



(12)发明专利

(10)授权公告号 CN 104750720 B

(45)授权公告日 2018.04.27

(21)申请号 201310743108.5

G06F 9/38(2006.01)

(22)申请日 2013.12.30

(56)对比文件

(65)同一申请的已公布的文献号

申请公布号 CN 104750720 A

CN 102567415 A, 2012.07.11,

CN 103095686 A, 2013.05.08,

US 2012166408 A1, 2012.06.28,

CN 1437727 A, 2003.08.20,

(43)申请公布日 2015.07.01

(73)专利权人 中国银联股份有限公司

地址 200135 上海市浦东新区含笑路36号

银联大厦

审查员 胡璇

(72)发明人 胡新松 孙战平 夏智 佟志臣

张兴尧 易师

(74)专利代理机构 中国专利代理(香港)有限公

司 72001

代理人 李湘 汤春龙

(51)Int. Cl.

G06F 17/30(2006.01)

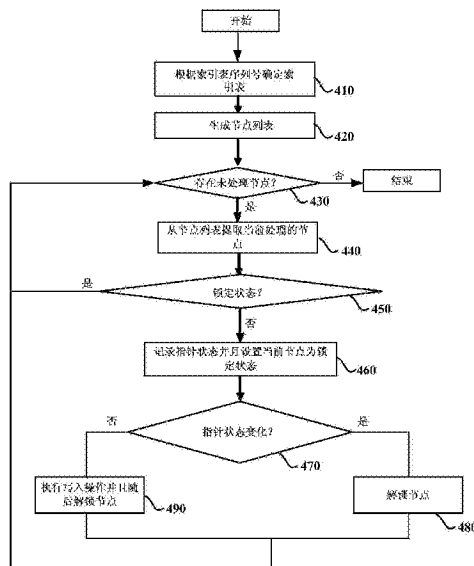
权利要求书2页 说明书5页 附图4页

(54)发明名称

多线程并发访问环境下高性能数据处理的实现

(57)摘要

本发明涉及数据库技术,特别涉及一种在多个线程并发访问环境下实现高性能数据处理的方法和实现该方法的数据库管理系统。按照本发明一个实施例的数据库管理系统包括一种数据库管理系统包括:多个数据源;与多个数据源相连的数据管理器,其包括:中央处理器;高速缓存;I/O设备,其中,所述中央处理器被配置为使得多个线程的每一个按照下列方式在高速缓存中写入数据:确定与写入的数据在高速缓存中相关联的存储位置;对于每个写入的数据,执行下列步骤:如果与该写入的数据相关联的存储位置处于解锁状态,则记录指针的状态;将该相关联的存储位置置于锁定状态;如果所述指针的状态未发生变化,则对该相关联的存储位置执行写入操作,否则,则不执行写入操作并将该相关联的存储位置置于解锁状态;在完成所述写入操作之后,将该相关联的存储位置置于锁定状态。



1. 一种在多个线程并发访问环境下的数据处理方法,其特征在于,对于所述多个线程的每一个,其按照下列方式在高速缓存中写入数据:

确定与写入的数据在高速缓存中相关联的存储位置,所述存储位置包含锁定状态和解锁状态,以分别表示存在某一线程正对其执行写入操作和未有任何线程对其执行写入操作;

对于每个写入的数据,执行下列步骤:

如果与该写入的数据相关联的存储位置处于解锁状态,则记录指针的状态;

将该相关联的存储位置置于锁定状态;

如果所述指针的状态未发生变化,则对该相关联的存储位置执行写入操作并且随后将该相关联的存储位置置于解锁状态,否则,则不执行写入操作并将该相关联的存储位置置于解锁状态;

其中,所述指针的状态指示最近一次被执行写入操作的存储位置。

2. 如权利要求1所述的数据处理方法,其中,所述写入的数据存储在所述高速缓存内的同步哈希表内,所述同步哈希表的每个节点包括相关联的数据、该相关联的数据的索引值以及表示解锁状态和锁定状态的标识。

