

随机游走、基于集群的随机游走方法、装置以及设备

技术领域

- [01] 本说明书涉及计算机软件技术领域，尤其涉及随机游走、基于集群的随机游走方法、
5 装置以及设备。

背景技术

- [02] 随着计算机和互联网技术的迅速发展，很多业务都可以在网上进行，图计算是处理社交方面的网上业务的一种常用手段。
- 10 [03] 例如，对于社交风控业务中的账户欺诈识别：每个用户分别作为一个节点，若两个用户之间存在转账关系，则对应的两个节点之间存在一条边，边可以是无向的，也可以是根据转账方向定义了方向的；以此类推，可以得到包含多个节点和多条边的图数据，进而基于图数据进行图计算以实现风控。
- [04] 随机游走算法是图计算中比较基础和重要的一环，其为上层复杂算法提供支持。在
15 现有技术中，一般采用这样的随机游走算法：在数据库中随机读取图数据包含的一个节点，再继续在该数据库中随机读取该节点的一个相邻节点，以此类推，实现在图数据中的随机游走。
- [05] 基于现有技术，需要能够应用于大规模图数据的更为高效的随机游走方案。

20 发明内容

- [06] 本说明书实施例提供随机游走、基于集群的随机游走方法、装置以及设备，用以解决如下技术问题：需要能够应用于大规模图数据的更为高效的随机游走方案。
- [07] 为解决上述技术问题，本说明书实施例是这样实现的：
- [08] 本说明书实施例提供一种基于集群的随机游走方法，包括：
- 25 [09] 所述集群获取图数据包含的各节点的信息；

- [10]根据所述各节点的信息,生成哈希表,所述哈希表的键包括所述节点的标识,值包括所述节点的相邻节点的标识;
- [11]根据所述哈希表,生成随机序列,所述随机序列反映在所述图数据中的随机游走。
- [12]本说明书实施例提供的一种随机游走方法,包括:
- 5 [13]获取根据图数据包含的各节点的信息生成的哈希表,所述哈希表的键包括所述节点的标识,值包括所述节点的相邻节点的标识;
- [14]根据所述哈希表,生成随机序列,所述随机序列反映在所述图数据中的随机游走。
- [15]本说明书实施例提供的一种基于集群的随机游走装置,所述装置属于所述集群,包括:
- 10 [16]获取模块,获取图数据包含的各节点的信息;
- [17]第一生成模块,根据所述各节点的信息,生成哈希表,所述哈希表的键包括所述节点的标识,值包括所述节点的相邻节点的标识;
- [18]第二生成模块,根据所述哈希表,生成随机序列,所述随机序列反映在所述图数据中的随机游走。
- 15 [19]本说明书实施例提供的一种随机游走装置,包括:
- [20]获取模块,获取根据图数据包含的各节点的信息生成的哈希表,所述哈希表的键包括所述节点的标识,值包括所述节点的相邻节点的标识;
- [21]生成模块,根据所述哈希表,生成随机序列,所述随机序列反映在所述图数据中的随机游走。
- 20 [22]本说明书实施例提供的一种基于集群的随机游走设备,所述设备属于所述集群,包括:
- [23]至少一个处理器;以及,
- [24]与所述至少一个处理器通信连接的存储器;其中,
- [25]所述存储器存储有可被所述至少一个处理器执行的指令,所述指令被所述至少一个
- 25 处理器执行,以使所述至少一个处理器能够:
- [26]获取图数据包含的各节点的信息;
- [27]根据所述各节点的信息,生成哈希表,所述哈希表的键包括所述节点的标识,值包

括所述节点的相邻节点的标识;

[28]根据所述哈希表,生成随机序列,所述随机序列反映在所述图数据中的随机游走。

[29]本说明书实施例提供一种随机游走设备,包括:

[30]至少一个处理器;以及,

5 [31]与所述至少一个处理器通信连接的存储器;其中,

[32]所述存储器存储有可被所述至少一个处理器执行的指令,所述指令被所述至少一个处理器执行,以使所述至少一个处理器能够:

[33]获取根据图数据包含的各节点的信息生成的哈希表,所述哈希表的键包括所述节点的标识,值包括所述节点的相邻节点的标识;

10 [34]根据所述哈希表,生成随机序列,所述随机序列反映在所述图数据中的随机游走。

[35]本说明书实施例采用的上述至少一个技术方案能够达到以下有益效果:有利于减少对原始保存图数据的数据库的访问,哈希表在生成后无需依赖该数据库,通过哈希表能够快速索引节点的相邻节点,该方案能够适用于大规模图数据且效率较高,在基于集群实施该方案的情况下,还能够进一步地提高效率。

15

附图说明

[36]为了更清楚地说明本说明书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说明书中记载的一些实施例,对于本领域普通技术人员来讲,在不付出创造性劳动性的前提下,还可以根据这些附图获得其他的附图。

20

[37]图1为本说明书的方案在一种实际应用场景下涉及的一种整体架构示意图;

[38]图2为本说明书实施例提供一种基于集群的随机游走方法的流程示意图;

[39]图3为本说明书实施例提供一种实际应用场景下,基于集群的哈希表生成流程示意图;

25 [40]图4为本说明书实施例提供一种实际应用场景下,基于集群的随机序列生成流程示意图;

[41]图 5 为本说明书实施例提供的一种随机游走方法的流程示意图；

[42]图 6 为本说明书实施例提供的对应于图 2 的一种基于集群的随机游走装置的结构示意图；

[43]图 7 为本说明书实施例提供的对应于图 5 的一种随机游走装置的结构示意图。

5

具体实施方式

[44]本说明书实施例提供随机游走、基于集群的随机游走方法、装置以及设备。

[45]为了使本技术领域的人员更好地理解本说明书中的技术方案，下面将结合本说明书实施例中的附图，对本说明书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
10 例仅仅是本申请一部分实施例，而不是全部的实施例。基于本说明书实施例，本领域普通技术人员在没有作出创造性劳动前提下所获得的所有其他实施例，都应当属于本申请保护的范围。

[46]本说明书的方案既适用于集群，也适用于单机。在集群下对于大规模图数据的处理效率更高，原因在于：可以拆分任务（比如，数据读取任务、数据同步任务等），进而
15 由集群中的多个机器并行执行被分配给自己的一部分任务。以下各实施例主要基于集群场景进行说明。

[47]方案涉及的集群可以有一个或者多个，以图 1 为例，涉及了两个集群。

[48]图 1 为本说明书的方案在一种实际应用场景下涉及的一种整体架构示意图。该整体架构中，主要涉及三部分：服务器集群、工作机集群、数据库。数据库保存有图数据，
20 供集群读取，服务器集群与工作机集群相互配合，根据从数据库读取的数据，实现在图数据中的随机游走。

[49]图 1 中的架构是示例性的，并非唯一。比如，方案可以涉及一个集群，该集群中包含至少一个调度机和多个工作机；再比如，方案也可以涉及一个工作机集群和一个服务器；等等；方案涉及的机器相互配合，实现在图数据中的随机游走。

25 [50]下面对本说明书的方案进行详细说明。

[51]图 2 为本说明书实施例提供的一种基于集群的随机游走方法的流程示意图。图 2 中各步骤由集群中的至少一个机器（或者机器上的程序）执行，不同步骤的执行主体可以

不同。

[52]图 2 中的流程包括以下步骤:

[53]S202: 所述集群获取图数据包含的各节点的信息。

5 [54]在本说明书实施例中,节点的信息可以包括:节点的标识、节点的相邻节点的标识
(以下以此为例)、或者标识以外的能够指示节点的相邻节点的信息等。各节点的信息
可以是一次性获取的,也可以是分多次获取的。

