网络计算环境分布式 COW 盘构建方法

谭怀亮，罗政，贺再红，李仁发
（湖南大学 信息科学与工程学院，湖南 长沙 410082）

摘 要：针对大规模网络计算环境的分布式计算和数据集中存储特点，结合 COW(copy-on-write)磁盘记录改写块的稀疏和突发特性，提出了一种分布式 COW 网络盘体系结构，将服务器处理所有客户主机相应 COW 盘的聚合开销分摊到各个客户主机自身，以加速网络计算环境系统构建速度；设计了一种改进的 64bit 位图压缩算法以有效减少 COW 盘位图文件大小，节省服务器磁盘空间和降低分布式 COW 盘网络传输开销；提出一种适合 COW 盘改写块突发特性的预取算法，以提高分布式 COW 盘相应 Cache 的命中率。实验结果表明基于改进的 COW 位图压缩和预取算法实现的分布式 COW 盘降低了多客户主机网络计算环境构建延迟。

关键词：网络计算；分布式 COW 盘；位图压缩；预取
中图分类号：TP393.04 **文献标识码**：A **文章编号**：1000-436X(2012)07-0036-08

Building method of distributed COW disk on network computing environment

TAN Huai-liang, LUO Zheng, HE Zai-hong, LI Ren-fa
(School of Information Science and Engineering, Hunan University, Changsha 410082, China)

Abstract: Aiming to distributed computing and data centralizing storage in large-scale network computing environments and combining sparse and bursting characteristic of COW(copy-on-write) disk storing modified data blocks, the distributed COW network disk architecture was presented. The system building process of network computing environments was speeded up by apportioning the aggregation spending that server manages all related COW disks of client hosts to each client host. A improving 64bit bitmap compress arithmetic was designed to reduce the size of COW disk's bitmap, so server disk space was saved and the network communication spending of distributed COW disk was less. The modified prefetching arithmetic based on COW bursting blocks was presented to improve the hit rate of Cache relating with distributed COW disk. Experiments on the prototype show that the distributed COW mechanism can effectively reduce system building latency of multi-client hosts network computing environment based on improving COW bitmap compress and prefetching arithmetic.

Key words: network computing; distributed COW disk; bitmap compress; prefetch

收稿日期：2011-06-03；**修回日期**：2012-03-05
基金项目：国家重点基础研究发展计划(“973”计划)基金资助项目(2007CB310900)；国家自然科学基金资助项目(60803130)；湖南省科技计划基金资助项目（2010GK3055）；教育部博士点基金资助项目（200805321029）
Foundation Items: The National Basic Research Program of China(973 Program)(2007CB310900); The National Natural Science Foundation of China(60803130)； The Science and Technology Plan of Hunan Province (2010GK3055)；Ph.D. Programs Foundation of the Ministry of Education of China (200805321029)

1　引言

网络计算模式[1]把计算任务分布到各客户主机（以下简称主机），而数据（系统数据和用户数据）集中存储于网络服务器中，实现了存储与计算的分离[2]，部署在服务器中的系统数据将可以供网络中所有主机共享。但每个主机有自身的私有属性数据（如各种硬件的型号和 ID 识别号、网络 IP 地址和用户登录参数等），通过共享的系统数据和该主机的私有数据可以组合一个可启动的系统映像，从而构建其计算环境。在极端情况下，多个主机计算环境在同时或很短的时间间隔内构建会导致服务器和网络出现严重的性能瓶颈，影响网络计算用户的应用体验。

传统的在服务器上为每个分布式主机分别映射各自的系统磁盘，并在其上安装所有系统软件与应用软件的部署方法存在服务器磁盘空间浪费、网络传输与服务器计算资源开销高等局限性。由于组织内计算环境的相似性，利用基于写时复制 COW（copy-on-write）机制的块共享技术，将存储在服务器上的可供网络中所有主机共享的系统数据部署为只读块设备，而将每个主机对系统数据的改写操作分布到对应的 COW 盘。在 COW 模式下，最初生成的共享系统数据盘称为源盘，通常用于安装操作系统和最常用的共享软件，每个主机对应有一个 COW 盘以记录其私有属性，主机可见的网络磁盘本质上是由服务器上的源盘与其对应的 COW 盘重构而成。将每个主机的网络磁盘重构过程卸载到其自身，以降低服务器集中负载，对加速主机的网络引导具有重要意义。

