



(12)发明专利

(10)授权公告号 CN 104978335 B

(45)授权公告日 2018.12.07

(21)申请号 201410136736.1

(22)申请日 2014.04.04

(65)同一申请的已公布的文献号

申请公布号 CN 104978335 A

(43)申请公布日 2015.10.14

(73)专利权人 阿里巴巴集团控股有限公司

地址 英属开曼群岛大开曼资本大厦一座四
层847号邮箱

(72)发明人 罗军

(74)专利代理机构 北京国昊天诚知识产权代理
有限公司 11315

代理人 许志勇

(51)Int.Cl.

G06F 17/30(2006.01)

G06F 21/62(2013.01)

(56)对比文件

CN 1609827 A,2005.04.27,

CN 1815469 A,2006.08.09,

CN 101286171 A,2008.10.15,

CN 103365929 A,2013.10.23,

审查员 周循

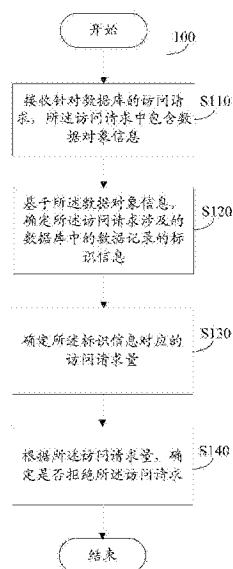
权利要求书2页 说明书13页 附图4页

(54)发明名称

数据访问控制方法和装置

(57)摘要

本申请涉及数据访问控制方法和装置。该方法包括:接收针对数据库的访问请求,访问请求中包含数据对象信息;基于数据对象信息,确定访问请求涉及的数据库中的数据记录的标识信息;确定标识信息对应的访问请求量;以及根据访问请求量,确定是否拒绝该访问请求。根据本申请的技术方案,能够以更细的流控粒度来控制对数据库的访问,即针对数据库中的各条记录进行访问控制,从而避免部分记录过热导致整个数据库性能下降的问题。另外,通过采用内存排队机制和等待超时机制来操控对数据库的访问请求量,避免了直接拒绝、一刀切的粗暴解决方式,从而减少系统资源和流量的损失,方便用户的使用并增强用户体验。



1. 一种数据访问控制方法,其特征在于,包括:
接收针对数据库的访问请求,所述访问请求中包含数据对象信息;
基于所述数据对象信息,确定所述访问请求涉及的数据库中的数据记录的标识信息;
确定所述标识信息对应的访问请求量,所述访问请求量根据所述标识信息对应的处于等待状态的访问请求的数量和/或所述标识信息对应的处于执行状态的访问请求的数量确定;以及
根据所述访问请求量,确定是否拒绝所述访问请求。
2. 根据权利要求1所述的方法,其特征在于,所述数据库包含一个或多个商品标识,每个商品标识包含一条或多条具有商品最小库存单元SKU标识的数据记录,所述数据对象信息中包括待访问数据对象的商品标识和商品最小库存单元SKU标识;以及
所述基于所述数据对象信息,确定所述访问请求涉及的数据库中的数据记录的标识信息的步骤,包括:
基于所述商品标识和商品最小库存单元SKU标识,确定所述访问请求涉及的数据库中的数据记录的标识信息。
3. 根据权利要求2所述的方法,其特征在于,还包括:
预先设置一记录表,所述记录表中记录有需要进行访问控制的数据对象的商品标识;
在接收到所述针对数据库的访问请求后,判断所述访问请求中包含的数据对象信息中的商品标识是否在所述记录表中,如果是,则执行所述确定所述访问请求涉及的数据库中的数据记录的标识信息的步骤。
4. 根据权利要求1所述的方法,其特征在于,所述确定所述标识信息对应的访问请求量的步骤,包括:
判断缓存中是否存在所述标识信息;
如果存在,则获取所述标识信息对应的访问请求量;以及
如果不存在,则在缓存中创建所述标识信息对应的访问请求量并对所述访问请求量赋予初始值。
5. 根据权利要求1所述的方法,其特征在于,所述根据所述访问请求量确定是否拒绝所述访问请求的步骤,包括:
判断所述标识信息对应的访问请求量是否超过第一阈值;以及
如果超过,则拒绝所述访问请求,否则接受所述访问请求。
6. 根据权利要求1所述的方法,其特征在于,还包括:
当在根据所述访问请求量确定是否拒绝所述访问请求的步骤中接受所述访问请求时,执行以下步骤:
判断缓存中是否存在所述标识信息;
如果存在,则将所述标识信息对应的访问请求量加一;
如果不存在,则在缓存中创建所述标识信息对应的访问请求量并赋予初始值。
7. 根据权利要求1所述的方法,其特征在于,还包括:
当在所述根据所述访问请求量确定是否拒绝所述访问请求的步骤中接受所述访问请求时,执行以下步骤:
判断缓存中是否存在所述标识信息;

如果存在,则将该访问请求加入与所述标识信息对应的等待队列中,所述等待队列中记录有与标识信息对应的处于等待状态的访问请求;

如果不存在,则在所述缓存中创建与所述标识信息对应的等待队列和执行队列,并将该访问请求加入该创建的等待队列中,其中,所述执行队列中记录有与所述标识信息对应的从等待队列中获得的处于执行状态的访问请求。

8.根据权利要求7所述的方法,其特征在于,所述确定所述标识信息对应的访问请求量,包括:

根据与所述标识信息对应的等待队列中的访问请求和/或执行队列的访问请求的数量,确定所述标识信息对应的访问请求量。

9.根据权利要求7所述的方法,其特征在于,还包括:

当所述执行队列中存在执行完毕的访问请求时,则从执行队列中删除该执行完毕的访问请求。

10.根据权利要求7所述的方法,其特征在于,在将所述访问请求加入所述标识信息对应的等待队列中之后,还包括:

判断与所述标识信息对应的执行队列中的访问请求的数量是否小于第二阈值;

如果是,则从与所述标识信息对应的等待队列中获得访问请求并加入到与所述标识信息对应的执行队列中。

11.根据权利要求7-10中任一项所述的方法,其特征在于,还包括:

当所述等待队列中的访问请求的等待时间超过第三阈值时,则从等待队列中删除该等待时间超过第三阈值的访问请求。

12.一种数据访问控制装置,其特征在于,包括:

接收模块,用于接收针对数据库的访问请求,所述访问请求中包含数据对象信息;

标识信息确定模块,用于基于所述数据对象信息,确定所述访问请求涉及的数据库中的数据记录的标识信息;

访问请求量确定模块,用于确定所述标识信息对应的访问请求量,所述访问请求量根据所述标识信息对应的处于等待状态的访问请求的数量和/或所述标识信息对应的处于执行状态的访问请求的数量确定;以及

