

分布式系统的数据同步方法、装置、介质、电子设备

本申请要求于 2019 年 7 月 15 日提交国家知识产权局、申请号为 201910636195.1, 申请名称为“分布式系统的数据同步方法、装置、介质、电子设备”的中国专利申请的优先权,其全部内容通过引用结合在本申请中。

技术领域

本申请涉及计算机技术领域,特别地,涉及一种分布式系统的数据同步方法、装置、介质和电子设备。

背景技术

分布式系统是由一组通过网络进行通信、为了完成共同的任务而协调工作的计算机节点组成的系统。在分布式系统中,每一个节点都被用来计算或者存储数据,并且在分布式系统中可能需要在节点之间进行数据同步的操作,比如在区块链系统中,当有新的区块链节点加入时需要从其它区块链节点中同步数据。

发明内容

本申请的目的在于提供一种分布式系统的数据同步方法及装置、计算机可读存储介质、电子设备,进而至少可以缩短分布式系统中数据同步的时间,提高同步效率。

本申请的其他特性和优点将通过下面的详细描述变得显然,或部分地通过本申请的实践而习得。

根据本申请实施例的一个方面,提供了一种分布式系统的数据同步方法,所述方法包括:

获取与目标节点相连的邻接节点的历史数据同步信息;

根据所述历史数据同步信息确定所述邻接节点的选择权重值,所述选择权重值用于衡量所述邻接节点向所述目标节点同步数据的效率;

根据所述邻接节点的选择权重值,选择用于向所述目标节点同步数据的目标邻接节点,以将所述目标邻接节点中的目标数据同步至所述目

标节点。

根据本申请实施例的一个方面，提供了一种分布式系统的数据同步装置，包括：获取模块，被设置为获取与目标节点相连的邻接节点的历史数据同步信息；确定模块，被设置为根据所述历史数据同步信息确定所述邻接节点的选择权重值，所述选择权重值用于衡量所述邻接节点向所述目标节点同步数据的效率；选择模块，被设置为根据所述邻接节点的选择权重值，选择用于向所述目标节点同步数据的目标邻接节点；同步模块，被设置为将所述目标邻接节点中的目标数据同步至所述目标节点。

根据本申请实施例的一个方面，提供了一种计算机可读存储介质，其上存储有计算机程序，该计算机程序包括可执行指令，当该可执行指令被处理器执行时，实现如上述实施例中所述的分布式系统的数据同步方法。

根据本申请实施例的一个方面，提供了一种电子设备，包括：一个或多个处理器；存储器，用于存储所述处理器的可执行指令，当所述可执行指令被所述一个或多个处理器执行时，使得所述一个或多个处理器实现如上述实施例中所述的分布式系统的数据同步方法。

附图说明

通过参照附图详细描述其示例性实施例，本申请的上述和其它特征及优点将变得更加明显。

图1是根据本申请实施例示出的分布式系统的数据同步方法的应用场景示意图；

图2是根据本申请实施例的示出的分布式系统的数据同步的流程图；

图3是根据本申请实施例示出的图2中所示的步骤210的细节流程图；

图4是根据本申请实施例示出的图2中所示的步骤220的细节流程图；

图5是根据本申请实施例示出的图2中所示的步骤230的细节流程图

图；

图 6 是根据本申请实施例示出的区块头数据同步的细节流程图；

图 7 是根据本申请实施例示出的完整区块数据同步的细节流程图；

图 8 是根据本申请实施例示出的一种分布式系统的数据同步装置的框图；

图 9 是根据本申请实施例示出的一种实现上述方法的计算机可读存储介质；以及

图 10 是根据本申请实施例示出的一种实现上述方法的电子设备示例框图。

具体实施方式

现在将参考附图更全面地描述示例性实施例。然而，示例性实施例能够以多种形式实施，且不应被理解为限于在此阐述的实施方式；相反，提供这些实施方式使得本申请将全面和完整，并将示例性实施例的构思全面地传达给本领域的技术人员。在图中，为了清晰，可能会夸大部分元件的尺寸或加以变形。在图中相同的附图标记表示相同或类似的结构，因而将省略它们的详细描述。

此外，所描述的特征、结构或特性可以以任何合适的方式结合在一个或更多实施例中。在下面的描述中，提供许多具体细节从而给出对本申请的实施例的充分理解。然而，本领域技术人员将意识到，可以实践本申请的技术方案而没有所述特定细节中的一个或更多，或者可以采用其它的方法、元件等。在其它情况下，不详细示出或描述公知结构、方法或者操作以避免模糊本申请的各方面。

在目前的分布式系统中，节点之间进行数据同步操作通常存在数据同步速度慢、同步效率较低的问题。

有鉴于此，本申请提供了一种分布式系统的数据同步方法、装置、介质和电子设备，能够解决上述的分布式系统中节点的数据同步速度慢、数据同步效率低的技术问题。

首先，说明分布式系统的数据同步方法的应用场景。

图 1 所示出的是分布式系统的数据同步方法的应用场景示意图。

如图所示，场景 100 是分布式系统中的一个局部，其中，电子设备 101、102、103、104 以及 105 分别是分布式系统中的一个节点，所述电子设备可以是智能手机、平板电脑、笔记本电脑等便携移动设备，也可以是计算机设备、现场终端、台式电脑、服务器、工作站等固定式设备。本申请的实施环境可以是整个分布式系统中的任何一个节点。如图 1 所示，可以将智能手机 101 作为数据同步的目标节点，则台式电脑 102、笔记本电脑 103、平板电脑 104 以及工作站 105 将作为智能手机 101 的邻接节点。根据本申请，智能手机 101 在进行数据同步的过程中，会首先分别同时获取与台式电脑 102、笔记本电脑 103、平板电脑 104 以及工作站 105 在历史上进行数据同步的历史数据同步信息，并根据所述历史数据同步信息确定用于衡量所述邻接节点向所述智能手机 101 同步数据的效率的选择权重值，最后根据所述选择权重值选择用于向所述智能手机 101 同步数据的目标邻接节点，并完成数据同步的过程。

