



(12)发明专利

(10)授权公告号 CN 106096023 B

(45)授权公告日 2019.03.08

(21)申请号 201610473734.0

审查员 高沛沛

(22)申请日 2016.06.24

(65)同一申请的已公布的文献号

申请公布号 CN 106096023 A

(43)申请公布日 2016.11.09

(73)专利权人 腾讯科技(深圳)有限公司

地址 518000 广东省深圳市福田区振兴路

赛格科技园2栋东403室

(72)发明人 廖锡光

(74)专利代理机构 广州三环专利商标代理有限

公司 44202

代理人 贾允 肖丁

(51)Int.Cl.

G06F 16/22(2019.01)

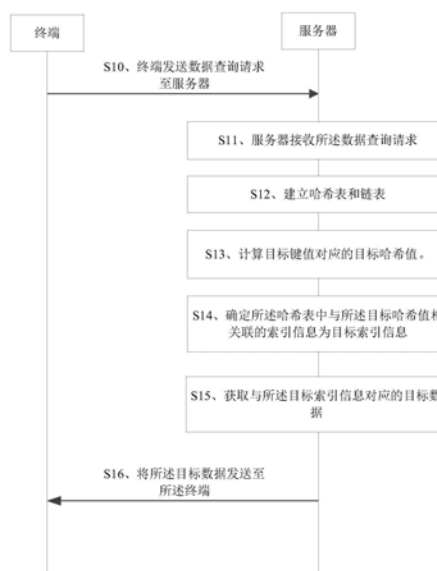
权利要求书2页 说明书9页 附图5页

(54)发明名称

数据读取方法、数据写入方法及数据服务器

(57)摘要

本发明实施例提供了一种数据读取方法,应用于数据服务器,其中,数据服务器存储有哈希表和链表,其中,哈希表用于存储键值的索引信息,链表用于存储与索引信息对应的多个区域的数据。该数据读取方法通过计算目标键值对应的目标哈希值;确定哈希表中与目标哈希值相关联的索引信息为目标索引信息;获取与目标索引信息对应的目标数据,并将目标数据发送至终端。可见,本方案采用哈希加链表的方式,简化了内存管理逻辑,读取速度快。



1. 一种数据读取方法,应用于数据服务器,其特征在于,所述数据服务器存储有哈希表和链表,所述哈希表用于存储键值的索引信息,所述键值的索引信息包括键值、所述键值的修改时间以及数据偏移指针,所述链表用于存储与所述索引信息对应的多个区域的数据,其中,一个键值对应多个区域,所述数据读取方法包括:

计算目标键值对应的目标哈希值;

获取所述目标哈希值在所述哈希表中每*i*行的位置*j*;

获取目标位置`hash[i][j]`的键值与所述目标键值相同的键值;

获取所述目标位置`hash[i][j]`的数据偏移指针的数值;

根据所述数据偏移指针的数值,确定所述键值对应的数据链表的首结点的位置;

根据预设的数据空间的长度以及所述首结点的位置,获取第一链表数据;

遍历所述第一链表数据,判断所述第一链表数据的区域名与目标数据是否一致,如果是,确定所述第一链表数据为目标数据。

2. 根据权利要求1所述的数据读取方法,其特征在于,

所述数据偏移指针为与所述键值对应的区域的数据在所述链表中的位置。

3. 一种数据写入方法,应用于数据服务器,其特征在于,所述数据服务器存储有哈希表和链表,所述哈希表用于存储键值的索引信息,所述键值的索引信息包括键值、所述键值的修改时间以及数据偏移指针,所述链表用于存储与所述索引信息对应的多个区域的数据,其中,一个键值对应多个区域,所述数据写入方法包括:

判断所述索引信息中是否包含目标键值,包括:计算目标键值对应的目标哈希值;获取所述目标哈希值在所述哈希表中每*i*行的位置*j*;获取目标位置`hash[i][j]`的键值与所述目标键值相同的键值;获取所述目标位置`hash[i][j]`的数据偏移指针的数值;根据所述数据偏移指针的数值,确定所述键值对应的数据链表的首结点的位置;根据预设的数据空间的长度以及所述首结点的位置,获取第一链表数据;遍历所述第一链表数据,判断所述第一链表数据的区域名与目标数据是否一致,如果是,确定所述第一链表数据为目标数据,

如果所述索引信息中包含目标键值,将所述目标键值对应的多个区域的数据进行存储;如果所述索引信息中没有包含目标键值,将所述索引信息中的键值赋值为目标键值,并对所述目标键值对应的多个区域的数据进行存储。

4. 根据权利要求3所述的数据写入方法,其特征在于,

所述将所述目标键值对应的多个区域的数据进行存储包括:

按照预设字段压缩所述多个区域的数据,并计算每个所述区域的数据的目标长度;

获取所述链表中长度大于所述目标长度的结点,将压缩后的数据存储到所述链表的数据空间。

5. 根据权利要求4所述的数据写入方法,其特征在于,当所述索引信息中不否包含所述目标键值,将所述索引信息中的键值赋值为目标键值之后,并对所述目标键值对应的多个区域的数据进行存储包括:

获取与所述目标键值相关联的目标哈希值在所述哈希表中每*i*行的位置*j*;

将所述键值对应的数据链表的首结点的位置赋值为目标位置`hash[i][j]`的数据偏移指针的数值。

6. 一种数据服务器,其特征在于,包括:

创建模块,用于存储哈希表和链表,所述哈希表用于存储键值的索引信息,所述键值的索引信息包括键值、所述键值的修改时间以及数据偏移指针,所述链表用于存储与所述索引信息对应的多个区域的数据,其中,一个键值对应多个区域;

第一计算模块,用于计算目标键值对应的目标哈希值;

第一确定模块,所述第一确定模块包括第一获取单元,所述第一获取单元包括:

第一获取子单元,用于获取所述目标哈希值在所述哈希表中每*i*行的位置*j*;