控制模块,用于根据所述访问请求量,确定是否拒绝所述访问请求。

数据访问控制方法和装置

技术领域

[0001] 本申请涉及数据库领域,尤其涉及数据访问控制方法和装置。

背景技术

[0002] 随着互联网技术的飞速发展,网上进行业务交互已经越来越普遍,随之而来的是对数据库的高并发、大请求量的访问。

[0003] 例如,在类似“双11”、“双12”等电商促销期间,客户端应用对服务器数据库的访问请求(读写请求)会突然加大,特别是对那些性价比高的畅销商品来说,操作数据库中相关记录的请求会瞬间大增,这导致在数据库服务端对某条记录产生行锁竞争和等待。当竞争非常激烈,即对某条数据库记录的修改并发度非常高的时候,数据库性能会急剧下降,从而影响其他记录的正常读写。因此,需要针对这种因部分记录过热导致整个数据库性能下降的问题寻求解决方案。

[0004] 在现有技术中,针对高并发、大请求量的情况,通常是从整体上对数据库的访问进行流量控制,包括应用层面的数据库(DB)流控和应用入口层面的HSF(High-Speed Service Framework,高速服务框架)流控。具体而言,DB流控主要关注同一时刻对某个逻辑库(数据库包括多个逻辑库)的请求量,其中判断对数据库的访问请求(包括读请求和写请求)是否达到该数据库处理能力的极限,如果达到则限流。而HSF流控主要关注一段时间内(一般是1秒)客户端应用对服务器系统的平均请求量,类似地,其中判断对服务器系统(整个数据库)的访问请求是否达到数据库处理能力的极限,如果达到则限流。

[0005] 可见,上述现有的两种流控方案都侧重于解决系统整体性能问题,都未涉及并且也无法解决目前出现的这种因部分记录过热导致整个数据库性能下降的问题。

[0006] 另外,在现有的流控方案中,当某一时刻的请求超过阈值时系统就会拒绝为这些请求提供服务。然而,系统流量突然飙升往往只是一瞬间的事情(如秒杀,特别是天猫双11的秒杀活动,持续时间很短)。如果系统直接把超过处理能力的请求拒绝掉,可能会导致用户不断地重试,这样反而增加了系统高压力的持续时间,而且用户体验也较差。

[0007] 因此,需要一种改进的数据访问控制方案,来克服上述现有技术中的这些问题。

发明内容

[0008] 本申请的一个目的在于提供一种改进的数据访问控制方案,以解决因部分记录过热导致整个数据库性能下降的问题。

[0009] 本申请的另一目的在于提供一种改进的数据访问控制方案,以解决现有流控机制导致用户不断重试、系统压力增大和用户体验差的问题。

[0010] 根据本申请一个方面的实施例,提供一种数据访问控制方法,其特征在于,包括:接收针对数据库的访问请求,所述访问请求中包含数据对象信息;基于所述数据对象信息,确定所述访问请求涉及的数据库中的数据记录的标识信息;确定所述标识信息对应的访问请求量;以及根据所述访问请求量,确定是否拒绝所述访问请求。

[0011] 根据本申请另一方面的实施例,提供一种数据访问控制装置,其特征在于,包括:接收模块,用于接收针对数据库的访问请求,所述访问请求中包含数据对象信息;标识信息确定模块,用于基于所述数据对象信息,确定所述访问请求涉及的数据库中的数据记录的标识信息;访问请求量确定模块,用于确定所述标识信息对应的访问请求量;以及控制模块,用于根据所述访问请求量,确定是否拒绝所述访问请求。

[0012] 与现有技术相比,根据本申请的技术方案,能够以更细的流控粒度来控制对数据库的访问,即针对数据库中的各条记录进行访问控制,从而避免部分记录过热导致整个数据库性能下降的问题。另外,通过采用内存排队机制和等待超时机制来操控对数据库的访问请求量,避免了直接拒绝、一刀切的粗暴解决方式,从而减少系统资源和流量的损失,方便用户的使用并增强用户体验。

附图说明

[0013] 此处所说明的附图用来提供对本申请的进一步理解,构成本申请的一部分,本申请的示意性实施例及其说明用于解释本申请,并不构成对本申请的不当限定。在附图中:

[0014] 图1示出根据本申请一个实施例的数据访问控制方法的流程图;

[0015] 图2示出根据本申请另一实施例的数据访问控制方法的流程图;

[0016] 图3示出根据本申请又一实施例的数据访问控制方法的流程图;

[0017] 图4示出根据本申请实施例的数据访问控制过程的示意图;

[0018] 图5示出根据本申请一个实施例的内存管理模式的示意图;以及

[0019] 图6示出根据本申请一个实施例的数据访问控制装置的结构框图。

具体实施方式

[0020] 本申请发明人注意到,如前面提及的,目前一般的流控方案都侧重于解决系统整体性能问题。如HSF流控:当应用负载较高时,则调低HSF流控阈值,降低系统整体流量,在系统层面属于一刀切的粗暴解决方案。并且,目前一般的流控方案为当某一时刻的请求超过阈值时,系统会拒绝为这些请求提供服务。然而,系统流量突然飙升往往只是一瞬间的事情(如秒杀,特别是天猫双11的秒杀活动,持续时间很短)。如果系统直接把超过处理能力的请求拒绝掉,可能会导致用户不断地重试,这样反而增加了系统高压力的持续时间,而且用户体验也较差。

[0021] 实际上系统大部分时间处理能力是有余量的。某一时刻系统面临的请求量可能会超过自身处理能力的范围,而在下一时刻可能请求量就恢复到处理能力水平以下。如果在请求高峰期做削峰处理后,对超出的请求直接拒绝服务,会带来系统资源和流量的损失。

[0022] 另外,本申请发明人还注意到,现有流控方案都是基于某个范围,例如DB流控基于空间范围(逻辑库),HSF流控基于时间范围(1秒),可见控制粒度比较粗,都未涉及到对数据库记录层面的控制,也无法解决因部分记录过热导致整个数据库性能下降的问题。

[0023] 因此,本申请的主要思想在于,引入更细粒度化的控制策略,在数据库记录层面上实施流控,即针对数据库中的一条记录或多条记录(特定数据对象或访问对象相关的记录)实施并发流控,以避免部分记录过热导致数据库性能下降,进而确保其它记录的正常读写,提高服务器系统的整体效率,方便用户的使用和提高用户的使用感受。

[0024] 另外,本申请的思想还在于,引入排队策略,使得超过系统处理能力的请求先在内存排队等候处理,等待超时后再拒绝。通过这种方式,可以使得在高并发、大请求量情况下尽可能地避免系统资源和流量的损失,方便用户的使用并且提高用户体验感受。

