

(12) 发明专利申请

(10) 申请公布号 CN 103051511 A

(43) 申请公布日 2013. 04. 17

(21) 申请号 201110306094. 1

(22) 申请日 2011. 10. 11

(71) 申请人 阿里巴巴集团控股有限公司

地址 英属开曼群岛大开曼资本大厦一座四层 847 号邮箱

(72) 发明人 马天笑

(74) 专利代理机构 北京润泽恒知识产权代理有限公司 11319

代理人 苏培华

(51) Int. Cl.

H04L 12/58 (2006. 01)

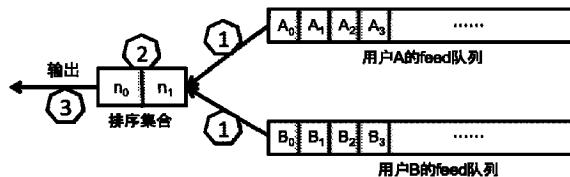
权利要求书 2 页 说明书 8 页 附图 3 页

(54) 发明名称

一种消息数据的处理方法及装置

(57) 摘要

本申请提供了一种消息数据的处理方法及装置，以解决现有的消息数据排序过程中浪费计算资源的问题。本申请借助于每个消息队列（如 Feed 队列）都是有序队列（如已经按照时间排好序）的特点，只对满足用户请求（即用户确实需要）的那部分数据进行排序，而不需要每次都对所有参与排序的消息队列进行排序，从而提高了计算资源的利用，进而提升了用户查询消息数据时的响应时间。



1. 一种消息数据的处理方法,其特征在于,包括:

为对应用户账户创建消息队列,并将用户发布和接收的消息数据放入对应的消息队列中;

接收针对某个用户账户的消息数据查询请求,并执行以下步骤:

步骤1,依据所述消息数据查询请求获取需进行排序的多个有序消息队列,所述需进行排序的多个有序消息队列包含该用户账户的消息队列,以及与该用户账户关联的热门用户账户的消息队列;

步骤2,从每个有序消息队列中分别按序读取一个消息数据,并将所读取的多个消息数据放入排序集合;

步骤3,对所述排序集合中的所有消息数据进行排序,得到排序结果,并从所述排序结果中选择一个消息数据输出;

步骤4,从所述输出的消息数据所属的消息队列中再按序读取一个消息数据,并将所读取的消息数据放入所述排序集合;

重复步骤3和步骤4,直到输出的所有消息数据满足所述请求要求。

2. 根据权利要求1所述的方法,其特征在于,在循环截止之前还包括:

当需进行排序的多个有序消息队列中,只有一个消息队列不为空时,重复将所述不为空的消息队列中的消息数据按序读取并输出,直到输出的所有消息数据个数满足所述请求要求的数量为止。

3. 根据权利要求1或2所述的方法,其特征在于,所述有序消息队列中的消息数据是按照时间倒序排列;所述从消息队列中按序读取一个消息数据包括:从消息队列中按照时间倒序读取第一个消息数据;

所述对排序集合中的所有消息数据进行排序包括:对排序集合中的所有消息数据按照时间倒序进行排序;

所述从排序结果中选择一个消息数据输出包括:从按照时间倒序的排序结果中选择第一个消息数据输出。

4. 根据权利要求1或2所述的方法,其特征在于:

所述排序集合的大小等于需进行排序的有序消息队列的个数。

5. 根据权利要求1或2所述的方法,其特征在于,还包括:

将输出的所有消息数据进行展示。

6. 一种消息数据的处理装置,其特征在于,包括:

队列创建模块,用于为对应用户账户创建消息队列,并将用户发布和接收的消息数据放入对应的消息队列中;

队列获取模块,用于接收针对某个用户账户的消息数据查询请求,并依据所述消息数据查询请求获取需进行排序的多个有序消息队列,所述需进行排序的多个有序消息队列包含该用户账户的消息队列,以及与该用户账户关联的热门用户账户的消息队列;

消息数据读取模块,用于从每个有序消息队列中分别按序读取一个消息数据,并将所读取的多个消息数据放入排序集合;

排序输出模块,用于对所述排序集合中的所有消息数据进行排序,得到排序结果,并从所述排序结果中选择一个消息数据输出;

循环调用模块，用于重复调用所述消息数据读取模块和排序输出模块，从所述输出的消息数据所属的消息队列中再按序读取一个消息数据，并将所读取的消息数据放入所述排序集合进行排序输出，直到输出的所有消息数据满足所述请求要求。

7. 根据权利要求 6 所述的装置，其特征在于，还包括：

特殊处理模块，用于当需进行排序的多个有序消息队列中，只有一个消息队列不为空时，重复将所述不为空的消息队列中的消息数据按序读取并输出，直到输出的所有消息数据个数满足所述请求要求的数量为止。

8. 根据权利要求 7 或 8 所述的装置，其特征在于，所述有序消息队列中的消息数据是按照时间倒序排列；所述消息数据读取模块是从每个有序消息队列中分别按照时间倒序读取第一个消息数据；

