基于多类顾客排队网络的 Exp-RAID 系统性能评价模型

谢广军[1] 刘军[2] 王刚[1] 刘晓光[1] 刘璟[1]

[1]（南开大学信息技术科学学院 天津 300071）

[2]（天津财经大学信息科学与技术系，天津 300222）

(xieguangjun1980@163.com)

摘要 本文针对 RAID 在线扩展系统这一典型的存储系统混合负载访问问题,采用多类顾客闭合排队网络(Multi-customer Closed Queueing Network MCQN) 对系统建立性能评价模型。我们推广了平均值分析(MVA)方法使之适应多类型负载的需求，并采用这一方法对模型进行了理论计算。通过将计算结果与实际系统性能的测试结果进行对比可以表明：本文建立的模型基本上反映了真实系统的性能变化趋势，通过对模型的分析，可以发现系统的瓶颈资源，预测系统某个部件性能的变化对整个系统的影响程度。

关键词 性能评价 MVA 法 RAID 扩展 闭合排队网络

中图法分类号 TP391

Performance Evaluation Model for Exp-RAID System Based on MCQN

Guangjun Xie[1], Jun Liu[2], Gang Wang[1], Xiaoguang Liu[1] and Jing Liu[1]

[1] (College of Information Technical Science, Nankai University, Tianjin 300071)

[2] (Department of Information Science and Technology, Tianjin University of Finance and Economics, Tianjin 300222)

Abstract we propose a Multi-customer Closed Queueing Model (MCQM) for RAID online extending and modify the Mean Value Analysis (MVA) to adapt to the requirements of multi-workload and make numerical computing using the modified MVA. Through experimental testing, numerical results are found to be basically in agreement with the testing results and reflect the dynamic trend of actual system performance. In addition, we find the bottleneck of system performance and predict the performance factors.

Keywords performance evaluation, MVA method, RAID expanding, closed queueing networks

1 引言

系统性能问题是当前存储领域设计和研究工作的重要内容，提高存储系统性能的前提是有效地对其性能进行评价和分析。目前学术界对存储系统性能方面的研究已经非常深入，总体来说，可以分为三类：测试法、仿真法和建模分析法。测试法的优点是可以得到系统的真实性能数据，缺点是只有在开发工作完成之后才能进行。仿真法可以在设计阶段先开发出系统的仿真软件，灵活性较高，但仿真软件的开发成本仍然较高。在系统设计阶段，为了预测和改进系统的性能，可以采用建模分析法，这种方法工作量小，不仅可以预测系统性能，还能反映出系统各个部分对整体性能的影响程度。通过建立系统的性能评价模型，

收稿日期：
基金项目：国家自然科学基金项目（编号：90612001），天津市科技发展计划重点项目（043185111-14），天津市高等学校科技发展基金（20061016），南开大学创新基金，及南开大学科学计算所支持

可以帮助设计人员有效地发现和消除性能瓶颈，具有成本和效率上的优势。所以，有关建模分析方法的研究已经成为存储系统性能研究领域的热点问题。

存储系统性能分析模型大体上包括两类，确定性模型和基于概率论的模型，确定性模型需要实现给出很多系统负载和运行特征参数，缺乏灵活性。而基于概率论的模型由于可以包含一些非确定的随机特征参数，近年来被广泛使用[1][2][3]，但这些研究工作大多数是在单一类型负载条件下进行建模，而当前的应用要求存储系统提供混合负载访问的能力，因此，有必要对多类型访问负载下的存储系统性能进行数学模型分析。磁盘阵列在线扩展系统是一种典型的混合负载应用，在 RAID 系统扩展进行过程中同时存在用于扩展 RAID 空间的数据迁移任务和上层模块下发的 I/O 处理任务。通过研究这种典型混合负载存储的系统性能评价方法，能够更加深入揭示存储系统中各个部件与整个系统性能之间的关系，有助于发现影响整个系统性能的瓶颈部件，通过对其进行改进可以获得系统性能的大幅度提升。