[0025] 为使本申请的目的、技术方案和优点更加清楚,以下结合附图及具体实施例,对本申请作进一步的详细说明。

[0026] 参照图1,图1示出根据本申请一个实施例的数据访问控制方法100的流程图。

[0027] 如图1所示,在步骤S110处,接收针对数据库的访问请求,所述访问请求中包含数据对象信息。

[0028] 具体而言,访问请求通常是针对特定数据对象发出的,该特定数据对象对应于数据库中的一条或多条数据记录,所以伴随访问请求的通常是针对数据库中的该一条或多条数据记录进行的查询、删除、修改等操作。与此同时,在针对数据库的访问请求中需要包含想要访问的数据对象的信息。

[0029] 在本申请的一个具体应用场景中,数据对象可以是各种商品,而数据库狭义上讲可以是指商品的库存记录,例如,该数据库可以包含一个或多个商品数据对象,每个商品数据对象都具有相应的商品标识,每个商品标识可以包含一条或多条具有商品最小库存单元(SKU)标识的数据记录,表1示出了数据库的示意性结构。如表1所示,每个商品标识对应数据库中的一条或多条数据记录,每条数据记录均具有商品SKU标识。数据记录的内容可以包括商品属性、商品SKU标识、库存量等。

[0030] 相应地,访问请求中的数据对象信息可以包括待访问数据对象(在本例中,待访问商品)的商品标识和商品SKU标识。

[0031] 表1

[0032]

数据对象 (商品)	商品标识	商品属性		商品 SKU 标识	库存量	记录标识
		颜色	尺码			
鞋子 A	PA	红色	37 码	R1	2	Q1
		红色	38 码	R2	3	Q2
		蓝色	37 码	R3	5	Q3
		蓝色	38 码	R4	0	Q4
		蓝色	39 码	R5	7	Q5
鞋子 B	PB	红色	37 码	R6	1	Q6
		红色	38 码	R7	1	Q7
		蓝色	37 码	R8	2	Q8
		蓝色	38 码	R9	2	Q9
		蓝色	39 码	R10	8	Q10
						

[0033] 接下来,在步骤S120处,基于数据对象信息,确定访问请求涉及的数据库中的数据记录的标识信息。

[0034] 根据本申请的构思,在记录层面上实施更细粒度的访问控制,即,针对数据库中的各条数据记录统计相应的访问请求量并据此进行流量控制。因此,可监控和记录针对各条数据记录的访问请求的数量,进而需要确定访问请求所涉及的数据库中的数据记录的标识信息,以便于进行该监控和记录。换言之,在本申请的上下文中,该标识信息用于表征访问请求所涉及的访问对象,即,数据库中的某条数据记录。在具体实施例中,可以分别统计针对数据库中各条数据记录的访问请求量,并以该标识信息为索引记录在缓存或内存中。

[0035] 具体地,在本申请的一个实施例中,可以基于数据对象信息中的商品标识和商品SKU标识,确定访问请求涉及的数据记录的标识信息。如表1所示,在数据库中,每个商品标识可以对应或包含一个或多个商品SKU标识,每个商品SKU标识可以对应一个记录标识。则在访问控制过程中,在接收到包含数据对象信息的访问请求时,可以根据数据对象信息中的商品标识和商品SKU标识,确定出对应的记录标识,作为访问请求涉及的数据记录的标识信息。在本申请的其它实施例中,也可以采用商品SKU标识作为访问请求涉及的数据记录的标识信息。

[0036] 由于数据库中的数据记录的数量极其庞大,如果针对所有记录都进行访问请求量的监控,运算量会非常大。并且有些记录访问量很低,而有些记录访问量极高,如果针对访问量很低的记录进行监控,意义也不大。因此,根据本申请的优选实施例,可以仅针对可能

访问量较高的数据对象进行细粒度的访问控制。

[0037] 在一个具体实施例中,可以预先设置一记录表,所述记录表中记录有需要访问控制的数据对象的商品标识。在接收到所述针对数据库的访问请求后,判断所述访问请求中包含的数据对象信息中的商品标识是否在所述记录表中。如果是,则确定访问请求涉及的数据库中的数据记录的标识信息,并监控标识信息对应的访问请求量以在记录层面上进行细粒度访问控制。否则不对该访问请求进行该细粒度访问控制。记录表的格式可以如表2所示。继续沿用上例,当接收到针对鞋子A的访问请求时,通过表2所示的记录表可以确定出鞋子A的商品标识存在于记录表中,即鞋子A是需要进行访问控制的数据对象。因此,基于访问请求中包含的数据对象信息中的商品标识和商品SKU标识,确定访问请求涉及的数据记录的标识信息,以在内存中建立对该鞋子A的所有商品SKU标识对应的所有数据记录的访问监控。即,以与该鞋子A的各商品SKU标识对应的各数据记录的标识信息为索引,分别针对相应的数据记录统计访问请求量,从而进行记录层面上的细粒度访问控制。而当接收到针对鞋子B的访问请求时,通过表2所示的记录表可以确定出鞋子B的商品标识并不存在于记录表中,即鞋子B不是需要进行访问控制的数据对象,因此,不需要在内存中建立对鞋子B的各商品SKU标识对应的各数据记录的监控。

[0038] 表2

[0039]

数据对象(商品)	商品标识
鞋子A	PA
.....	

[0040] 由此,可以有针对性地针对某些访问请求执行更细粒度的流量控制,从而更有效地提高数据库性能,方便用户使用和增强用户体验。

[0041] 在确定了访问请求涉及的数据库中数据记录的标识信息之后,在步骤S130处,确定标识信息对应的访问请求量。

[0042] 具体而言,根据本申请的实施例,服务器可以针对每个访问请求涉及的访问对象(数据库中的某条数据记录)进行监控,其中可以针对不同访问对象被访问的次数进行统计,得到与不同访问对象(某条数据记录)对应的访问请求量。该访问请求量可以存储在缓存或内存中以供访问控制使用,格式可以如表3所示。

[0043] 表3

[0044]

标识信息(例如记录标识)	访问请求量(单位:次)
Q1	5,000
Q2	3,500
Q3	1,500
Q4	10,000
Q5	1,000
.....	