所述排序输出模块是对排序集合中的所有消息数据按照时间倒序进行排序，得到排序结果，并从所述排序结果中选择第一个消息数据输出。

9. 根据权利要求 7 或 8 所述的装置，其特征在于：

所述排序集合的大小等于需进行排序的有序消息队列的个数。

10. 根据权利要求 7 或 8 所述的装置，其特征在于，还包括：

展示模块，用于将输出的所有消息数据进行展示。

一种消息数据的处理方法及装置

技术领域

[0001] 本申请涉及网络数据处理技术,特别是涉及一种消息数据的处理方法及装置。

背景技术

[0002] 消息数据是一种数据格式,用于向用户提供频繁更新的内容,例如 Feed 数据就是一种消息数据。下面将以 Feed 数据为例进行说明。

[0003] 对于 Feed 的应用,可以理解为是一种信息传输方式,与所熟悉的电子邮件、万维网 (WWW)、即时消息传送 (IM) 并列。通过 Feed 这种信息传输方式,可以在同样的时间内让用户浏览更大量的信息。

[0004] 在应用 Feed 的典型场景中,用户在应用系统中产生的一些内容,如用户发表的一篇微博,这些用户产生的内容会被应用系统推送给用户的好友或者粉丝,并在用户的好友及其粉丝的个人微博中展示出来,这些被展示出来的内容即是一种 Feed 数据。

[0005] 通常,应用系统为了维护系统中数据量庞大的 Feed 数据,会为每个用户设置对应的 Feed 队列(也可以称为 Feed 列表),Feed 队列既用于存储这个用户通过应用系统向其他用户发布的信息,又用于存储该用户从其他用户接收的信息,即其他用户发布给该用户的信息。

[0006] 当某用户 A 希望浏览 Feed 数据,如登录微博时,向应用系统发送查询 Feed 数据的请求。应用系统为了获取该用户可以看到的所有 Feed 数据,有时需要对多个不同 Feed 队列中的 Feed 数据按照时间进行排序。其中所述多个不同的 Feed 队列既包含用户自己的 Feed 队列,也包含与用户 A 相关的其他多个用户的不同 Feed 队列,如微博中用户 A 所关注的热门用户的 Feed 队列。排序后,再将所有的 Feed 数据按照时间倒序呈现给用户 A,即优先展示最新的 Feed 数据,时间比较久的那些 Feed 数据则放到后面进行展示。

[0007] 现有的技术中,常用的排序办法是将需要排序的所有 Feed 队列全部放到一个集合中,并对这个集合中的 Feed 数据按照时间进行排序。由于用户一般只关注最近一段时间内的 Feed 数据,对于离用户登录时间比较久远的 Feed 数据,用户不太会关注,所以系统会选择返回用户请求的那一页的 Feed 数据。通常只有在用户翻动页面时,系统才会将后续的 Feed 数据返回展示。

[0008] 但是,按照上述排序方法,每次用户请求的时候都要对需要排序的所有 Feed 队列进行一次按时间的排序,这对计算资源造成了很大的浪费。而且,对所有 Feed 数据排序的过程也会影响请求的响应时间,造成用户的等待,影响用户体验。同理,对于类似 Feed 数据的其他消息数据,也存在同样的问题。

发明内容

[0009] 本申请提供了一种消息数据的处理方法及装置,以解决现有的消息数据排序过程中浪费计算资源的问题。

[0010] 为了解决上述问题,本申请公开了一种消息数据的处理方法,包括:

- [0011] 为对应用户账户创建消息队列，并将用户发布和接收的消息数据放入对应的消息队列中；
- [0012] 接收针对某个用户账户的消息数据查询请求，并执行以下步骤：
- [0013] 步骤 1，依据所述消息数据查询请求获取需进行排序的多个有序消息队列，所述需进行排序的多个有序消息队列包含该用户账户的消息队列，以及与该用户账户关联的热门用户账户的消息队列；
- [0014] 步骤 2，从每个有序消息队列中分别按序读取一个消息数据，并将所读取的多个消息数据放入排序集合；
- [0015] 步骤 3，对所述排序集合中的所有消息数据进行排序，得到排序结果，并从所述排序结果中选择一个消息数据输出；
- [0016] 步骤 4，从所述输出的消息数据所属的消息队列中再按序读取一个消息数据，并将所读取的消息数据放入所述排序集合；
- [0017] 重复步骤 3 和步骤 4，直到输出的所有消息数据满足所述请求要求。
- [0018] 优选的，在循环截止之前还包括：当需进行排序的多个有序消息队列中，只有一个消息队列不为空时，重复将所述不为空的消息队列中的消息数据按序读取并输出，直到输出的所有消息数据个数满足所述请求要求的数量为止。
- [0019] 优选的，所述有序消息队列中的消息数据是按照时间倒序排列；所述从消息队列中按序读取一个消息数据包括：从消息队列中按照时间倒序读取第一个消息数据；
- [0020] 所述对排序集合中的所有消息数据进行排序包括：对排序集合中的所有消息数据按照时间倒序进行排序；
- [0021] 所述从排序结果中选择一个消息数据输出包括：从按照时间倒序的排序结果中选择第一个消息数据输出。
- [0022] 优选的，所述排序集合的大小等于需进行排序的有序消息队列的个数。
- [0023] 优选的，所述方法还包括：将输出的所有消息数据进行展示。
- [0024] 本申请还提供了一种消息数据的处理装置，包括：
- [0025] 队列创建模块，用于为对应用户账户创建消息队列，并将用户发布和接收的消息数据放入对应的消息队列中；
- [0026] 队列获取模块，用于接收针对某个用户账户的消息数据查询请求，并依据所述消息数据查询请求获取需进行排序的多个有序消息队列，所述需进行排序的多个有序消息队列包含该用户账户的消息队列，以及与该用户账户关联的热门用户账户的消息队列；
- [0027] 消息数据读取模块，用于从每个有序消息队列中分别按序读取一个消息数据，并将所读取的多个消息数据放入排序集合；
- [0028] 排序输出模块，用于对所述排序集合中的所有消息数据进行排序，得到排序结果，并从所述排序结果中选择一个消息数据输出；
- [0029] 循环调用模块，用于重复调用所述消息数据读取模块和排序输出模块，从所述输出的消息数据所属的消息队列中再按序读取一个消息数据，并将所读取的消息数据放入所述排序集合进行排序输出，直到输出的所有消息数据满足所述请求要求。
- [0030] 优选的，所述装置还包括：特殊处理模块，用于当需进行排序的多个有序消息队列中，只有一个消息队列不为空时，重复将所述不为空的消息队列中的消息数据按序读取并

