



(12) 发明专利申请

(10) 申请公布号 CN 102681792 A

(43) 申请公布日 2012. 09. 19

(21) 申请号 201210110332. 6

(22) 申请日 2012. 04. 16

(71) 申请人 华中科技大学

地址 430074 湖北省武汉市洪山区珞喻路  
1037 号

(72) 发明人 冯丹 胡洋 刘景宁 童薇 江泓  
田磊 秦亦

(74) 专利代理机构 华中科技大学专利中心  
42201

代理人 李佑宏

(51) Int. Cl.

G06F 3/06 (2006. 01)

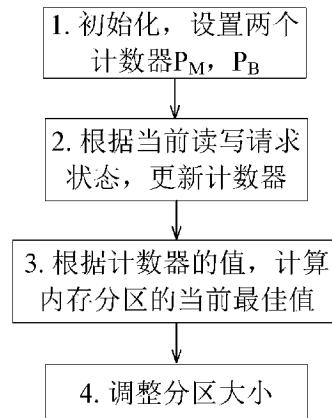
权利要求书 1 页 说明书 6 页 附图 3 页

(54) 发明名称

一种固态硬盘内存分区方法

(57) 摘要

本发明公开了一种基于闪存的固态硬盘的内存分区方法,包括(1)设置两个关键参数 $P_B$ 和 $P_M$ ,分别记录第一额外操作开销和第二额外操作开销;(2)根据不同情况下的读写请求,更新两个关键参数 $P_B$ 和 $P_M$ ; (3)经过一定周期,根据 $P_B$ 和 $P_M$ 的变化,使用一组计算公式,找到最佳的数据缓存区域和映射关系区域的大小比例;(4)根据当前内存的组成情况和第三步计算获得的数据缓存区域和映射关系区域的最佳大小比例,调整内存的组成。本发明的内存分区方法适用于基于闪存的固态硬盘中内存的管理,不受可扩展性的问题影响,对具有不同容量的内存的固态硬盘设备都适用,可以在不同负载和使用环境下,极大提高固态硬盘的读写性能。



1. 一种固态盘的内存分区方法,包括如下具体步骤:

(1) 建立两个计数器  $P_M$  和  $P_B$ , 分别用于记录第一额外操作开销和第二额外操作开销, 其中所述第一额外操作开销包括将映射关系读出过程所消耗的时间, 以及将映射关系写回闪存过程所消耗的时间; 所述第二额外操作开销指数据写回闪存过程消耗的时间;

(2) 根据读写请求的状态更新计数器  $P_M$  和  $P_B$ , 具体为:

对于读请求, 首先从内存中寻找该请求的映射关系, 如果该映射关系不在内存中, 首先将映射关系从闪存中读出, 同时查找内存中的映射关系区域是否还有空闲空间, 如果有, 则将所述读出的映射关系存放在内存的映射关系区域, 然后更新计数器  $P_M$ ; 如果没有, 首先将映射关系区域中旧的映射关系写回闪存, 然后将所述读出的映射关系保存到内存中, 最后更新计数器  $P_M$ ;

对于写请求, 首先在内存的数据缓存区中寻找该写请求数据, 如果该写请求数据的旧数据不在数据缓存区, 查看内存的数据缓存区是否还有空间, 如果有, 则将该写请求数据直接写到内存的数据缓存区; 如果没有, 则将数据缓存区中的旧数据写回闪存, 然后将该写请求的数据写到内存的数据缓存区, 最后更新计数器  $P_B$ ;

(3) 每响应一次读写请求或一定周期之后, 调整内存中映射关系区和数据缓存区的大小比例, 即计算当前状态下, 内存中映射关系区和数据缓存区的最佳大小比例;