[0045] 当接收到访问请求时,根据确定的访问请求涉及的数据记录的标识信息,可以获得到与该标识信息对应的访问请求量。

[0046] 接下来,在步骤S140处,根据访问请求量,确定是否拒绝该访问请求。

[0047] 根据本申请的优选实施例,在接收到访问请求并获取到所涉及的访问请求量之后,例如,可以通过先将该访问请求量加一来更新与该标识信息对应的该访问请求量,然后再根据更新后的访问请求量来确定是否拒绝该访问请求。由此可以更好地进行流量控制,因为在类似“双11”之类的电商促销期间,针对热门商品的购买请求往往在同一时间猛增,所以通过预先估计访问请求量(预先加一)可以更好地保护数据库免受冲击,进而提高数据库的性能。

[0048] 根据本申请的实施例,可以根据阈值比较方法来确定是否拒绝该访问请求。具体地,当与标识信息对应的访问请求量超过预定阈值(第一阈值)时,可以拒绝当前的访问请求;而当与标识信息对应的访问请求量未超过预定阈值(第一阈值)时,可以针对当前的访问请求执行数据访问操作。

[0049] 由此可以在数据库记录层面上实现流量控制,避免部分记录过热情况的发生,进而避免部分记录过热引起整个数据库性能下降。

[0050] 下面结合图2对本申请实施例的数据访问控制方法进行更详细的描述。图2示出了根据本申请另一实施例的数据访问控制方法200的流程图。

[0051] 如图2所示,在步骤S201处,接收针对数据库的访问请求,该访问请求中包含数据对象信息。

[0052] 在步骤S202处,基于数据对象信息,确定访问请求涉及的数据库中的数据记录的标识信息。

[0053] 该步骤S201和S202的处理类似于前面结合图1描述的步骤S110和S120,这里不再赘述。

[0054] 在确定了当前访问请求涉及的数据库中的数据记录的标识信息之后,在步骤S203处,判断缓存中是否存在该标识信息。

[0055] 如果判定缓存中存在标识信息,则表明在缓存中已经建立对该标识信息的访问请求量的监控,此时进入步骤S204。

[0056] 在步骤S204处,可以从缓存中获取与该标识信息对应的访问请求量。在优选实施例中,可以在获取到缓存中的已有访问请求量之后,针对当前的访问请求更新该访问请求量,例如,加一,然后根据更新后的访问请求量再确定是否拒绝当前的访问请求。当然本申请并不限于此,也可以依据获取到的已有访问请求量确定是否拒绝当前的访问请求,当接受该访问请求之后再更新缓存中存储的访问请求量。

[0057] 如果判定缓存中不存在标识信息,则表明在缓存中还没有建立对该标识信息的访问请求量的监控,此时进入步骤S205。在步骤S205处,在缓存中创建与该标识信息对应的访问请求量并赋予初始值。例如,可以为访问请求量赋予初始值零或一或其它任意值。

[0058] 在获取到与该标识信息对应的访问请求量之后,进入步骤S206,判断该访问请求量是否超过第一阈值。

[0059] 如果判定该访问请求量超过第一阈值,则进入步骤S207,拒绝该访问请求;如果判断该访问请求量未超过第一阈值,则进入步骤S208,接受该访问请求。

[0060] 也就是说,根据针对该标识信息的访问请求的数量来局部控制针对数据库中特定数据记录的访问流量,从而避免因部分记录过热而影响其他记录的正常访问,同时这样也

可以控制数据库的访问流量,以避免访问压力过大造成数据库性能下降。

[0061] 根据本申请的实施例,第一阈值可以是根据数据库的处理能力、应用本身的性能、应用可以接受的最长响应时间等因素来预先设定的。也就是,可以采用根据这些因素设定阈值的方法来限制单条数据库记录更新所允许的最大并发数。在一个具体实施例中,目前例如mysql数据库能够承受的并发修改进程数一般为10,则第一阈值通常可以设为5到7之间。在具体业务中通常经过性能测试可以获得一个较合理的值。

[0062] 如前面提到的,当接受所述访问请求时,可以更新缓存中与该标识信息对应的访问请求量,以供后续访问控制使用。具体地,可以执行这样的步骤:判断缓存中是否存在所述标识信息;如果存在,则将所述标识信息对应的访问请求量加一;如果不存在,则在缓存中创建所述标识信息对应的访问请求量并赋予初始值。

[0063] 下面结合图3对本申请实施例的数据访问控制方法进行更进一步的详细描述。图3示出了根据本申请又一实施例的数据访问控制方法300的流程图。其中主要描述当接受了访问请求之后如何针对各数据记录进行访问请求量的监控以及相关联的内存管理。

[0064] 如图3所示,当接受了访问请求之后,进入步骤S301,判断缓存中是否存在该访问请求涉及的数据库中的数据记录的标识信息。

[0065] 如前面结合图1和图2描述的那样,在接收到访问请求后,可以确定出该访问请求涉及的数据库中数据记录的标识信息,这里不再赘述。

[0066] 在步骤S301中,如果判定缓存中存在该标识信息,则进入步骤S302,将当前的访问请求加入与该标识信息对应的等待队列中。如果判定缓存中不存在该标识信息,则进入步骤S303,在缓存中创建与该标识信息对应的等待队列和执行队列,并将访问请求加入所创建的等待队列中。

[0067] 需要指出的是,在本申请实施例中,针对访问请求涉及的各数据记录(访问对象),都分别建立等待队列和执行队列。等待队列用于存放针对该数据记录的各待允许执行的访问请求,即,记录有与标识信息对应的处于等待状态的访问请求。执行队列用于存放针对该数据记录的各待执行的访问请求,即,记录有与标识信息对应的处于执行状态的访问请求。等待队列和执行队列可以如表4所示。

[0068] 表4

[0069]

标识信息(记录标识)	等待队列	执行队列
Q1	Q1 访问请求 4	Q1 访问请求 1
	Q1 访问请求 5	Q1 访问请求 2
	Q1 访问请求 6	Q1 访问请求 3

[0070]

Q2	Q2 访问请求 4	Q2 访问请求 1
	Q2 访问请求 5	Q2 访问请求 2
		Q2 访问请求 3
Q3	Q3 访问请求 4	Q3 访问请求 1
		Q3 访问请求 2
		Q3 访问请求 3
Q4	Q4 访问请求 4	Q4 访问请求 1
	Q4 访问请求 5	Q4 访问请求 2
	Q4 访问请求 6	Q4 访问请求 3
	Q4 访问请求 7	
Q5		Q5 访问请求 1
		Q5 访问请求 2
.....		