第二获取子单元,用于获取目标位置hash[*i*][*j*]的键值与所述目标键值相同的键值;

第一获取模块,所述第一获取模块包括第二获取单元,所述第二获取单元包括:

第三获取子单元,用于获取所述目标位置hash[*i*][*j*]的数据偏移指针的数值;

第一确定子单元,用于根据所述数据偏移指针的数值,确定所述键值对应的数据链表的首结点的位置;

第四获取子单元,用于根据预设的数据空间的长度以及所述首结点的位置,获取第一链表数据;

第一判断子单元,用于遍历所述第一链表数据,判断所述第一链表数据的区域名与目标数据是否一致,如果是,确定所述第一链表数据为目标数据。

7. 根据权利要求6所述的数据服务器,其特征在于,还包括,

第一判断模块,用于判断所述索引信息中是否包含目标键值,如果是,将所述目标键值对应的多个区域的数据进行存储;如果否,将所述索引信息中的键值赋值为目标键值,并对所述目标键值对应的多个区域的数据进行存储。

8. 根据权利要求7所述的数据服务器,其特征在于,所述第一判断模块包括:

压缩单元,用于按照预设字段压缩所述多个区域的数据,并计算每个所述区域的数据的目标长度;

第三获取单元,用于获取所述链表中长度大于所述目标长度的结点,将压缩后的数据存储在所述链表的数据空间。

9. 根据权利要求8所述的数据服务器,其特征在于,所述第一判断模块包括:

第四获取单元,用于获取与所述目标键值相关联的目标哈希值在所述哈希表中每*i*行的位置*j*;

赋值单元,用于将所述键值对应的数据链表的首结点的位置赋值为目标位置hash[*i*][*j*]的数据偏移指针的数值。

数据读取方法、数据写入方法及数据服务器

技术领域

[0001] 本发明涉及数据处理技术领域,更具体地说,涉及一种数据读取方法、数据写入方法及数据服务器。

背景技术

[0002] 随着大数据时代的到来,数据量不断增加,数据库的访问量也不断增大,用户对数据的访问速度要求也越来越高。通常,为了提高数据的访问速度,需要将要访问的数据写入数据服务器的内存,再进行访问。

[0003] 目前,采用两层哈希表的结构实现数据的读取和写入,其中,第一层哈希表存储Key的索引,第二层存储Field的索引,发明人发现,该数据存储方式,在数据写入时,要求每一个Key新建时,需要分配一个连续的内存空间,用于创建第二层哈希结构,使得内存管理逻辑非常繁重。而且,采用此方式,在同时读取多个Field值时,需要计算多次哈希,数据读取速度慢。

[0004] 因此,如何快速读取数据成为本领域技术人员亟待解决的问题。

发明内容

[0005] 有鉴于此,本发明实施例提供数据读取方法、数据写入方法及数据服务器,通过哈希加链表的方式,简化了内存管理逻辑,读取速度快。

[0006] 为实现上述目的,本发明实施例提供如下技术方案:

[0007] 一种数据读取方法,应用于数据服务器,

[0008] 数据服务器存储有哈希表和链表,所述哈希表用于存储键值的索引信息,所述链表用于存储与所述索引信息对应的多个区域的数据;

[0009] 计算目标键值对应的目标哈希值;

[0010] 确定所述哈希表中与所述目标哈希值相关联的索引信息为目标索引信息;

[0011] 获取与所述目标索引信息对应的目标数据。

[0012] 一种数据写入方法,应用于数据服务器,

[0013] 数据服务器存储有哈希表和链表,所述哈希表用于存储键值的索引信息,所述链表用于存储与所述索引信息对应的多个区域的数据;

[0014] 判断所述索引信息中是否包含目标键值,如果是,将所述目标键值对应的多个区域的数据进行存储;如果否,将所述索引信息中的键值赋值为目标键值,并对所述目标键值对应的多个区域的数据进行存储。

[0015] 一种数据服务器,包括:

[0016] 创建模块,用于存储哈希表和链表,所述哈希表用于存储键值的索引信息,所述链表用于存储与所述索引信息对应的多个区域的数据;

[0017] 第一计算模块,用于计算目标键值对应的目标哈希值;

[0018] 第一确定模块,用于确定所述哈希表中与所述目标哈希值相关联的索引信息为目

标索引信息；

[0019] 第一获取模块，用于获取与所述目标索引信息对应的目标数据。

[0020] 基于上述技术方案，本发明实施例提供了一种数据读取方法，应用于数据服务器，其中，数据服务器存储有哈希表和链表，其中，哈希表用于存储键值的索引信息，链表用于存储与索引信息对应的多个区域的数据。该数据读取方法通过计算目标键值对应的目标哈希值；确定哈希表中与目标哈希值相关联的索引信息为目标索引信息；获取与目标索引信息对应的目标数据，并将目标数据发送至终端。可见，本方案采用哈希加链表的方式，简化了内存管理逻辑，读取速度快。

附图说明

[0021] 为了更清楚地说明本发明实施例或现有技术中的技术方案，下面将对实施例或现有技术描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本发明的实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据提供的附图获得其他的附图。

[0022] 图1为本发明实施例提供的数据处理系统的结构框图；

[0023] 图2为本发明实施例提供的数据处理系统的信令流程图；

[0024] 图3为本发明实施例提供的数据处理系统的又一流程图；

[0025] 图4为本发明实施例提供的数据读取方法的流程图；

[0026] 图5为本发明实施例提供的数据读取方法的又一流程图；

[0027] 图6为本发明实施例提供的数据服务器的结构示意图；

[0028] 图7为本发明实施例提供的数据服务器的又一结构示意图；

[0029] 图8为本发明实施例提供的服务器的硬件结构框图。

具体实施方式

[0030] 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、完整地描述，显然，所描述的实施例仅仅是本发明一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都对应本发明保护的范围。