输出,直到输出的所有消息数据个数满足所述请求要求的数量为止。

[0031] 优选的,所述有序消息队列中的消息数据是按照时间倒序排列;所述消息数据读取模块是从每个有序消息队列中分别按照时间倒序读取第一个消息数据;

[0032] 所述排序输出模块是对排序集合中的所有消息数据按照时间倒序进行排序,得到排序结果,并从所述排序结果中选择第一个消息数据输出。

[0033] 优选的,所述排序集合的大小等于需进行排序的有序消息队列的个数。

[0034] 优选的,所述装置还包括:展示模块,用于将输出的所有消息数据进行展示。

[0035] 与现有技术相比,本申请包括以下优点:

[0036] 本申请借助于每个消息队列(如Feed队列)都是有序队列(如已经按照时间排好序)的特点,只对满足用户请求(即用户确实需要)的那部分数据进行排序,而不需要每次都对所有参与排序的消息队列进行排序,从而提高了计算资源的利用,进而提升了用户查询消息数据时的响应时间。

附图说明

[0037] 图1是本申请实施例所述将多个Feed队列合并排序的示意图;

[0038] 图2是本申请实施例所述一种消息数据的处理方法流程图;

[0039] 图3是本申请例1中对两个Feed队列进行合并排序的示意图;

[0040] 图4是本申请例2中对三个Feed队列进行合并排序的示意图;

[0041] 图5是本申请实施例所述一种消息数据的处理装置结构图;

[0042] 图6是本申请另一实施例所述一种消息数据的处理装置结构图。

具体实施方式

[0043] 为使本申请的上述目的、特征和优点能够更加明显易懂,下面结合附图和具体实施方式对本申请作进一步详细的说明。

[0044] 在消息数据的应用系统(以下简称为系统)中,为了获取一个用户可以看到的消息数据,有时需要对多个不同的消息队列进行排序。以Feed数据为例,具体的一种应用场景如下:

[0045] 在微博网站、交友互动平台等Feed应用系统中,每个用户都有自己的Feed队列。对于普通用户,由于其好友的数量有限,所以普通用户发布和接收的Feed数据量也不会太大,因此,系统通常为普通用户设置一个Feed队列,用来存储该用户发布和接收的Feed数据,即:每当该用户发布一条Feed数据,或从其好友处接收一条Feed数据,系统都会将这些Feed数据拷贝到该用户的Feed队列中。这样,如果获取该用户可以看到的所有Feed数据,直接读取该用户的Feed队列即可。

[0046] 但是,对于热门用户而言,其粉丝数是非常多的,可能有几十万甚至上百万,如果将热门用户发布的一条Feed数据拷贝到每个粉丝的Feed队列中,将耗费非常多的工作量和时间,给系统带来很大的压力。为解决此问题,系统通常为热门用户设置两个Feed队列,一个用于存储热门用户自己发布的Feed数据,另一个用于存储热门用户从其他用户(如好友)接收的Feed数据。这样,当获取一个用户可以看到的所有Feed数据时,就需要对该用户的Feed队列和该用户所关注的热门用户的Feed队列(存储接收的Feed数据的队列)

进行排序。

[0047] 本申请提出一种消息数据的处理方法及装置,可以对多个不同的消息队列进行排序,并将排序后的消息数据呈现给用户。例如,参照图1所示,可以将用户A的Feed队列、用户B的Feed队列、用户C的Feed队列和用户D的Feed队列进行合并,并对合并后的Feed数据进行排序,然后将要呈现给用户的Feed数据输出。

[0048] 下面通过实施例对本申请所述方法的实现流程进行详细说明。

[0049] 参照图2,其为本申请实施例所述一种消息数据的处理方法流程图。仍以Feed数据为例,流程如下:

[0050] 步骤S1,为对应用户账户创建消息队列,并将用户发布和接收的消息数据放入对应的消息队列中;

[0051] 如前所述,在消息数据的应用系统中,每个用户都有自己的消息队列,也即:系统会为每个注册的用户账户创建一个或多个空消息队列。如果是普通用户,则创建一个空消息队列,该用户每发布一条消息数据以及每接收一条消息数据,都会放入这个空消息队列。但是对于拥有大量好友或粉丝的热门用户,系统会对于该热门用户的账户创建两个空消息队列,其中一个空消息队列用于存放该热门用户发布的每一条消息数据,而另一个空消息队列用于存放该热门用户接收到的每一条消息数据。

[0052] 步骤S2,接收针对某个用户账户的消息数据查询请求,并执行以下步骤:

[0053] 步骤1,依据所述消息数据查询请求获取需进行排序的多个有序消息队列,所述需进行排序的多个有序消息队列包含该用户账户的消息队列,以及与该用户账户关联的热门用户账户的消息队列;

[0054] 在微博网站、交友互动平台等Feed应用系统中,一个用户的所有好友都可称为该用户的关联用户。在这些关联用户中,有些是普通用户,有些是热门用户。如果请求查询某一个用户的Feed数据,而该用户的好友中包含了热门用户,则用于进行排序的Feed队列不仅包含该用户自己的Feed队列,还包含该用户的好友中热门用户的Feed队列(主要是存放发布的Feed数据的那个队列)。

[0055] 例如,某用户A为普通用户,其关注的热门用户有用户B和用户C。当用户A向系统请求展示可以看到的Feed数据时,系统会根据该请求获取用户A的Feed队列A、用户A关注的热门用户B的Feed队列B和热门用户C的Feed队列C。

[0056] 其中,所述的每个Feed队列都可以是一个有序的队列。根据实际应用的不同,所述“有序”可以是按照时间顺序,或按照数据大小的顺序,或按照数据标识的顺序,等等排序方式进行排序。总之,所述Feed队列是一个有序的队列,即Feed队列中的所有Feed元素已按照某种方式排好了序。

[0057] 对于微博网站、交友互动平台等Feed应用系统,由于用户一般只关注最近一段时间内的Feed,所以系统中每个Feed队列中的元素都可按照时间倒序进行排列,即将发布时间最新的Feed元素排在队列的最前端。

[0058] 步骤2,从每个有序消息队列中分别按序读取一个消息数据,并将所读取的多个消息数据放入排序集合;

[0059] 仍以Feed应用系统为例,所述排序集合用于对多个Feed队列的Feed数据进行合并和排序,排序集合的大小等于需进行排序的有序Feed队列的个数。例如,需进行排序的

有序 Feed 队列共有 3 个，则对应的排序集合能存放 3 个 Feed 数据。

[0060] 由于每个 Feed 队列都是有序的，所以从队列中顺次读取的 Feed 数据也是有序的。如果 Feed 队列是按照时间倒序排序，那么读取时就可以从 Feed 队列中按照时间倒序读取第一个 Feed 数据，所述第一个 Feed 数据是指排在队列最前端的 Feed 数据，也是发布时间最新的 Feed 数据。

[0061] 当然，在其他应用中，针对所述按照时间倒序排列的 Feed 队列，也可以从队尾开始读取 Feed 数据，这时读取出来的 Feed 数据就是发布时间最早的 Feed 数据。也就是说，上述“按序读取”的顺序不一定必须与“有序 Feed 队列”中 Feed 数据的排列顺序一致。

[0062] 步骤 3，对所述排序集合中的所有消息数据进行排序，得到排序结果，并从所述排序结果中选择一个消息数据输出；

[0063] 仍以 Feed 应用系统为例，由于排序集合中的 Feed 数据来自不同的 Feed 队列，并无顺序，需要进行排序后才能选出符合应用要求的 Feed 数据输出。在对排序集合中的所有 Feed 数据进行排序时，需按照应用要求选择排序方式，排序方式不一定必须与上述步骤 1 中“有序 Feed 队列”中 Feed 数据的排列顺序一致，但要与上述步骤 2 中“按序读取”的顺序保持一致。

[0064] 例如，如果应用要求输出最近一段时间内的 Feed，则上述步骤 2 中“按序读取”是按照时间倒序读取队列中的 Feed，并且在步骤 3 中，排序集合中的所有 Feed 数据也是按照时间倒序排序，并选择排序结果中的第一个 Feed 数据输出，即输出排序结果中发布时间最新的 Feed 数据。

[0065] 步骤 4，从所述输出的消息数据所属的 Feed 队列中再按序读取一个消息数据，并将所读取的消息数据放入所述排序集合；

[0066] 步骤 S3，重复步骤 3 和步骤 4，直到输出的所有消息数据满足所述请求要求。

[0067] 其中，所述请求要求可以是数量上的，如当前屏幕显示数量等；也可以是时间上的，如当天，本周的消息数据等，或者数量、时间等各种条件的组合。

[0068] 例如，步骤 3 输出的 Feed 数据属于 Feed 队列 A，则继续从 Feed 队列 A 按序读取一个 Feed 数据再放入所述排序集合，再次与排序集合中原来存放的 Feed 队列 B 和 Feed 队列 C 的 Feed 数据一起进行排序。按照此方式重复循环，直到输出的所有 Feed 数据个数满足所述请求要求的数量为止，如系统设置每一页展示的 Feed 数据个数为 10 个，当用户请求获取第一页的输出结果时，排序集合输出 10 个 Feed 数据即停止所述循环；当用户点击“下一页”继续请求获取第二页的输出结果时，系统再从步骤 1 开始执行。

[0069] 需要注意的是，上述循环执行过程中，步骤 3 中每次的排序方式需一致，步骤 4 中每次按序读取 Feed 数据的顺序需一致。

[0070] 可选的，如果所述请求为 Feed 数据查询请求，系统还可以将满足请求个数的所有 Feed 数据展示在用户客户端上，供用户浏览。当然，根据实际应用的不同，还可以对输出的 Feed 数据进行其他后续处理，如进行统计分析或传送给其他应用系统等。