[0071] 在将访问请求加入了对应的等待队列中(步骤S302或S303)之后,进入步骤S304,判断与标识信息对应的执行队列中的访问请求的数量是否小于第二阈值。

[0072] 也就是说,在访问请求进入等待队列之后,申请进入执行队列。具体而言,可以根

据执行队列中的访问请求的数量,来判断是否可将当前的访问请求放入执行队列中。

[0073] 根据本申请的实施例,第二阈值可以是根据数据库的处理能力等因素来预先设定的。也就是,采用根据这些因素设定阈值的方法来限制单条数据库记录更新所允许的最大并发数。在一个具体实施例中,目前例如mysql数据库能够承受的并发修改进程数一般为10,当数据库向几十甚至上百个上层应用提供服务时,一般将单个应用系统的第二阈值配置为1或2;此外这个阈值的设置还与应用本身的性能有关。

[0074] 类似于前面提及的,本申请实施例引入排队机制,这里设置执行队列以存放待执行操作的访问请求。

[0075] 更具体而言,当在步骤S304处判定与该标识信息对应的执行队列中的访问请求的数量大于等于第二阈值时,进入步骤S305。在步骤S305处,判断等待队列中的各访问请求的等待时间是否超过第三阈值。

[0076] 根据本申请的实施例,第三阈值可以是根据数据库处理能力、应用可接受的最长响应时间等因素来预先设定的。在一个具体实施例中,第三阈值可以为10ms。当然本申请对其具体取值并不做任何限制,可以根据需要进行任意设置。

[0077] 当在步骤S305处判定等待时间超过(如大于等于)第三阈值时,进入步骤S306,拒绝等待时间超过第三阈值的访问请求。

[0078] 当在步骤S305处判定等待时间未超过(如小于)第三阈值时,进入步骤S304,即继续申请进入执行队列。

[0079] 当在步骤S304处判定执行队列中的访问请求的数量小于第二阈值时,进入步骤S307。在步骤S307处,从该标识信息对应的等待队列中获得访问请求并加入到该标识信息对应的执行队列中,以待执行相应的数据访问操作。

[0080] 当执行队列中存在执行完毕的访问请求时,可删除该执行完毕的访问请求。

[0081] 在本申请实施例的这种基于等待队列和执行队列的内存管理机制下,可以根据与标识信息对应的等待队列中的访问请求的数量确定该标识信息对应的访问请求量,也可以根据与标识信息对应的执行队列中的访问请求的数量确定该标识信息对应的访问请求量,或者可以根据与标识信息对应的等待队列和执行队列中的访问请求总量确定该标识信息对应的访问请求量,以供访问控制时使用。

[0082] 图4示出根据本申请实施例的数据访问控制过程的示意图,其中每个进入的访问请求都需要经过这样的处理过程。如图4所示,W-Q表示等待队列中访问请求的数量,R-Q表示执行队列中访问请求的数量。

[0083] 根据本申请的实施例,在接收到访问请求之后,先申请进入等待队列,如果访问请求申请等待队列成功,则等待队列的计数W-Q加1。然后申请进入执行队列,如果等待队列中的访问请求抢占执行队列成功,则R-Q加1并且W-Q减1。如果请求执行队列失败,则继续在等待队列中等待。如果W-Q对应的等待队列已满或等待超时,则拒绝该访问请求或将该访问请求从等待队列中剔除,并且W-Q减1。而针对执行队列中的访问请求,则依次执行相应的数据访问操作,即对数据库进行相应的增加、删除、修改等更新操作,并且每执行完一个数据访问操作,R-Q就减1,即,将该访问请求从执行队列中剔除。

[0084] 在实际操作中,由于数据库中的记录数量巨大,而根据本申请构思,针对每个数据库记录(访问对象)都需要计数等待允许执行的访问请求量和待执行的访问请求量,即,针

对每个数据库记录都需要维护对应的访问请求的等待队列和执行队列。因此需要一种有效的内存管理方法,否则内存会越来越庞大,最终导致系统性能下降甚至宕机。

[0085] 根据本申请的具体实施例,可以通过环形数据结构例如环形缓冲区(RingBuffer)来管理等待队列和执行队列的访问请求。图5示出了根据本申请一个实施例的内存管理模式的示意图,下面就结合图5对本申请实施例的环形数据结构下的内存管理方式进行描述。

[0086] 首先说明的是,本申请实施例对请求的控制粒度可以细化到数据库的每条记录(例如,商品的各SKU记录)或包括多条记录的记录块(例如,某商品的记录),每条记录或每个记录块都有一个唯一标识(图4中的记录id),即对应于本文中使用的数据库记录的标识信息。由于对每条被请求的数据库记录即每种访问对象都需要做访问流量控制,故每条请求过的数据可在内存中有一个计数器(控制器),计数器中可以包含两个子计数器:等待队列计数器和执行队列计数器。根据本申请的实施例,等待队列和执行队列的计数器可以由原子计数器(AtomicInteger)实现。

[0087] 如图5所示,可以设计一个分为三段的环形数据结构来进行计数器管理和内存控制,例如,每段可以代表一分钟时间,三分钟可以为一个循环。需要指出的是,环形设计便于计数器管理和内存控制,是一种优选方式。另外,本申请并不限于这里例举的三段设计,也可以为其它段设计,每段的时间也可以任意设置。

[0088] 具体来说,环中每个节点都是一个映射(Key-Value对),其中Key分别为0、1、2, Value又是一个映射的集合,此映射的Key是标识信息(记录标识,图5中示为记录id), Value是一个计数器对象(计数器),实际上对应于前述的与数据库对象对应的访问请求量,如前面的表3所示。

[0089] 环节点0、1、2分别表示当前系统的分钟时间对3取模,即每分钟内所有被请求记录的计数器对象都集中在环的同一个节点。

[0090] 分钟切换时,环节点也会随之切换。当接收到一个记录访问请求时,会先判断当前环节点(假设0节点)内是否存在相应的计数器对象。如果有,则直接更新此计数器对象的数据;如果没有,则去上一环节点(2节点)内查找,存在则更新此计数器对象的数据,并将该计数器对象更新到当前环节点;如果当前节点与上一节点内都没有对应计数器对象,则在当前节点内新建对应该记录的计数器对象。

[0091] 每次发生环节点切换后,都会把当前节点的下一环节点(1节点)包含的所有计数器对象清理掉,供垃圾回收器回收。

[0092] 可见,根据本申请实施例的环形数据结构的内存管理模式,可以将特定时段内(本例中为两分钟内)未执行的访问请求自动清除,便于计数器的管理和内存控制,从而实现柔性的内存管理模型,提高系统性能。

[0093] 应理解到,上述环形数据结构中的切换时间并不限于上述示例,而是可以根据需要任意配置。并且,本申请并不限于这种环形数据结构的内存管理模式,而是可以采用任意合适的其它内存管理模式。

[0094] 以上结合图2至图5描述了根据本申请另一实施例的数据访问控制方法。其中通过在数据库记录层面上针对记录的高并发请求引入排队处理机制,从而在实现更细粒度的流控的同时,进一步实现弹性的请求流控策略,即不直接拒绝,而是有等待超时机制,从而避免系统资源和流量的损失,进而方便用户的使用并且提高用户体验感受。