[0031] 本发明实施例提供了一种数据读取方法，应用于数据服务器，其中，数据服务器存储有哈希表和链表，其中，哈希表用于存储键值的索引信息，链表用于存储与索引信息对应的多个区域的数据。该数据读取方法通过计算目标键值对应的目标哈希值；确定哈希表中与目标哈希值相关联的索引信息为目标索引信息；获取与目标索引信息对应的目标数据，并将目标数据发送至终端。可见，本方案采用哈希加链表的方式，简化了内存管理逻辑，读取速度快。

[0032] 图1为本发明实施例提供的数据处理系统的结构框图，本发明实施例提供的数据读取方法以及数据写入方法可基于图1所示系统实现，参照图1，本发明实施例提供的数据处理系统可以包括：服务器10，至少一个终端20；

[0033] 服务器10可以为网络侧设置的用于数据处理的设备，服务器10可以为单台服务器，也可以为由多台服务器组成的服务器群组；

[0034] 至少一个终端20为用户侧的用于提交数据查询请求的设备,如手机、平板电脑、笔记本电脑等。

[0035] 基于图1所示系统,图2示出了本发明实施例提供的数据处理系统的信令流程图,结合图1和图2所示,该流程可以包括:

[0036] S10、终端发送数据查询请求至服务器。

[0037] S11、服务器接收所述数据查询请求。

[0038] S12、建立哈希表和链表。

[0039] 其中,所述哈希表用于存储键值Key的索引信息,所述键值的索引信息可以包括键值、所述键值的修改时间以及数据偏移指针,所述数据偏移指针为与所述键值对应的区域的数据在所述链表中的位置。

[0040] 具体的,哈希表可以使用二维数组保存,如一个桶长为m、桶深为n的哈希表,用形如hash[n][m]的二维数组保存。即该二维数组的结点定义为:

```
[0041] struct THashElem{  
[0042]     char key[128]; //Key值  
[0043]     uint32_t modify_time; //修改时间  
[0044]     uint32_t position; //数据在Field数据链表的位置  
[0045] }。
```

[0046] 另外,还会保存一个长度为n的递减素数数组mod[n],其中,mod[0]为小于m的最大素数。具体的,mod是用来计算每个哈希值所映射的位置。例如某一个Key计算出来的哈希值是k=18932,哈希表第一行的长度是mod[0]=9973,则k对mod[0]取模得到的值8959为k在第一行的位置。

[0047] 又由于哈希表每一行的长度必须要不一样,以保证当k在上一行冲突时,在下一行可以映射到不同的位置,因此,mod[n]为递减数组。

[0048] 而链表采用一维数组,链表用于存储与所述索引信息对应的多个区域Field的数据。

[0049] 数据中每个Field根据FieldName排序,并以FieldLen/FieldName/ValueLen/ValueType/ValueBuf方式紧缩存储。其中,FieldLen为Field的长度,在本实施例中,该Field的长度为4字节;FieldName:为Field的名称;ValueLen为Field对应值的长度,也是4字节;ValueType为Field对应值的类型,如整数、字符串、数组等,本实施例中ValueType也选用4字节;ValueBuf为Field对应值的具体空间。具体的,数组的下标代表链表的位置。即上述哈希表对结点的定义里的THashElem.position就是此链表数组的下标值。

[0050] 数组中每个结点定义为:

```
[0051] struct TLinkElem{  
[0052]     uint32_t next_positon;  
[0053]     char buffer[512];  
[0054] };
```

[0055] 每个Key对应的数据部分,可能多个结点链接成的数据链表组成。其中每个数据链表包括一个表头部分,定义如下:

```
[0056] struct TLinkBufHead{
```

[0057] char key[128]; //Key的值

[0058] uint32_t crc; //数据空间的crc校验值

[0059] uint32_t buffer_len; //数据空间长度

[0060] };

[0061] 数据的每个Field均以FieldLen/FieldName/ValueLen/ValueType/ValueBuf的方式紧缩存储,并且按FieldName的值从小到大排序。

[0062] S13、计算目标键值对应的目标哈希值。

[0063] 此处,可以通过预设算法计算Key值对应的哈希值,例如采用FNV算法,当然,采用其他算法也可以。

[0064] S14、确定所述哈希表中与所述目标哈希值相关联的索引信息为目标索引信息。

[0065] S15、获取与所述目标索引信息对应的目标数据。

[0066] 需要说明的是,此处可以通过获取所述索引信息中根据所述目标哈希值得到的键值。再获取与所述键值对应的数据偏移指针,根据所述数据偏移指针,确定所述目标数据。

[0067] 其中,服务器在获取所述索引信息中根据所述目标哈希值得到的键值时,可以通过在获取所述目标哈希值在所述哈希表中每i行的位置j后,获取目标位置hash[i][j]的键值与所述目标键值相同的键值。

[0068] 服务器在获取与所述键值对应的数据偏移指针,根据所述数据偏移指针,确定所述目标数据时,可以通过获取所述目标位置hash[i][j]的数据偏移指针的数值;根据所述数据偏移指针的数值,确定所述键值对应的数据链表的首结点的位置;再根据预设的数据空间的长度以及所述首结点的位置,获取第一链表数据;最后遍历所述第一链表数据,判断所述第一链表数据的区域名与目标数据是否一致,如果是,确定所述第一链表数据为目标数据。

[0069] S16、将所述目标数据发送至所述终端。

[0070] 现结合一具体实例,对本方案提供的数据处理系统在实现数据读取的工作原理进行说明,例如,现在需要查询一个Key值为k,Field为f1、f2、f3的值(其中f1、f2、f3按字母由小到大排序),操作如下:

[0071] 1、先用FNV算法计算k对应的哈希值,假设哈希值为h。

[0072] 2、对哈希表每一行i(i从0开始),计算h在该行的位置,假设值为j。

[0073] 3、判断hash[i][j]的key字段与k是否一致。是则继续,否则增加i的值,返回操作步骤2。

[0074] 4、取出hash[i][j]的position字段的值,假设为p。则link[p]为k对应的数据链表首结点的位置。

[0075] 5、从link[p]中取出buffer部分,解析出TLinkBufHead,并根据buffer_len获取整个链表的数据部分。

[0076] 6、依次遍历整个数据部分,先取出4个字节作为FieldLen的值,取出FieldLen个字节作为FieldName的值。判断FieldName与f1是否一致,是则继续,否则重复执行本步骤,直至找到FieldName的值与f1相同。

[0077] 7、取出4字节作为ValueLen的值,取出ValueLen个字节作为ValueType和ValueBuf的值,此值即为f1的值。

[0078] 继续步骤7,查询f2、f3的值。

[0079] 上面介绍了本数据处理系统读取数据的处理流程,本实施例还提供了一种数据写入的处理流程,如图3所示,具体为:

[0080] 步骤S21、服务器判断所述索引信息中是否包含目标键值,如果是,将所述目标键值对应的多个区域的数据进行存储;如果否,将所述索引信息中的键值赋值为目标键值,并对所述目标键值对应的多个区域的数据进行存储。

[0081] 其中,服务器将所述目标键值对应的多个区域的数据进行存储,具体可以按照预设字段压缩所述多个区域的数据,并计算每个所述区域的数据的目标长度。之后获取所述链表中长度大于所述目标长度的结点,将压缩后的数据存储在所述链表的数据空间。

[0082] 具体的,该服务器还可以获取与所述目标键值相关联的目标哈希值在所述哈希表中每i行的位置j;

[0083] 将所述键值对应的数据链表的首结点的位置赋值为目标位置hash[i][j]的数据偏移指针的数值。

[0084] 现结合一具体实例,对本数据写入流程进行说明,如下:

[0085] 假设哈希表使用名为hash的二维数组保存,Field数据链表使用名为link的一维数组保存。

[0086] 现在需要写入一个Key值为k,Field为f1、f2、f3的值(其中f1、f2、f3按字母由小到大排序),操作如下:

[0087] 1、判断k在hash中是否存在。

[0088] 其具体实现步骤参见上述数据读取流程中的步骤1-4,如果存在,是则将f1、f2、f3按FieldLen/FieldName/ValueLen/ValueType/ValueBuf紧缩存储到临时空间t内,并计算其总大小,假设为l。

[0089] 如果不存在,则判断hash[i][j]的key值是否为空,是则将hash[i][j]的key值赋值为k,position值赋值为0;否则查找下一个空结点。

[0090] 从link中的空闲结点中,取出总大小大于l的结点,并组成数据链表。并将临时空间t的数据保存在数据链表的buf空间内。同时设置链表的表头TLinkBufHead的key字段为k,crc字段赋值为根据t计算的crc校验值,buffer_len字段赋值为l。

[0091] 2、将hash[i][j]的position字段赋值为数据链表首结点的下标值。如果赋值之前,position字段为0,则返回成功,如果否,则删除hash[i][j]原position字段对应的link的数据链表的内容。

[0092] 采用本实施例提供的数据处理系统,能够快速的实现数据读取或写入,尤其是针对一个Key对应多个Field的情况,如,缓存视频数据时,Key可以是此影片ID,Field可以包括影片的标题、海报、剧集列表、明星、剧照等。对于影片字段名这类数据,每一个Key需要缓存的Field字段相似的场景,Field数据使用链表的方式代替哈希表,可以减少查询的次数。

[0093] 例如,Key中需要缓存的总Field个数为20,需要查询的Field个数为15。用本方案,即采用链表的方式,查询次数为20,而现有技术中,采用哈希表的方式查询的次数为 $15 * \ln(20) = 34$ 。相比可知,本方案采用哈希加链表的方式,简化了内存管理逻辑,读取速度快。

[0094] 下面以服务器的角度对本发明实施例提供的数据读取方法进行介绍,下文描述的

数据读取方法可与上文描述的信令流程内容相互参照。

[0095] 图4为本发明实施例提供的数据读取方法的流程图,该方法可应用于服务器,包括:

[0096] 步骤S101、计算目标键值对应的目标哈希值;

[0097] 步骤S102、确定所述哈希表中与所述目标哈希值相关联的索引信息为目标索引信息;

[0098] 步骤S103、获取与所述目标索引信息对应的目标数据。

[0099] 需要说明的是,本数据读取方法基于数据服务器的硬件架构实现,其中,数据服务器存储有哈希表和链表,哈希表用于存储键值的索引信息,链表用于存储与所述索引信息对应的多个区域的数据。当然,该哈希表和链表可以是在执行上述数据读取方法之前进行构建,并存储于数据服务器中。

[0100] 优选的,所述键值的索引信息包括键值、所述键值的修改时间以及数据偏移指针,所述数据偏移指针为与所述键值对应的区域的数据在所述链表中的位置;

[0101] 所述确定所述哈希表中与所述目标哈希值相关联的索引信息为目标索引信息包括:

[0102] 获取所述索引信息中根据所述目标哈希值得到的键值;

[0103] 所述获取与所述目标索引信息对应的目标数据包括:

[0104] 获取与所述键值对应的数据偏移指针,根据所述数据偏移指针,确定所述目标数据。

[0105] 优选的,所述获取所述索引信息中根据所述目标哈希值得到的键值包括:

[0106] 获取所述目标哈希值在所述哈希表中每*i*行的位置*j*;

[0107] 获取目标位置 $hash[i][j]$ 的键值与所述目标键值相同的键值。

[0108] 优选的,所述获取与所述键值对应的数据偏移指针,根据所述数据偏移指针,确定所述目标数据包括:

[0109] 获取所述目标位置 $hash[i][j]$ 的数据偏移指针的数值;