Exp-RAID[9]是我们之前工作中实现的基于 Linux 平台的软件 RAID 在线扩展系统，RAID 在线扩展应用是一种典型的多类型负载应用，本文采用多类顾客闭合排队网络对 Exp-RAID 的工作流程进行抽象，建立了其性能评价模型，并且采用多类顾客闭合网络的平均值分析（Mean Value Analysis , MVA）方法[4][5]计算系统性能的理论值，通过与实际系统性能测试实验得到的结果进行对比，表明我们的性能评价方法基本上反映了真实系统的性能变化特征。本文的工作为存储系统在多类型负载下的性能评价提供了一种新的方法。

2. Exp-RAID系统的结构

独立磁盘冗余阵列（Redundant Arrays of Independent Disks，RAID）技术已经成为当前海量存储系统中最重要的技术之一。然而，RAID 的容量毕竟是有限的，当 RAID 的存储空间不能满足上层应用的需求时，需要通过添加磁盘的方式扩展其存储空间。具体实现方式是：将新增磁盘加入到原 RAID 设备中，通过对原有数据进行重新组织，形成一个更大空间的 RAID 设备，这一过程称为 RAID 扩展。当前大多数的应用领域都要求服务器提供 7*24 的连续服务，对于 RAID 扩展也需要在线进行，这意味着在 RAID 扩展任务迁移数据的同时，RAID 系统仍然要处理上层用户请求。

Exp-RAID 系统是我们课题组开发的具有在线扩展能力的软件 RAID 系统，采用平凡扩展方法实现。所谓平凡方法是指扩展后的磁盘阵列保持原阵列的数据布局方式不变，同时移动位置发生变化的数据。如图 1 所示，扩展前为三块磁盘组成的左对称方式 RAID5，当添加磁盘 4 进行扩展后得到的是 4 块磁盘构成的左对称方式布局的 RAID5。

为了满足扩展后 RAID 设备的数据布局要求，Exp-RAID 采用一个内核线程[6]专门负责数据的迁移，具体来说，数据迁移线程的工作流程为：从原 RAID 布局的对应位置读取若干个条纹单元数据(Exp-RAID 中一般读取扩展前后 RAID 条纹长度的最小公倍数个数据单元作为一次数据迁移的粒度)，重新计算校验单元，然后按照新的布局方式写入各个磁盘。在扩展的同时 Exp-RAID 仍然需要处理上层模块的读写请求，Exp-RAID 对上层模块的读写请求的处理方式为：请求由 CPU 发出后，首先查找系统内存中是否存在该请求数据的副本，如果存在，则直接从内存中读取，否则，由 RAID 驱动程序对请求进行分解，将子请求下发到各个磁盘进行处理，直到所有子请求返回。

磁盘 1	磁盘 2	磁盘 3	磁盘 4
D_0	D_1	P_0	S
D_2	P_1	D_3	S
P_2	D_4	D_5	S
……			

图 1a. 扩展前布局

磁盘 1	磁盘 2	磁盘 3	磁盘 4
D_0	D_1	D_2	P_0
D_3	D_4	P_1	D_5
……			
S	S	S	S
……			

图 1b. 扩展后布局

3.Exp-RAID的闭合排队网络模型

从以上分析可以看出，Exp-RAID 中 I/O 请求和数据迁移两类负载同时存在，都需要系统进行处理。对这些负载提供服务的主要系统部件包括 CPU 节点，内存节点和磁盘节点三类，我们将这些部件抽象为排队网络中的服务节点，而将读写请求和数据迁移请求抽象为排队网络中的顾客。根据这些抽象和 Exp-RAID 对两类负载的处理流程，我们可以建立 Exp-RAID 的排队网络模型，如图 2 所示。模型中存在两类顾客，一类是用户的读写请求（以下称之为第

一类顾客），另一类是数据迁移任务（以下称之为第二类顾客），各节点对所有顾客的服务方式均为单队列 FCFS(First Come First Service)。 图 2 中实线表示 I/O 请求的数据流，虚线表示数据迁移线程的数据流。