[0095] 可以认识到,本申请实施例的数据访问控制方法是一种流控策略,其适应于高并发场景,和系统整体请求量无直接关系,而是和控制粒度有关,并且与其它流控手段并不冲突。

[0096] 与以上描述的数据访问控制方法类似,本申请实施例还提供了相应的数据访问控制装置。

[0097] 图6示出了根据本申请一个实施例的数据访问控制装置600的结构框图。如图6所示,装置500可以包括接收模块610、标识信息确定模块620、访问请求量确定模块630以及控制模块640。

[0098] 具体而言,接收模块610可以用于接收针对数据库的访问请求,所述访问请求中包含数据对象信息。标识信息确定模块620可以用于基于所述数据对象信息,确定所述访问请求涉及的数据库中的数据记录的标识信息。访问请求量确定模块630可以用于确定所述标识信息对应的访问请求量。控制模块640可以用于根据所述访问请求量,确定是否拒绝所述访问请求。

[0099] 根据本申请的实施例,所述数据库可以包含一个或多个商品标识,每个商品标识包含一条或多条具有商品最小库存单元SKU标识的数据记录。

[0100] 根据本申请的实施例,所述数据对象信息中可以包括待访问数据对象的商品标识和商品最小库存单元SKU标识。

[0101] 根据本申请的实施例,所述标识信息确定模块620可以用于基于所述商品标识和商品最小库存单元SKU标识,确定所述访问请求涉及的数据库中的数据记录的标识信息。

[0102] 根据本申请的实施例,装置600还可以包括:预置模块,用于预先设置一记录表,所述记录表中记录有需要进行访问控制的数据对象的商品标识;判断和执行模块,用于在接收到所述针对数据库的访问请求后,判断所述访问请求中包含的数据对象信息中的商品标识是否在所述记录表中,如果是,则执行所述确定所述访问请求涉及的数据库中的数据记录的标识信息的步骤。

[0103] 根据本申请的实施例,访问请求量确定模块630可以包括缓存判断和确定子模块,用于:判断缓存中是否存在所述标识信息;如果存在,则获取所述标识信息对应的访问请求量;以及如果不存在,则在缓存中创建所述标识信息对应的访问请求量并对所述访问请求量赋予初始值。

[0104] 根据本申请的实施例,控制模块640可以包括阈值控制子模块,用于:判断所述标识信息对应的访问请求量是否超过第一阈值;以及如果超过,则拒绝所述访问请求,否则接受所述访问请求。

[0105] 根据本申请的一个实施例,装置600还可以包括:访问请求量更新模块,用于:当在根据所述访问请求量确定是否拒绝所述访问请求的步骤中接受所述访问请求时,执行以下步骤:判断缓存中是否存在所述标识信息;如果存在,则将所述标识信息对应的访问请求量加一;如果不存在,则在缓存中创建所述标识信息对应的访问请求量并赋予初始值。

[0106] 根据本申请的一个实施例,装置600还可以包括:队列管理模块,用于:判断缓存中是否存在所述标识信息;如果存在,则将该访问请求加入与所述标识信息对应的等待队列中,所述等待队列中记录有与标识信息对应的处于等待状态的访问请求;如果不存在,则在所述缓存中创建与所述标识信息对应的等待队列和执行队列,并将该访问请求加入该创建

的等待队列中,其中,所述执行队列中记录有与所述标识信息对应的从等待队列中获得的处于执行状态的访问请求。

[0107] 根据本申请的实施例,可以根据与所述标识信息对应的等待队列中的访问请求和/或执行队列的访问请求的数量,确定所述标识信息对应的访问请求量。

[0108] 根据本申请的实施例,队列管理模块还可以用于当所述执行队列中存在执行完毕的访问请求时,从执行队列中删除该执行完毕的访问请求。

[0109] 根据本申请的实施例,队列管理模块还可以用于:在将所述访问请求加入所述标识信息对应的等待队列中之后,判断与所述标识信息对应的执行队列中的访问请求的数量是否小于第二阈值;如果是,则从与所述标识信息对应的等待队列中获得访问请求并加入到与所述标识信息对应的执行队列中。

[0110] 根据本申请的实施例,队列管理模块还可以用于:当所述等待队列中的访问请求的等待时间超过第三阈值时,则从等待队列中删除该等待时间超过第三阈值的访问请求。

[0111] 以上描述的数据访问控制装置与之前描述的数据访问控制方法的处理是对应的,因此,关于其具体细节,可以参见之前描述的数据访问控制方法,这里不再赘述。

[0112] 本领域技术人员应明白,本申请的实施例可提供为方法、系统或计算机程序产品。因此,本申请可采用完全硬件实施例、完全软件实施例或结合软件和硬件方面的实施例的形式。而且,本申请可采用在一个或多个其中包含有计算机可用程序代码的计算机可用存储介质(包括但不限于磁盘存储器、CD-ROM、光学存储器等)上实施的计算机程序产品的形式。

[0113] 在一个典型的配置中,计算设备包括一个或多个处理器(CPU)、输入/输出接口、网络接口和内存。

[0114] 内存可能包括计算机可读介质中的非永久性存储器,随机存取存储器(RAM)和/或非易失性内存等形式,如只读存储器(ROM)或闪存(flash RAM)。内存是计算机可读介质的示例。

[0115] 计算机可读介质包括永久性和非永久性、可移动和非可移动媒体可以由任何方法或技术来实现信息存储。信息可以是计算机可读指令、数据结构、程序的模块或其他数据。计算机的存储介质的例子包括,但不限于相变内存(PRAM)、静态随机存取存储器(SRAM)、动态随机存取存储器(DRAM)、其他类型的随机存取存储器(RAM)、只读存储器(ROM)、电可擦除可编程只读存储器(EEPROM)、快闪记忆体或其他内存技术、只读光盘只读存储器(CD-ROM)、数字多功能光盘(DVD)或其他光学存储、磁盒式磁带,磁带磁盘存储或其他磁性存储设备或任何其他非传输介质,可用于存储可以被计算设备访问的信息。按照本文中的界定,计算机可读介质不包括暂存电脑可读媒体(transitory media),如调制的数据信号和载波。

[0116] 还需要说明的是,术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含,从而使得包括一系列要素的过程、方法、商品或者设备不仅包括那些要素,而且还包括没有明确列出的其他要素,或者是还包括为这种过程、方法、商品或者设备所固有的要素。在没有更多限制的情况下,由语句“包括一个……”限定的要素,并不排除在包括所述要素的过程、方法、商品或者设备中还存在另外的相同要素。

[0117] 以上所述仅为本申请的实施例而已,并不用于限制本申请。对于本领域技术人员来说,本申请可以有各种更改和变化。凡在本申请的精神和原理之内所作的任何修改、等同

