-16 本文受国家自然科学基金(61173004,61100068),国家高技术研究发展计划(863)(2012AA011204),国家科技支撑计划资助项目(20112BAH14B02)资助。
陈铁南(1991-),男,硕士生,主要研究方向为网络分布式计算与软件工程,E-mail:chentienan12@otcaix.iscas.ac.cn;唐 震(1990-),男,博士生,主要研究方向为网络分布式计算与软件工程;王晓冉(1989-),女,硕士生,主要研究方向为网络分布式计算与软件工程;任 凯(1990-),男,硕士生,主要研究方向为网络分布式计算与软件工程;支孟轩(1988-),男,硕士,工程师。

解决了性能测试对软硬件资源的极大需求。

云计算强调互联网接入，即用户通过浏览器对服务进行访问与使用，在给用户提供更为友好、简单的用户接口的同时，也将测试用户从难以理解的日志以及异常中解放出来。

云计算强调资源托管，将软硬件资源托管给更为专业的运行维护管理团队，从而解决了性能测试所需的大量的管理维护费用。

故而性能测试即服务的理念也就逐渐清晰，基于云计算的大规模性能测试服务平台正是基于这一理念研发的。

2 研究内容

传统的性能测试工具一般由以下模块组成，即测试用例设计模块，用于设计生成测试脚本；负载控制模块，用于分发用例设计模块产生的测试脚本，并调度负载注入代理执行测试脚本；负载注入代理，用于真正地对待测系统产生压力，并且在对待测系统不断产生压力的同时，需要得到待测系统性能状态的量化度量(其中包括响应时间、吞吐率、错误率、正确执行率等)；性能监测模块，用于检测待测系统的资源状态量化度量(其中包括 CPU、内存、网络资源、硬盘资源等)；性能分析模块，则是用于收集负载注入代理反馈的性能度量以及性能监测模块反馈的资源度量，并对其进行分析，最后生成性能测试报告。待测系统，顾名思义，即是性能测试的对象。

图 1 即 Bench4Q 的系统架构图。

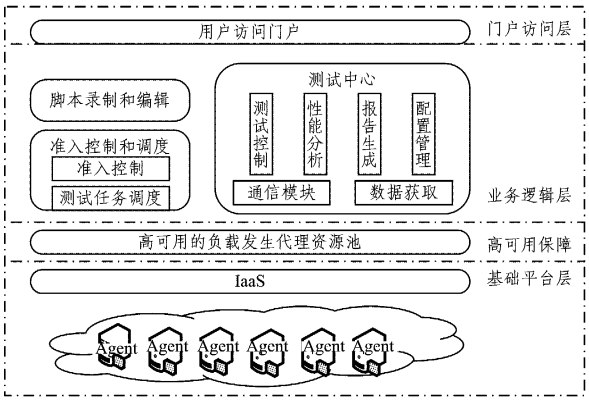

图 1 Bench4Q 系统架构图

下面将根据图 1 中的几个模块分别介绍其在大规模性能测试服务平台中的实现，以及在云化为更大规模的过程中遇到的挑战和采取的解决方案。

当然作为一个服务提供平台，提供一个测试用户自主访问门户是基本需求，Bench4Q 性能测试服务平台也正是提供这种基于浏览器的方式进行访问。

2.1 测试用例设计模块

2.1.1 测试脚本自动化生成

在传统的性能测试工具中，测试脚本采用一种人工进行编写的方式，这种方式不但出错的可能性高而且也极大地加重了测试人员的负担。

在 Bench4Q 大规模性能测试服务平台中，采取了一种基于 Web 代理的自动化的脚本录制功能。在这一功能的实现中，Web 代理在转发用户浏览器请求和待测系统的响应的过程中自动化地生成结构化的测试脚本。

测试脚本录制原理图如图 2 所示。

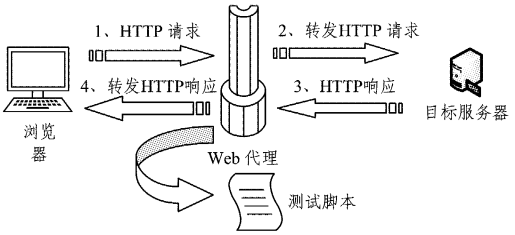

图 2 基于 Web 代理的脚本录制

2.1.2 设计复杂的测试场景

由于大多数的网络应用系统都有极高的复杂性，依靠单一的测试脚本未必能够将其测试充分，许多的网络应用系统都有其极高的测试需求，为了满足这种较高的测试需求，Bench4Q 对复杂测试场景的设计提供了支持。