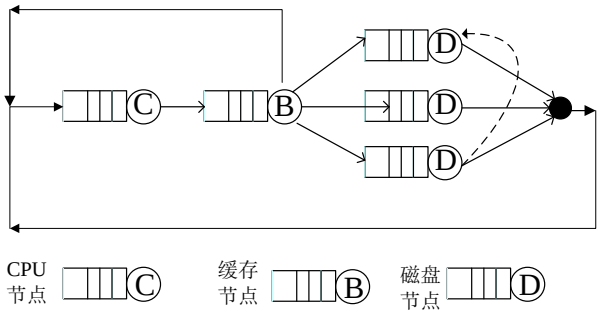

图 2. Exp-RAID 系统的闭合排队网络模型

在重负载条件下，对一个请求处理完成时，CPU 会马上生成一个同样大小的请求来对下一个存储单元地址进行读写，从模型的角度，我们可以认为任务并未返回和产生，而是循环于系统的各个服务节点之间。这样看来，两类任务在整个 RAID 扩展过程中总是保持不变的，所以我们建立的排队网络是闭合的[8]。

3.1 组件平均服务时间

我们用 $ST_i(b)$ 表示节点 i 处理大小为 b(KB)的数据时所花费的平均服务时间，下面我们给出各个节点对请求的平均服务时间的计算方式：

对于 CPU 节点，其调度策略是复杂的，一般假设服务时间服从指数分布。考虑实际系统中磁盘与内存间的 DMA 能力，我们认为 CPU 节点队每类顾客的平均服务时间与处理数据大小无关。

我们用 V_{RAM} 代表 RAM 的读写速率，则缓存节点的平均服务时间表示为：

$$ST_{buf}(b)=\frac{b}{V_{RAM}} \qquad \text{（公式 1）}$$

一次磁盘操作包括磁盘寻道时间，磁盘旋转时间和数据传输时间。而前两者统称为定位时间。本文为了简化模型的计算，忽略磁盘定位时间，用 V_{disk} 代表磁盘传输速率，则磁盘节点的平均服务时间表示为：

$$ST_{disk}(b)=\frac{b}{V_{disk}} \qquad \text{（公式 2）}$$

3.2 服务需求分析

服务需求描述了服务节点对顾客服务时间的数学期望，这一指标反映了系统组件对整体性能的影响程度，是对性能评价模型进行分析的基础指标。节点对某类顾客的服务需求等于顾客在该节点平均服务时间与顾客访问该节点概率的乘积，对于每一个服务节点 k, $V_{c,k}$ 代表第 c 类顾客对服务节点 k 的访问概率。$S_{c,k}$ 代表服务节点 k 对 c 类顾客的平均服务时间，则节点 k 对 c 类顾客的服务需求为

$$D_{c,k}=V_{c,k}*S_{c,k} \qquad \text{（公式 3）}$$

本文接下来将分析两类任务在不同节点中的服务需求，为了使问题简化，本文只讨论大数据读写（读写请求数据远远大于条纹长度）请求的情况。在本文给出的公式中约定，带上标*的符号表示第二类顾客。