3. 如权利要求1所述的数据处理方法,其中,对于在所述高速缓存中写入的数据,通过定期执行下列步骤,将其写入数据源中:

对所述高速缓存中写入的数据按照更新时间的顺序排序;

将前次执行写入数据源操作结束后发生更新的数据写入数据源中。

4. 如权利要求3所述的数据处理方法,其中,所述数据源为异构数据源。

5. 一种数据库管理系统,其特征在于,包括

多个数据源;

与多个数据源相连的数据管理器,其包括:

中央处理器;

高速缓存;

I/O设备,

其中,所述中央处理器被配置为使得多个线程的每一个按照下列方式在高速缓存中写入数据:

确定与写入的数据在高速缓存中相关联的存储位置,所述存储位置包含锁定状态和解锁状态,以分别表示存在某一线程正对其执行写入操作和未有任何线程对其执行写入操作;

对于每个写入的数据,执行下列步骤:

如果与该写入的数据相关联的存储位置处于解锁状态,则记录指针的状态;

将该相关联的存储位置置于锁定状态;

如果所述指针的状态未发生变化,则对该相关联的存储位置执行写入操作并且随后将该相关联的存储位置置于解锁状态,否则,则不执行写入操作并将该相关联的存储位置置于解锁状态;

其中,所述指针的状态指示最近一次被执行写入操作的存储位置。

6. 如权利要求5所述的数据库管理系统,其中,所述写入的数据存储在所述高速缓存内

的同步哈希表内,所述同步哈希表的每个节点包括相关联的数据、该相关联的数据的索引值以及表示解锁状态和锁定状态的标识。

7.如权利要求5所述的数据库管理系统,其中,所述中央处理器被配置为对于在所述高速缓存中写入的数据,通过定期执行下列步骤,将其写入数据源中:

对所述高速缓存中写入的数据按照更新时间的顺序排序;

指示所述I/O设备将前次执行写入数据源操作结束后发生更新的数据写入所述数据源中。

8.如权利要求7所述的数据库管理系统,其中,所述数据源为异构数据源。

多线程并发访问环境下高性能数据处理的实现

发明领域

[0001] 本发明涉及数据库技术,特别涉及一种在多线程并发访问环境下实现高性能数据处理的方法和实现该方法的数据库管理系统。

背景技术

[0002] 现代信息社会中,信息的重要性不言而喻。信息处理中非常重要的部分是数据库。数据集市指的是一种专项数据集合,这些数据取自一个或多个联机交易系统以及其他关联系统的数据库、数据仓库等各种数据源,并且按照特定的业务需求进行处理、加工,从而用来分析相关专门业务问题或功能目标。

[0003] 交易数据的实时性获取和实时数据分析对于互联网业务的运营和风险控制具有极其重要的意义,但是在对多个异构数据源到目标数据的高性能抽取存时存在下列技术难点:

[0004] 1.不能影响联机交易的系统的性能和稳定性;

[0005] 2.在数据源具有异构性时,如何保持数据的完整性和数据库事务的ACID(原子性、一致性、隔离性和持久性)属性;以及

[0006] 3.大数据量的并发处理需保证实时高效性。

[0007] 在上述几个方面中,尤以实时高效性更为突出。

发明内容

[0008] 本发明的一个目的是提供一种在多线程并发访问环境下的数据处理方法,其能够在保证数据完整性的同时高效实时地处理数据。

[0009] 在按照本发明一个实施例的在多线程并发访问环境下的数据处理方法中,对于所述多个线程的每一个,其按照下列方式在高速缓存中批量写入数据:

[0010] 确定与写入的数据在高速缓存中相关联的存储位置,所述存储位置包含锁定状态和解锁状态,以分别表示存在某一线程正对其执行写入操作和未有任何线程对其执行写入操作;

[0011] 对于每个写入的数据,执行下列步骤:

[0012] 如果与该写入的数据相关联的存储位置处于解锁状态,则记录指针的状态;

[0013] 将该相关联的存储位置置于锁定状态;