2　相关工作

20 世纪 90 年代以来，许多计算模式被提出用于解决网络计算中硬件与软件的松耦合绑定、软件升级、网络计算性能加速和系统维护等问题，例如 network computer[3]，NetPC[4]，thin clients[5]和无盘工作站[6]。文献[7]提出的网络计算模式中，主机本地没有硬盘，用户可以根据需要去选择加载某个操作系统和应用程序，然后利用主机的本地资源(CPU、内存及 I/O 设备)执行计算任务。文献[8]提出的 SonD 网络计算部署架构将服务器映像以网络磁盘的形式和物理服务器进行动态绑定。网络计算模式将分布式计算和集中存储管理结合起来，并将操作系统和应用程序作为一种资源存储到服务器上，通过 COW 或类似技术来提供不同主机共享的系统数据和私有数据。文献[9~11]提出的 COW 改进与优化技术广泛应用于虚拟机系统映像在低速网络链路上的迁移，文献[12，13]在文件和块层设计了存储数据多个版本快照的 COW 方法。这些文献中提出的 COW 文件或磁盘统一由某一主机或服务器集中处理，容易成为系统性能的瓶颈。

文献[14]提出的字对齐混合位图压缩(WAH)通过将位图按 31bit 固定比特数分组，并在分组的最高比特位添加 1bit 0 或 1 来区分文传字和填充字，对填充字合并来压缩位图空间，文献[15，16]在 WAH 基础上进行改进，进一步提升位图压缩效率。目前的位图压缩方法广泛应用于数据库索引，COW 盘通过位图来标识对应的数据块是否为改写块，该位图具有自身的一些特征，可以在已有的位图压缩基础上，基于现代 64bit 处理器的高速运算来进一步优化 COW 盘位图压缩与存储，降低其在低速网络上的传输开销。

文献[17]提出的顺序预取算法在磁盘 Cache 中广泛采用，当请求地址不连续时，预取失效。文献[18]分析请求地址块之间的间隔 P 来实行预取，比顺序预取更具灵活性，但预取准确率也与 P 变化有关，如果 P 不稳定，则预取成功率较低。文献[19]基于历史访问规律，建立马尔科夫概率转移矩阵，从而决定下一个预取的块，避免了预取的盲目性，但预测的准确率决定了算法的效率，特别是当预取状态较多时，时间复杂度和空间复杂度将急剧增加。

基于以上的相关研究和作者前期工作对网络计算模式下远程磁盘引导和多播加速的研究[20,21]，本文提出网络计算环境分布式 COW 盘体系结构及 64bit 位图压缩和预取算法。

3　分布式 COW 网络盘体系结构

3.1　传统 COW 盘结构与开销

每个 COW 磁盘通常由块位图、对共享只读盘改写块的重定位块组成，重定位块构成了 COW 盘的有效数据块，通过位图的某些比特（bit）置 1 来表示在对应的块地址处有改写块。每个主机由于有自身的私有属性数据需改写只读的共享源盘，而产生对应的 COW 盘来保存改写块。为了快速定位网络 I/O 命令的数据块，服务器通常事先将每个主机

对应的 COW 盘的块位图映射到服务器内存中以避免位图查询中频繁的磁盘操作。由于所有主机的网络 I/O 命令集中汇聚到服务器进行排队、位图查询和磁盘操作，传统 COW 盘结构与服务器集中解析网络 I/O 命令来定位数据块会带来较大的开销：1）时间开销，设 n 个主机以均等机会向服务器产生 I/O 命令，并在同一队列排队等候，查询 COW 位图一次的时间开销为 t，则定位 I/O 命令数据块需等待的时间开销 T 可表示为函数 $T=tn$，因此定位一个块的时间开销正比于主机数量；2）服务器内存开销，设每个主机对应的 COW 盘的块位图大小为 s，n 台主机位图映射的内存开销 M 可表示为函数 $M=s\times n$，因此位图映射的内存开销将随着主机数量的增加而线性增长。

3.2　网络计算环境主机 COW 盘改写块统计特性

通过作者前期对 3 台同构主机分别从共享系统卷（SSV）引导与配置过程生成的 COW 盘数据块的统计分析[22]可知

$$\frac{|X_i|}{|SSV|}\approx 6\%\ (i=1,2,3) \tag{1}$$

$$X=\{x_1,x_1+1,x_1+2,\cdots,x_n,x_n+1,\cdots,x_n+k,\cdots\} \tag{2}$$

式（1）说明主机计算环境构建过程对 SSV 的改写非常小，绝大部分扩展块将不会被修改，即 COW 盘有效数据非常稀疏。式（2）序列 X 说明 COW 盘的改写块具有突发特性，即要么对源盘没有改写，要么会改写连续的多个块，改写块的起始地址称为突发地址，连续的块数称为突发长度。