(4) 根据上述获得的映射关系区和数据缓存区的最佳大小比例, 调整内存的组成, 即可实现固态盘内存实时分区。

2. 根据权利要求 1 所述的一种固态盘的内存分区方法, 其特征在于, 所述的步骤 (2) 中, 如果内存中的映射关系区域有空闲空间, 则使用下式更新计数器  $P_M$ :

$$P_{M(\text{新值})} = P_{M(\text{旧值})} + P_R \quad (1)$$

如果内存中的映射关系区域没有空闲空间, 则使用下式更新计数器  $P_M$ :

$$P_{M(\text{新值})} = P_{M(\text{旧值})} + P_R + P_W \quad (2)。$$

3. 根据权利要求 1 或 2 所述的一种固态盘的内存分区方法, 其特征在于, 所述步骤 (2) 中, 通过如下公式更新计数器  $P_B$ :

$$P_{B(\text{新值})} = P_{B(\text{旧值})} + P_W \quad (3)。$$

4. 根据权利要求 1-3 之一所述的一种固态盘的内存分区方法, 其特征在于, 所述步骤 (3) 中, 通过如下公式计算所述映射关系区域和数据缓存区域的最佳大小比例:

$$\frac{\text{映射关系区域}}{\text{当前最佳容量}} = \frac{P_M}{P_B + P_M} \times \text{内存总容量} \quad (4)$$

$$\frac{\text{数据缓存区域}}{\text{当前最佳容量}} = \frac{P_B}{P_B + P_M} \times \text{内存总容量} \quad (5)。$$

## 一种固态硬盘内存分区方法

### 技术领域

[0001] 本发明属于计算机外存储设备领域,具体涉及一种基于闪存的固态硬盘中内存的分区方法。

### 背景技术

[0002] 固态硬盘是近十年出现的一种新型计算机外存储设备。通常,固态硬盘分一下几种类型:基于闪存的固态硬盘、基于动态随机存储器的固态硬盘、基于其他固态介质的固态硬盘。在本发明专利中的固态硬盘特指基于闪存的固态硬盘,以下简称固态硬盘。闪存是一种具有高密度,大容量,非易失性等特点的新型存储介质,它相对磁介质,具有较低的读写延时、较低的能耗;但是闪存芯片自身存在的一些缺陷限制了这类存储器的应用。其一,必须先擦除后写入,擦除操作的单元为块,写入操作的单元为页,一个块中有多个页。其二,闪存的擦除次数是有限的。因为先擦后写的原因,闪存中需要一个映射表来记录逻辑地址与物理地址之间的关系(称之为映射关系)。固态硬盘中除了一个处理器、大量的闪存芯片外,还有一个或多个内存芯片。相对闪存,内存的读写速度较为快速。

[0003] 固态硬盘中的内存通常有两个作用:存放映射关系、做数据缓存区。映射关系需要被快速读写,所以必须存放在内存中。数据缓存区是用户数据在内存中存放的区域,它利用一段时间内,请求可能重复出现的特点,将重复出现的请求通过访问内存而直接被响应,减少访问闪存的次数。这提高了固态硬盘的性能,同时减少了闪存的写操作次数,降低了擦除次数,从而提高了固态硬盘的寿命。数据缓存区越大,请求直接访问内存的次数也就越多,所以固态硬盘的性能就会越好。通常,因为固态硬盘中,所有的映射关系总量较大,在内存中只能存放部分映射关系,其他映射关系存放在闪存中。当请求的数据的映射关系不在内存中时,需要先将映射关系读出,然后再读写数据;相反,如果映射关系在内存中,可以直接读写请求的数据。因此,越多的映射关系存放在内存中,意味着更好的固态硬盘性能。可以看到,内存存放映射关系区域和数据缓存区域对内存的有限空间,存在一个争用的问题,即如何分配内存空间给映射关系区域和数据缓存区域的问题。

[0004] 传统的解决方法是一种固定比例的方法,既映射关系区域和数据缓存区域的内存大小比例是固定的。例如两者大小是 1 : 1,数据缓存区和映射表所占用的空间大小一样。这种固定比例方法的问题是,不能很好地适应应用环境的变化,因为不同的应用环境下,对数据缓存区和映射关系区域大小的需求不尽相同,这种固定比例的方式将影响固态硬盘的整体性能。