如前所述，CPU 节点对各类顾客的平均服务时间均为定值（与服务数据量无关）所以，CPU 节点对两类顾客的服务需求为服务需求表示为：

$$D_{CPU}=CPU\_Delay \qquad \text{（公式 4a）}$$

$$D^{*}_{CPU}=CPU\_Delay^{*} \qquad \text{（公式 4b）}$$

缓存节点对第一类顾客，访问概率为 1，所以其服务需求为处理一个读写请求的时间。即：

$$D_{buf}=ST_{buf}(S_{req}) \qquad \text{（公式 5a）}$$

而第二类顾客在缓存节点要进行一次读出操作和一次写入操作，数据大小均为一次迁移数据大小 $S_{migrage\_unit}$，所以其服务需求为：

$$D^{*}_{buf}=ST_{buf}(2*S_{migrate\_unit}) \qquad \text{(公式 5b)}$$

在 RAID 扩展过程中，对大数据请求，如果查找缓存失败，则需要分发到组成 RAID 的磁盘中。而这一分发方式受当前数据迁移地址的影响。当请求地址为未迁移区域，则按照原 RAID 映射函数进行分发，否则，将采用迁移后的 RAID 映射方式进行分发。我们假设请求是均匀分布的，则在扩展开始阶段，几乎所有的请求都采用原 RAID 的映射函数，随着完成扩展区域的增大，采用新 RAID 映射的概率逐渐增大。很明显，整个扩展过程中，那么显然两种映射方式的概率各为 50%。所以，假设采用原阵列（m 个磁盘）映射每个磁盘的数据大小为 $S_{sub\_req1}=S_{sub\_req}/m$，通过添加 n 块磁盘进行 RAID 在线扩展，而采用扩展后的映射函数的每个磁盘分得的请求数据大小为 $S_{sub\_req2}=S_{sub\_req}/(m+n)$，则概率意义下第一类请求的服

务需求为

$$D_{\text{disk}} = (1-P_{\text{hit}})ST_{\text{disk}}(\frac{S_{\text{sub\_req1}}+S_{\text{sub\_req2}}}{2}) \quad \text{(公式 6a)}$$

其中，P_{hit} 表示缓存命中率，本文采用文献[3]中的方式计算。

对第二类顾客，迁移一个单元的数据需要一次读磁盘过程和一次写磁盘过程，显然，读磁盘过程按照原 RAID 进行请求分解，而写磁盘过程则按照新 RAID 进行分解。由此造成每个磁盘上读写数据的大小也不相同。磁盘节点对第二类顾客服务需求表示为：

$$D^{*}_{\text{disk}} = ST_{\text{disk}}(\frac{S_{\text{migrate\_unit}}}{m}+\frac{S_{\text{migrate\_unit}}}{m+n}) \quad \text{(公式 6b)}$$

4. 性能评价模型的MVA分析方法

平均值分析(MVA)法是 Little 公式的简单应用，应用该算法我们只需考查服务特征的平均值而尽量避免直接处理稳定状态概率。通过 MVA 分析方法，我们可以通过递归的方式计算闭合排队网络在不同顾客数目下的吞吐量，各节点平均响应时间以及平均排队长度等指标，而且很容易推广到闭合排队网络中有多种不同行为特征顾客的情况。在本节中，我们给出了 Exp-RAID 性能评价模型的 MVA 分析算法，通过此算法，可以计算出 Exp-RAID 性能的理论值。在以下的公式中，用下标 i (i∈{*CPU，RAM，disk*})代表节点类型，若系统中存在 m 个第一类顾客，n 个第二类顾客，用 $R_i(m,n)$代表顾客在节点 i 的平均响应时间，$Q_i(m,n)$代表节点 i 的平均队列长度，$T_{system}(m,n)$代表系统对顾客的吞吐量。

在系统中存在 m 个第一类顾客，n 个第二类顾客的情况下，每类顾客在节点 i(i∈{*CPU，RAM，disk*}) 的平均响应时间 $R_i(m,n)$等于顾客平均排队时间和平均服务时间之和，可以用如下公式计算：

$$R_i(m,n) = D_i*(1+Q_i(m-1,n)) \quad \text{(公式 7a)}$$

$$R^{*}_i(m,n) = D^{*}_i*(1+Q_i(m,n-1)) \quad \text{(公式 7b)}$$

这一公式的第一项是节点服务需求，显然就是顾客在该节点的平均服务时间，第二项用除去当前顾客时节点的稳定对列长度来代表当前顾客到达时观测到的排队顾客数，再乘以服务需求也就是顾客的平均排队时间，这一特性称为到达定理。

第一类顾客的吞吐量可以表示为：

$$T_{system}(m,n) = \begin{cases} 0 & if\ m=0 \\ \dfrac{m}{\sum_i R_i(m,n)} & \text{else} \end{cases} \quad \text{（公式 8a）}$$