[55]一般地,原始的图数据保存于数据库中,在这种情况下,需要通过访问数据库,读
取得到各节点的信息。为了避免重复读取数据增加数据库的负担,集群中的多个机器可
以分别读取不重复的一部分节点的信息,进一步地,多个机器可以并行读取数据库,以
10 快速获取节点的信息。

[56]例如,可以由工作机集群中的各工作机并行地、分别从数据库读取一部分节点的
信息并进行处理,再将处理后得到的数据同步至服务器集群。或者,各工作机也可以将读
取的节点的信息,直接同步至服务器集群,由服务器集群进一步地处理。所述处理至少
包括生成哈希表。

15 [57]S204: 根据所述各节点的信息,生成哈希表,所述哈希表的键包括所述节点的标识,
值包括所述节点的相邻节点的标识。

[58]在本说明书实施例中,任意节点的标识及该节点的相邻节点的标识,可以构成哈希
表中的键值对。进而,根据哈希表以及任意节点的标识,能够快速索引到该节点的任意
相邻节点的标识,从而,有利于高效地在图数据中随机游走。

20 [59]为了支持索引时进行的哈希运算,各节点的标识优选地为数字。比如,用各节点的
标识大小定义各节点的顺序,从 0 开始计数,顺序最先的节点的标识为 0、顺序第二的
节点的标识为 1,依次类推。以下各实施例基于该例中的定义进行说明。

[60]当然,若节点原本的标识并非数字,也可以基于一一映射的规则,将所述原本的标
识映射为数字后,作为节点的标识用于生成哈希表。

25 [61]在本说明书实施例中,一个节点可能有多个相邻节点,则哈希表中对应的键值对中
的值相应地可能包括这多个相邻节点的标识,哈希表可以用诸如数组、或者链表等能够
容纳多个元素的数据结构存储所述值。如此,也能够便利地通过这些数据结构,索引任
意相邻节点的标识。

[62]以数组为例，每个节点分别对应一个所述数组，该数组存储对应节点的各相邻节点的标识，其中，每个所述相邻节点的标识分别为该数组中的一个元素，通过针对数组元素的访问操作，即可索引到相邻节点的标识。

5 [63]S208：根据所述哈希表，生成随机序列，所述随机序列反映在所述图数据中的随机游走。

[64]在本说明书实施例中，随机序列为多个节点的标识构成的序列，各标识在该随机序列中的顺序即为随机游走顺序，随机序列的最大长度一般由预定的随机游走步数决定。

[65]在得到哈希表后，可以相互独立地多次执行步骤 S206，进而得到多个相互独立的随机序列。比如，各工作机分别根据哈希表，生成一个或者多个随机序列。

10 [66]通过图 2 的方法，有利于减少对原始保存图数据的数据库的访问，哈希表在生成后无需依赖该数据库，通过哈希表能够快速索引节点的相邻节点，该方案能够适用于大规模图数据且效率较高，由于基于集群实施该方法，因此，还能够进一步地提高效率。

[67]基于图 2 的方法，本说明书实施例还提供了该方法的一些具体实施方案，以及扩展方案，下面以图 1 中的架构为例，进行说明。

15 [68]在本说明书实施例中，所述集群可以包括服务器集群和工作机集群，对于步骤 S202，所述集群获取图数据包含的各节点的信息，具体可以包括：

[69]所述工作机集群从数据库读取图数据包含的各节点的相邻节点的标识，其中，每个工作机读取一部分节点的相邻节点的标识。需要说明的是，若对于工作机集群，节点的标识本身也是未知的，则工作机集群可以读取节点的标识，并根据节点的标识（在数据库
20 库中作为主键），读取节点的相邻节点的标识。

[70]例如，假定有标识分别为 0~4 的 5 个节点。工作机集群包括工作机 0、工作机 1、工作机 2，每个工作机分别从数据库读取一部分节点的相邻节点的标识；比如，工作机 0 读取节点 1 的相邻节点的标识（分别为 0、2），以及节点 2 的标识（分别为 1、3、4）；工作机 1 读取节点 0 的相邻节点的标识（为 1）；工作机 2 读取节点 3 的相邻节点的标识（分别为 2、4），以及节点 4 的标识（分别为 2、3）。
25

