文章编号:1001 - 9081(2006)02 - 0364 - 04

移动计算环境中基于广播的数据缓存策略

邵雄凯,何 瑜
(湖北工业大学 计算机学院, 湖北 武汉 430068)
(heyu_99@21cn.com)

摘 要:在研究了数据广播频率的变化对缓存策略的影响之后,提出以移动客户机(Mobile Client,MC)从广播中读取数据的平均响应时间为标准,来决定是否缓存该数据项。其优点是可以避免每次访问这些数据项等待较长时间,同时也可降低移动客户机缓存管理的代价,提高其工作性能。

关键词:数据缓存; 广播; 移动数据库
中图分类号: TP311.13 **文献标识码**:A

New cache strategy based on broadcasting in mobile computing environments

SHAO Xiong-Kai, HE Yu
(*College of Computer Science and Technology, Hubei University of Technology, Wuhan Hubei* 430068, *China*)

Abstract: The cache strategy should not only concern the historical accessing situations of the data, but also consider the broadcasting probability of the data. After carefully researching the influence of the broadcasting frequency changes to the cache strategies, a method to determine whether the data should be replicated or not was proposed. This method is based on the average respond time of the Mobile Client (MC) spending in reading data from the broadcast. This new cache strategy can make MC avoid waiting for a long time while accessing some data, and cut down the cost of cache management. Therefore, the MC's performance will be advanced obviously.

Key words: data cache; broadcasting; mobile database

0 引言

移动计算环境为移动客户机(MC)随时随地访问数据提供了基础,它的应用前景已越来越广阔。与基于固定网络的传统分布式计算环境相比较,移动计算环境有许多不同之处,如:移动性、频繁断接性、网络条件的多样性和非对称性等。由于这些特性的存在,使得传统的数据库管理与访问技术在移动数据库中不能适用或不能完全适用,因此必须针对移动计算的特点对这些技术展开进一步研究,复制与缓存技术就是其一。

这个方面的研究有很多,如基于文件系统的缓存策略的文献[1~3]:

1) 文献[1]中提出的 Coda 文件系统,主要采用用户指定的方法来进行数据缓存。用户先通过脚本语言指定感兴趣的文件,然后由缓存管理器指导这些文件和 LRU 算法得到缓存数据集,将这些数据集结合起来就构成了断连前的本地缓存。但是该方法需要用户指导缓存过程。

2) Seer 系统采用的自动预测缓存数据[2],是基于寻找用户过去访问的文件之间的语义关系来确定应缓存的数据。文件被按照语义距离的度量而捆绑在一起,而语义距离的度量说明了它们的密切程度。

3) 文献[3]提出了通过缓存树半自动进行数据缓存处理,缓存树通过参考过去文件的使用情况来建立。通过检测应用程序和数据文件之间的调用关系(即过去文件的使用情况),该算法把一个项目中相关的文件组织成一棵或多棵树,然后由用户通过图形化工具选择需要缓存的树。

这三种缓存策略都不太适用于数据库系统,主要有如下一些原因[4]:

1) 在文件系统中,收集的数据项的大小单位是文件,使得数据收集在文件系统中比数据库系统要简单:与之对应的是关系(关系 DB)和类扩展(面向对象 DB)。而对于数据库系统,关系和类扩展这样的粒度并不是足够的好。一般而言,收集的单位越小(如元组/对象)则越有意义且性能更好。

2) 在数据库中,依据过去的关系推断将来依赖关系比在文件系统中要困难。这是因为在文件系统中,文件具有同构的组织,发现它们是否相关联是比较容易的。而且同一目录下的文件关系比其他目录下的文件关系更密切。

3) 一个文件系统的用户通常仅需要有限的文件数量(整个文件系统的一小部分)就可完成其任务;而数据库的用户查询一个问题可能需要涉及大部分的数据库。

文献[4~7]在移动计算环境下也提出了一些缓存策略,但它们都有一定的局限性。

文献[4]提出了一种预测分析策略,通过分析客户机过去查询的存取数据的模型,总结这些片段的大小,作为缓存的依据。它的具体实现方法是将关系数据库划分为一些垂直和水平片段。垂直片段的划分不是按照属性的亲近关系,而是按照存取历史,即哪些属性一起存取,哪些不是一起存取。把经常一起存取的属性作为缓存单元,依照历史,水平片断和垂直片断都赋于不同的优先级,最后只有水平片断和垂直片断优先级均较高的且相交的片段才被缓存。

文献[5]提出了一种低开销自动数据收集算法,其基本思想是:根据用户的数据访问过程同步建立数据对象之间的

收稿日期:2005 - 08 - 03

作者简介:邵雄凯(1963 -),男,湖北武汉人,副教授,主要研究方向:分布式和移动式数据库; 何瑜(1980 -),女,湖北武汉人,硕士研究生,主要研究方向:分布式和移动式数据库.