[0014] 如果所述指针的状态未发生变化,则对该相关联的存储位置执行写入操作,否则,则不执行写入操作并将该相关联的存储位置置于解锁状态;

[0015] 在完成所述写入操作之后,将该相关联的存储位置置于锁定状态。

[0016] 优选地,在上述数据处理方法中,所述写入的数据存储在所述高速缓存内的同步哈希表内,所述同步哈希表的每个节点包括相关联的数据、该相关联的数据的索引值以及表示解锁状态和锁定状态的标识。

[0017] 优选地,在上述数据处理方法中,对于在所述高速缓存中写入的数据,通过定期执

行下列步骤,将其写入数据源中:

[0018] 对所述高速缓存中写入的数据按照指定规则的的顺序排序;

[0019] 将前次执行写入数据源操作结束后发生更新的数据写入数据源中。

[0020] 优选地,在上述数据处理方法中,所述数据源为异构数据源。

[0021] 优选地,在上述数据处理方法中,所述指针的状态指示最近一次被执行写入操作的存储位置。

[0022] 本发明的还有一个目的是提供一种数据库管理系统,其能够在保证数据完整性的同时高效实时地处理数据。按照本发明一个实施例的数据库管理系统,,其能够在保证数据完整性的同时高效实时地处理数据。

[0023] 按照本发明一个实施例的数据库管理系统包括:

[0024] 多个数据源;

[0025] 与多个数据源相连的数据管理器,其包括:

[0026] 中央处理器;

[0027] 高速缓存;

[0028] I/O设备,

[0029] 其中,所述中央处理器被配置为使得多线程的每一个按照下列方式在高速缓存中写入数据:

[0030] 确定与写入的数据在高速缓存中相关联的存储位置,所述存储位置包含锁定状态和解锁状态,以分别表示存在某一线程正对其执行写入操作和未有任何线程对其执行写入操作;

[0031] 对于每个写入的数据,执行下列步骤:

[0032] 如果与该写入的数据相关联的存储位置处于解锁状态,则记录指针的状态;

[0033] 将该相关联的存储位置置于锁定状态;

[0034] 如果所述指针的状态未发生变化,则对该相关联的存储位置执行写入操作,否则,则不执行写入操作并将该相关联的存储位置置于解锁状态;

[0035] 在完成所述写入操作之后,将该相关联的存储位置置于锁定状态。

附图说明

[0036] 从结合附图的以下详细说明中,将会使本发明的上述和其它目的及优点更加完全清楚。

[0037] 图1为按照本发明一个实施例的数据库管理系统的示意图。

[0038] 图2为按照本发明一个实施例的在高速缓存中读取变化性较小数据的流程图。

[0039] 图3为按照本发明一个实施例的在高速缓存中写入变化性较小数据的流程图。

[0040] 图4为按照本发明一个实施例的在多个线程并发访问环境下的数据处理方法的流程图。

[0041] 图5为图4所示实施例中所用的哈希表结构的示意图。

具体实施方式

[0042] 下面参照其中图示了本发明示意性实施例的附图更为全面地说明本发明。但本发

明可以按不同形式来实现,而不应解读为仅限于本文给出的各实施例。给出的上述各实施例旨在使本文的披露全面完整,从而使对本发明保护范围的理解更为全面和准确。

[0043] 诸如“包含”和“包括”之类的用语表示除了具有在说明书和权利要求书中有直接和明确表述的单元和步骤以外,本发明的技术方案也不排除具有未被直接或明确表述的其它单元和步骤的情形。

[0044] 图1为按照本发明一个实施例的数据库管理系统的示意图。

[0045] 如图1所示,按照本实施例的数据库管理系统1包括多个数据源D1-Dn和与多个数据源D1-Dn相连的数据管理器11。数据管理器11包括中央处理器111、高速缓存112和I/O设备113,中央处理器111、高速缓存112和I/O设备113相互连接,中央处理器111被配置为使得多个线程的每一个按照下面将要借助图4和5所述的方式在高速缓存中写入数据。