[0071] 下面通过两个例子对上述流程进行解释说明。

[0072] 例 1：

[0073] 参照图 3，是本申请例 1 中对两个 Feed 队列进行合并排序的示意图。

[0074] 假设用户 A 为普通用户，用户 A 关注热门用户 B，当用户 A 向 Feed 应用系统发送查

询请求,请求获取用户 A 可以看到的第一页的 Feed 数据时,系统首先获取用户 A 的 Feed 队列,以及用户 A 关注的热门用户 B 的 Feed 队列,并对这两个 Feed 队列进行以下处理:

[0075] S1,分别从用户 A 的 Feed 队列和用户 B 的 Feed 队列中各取出第一个 Feed 数据,然后将这两个 Feed 数据放入到排序集合中;

[0076] 例如,分别读取 A0 和 B0 放入排序集合;

[0077] S2,对放入到排序集合中的两个 Feed 数据进行排序,得到按照时间倒序排列的一个结果;

[0078] 假设按照时间倒序排列的结果是 :B0、A0;

[0079] S3,对外输出结果,当从排序集合中取出了一个排在最前面的 Feed 数据之后,会从这个 Feed 数据所属的那个 Feed 队列中再取出下一个 Feed 数据放到排序集合中,并重新对这个排序集合中的所有 Feed 数据进行排序;

[0080] 例如,将 B0 输出,然后再从用户 B 的 Feed 队列中读取 B1 并放入排序集合,此时排序集合中的 Feed 数据为 A0 和 B1,对 A0 和 B1 排序;

[0081] S4,排序集合不断地输出结果,不断重复 S3 中的动作,直到满足查询要求的 Feed 数量为止结束。

[0082] 若输出 A0,则重复 S3,从用户 A 的 Feed 队列中读取 A1 并放入排序集合,此时排序集合中的 Feed 数据为 A1 和 B1,对 A1 和 B1 排序, ……,依此循环,直到满足查询要求的 Feed 数量为止结束。

[0083] 以上是在两个用户 Feed 队列的场景下说明了本申请所述方法的执行步骤,对于超过两个用户 Feed 队列的情况,只需要扩大排序集合,使其大小等于需要进行归并排序的 Feed 队列数量就可以,具体参见例 2。

[0084] 例 2 :

[0085] 参照图 4,是本申请例 2 中对三个 Feed 队列进行合并排序的示意图。

[0086] S1,分别从用户 A 的 Feed 队列、用户 B 的 Feed 队列和用户 C 的 Feed 队列中各取出第一个 Feed 数据 A0、B0 和 C0,然后将这三个 Feed 数据放入到排序集合中;

[0087] S2,对放入到排序集合中的三个 Feed 数据进行排序,得到按照时间倒序排列的一个结果;

[0088] 假设按照时间倒序排列的结果是 :C0、A0、B0;

[0089] S3,对外输出 C0,并从 C0 所属的用户 C 的 Feed 队列中再取出下一个 Feed 数据 C1 放到排序集合中,并重新对这个排序集合中的所有 Feed 数据进行排序;

[0090] 此时,排序集合中的 Feed 数据为 A0、B0 和 C1;

[0091] S4,排序集合不断地输出结果,不断重复 S3 中的动作,直到满足查询要求的 Feed 数量为止结束。

[0092] 若输出 A0,则重复 S3,从用户 A 的 Feed 队列中读取 A1 并放入排序集合,此时排序集合中的 Feed 数据为 A1、B0 和 C1, ……,依此循环,直到满足查询要求的 Feed 数量为止结束。

[0093] 此外,基于例 1 和例 2 还需要说明的是,在上述循环截止之前,还可能存在以下特殊情况:

[0094] 当需进行排序的多个有序 Feed 队列中,只有一个 Feed 队列不为空,其余 Feed 队

列均为空（即 Feed 数据个数为零）时，重复将所述不为空的 Feed 队列中的 Feed 数据按序读取并输出，直到输出的所有 Feed 数据个数满足所述请求要求的数量为止。

[0095] 例如，图 3 所示的两个 Feed 队列的排序过程中，会出现某个用户的 Feed 队列中没有更多 Feed 数据的情况。假设用户 A 的 Feed 队列比用户 B 的 Feed 队列短，且某一次查询中，执行过程中发现用户 A 的 Feed 队列已经被耗尽，这个时候，系统会删除排序集合中的一个 Feed 数据，也就是说这个时候排序集合中就只剩下下一个 Feed 数据了，后面的输出就没有排序的过程了，直接将用户 B 的 Feed 队列中的 Feed 数据对外输出。

[0096] 同样，对于图 4 所示的三个 Feed 队列的排序过程，也会出现某个用户的 Feed 队列中没有更多 Feed 数据的情况。假设某一次查询中，发现用户 A 和用户 B 的 Feed 队列已经被耗尽，此时系统会删除排序集合中的两个 Feed 数据，后面的输出就没有排序的过程了，直接将用户 C 的 Feed 队列中的 Feed 数据对外输出。但是，如果执行过程中只有用户 A 的 Feed 队列已经被耗尽，系统会删除排序集合中的一个 Feed 数据，使得排序集合中只剩下两个 Feed 数据，此时排序集合还会按照前述的方法，对用户 B 的 Feed 队列和用户 C 的 Feed 队列中的 Feed 数据进行排序输出。