逻辑关系,称之为关联度,把整个数据对象集合按照关联度分割成为若干个组,每个组内的数据对象具有比较紧密的逻辑关系。这样,当其中某一个数据对象被访问后,其他数据对象有很大可能在同一个作业期内也被访问。

文献[6]提出一种滑动窗口的数据复制策略:滑动窗口中保存 K 个读/写请求,如果 K 个请求中,读请求的数量大于写请求的数量,则移动客户机就缓存该数据;如果写请求的数量大于读请求的数量,则移动客户机就不缓存该数据。它是按照通信开销大小来实现的。

文献[8]提出一种 LIX 缓存替换算法,它是在 LRU 算法的基础上,通过加入对象广播频率因素修改而成。虽然它考虑了对象广播频率因素,但它有如下一些假定限制:假定广播模式固定(每个周期广播的数据项和每个数据项的广播频率是固定不变的)和假定数据信息固定不变。

总的来说,数据缓存应考虑下列三个方面的问题:缓存数据的单位、缓存数据的内容和缓存数据的时间。

不同的应用可能有不同的数据收集单位,这主要取决于数据模型。例如:在文件系统中,缓存单位应该是文件;在关系数据库系统中,可能是元组,一组元组或者全部关系;在面向对象的数据库系统中,可能是对象,一组对象或者全部类。其中最重要的问题是收集哪些数据项,哪些数据项将被使用,而这个方面很大程度上取决于系统的应用。

对预期的数据有明确和模糊两种处理方法:

第一种方法要求用户明确指定要收集的数据项(对缓存的描述说明);

第二种方法是模糊的包含关于数据项的信息,通过过去操作的历史(数据存取的历史)来预测将要缓存的数据。

可以看出文献[1~7]的研究主要集中在前两个方面,对于缓存数据的时间以及在广播的方式下移动客户机应该采用何种缓存策略,缓存什么样的数据,这方面的研究还比较少,因此本文就这方面的问题作以下讨论。我们首先建立一个广播模型,在该模型下讨论广播频率对缓存策略的影响。

1　广播模型

1.1　广播模型

我们建立的广播模型如下:固定广播尺寸的大小,广播周期为 T,采用$(1,m)$ 索引方式,即在一次广播中 m 次播放索引段。参数如下:

1) 每个广播周期尺寸的大小固定为 $Data$ 个桶;

2) n 表示每个桶的容量,即每个桶中所包含的元组数量;

3) k 表示索引树的层次(深度);

4) $Index$ 表示索引树中桶的数量;

5) C 表示每个簇的桶数;

6) $(\frac{Data}{C})$ 表示广播中的簇数。

当索引树是满平衡树时,树的深度(层次):

$$k = \lceil \log n^{(\frac{Data}{C})} \rceil$$

$$Index = \sum_{i=0}^{k-1} n^i \text{(索引树中桶的数量)}$$

由文献[7]知,MC 从广播周期中获取数据项的响应时间 qt 为:

$$qt = \frac{1}{2} * [(m+1) * Index + (\frac{1}{m} + 1) * Data] + C$$

1.2　广播中数据的组织

广播中的数据包含三个部份:

1) 数据的索引部份;

2) 被更新的数据项,上一个周期内被更新的数据部分,必定要在当前周期中广播,并标志上更新标志;

3) 被 MC 请求的数据项,当被请求数据项的数量大于广播周期所能容纳的大小时,就按优先级的高低来组织,优先级高的先广播,优先级低的被推迟广播。

1.3　数据项广播优先级的确定

某一数据项被 MC 请求的次数越多,说明它的受欢迎程度越高,因此它的优先级就越高;被请求的次数越少,它的优先级就越低。这样安排优先级,能够及时满足大多数 MC 的请求。但这种安排优先级的方法有可能导致被请求次数少的数据项长期不能被广播,使某些 MC 长期处于等待状态。因此,这种情况是不能允许的。解决的办法如下:

在固定服务器上,为每一个被请求的数据项 D_i 设置一个请求计数器 $qc[i]$,当 D_i 被请求一次时,$qc[i]$ 就加 1,当 D_i 被广播后,$qc[i]$ 就清零;如果 D_i 未被广播,则每过一个周期 $qc[i]$ 就加上一个常数 q,直到 D_i 被广播为止,这样被请求次数少的 D_i,随着时间的增长,优先级就越来越高,被广播的可能性也就越大。

常数 q 可以这样来确定:假设每个被请求的 D_i 至少在 j 个广播周期内就要被广播一次,则:

$$q = \frac{\max\{qc[1], qc[2], \cdots, qc[n]\} - qc[i]}{j}$$

2　缓存策略

虽然文献[3~6]中的几种缓存策略适用于数据库系统,但并不适用于广播环境下的缓存方式。这是因为广播环境下的缓存策略不仅要考虑数据的历史存取情况,还要考虑数据项在广播中出现的频率。

一方面,如果 MC 能够把需要的数据全部缓存到本地,那么 MC 就不需要从广播中读取数据,即请求数据的平均响应时间为最小,MC 的工作效率就应该最高。但是,MC 本地缓存的数据量越大,维护缓存的一致性所引起的开销也就越大,可能导致 MC 的性能反而下降。

另一方面,由于服务器采用广播方式传送数据,因此通信开销基本固定,不是主要问题。移动主机从广播中读取数据的平均响应时间对 MC 的性能有很大影响,而某项数据被广播的频率又直接影响到对它的响应时间。

2.1　缓存策略的基本思想

基于上述两方面的考虑,提出一种缓存策略使得 MC 的工作性能得到进一步提高。

MC 读取数据的方法是首先在本地缓存中查找需要的数据项 D_i,如果在本地缓存中查找不到 D_i,MC 就向服务器发出对 D_i 的请求,并准备在后续的广播周期中接收数据。而 MC 对数据的修改必须要提交给服务器执行。在每个广播周期,MC 都要接收广播中的索引段,依照索引段中的修改标志,查找 MC 本地缓存是否有被更新的数据项。如果有,则接收该数据并更新缓存中的项。

在 MC 中设置一个缓存表,该表的每一个表项包含四个项目:数据项的名称 D_i;该数据项的读次数 Nr_i;更新次数 Nu_i;广播频率 P_i。

定义 1　读/写次数采用文献[6]中的滑动窗口法,设滑

动窗口的大小为 k,即读和写的请求次数之和不大于 k。

当 $Nr_i + Nu_i < k$ 时:

1) 如果每读一次 D_i 时,Nr_i 加 1

2) 如果 D_i 每被更新一次时,Nu_i 加 1

当 $Nr_i + Nu_i = k$ 时:

1) 如果每读一次 D_i 时,Nu_i 减 1,Nr_i 加 1

2) 如果 D_i 每被更新一次时,Nr_i 减 1,Nu_i 加 1

文献[6]的缓存基本思想是:当 $Nr_i \geqslant Nu_i$ 时,就缓存 D_i;当 $Nr_i < Nu_i$ 时,就放弃对 D_i 的缓存。该方法考虑了通信开销的大小,但没有考虑数据对象在广播中的广播频率。我们在考虑数据的读写请求次数的基础上,还考虑了数据的广播频率因素,以数据的平均响应时间作为是否缓存该数据项的衡量标准。

本文下面推导在广播方式时,缓存 D_i 的条件。

定义 2 数据项 D_i 的广播频率 P_i 是指 D_i 在 $1/P_i$ 个广播周期内才被广播一次。

定义 3 如果 MC 不缓存 D_i,则 Nr 次的数据请求响应时间 T_{r1} 为:

$$T_{r1} = \left\{\frac{1}{2} * [(m+1) * Index + (\frac{1}{m} + 1) * Data] + C + \left(\frac{1}{P_i} - 1\right)T\right\} * Nr \quad (1)$$

$$T = Data + m * Index \quad (2)$$

由于没有缓存 D_i,故对维持 D_i 一致所需要花费的时间 $Tu_1 = 0$

定义 4 如果 MC 中缓存 D_i,则 MC 对 D_i 的 Nr 次请求响应时间 $Tr_2 = 0$,MC 维护 D_i 的一致性需要花费的时间 T_{u2} 为:

$$T_{u2} = \left\{\frac{1}{2} * [(m+1) * Index + (\frac{1}{m} + 1) * Data] + C\right\} * Nu \quad (3)$$