[0046] 多个数据源D1-Dn可以是异构数据库。应用程序的每个连接都由连接的数据库的数据库别名作为唯一标识,数据库别名通过应用程序的参数从配置文件中获取,并且作为全局变量保存。每个业务流程都可以有N个来源库(DB_SRC1...N)和目标库(DB_DST)。应用程序通过输入的模块名自动识别判断数据源并自动连接,目标库DB_DST为数据集市库,主业务流程内部用(DB_SRC1...N)和(DB_DST)来控制多个数据库之间的切换,不需要关注对应的数据库实际的数据库名称、类型、结构等。应用程序与数据库采用下列自动重连机制:如果连续N次(这里的N是可配置的)与数据库重连都失败,则返回失败的消息,如果成功连接,则执行数据库操作,并且下次执行时,不再重新连接。此外,对于事务控制,应用程序在连接到目标库DB2_DST之前,不需要结束当前的事务。在一个工作单元中,可以有多个数据库连接,但是只有一个处于激活状态,其它都处于睡眠状态。可选地,可以用SETCONNECTIONTOdb_name语句来切换数据库连接。

[0047] 在本实施例中,为了避免与数据库的频繁交互操作,可以使用高效共享内存缓存或高速缓存112来存储变化性较小的数据(例如数据字典等,以下又称为静态数据)。具体而言,可在数据库管理系统1首次启动时将静态数据加载到高速缓存112中,以后主动或者定时刷新这些静态数据,便于多进程并发同时访问。

[0048] 上述静态数据在高速缓存112内按哈希表方式存放。具体而言,在首次将静态数据存入高速缓存内时,取一个预设值(例如一定长度的表长),将待存储的静态数据的索引值相对于该预设值取模,如果模运算的结果未发生哈希冲突,则将该结果作为相应数据的存储位置。当发生哈希冲突时,本实施例采用下列方式来解决:将模运算的结果加1再取模,如果本次模运算的结果未发生哈希冲突,则将该结果作为存储位置,否则重复模运算的结果加1再取模的步骤,直到不存在哈希冲突。

[0049] 图2为按照本发明一个实施例的在高速缓存中读取静态数据的流程图。

[0050] 如图2所示,在步骤210中,中央处理器111根据与待读取数据对应的索引表序列号确定相应的索引表。接着进入步骤220,中央处理器111将与待读取数据对应的索引值相对于预设值作模运算。

[0051] 随后进入步骤230,中央处理器111根据模运算的结果在索引表中查找与该结果对应的记录,如果存在记录并且该记录中包含的索引值与待读取数据对应的索引值不匹配,则进入步骤240,否则进入步骤250。

[0052] 在步骤240中,中央处理器111将步骤220中得到的模运算结果加1并且再次相对于

预设值作模运算,随后返回步骤230。

[0053] 在步骤250中,中央处理器111判断在索引表中是否存在与模运算结果对应的记录,如果存在,则进入步骤260,将该存储位置记录为正值并且向读取数据的应用程序返回该存储位置为正值的消息,应用程序由此可读取相应记录中的数据;否则,则进入步骤270,在索引表中将该存储位置记录为负值。

[0054] 图3为按照本发明一个实施例的在高速缓存中写入变化性较小数据的流程图。

[0055] 如图3所示,在步骤310中,中央处理器111根据与写入数据对应的索引表序列号确定相应的索引表。接着进入步骤320,中央处理器111将与待写入数据对应的索引值相对于预设值作模运算。

[0056] 随后进入步骤330,中央处理器111在索引表中检查与模运算结果对应的存储位置是否为负值,如果为负值,则进入步骤340,将数据写入该存储位置。

[0057] 在数据集市应用中,写入操作实时性要求高且操作频繁,为此在本实施例中,优选地采用读写分离机制。具体而言,可以将数据库读和写的操作对应于不同的数据库服务器,其中,主数据库提供写入操作,从数据库提供读取操作,当主数据库进行写入操作时,使用Memcache将数据同步到从数据库。