[0097] 综上所述，本申请实施例所述的方法借助于每个消息队列都是有序队列（如已经按照时间排好序）的特点，只对满足用户请求（即用户确实需要）的那部分数据进行排序，而不需要每次都对所有参与排序的 Feed 队列进行排序，从而提高了计算资源的利用，进而提升了用户查询消息时的响应时间。

[0098] 需要说明的是，对于前述的各方法实施例，为了简单描述，故将其都表述为一系列的动作组合，但是本领域技术人员应该知悉，本申请并不受所描述的动作顺序的限制，因为依据本申请，某些步骤可以采用其他顺序或者同时进行。其次，本领域技术人员也应该知悉，说明书中所描述的实施例均属于优选实施例，所涉及的动作并不一定是本申请所必须的。

[0099] 基于上述方法实施例的说明，本申请还提供了相应的装置实施例，来实现上述方法实施例所述的内容。

[0100] 参照图 5，是本申请实施例所述一种消息数据的处理装置结构图。

[0101] 所述 Feed 数据处理装置可以包括队列创建模块 10、队列获取模块 20、消息数据读取模块 30、排序输出模块 40 和循环调用模块 50，其中，

[0102] 队列创建模块 10，用于为对应用户账户创建消息队列，并将用户发布和接收的消息数据放入对应的消息队列中；

[0103] 队列获取模块 20，用于接收针对某个用户账户的消息数据查询请求，并依据所述消息数据查询请求获取需进行排序的多个有序消息队列，所述需进行排序的多个有序消息队列包含该用户账户的消息队列，以及与该用户账户关联的热门用户账户的消息队列；

[0104] 消息数据读取模块 30，用于从每个有序消息队列中分别按序读取一个消息数据，并将所读取的多个消息数据放入排序集合；

[0105] 排序输出模块 40，用于对所述排序集合中的所有消息数据进行排序，得到排序结果，并从所述排序结果中选择一个消息数据输出；

[0106] 循环调用模块 50，用于重复调用所述消息数据读取模块 30 和排序输出模块 40，从所述输出的消息数据所属的消息队列中再按序读取一个消息数据，并将所读取的消息数据

放入所述排序集合进行排序输出，直到输出的所有消息数据满足所述请求要求。

[0107] 其中，所述排序集合的大小等于需进行排序的有序消息队列的个数。

[0108] 可选的，所述有序消息队列中的消息数据是按照时间倒序排列；

[0109] 而且，所述消息数据读取模块 30 是从每个有序消息队列中分别按照时间倒序读取第一个消息数据；

[0110] 所述排序输出模块 40 是对排序集合中的所有消息数据按照时间倒序进行排序，得到排序结果，并从所述排序结果中选择第一个消息数据输出。

[0111] 在本申请的另一装置实施例中，参照图 6 所示，所述消息数据处理装置还可以包括：

[0112] 特殊处理模块 60，用于当需进行排序的多个有序消息队列中，只有一个消息队列不为空时，重复将所述不为空的消息队列中的消息数据按序读取并输出，直到输出的所有消息数据个数满足所述请求要求的数量为止。

[0113] 在本申请的另一装置实施例中，参照图 6 所示，所述消息数据处理装置还可以包括：

[0114] 展示模块 60，用于将输出的所有消息数据进行展示。

[0115] 对于上述消息数据处理装置实施例而言，由于其与方法实施例基本相似，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。

[0116] 本说明书中的各个实施例均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的部分互相参见即可。

[0117] 以上对本申请所提供的一种消息数据的处理方法及装置，进行了详细介绍，本文中应用了具体个例对本申请的原理及实施方式进行了阐述，以上实施例的说明只是用于帮助理解本申请的方法及其核心思想；同时，对于本领域的一般技术人员，依据本申请的思想，在具体实施方式及应用范围上均会有改变之处，综上所述，本说明书内容不应理解为对本申请的限制。