需要注意的是，在本申请中，分布式系统中各节点之间的数据传输可以通过通信连接方式的连接（例如有线通信通信网连接或者无线通信通信网连接）来实现。

根据本申请的第一方面，提供了一种分布式系统的数据同步方法。

参见图 2，示出了分布式系统的数据同步方法的流程图，该数据同步方法的执行主体可以是分布式系统中包含的需要进行数据同步的节点，比如可以是充当任意一个节点的电子设备。该数据同步方法可包括如下步骤：

步骤 210，获取与目标节点相连的邻接节点的历史数据同步信息。

步骤 220，根据所述历史数据同步信息确定所述邻接节点的选择权重值，所述选择权重值用于衡量所述邻接节点向所述目标节点同步数据的效率。

步骤 230，根据所述邻接节点的选择权重值，选择用于向所述目标节点同步数据的目标邻接节点，以将所述目标邻接节点中的目标数据同步至所述目标节点。

下面将对如上实施步骤进行详细说明。

关于步骤 210：获取与目标节点相连的邻接节点的历史数据同步信息。

具体地，所述目标节点是指在分布式系统中需要进行数据同步的节点，例如在区块链系统中的任意一个节点。所述邻接节点是指在所述分布式系统中与所述目标节点相连接的其它节点。

在一示例性实施例中，步骤 210 中获取与目标节点相连的邻接节点的历史数据同步信息的过程可以通过如图 3 所示的方式完成，具体包括以下步骤：

步骤 2101，检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的目标数据，得到需要向所述目标节点同步目标数据的候选邻接节点。

在一实施例的具体实现中，所述分布式系统可以包括区块链系统，所述目标数据可以包括区块数据。所述检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据，可以通过如下方式实现：

首先，获取目标节点中区块链账本的第一区块头高度。然后获取与目标节点相连的邻接节点中区块链账本的第二区块头高度。最后比较所述第一区块头高度与所述第二区块头高度的大小，以检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据。

具体而言，区块链系统是分布式系统的一种，其主要作用是储存数据，任何需要保存的数据，都可以写入区块链，也可以从区块链里面读取。区块链中的数据主要以区块的形式存在，即一个区块是整个区块链数据中的基本单位，一个区块包括一个区块头和一个区块体，其中，区块头中记录着当前区块的特征值数据，例如当前时间、区块头高度、上一个区块的哈希以及区块体的哈希等等，而区块体则记录着需要保存的实际数据。区块链中的区块按照严格的顺序单向连接从而形成一个数据链条。在区块链系统中每隔一段时间就会产生一个区块，产生的新区块连接在上一个产生的区块之后，随着时间的推移，区块链数据会不断的

更新。此外，在区块链系统中，当新的服务器加入区块链网络成为一个节点之后，新加入的节点需要同步区块链系统中的区块数据，并且随着区块链数据的不断更新，区块链系统中的各个节点也需要不断地更新区块数据。

在区块链数据中，每一个区块数据都是独一无二的，因此每一个区块数据的身份可以通过区块头中的区块头高度或者哈希值来进行标识，其中，区块链数据中的区块头高度随着区块的不断更新而逐步递增，即每产生一个区块，产生的新区块的区块头高度就会增加 1 个。基于此，本申请通过利用区块头高度来检测区块链系统中与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据。

具体地，分别获取目标节点中区块链账本的第一区块头高度和与目标节点相连的邻接节点中区块链账本的第二区块头高度，并比较所述第一区块头高度和第二区块头高度的大小，若第二区块头高度大于第一区块头高度，则说明与所述目标节点相连的邻接节点中存在需要向所述目标节点进行同步的区块数据。

在获取与目标节点相连的邻接节点的历史数据同步信息的过程中，通过检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据，进而能得到需要向所述目标节点同步数据的候选邻接节点。这样做的优点在于，仅仅确定候选邻接节点的选择权重值，并从所述候选邻接节点中选择目标邻接节点，可以过滤掉一些没有数据同步前景的邻接节点，从而减少获取一些没有必要的历史数据同步信息，减轻系统的存储和计算压力，同时也提高了所述历史数据同步信息的利用率。

步骤 2102，获取所述候选邻接节点的历史数据同步信息。

在如上所述的实施例，检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据，得到需要向所述目标节点同步数据的候选邻接节点的好处就在于，在获取所述邻接节点的历史数据同步信息之前，可以通过检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据来进一步地在所有邻接

节点中确定具有数据同步前景的候选邻接节点，这样做的优点在于，可以过滤掉一些没有数据同步前景的邻接节点，从而减少获取一些没有必要的历史数据同步信息，减轻系统的存储和计算压力，同时也提高了所述历史数据同步信息的利用率。

关于步骤 220：根据所述历史数据同步信息确定所述邻接节点的选择权重值，所述选择权重值用于衡量所述邻接节点向所述目标节点同步数据的效率。

在一示例性实施例中，所述历史数据同步信息可以包括目标节点与所述邻接节点在历史上同步数据过程中的出错次数，步骤 220 中根据所述历史数据同步信息确定所述邻接节点的选择权重值的过程可以如图 4 所示，具体包括以下步骤：

步骤 2201，检测所述出错次数未超过出错次数阈值的邻接节点。

在本实施例中，所述出错次数是指目标节点与所述邻接节点在历史上同步数据过程中的出错次数。具体而言，分布式系统中的节点与节点之间在进行数据同步时，由于节点本身故障或者数据传输网络故障等原因，难免会出现节点同步失败的情况发生，例如数据同步请求超时、数据在同步下载过程中出现异常以及对节点访问失败等等。