规则 1 如果数据 D_i 的 Nr 次数据请求响应时间 Tr_1 大于或等于 MC 维护其所需花费时间 Tu_2 时,应缓存 D_i;即当 $Tr_1 \geqslant Tu_2$ 时,应缓存 D_i

规则 2 反之如果数据 D_i 的 Nr 次数据请求响应时间 Tr_1 小于 MC 维护其所需花费时间 Tu_2 时,可不缓存 D_i;即当 $Tr_1 < Tu_2$ 时,不缓存 D_i

2.2 Nr、Nu 和 P_i 的确定

Nr,Nu 的统计非常简单,开销也小,现在的问题是如何确定数据项 D_i 的 P_i。P_i 为数据项 D_i 在每次广播中可能出现的概率,在我们假定的广播模型中应有 $0 < P_i \leqslant 1$。一个看似简单的方法是:统计总共的广播次数和每个数据项被广播的次数,然后就可以动态计算 P_i。该方法看似简单,其实是很难实现,原因在于:1) 总的广播次数随着时间的延长会越来越大;2) MC 必须对每个周期中的每个数据项都要收听,这对 MC 来说是难以想象的浪费。所以,我们提出一种近似计算 P_i 的方法。

定义 5 设 qT_i 为 MC 对 D_i 的请求时间,RT_i 为 MC 接收到 D_i 的时间,因此 MC 对 D_i 的请求响应时间 $T_i = RT_i - qT_i$,设广播周期为 T,则该数据项在一个广播中被广播的概率 P_i:

$$P_i = \begin{cases} 1, & \text{当 } T_i \leqslant T \text{ 时} \\ 1 / \left\lceil \frac{T_i}{T} \right\rceil, & \text{当 } T_i > T \text{ 时} \end{cases}$$

因为当 $T_i \leqslant T$ 时,说明 MC 对 D_i 的请求是在一个广播周期中就能被满足,$P_i = 1$;当 $T_i > T$ 时,说明 D_i 在 MC 请求之后的一个周期内未被广播。例如,若 $T_i = 3.2T$,说明 D_i 在被请求后的第四个广播周期才被广播,因此它的广播频率为 1/4。需要注意的是,每当对 D_i 的请求被响应后,应重新计算一次 P_i。

至此,我们就可以由 Nr、Nu 和 P_i 的大小比较,就可以确定是该缓存 D_i 还是放弃缓存 D_i。

3 性能评估及总结

表 1 模拟广播系统参数设置

参数	值	参数	值
一次广播索引片段数 m	5	索引树的深度 k	3
广播尺寸 $Data$ 个桶	128	每个簇的桶数 C	2
每个桶包含 n 个元组	4	索引树中桶的数量 $Index$	21

将定义 3 中的(2)式 $T = Data + m * Index$ 分别代入到(1)、(3)式中,可得:

$$Tr_1 = \left\{\left[\frac{1}{2}\left(1 + \frac{1}{m}\right) + \left(\frac{1}{P_i} - 1\right)\right] * T + C\right\} * Nr \quad (4)$$

$$Tu_2 = \left[\frac{1}{2}\left(1 + \frac{1}{m}\right) * T + C\right] * Nu \quad (5)$$

故可以得到:

$$\frac{Tr_1}{Tu_2} = \frac{Nr}{Nu} * \left[1 + \frac{\left(\frac{1}{P_i} - 1\right) * T}{\frac{1}{2}\left(\frac{1}{m} + 1\right) * T + C}\right] \quad (6)$$

令 $Tr_1 = Tu_2$,即 $\frac{Tr_1}{Tu_2} = 1$,显然在不同的广播环境中 Nr、Nu、P_i 可以取不同的值;所以假定在该广播环境下 $Nr = r0$,$Nu = u0$,$P_i = P0$ 且 $r0 + u0 \leqslant k0$ 时,满足 $\frac{Tr_1}{Tu_2} = 1$,则:

1) 设 $P_i = P0$ 不变,若 $N'_r = r0 + n, n > 0$,则 $N'_u = \begin{cases} u0, & \text{当 } r0 + u0 + n < k0 \text{ 时} \\ u0 - n, & \text{当 } r0 + u0 = k0 \text{ 时} \end{cases}$;那么 $\frac{N'_r}{N'_u} > \frac{r0}{u0}$ 则 $\frac{T'_{r1}}{T'_{u2}} > 1$,此时应缓存数据项 D_i;反之若 $N'_r = r0 - n$,则 $\frac{T'_{r1}}{T'_{u2}} < 1$,那么不缓存数据项 D_i;

2) 设 $Nr = r0$,$Nu = u0$ 不变,若 $P'_i > P0$ 则 $\frac{T'_{r1}}{T'_{u2}} < 1$,那么不缓存数据项 D_i;反之则应缓存数据项 D_i。

表 2 Nr/Nu 固定不变时随着 P_i 变化 A1 的缓存情况

P_i 的值	Tr_1/Tu_2 的值	是否缓存 A1 (Y/N)
1 月 10 日	12.63	Y
3 月 10 日	3.87	Y
5 月 10 日	2.11	Y
7 月 10 日	1.36	Y
9 月 10 日	0.94	N
1	0.8	N

表 3 P_i 固定不变时随 Nr/Nu 的变化 A1 的缓存情况

Nr	Nu	Nr/Nu 的值	Tr_1/Tu_2 的值	是否缓存 A1 (Y/N)
2	5	2/5 <1	0.47 <1	N
3	5	3/5 <1	0.70 <1	N
4	5	4/5 <1	0.94 <1	N
5	5	1	1.18 >1	Y
6	4	3/2 >1	1.77 >1	Y
7	3	7/3 >1	2.75 >1	Y
8	2	4 >1	4.72 >1	Y

例:MC 对数据项 1 的读写请求情况和广播频率的假设如下,设 $Nr + Nu \leqslant k = 10$,当 $Nr/Nu = 4/5$ 时,根据设定的广播参数和(6) 式可得表 2。

当 $P_i = 9/10$ 时,根据设定的广播参数和(6) 式可得表 3。将 $P_i = 7/10$ 和 $P_i = 1$ 的情况做比较,得到表 4 和图 1。

表 4　$P_i = 7/10$ 和 $P_i = 1$ 时 $A1$ 的缓存情况比较

Nr	Nu	Nr/Nu 的值	$P_i = 7/10$		$P_i = 1$	
			Tr_1/Tu_2 的值	是否缓存 $D1$(Y/N)	Tr_1/Tu_2 的值	是否缓存 $D1$ (Y/N)
2	5	2/5 <1	0.68 <1	N	2/5 <1	N
3	5	3/5 <1	1.02 >1	Y	3/5 <1	N
4	5	4/5 <1	1.36 >1	Y	4/5 <1	N
5	5	1	1.70 >1	Y	1	Y
6	4	3/2 >1	2.55 >1	Y	3/2 >1	Y
7	3	7/3 >1	3.97 >1	Y	7/3 >1	Y
8	2	4 >1	6.81 >1	Y	4 >1	Y

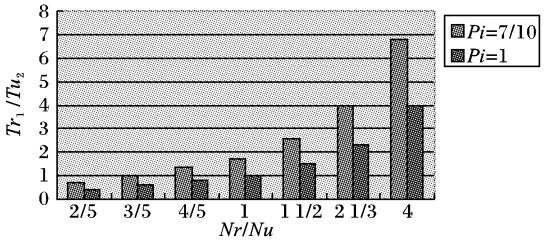

图 1　Tr_1/Tu_2 – Nr/Nu 之间的关系

由表 2 可看出,当 Nr/Nu 固定时,P_i 越小则 Tr_1/Tu_2 就越大,即 Tr_1 越大于 Tu_2,就越应该缓存数据 Di。

由表 3 可看出,当 P_i 固定时,Nr/Nu 越大则 Tr_1/Tu_2 就越大,即 Tr_1 越大于 Tu_2,就越应该缓存数据 Di。

因此,将 $P_i = 7/10$ 和 $Pi = 1$ 的情况做比较,可以看出:

1) 在 MC 同时请求访问的多个数据项中,哪个数据项的 Nr/Nu 越大,同时它广播的 P_i 越小,就应先缓存哪个。

2) 如果 MC 同时请求访问多个数据项,而这些数据项的 Nr/Nu 正好相等,那么哪个数据项的 P_i 越小就应先缓存哪个。所以根据广播的特点,移动主机并不像文献[6] 那样仅仅只缓存 $Nr \geqslant Nu$ 的数据项,而是缓存那些 Nr/Nu 较大且在广播中频率较低的数据对象。在多盘广播方式下,MC 缓存那些 Tr_1/Tu_2 较大的数据项,可以避免每次访问这些数据项等待较长时间,同时也可降低 MC 缓存管理的代价,提高其工作性能,因此这种缓存策略能够更好地适应广播环境。

参考文献:

[1] KISTLER JJ. Disconnected Operation in a Distributed File System [M]. Ph. D. Dissertation, Carnegie-Mellon University, May 1993.

[2] KUENNING G. The Design of the Seer Predictive Caching System [A]. In: Proceedings of the IEEE workshop on Mobile Systems and Applications[C]. Santa Cruz, CA, Dec. 1994. 37 – 43.

[3] TAIT C, LEI H, ACHARYA S, *et al.* Intelligent File Hoarding For Mobile Computers[A]. In: Proceedings of the 1st International Conference on Mobile Computing and Networking (MobiCom'95) [C]. Berkeley, CA, Nov. 1995. 119 – 125.

[4] PISSINOU N, DUNU C, MAKKI K. A New Framework for Handling Mobile Clients in a Client-Server Database System[J]. Computer Communications, 2000, 23(10): 936 – 941.

[5] 周桓. 移动环境中的 Cache 技术研究 [D]. 北京: 中国科学院软件研究所, 2001.

[6] HUANG YX, SISTLA P, WOLFSON O. Data Replication for Mobile Computers[A]. In: Proceedings of ACM SIGMOD Conference [C]. 1994. 13 – 24.

[7] IMIELINSKI T, VISWANATHAN S, BADRINATH BR. Data on Air: Organization and Access[J]. IEEE Transactions on Knowledge and Data Engineering, May/June 1997, 9(3): 353 – 371.

[8] ACHARYA S, ALONSO R, FRANKLIN M, *et al*. Broadcast Disks: Data Management for Asymmetric Communication Environments[A]. In: Proceedings of ACM SIGMOD Conference on Management of Data[C]. 1995. 199 – 210.

(上接第 359 页)

4　结语

本文分析了基于角色的访问控制的理论基础,设计并实现了一个基于角色的访问控制模型系统。该模型系统能很好地满足轻量级应用的需求,提高权限管理的效率。同时,该访问控制模型系统具有可重用性,能够在一定范围内应用于其他业务需求类似的信息管理系统中。在后续的工作中我们将就如何保证角色/权限信息库中 XML 文件的可靠性、完整性以及如何提高系统的运行性能和可复用性做进一步的研究。本文的工作为设计并实现基于角色的轻量级访问控制系统提供了实际工程经验,做出了有益探索。

参考文献:

[1] DAVID F. Ferraiolo and Richard Kuhn, Role-Based Access Control [A]. Proceedings of the 15th NIST-NSA National Computer Security Conference[C]. 1992.

[2] TSUDIK G. Access Control and Policy Enforcement in Internetworks [D]. US: University of Southern California. April, 1991.

[3] CHADWICK DW, OTENKO A, BALL E. Role-Based Access Control With X.509 Attribute Certificates[J]. Internet Computing, 2003, 7(2): 62 – 69.

[4] 张海峰, 赵凯, 陆徜. 轻量级认证与授权研究综述[J]. 计算机工程, 2003. 168.

[5] CHADWICK DW, OTENKO A. The PERMIS X.509 Role Based Privilege Management Infrastructure[J]. ACM, 2003, 12(2): 277 – 179.

[6] WANG CJ, WU JP, DUAN HX. Using Attribute Certificate to Design Role-Based Access Control, Parallel and Distributed Computing [J]. Applications and Technologies. IEEE, 2003. 217 – 218

[7] MCLAUGHBLIN B. Java and XML 数据绑定[M]. 李仁勇, 祁力, 译. 北京: 中国电力出版社, 2003.