(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 113138909 A
(43)申请公布日 2021.07.20

(21)申请号 202010065063.0

(22)申请日 2020.01.20

(71)申请人 OPPO广东移动通信有限公司
地址 523860 广东省东莞市长安镇乌沙海滨路18号

(72)发明人 崔晓刚

(74)专利代理机构 北京恒博知识产权代理有限公司 11528
代理人 范胜祥

(51)Int.Cl.
G06F 11/34(2006.01)

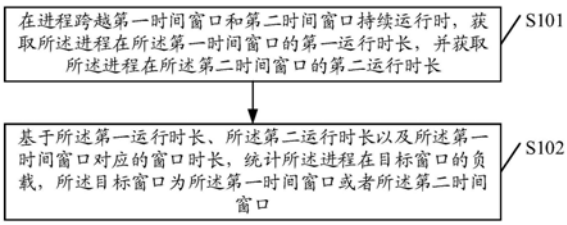
权利要求书2页 说明书10页 附图5页

(54)发明名称

负载统计方法、装置、存储介质及电子设备

(57)摘要

本申请实施例公开了一种负载统计方法、装置、存储介质及电子设备，方法包括：在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进程在所述第一时间窗口的第一运行时长，并获取所述进程在所述第二时间窗口的第二运行时长；基于所述第一运行时长、所述第二运行时长以及所述第一时间窗口对应的窗口时长，统计所述进程在目标窗口的负载，所述目标窗口为所述第一时间窗口或者所述第二时间窗口。因此，采用本申请实施例，可以在进程跨时间窗口运行时，统计的负载更接近真实负载。



CN 113138909 A

1.一种负载统计方法,其特征在于,所述方法包括:

在进程跨越第一时间窗口和第二时间窗口持续运行时,获取所述进程在所述第一时间窗口的第一运行时长,并获取所述进程在所述第二时间窗口的第二运行时长;

基于所述第一运行时长、所述第二运行时长以及所述第一时间窗口对应的窗口时长,统计所述进程在目标窗口的负载,所述目标窗口为所述第一时间窗口或者所述第二时间窗口。

2.根据权利要求1所述的方法,其特征在于,所述基于所述第一运行时长、所述第二运行时长以及所述第一时间窗口对应的窗口时长,统计所述进程在目标窗口的负载,包括:

获取所述第一运行时长和所述第二运行时长中的较大时长;

计算所述第一运行时长与所述第二运行时长的和,得到总运行时长;

计算所述总运行时长与所述第一时间窗口对应的窗口时长的商,将所述商作为所述进程在所述较大时长对应的目标窗口的负载。

3.根据权利要求2所述的方法,其特征在于,所述将所述商作为所述进程在所述较大时长对应的目标窗口的负载,包括:

当所述第一运行时长大于所述第二运行时长时,将所述商作为所述进程在所述第一时间窗口的负载;

当所述第一运行时长小于所述第二运行时长时,将所述商作为所述进程在所述第二时间窗口的负载;

当所述第一运行时长等于所述第二运行时长时,将所述商作为所述进程在所述第一时间窗口或者所述第二时间窗口的负载。

4.根据权利要求2所述的方法,其特征在于,所述计算所述第一运行时长与所述第二运行时长的和,得到总运行时长,包括:

获取所述较大时长对应的目标窗口内所述进程非持续运行的第三运行时长;

计算所述第一运行时长、所述第二运行时长以及所述第三运行时长的和,得到总运行时长。

5.根据权利要求2或4所述的方法,其特征在于,所述得到总运行时长之后,还包括:

当所述总运行时长大于所述窗口时长时,将所述总运行时长设置为所述窗口时长。

6.一种负载统计装置,其特征在于,所述装置包括:

运行时长获取模块,用于在进程跨越第一时间窗口和第二时间窗口持续运行时,获取所述进程在所述第一时间窗口的第一运行时长,并获取所述进程在所述第二时间窗口的第二运行时长;

负载统计模块,用于基于所述第一运行时长、所述第二运行时长以及所述第一时间窗口对应的窗口时长,统计所述进程在目标窗口的负载,所述目标窗口为所述第一时间窗口或者所述第二时间窗口。