### 发明内容

[0005] 本发明所解决的技术问题在于,提出一种固态硬盘的内存分区方法,在不增加固态硬盘的系统开销和软件开销的前提下,获得数据缓存区和映射关系区域之间的最佳比例关系,确保固态硬盘在不同负载下均能取得最佳或者接近最佳的性能。

[0006] 为实现上述目的,本发明采用如下技术方案:

[0007] 一种固态盘的内存分区方法,包括如下具体步骤:

[0008] (1) 建立两个计数器  $P_M$  和  $P_B$ , 分别用于记录因为内存中映射关系区域过小导致的第一额外操作开销, 以及因为内存中数据缓存区域过小导致的第二额外操作开销。

[0009] 其中, 所述第一额外操作开销包括将映射关系读出过程所消耗的时间, 以及将映射关系写回闪存过程所消耗的时间; 第二额外操作开销指数据写回闪存过程消耗的时间。

[0010] (2) 根据读写请求的状态更新计数器  $P_M$  和  $P_B$ , 具体为:

[0011] 对于读请求, 首先从内存中寻找该请求的映射关系, 如果该映射关系不在内存中, 需要首先将映射关系从闪存中读出, 同时查找内存中的映射关系区域是否还有空闲空间, 如果有, 则将所述读出的映射关系存放在内存的映射关系区域, 然后更新计数器  $P_M$ ; 如果没有, 首先将映射关系区域中旧的映射关系写回闪存, 然后将所述读出的映射关系保存到内存中, 最后更新计数器  $P_M$ 。

[0012] 对于写请求, 首先在内存的数据缓存区中寻找该写请求数据, 如果该写请求数据的旧数据不在数据缓存区, 查看内存的数据缓存区是否还有空间, 如果有, 则将该写请求数据直接写到内存的数据缓存区; 如果没有, 则将数据缓存区中的旧数据写回闪存, 然后将该写请求的数据写到内存的数据缓存区, 最后更新计数器  $P_B$ 。

[0013] (3) 每响应一次读写请求或一定周期之后, 调整内存中映射关系区域和数据缓存区域的大小比例, 即计算当前状态下, 内存中映射关系区域和数据缓存区域的最佳大小比例。

[0014] (4) 根据上述获得的映射关系区域和数据缓存区域的理论最佳比例, 实时调整内存的组成, 完成固态盘内存实时分区。

[0015] 作为本发明的改进, 所述步骤 (2) 中, 如果内存中的映射关系区域有空闲空间, 则使用下式更新计数器  $P_M$ :

$$[0016] \quad P_{M(\text{新值})} = P_{M(\text{旧值})} + P_R \quad (1)$$

[0017] 如果内存中的映射关系区域没有空闲空间, 则使用下式更新计数器  $P_M$ :

$$[0018] \quad P_{M(\text{新值})} = P_{M(\text{旧值})} + P_R + P_W \quad (2)$$

[0019] 作为本发明的改进, 所述步骤 (2) 中, 通过如下公式更新计数器  $P_B$ :

$$[0020] \quad P_{B(\text{新值})} = P_{B(\text{旧值})} + P_W \quad (3)$$

[0021] 作为本发明的改进, 所述步骤 (3) 中, 通过如下公式计算所述映射关系区域和数据缓存区域的最佳大小比例:

[0022]

$$\text{映射关系区域} \\ \text{当前最佳容量} = \frac{P_M}{P_B + P_M} \times \text{内存总容量} \quad (4)$$

[0023]

$$\text{数据缓存区域} \\ \text{当前最佳容量} = \frac{P_B}{P_B + P_M} \times \text{内存总容量} \quad (5)$$

[0024] 本发明中, 将内存空间除去必要的元数据, 包括各种位图, 校验信息外, 剩下的空间分成两个部分, 既数据缓存区域和映射关系区域。数据缓存区域用来保存经常用到的数据, 映射关系区域用来保存经常用到的映射关系。