[0058] 图4为按照本发明一个实施例的在多线程并发访问环境下的数据处理方法的流程图。这里采用图1所示的数据库管理系统的结构,并且假设在多线程并发访问高速缓存的环境下,其中一个线程A向高速缓存中批量写入业务数据(例如交易数据等)的场景。需要指出的是,这里的业务数据区别于前述静态数据。

[0059] 如图4所示,在步骤410中,中央处理器111根据与线程A的多个待写入交易数据对应的索引表序列号确定相应的索引表Sync Hash Table。

[0060] 图5为索引表Sync Hash Table的结构示意图。如图5所示,该索引表采用哈希表结构,其包含多个节点Sync Hash Node,每个节点Sync Hash Node包括被写入交易数据value、该交易数据的索引值key以及加锁状态lockstatus。加锁状态lockstatus包括解锁状态和锁定状态两个状态,分别表示节点处于允许执行写入操作和不允许执行写入操作的状态。

[0061] 接着进入步骤420,中央处理器111根据每个待写入交易数据的索引值查找其在哈希表Sync Hash Table中相应的节点或存储位置,从而生成可用的节点列表。具体而言,在本步骤中,中央处理器111将与待写入交易数据对应的索引值相对于预设值作模运算,然后根据模运算的结果在索引表Sync Hash Table中查找与该结果对应的节点,如果该节点包含的索引值与待写入交易数据对应的索引值匹配,则将该节点加入节点列表;如果该节点包含的索引值与待写入交易数据对应的索引值不匹配,则将模运算结果加1并且再次相对于预设值作模运算,然后根据新的模运算结果在索引表Sync Hash Table中查找对应的节点,如果新的节点所包含的索引值与待写入交易数据对应的索引值匹配,则将新的节点放入节点列表,否则,则重复执行将模运算结果加1、相对于预设值作模运算和索引值匹配判断的操作。当所有待写入交易数据被遍历后,即生成最终的节点列表。

[0062] 随后进入步骤430,中央处理器111确定节点列表中是否存在尚未处理的节点,如果存在则进入步骤440,否则结束图4所示的流程。

[0063] 在步骤440中,中央处理器111从节点列表中提取一个尚未处理的节点作为线程A

当前处理的节点,并随后进入步骤450。

[0064] 在步骤450中,中央处理器111确定当前正在被处理的节点是否处于锁定状态,如果处于锁定状态,则表明该节点正由其它线程在执行写入操作,因此返回步骤430,否则进入步骤460。

[0065] 在步骤460中,中央处理器111记录指针的状态并且将线程A当前处理的节点的状态设置为锁定状态。这里的指针状态用于指示最近一次被执行写入操作的节点或存储位置。

[0066] 随后进入步骤470,中央处理器111确定指针的状态是否发生变化,如果发生变化,则表明其它线程刚刚完成一次对其它节点的写入操作,为了防止其它线程写入数据的丢失,线程A将不对当前处理的节点执行写入操作而是进入步骤480,将当前处理的节点设置为解锁状态并随后返回步骤430;如果指针状态未发生变化,则进入步骤490,线程A对当前处理的节点执行写入操作并且随后将当前处理的节点设置为解锁状态。在步骤490之后,图4所示的流程返回步骤430。

[0067] 本发明具有下列优点:

[0068] 借助于自定义数据结构的索引表Sync Hash Table,可以支持高并发、高吞吐量,且可以保证线程安全。使用多个锁来控制对Sync Hash Table的不同节点Hash Node的修改,每个节点Hash Node有独立的锁控制,互不影响。

[0069] 在并发PUT操作时,首先访问每个节点Hash Node的加锁状态,若是加锁状态,则寻找下一个可用的节点Hash Node,保证每个节点Hash Node是同步的,从而避免了同时有多个线程对一个节点Hash Node执行写入操作,只有等到该节点此次写入操作完成后才释放锁状态,供下个线程调用。

[0070] 与此同时,在每次执行写入操作前,首先判断指针状态是否变化,如果变化,则重新寻找可用的节点Hash Node执行写入操作,由此保证了整体的一致性。

[0071] 对于在高速缓存中写入的数据,可以通过定期执行下列步骤,将其写入数据源中:对高速缓存中写入的数据按照更新时间的顺序排序;将前次执行写入数据源操作结束后发生更新的数据写入数据源中。