3.3　分布式 COW 网络盘体系结构

如图 1 所示的分布式 COW 网络盘体系结构，主要组件包含运行在每个主机的 I/O 命令与块地址解析器和客户端 COW_client 预取器、运行在服务器端的 COW 服务进程 COW_server_process、以及存储在每个主机物理内存中的 Cache 和服务器上所有的 COW 磁盘文件。COW_server_process 在侦听到每个主机的连接后创建与其编号 i 对应的线程 COW_thread(i)，该线程专门处理相应主机的源盘读请求和 COW 磁盘文件读写请求；每个主机的 COW_client 一方面与服务器的 COW_server_process 进程建立连接，构建主机网络 I/O 命令或响应的传输路径，另一方面将 I/O 命令指定的数据块读取到主机 Cache 中，并根据局部性原理预取相邻块到 Cache 中；主机物理内存中的 Cache 有 2 类，一类为缓存源盘的读数据块，另一类为缓存 COW 盘的读或写数据块。由这 2 类 Cache 和 COW_client 预取器构成了主机的虚拟系统磁盘。

为了节省服务器磁盘空间和降低网络传输开销，服务器上的每个 COW 盘中的改写块标记是以压缩方式保存的稀疏位图文件，在主机引导初始化时，由引导程序（如 PXE 或 iSCSI Boot BIOS）提前将该主机对应的 COW 盘位图文件读取并解压到主机物理内存中，后续对虚拟系统磁盘的读写将通过 I/O 命令与块地址解析器检测稀疏位图（已位于该主机物理内存中）来决定该 I/O 命令是定向到源盘还是 COW 盘，有如下 4 种情况出现。

1) 如果是定向到源盘的读，则首先检查源盘 Cache 中是否能命中，如未命中，COW_client 被调用以完成后续的源盘网络 I/O 读取，并预取相邻块到源盘 Cache 中。

2) 如果是定向到 COW 盘的读，通过检查 COW 盘 Cache 是否能命中，如未命中，COW_client 被调用以完成后续的 COW 盘网络 I/O 读取，并预取相邻块到 COW 盘 Cache 中。

3) 如果是定向到源盘的写，则需将对源盘的写

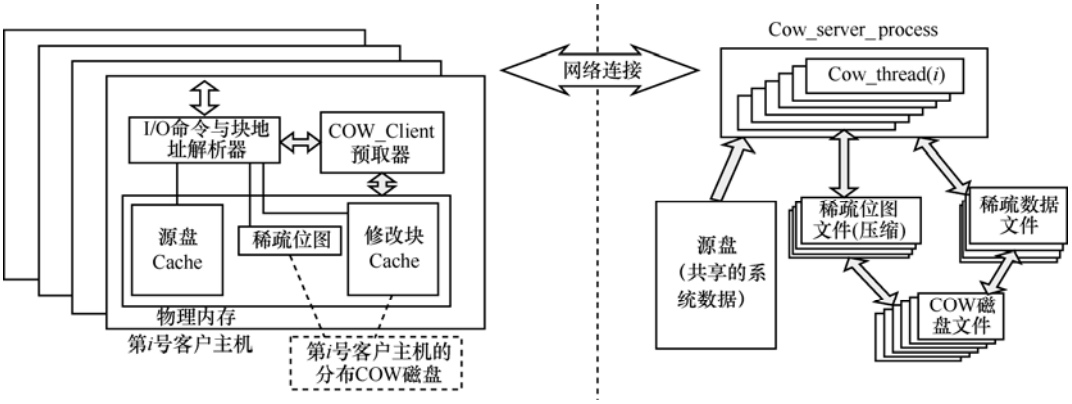


图 1　分布式 COW 网络盘体系结构

变换成 COW 盘的写。即先将写数据缓存到 COW 盘 Cache 中，并将稀疏位图中与改写块地址对应的位图位置位（即由 0 变为 1），在网络空闲或 Cache 置换时调用 COW_client 以完成后续的 COW 盘网络 I/O 合并写。

4) 如果是定向到 COW 盘的写，则先将写数据缓存到 COW 盘 Cache 中，在网络空闲或 Cache 置换时调用 COW_client 以完成后续的 COW 盘网络 I/O 合并写。

因此，分布在每个主机的稀疏位图、改写块 Cache 和预取器构成了该主机的分布式 COW 盘。稀疏位图的压缩与传输、I/O 块预取机制是主机计算环境快速构建的关键。

4　COW 网络盘位图压缩

4.1　64bit 稀疏位图压缩算法

结合 COW 盘改写特性，对字对齐混合位图压缩（WAH, word aligned hybrid）算法进行改进，利用现代处理器高速 64bit 运算指令，来提高 COW 盘的改写块位图时空性能。

如图 2 所示，设 COW 盘位图由 280bit 组成，压缩算法由以下 4 步组成。

1) 未压缩位图被分成若干等大小的组，组的比特数等于处理器体系结构字长减 1bit，即 63bit，最后一组不够 63bit 长时通过附加 35bit 0 以达到 63bit 大小，得到 5 组 63bit 长的组。