[0025] 本发明中, 当一个读请求到达固态盘后, 如果其目标数据在缓存中, 则直接读取缓存中的数据, 否则需要读取闪存中的数据。在读取闪存中的数据时, 需要首先查询目标数

据的映射关系,如果这个映射关系存在于内存的映射关系区域,则可以立即获得目标数据在闪存上的地址,否则需要首先从闪存中将相对应的映射关系读出,在读出这个映射关系之后才能从闪存中读出目标数据。在读出映射关系时,存在这样一种可能,既,因为内存中的映射关系区域满了,新读出的映射关系很可能在后续继续使用,所以需要将其记录在内存的映射关系区域,为了给这个新的映射关系腾出空间,需要首先将处于映射关系区域最后的映射关系写回闪存。从这个读操作的操作过程来看,如果这个读请求的映射关系处于这个映射关系区域的话,就可以避免一次多余的读取闪存的操作(读取目标数据的映射关系)或者加上一次多余的写闪存的操作(为将目标数据的映射关系记录在映射关系区域,而需要写回在映射关系区域的旧的映射关系)。可以做一下假设,如果映射关系区域足够大,大到这个映射关系正好保存在内存中,则上述一次多余的读闪存操作或者加上一次多余的写闪存操作可以避免,在本发明中,将这种多余的访问闪存的操作称之为额外操作。

[0026] 当一个写请求到达时,如果这个写请求的数据存在于数据缓存区,写请求的新数据可以直接写到数据缓存区,否则需要将数据缓存区的最后的数据写回闪存再将这个写请求的数据写到数据缓存区的最前面,这个为写请求的数据腾出缓存空间的写回操作被认为是额外操作。与此同时,因为写回了一个数据,这个数据记录在闪存中一个新地址,这将产生一个新的映射关系,这个映射关系需要写回闪存,这个写回映射关系的操作也被认为是额外操作。

[0027] 额外操作可以影响固态盘的读写性能,减少额外操作可以直接提高固态盘的读写性能,额外操作产生的原因是固态盘中数据缓存区域偏小以及映射关系区域偏小,减少额外操作的唯一方法就是增加数据缓存区域的大小以及映射关系区域的大小。固态盘中内存的空间容量有限,不能两者均增加。在本发明中,需要在不同负载下,提高不同的数据缓存区域大小和映射关系区域大小,为了适应不同负载的特定,所以设计了一种基于惩罚的内存分区方法。这里所谓的惩罚就是前面所提到的额外操作,因为这些额外操作对固态盘的性能影响可以认为是因为数据缓存区域和映射关系区域过小导致的一种对性能的惩罚。

[0028] 采用本发明提供的方法,可以在固态盘系统运行时,根据当前的运行环境,寻找到最优的映射关系区域和数据缓存区域的容量比例,根据这个比例动态调整内存构成,在内存容量一定的情况下,达到性能最佳的目标。

#### 附图说明

[0029] 图 1 为本发明实施例的整体流程图;

[0030] 图 2 为本发明的实施例的读请求操作过程;

[0031] 图 3 为本发明的实施例的写请求操作过程;

#### 具体实施方式

[0032] 下面将结合本发明实施例中的附图,对本发明实施例中的技术方案进行清楚、完整的描述。

[0033] 本发明是一种固态盘内存分区方法,使用该方法可以主动,自适应地调整内存中映射关系区域和数据缓存区域的大小比例,具体包括如下步骤:

[0034] 图 1 是本方法的整体流程图。

[0035] 第一步:如图 1 中第一点所述,在固态硬盘上电开始工作后,进行固态硬盘的初始化工作,在完成其他常规初始化工作的同时,需要建立两个计数器,分别为  $P_M$  和  $P_B$ 。这两个计数器分别记录:因为内存中映射关系区域过小导致的第一额外操作开销,以及因为内存中数据缓存区域过小导致的第二额外操作开销。