[0072] 由于可以在不背离本发明基本精神的情况下,以各种形式实施本发明,因此上面描述的具体实施方式仅是说明性的而不是限制性的。本发明的范围由所附权利要求定义,对上面描述方式所作的各种变化或变动都属于所附权利要求的保护范围。

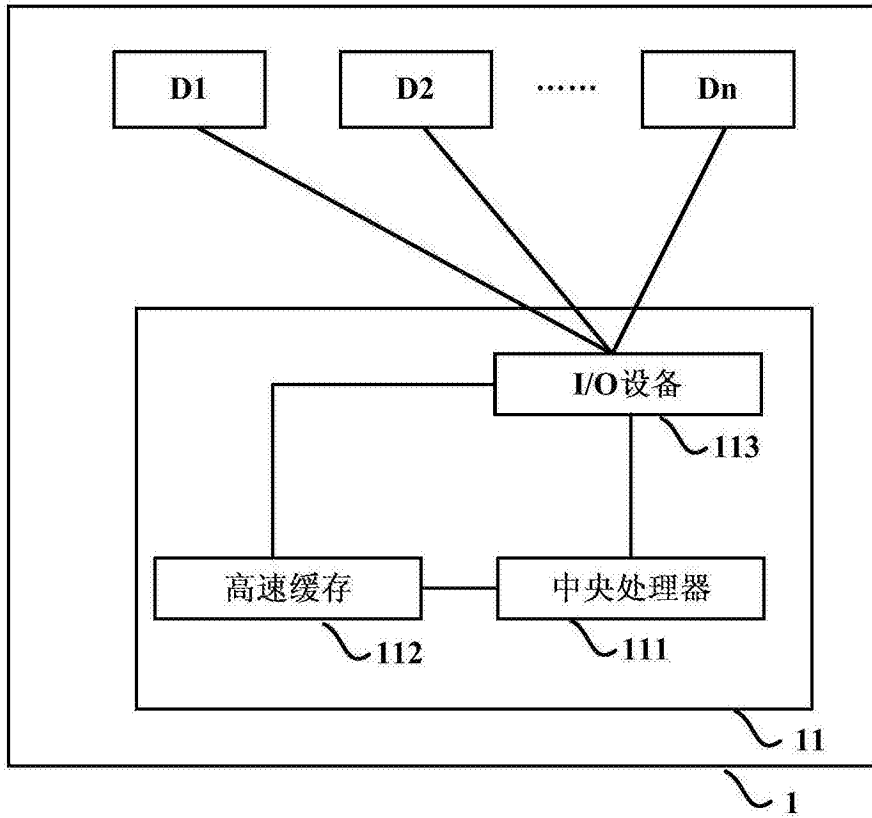


图1

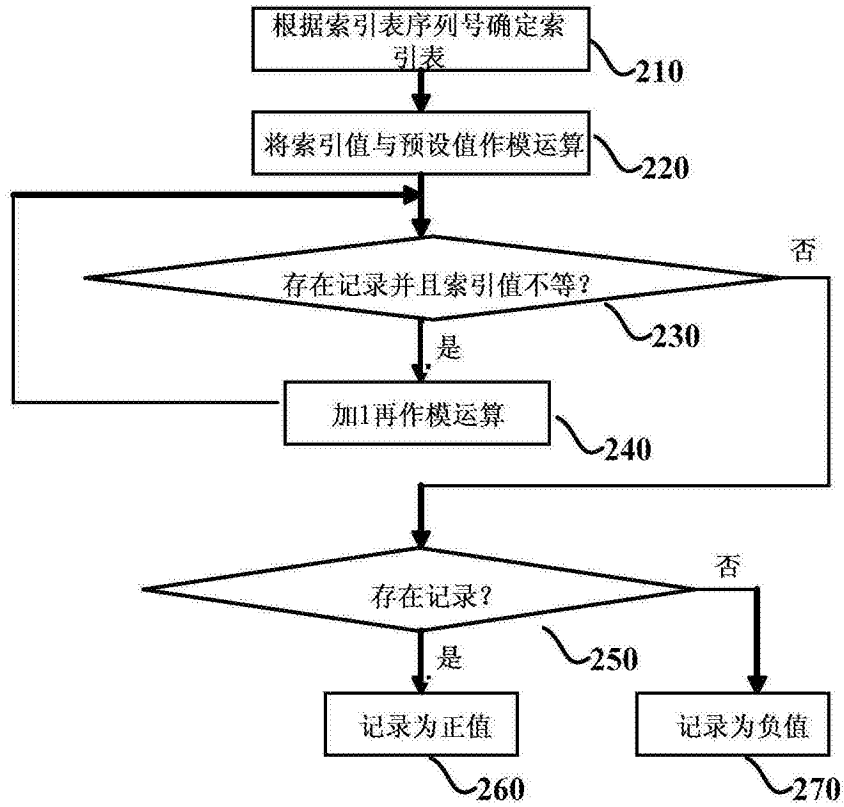


图2

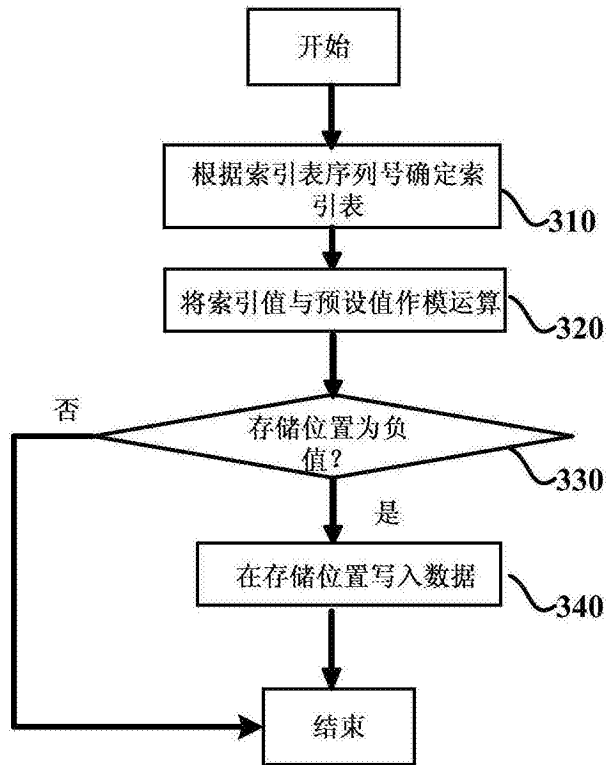