本实施例可以通过设置一个所述出错次数的阈值，对与目标节点相连的邻接节点进行一个筛选，即对所述出错次数超过某一定数目的邻接节点进行过滤，对所述出错次数未超过某一定数目的邻接节点进行保留，最终将被保留下来的邻接节点作为进行后续目标邻接节点确定的一个候选。所述出错次数可以根据实际情况来进行设定，例如设置为 3 次。若邻接节点 1、邻接节点 2、邻接节点 3 以及邻接节点 4 在历史上同步数据过程中的出错次数分别为 4 次、3 次、1 次以及 0 次，则最终被保留下来的邻接节点就可以是邻接节点 2、邻接节点 3 以及邻接节点 4。

步骤 2202，根据所述历史数据同步信息确定所述出错次数未超过所述出错次数阈值的邻接节点的选择权重值。

根据如上所述的实施例，通过设置出错次数阈值来过滤掉数据同步经常出错的邻接节点，保留所述出错次数较少的邻接节点，可以提高数

据在后续同步过程中的同步成功率，此外还可以减少系统计算所述选择权重值的负担，进而减轻系统计算压力，提高数据同步效率。

在一示例性实施例中，所述历史数据同步信息可以包括历史数据传输速度，则所述根据所述历史数据同步信息确定所述邻接节点的选择权重值可以通过以下公式计算所述邻接节点的选择权重值：

$$W_i = \frac{\sum_{j=m+1}^{j=m+n} S_j}{n}$$

其中， W_i 表示第*i*个邻接节点的选择权重值；*j*表示目标节点与第*i*个邻接节点在历史上第*j*次进行数据同步；*m*表示目标节点与第*i*个邻接节点在历史上第*m*次进行数据同步；*n*表示目标节点与第*i*个邻接节点在历史上最近*n*次进行数据同步的次数； S_j 表示目标节点与第*i*个邻接节点在历史上第*j*次进行数据同步过程中的历史数据传输速度。

具体而言，所述历史数据传输速度可以是指历史上的数据同步过程中所传输的目标数据的大小与数据传输时间的比值，而数据传输时间又可以是指目标节点在完全接收到所述目标数据时的时刻与向目标邻接节点请求下载所述目标数据时的时刻之间的差值。例如在某一次数据同步的过程中，目标节点向目标邻接节点请求下载目标数据时的时刻为2019年7月1日12时00分00秒，目标节点在完全接收所述目标数据时的时刻为2019年7月1日12时00分03秒，则数据传输时间为3.00秒，若所述目标数据的大小为30Mb，则此次历史数据传输速度为10 Mb/s。

在一实施例的具体实现中，可以要求将目标节点与一邻接节点在历史上最近5次数据同步过程中数据传输速度的平均值作为所述邻接节点的选择权重值，如表1所示，目标节点与一邻接节点在历史上最近5次数据同步过程中的数据传输速度分别为：10 Mb/s、30 Mb/s、20 Mb/s、15 Mb/s、25 Mb/s。所以通过计算，可以确定所述选择权重值为： $W_i=20$ 。

<i>j</i>	S_j
m+5	10 Mb/s
m+4	30 Mb/s
m+3	20 Mb/s

m+2	15 Mb/s
m+1	25 Mb/s

表 1

根据如上所述的实施例，通过计算所述邻接节点在历史上最近 n 次数据同步的平均数据传输速度来作为确定所述选择权重值的一个因素，可以很好地衡量所述邻接节点的数据同步效率，从而更有针对性地选择目标邻接节点，进而提高数据同步效率。

在一示例性实施例中，所述历史数据同步信息可以包括历史数据传输速度和历史数据请求时刻，则所述根据所述历史数据同步信息确定所述邻接节点的选择权重值可以通过以下公式计算所述邻接节点的选择权重值：

$$W_i = \frac{\sum_{j=m+1}^{j=m+n} S_j + \sum_{j=m+1}^{j=m+n} (t - t_j)}{n}$$

其中， W_i 表示第 i 个邻接节点的选择权重值； j 表示目标节点与第 i 个邻接节点在历史上第 j 次进行数据同步； m 表示目标节点与第 i 个邻接节点在历史上第 m 次进行数据同步； n 表示目标节点与第 i 个邻接节点在历史上最近 n 次进行数据同步的次数； S_j 表示目标节点与第 i 个邻接节点在历史上第 j 次进行数据同步过程中的历史数据传输速度； t_j 表示目标节点与第 i 个邻接节点在历史上第 j 次进行数据同步时的历史数据请求时刻； t 表示计算所述第 i 个邻接节点的选择权重值时的时刻。

具体而言，在本实施例中，所述历史数据传输速度也可以是指数据同步过程中所传输的目标数据的大小与数据传输时间的比值。所述历史数据请求时刻可以是指历史上的数据同步过程中目标节点向第 i 个邻接节点请求下载目标数据时的时刻，在本实施例中主要用于衡量计算所述第 i 个邻接节点的选择权重值时的时刻距离目标节点与第 i 个邻接节点在历史上第 j 次进行数据同步时的历史数据请求时刻之间的时间长度，所述时间长度可以衡量所述目标节点与第 i 个邻接节点之间进行数据同步的频繁程度，即所述时间长度越短，即表示所述目标节点与第 i 个邻接节点之间进行数据同步越频繁。