[0036] 其中,额外操作开销指额外操作所消耗的时间,对于第一额外操作开销,其中的额外操作包括:(a) 当外部读请求的映射关系不在映射关系区域时,需要将映射关系读出,这个读出的过程就是额外操作。(b) 当新的映射关系需要记录在映射关系区域时(如第 a 点所述),如果映射关系区域空间不足时,需要将旧的映射关系写回闪存,这个映射关系的写回过程就是额外操作。即第一额外操作开销包括将映射关系读出过程所消耗的时间,以及将映射关系写回闪存过程所消耗的时间。

[0037] 对于第二额外操作开销,其中的额外操作指数据回写过程,具体为:当外部写请求的数据不在数据缓存区时,需要将数据缓存区中的数据写回闪存,为写请求的数据腾出内存空间,即数据写回闪存过程消耗的时间即为第二额外操作开销。

[0038] 内存空间除去必要的元数据,包括各种位图,校验信息外,剩下的空间分成两个部分,既数据缓存区域和映射关系区域。数据缓存区域用来保存经常用到的数据,映射关系区域用来保存经常用到的映射关系。在这里所谓的经常用到的是泛指一些采用类似最近最少使用算法(LRU),最少使用算法(LFU)保存下来的数据和映射关系,本实施例中并不特指在数据缓存区域和映射关系区域采用哪一种排序算法,即目前的各种方式都可以,本实施例可以优选使用 LRU 算法作为示例。

[0039] 第二步:如图 1,因为外部请求分为两种,读请求和写请求,在第二步中,需要根据读写请求的状态更新计数器  $P_M$  和  $P_B$ 。

[0040] 图 2 是读请求到达固态硬盘时的处理流程示意图。当读请求到达后,首先从内存中寻找该请求的映射关系,如果这个映射关系存在内存中,可以直接从内存中获取映射关系,然后立即读取目标数据,这个过程没有额外操作。但是当映射关系不在内存中,需要首先将映射关系从闪存中读出,同时需要查找内存中的映射关系区域是否还有空闲空间,如果还有空闲空间,就将刚刚读出的映射关系存放在内存的映射关系区域,然后使用公式 1 更新计数器  $P_M$ ;当内存的映射关系区域没有空闲空间,需要首先将映射关系区域中旧的映射关系写回闪存,为刚读出的映射关系腾出内存空间,然后将刚读出的映射关系保存到内存中,最后使用公式 2 更新计数器  $P_M$ 。

[0041] 图 3 是写请求到达固态硬盘时的处理流程图。当写请求到达时,首先在内存的数据缓存区中寻找该数据,如果该数据存在于数据缓存区中,直接将要写入的数据更新原数据。如果该数据的旧数据不在数据缓存区,为了将这个数据保存到内存中,需要查看内存的数据缓存区是否还有空间,如果还有空闲空间,可以将该写请求的数据直接写到内存的数据缓存区;如果内存中数据缓存区没有空闲空间,需要将数据缓存区中的旧数据写回闪存,然后才能将该写请求的数据写到内存的数据缓存区,最后使用公式 3 更新计数器  $P_B$ 。

[0042] 公式 1  $P_{M(\text{新值})} = P_{M(\text{旧值})} + P_R$

[0043] 公式 2  $P_{M(\text{新值})} = P_{M(\text{旧值})} + P_R + P_W$

[0044] 公式 3  $P_{B(\text{新值})} = P_{B(\text{旧值})} + P_W$

[0045] 在公式 1, 2, 3 中,  $P_R, P_W$  分别表示对闪存进行一次读操作的时间开销和对闪存进行

一次写操作的时间开销。 $P_M$ (新值),  $P_B$ (新值) 分别表示经过这次更新之后的  $P_M$  值,  $P_B$  值。 $P_M$ (旧值),  $P_B$ (旧值) 分别表示这次更新之前的  $P_M$  值,  $P_B$  值。