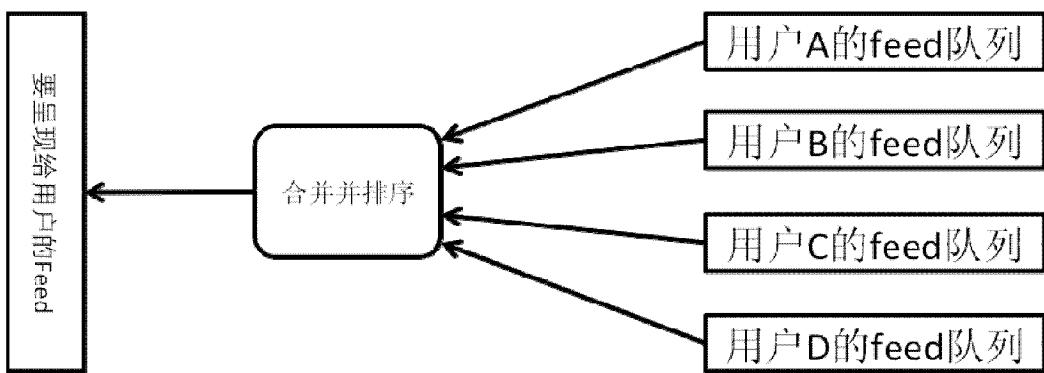


图 1

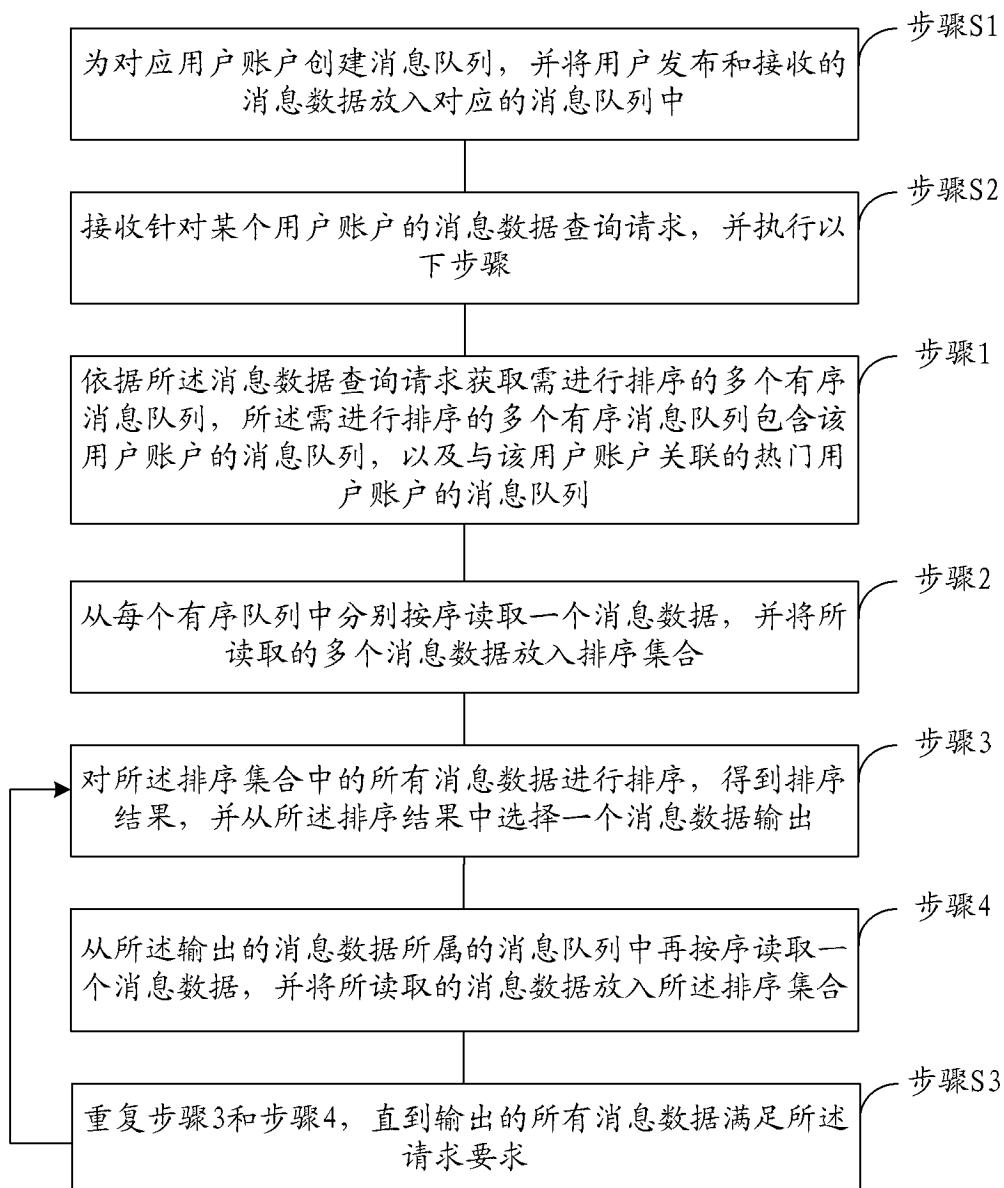


图 2

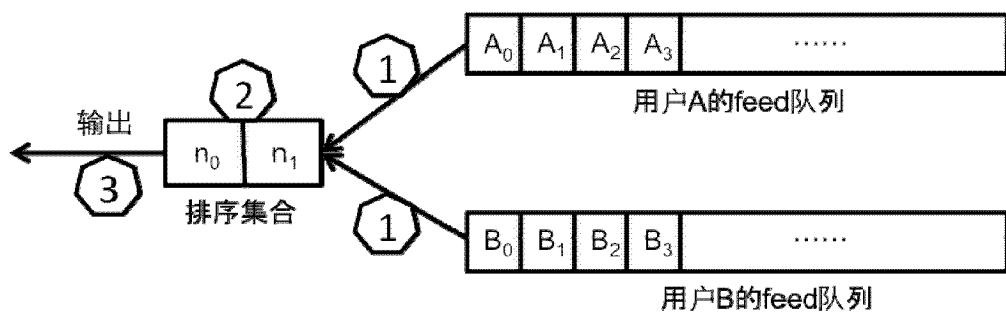


图 3