在一实施例的具体实现中，可以要求将目标节点与第 i 个邻接节点在历史上最近 5 次数据同步过程中数据传输速度的平均值与计算所述第 i 个邻接节点的选择权重值时的时刻距离目标节点与第 i 个邻接节点在历史上最近 5 次进行数据同步时的历史数据请求时刻之间的时间长度的平均值的和作为所述邻接节点的选择权重值。如表 2 所示，目标节点与第 i 个邻接节点在历史上最近 5 次数据同步过程中的数据传输速度分别为：10 Mb/s、30 Mb/s、20 Mb/s、15 Mb/s、25 Mb/s。目标节点在历史上最近 5 次向第 i 个邻接节点请求下载目标数据时的时刻分别为：2019 年 7 月 1 日 12 时 00 分 00 秒、2019 年 7 月 1 日 12 时 00 分 10 秒、2019 年 7 月 1 日 12 时 00 分 20 秒、2019 年 7 月 1 日 12 时 00 分 30 秒、2019 年 7 月 1 日 12 时 00 分 40 秒。

j	S_j	t_j
m+5	10 Mb/s	2019 年 7 月 1 日 12 时 00 分 00 秒
m+4	30 Mb/s	2019 年 7 月 1 日 12 时 00 分 10 秒
m+3	20 Mb/s	2019 年 7 月 1 日 12 时 00 分 20 秒
m+2	15 Mb/s	2019 年 7 月 1 日 12 时 00 分 30 秒
m+1	25 Mb/s	2019 年 7 月 1 日 12 时 00 分 40 秒

表 2

若计算所述第 i 个邻接节点的选择权重值时的时刻 t 为 2019 年 7 月 1 日 12 时 00 分 50 秒，则通过本实施例的计算公式可以计算出所述邻接节点的选择权重值为： $W_i=50$ 。

根据如上所述的实施例，将通过计算所述邻接节点在历史上最近 n 次数据同步的平均数据传输速度来作为确定所述选择权重值的一个因素，可以很好地衡量所述邻接节点的数据同步效率。但是，如果某一个邻接节点的数据传输速度过高，就会导致其他节点总是选择这一个邻接节点作为目标邻接节点，由此带来节点负载过重而导致数据同步效率下降。而对于数据传输速度较高的邻接节点来说，被选作目标邻接节点的次数就会较为频繁，进一步地，计算这个邻接节点的选择权重值时的时刻距离目标节点与这个邻接节点在历史上第 j 次进行数据同步时的历史

数据请求时刻之间的时间长度就会较短。基于此，将所述目标节点向所述邻接节点请求同步数据的历史数据请求时刻来作为确定所述选择权重值的另一个因素，可以均衡分布式系统中的与目标节点相连的邻接节点资源，以防止某一个邻接节点被过度利用而导致的所述邻接节点负载过重的情况发生，基于如上实施例，可以选择出较优的邻接节点。

如上所述，可以理解的是，根据所述历史数据同步信息确定所述邻接节点的选择权重值的方式可以是任意的，并不限于如上所示出的那些示例。

关于步骤 230：根据所述邻接节点的选择权重值，选择用于向所述目标节点同步数据的目标邻接节点，以将所述目标邻接节点中的目标数据同步至所述目标节点。

在一示例性实施例中，根据所述邻接节点的选择权重值，选择用于向所述目标节点同步数据的目标邻接节点，可以包括如图 5 所示的步骤，

图 5 是根据本申请实施例示出的步骤 230 的细节流程图，具体包括以下步骤：

步骤 2301，检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据，得到需要向所述目标节点同步数据的候选邻接节点。

在一实施例的具体实现中，所述分布式系统可以包括区块链系统，所述目标数据可以包括区块数据，所述检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据，可以通过如下方式实现：

分别获取目标节点中区块链账本的第一区块头高度和与目标节点相连的邻接节点中区块链账本的第二区块头高度。然后比较所述第一区块头高度与所述第二区块头高度的大小，以检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据，若第二区块头高度大于第一区块头高度，则说明与所述目标节点相连的邻接节点中存在需要向所述目标节点进行同步的区块数据，并将存在向所述目标节点进行区块数据同步的邻接节点作为候选邻接节点。

步骤 2302, 根据所述候选邻接节点的选择权重值, 选择用于向所述目标节点同步数据的目标邻接节点。

在一实施例的具体实现中, 根据所述候选邻接节点的选择权重值, 选择用于向所述目标节点同步数据的目标邻接节点可以通过如下方式实现:

根据所述候选邻接节点的选择权重值, 对所述候选邻接节点按照从大到小的顺序进行排序, 将所述选择权重值最大的邻接节点作为目标邻接节点。

根据如上所述的实施例, 通过检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据来得到需要向所述目标节点同步数据的候选邻接节点, 可以避免某一些被选择的目标邻接节点因为数据不全而导致数据同步失败的情况发生, 进而减少计算机网络资源的浪费。

在一示例性实施例中, 所述分布式系统可以包括区块链系统, 所述目标数据可以包括完整区块数据, 所述根据所述邻接节点的选择权重值, 选择用于向所述目标节点同步数据的目标邻接节点, 以将所述目标邻接节点中的目标数据同步至所述目标节点可以通过如下方式实现:

根据所述邻接节点的选择权重值, 选择用于向所述目标节点同步所述完整区块数据的第二目标邻接节点, 以从所述第二目标邻接节点中同步所述完整区块数据。

在一示例性实施例中, 所述分布式系统可以包括区块链系统, 所述目标数据可以包括区块头数据和完整区块数据, 所述根据所述邻接节点的选择权重值, 选择用于向所述目标节点同步数据的目标邻接节点, 以将所述目标邻接节点中的目标数据同步至所述目标节点可以通过如下方式实现:

根据所述邻接节点的选择权重值, 选择用于向所述目标节点同步所述区块头数据的第一目标邻接节点, 以从所述第一目标邻接节点中同步所述区块头数据,

在同步所述区块头数据之后, 根据所述邻接节点的选择权重值, 选

择用于向所述目标节点同步所述完整区块数据的第二目标邻接节点，以从所述第二目标邻接节点中同步所述完整区块数据。

在一实施例的具体实现中，在选择用于向所述目标节点同步所述完整区块数据的第二目标邻接节点之前，还可以实现如下步骤：

获取目标节点中区块链账本的具有完整区块数据的第三区块头高度；

获取目标节点中区块链账本的第四区块头高度；

比较所述第三区块头高度与所述第四区块头高度的大小，以检测所述目标节点是否需要选择用于向所述目标节点同步所述完整区块数据的第二目标邻接节点，若所述第三区块头高度小于第四区块头高度，则说明所述目标节点需要选择用于向所述目标节点同步所述完整区块数据的第二目标邻接节点。

需要说明的是，在目标节点中的区块链账本中，存在着具有完整区块数据的区块和不具有完整区块数据的区块，其中，具有完整区块数据的区块包含有区块头和区块体，而不具有完整区块数据的区块仅仅只包含区块头。因此，在本具体实施例中，所述第三区块头高度是指具有完整区块数据的区块的区块头的高度，所述第四区块头高度是指不具有完整区块数据的区块的区块头的高度，当具有完整区块数据的区块的区块头的高度小于不具有完整区块数据的区块的区块头的高度时，则说明所述目标节点中存在不具有完整区块数据的区块，进而说明所述目标节点需要选择用于向所述目标节点同步所述完整区块数据的第二目标邻接节点。

以下以分布式系统为区块链系统为例，对本申请实施例中的数据同步方案进行详细阐述，其中，在该实施例中，以同步区块数据的过程包括同步区块头数据的过程和同步完整区块数据的过程为例进行说明：

在一个实施例的具体实现中，目标节点同步区块头数据的具体实现可以按照如图6所示的流程循环实施，所述流程具体包括如下步骤：

步骤610，获取本地区块链账本的区块头高度。

步骤620，计算与目标节点相连的各个邻接节点的选择权重值，并

排序。

步骤 630，检查所有邻接节点中是否含有本节点需要的下一个的区块头数据，若是，则执行步骤 640；若否，则执行步骤 610，以进行一次同步循环。

步骤 640，确定选择权重值最高的且存在所需区块头数据的邻接节点。

步骤 650，向所述邻接节点发送区块头数据同步下载的请求，并记录所述请求的时刻，删除最早一次的数据同步下载请求的时刻。

步骤 660，接收区块头数据，将合法的区块头数据写入本地的区块链账本。

步骤 670，计算本次区块头数据同步的数据传输速度。

需要注意的是，为了使本领域技术人员更加容易地理解如上所述步骤 650，这里通过一具体的示例加以说明。例如，在向所述邻接节点发送区块头数据同步下载的请求之前，在历史上已存在有 3 次向所述邻接节点发送区块头数据同步下载的请求记录，且所述 3 次请求记录的请求时刻分别为 2019 年 7 月 1 日 12 时 00 分 00 秒、2019 年 7 月 1 日 12 时 01 分 01 秒、2019 年 7 月 1 日 12 时 02 分 01 秒，其中 2019 年 7 月 1 日 12 时 00 分 00 秒是第最早一次向所述邻接节点发送区块头数据同步下载请求的请求时刻。当 2019 年 7 月 1 日 12 时 03 分 01 秒再一次向所述邻接节点发送区块头数据同步下载请求时，则记录这一时刻，即记录 2019 年 7 月 1 日 12 时 03 分 01 秒，并删除最早一次向所述邻接节点发送区块头数据同步下载请求的请求时刻，即删除 2019 年 7 月 1 日 12 时 00 分 00 秒。最终所记录的最近 3 次向所述邻接节点发送区块头数据同步下载请求的请求时刻分别为：2019 年 7 月 1 日 12 时 01 分 01 秒、2019 年 7 月 1 日 12 时 02 分 01 秒以及 2019 年 7 月 1 日 12 时 03 分 01 秒。

在一个实施例的具体实现中，目标节点同步完整区块数据的具体实现可以按照如图 7 所示的流程循环实施，所述流程具体包括如下步骤：

步骤 710，获取本地区块链账本的区块头高度和完整区块数据的区块头高度。

步骤 720, 检测所述区块头高度是否高于完整区块数据的区块头高度, 若是, 则执行步骤 730; 若否, 则执行步骤 710, 以进行下一次同步循环。

步骤 730, 计算与目标节点相连的各个邻接节点的选择权重值, 并排序。

步骤 740, 检测排序后的邻接节点中是否含有所述目标节点的下一个完整区块数据, 若是, 则执行步骤 750; 若否, 则执行步骤 710, 以进行下一次同步循环。

步骤 750, 确定选择权重值最高的且存在所需完整区块数据的邻接节点。

步骤 760, 向所述邻接节点发送完整区块数据同步下载的请求, 并记录所述请求的时刻, 删除最早一次的数据同步下载请求的时刻。

步骤 770, 接收所述完整区块数据, 将合法的完整区块数据存入完整区块数据缓存区。

步骤 780, 计算本次完整区块数据同步的数据传输速度, 删除最早一次的完整区块数据同步的数据传输速度。

需要注意的是, 为了使本领域技术人员更加容易地理解如上所述步骤 760, 这里通过一具体的示例加以说明。例如, 在向所述邻接节点发送完整区块数据同步下载的请求之前, 在历史上已存在有 3 次向所述邻接节点发送完整区块数据同步下载的请求记录, 且所述 3 次请求记录的请求时刻分别为 2019 年 7 月 1 日 12 时 00 分 00 秒、2019 年 7 月 1 日 12 时 01 分 01 秒、2019 年 7 月 1 日 12 时 02 分 01 秒, 其中 2019 年 7 月 1 日 12 时 00 分 00 秒是第最早一次向所述邻接节点发送完整区块数据同步下载请求的请求时刻。当 2019 年 7 月 1 日 12 时 03 分 01 秒再一次向所述邻接节点发送完整区块数据同步下载请求时, 则记录这一时刻, 即记录 2019 年 7 月 1 日 12 时 03 分 01 秒, 并删除最早一次向所述邻接节点发送完整区块数据同步下载请求的请求时刻, 即删除 2019 年 7 月 1 日 12 时 00 分 00 秒。最终所记录的最近 3 次向所述邻接节点发送完整区块数据同步下载请求的请求时刻分别为: 2019 年 7 月 1 日 12 时 01