第二类顾客的吞吐量表示为：

$$T^{*}_{system}(m,n) = \begin{cases} 0 & if\ n=0 \\ \dfrac{n}{\sum_i R_i(m,n)} & \text{else} \end{cases} \quad \text{（公式 8b）}$$

各节点的排队长度可以通过公式 9 进行计算：

$$Q_i(m,n) = T_{system}(m,n)*R_i(m,n) + T^{*}_{system}(m,n)*R^{*}_i(m,n) \quad \text{（公式 9）}$$

公式 7-9 构成了顾客数的递归关系，经过一轮递归，顾客数据将减少一个，递归结束的条件为：

$$Q_i(0,0) = 0 \quad \text{（公式 10）}$$

设定一个初始 m、n 值，根据公式 7-10，经过若干次递归运算，可以计算出闭合排队网络模型的性能指标。

5. 理论计算与实验结果分析置乱算法描述

5.1 测试环境

本文测试采用的硬件环境包括一台挂有 4 块 SCSI 磁盘的 PC 机，具体的配置如下：CPU 采用 Intel 赛扬 1.8G，内存 256M PC266 ，磁盘采用 IBM SCSI ultra160，单磁盘容量为 36G，SCSI 适配器采用两块 PCI SCSI 适配卡。每个 SCSI 适配卡上面通过 SCSI 总线连接两个 SCSI 磁盘。操作系统采用标准的 Red hat Linux Enterprise AS 4.0，内核版本号为 2.6.9-5。我们的实验开始阶段创建一个三块磁盘的软件 RAID5（条纹单元大小为 64KB）,然后添加一块磁盘进行扩展，同时，启动 1-10 个 iozone 线程对 RAID 上建立的文件系统进行读写操作。。

5.2 测试与理论计算结果

在 Exp-RAID 系统中，由于第二类任务也就是数据迁移线程采用单线程进行拷贝，所以 n=1, 而 m 对应于实验中 iozone 进程数，计算其他参数见表 1，根据本文中讨论的 MVA 方法，可以计算出 Exp-RAID 吞吐量的理论值。

参数	参数值	参数	参数值
CPU _ Delay	8ms	CPU _ Delay*	3.5ms
V_{RAM}	82MB/S	P_{Hit}	0.21

V_{disk}	35MB/s	$S_{migrate\_unit}$	256KB

表 1. Exp-RAID 性能参数表

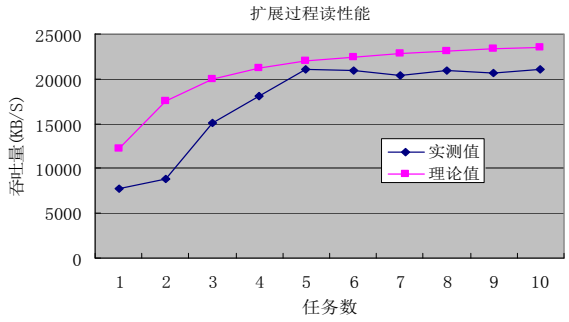

图 3. Exp-RAID 读性能理论值与实测值

图 3、图 4 给出了在扩展过程中进行读写操作的性能的理论计算值和实测值，读文件的大小为 1GB，请求大小为 512K。从上图可以看出，采用本文的模型对系统进行计算所得到的性能结果基本上反映了实际系统性能随任务数的变化的趋势。当然，我们的模型与实际测试结果相比存在着一定程度的误差，这是由磁盘定位时间，CPU 调度机制，以及两类任务相互冲突等多方面原因造成的。从图线中可以看出，随着读写请求线程的增多，系统吞吐量上升很明显，但当系统中的读任务达到 4-5 个时，系统明显达到了一种饱和状态，系统的吞吐量上升很慢，这说明系统中已经有部件处于满负荷工作（所有时间均处于对顾客的服务中）。