替换、改进等,均应包含在本申请的权利要求范围之内。

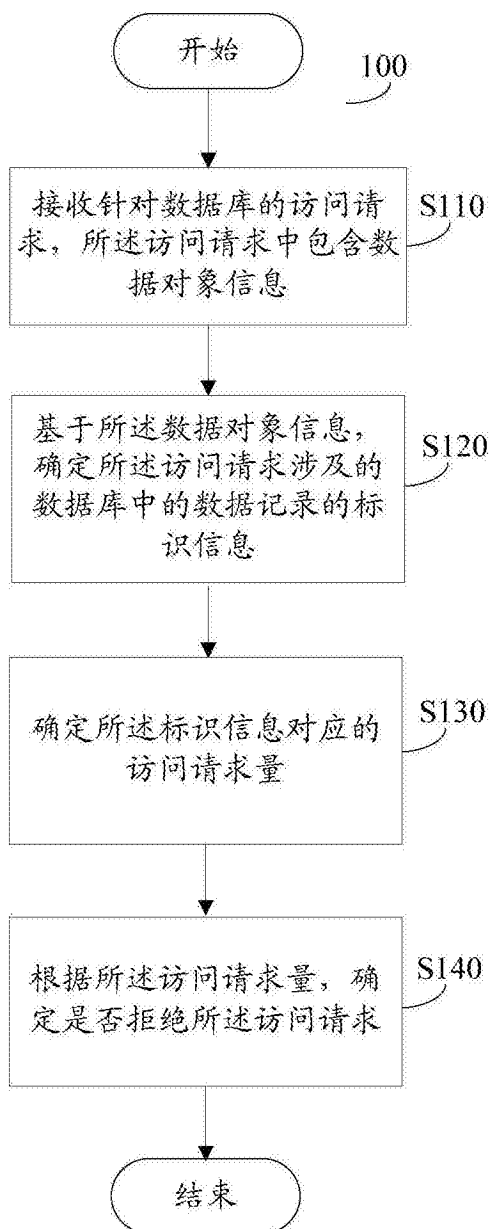


图1

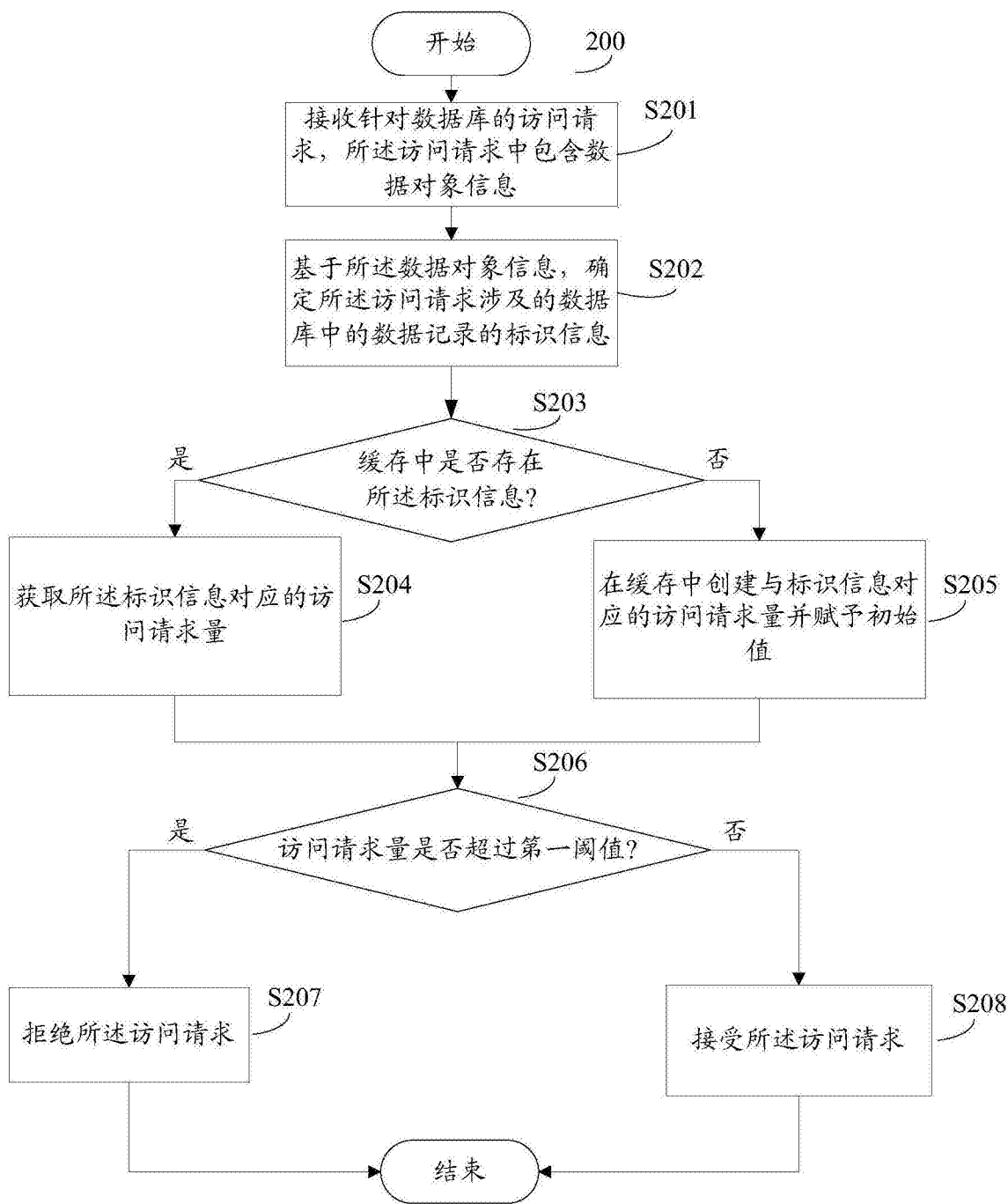


图2

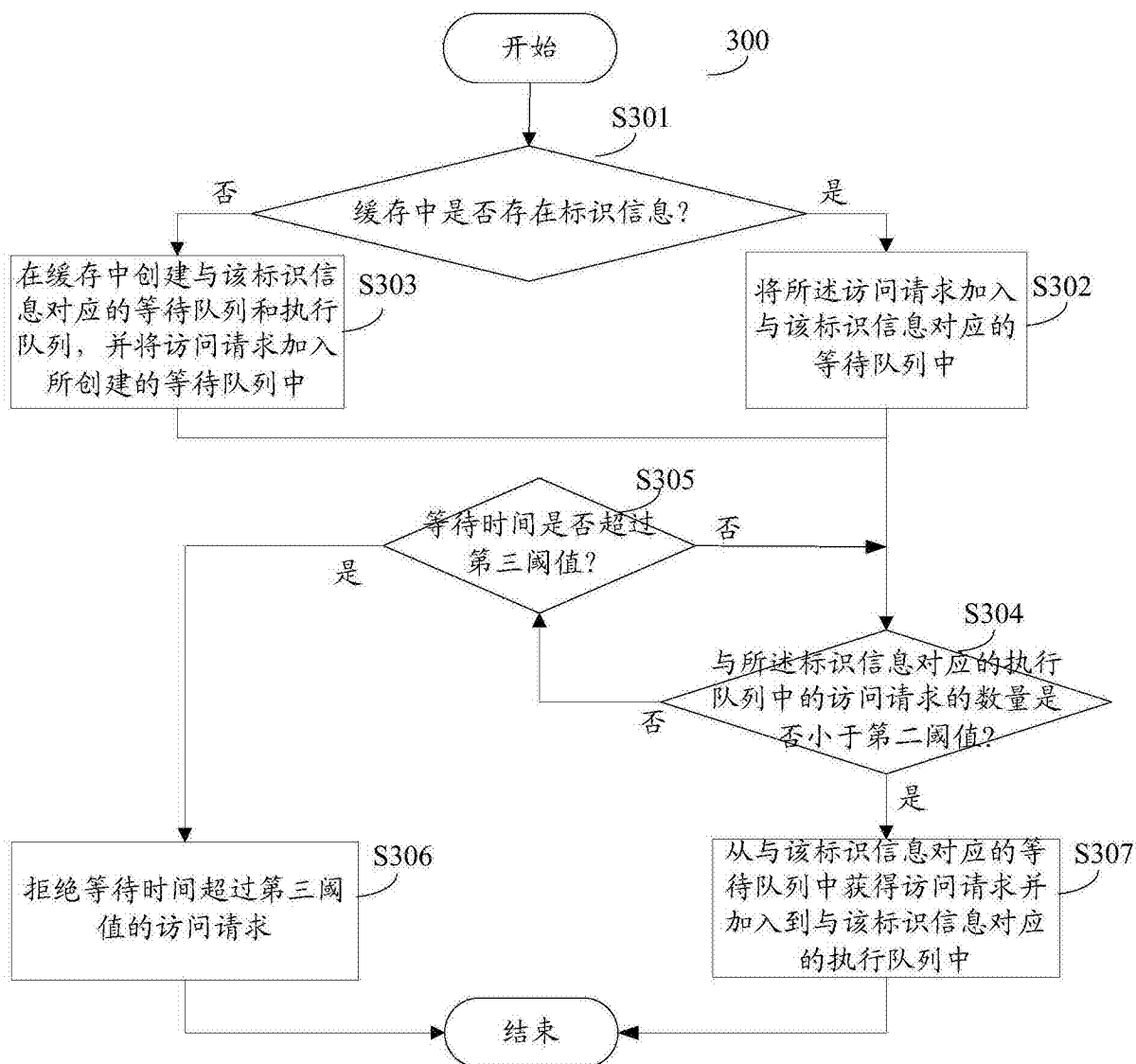