图3

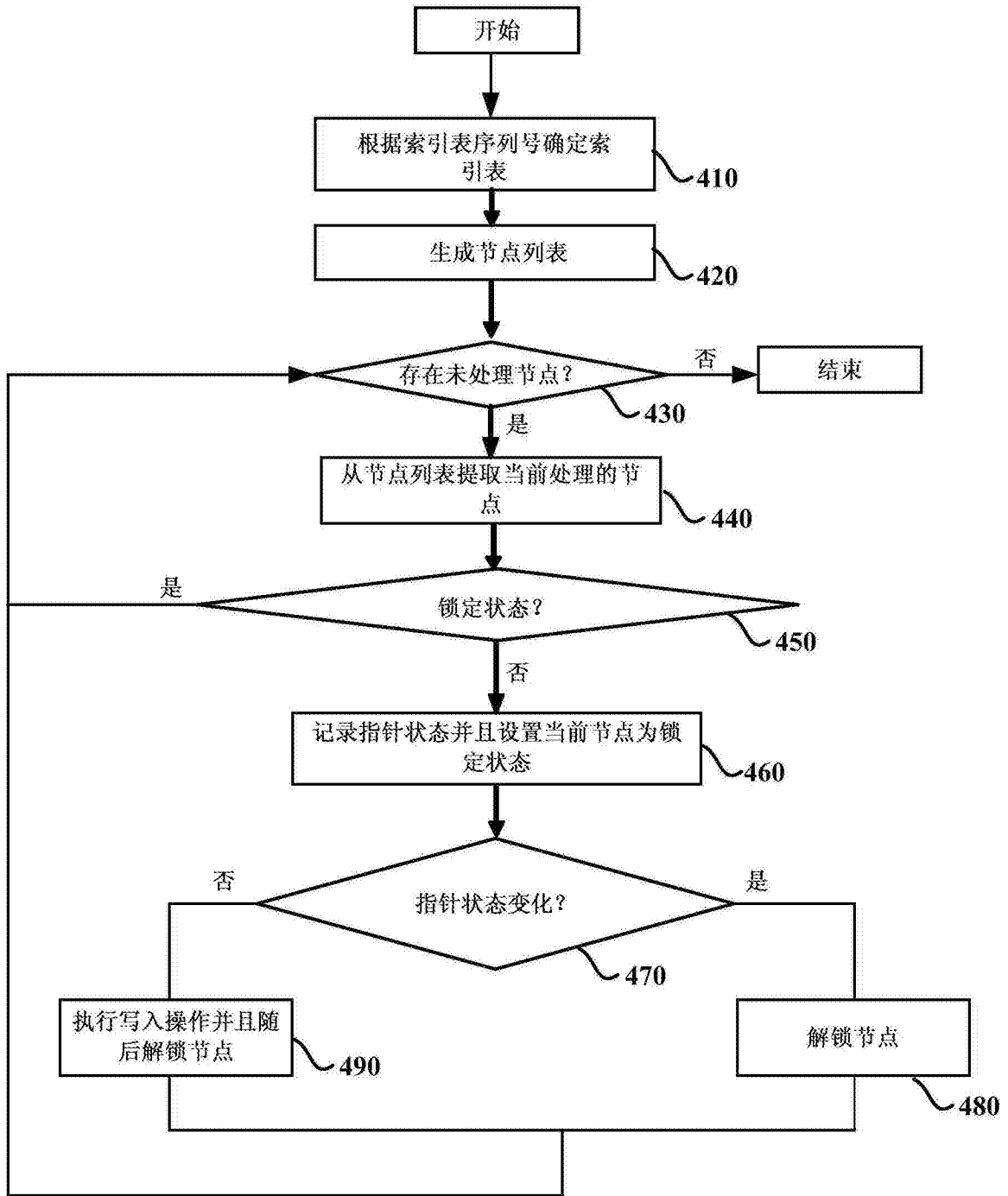


图4

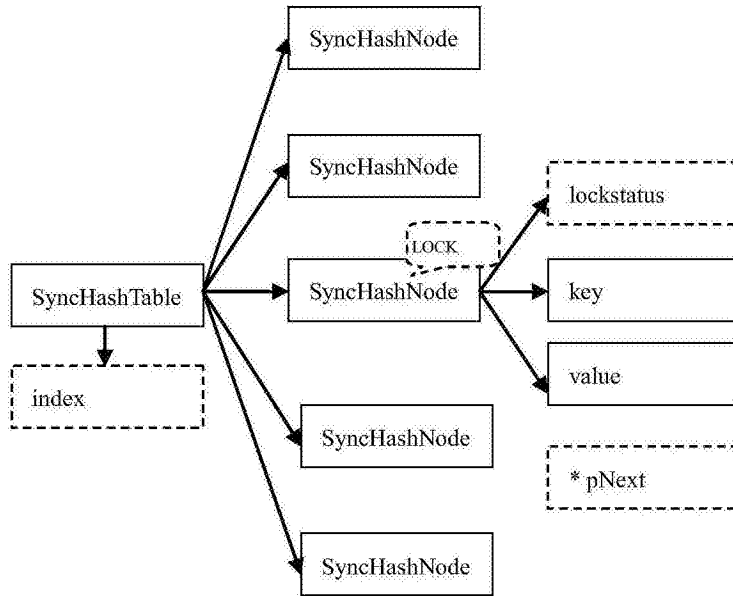


图5