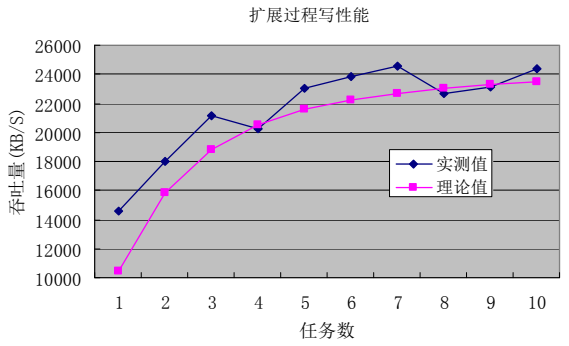

图 4. Exp-RAID 写性能理论值与实测值

通过这一模型，我们可以找到系统的瓶颈资源，很明显，首先达到满负荷工作的服务节点就是服务需求最大的节点。由此我们可以得出，磁盘的读写速度是影响 Exp-RAID 性能的主要因素。

6. 结论

多类负载存储系统是当前存储系统的主要趋势，由于负载行为的多样性，对其进行性能评价是复杂的。本文应用多类顾客闭合排队网络对 Exp-RAID 这种多类负载系统建立了性能评价模型，并且使用 MVA 方法对模型进行了理论计算。将计算结果与实际测试结果进行对比可以发现，这种性能评价方法对混合负载存储系统是有效的，同时，应用这一模型可以有效地发现瓶颈资源，对目标系统进行针对性改进。

当然，本文的工作还存在很多不足，需要今后加以改进。首先，我们在很多地方进行了近似，例如磁盘模型就忽略了磁盘定位过程，CPU 的 FCFS 排队假设和实际系统也并不完全相符，下一步需要将模型更加精确和细化。其次，本文只讨论了大数据(整条纹)读写的情况，RAID5 的小数据写与大数据写表现为完全不同的处理形式，今后工作中可以考虑小数据写的情况。最后，可以考虑将这一方法应用于更加复杂的存储系统，例如流媒体系统中就存在着流媒体应用和传统文件下载应用等多种类型的负载。

参 考 文 献

[1] S.Chen D.Towsley .A performance evaluation of RAID architectures. IEEE Transactions on Computers, 1996.45(10) ： 1116-1130

[2] E.Shriver,A.Merchant.An analytic behavior model for disk drivers with readahead caches and request reordering.Joint International Conference on Measurement and Modeling of Computer System.June 1998:182-191

[3] E.Varki,Arif Merchant,Jangzhang Xu,Xiaozhou Qiu.ssues and challenges in the performance analysis of real disk arrays.IEEE Transations on Parallel and Distributed System,June 2004 15(6):559-574

[4] Baojiang Cui, Jun Liu, Gang Wang, Jing Liu .Study on performance of IP-SWAN based on distributed NS-RAID.IEEE Computer Software and Applications Conference. 2004.. Proceedings of the 28th Annual International, 28-30 Sept. 2004 • vol.1:566 – 571

[5] 林闯.计算机网络与计算机系统的性能评价.北京.：清华大学出版社，2001 年 4 月

[6] 毛德操，胡希明，Linux 内核源代码情景分析.浙江：浙江大学出版社，2001 年 9 月

[7] Competitive Parallel Disk Prefetching and Buffer Management，Journal of Algorithms，2000，36：152-181

[8] Raif O. Onvural， Survey of Closed Queueing Networks with Blocking，ACM Computing Surveys，1990，Vol. 22, No. 2：83-121

[9] 谢广军，磁盘阵列在线扩展问题研究，天津:南开大学硕士论文，2006 年 6 月

谢广军，男，1980 年生，博士研究生，研究方向：并行与分布式系统，网络存储。

刘军，男，1963 年生，副教授，研究领域：存储系统，系统性能评价等。

王刚，男，1974 年生，副教授，研究领域：网络存储，并行计算，容错编码理论等。

刘晓光，男，1974 年生，副教授，研究方向：网络存储、并行计算等。

刘璟 ，男，1942 年生，教授，博士生导师，研究领域：并行与分布式系统，网络存储，容错编码，算法分析等。