[71]在本说明书实施例中，各工作机可以根据自己读取标识的相邻节点及其对应节点的标识，生成非全量的哈希表。

[72]进一步地，工作机集群可以将这些非全量的哈希表同步给服务器集群。从而，服务

- 器集群能够得到由这些非全量的哈希表构成的全量的哈希表，具体地：服务器集群可以通过专门整合（比如，拆分哈希表、合并哈希表等）这些非全量的哈希表，得到全量的哈希表；也可以不专门整合而只是在工作机集群同步完毕后，将同步得到的全部数据视为一个整体，即全量的哈希表。服务器集群中的各服务机可以分别保存全量的哈希表，
- 5 也可以只保存全量的哈希表的一部分。

- [73]步骤 S204 所述的哈希表可以是所述全量的哈希表，也可以是所述非全量的哈希表。需要说明的是，若是非全量的哈希表，则后续的随机游走相应地在该非全量的哈希表涉及的节点（这些节点只是图数据中的一部分节点）中进行，任意工作机可以根据自己生成的非全量的哈希表，生成随机序列，而未必要依赖于上述同步和服务集群。
- 10 [74]对上述同步后的动作继续说明。服务器集群可以进一步地将全量的哈希表再向各工作机分别同步，以便各工作机能够根据全量的哈希表，生成随机序列。

[75]沿用上例，结合图 3 进行说明。图 3 为本说明书实施例提供的一种实际应用场景下，基于集群的哈希表生成流程示意图。

- [76]在图 3 中，数据库中的数据表以节点的标识作为主键，记录了各节点的相邻节点的标识，其中，节点 0 的相邻节点为节点 1，节点 1 的相邻节点为节点 0、节点 2，节点 2 的相邻节点为节点 1、节点 3、节点 4，节点 3 的相邻节点为节点 2、节点 4，节点 4 的相邻节点为节点 2、节点 3。工作机 0~2 如前所述，优选地可以并行分别从数据库读取一部分节点的相邻节点的标识。
- 15

- [77]每个工作机根据自己读取的标识，对应地生成非全量的哈希表。工作机 0 生成的哈希表中包含两个键值对，工作机 1 生成的哈希表中包含一个键值对，工作机 2 生成的哈希表中包含两个键值对。
- 20

[78]工作机集群将生成的各非全量的哈希表都同步至服务器集群，可以看到服务器集群得到了全量的哈希表并分部分保存在服务器 0~2 中。

- [79]服务器集群将全量的哈希表分别同步给各工作机。则各工作机可以分别独立地根据全量的哈希表，生成随机序列。
- 25

[80]在本说明书实施例中，对于步骤 S206，所述根据所述哈希表，生成随机序列，具体可以包括：

[81]所述工作机在所述各节点的标识中，随机确定一个标识，作为目标节点的标识；以所述目标节点的标识为键，在所述哈希表中索引得到对应的值，所述对应的值包括所述

目标节点的相邻节点的标识；确定所述对应的值包括的相邻节点的标识的数量；随机确定一个小于所述数量的非负整数，并获取所述对应的值包括的第所述非负整数个相邻节点的标识；通过将该第所述非负整数个相邻节点重新作为目标节点进行迭代计算，生成由依次得到的各目标节点的标识构成的随机序列。

- 5 [82]进一步地沿用图 3 的例子，结合图 4 说明。图 4 为本说明书实施例提供的一种实际应用场景下，基于集群的随机序列生成流程示意图。

[83]假定图数据共包含 N 个节点，第 n 个所述节点的标识为 n ， n 从 0 开始计数。在图 5 中， $N=5$ ，全量的哈希表中相应地包含 5 个键值对，每个键值对的值存储于一个数组中，该键对应的节点的标识为第 n 个相邻节点的标识为该数组的第 n 个元素。

