



(12)发明专利申请

(10)申请公布号 CN 106326410 A

(43)申请公布日 2017.01.11

(21)申请号 201610710436.9

(22)申请日 2016.08.23

(71)申请人 无锡天脉聚源传媒科技有限公司
地址 214000 江苏省无锡市新区江苏软件
外包产业园射手座A座5楼

(72)发明人 江滔

(74)专利代理机构 北京尚伦律师事务所 11477
代理人 张亮

(51)Int.Cl.
G06F 17/30(2006.01)

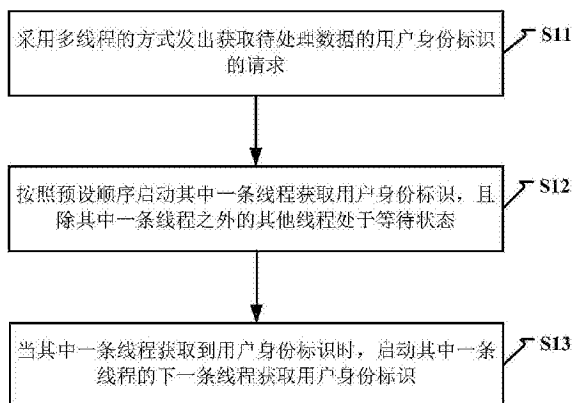
权利要求书2页 说明书5页 附图3页

(54)发明名称

一种数据查询方法及装置

(57)摘要

本发明公开了一种数据查询方法及装置,用于实现多线程处理数据时对用户ID的正确查询。所述方法包括:采用多线程的方式发出获取待处理数据的用户身份标识的请求;按照预设顺序启动其中一条线程获取所述用户身份标识,且除所述其中一条线程之外的其他线程处于等待状态;当所述其中一条线程获取到所述用户身份标识时,启动所述其中一条线程的下一条线程获取所述用户身份标识。该技术方案避免了在数据库中未存储有用用户身份标识时多条线程重复生成用户身份标识的情况,从而提高数据处理的效率。



1. 一种数据查询方法,其特征在于,应用于支持脚本语言Lua的数据库中,所述方法包括:

采用多线程的方式发出获取待处理数据的用户身份标识的请求;

按照预设顺序启动其中一条线程获取所述用户身份标识,且除所述其中一条线程之外的其他线程处于等待状态;

当所述其中一条线程获取到所述用户身份标识时,启动所述其中一条线程的下一条线程获取所述用户身份标识。

2. 根据权利要求1所述的方法,其特征在于,所述获取所述用户身份标识,包括:

判断所述数据库中是否存储有所述用户身份标识;

当所述数据库中存储有所述用户身份标识时,获取所述存储的用户身份标识;

当所述数据库中未存储有所述用户身份标识时,根据所述待处理数据的数字身份识别框架OpenID和频道ID生成所述用户身份标识。

3. 根据权利要求2所述的方法,其特征在于,所述根据所述待处理数据的数字身份识别框架OpenID和频道ID生成所述用户身份标识之后,所述获取所述用户身份标识,还包括:

将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID对应存储于所述数据库中。

4. 根据权利要求2或3所述的方法,其特征在于,所述生成所述用户身份标识,包括:

采用原子自增的方式生成所述用户身份标识。

5. 根据权利要求3所述的方法,其特征在于,所述将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID对应存储于所述数据库中,包括:

将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID以哈希表的方式存储于所述数据库中。

6. 一种数据查询装置,其特征在于,应用于支持脚本语言Lua的数据库中,所述装置包括:

请求模块,用于采用多线程的方式发出获取待处理数据的用户身份标识的请求;

第一获取模块,用于按照预设顺序启动其中一条线程获取所述用户身份标识,且除所述其中一条线程之外的其他线程处于等待状态;

第二获取模块,用于当所述其中一条线程获取到所述用户身份标识时,启动所述其中一条线程的下一条线程获取所述用户身份标识。

7. 根据权利要求6所述的装置,其特征在于,所述第一获取模块包括:

判断单元,用于判断所述数据库中是否存储有所述用户身份标识;

获取单元,用于当所述数据库中存储有所述用户身份标识时,获取所述存储的用户身份标识;