2) 标记相邻的全 0 或全 1 组为候选的组合并，图中，第 2、3 组是组合并的候选组，因为该组完全由相同的 0 比特组成。

3) 附加 1bit 作为组的最重要比特 MSB（most significant bit），附加比特设置为 1 表示一组全相同比特。MSB 以 1 开头的这些 63bit 长的字组称为填充字。填充字有 2 种类型，即 0 填充字和 1 填充字，通过第 2 个 MSB 来区分。合并的候选组变换成填充字。字中的最后 50bit LSB（least significant bit）作为计数器来记录每个填充字包含的合并组数。图中第 2 组变换为一个 0 填充字，其合并的组数为 2；附加的 MSB 设置为 0 表示不全相同比特的组，即为 0 和 1 的混合组。MSB 以 0 开头的 64bit 长的字称为文传字。

4) 根据 COW 盘突发特性，在 0 填充字后紧随的文传字一般有连续的置 1bit。计算出突发比特的起始位置和长度，并放置到在前的填充字内。如表 1 所示，使用类型比特和计数器比特之间的前 6bit 记录突发位置、后 6bit 记录突发长度；后 50bit 作计数器，前 2bit 表示字类型。图 2 中，第 2、3 填充字合并后，跟随其后的是具有突发改写位的文传字（第 4 组），突发改写比特起始于第 55bit，突发长度为 3bit，它们被附在前面的填充字内，该文传字从位图中移除。

表 1　　填充字结构

填充字	类型	突发改写比特起始比特	改写比特比特数	合并的填充字计数器
1	0 或 1	6bit	6bit	50bit

4.2　算法性能分析

1) 压缩空间

在均匀分布位图内，有一个改写比特（即置 1 的比特）的概率独立于位的位置。这样位图能够通过比特密度 d（置 1 的比特数/整个比特数）来刻画，

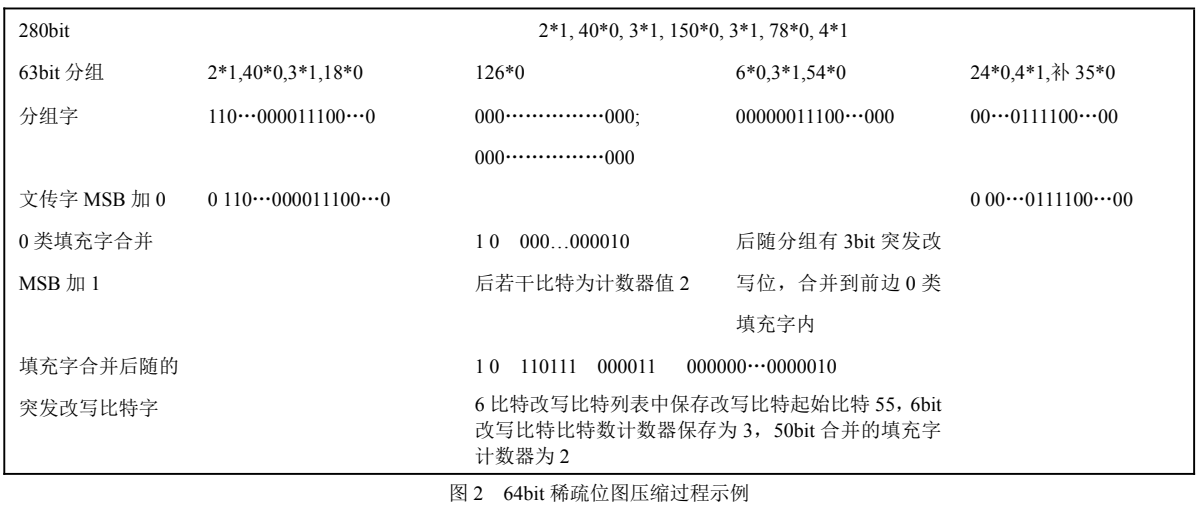


图 2　64bit 稀疏位图压缩过程示例

q 为未改写比特（置 0 的比特）的比特密度，则 $q=1-d$ 。根据文献[14]可得出，压缩位图的字数 $W=M-G$，式中 M 是未压缩前的整个分组数，G 为能被合并的计数组数（2 个连续的相邻字构成一对计数组），其期望值 $\overline{G}=(M-1)P_{\text{merge}}$，$M-1$ 表示合并前所有可能的计数组，P_{merge} 是计数组合并的概率。则压缩后整个字数的期望值：

$$\begin{aligned}\overline{W}&=M-\overline{G}=M-(M-1)P_{\text{merge}}\\&=M-(M-1)[P_u(0,w-1,d)\sum_{k=0}^{s}P_u(k,w-1,d)+\\&\quad P_u(w-1,w-1,d)\sum_{k=0}^{s}P_u(w-1-k,w-1,d)]\\&=M-(M-1)[q^{w-1}\sum_{k=0}^{s}C_k^{w-1}d^kq^{w-1-k}+\\&\quad d^{w-1}\sum_{k=0}^{s}C_k^{w-1}d^{w-1-k}q^k]\end{aligned}$$