- 10 [84]工作机随机生成一个属于 $[0, N-1=4]$ 的整数，即工作机在各节点的标识中，随机确定的目标节点的标识；以目标节点的标识为键，在全量的哈希表中索引得到对应的值；查询确定存储该值的数组包含的元素数量，即该值包括的相邻节点的标识的数量；随机确定一个小于该数量的非负整数，记作 j ；通过读取该数组的第 j 个元素，获取目标节点的第 j 个相邻节点的标识。

- 15 [85]假定目标节点的标识为 2， $j=1$ 。则目标节点为节点 2，该数组为 $[1, 3, 4]$ ，获取的目标节点的第 1 个相邻节点的标识为该数组的第 1 个元素，即 3。从而，实现从节点 2 随机游走到节点 3，进而将节点 3 作为目标节点迭代计算，继续随机游走，如此，依次经过的多个节点的标识构成随机序列。

- [86]在图 4 中，预先设定随机游走步数为 8，批数为 5。用矩阵进行表示，随机游走步数
20 比如为该矩阵的列数，批数为该矩阵的行数，该矩阵的每一行可以存储一个随机序列。

[87]随机游走步数定义了一个随机序列的最大长度，每当随机序列达到该最大长度时，可以不依赖该随机序列而开始生成下一个随机序列。

- [88]批数定义了每个工作机在向数据库写入已生成前，生成随机序列的最大个数，到达该最大个数时，工作机可以将自己已生成未写入的多个随机序列（表示为对应的矩阵）
25 写入数据库。比如，图 5 中工作机 2 当前已生成未写入的随机序列已经到达最大个数 5，则可以将对应的矩阵写入数据库。

[89]以图 4 中工作机 0 生成的第一个随机序列（3，4，3，2，4，2，3，2）为例，该随机序列即表示依次经过下列节点的随机游走过程：节点 3、节点 4、节点 3、节点 2、节点 4、节点 2、节点 3、节点 2。

[90]进一步地，还可以预先设定阈值，用于限定整个工作机集群生成的随机序列的最大总数量。当到达该设定阈值时，各工作机可以停止生成随机序列。

[91]另外，在实际应用中，工作机集群中的某些工作机可能会出现异常，导致之前同步得到的哈希表丢失。比如，若工作机将得到的哈希表只存储在内存中，则宕机后内存中的数据会丢失。在这种情况下，当这些工作机恢复正常时，可以从服务器集群重新获取丢失的哈希表。图 4 中通过工作机 2 示出了这种情况。

[92]上面主要是基于集群场景，对本说明书的方案进行说明的，本说明书的方案也可以脱离集群场景。比如，基于同样的思路，本说明书实施例还提供了一种随机游走方法的流程图示意图，如图 5 所示。