Bench4Q 的复杂测试场景设计是通过组合测试脚本实现的。其主要目标即是模拟出如图 3 所示的资源争用及其可能导致的死锁、资源瓶颈等复杂测试场景。

复杂测试场景例如数据库死锁的产生如图 3 所示。

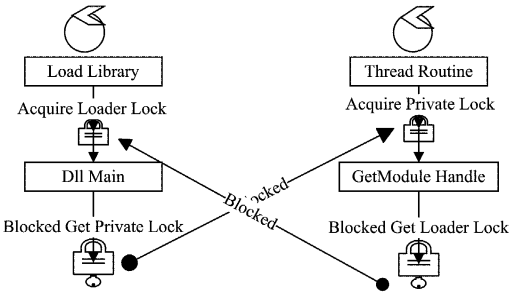

图 3 复杂测试场景—数据库死锁

2.1.3 小结

使用此种方式进行的复杂场景设计具有很高的灵活性，例如可以单独配置场景中每个脚本的运行时的热身时间、执行区间以及冷却时间。

2.2 负载控制模块

针对大规模的负载需求，负载控制模块需要维护一个规模由数十到数千的负载注入代理池，然而规模的增加使其维护的复杂性急剧增长。

1. 由于测试规模的不确定性，使得负载注入代理不足的情况可能时有发生，因此提供一个弹性可扩展的负载注入代理池。

2. 由于测试过程可能会很长，而且虚拟机和网络的不稳定性导致负载注入代理在测试过程中失效的可能性大大增加，因此十分有必要提供一个高可用的负载注入代理池。

以下将根据这两个部分进行详细介绍。

2.2.1 弹性可扩展的负载注入代理池

由于测试规模的不确定性，可能这一需求极大地超出了当前负载注入代理池中的最大规模。此时，性能测试服务平台应该在不影响其他测试用户的测试的情况下，使管理员可以动态以热插拔的方式进行负载注入代理，以满足大规模测试用户的负载需求。

其原理图如图 4 所示。

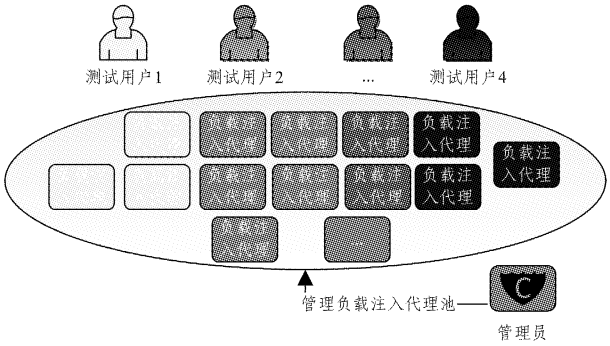

图 4　弹性可扩展的负载注入代理池

2.2.2　高可用的负载注入代理池

以一个实际场景为例，一个测试用户拟进行一次 7 * 24 小时的疲劳测试，如果在测试过程中出现了部分负载注入代理失效的情况，那么将导致失效点之后的测试结果和报告无效或者不满足用户的原始需求。

从这样的一个角度出发，可以看出提供一个高可用的负载注入代理池的重要性。

Bench4Q 采用一种轮询机制来保证节点的高可用特性，一旦测试过程中出现节点失效，备份节点就可以在一个轮询周期内完成替补切换，而这一轮询周期一般是在 15 秒到 30 秒的时间内完成。

其实效替补图如图 5 所示。

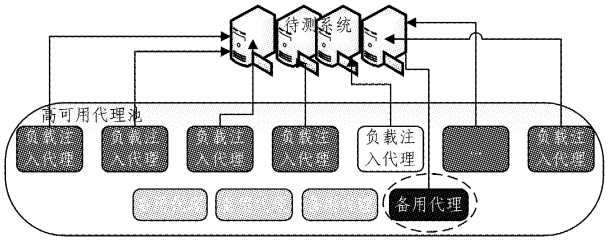

图 5　高可用的负载注入代理池

2.3　负载注入代理

负载注入代理的任务是整个测试的核心部分，其需要真正地对待测系统产生负载（即使用各种应用协议对待测系统进行服务请求），并需实时地收集待测系统的各项性能状态指标（如平均响应时间、吞吐率、错误率等）。

以常规的网络应用为例，其可能需要的协议就有如 HTTP(S)、FTP、Socket、IMAP 等，而针对 Web API 编程接口则又有 XML-RPC、RMI 等。不同的待测系统所需的协议可能不同，这对负载注入代理提出了最为关键的问题，即支持协议的可扩展性。