由于 COW 盘中改写块数相对于整个系统磁盘块数所占的比例非常小，即上式中 COW 盘位图的位密度 d 非常小，M 值很大，通过二项式分解，上式近似为

$$\begin{aligned}\overline{W}&\approx M[1-q^{2(w-1)}-(w-1)dq^{2w-3}]\\&\approx M(w-1)[2d-d(1-(2w-3)d)]\\&\approx Ld=h/l\end{aligned}$$

其中，L 为未压缩位图的整个比特数，即为 $L=M(w-1)$；h 为设置为改写比特的比特数，l 为改写块的平均突发长度。

由上式可以看出，压缩后的 COW 盘位图文件大小直接正比于改写比特数，反比于平均突发长度。COW 盘位图文件的字基本上是由大量的 0 类型填充字、内含改写起始比特与比特数的 0 类型填充字和少量文传字组成。因此压缩后的 COW 盘位图文件将占用更低的磁盘空间，且分布式传输到主机花费的网络开销非常小。

2) 时间开销

与传统的集中式 COW 盘管理与位图索引方法比较，分布式 COW 盘的时间开销主要由以下几部分组成：压缩后的 COW 稀疏位图文件网络传输开销、解压、主机位图查询时间，消除了集中在服务器上的排队等待时间。相对于排队等待时间，压缩后的 COW 稀疏位图文件的网络传输、解压开销非常低，因此，分布式 COW 释放了部分服务器资源，特别是 Cache 和队列资源、I/O 资源，降低了 COW 盘位图的查询开销。

5 COW 网络盘预取机制

在网络计算环境中，构成 COW 盘的改写块地址具有突发特性，即在宏观上是离散和跳转的，但有局部连续特性，其集合表示如下：

$$\cdots\bigcup\{B_i,B_{i+1},B_{i+2}\}\bigcup\{B_k,B_{k+1},B_{k+2},B_{K+3}\}\cdots$$
$$\bigcup\{B_m\}\cdots\bigcup\{B_n,B_{n+1},\cdots,B_{n+L}\}\bigcup\cdots\bigcup\{B_p\}\cdots$$

其中，子集 $\{B_n,B_{n+1},\cdots,B_{n+L}\}$ 的数据块是顺序块，子集 $\{B_m\}$、$\{B_p\}$ 等为孤立块。子集亦可用（突发地址 a_i，突发长度 l_i）对来表示。为了提高预取的准确率并确保算法的简洁高效，设计了基于突发长度与采样计数结合的顺序与跳转预取算法。

1) 预取块控制器 PBC，其组成如下：上一次传输的扩展块首地址 a 和块数 m、后续传输的预期扩展块首地址 b 和块数 n、扩展块命中计数器 j、扩展块块数计数器 k、已完成的 I/O 请求采样计数 p、采样阈值 s、确定算法是只采样计数还是进行预取的控制字 y。在初始化时，s 为默认值 0x100，y 为 TRUE，表示初始时只采样计数，其他成员皆为 0。

2) 预取条件：采样计数器 p 记录的 I/O 次数达到阈值 s，且统计的命中计数器达到某一程度，y 将被设置为 FALSE，实际的预取将发生。当预取效率降低到某一程度，y 又被设置为 TRUE。阈值 s 可根据实际测试来选择配置。

3) 预取地址与预取量：基于访问局部性原理，并通过检测稀疏位图来定位后续可能预取的扩展块是定位到源盘还是 COW 盘，如果是源盘，则预取与上次被访问或预取的块最相邻的且未在主端源盘 Cache 缓存的块作为下次预取的起始块，预取量为上次 I/O 操作的块数；如果是 COW 盘，则从突发地址 a_i 处预取突发长度 l_i 的改写块到 COW 盘 Cache 中。

4) 预取算法流程如下：设当前 I/O 请求的扩展块首地址 c 和扩展块块数 o。

① 更新 pbc 指向的全局预取控制器，如果 c 等于 a 或 b，j 递增；如果 o 等于 n 或 m，k 递增；采样计数器 p 递增。

② 当 p 达到阈值 s，如果 j 计数器达到 s 的 50%，k 计数器达到 s 的 25%，使能预取，否则只采样计

数；此处可调整 s；同时，为下次采样，p、j 和 k 计数器清零。

③ 确定下一次 I/O 期待预取的扩展块首地址 pref_f 和扩展块块数 pref_c。检测稀疏位图来定位预取的扩展块是定位到源盘还是 COW 盘，如果是定位到源盘，则顺序预取，即 pref_f=c+o (当 $c \geqslant a$) 或者 pref_f=c−(o+1)，pref_c=o；否则如果是定位到 COW 盘，则跳转预取，即 pref_f= a_i ，pref_c= l_i 。