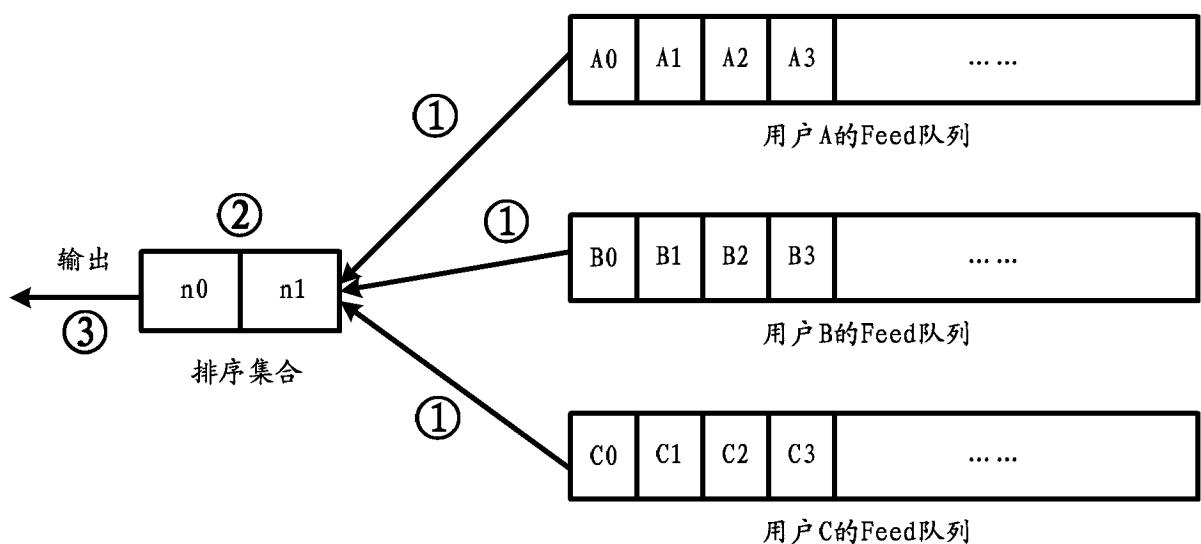


图 4

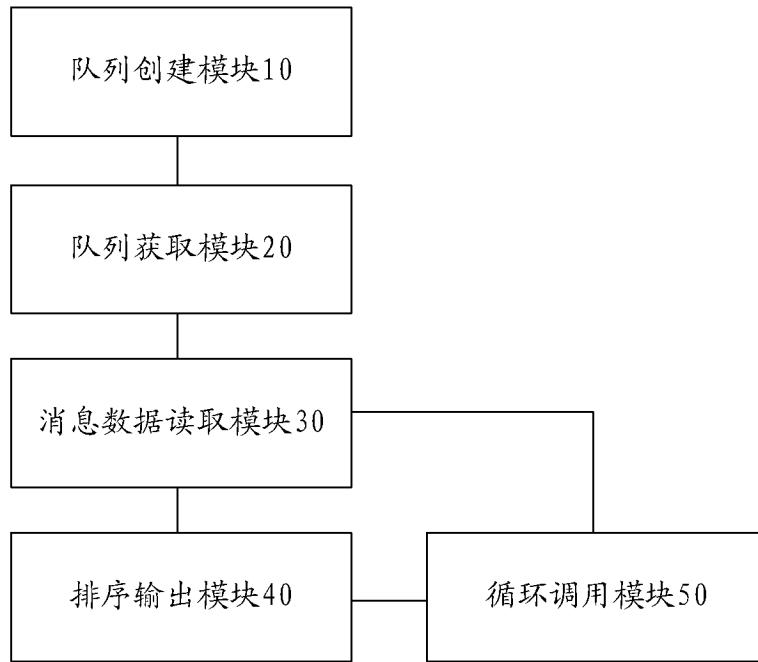


图 5

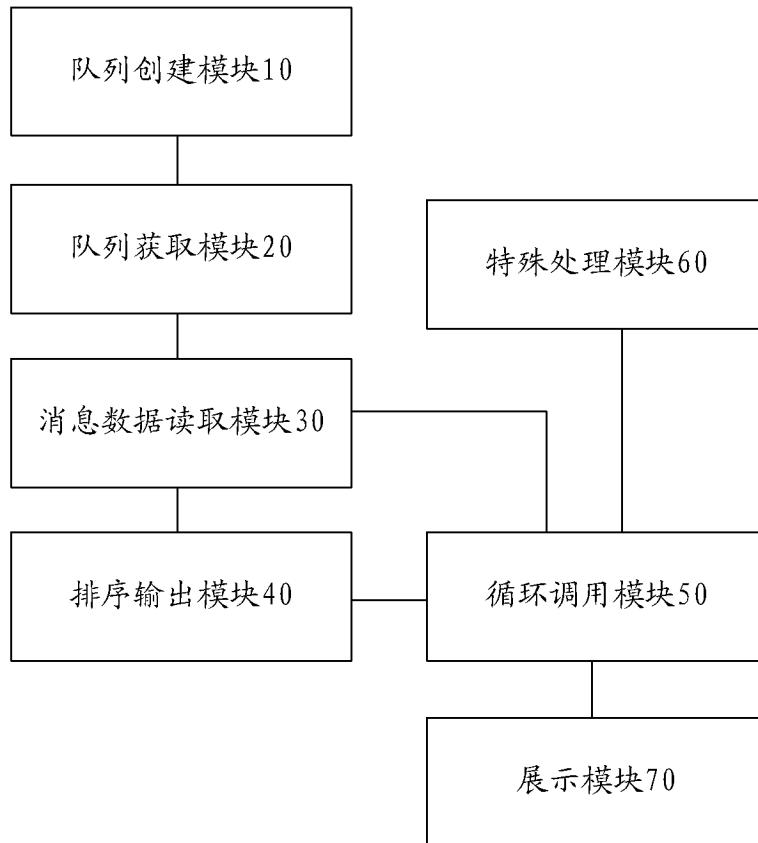


图 6