分 01 秒、2019 年 7 月 1 日 12 时 02 分 01 秒以及 2019 年 7 月 1 日 12 时 03 分 01 秒。

如上所述实施例，通过在同步所述完整区块数据之前同步区块头数据的优点在于两个方面。一方面，由于完整区块数据的数据量较大以及其较高的安全性要求，在对新加入到区块链系统中的目标节点进行完整区块数据同步时，特别容易出现数据同步时间长、数据同步请求被拒绝的情况发生。而区块头数据的数据量较小以及安全性要求不高，特别适合对新加入到区块链系统中的目标节点进行数据同步。另一方面，目标节点首先同步区块头数据，有利于检测所述目标节点是否需要选择用于向所述目标节点同步所述完整区块数据的目标邻接节点，当检测到所述目标节点不需要选择用于向所述目标节点同步所述完整区块数据的目标邻接节点时，所述目标节点就不需要计算邻接节点的选择权重值，进而节省计算机资源。

在一实施例的具体实现中，从所述第二目标邻接节点中同步所述完整区块数据可以通过如下方式实现：

下载所述第二目标邻接节点中的完整区块数据；验证所述完整区块数据的合法性；将合法的完整区块数据加入至所述目标节点的区块数据缓存区，以基于所述区块数据缓存区中存储的数据向所述目标节点同步所述完整区块数据。

在一实施例的具体实现中，基于所述区块数据缓存区中存储的数据向所述目标节点同步所述完整区块数据可以通过如下方式完成：

按照所述目标节点中的区块链账本所指示的区块顺序，从所述区块数据缓存区中读取相应的区块数据，并写入所述目标节点的区块链账本中。

在一示例性实施例中，将所述目标邻接节点中的目标数据同步至所述目标节点可以通过如下方式完成：

向所述目标邻接节点发出针对所述目标数据的下载请求，以使所述目标邻接节点根据所述下载请求向所述目标节点发送所述目标数据。

在一实施例的具体实现中，在将所述目标邻接节点中的目标数据同

步至所述目标节点的过程中，若对所述目标数据同步失败，则记录所述目标节点与所述目标邻接节点之间进行数据同步的出错次数。

所述目标数据同步失败可以是指数据同步请求超时，还可以是指数据在同步下载过程中出现异常，也可以是指对目标邻接节点访问的失败。

在一实施例的具体实现中，在将所述目标邻接节点中的目标数据同步至所述目标节点的过程中，若对所述目标数据同步成功，则记录所述目标节点与所述目标邻接节点之间进行数据同步的数据传输速度和数据请求时刻。

如上所述，可以理解的是，根据所述邻接节点的选择权重值，选择用于向所述目标节点同步数据的目标邻接节点，以将所述目标邻接节点中的目标数据同步至所述目标节点的方式可以是任意的，并不限于如上所示出的那些。

综上所述，本申请实施例的技术方案，通过根据与目标节点相连的邻接节点的历史数据同步信息来为各邻接节点确定一个能反映所述邻接节点向所述目标节点同步数据的同步效率的选择权重值，并根据不同邻接节点所对应的选择权重值来最终确定能够向所述目标节点同步数据的目标邻接节点，使得能够综合考虑各节点的历史数据同步信息来选择较为合适的邻接节点进行数据同步，比如可以考虑各节点的实际拥塞情况以及负载情况等信息选择出较为科学、合理的目标邻接节点，从而能够缩短分布式系统中数据同步的时间，提高同步效率。进而能够解决现有技术中存在的分布式系统中节点的数据同步速度慢，数据同步效率低的技术问题。

以下介绍本申请的装置实施例，可以用于执行本申请上述实施例中的分布式系统的数据同步方法。对于本申请装置实施例中未披露的细节，请参照本申请上述的分布式系统的数据同步方法的实施例。

图8示出了根据本申请的一个实施例的分布式系统的数据同步装置的框图。

参照图8所示，根据本申请的一个实施例的分布式系统的数据同步装置800，包括：获取模块810、确定模块820、选择模块830、同步模

块 840。

其中，获取模块 810 用于获取与目标节点相连的邻接节点的历史数据同步信息；确定模块 820 用于根据所述历史数据同步信息确定所述邻接节点的选择权重值，所述选择权重值用于衡量所述邻接节点向所述目标节点同步数据的效率；选择模块 830 用于根据所述邻接节点的选择权重值，选择用于向所述目标节点同步数据的目标邻接节点；同步模块 840 用于将所述目标邻接节点中的目标数据同步至所述目标节点。

在本申请的一些实施例中，基于前述方案，选择模块 830 配置为：检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据，得到需要向所述目标节点同步数据的候选邻接节点；根据所述候选邻接节点的选择权重值，选择用于向所述目标节点同步数据的目标邻接节点。

在本申请的一些实施例中，基于前述方案，获取模块 810 包括：检测模块，用于检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据，得到需要向所述目标节点同步数据的候选邻接节点；执行模块，用于获取所述候选邻接节点的历史数据同步信息。

在本申请的一些实施例中，基于前述方案，所述分布式系统包括区块链系统，所述目标数据包括区块数据，所述检测模块配置为：获取目标节点中区块链账本的第一区块头高度；获取与目标节点相连的邻接节点中区块链账本的第二区块头高度；比较所述第一区块头高度与所述第二区块头高度的大小，以检测与所述目标节点相连的各个邻接节点是否存在需要向所述目标节点进行同步的数据。