Bench4Q 采用一种插件化的方式来对不同的测试协议进行组织，Bench4Q 维持一个插件总线，那么一旦要新增一个负载测试协议，就只需向插件总线上动态添加，即可被负载注入代理识别并执行。

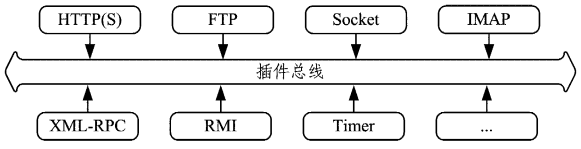

图 6　负载协议的插件化管理

2.3.1　负载注入代理的海量原始数据存储

负载注入代理在执行测试的过程中，需要记录每次请求（在 Bench4Q 中称之为一次 behavior）的发出时间、响应时间、响应情况（响应码是 200、404 或是 302 等）以备测试完成结束时的分析和报告形成。

对测试过程细粒度的记录，给 Bench4Q 的后期分析工作提供了极大的支持，同时也为其代理带来困扰，即在测试时间逐渐变长时数据量也就急剧增长。

经过统计每个负载注入代理（最大支持 1000 并发用户）每分钟运行产生的原始数据量约为 10M，那么每个负载注入代理运行 24 小时将产生大小约为 10G 的测试过程原始数据。如果进行一个并发量为 10 万、并发用户测试时间为 24 小时的测试，那么所产生的测试过程中的原始数据将达到 10T。

其示意图如图 7 所示。

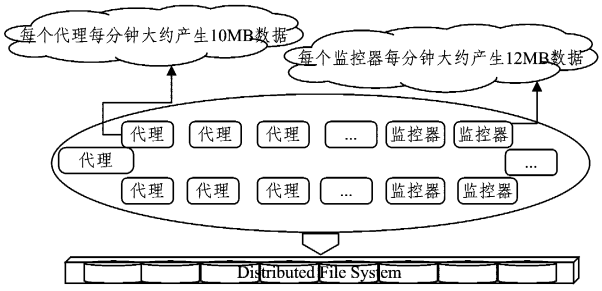

图 7　测试过程海量数据存储

针对如此海量的测试过程原始数据，我们采用了分布式文件系统的方式对其进行存储。当然为了方便数据的下一步使用，在文件的命名上采用了基于负载注入代理所执行的“当次测试的 UUID-时间戳”的命名方式对测试过程原始数据进行组织存储。

2.4　性能监测模块

性能监测模块主要负责对待测系统的硬件资源以及逻辑资源（如 CPU 使用率、每个进程的占用情况、内存使用率、网络接口 IO、硬盘 IO 等）的监控，其目的主要是确定在峰值测试的过程中，确定待测系统的瓶颈是否出现在其资源配置方面（即是否资源不足导致系统的瓶颈），并辅助性能表现分析待测系统瓶颈的可能原因。