生成单元,用于当所述数据库中未存储有所述用户身份标识时,根据所述待处理数据的数字身份识别框架OpenID和频道ID生成所述用户身份标识。

8. 根据权利要求7所述的装置,其特征在于,所述第一获取模块还包括:

存储单元,用于根据所述待处理数据的数字身份识别框架OpenID和频道ID生成所述用户身份标识之后,将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID对应存储于所述数据库中。

9. 根据权利要求7或8所述的装置,其特征在于,所述生成单元,还用于采用原子自增的方式生成所述用户身份标识。

10. 根据权利要求8所述的装置,其特征在于,所述存储单元,还用于将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID以哈希表的方式存储于所述数据库中。

一种数据查询方法及装置

技术领域

[0001] 本发明涉及互联网技术领域,尤其涉及一种数据查询方法及装置。

背景技术

[0002] 数据库是按照数据结构来组织、存储和管理数据的仓库,数据库对数据的查询、处理速度远远超过普通文件的处理速度。随着移动互联网业务和用户数量的快速增长,传统的入库机制已经很难满足入库管理的需求,因此,采用多线程处理数据的方法应运而生。

[0003] 在处理数据时,每行数据都需要根据数字身份识别框架OpenID和频道ID这两项从数据库中查找用户ID,如果查找不到,则生成用户ID并记录到数据库中。在单线程的情况下以上处理方式是没问题的,但是如果是多线程处理数据时则会出现问题。当多个线程同时或者先后根据相同的OpenID和频道ID来查找用户ID时,如果数据库中恰好没有对应的用户ID,则需要生成用户ID。由于多个线程是先后来查找的、可能前一个线程刚好生成了一个用户ID、但还未来得及把生成的用户ID记录到数据库中,另外一个线程就来查找相同的OpenID和频道ID对应的用户ID了,此时后来的线程发现没有对应的用户ID,就会又生成了一个新的用户ID并存入到数据库中,这样则把之前线程生成的用户ID给改变了,这样在后面进行业务分析处理时被修改之前的用户ID相关的数据就成为了错误数据了。

发明内容

[0004] 本发明实施例提供一种数据查询方法及装置,用于实现多线程处理数据时对用户ID的正确查询。

[0005] 一种数据查询方法,应用于支持脚本语言Lua的数据库中,包括以下步骤:

[0006] 采用多线程的方式发出获取待处理数据的用户身份标识的请求;

[0007] 按照预设顺序启动其中一条线程获取所述用户身份标识,且除所述其中一条线程之外的其他线程处于等待状态;

[0008] 当所述其中一条线程获取到所述用户身份标识时,启动所述其中一条线程的下一条线程获取所述用户身份标识。

[0009] 在一个实施例中,所述获取所述用户身份标识,包括:

[0010] 判断所述数据库中是否存储有所述用户身份标识;

[0011] 当所述数据库中存储有所述用户身份标识时,获取所述存储的用户身份标识;

[0012] 当所述数据库中未存储有所述用户身份标识时,根据所述待处理数据的数字身份识别框架OpenID和频道ID生成所述用户身份标识。

[0013] 在一个实施例中,所述根据所述待处理数据的数字身份识别框架OpenID和频道ID生成所述用户身份标识之后,所述获取所述用户身份标识,还包括:

[0014] 将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID对应存储于所述数据库中。

[0015] 在一个实施例中,所述生成所述用户身份标识,包括:

- [0016] 采用原子自增的方式生成所述用户身份标识。
- [0017] 在一个实施例中,所述将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID对应存储于所述数据库中,包括:
- [0018] 将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID以哈希表的方式存储于所述数据库中。
- [0019] 一种数据查询装置,应用于支持脚本语言Lua的数据库中,所述装置包括:
- [0020] 请求模块,用于采用多线程的方式发出获取待处理数据的用户身份标识的请求;
- [0021] 第一获取模块,用于按照预设顺序启动其中一条线程获取所述用户身份标识,且除所述其中一条线程之外的其他线程处于等待状态;
- [0022] 第二获取模块,用于当所述其中一条线程获取到所述用户身份标识时,启动所述其中一条线程的下一条线程获取所述用户身份标识。
- [0023] 在一个实施例中,所述第一获取模块包括:
- [0024] 判断单元,用于判断所述数据库中是否存储有所述用户身份标识;
- [0025] 获取单元,用于当所述数据库中存储有所述用户身份标识时,获取所述存储的用户身份标识;
- [0026] 生成单元,用于当所述数据库中未存储有所述用户身份标识时,根据所述待处理数据的数字身份识别框架OpenID和频道ID生成所述用户身份标识。
- [0027] 在一个实施例中,所述第一获取模块还包括:
- [0028] 存储单元,用于根据所述待处理数据的数字身份识别框架OpenID和频道ID生成所述用户身份标识之后,将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID对应存储于所述数据库中。
- [0029] 在一个实施例中,所述生成单元,还用于采用原子自增的方式生成所述用户身份标识。
- [0030] 在一个实施例中,所述存储单元,还用于将所述生成的用户身份标识和所述数字身份识别框架OpenID以及所述频道ID以哈希表的方式存储于所述数据库中。
- [0031] 本发明实施例的一些有益效果可以包括:
- [0032] 采用本发明实施例提供的技术方案,能够在多线程都发出获取待处理数据的用户身份标识的请求时,按照预设顺序启动其中一条线程获取用户身份标识,且除该线程之外的其他线程均处于等待状态,避免了在数据库中未存储有用户身份标识时多条线程重复生成用户身份标识的情况,并且,仅在当前线程获取到用户身份标识之后,才会启动下一条线程获取用户身份标识,使得之后的线程在获取用户身份标识时,可直接获取到数据库中存储的用户身份标识或者第一条线程生成的用户身份标识,而无需重复性生成错误的用户身份标识,从而提高数据处理的效率。
- [0033] 本发明的其它特征和优点将在随后的说明书中阐述,并且,部分地从说明书中变得显而易见,或者通过实施本发明而了解。本发明的目的和其他优点可通过在所写的说明书、权利要求书、以及附图中所特别指出的结构来实现和获得。
- [0034] 下面通过附图和实施例,对本发明的技术方案做进一步的详细描述。

附图说明

[0035] 附图用来提供对本发明的进一步理解,并且构成说明书的一部分,与本发明的实施例一起用于解释本发明,并不构成对本发明的限制。在附图中:

[0036] 图1为本发明实施例中一种数据查询方法的流程图;

[0037] 图2为本发明实施例中一种数据查询方法中获取用户ID的流程图;

[0038] 图3为本发明实施例中一种数据查询装置的框图;

[0039] 图4为本发明实施例中一种数据查询装置中第一获取模块的框图。

具体实施方式

[0040] 以下结合附图对本发明的优选实施例进行说明,应当理解,此处所描述的优选实施例仅用于说明和解释本发明,并不用于限定本发明。

[0041] 图1为本发明实施例中一种数据查询方法的流程图。该数据查询方法应用于支持脚本语言Lua的数据库(例如数据库Redis)中,如图1所示,包括以下步骤S11-S13:

[0042] 步骤S11,采用多线程的方式发出获取待处理数据的用户身份标识的请求。

[0043] 步骤S12,按照预设顺序启动其中一条线程获取用户身份标识,且除其中一条线程之外的其他线程处于等待状态。

[0044] 步骤S13,当其中一条线程获取到用户身份标识时,启动其中一条线程的下一条线程获取用户身份标识。

[0045] 其中,用户身份标识即用户ID,因此以下简称用户ID。

[0046] 上述实施例中,执行步骤S12时,预设顺序可设为多线程执行任务的先后顺序,例如,按照多线程执行任务的先后顺序,可先启动第一条线程获取用户ID,当第一条线程获取到用户ID时,再启动第二条线程获取用户ID,以此类推,直至所有线程都获取到用户ID。所启动的其中一条线程可按照如图2所示的步骤S21-S24来获取用户ID:

[0047] 步骤S21,判断所述数据库中是否存储有用户ID;当数据库中存储有用户ID时,执行步骤S22;当数据库中未存储有用户ID时,执行步骤S23。

[0048] 步骤S22,获取存储的用户ID。

[0049] 步骤S23,根据待处理数据的数字身份识别框架OpenID和频道ID生成用户ID。

[0050] 执行步骤S23时,可采用原子自增的方式生成用户ID。例如,当数据库为Redis时,可采用Redis中的incr命令生成用户ID。

[0051] 步骤S24,将生成的用户ID和数字身份识别框架OpenID以及频道ID对应存储于数据库中。

[0052] 执行步骤S24时,可将生成的用户ID和数字身份识别框架OpenID以及频道ID以哈希表的方式存储于数据库中。在该哈希表中,频道ID对应哈希表中的key,数字身份识别框架OpenID对应哈希表中的field,用户ID对应哈希表中的value。