图3

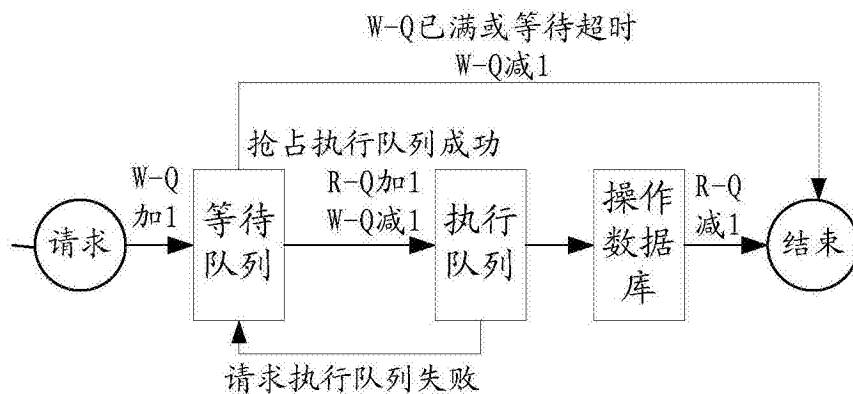


图4

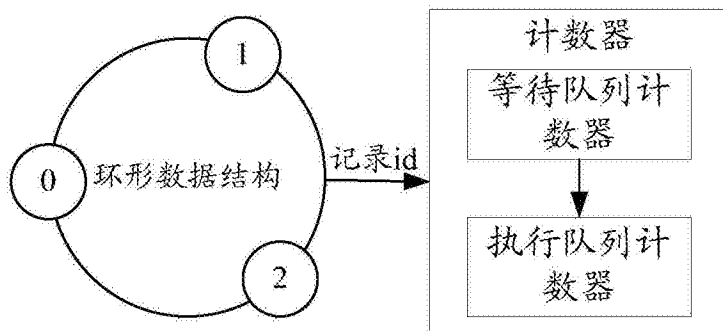


图5

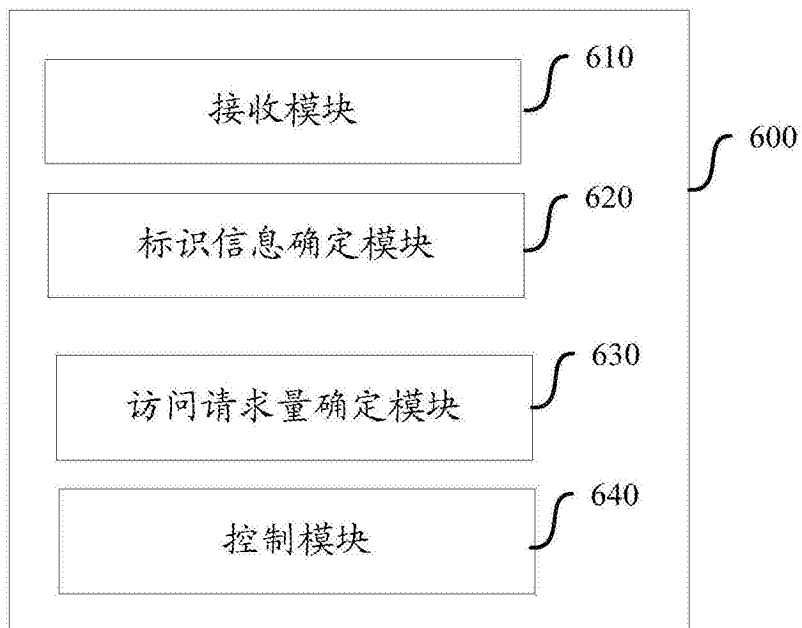


图6