例如，如果出现 CPU 利用率低，但硬盘 IO 极高，则可以说明待测系统对硬盘的频繁读写可能对系统的瓶颈有一定的影响。

其监测效果图如图 8 所示（以 CPU 状态为例）。

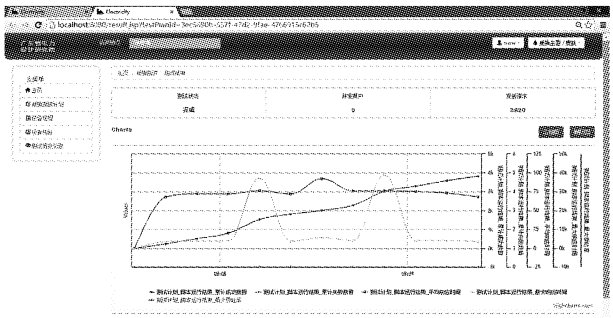

图 8　监测模块的监测图展示

在监测模块的运行过程中，也将会产生大量的监控原始数据，而这些原始数据对于后期的分析挖据又是十分重要的，故而对其依然采用分布式文件系统的方式进行存储。

2.5 性能分析模块

在当前的 Bench4Q 系统中，性能分析模块用于生成本次测试的测试报告，将在后期增加更多的测试数据的分析挖掘等增强功能。

性能分析模块所产生的测试报告如图 9 所示。

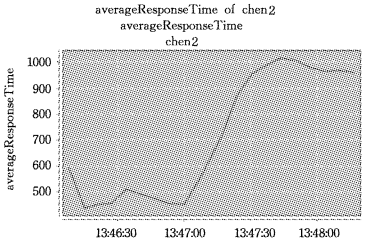

图 9 测试报告展示

2.6 待测系统

待测系统，顾名思义，即是性能测试的对象，以及性能监测模块的部署位置。

3 实验验证

本次实验验证中，使用的对比双方分别为单机的 Bench4Q 性能测试工具与云化后的 Bench4Q 性能测试服务平台。待测系统是部署在 Tomcat 服务器和 MySQL 数据库上的网上书店系统。图 10、图 11 分别显示使用传统的 Bench4Q 云化之前单机测试效果与云化之后大规模测试服务平台的测试效果。其中横轴代表每秒并发用户数（单位为个），左侧纵轴代表吞吐量 WIPS（Web Interactions Processed Per Second），右侧纵轴表示平均响应时间 WIRT（Web Interactions Response Time，单位为毫秒）。

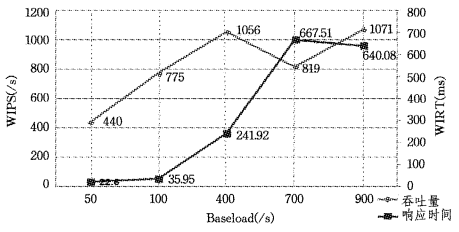

图 10 单机 Bench 工具测试效果

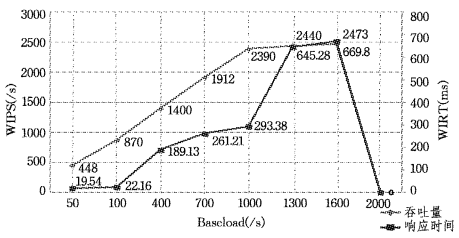

图 11 云化后 Bench4Q 平台测试效果

如图 10 所示，传统的 Bench4Q 工具使用单机执行测试任务，当并发虚拟用户数超过 500 之后，负载注入代理成为瓶颈，所产生的实际负载远低于用户期望值，故而会出现当负载超过 500 时，响应时间反而下降的反常情况，实际上虚拟用户在 Bench4Q 工具中进行了排队，使之成为了瓶颈。

如图 11 所示，云化后的 Bench4Q 性能测试服务平台很好地完成了测试任务，即使负载量超过了 2000，负载注入代理池也并没有出现瓶颈现象。这种测试的并发用户数更为真实可信。并且经过此次测试得出了待测系统的并发用户峰值大致在 1600～2000 之间（这是传统单机测试工具所达不到的峰值状态）。

结束语 大规模性能测试服务平台 Bench4Q 在运用云计算技术的同时，增强了对于负载注入代理池的控制和管理，加之以负载注入代理的插件化灵活扩展，使得大规模性能测试成为价格更为低廉、更为实惠的需求。使得测试用户得以在相对低廉的按需的测试投入下，得到更加真实的而非传统 1:20 的微缩仿真模拟，在这种更为真实的负载环境下，得以测试出正常的、峰值测试条件下的待测系统的性能表现。在云计算的潮流中，服务的理念越来越重要，所有的软件、硬件以及相关的系统工具都在尝试将自己作为一种服务提供，Bench4Q 就是将测试及服务理念逐渐实现的一个大规模性能测试服务平台。

参 考 文 献

[1] Reuters. SOASTA Leverages the Cloud to Test 1,000,000 Users on Myspace[OL]. http://cn. reuters. com/article/pressRelease/idUS121904%2B17-Nov-2009%2BMW20091117

[2] Zhang Wen-bo, Wang Sa, Wang Wei, et al. Bench4Q: A QoS-Oriented E-commerce Benchmark[C]//Proceeding of 35th Annual IEEE International Computer Software and Applications Conference (COMPSAC 2011). Munich, Germany, 2011: 38-47

[3] Wikipedia. HP LoadRunner[OL]. http://en. wikipedia. org/wiki/HP LoadRunner, 2012

[4] Zhang Q, Cheng L, Boutaba R. Cloud computing: State-of-the-art and research challenges[J]. Journal of Internet Services and Applications, 2010, 1(1): 7-18

[5] 林海略，韩燕波. 多租户应用的性能管理关键问题研究[J]. 计算机学报，2010，33(10)：1881-1895

[6] OW2. Bench4Q[OL]. http://forge. ow2. org/projects/jaspte, 2012

[7] Wu Lin-lin, Garg S K, Buyya R. SLA-based admission control for a Software-as-a-Service provider in Cloud computing environments[J]. Journal of Computer and System Sciences, 2012, 78(5): 1280-1299

[8] Popek G J, Goldberg R P. Formal Requirements for Virtualizable Third Generation Architectures[J]. Commun. ACM, 1974, 17: 412-421