[0053] 当上述其中一条线程获取到用户ID之后,则启动下一条线程获取用户ID,下一条线程获取用户ID的方式与图2中所示的步骤S21-S24相同,即每条线程获取用户ID的方式均相同,因此不再赘述。

[0054] 在一个具体实施例中,需要处理数据库Redis中的一行数据,当多条线程发出获取该行数据对应的用户ID请求时,由主线程启动多线程中的第一条线程从Redis中获取用户ID,同时多线程中的其他线程均处于等待状态。如果Redis中存储有该行数据对应的用户

ID,则第一条线程直接获取该用户ID;如果Redis中未存储有该行数据对应的用户ID,则第一条线程根据该行数据的数字身份识别框架OpenID和频道ID、并采用Redis中的incr命令生成该行数据对应的用户ID,同时将所生成的用户ID与该行数据的OpenID、频道ID对应存储在Redis中。此时,第一条数据已获取到用户ID,则由主线程启动下一条线程(即第二条线程)获取该行数据对应的用户ID,此时Redis中必然存储有该行数据对应的用户ID(因为即使Redis中预先未存储有该行数据对应的用户ID,也已由第一条线程在获取用户ID时生成并存储该行数据对应的用户ID),因此第二条线程可直接获取Redis中的用户ID,而无需再重复性生成错误的用户ID。可见,该实施例中的技术方案能够有效避免多条线程同时获取用户ID时导致的用户ID发生错误的情况。

[0055] 采用上述任一实施例提供的技术方案,能够在多线程都发出获取待处理数据的用户身份标识的请求时,按照预设顺序启动其中一条线程获取用户身份标识,且除该线程之外的其他线程均处于等待状态,避免了在数据库中未存储有用户身份标识时多条线程重复生成用户身份标识的情况,并且,仅在当前线程获取到用户身份标识之后,才会启动下一条线程获取用户身份标识,使得之后的线程在获取用户身份标识时,可直接获取到数据库中存储的用户身份标识或者第一条线程生成的用户身份标识,而无需重复性生成错误的用户身份标识,从而提高数据处理的效率。

[0056] 图3为本发明实施例中一种数据查询装置的框图。如图3所示,该装置应用于支持脚本语言Lua的数据库中,包括:

[0057] 请求模块31,用于采用多线程的方式发出获取待处理数据的用户身份标识的请求;

[0058] 第一获取模块32,用于按照预设顺序启动其中一条线程获取用户身份标识,且除其中一条线程之外的其他线程处于等待状态;

[0059] 第二获取模块33,用于当其中一条线程获取到用户身份标识时,启动其中一条线程的下一条线程获取用户身份标识。

[0060] 在一个实施例中,如图4所示,第一获取模块32包括:

[0061] 判断单元321,用于判断数据库中是否存储有用户身份标识;

[0062] 获取单元322,用于当数据库中存储有用户身份标识时,获取存储的用户身份标识;

[0063] 生成单元323,用于当数据库中未存储有用户身份标识时,根据待处理数据的数字身份识别框架OpenID和频道ID生成用户身份标识。

[0064] 在一个实施例中,第一获取模块32还包括:

[0065] 存储单元,用于根据待处理数据的数字身份识别框架OpenID和频道ID生成用户身份标识之后,将生成的用户身份标识和数字身份识别框架OpenID以及频道ID对应存储于数据库中。

[0066] 在一个实施例中,生成单元323还用于采用原子自增的方式生成用户身份标识。

[0067] 在一个实施例中,存储单元还用于将生成的用户身份标识和数字身份识别框架OpenID以及频道ID以哈希表的方式存储于数据库中。

[0068] 采用本发明实施例提供的装置,能够在多线程都发出获取待处理数据的用户身份标识的请求时,按照预设顺序启动其中一条线程获取用户身份标识,且除该线程之外的其

他线程均处于等待状态,避免了在数据库中未存储有用户身份标识时多条线程重复生成用户身份标识的情况,并且,仅在当前线程获取到用户身份标识之后,才会启动下一条线程获取用户身份标识,使得之后的线程在获取用户身份标识时,可直接获取到数据库中存储的用户身份标识或者第一条线程生成的用户身份标识,而无需重复性生成错误的用户身份标识,从而提高数据处理的效率。