[0110] 根据所述数据偏移指针的数值,确定所述键值对应的数据链表的首结点的位置;

[0111] 根据预设的数据空间的长度以及所述首结点的位置,获取第一链表数据;

[0112] 遍历所述第一链表数据,判断所述第一链表数据的区域名与目标数据是否一致,如果是,确定所述第一链表数据为目标数据。

[0113] 请参阅图5,为本实施例提供的一种数据写入方法的流程图,该数据写入方法应用于数据服务器,具体包括:

[0114] 步骤S201、判断所述索引信息中是否包含目标键值,如果是,将所述目标键值对应的多个区域的数据进行存储;如果否,将所述索引信息中的键值赋值为目标键值,并对所述目标键值对应的多个区域的数据进行存储。

[0115] 优选的,所述将所述目标键值对应的多个区域的数据进行存储包括:

[0116] 按照预设字段压缩所述多个区域的数据,并计算每个所述区域的数据的目标长度;

[0117] 获取所述链表中长度大于所述目标长度的结点,将压缩后的数据存储到所述链表的数据空间。

[0118] 优选的,当所述索引信息中不否包含所述目标键值,将所述索引信息中的键值赋值为目标键值之后,所述并对所述目标键值对应的多个区域的数据进行存储包括:

[0119] 获取与所述目标键值相关联的目标哈希值在所述哈希表中每*i*行的位置*j*;

[0120] 将所述键值对应的数据链表的首结点的位置赋值为目标位置hash[*i*][*j*]的数据偏移指针的数值。

[0121] 其方法实施例的工作原理可参见上述系统实施例。

[0122] 下面对本发明实施例提供的服务器进行介绍,下文描述的服务器可与上文以服务器角度描述的数据读取方法和数据写入系统相互对应参照。

[0123] 图6为本发明实施例提供的服务器的结构框图,参照图6,该服务器可以包括:

[0124] 第一计算模块20,用于计算目标键值对应的目标哈希值;

[0125] 第一确定模块30,用于确定所述哈希表中与所述目标哈希值相关联的索引信息为目标索引信息;

[0126] 第一获取模块40,用于获取与所述目标索引信息对应的目标数据。

[0127] 可选的,还可以包括创建模块,用于存储哈希表和链表,所述哈希表用于存储键值的索引信息,所述链表用于存储与所述索引信息对应的多个区域的数据。

[0128] 可选的,如图7所示,所述第一确定模块30包括第一获取单元301,用于获取所述索引信息中根据所述目标哈希值得到的键值;

[0129] 第一获取模块40包括第二获取单元401,用于获取与所述键值对应的数据偏移指针,根据所述数据偏移指针,确定所述目标数据。

[0130] 有选的,所述第一获取单元包括:

[0131] 第一获取子单元,用于获取所述目标哈希值在所述哈希表中每*i*行的位置*j*;

[0132] 第二获取子单元,用于获取目标位置hash[*i*][*j*]的键值与所述目标键值相同的键值。

[0133] 优选的,所述第二获取单元包括:

[0134] 第三获取子单元,用于获取所述目标位置hash[*i*][*j*]的数据偏移指针的数值;

[0135] 第一确定子单元,用于根据所述数据偏移指针的数值,确定所述键值对应的数据链表的首结点的位置;

[0136] 第四获取子单元,用于根据预设的数据空间的长度以及所述首结点的位置,获取第一链表数据;

[0137] 第一判断子单元,用于遍历所述第一链表数据,判断所述第一链表数据的区域名与目标数据是否一致,如果是,确定所述第一链表数据为目标数据。

[0138] 优选的,还包括,

[0139] 第一判断模块,用于判断所述索引信息中是否包含目标键值,如果是,将所述目标键值对应的多个区域的数据进行存储;如果否,将所述索引信息中的键值赋值为目标键值,并对所述目标键值对应的多个区域的数据进行存储。

[0140] 优选的,所述第一判断模块包括:

[0141] 压缩单元,用于按照预设字段压缩所述多个区域的数据,并计算每个所述区域的数据的目标长度;

[0142] 第三获取单元,用于获取所述链表中长度大于所述目标长度的结点,将压缩后的

数据存储到所述链表的数据空间。

[0143] 优选的,所述第一判断模块包括:

[0144] 第四获取单元,用于获取与所述目标键值相关联的目标哈希值在所述哈希表中每*i*行的位置*j*;

[0145] 赋值单元,用于将所述键值对应的数据链表的首结点的位置赋值为目标位置hash [*i*][*j*]的数据偏移指针的数值。

[0146] 本发明实施例提供的服务器的硬件结构可如图8所示,包括:处理器1,通信接口2,存储器3和通信总线4;

[0147] 其中处理器1、通信接口2、存储器3通过通信总线4完成相互间的通信;

[0148] 可选的,通信接口2可以为通信模块的接口,如GSM模块的接口;

[0149] 处理器1,用于执行程序;

[0150] 存储器3,用于存放程序;

[0151] 程序可以包括程序代码,所述程序代码包括计算机操作指令。

[0152] 处理器1可能是一个中央处理器CPU,或者是特定集成电路ASIC (Application Specific Integrated Circuit),或者是被配置成实施本发明实施例的一个或多个集成电路。

[0153] 存储器3可能包含高速RAM存储器,也可能还包括非易失性存储器(non-volatile memory),例如至少一个磁盘存储器。

[0154] 其中,程序可具体用于:

[0155] 存储哈希表和链表,所述哈希表用于存储键值的索引信息,所述链表用于存储与所述索引信息对应的多个区域的数据;

[0156] 计算目标键值对应的目标哈希值;

[0157] 确定所述哈希表中与所述目标哈希值相关联的索引信息为目标索引信息;

[0158] 获取与所述目标索引信息对应的目标数据。

[0159] 或,

[0160] 存储哈希表和链表,所述哈希表用于存储键值的索引信息,所述链表用于存储与所述索引信息对应的多个区域的数据;