10 [93]图 5 中的流程的执行主体可以是单一的计算设备，也可以是多个计算设备，该流程包括以下步骤：

[94]S502：获取根据图数据包含的各节点的信息生成的哈希表，所述哈希表的键包括所述节点的标识，值包括所述节点的相邻节点的标识。

15 [95]在步骤 S502 中，哈希表具体由谁生成，本申请并不做限定。一般地，只要图数据未发生变化，根据该图数据已生成的哈希表可以一直复用。

[96]S504：根据所述哈希表，生成随机序列，所述随机序列反映在所述图数据中的随机游走。

[97]基于同样的思路，本说明书实施例还提供了上面各方法的对应装置，如图 6、图 7 所示。

20 [98]图 6 为本说明书实施例提供的对应于图 2 的一种基于集群的随机游走装置的结构示意图，该装置属于所述集群，包括：

[99]获取模块 601，获取图数据包含的各节点的信息；

[100] 第一生成模块 602，根据所述各节点的信息，生成哈希表，所述哈希表的键包括所述节点的标识，值包括所述节点的相邻节点的标识；

25 [101] 第二生成模块 603，根据所述哈希表，生成随机序列，所述随机序列反映在所述图数据中的随机游走。

[102] 可选地，所述集群包括服务器集群和工作机集群；

[103] 所述获取模块 601 获取图数据包含的各节点的信息，具体包括：

- [104] 所述工作机集群从数据库读取图数据包含的各节点的相邻节点的标识, 其中, 每个工作机读取一部分节点的相邻节点的标识。
- [105] 可选地, 所述第一生成模块 602 根据所述各节点的信息, 生成哈希表, 具体包括:
- 5 [106] 各所述工作机分别根据自己读取标识的相邻节点及其对应节点的标识, 生成非全量的哈希表;
- [107] 所述工作机集群将各所述非全量的哈希表向所述服务器集群同步;
- [108] 所述服务器集群根据各所述非全量的哈希表, 得到全量的哈希表。
- [109] 可选地, 所述第二生成模块 603 根据所述哈希表, 生成随机序列前, 所述服务器
- 10 器集群将所述全量的哈希表向各所述工作机同步, 以便各所述工作机根据所述全量的哈希表, 生成随机序列。
- [110] 可选地, 所述第二生成模块 603 根据所述哈希表, 生成随机序列, 具体包括:
- [111] 所述工作机在所述各节点的标识中, 随机确定一个标识, 作为目标节点的标识;
- [112] 以所述目标节点的标识为键, 在所述哈希表中索引得到对应的值, 所述对应的
- 15 值包括所述目标节点的相邻节点的标识;
- [113] 确定所述对应的值包括的相邻节点的标识的数量;
- [114] 随机确定一个小于所述数量的非负整数, 并获取所述对应的值包括的第所述非负整数个相邻节点的标识;
- [115] 通过将该第所述非负整数个相邻节点重新作为目标节点进行迭代计算, 生成由
- 20 依次得到的各目标节点的标识构成的随机序列。
- [116] 可选地, 所述对应的值存储于数组, 所述目标节点的第 n 个相邻节点的标识为该数组的第 n 个元素, n 从 0 开始计数;
- [117] 所述非负整数记作 j , 所述第二生成模块 603 获取所述对应的值包括的第所述非负整数个相邻节点的标识, 具体包括:
- 25 [118] 通过读取该数组的第 j 个元素, 获取所述目标节点的第 j 个相邻节点的标识。
- [119] 可选地, 所述工作机生成由依次得到的各目标节点的标识构成的随机序列, 具体包括:

[120] 所述工作机当依次得到的各目标节点总数量达到预设的随机游走步数时，生成由所述依次得到的各目标节点的标识构成的随机序列。

[121] 可选地，所述第二生成模块 603 生成随机序列，具体包括：

[122] 各所述工作机分别生成随机序列，直至生成的随机序列总数量达到设定阈值。

5 [123] 可选地，所述工作机若本地已有的所述哈希表丢失，则重新从所述服务器集群获取。

[124] 图 7 为本说明书实施例提供的对应于图 5 的一种随机游走装置的结构示意图，该装置包括：

[125] 获取模块 701，获取根据图数据包含的各节点的信息生成的哈希表，所述哈希表
10 的键包括所述节点的标识，值包括所述节点的相邻节点的标识；

[126] 生成模块 702，根据所述哈希表，生成随机序列，所述随机序列反映在所述图数据中的随机游走。

[127] 基于同样的思路，本说明书实施例还提供了对应于图 2 的一种基于集群的随机游走设备，该设备属于所述集群，包括：

15 [128] 至少一个处理器；以及，

[129] 与所述至少一个处理器通信连接的存储器；其中，

[130] 所述存储器存储有可被所述至少一个处理器执行的指令，所述指令被所述至少一个处理器执行，以使所述至少一个处理器能够：

[131] 获取图数据包含的各节点的信息；

20 [132] 根据所述各节点的信息，生成哈希表，所述哈希表的键包括所述节点的标识，值包括所述节点的相邻节点的标识；

[133] 根据所述哈希表，生成随机序列，所述随机序列反映在所述图数据中的随机游走。

[134] 基于同样的思路，本说明书实施例还提供了对应于图 5 的一种随机游走设备，
25 包括：

[135] 至少一个处理器；以及，

[136] 与所述至少一个处理器通信连接的存储器；其中，

- [137] 所述存储器存储有可被所述至少一个处理器执行的指令，所述指令被所述至少一个处理器执行，以使所述至少一个处理器能够：
- [138] 获取根据图数据包含的各节点的信息生成的哈希表，所述哈希表的键包括所述节点的标识，值包括所述节点的相邻节点的标识；
- 5 [139] 根据所述哈希表，生成随机序列，所述随机序列反映在所述图数据中的随机游走。
- [140] 基于同样的思路，本说明书实施例还提供了对应于图 2 的一种非易失性计算机存储介质，存储有计算机可执行指令，所述计算机可执行指令设置为：
- [141] 获取图数据包含的各节点的信息；
- 10 [142] 根据所述各节点的信息，生成哈希表，所述哈希表的键包括所述节点的标识，值包括所述节点的相邻节点的标识；
- [143] 根据所述哈希表，生成随机序列，所述随机序列反映在所述图数据中的随机游走。
- [144] 基于同样的思路，本说明书实施例还提供了对应于图 5 的一种非易失性计算机存储介质，存储有计算机可执行指令，所述计算机可执行指令设置为：
- 15 [145] 获取根据图数据包含的各节点的信息生成的哈希表，所述哈希表的键包括所述节点的标识，值包括所述节点的相邻节点的标识；
- [146] 根据所述哈希表，生成随机序列，所述随机序列反映在所述图数据中的随机游走。
- 20 [147] 上述对本说明书特定实施例进行了描述。其它实施例在所附权利要求书的范围内。在一些情况下，在权利要求书中记载的动作或步骤可以按照不同于实施例中的顺序来执行并且仍然可以实现期望的结果。另外，在附图中描绘的过程不一定要求示出的特定顺序或者连续顺序才能实现期望的结果。在某些实施方式中，多任务处理和并行处理也是可以的或者可能是有利的。
- 25 [148] 本说明书中的各个实施例均采用递进的方式描述，各个实施例之间相同相似的部分互相参见即可，每个实施例重点说明的都是与其他实施例的不同之处。尤其，对于装置、设备、非易失性计算机存储介质实施例而言，由于其基本相似于方法实施例，所以描述的比较简单，相关之处参见方法实施例的部分说明即可。

[149] 本说明书实施例提供的装置、设备、非易失性计算机存储介质与方法是对应的，因此，装置、设备、非易失性计算机存储介质也具有与对应方法类似的有益技术效果，由于上面已经对方法的有益技术效果进行了详细说明，因此，这里不再赘述对应装置、设备、非易失性计算机存储介质的有益技术效果。