[0069] 关于上述实施例中的装置,其中各个模块执行操作的具体方式已经在有关该方法的实施例中进行了详细描述,此处将不做详细阐述说明。

[0070] 本领域内的技术人员应明白,本发明的实施例可提供为方法、系统、或计算机程序产品。因此,本发明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,本发明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和光学存储器等)上实施的计算机程序产品的形式。

[0071] 本发明是参照根据本发明实施例的方法、设备(系统)、和计算机程序产品的流程图和/或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器,使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装置。

[0072] 这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中,使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品,该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

[0073] 这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上,使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理,从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的步骤。

[0074] 显然,本领域的技术人员可以对本发明进行各种改动和变型而不脱离本发明的精神和范围。这样,倘若本发明的这些修改和变型属于本发明权利要求及其等同技术的范围之内,则本发明也意图包含这些改动和变型在内。

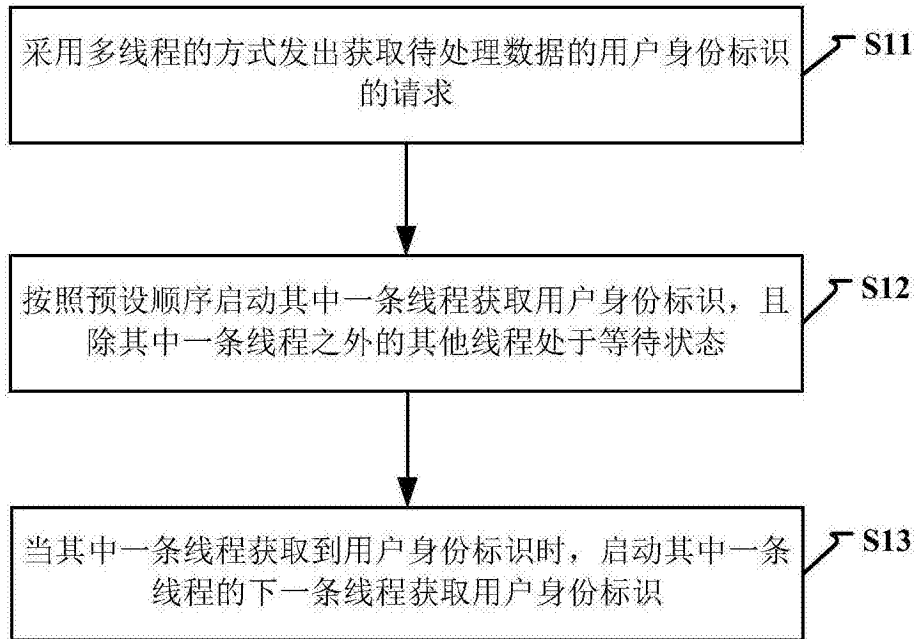


图1

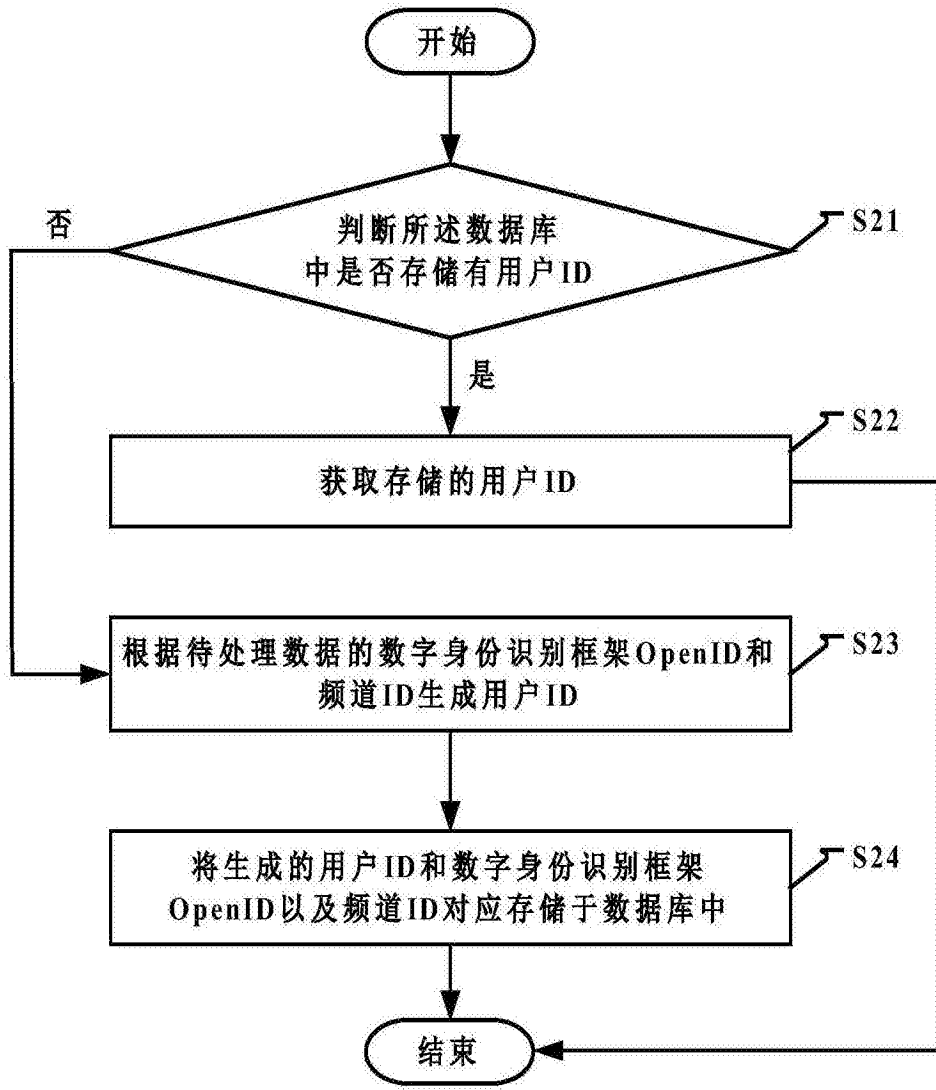


图2

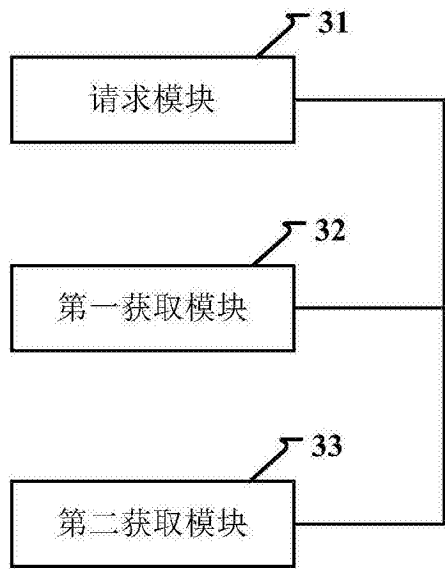


图3

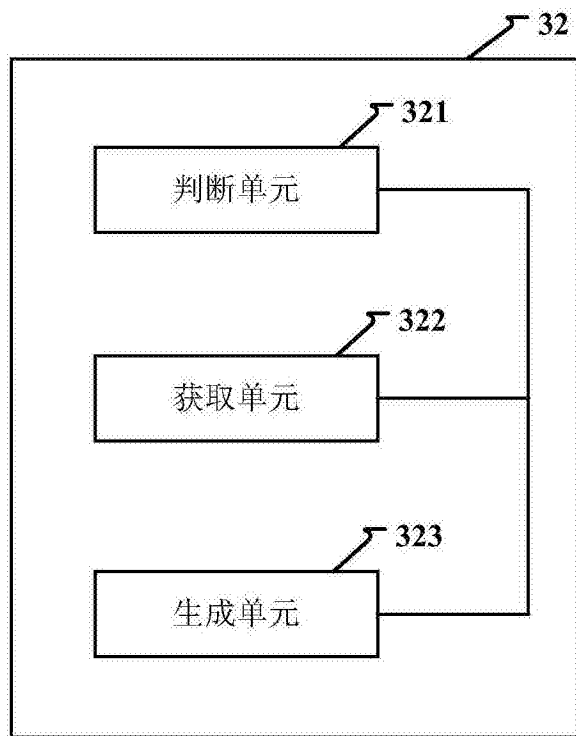


图4