在本申请的一些实施例中，基于前述方案，所述历史数据同步信息包括目标节点与所述邻接节点在历史上同步数据过程中的出错次数，所述确定模块 820 配置为：检测出所述出错次数未超过出错次数阈值的邻接节点；根据所述历史数据同步信息确定所述出错次数未超过所述出错次数阈值的邻接节点的选择权重值。

在本申请的一些实施例中，基于前述方案，所述历史数据同步信息

包括历史数据传输速度和历史数据请求时刻,所述确定模块 820 配置为通过以下公式计算所述邻接节点的选择权重值:

$$W_i = \frac{\sum_{j=m+1}^{j=m+n} S_j + \sum_{j=m+1}^{j=m+n} (t - t_j)}{n}$$

其中, W_i 表示第 i 个邻接节点的选择权重值; j 表示目标节点与第 i 个邻接节点在历史上第 j 次进行数据同步; m 表示目标节点与第 i 个邻接节点在历史上第 m 次进行数据同步; n 表示目标节点与第 i 个邻接节点在历史上最近 n 次进行数据同步的次数; S_j 表示目标节点与第 i 个邻接节点在历史上第 j 次进行数据同步过程中的历史数据传输速度; t_j 表示目标节点与第 i 个邻接节点在历史上第 j 次进行数据同步时的历史数据请求时刻; t 表示计算所述第 i 个邻接节点的选择权重值时的时刻。

在本申请的一些实施例中,基于前述方案,所述同步模块 840 配置为:向所述目标邻接节点发出针对所述目标数据的下载请求,以使所述目标邻接节点根据所述下载请求向所述目标节点发送所述目标数据;所述方法还包括:在将所述目标邻接节点中的目标数据同步至所述目标节点的过程中,若对所述目标数据同步失败,则记录所述目标节点与所述目标邻接节点之间进行数据同步的出错次数。

在本申请的一些实施例中,基于前述方案,所述分布式系统包括区块链系统,所述目标数据包括完整区块数据,所述选择模块 830 配置为:根据所述邻接节点的选择权重值,选择用于向所述目标节点同步所述完整区块数据的第二目标邻接节点;所述同步模块 840 配置为:从所述第二目标邻接节点中同步所述完整区块数据。

在本申请的一些实施例中,基于前述方案,所述分布式系统包括区块链系统,所述目标数据包括区块头数据和完整区块数据,所述选择模块 830 配置为:根据所述邻接节点的选择权重值,选择用于向所述目标节点同步所述区块头数据的第一目标邻接节点;所述同步模块 840 配置为:从所述第一目标邻接节点中同步所述区块头数据;

所述选择模块 830 还配置为:在同步所述区块头数据之后,根据所述邻接节点的选择权重值,选择用于向所述目标节点同步所述完整区块

数据的第二目标邻接节点；所述同步模块 840 还配置为：从所述第二目标邻接节点中同步所述完整区块数据。

在本申请的一些实施例中，基于前述方案，所述分布式系统包括区块链系统，所述目标数据包括完整区块数据，所述选择模块 830 配置为：在选择用于向所述目标节点同步所述完整区块数据的第二目标邻接节点之前，获取目标节点中区块链账本的具有完整区块数据的第三区块头高度；获取目标节点中区块链账本的第四区块头高度；比较所述第三区块头高度与所述第四区块头高度的大小，以检测是否需要选择用于向所述目标节点同步所述完整区块数据的第二目标邻接节点。

在本申请的一些实施例中，基于前述方案，所述分布式系统包括区块链系统，所述目标数据包括完整区块数据，所述同步模块 840 配置为：下载所述第二目标邻接节点中的完整区块数据；验证所述完整区块数据的合法性；将合法的完整区块数据加入至所述目标节点的区块数据缓存区，以基于所述区块数据缓存区中存储的数据向所述目标节点同步所述完整区块数据。

在本申请的一些实施例中，基于前述方案，所述分布式系统包括区块链系统，所述目标数据包括完整区块数据，所述同步模块 840 配置为：按照所述目标节点中的区块链账本所指示的区块顺序，从所述区块数据缓存区中读取相应的区块数据，并写入所述目标节点的区块链账本。

应当注意，尽管在上文详细描述中提及了分布式系统的数据同步方法以及分布式系统的数据同步装置的若干模块，但是这种划分并非强制性的。实际上，根据本申请的实施方式，上文描述的两个或更多模块和功能可以在一个模块中具体化。反之，上文描述的一个模块的特征和功能可以进一步划分为由多个模块来具体化。作为模块显示的部件可以是或者也可以不是物理单元，即可以位于一个地方，或者也可以分布到多个网络单元上。可以根据实际的需要选择其中的部分或者全部模块来实现本申请方案的目的。本领域普通技术人员在不付出创造性劳动的情况下，即可以理解并实施。

作为另一方面，本申请还提供了一种计算机可读存储介质，其上存

储有能够实现本说明书上述方法的程序产品。在一些可能的实施方式中，本申请的各个方面还可以实现为一种程序产品的形式，其包括程序代码，当所述程序产品在终端设备上运行时，所述程序代码用于使所述终端设备执行本说明书上述“示例性方法”部分中描述的根据本申请各种示例性实施方式的步骤。

参考图 9 所示，描述了根据本申请的实施方式的用于实现上述方法的程序产品 900，其可以采用便携式紧凑盘只读存储器(CD-ROM)并包括程序代码，并可以在终端设备，例如个人电脑上运行。然而，本申请的程序产品不限于此，在本文件中，可读存储介质可以是任何包含或存储程序的有形介质，该程序可以被指令执行系统、装置或者器件使用或者与其结合使用。