[0046] 第三步: 每响应一次读写请求或一定周期之后, 需要调整内存中映射关系区域和数据缓存区域的大小比例, 如图 1 中第三点所示。一定周期表示固定的时间段, 例如 1 小时, 这个时间段的长短可以根据负载的特点具体设定。因为之前两个计数器  $P_M$  和  $P_B$  进行了修正, 在这一步, 需要根据公式 4, 5 计算当前状态下, 内存中映射关系区域和数据缓存区域的最佳大小比例。

[0047]

$$\text{公式4} \quad \begin{array}{l} \text{映射关系区域} \\ \text{当前最佳容量} \end{array} = \frac{P_M}{P_B + P_M} \times \text{内存总容量}$$

[0048]

$$\text{公式5} \quad \begin{array}{l} \text{数据缓存区域} \\ \text{当前最佳容量} \end{array} = \frac{P_B}{P_B + P_M} \times \text{内存总容量}$$

[0049] 在公式 4, 5 中, 映射关系区域当前最佳容量和数据缓存区域当前最佳容量分别是在当前状态下, 映射关系区域和数据缓存区域的理论最佳大小。这个理论最佳大小比例可能与当前的大小比例有差别, 所以需要进入到第四步。

[0050] 第四步: 根据第三步获得的映射关系区域和数据缓存区域的理论最佳比例, 实时调整内存的组成。

[0051] 在第四步中当映射关系区域的实际容量大于理论最佳容量, 这意味着映射关系区域过大, 数据缓存区域过小, 需要将映射关系区域的部分映射关系数据写回闪存, 腾出的内存空间划拨作为新增加的数据缓存区域; 当数据缓存区域的实际容量大于理论最佳容量, 这意味着数据缓存区域过大, 映射关系区域过小, 需要将数据缓存区域的部分数据写回闪存, 腾出的内存空间划拨作为新增加的映射关系区域。

[0052] 在整个固态硬盘运行过程中, 循环执行经过这四步操作, 可以实时地调整映射关系区域和数据缓存区域的容量, 保证当前两者的容量比例是最优的。

[0053] 在本实施例中, 对于特定闪存颗粒, 寻找到读操作与写操作的大致操作比例, 使用这些比例作为额外操作时间的绝对值。例如, 假设固态硬盘所使用的闪存芯片的读写周期时间是 25ns, 读介质时间为 20us, 写介质时间为 200us, 一个页大小为 2KB, 那么读写操作的时间比例是  $(25 \times 2048 \times 10^{-3} + 20) : (25 \times 2048 \times 10^{-3} + 200) = 712 : 251.2 \approx 7 : 25$ 。在本实施例中, 根据不同的情况, 修改  $P_M$  和  $P_B$  的值, 在前面已经描述了最常见的两种情况的修改方法, 而实际上, 在本实施例中存在 16 种不同的情况, 这 16 种不同的情况分别对应一种计算公式。通过 16 种公式, 可以累计得到一段时间内的  $P_M$  和  $P_B$  值, 根据  $P_M$  和  $P_B$  的比值, 来动态决定映射关系区域与数据缓存区域的大小比例。因为  $P_M$  和  $P_B$  的比值是因为各自区域过小导致的额外操作的总量, 这个比例反映了两种区域对内存区域的需要程度, 所以用  $P_M$  和  $P_B$  的比值可以直接决定内存的分区比例。

[0054] 为了实时的反映不同负载, 以及同一负载在不同时间段内对两种区域的需要程度, 在本实施例中, 还可以设置一个时间窗口, 这个时间窗口是一个经验值。通常来说, 可以将这个时间窗口设置为一小时。在每个时间窗口开始时, 将  $P_M$  和  $P_B$  的值清零,  $P_M$  和  $P_B$  开始重新统计目前时间段内的负载对于两种区域的需求程度。

[0055] 在每个响应一个外部请求后, 根据当前  $P_M$  和  $P_B$  的比值, 调整两种区域的大小。当

目前映射关系区域所占内存区域过大时,将处于映射关系区域后面的部分映射关系写回闪存,新产生的内存空间立即分配给数据缓存区域,直到两者的比例符合当前  $P_M$  和  $P_B$  的值。反之,则将数据缓存区域的部分数据写回,进而扩大映射关系区域的空间。