④ 更新下次预期传输的扩展块首地址和块数，即 b=pref_f，n=o。

⑤ 如果预取控制字 y 为 FALSE，进行真正预取，即构造 I/O 请求分组，由 COW_client 发送到服务器读取源盘或相应 COW 盘中的数据块，在主机响应分组的中断服务例程中将数据块 DMA 到源盘 Cache 或 COW 盘 Cache 中，并改写 Cache 描述符状态。

⑥ 保存该次 I/O 的首块地址和块数到 PBC 中，即 m=o，a=c。

6　实验

6.1　实现与实验平台

基于分布式 COW 盘构建方法，在 Linux 存储服务器上开发了分布式 COW 网络盘块设备驱动及其管理程序以支持 COW 盘位图压缩和 COW 盘有效数据块预取；在 IP-SAN 扩展 BIOS[21]中加入 COW 压缩位图文件解压和物理内存分配；在操作系统保护模式网络虚拟磁盘驱动程序加入 I/O 命令与块地址解析器、客户端程序 COW_client 预取器及其 Cache 管理模块。在吉比特每秒网络和百兆比特每秒网络分别搭建了网络计算平台，并进行了实验。

1) 服务器：Intel 64 双核至强 2.0GHz，2/8GB 内存/2 块双端口 Intel 吉比特每秒网卡，利用 SSD 固态硬盘作为网络计算环境多主机共享的系统数据源盘（分区为 80GB，扩展块大小为 4KB），并开启共享数据多播[20]，COW 盘及其相应的压缩位图文件部署在 320GB 的 SATA HDD 硬盘上，操作系统为 Linux 64 2.6.32。

2) 主机：Intel 双核 1.8GHz/2GB 内存/带 IP-SAN 扩展 BIOS 的 8 169 吉比特每秒网卡或 8 139 比特每秒网卡，无本地磁盘。

3) 吉比特每秒/百兆比特每秒自适应交换机连接。

4) 实验系统：Windows 7 和共享的应用软件。

6.2　主机构建计算环境延迟测试

网络计算系统中搭建主机计算环境前，须先从服务器的源盘和 COW 盘加载系统数据块到主机 Cache 中，重构其虚拟系统盘，并完成系统启动过程。图 3 中的 1 号线为使用 gPXE 从 iSCSI 服务器启动 Windows 7 所花费的时间，2 号线为启用分布式 COW 和源盘共享数据多播启动 Windows 7 所花费的时间。

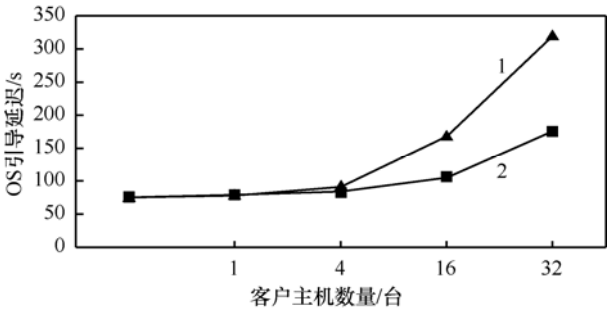


图 3　多主机的操作系统启动时间

由图 3 看出，使用 gPXE 启动方式构建网络计算环境时，32 台主机近似同时开启的引导延迟为 320s，引入分布式 COW 方法后，启动延迟降为 176s。主机的 I/O 响应时间比传统方法降低了约 45%，且启动加速的优势随着主机数量的增加更加明显。

6.3　COW 盘位图分布与压缩的性能测试

为了验证 COW 盘位图分布和压缩的有效性，分别测试了以下 3 类数据： 1) COW 位图的服务器内存开销；2)读取 COW 盘改写块的时间开销；3)稀疏位图的网络传输开销。所有的测试都是以 4KB 作为扩展块大小。

从图中的测试结果可知，所有主机的 COW 盘位图都在服务器内存中分配时，随着主机数量的增加，由于服务器确定每个主机的网络 I/O 命令是定向共享源盘还是 COW 盘前，都需要将相应的 COW 盘位图装入服务器物理内存，并将来自所有主机的 I/O 命令统一排队，COW 盘位图消耗的服务器内存和读取 COW 盘改写块的 I/O 命令响应时间皆随主机数量呈线性增长（图 4、图 5 的 1 号线）。如图 4、图 5 的 2 号线所示，通过构建分布式 COW 网络盘，将 COW 盘位图分配和 I/O 命令定向检测等负载卸载到各个主机，并配合预取方法将源盘和 COW 盘有效数据块取到主机相应 Cache 中，显著降低服务器资源消耗，节省大量的时间开销，宏观上提升了网络计算环境系统构建速度。