[0161] 判断所述索引信息中是否包含目标键值,如果是,将所述目标键值对应的多个区域的数据进行存储;如果否,将所述索引信息中的键值赋值为目标键值,并对所述目标键值对应的多个区域的数据进行存储。

[0162] 综上所述,本发明实施例提供了一种数据读取方法,应用于数据服务器,其中,数据服务器存储有哈希表和链表,其中,哈希表用于存储键值的索引信息,链表用于存储与索引信息对应的多个区域的数据。该数据读取方法通过计算目标键值对应的目标哈希值;确定哈希表中与目标哈希值相关联的索引信息为目标索引信息;获取与目标索引信息对应的目标数据,并将目标数据发送至终端。可见,本方案采用哈希加链表的方式,简化了内存管理逻辑,读取速度快。

[0163] 本说明书中各个实施例采用递进的方式描述,每个实施例重点说明的都是与其他实施例的不同之处,各个实施例之间相同相似部分互相参见即可。对于实施例公开的装置而言,由于其与实施例公开的方法相对应,所以描述的比较简单,相关之处参见方法部分说

明即可。

[0164] 专业人员还可以进一步意识到,结合本文中所公开的实施例描述的各示例的单元及算法步骤,能够以电子硬件、计算机软件或者二者的结合来实现,为了清楚地说明硬件和软件的可互换性,在上述说明中已经按照功能一般性地描述了各示例的组成及步骤。这些功能究竟以硬件还是软件方式来执行,取决于技术方案的特定应用和设计约束条件。专业技术人员可以对每个特定的应用来使用不同方法来实现所描述的功能,但是这种实现不应认为超出本发明的范围。

[0165] 结合本文中所公开的实施例描述的方法或算法的步骤可以直接用硬件、处理器执行的软件模块,或者二者的结合来实施。软件模块可以置于随机存储器(RAM)、内存、只读存储器(ROM)、电可编程ROM、电可擦除可编程ROM、寄存器、硬盘、可移动磁盘、CD-ROM、或技术领域内所公知的任意其它形式的存储介质中。

[0166] 对所公开的实施例的上述说明,使本领域专业技术人员能够实现或使用本发明。对这些实施例的多种修改对本领域的专业技术人员来说将是显而易见的,本文中所定义的一般原理可以在不脱离本发明的精神或范围的情况下,在其它实施例中实现。因此,本发明将不会被限制于本文所示的这些实施例,而是要符合与本文所公开的原理和新颖特点相一致的最宽的范围。