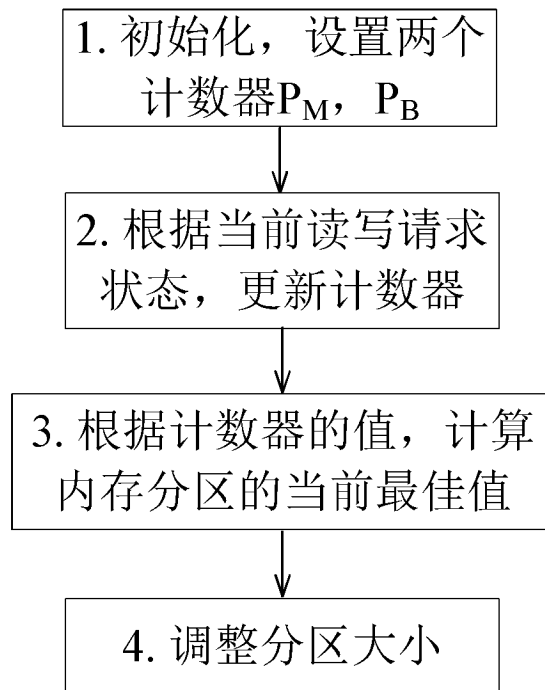


图 1

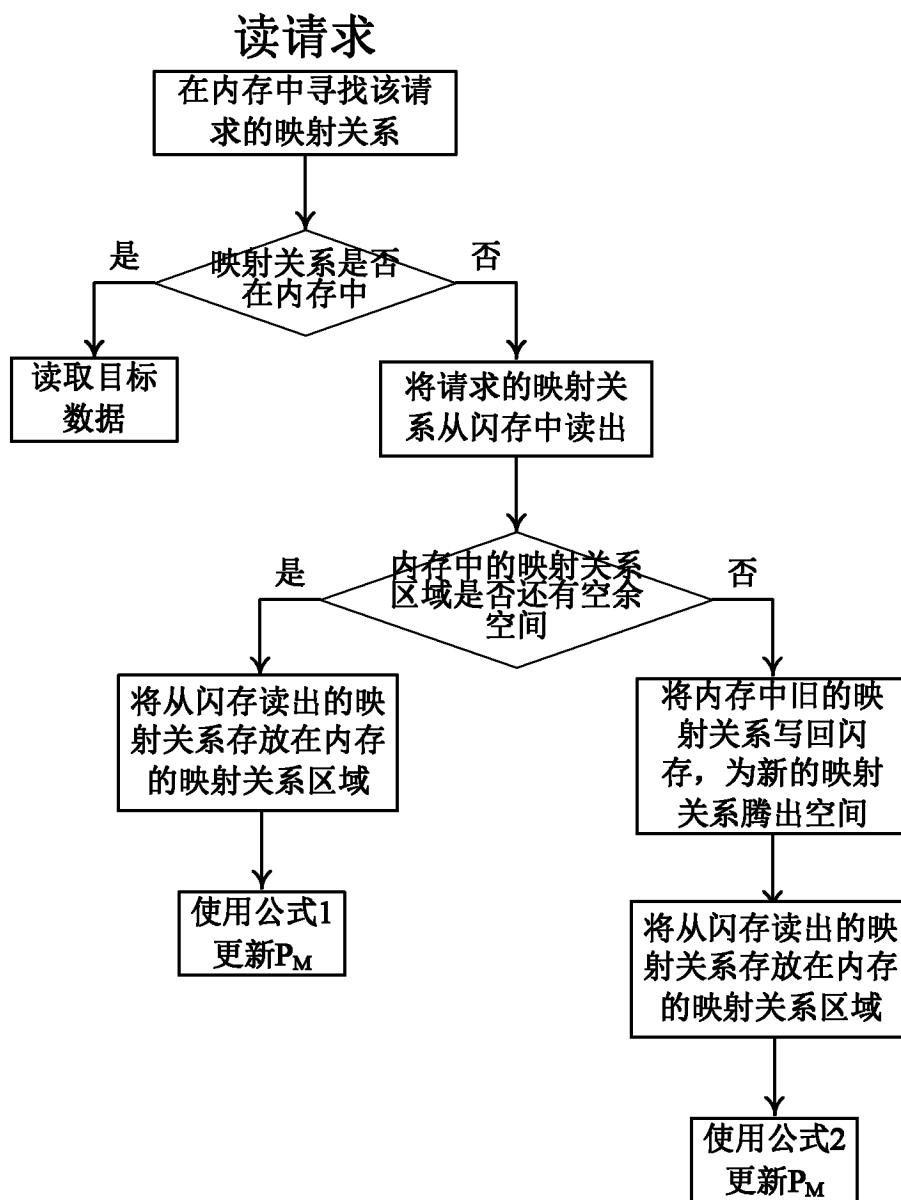