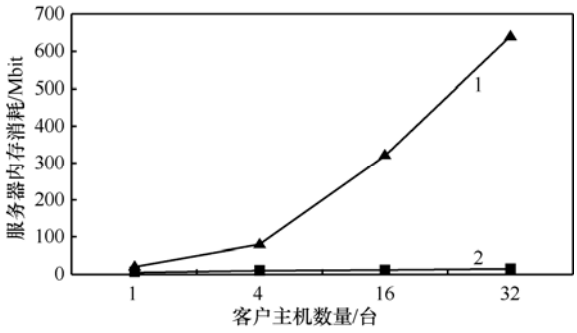


图 4　COW 位图的服务器内存开销

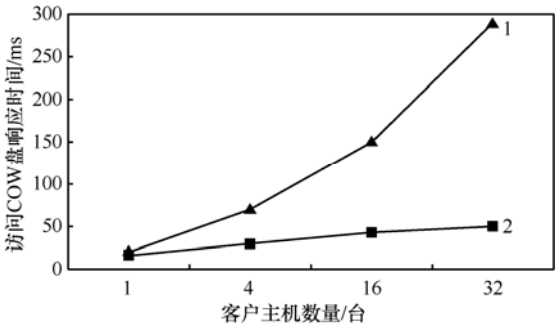


图 5　访问 COW 盘改写块的时间开销

而对于分布式 COW 盘位图压缩的有效性测试，由表 2 可以看出：一方面由于 COW 盘改写块所占的比例非常低，在以 4KB 作为扩展块大小时为 6%，且大部分的改写块具有突发特性，即在某个地址处有改写，则后续相邻块也会被改写，压缩后需传输的位图文件大小只有未压缩前的 10%左右，基本上与改写块数量成比例，证明了算法的有效性；另一方面与吉比特每秒网络对比，分布式 COW 盘位图压缩在百兆比特每秒网络上所带来的优势更加明显，主要由于多个主机的 COW 盘位图文件同时在百兆网络上传输，如果未压缩，将消耗可观的网络带宽，而压缩后的网络带宽占用显著减少。对于位图的压缩与解压开销，由于现代 64bit 处理器的操作运算相对于网络传输速度和 I/O 处理速度快多个数量级，因此 COW 盘位图压缩和解压所产生的额外开销可以忽略不计。

表 2　分布式 COW 盘位图压缩与未压缩对比

类型	COW 盘位图文件大小/Mbit	百兆比特每秒网络传输延迟/ms	吉比特每秒网络传输延迟/ms
未压缩	20	260	35
压缩	2.1	28	11

6.4　COW 盘有效数据块预取命中率测试

为了测试预取算法的有效性，分别测试了 COW 盘有效数据块在 2 种预取方法下的主机 Cache 命中率：1)只顺序预取；2) 基于突发长度与采样计数结合的顺序与跳转预取。

测试结果表明，第 1 种方法命中率较低为 67%，第 2 种方法由于预取具有一定的参照标准，避免了不必要的预取操作，主机 Cache 的命中率（达到了 90%）显著提高，并且算法简洁高效，额外负荷完全可以忽略。

7　结束语

本文提出的分布式 COW 网络盘体系结构、改进的 64bit 位图压缩和预取算法，分摊了服务器对 COW 盘数据块 I/O 处理的聚合负载，降低了服务器内存和网络的传输开销，实验结果表明多客户主机在构建网络计算环境时，启动延迟和 I/O 响应时间大幅度降低，且启动与运行加速的优势随着客户主机数量的增加更加明显。

参考文献：

[1] SAHA D, MUKHERJEE A. Pervasive computing:a paradigm for the 21st century [J]. IEEE Computer, 2003,36 (3):25-31.

[2] 马一力,傅湘林,韩晓明.存储与计算的分离[J].计算机研究与发展,2005,42(3):520-530.
MA Y L, FU X L, HAN X M. The separation between storage and computation[J]. Journal of Computer Research and Development, 2005, 42(3):520-530.

[3] HENTWICH R G, KAPPNER T. Network computers-ubiquitous computing or dumb multimedia[A]. The 3rd Int' l Symp on Autonomous Decent Ralized Systems[C]. Berlin,1997.

[4] COMERFORD R. The battle for the desktop[J]. IEEE Spectrum,1997,34(5):20-28.

[5] RICHARDSON T, STAFFORD F Q, WOOD K R. Virtual network computing[J]. IEEE International Computing,1998,2(1):33-38.