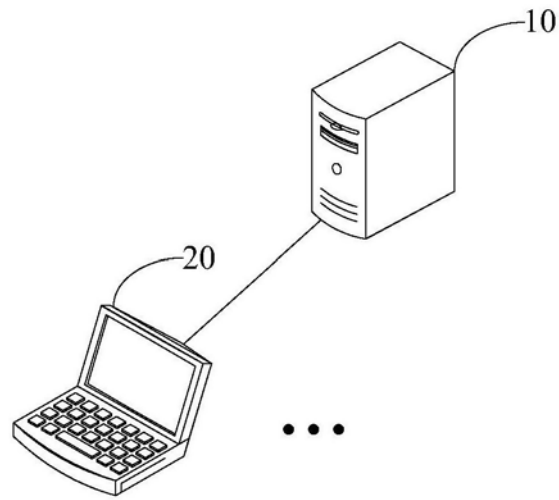


图1

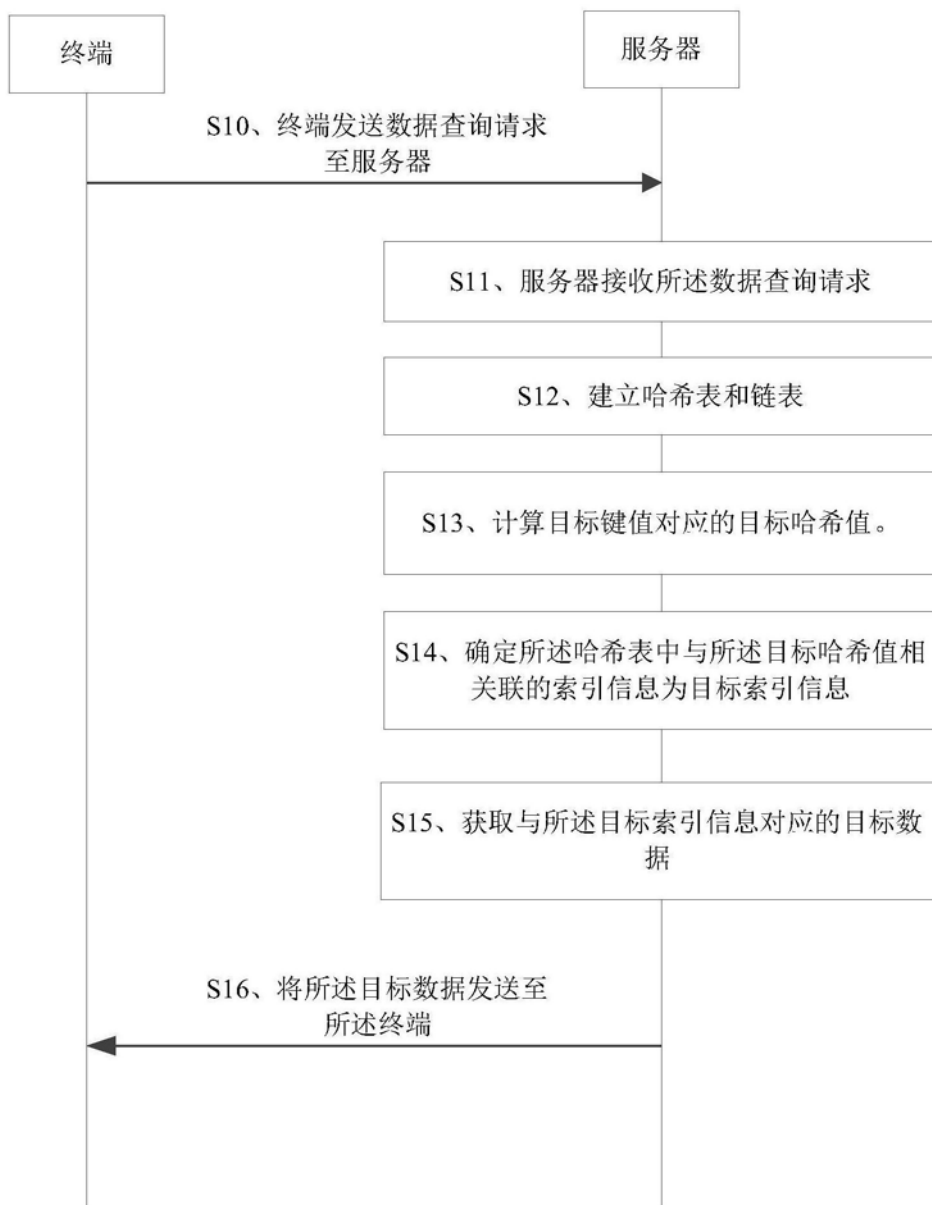


图2

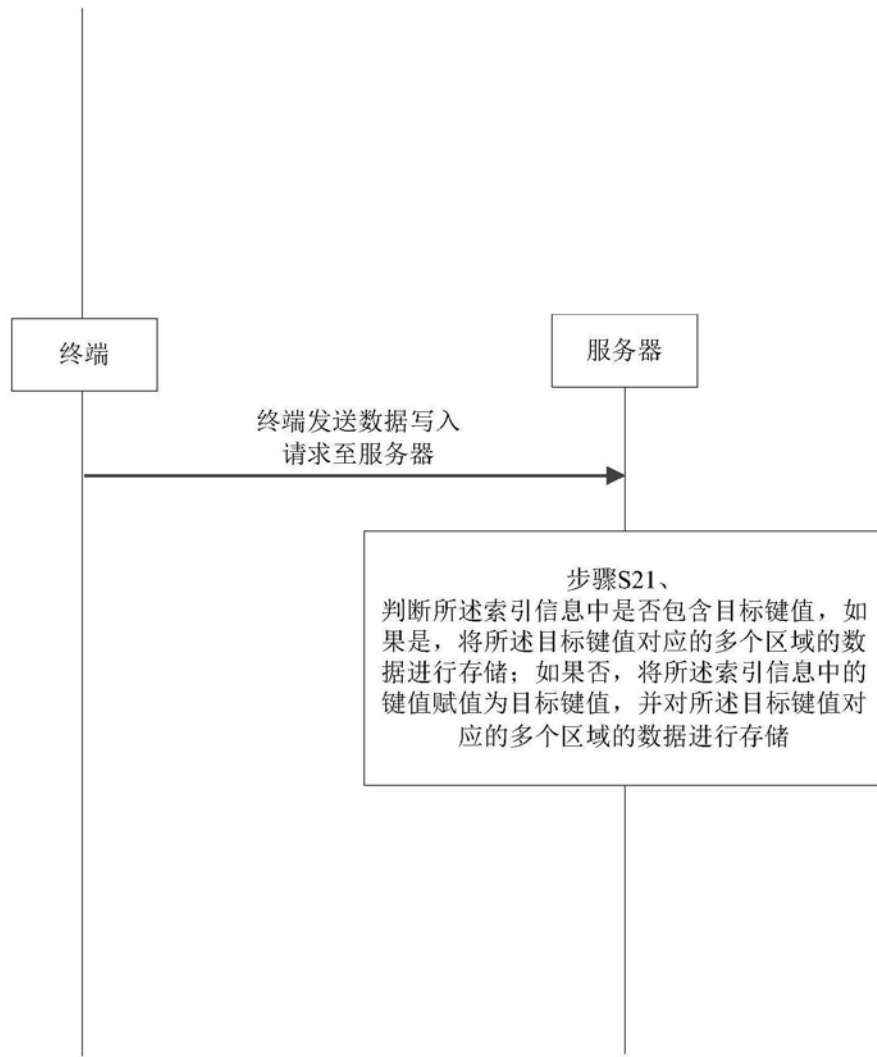


图3

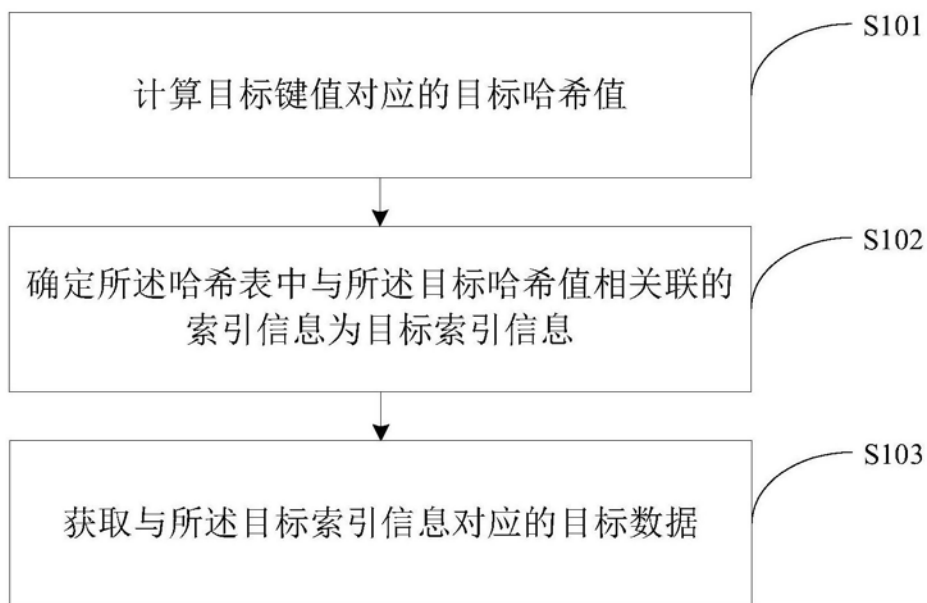


图4