图 2

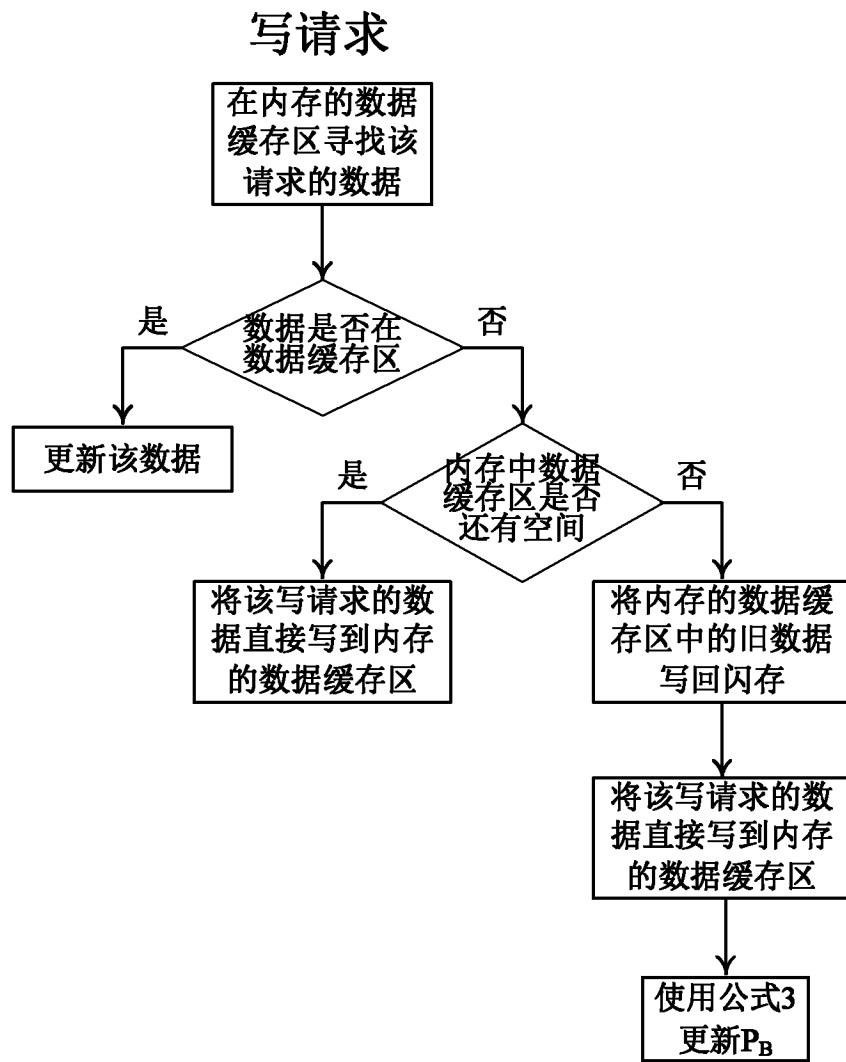


图 3