[6] INTEL INC. Preboot execution environment (PXE) specification[EB/OL]. http://developer.intel.com/ial/wfm/wfmspecs.htm,1998.

[7] ZHANG Y X.Transparence computing: concept, architecture and implementation[J]. Chinese Journal of Electronics,2004,32(12A): 169-174.

[8] YIN Y, LIU Z J, TANG H Y. SonD: a fast service deployment system based on IP SAN[A]. IPDPS[C]. Miami,FL, 2008. 1-10.

[9] SAPUNTZAKIS C, CHANDRA R, PFAFF B. Optimizing the migration of virtual computers[A]. Proc of the 5th Symp on Operating Systems Design and Implementation[C]. Boston,2002. 377-390.

[10] BELLARD F. QEMU, a fast and portable dynamic translator[A]. Proc of the USENIX Annual Technical Conf(USENIX 2005)[C]. Berkeley: USENIX Association,2005. 41-46.

[11] 陈彬, 肖侬, 蔡志平. 基于优化 COW 虚拟块设备的虚拟机按需部署机制[J]. 计算机学报, 2009, 32(10):1915-1926.
CHEN B, XIAO N, CAI Z P. On-demand deployment of virtual machines based on optimized COW virtual block device[J]. Chinese Journal of Computers, 2009,32(10):1915-1926.

[12] ZACHARY P, RANDAL B. Ext3cow: a time-shifting file system for regulatory compliance[J]. ACM Transactions on Storage, 2005,1(2): 190-212.

[13] MONTERIRO J, DALLE O.CORRAL: stackable copy-on-write versioning device using Linux device-mapper[A]. USENIX Annual Technical Conference (USENIX'08)[C]. Poster, 2008.

[14] WU K, OTOO E J, SHOSHANI A. Optimizing bitmap indices with efficient compression[J]. ACM Transactions on Database System, 2006, 31 (1):1-38.

[15] MICHAL S, ROBERT W. RLH: bitmap compression technique based on run-length and Huffman encoding[J]. Information Systems,2009, 34(4,5): 400-414.

[16] ALESSANDRO C,ROBERTO D P. Concise: compressed 'n' composable integer set[J]. Information Processing Letters, 2010, 110(16): 644-650.

[17] SMITH A J. Cache memories[J]. ACM Computer,1982,14(3):473-530.

[18] JOHN W C F,JANAK H P. Data prefetching in multiprocessor vector cache memories[A]. Proceedings of the 18th Annual International Symposium on Computer Architecture[C]. Toronto,1991. 54-63.

[19] 谢学军,叶以正,邱善勤.基于马尔可夫模型的数据值预取方案[J]. 电子学报, 2007, 35(2):307-310.
XIE X J,YE Y Z,QIU S Q. Data value prefetching method based on markov model[J]. Chinese Journal of Electronics, 2007,35(2):307-310.

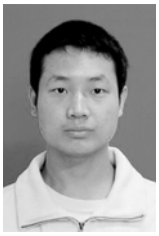
[20] 谭怀亮, 朱存望, 张镇平. iSCSI 网络计算模式下的可靠多播策略[J]. 计算机研究与发展,2011, 48(9): 216-223.
TAN H L, ZHU C W, ZHANG Z P. The reliable multicast method on iSCSI network computer[J]. Journal of Computer Research and Development, 2011, 48(9): 216-223.

[21] 谭怀亮, 尹斌. 支持 IP-SAN 远程映射与引导的固件协议栈[J]. 通信学报, 2009, 30(10): 58-67.
TAN H L,YIN B. Firmware protocol stack supporting remote boot and storage volumes mapping in IP-SAN[J]. Journal on Communications, 2009,30(10):58-67.

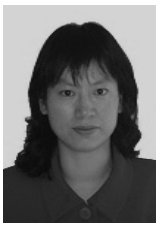
[22] 谭怀亮,王燕,孙建华.分布式系统卷重构过程的改写块预取方法[J]. 湖南大学学报, 2009,36(1):77-80.
TAN H L,WANG Y,SUN J H. Prefetch method for rewrite block in the rebuilding process of distributed system volumes[J]. Journal on Hunan University, 2009, 36(1):77-80.

作者简介：

谭怀亮（1969-），男，湖南双峰人，湖南大学副教授，主要研究方向为网络存储和嵌入式系统及应用。



罗政（1988-），男，湖南浏阳人，湖南大学硕士生，主要研究方向为网络存储和嵌入式系统及应用。



贺再红（1972-），女，湖南娄底人，湖南大学讲师，主要研究方向为计算机网络。

李仁发（1957-），男，湖南郴州人，湖南大学教授、博士生导师，主要研究方向为嵌入式计算、无线网络、网络与数字媒体。