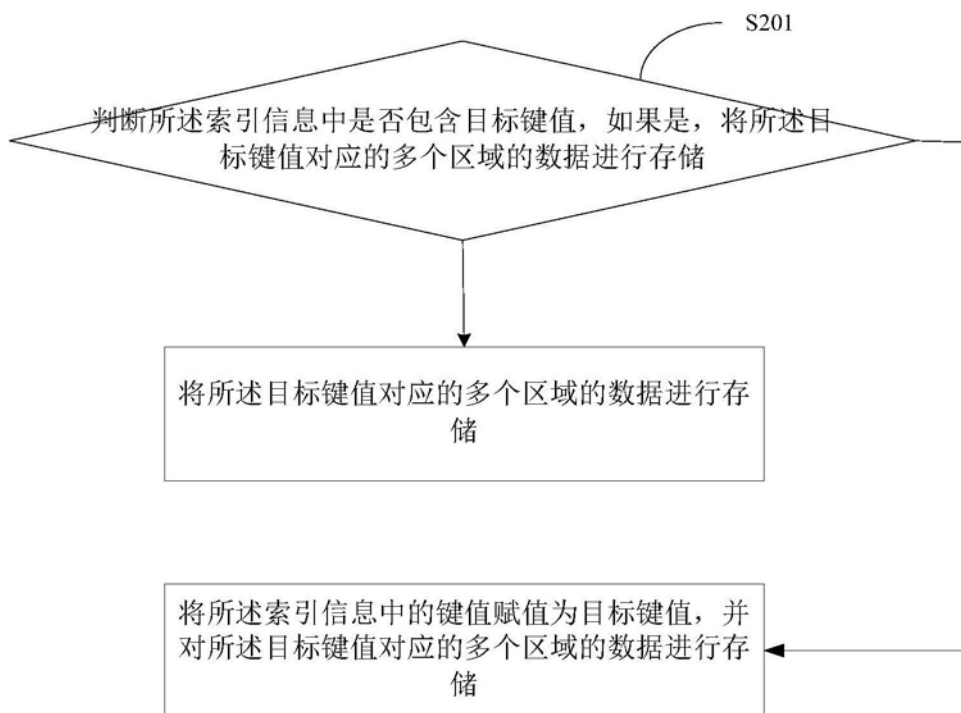


图5

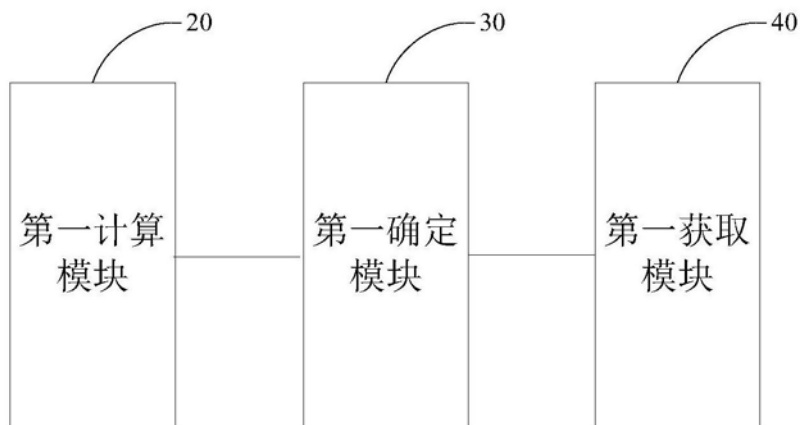


图6

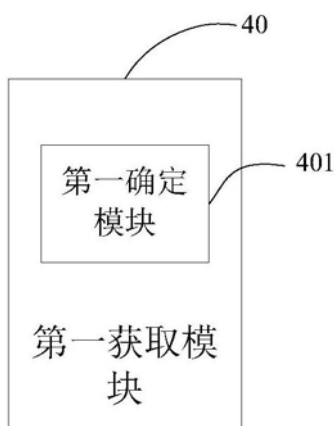


图7

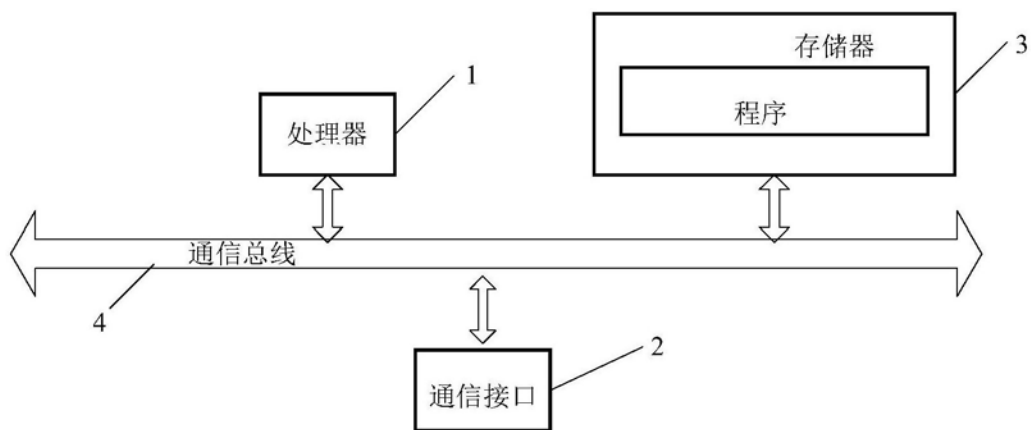


图8