- 5 [150] 在 20 世纪 90 年代，对于一个技术的改进可以很明显地区分是硬件上的改进（例如，对二极管、晶体管、开关等电路结构的改进）还是软件上的改进（对于方法流程的改进）。然而，随着技术的发展，当今的很多方法流程的改进已经可以视为硬件电路结构的直接改进。设计人员几乎都通过将改进的方法流程编程到硬件电路中来得到相应的硬件电路结构。因此，不能说一个方法流程的改进就不能用硬件实体模块来实现。例如，
- 10 可编程逻辑器件（Programmable Logic Device, PLD）（例如现场可编程门阵列（Field Programmable Gate Array, FPGA））就是这样一种集成电路，其逻辑功能由用户对器件编程来确定。由设计人员自行编程来把一个数字系统“集成”在一片 PLD 上，而不需要请芯片制造厂商来设计和制作专用的集成电路芯片。而且，如今，取代手工地制作集成电路芯片，这种编程也多半改用“逻辑编译器（logic compiler）”软件来实现，它与
- 15 程序开发撰写时所用的软件编译器相类似，而要编译之前的原始代码也得用特定的编程语言来撰写，此称之为硬件描述语言（Hardware Description Language, HDL），而 HDL 也并非仅有一种，而是有许多种，如 ABEL（Advanced Boolean Expression Language）、AHDL（Altera Hardware Description Language）、Confluence、CUPL（Cornell University Programming Language）、HDCal、JHDL（Java Hardware Description Language）、Lava、
- 20 Lola、MyHDL、PALASM、RHDL（Ruby Hardware Description Language）等，目前最普遍使用的是 VHDL（Very-High-Speed Integrated Circuit Hardware Description Language）与 Verilog。本领域技术人员也应该清楚，只需要将方法流程用上述几种硬件描述语言稍作逻辑编程并编程到集成电路中，就可以很容易得到实现该逻辑方法流程的硬件电路。
- 25 [151] 控制器可以按任何适当的方式实现，例如，控制器可以采取例如微处理器或处理器以及存储可由该（微）处理器执行的计算机可读程序代码（例如软件或固件）的计算机可读介质、逻辑门、开关、专用集成电路（Application Specific Integrated Circuit, ASIC）、可编程逻辑控制器和嵌入微控制器的形式，控制器的例子包括但不限于以下微控制器：ARC 625D、Atmel AT91SAM、Microchip PIC18F26K20 以及 Silicone Labs
- 30 C8051F320，存储器控制器还可以被实现为存储器的控制逻辑的一部分。本领域技术人员也知道，除了以纯计算机可读程序代码方式实现控制器以外，完全可以通过将方法步

骤进行逻辑编程来使得控制器以逻辑门、开关、专用集成电路、可编程逻辑控制器和嵌入微控制器等的形式来实现相同功能。因此这种控制器可以被认为是一种硬件部件，而对其内包括的用于实现各种功能的装置也可以视为硬件部件内的结构。或者甚至，可以将用于实现各种功能的装置视为既可以是实现方法的软件模块又可以是硬件部件内的结构。

[152] 上述实施例阐明的系统、装置、模块或单元，具体可以由计算机芯片或实体实现，或者由具有某种功能的产品来实现。一种典型的实现设备为计算机。具体的，计算机例如可以为个人计算机、膝上型计算机、蜂窝电话、相机电话、智能电话、个人数字助理、媒体播放器、导航设备、电子邮件设备、游戏控制台、平板计算机、可穿戴设备或者这些设备中的任何设备的组合。

[153] 为了描述的方便，描述以上装置时以功能分为各种单元分别描述。当然，在实施本说明书时可以把各单元的功能在同一个或多个软件和/或硬件中实现。

[154] 本领域内的技术人员应明白，本说明书实施例可提供为方法、系统、或计算机程序产品。因此，本说明书实施例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，本说明书实施例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。

[155] 本说明书是参照根据本说明书实施例的方法、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。

[156] 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。

[157] 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。

5 [158] 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。

[159] 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。

10 [160] 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, 15 磁带磁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。

[161] 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, 20 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。

[162] 本说明书可以在由计算机执行的计算机可执行指令的一般上下文中描述,例如程序模块。一般地,程序模块包括执行特定任务或实现特定抽象数据类型的例程、程序、 25 对象、组件、数据结构等等。也可以在分布式计算环境中实践本说明书,在这些分布式计算环境中,由通过通信网络而被连接的远程处理设备来执行任务。在分布式计算环境中,程序模块可以位于包括存储设备在内的本地和远程计算机存储介质中。

[163] 本说明书中的各个实施例均采用递进的方式描述,各个实施例之间相同相似的部分互相参见即可,每个实施例重点说明的都是与其他实施例的不同之处。尤其,对于

系统实施例而言，由于其基本相似于方法实施例，所以描述的比较简单，相关之处参见方法实施例的部分说明即可。

- [164] 以上所述仅为本说明书实施例而已，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、
- 5 等同替换、改进等，均应包含在本申请的权利要求范围之内。