所述程序产品可以采用一个或多个可读介质的任意组合。可读介质可以是可读信号介质或者可读存储介质。可读存储介质例如可以为但不限于电、磁、光、电磁、红外线、或半导体的系统、装置或器件，或者任意以上的组合。可读存储介质的更具体的例子(非穷举的列表)包括：具有一个或多个导线的电连接、便携式盘、硬盘、随机存取存储器(RAM)、只读存储器 (ROM)、可擦式可编程只读存储器 (EPROM 或闪存)、光纤、便携式紧凑盘只读存储器(CD-ROM)、光存储器件、磁存储器件、或者上述的任意合适的组合。

计算机可读信号介质可以包括在基带中或者作为载波一部分传播的数据信号，其中承载了可读程序代码。这种传播的数据信号可以采用多种形式，包括但不限于电磁信号、光信号或上述的任意合适的组合。可读信号介质还可以是可读存储介质以外的任何可读介质，该可读介质可以发送、传播或者传输用于由指令执行系统、装置或者器件使用或者与其结合使用的程序。

可读介质上包含的程序代码可以用任何适当的介质传输，包括但不限于无线、有线、光缆、RF 等等，或者上述的任意合适的组合。

可以以一种或多种程序设计语言的任意组合来编写用于执行本申请操作的程序代码，所述程序设计语言包括面向对象的程序设计语言—

诸如 Java、C++等，还包括常规的过程式程序设计语言—诸如“C”语言或类似的程序设计语言。程序代码可以完全地在用户计算设备上执行、部分地在用户设备上执行、作为一个独立的软件包执行、部分在用户计算设备上部分在远程计算设备上执行、或者完全在远程计算设备或服务服务器上执行。在涉及远程计算设备的情形中，远程计算设备可以通过任意种类的网络，包括局域网（LAN）或广域网（WAN），连接到用户计算设备，或者，可以连接到外部计算设备（例如利用因特网服务提供商来通过因特网连接）。

作为另一方面，本申请还提供了一种能够实现上述方法的电子设备。

所属技术领域的技术人员能够理解，本申请的各个方面可以实现为系统、方法或程序产品。因此，本申请的各个方面可以具体实现为以下形式，即：完全的硬件实施方式、完全的软件实施方式（包括固件、微代码等），或硬件和软件方面结合的实施方式，这里可以统称为“电路”、“模块”或“系统”。

下面参照图 10 来描述根据本申请的这种实施方式的电子设备 1000。图 10 显示的电子设备 1000 仅仅是一个示例，不应对本申请实施例的功能和使用范围带来任何限制。

如图 10 所示，电子设备 1000 以通用计算设备的形式表现。电子设备 1000 的组件可以包括但不限于：上述至少一个处理单元 1010、上述至少一个存储单元 1020、连接不同系统组件（包括存储单元 1020 和处理单元 1010）的总线 1030。

其中，所述存储单元存储有程序代码，所述程序代码可以被所述处理单元 1010 执行，使得所述处理单元 1010 执行本说明书上述“实施例方法”部分中描述的根据本申请各种示例性实施方式的步骤。

存储单元 1020 可以包括易失性存储单元形式的可读介质，例如随机存取存储单元（RAM）1021 和/或高速缓存存储单元 1022，还可以进一步包括只读存储单元（ROM）1023。

存储单元 1020 还可以包括具有一组（至少一个）程序模块 1025 的程序/实用工具 1024，这样的程序模块 1025 包括但不限于：操作系统、

一个或者多个应用程序、其它程序模块以及程序数据，这些示例中的每一个或某种组合中可能包括网络环境的实现。

总线 1030 可以为表示几类总线结构中的一种或多种，包括存储单元总线或者存储单元控制器、外围总线、图形加速端口、处理单元或者使用多种总线结构中的任意总线结构的局域总线。

电子设备 1000 也可以与一个或多个外部设备 1200（例如键盘、指向设备、蓝牙设备等）通信，还可与一个或者多个使得用户能与该电子设备 1000 交互的设备通信，和/或与使得该电子设备 1000 能与一个或多个其它计算设备进行通信的任何设备（例如路由器、调制解调器等等）通信。这种通信可以通过输入/输出（I/O）接口 1050 进行。并且，电子设备 1000 还可以通过网络适配器 1060 与一个或者多个网络（例如局域网（LAN），广域网（WAN）和/或公共网络，例如因特网）通信。如图所示，网络适配器 1060 通过总线 1030 与电子设备 1000 的其它模块通信。应当明白，尽管图中未示出，可以结合电子设备 1000 使用其它硬件和/或软件模块，包括但不限于：微代码、设备驱动器、冗余处理单元、外部磁盘驱动阵列、RAID 系统、磁带驱动器以及数据备份存储系统等。

通过以上的实施方式的描述，本领域的技术人员易于理解，这里描述的示例实施方式可以通过软件实现，也可以通过软件结合必要的硬件的方式来实现。因此，根据本申请实施方式的技术方案可以以软件产品的形式体现出来，该软件产品可以存储在一个非易失性存储介质（可以是 CD-ROM，U 盘，移动硬盘等）中或网络上，包括若干指令以使得一台计算设备（可以是个人计算机、服务器、终端装置、或者网络设备等）执行根据本申请实施方式的方法。

此外，上述附图仅是根据本申请示例性实施例的方法所包括的处理的示意性说明，而不是限制目的。易于理解，上述附图所示的处理并不表明或限制这些处理的时间顺序。另外，也易于理解，这些处理可以是例如在多个模块中同步或异步执行的。

应当理解的是，本申请并不局限于上面已经描述并在附图中示出的精确结构，并且可以在不脱离其范围执行各种修改和改变。本申请的范

围仅由所附的权利要求来限制。