7.根据权利要求6所述的装置,其特征在于,所述负载统计模块,包括:

较大时长获取单元,用于获取所述第一运行时长和所述第二运行时长中的较大时长;

总时长计算单元,用于计算所述第一运行时长与所述第二运行时长的和,得到总运行时长;

负载计算单元,用于计算所述总运行时长与所述第一时间窗口对应的窗口时长的商,

将所述商作为所述进程在所述较大时长对应的目标窗口的负载。

8.根据权利要求7所述的装置,其特征在于,所述负载计算单元,具体用于:

当所述第一运行时长大于所述第二运行时长时,将所述商作为所述进程在所述第一时间窗口的负载;

当所述第一运行时长小于所述第二运行时长时,将所述商作为所述进程在所述第二时间窗口的负载;

当所述第一运行时长等于所述第二运行时长时,将所述商作为所述进程在所述第一时间窗口或者所述第二时间窗口的负载。

9.一种计算机存储介质,其特征在于,所述计算机存储介质存储有多条指令,所述指令适于由处理器加载并执行如权利要求1～5任意一项的方法步骤。

10.一种电子设备,其特征在于,包括:处理器和存储器;其中,所述存储器存储有计算机程序,所述计算机程序适于由所述处理器加载并执行如权利要求1～5任意一项的方法步骤。

负载统计方法、装置、存储介质及电子设备

技术领域

[0001] 本申请涉及计算机技术领域，尤其涉及一种负载统计方法、装置、存储介 质及电子设备。

背景技术

[0002] 当前的安卓Linux内核的负载跟踪机制主要采用窗口辅助负载跟踪 (Window Asisted Load Tracking,WALT)负载追踪计算方法，通过将时间分割 为多个时间窗口window的形式，统计每个进程在某一个时间窗口内的运行时间， 如图1所示，在一个window size的周期内，进程在第一时间窗口内持续运行时 间从s1到e1，那么所对应的负载Load＝(e1-s1)/window_size*100％。

[0003] 由于时间窗口存在界限，当一个进程运行时间正好跨时间窗口的边界的话， 那么这个进程的运行时间就被分为两部分并分别计入不同的时间窗口之内，将 会导致统计的负载偏小。

发明内容

[0004] 本申请实施例提供了一种负载统计方法、装置、存储介质及电子设备，可 以在进程跨时间窗口运行时，统计的负载更接近真实负载。所述技术方案如下：

[0005] 第一方面，本申请实施例提供了一种负载统计方法，所述方法包括：

[0006] 在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进程在所 述第一时间窗口的第一运行时长，并获取所述进程在所述第二时间窗口的第二 运行时长；

[0007] 基于所述第一运行时长、所述第二运行时长以及所述第一时间窗口对应的 窗口时长，统计所述进程在目标窗口的负载，所述目标窗口为所述第一时间窗 口或者所述第二时间窗口。

[0008] 第二方面，本申请实施例提供了一种负载统计装置，所述装置包括：

[0009] 运行时长获取模块，用于在进程跨越第一时间窗口和第二时间窗口持续运 行时，获取所述进程在所述第一时间窗口的第一运行时长，并获取所述进程在 所述第二时间窗口的第二运行时长；

[0010] 负载统计模块，用于基于所述第一运行时长、所述第二运行时长以及所述 第一时间窗口对应的窗口时长，统计所述进程在目标窗口的负载，所述目标窗 口为所述第一时间窗口或者所述第二时间窗口。

[0011] 第三方面，本申请实施例提供一种计算机存储介质，所述计算机存储介质 存储有多条指令，所述指令适于由处理器加载并执行上述的方法步骤。

[0012] 第四方面，本申请实施例提供一种电子设备，可包括：处理器和存储器； 其中，所述存储器存储有计算机程序，所述计算机程序适于由所述处理器加载 并执行上述的方法步骤。

[0013] 本申请一些实施例提供的技术方案带来的有益效果至少包括：

[0014] 在本申请实施例中，在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进程在所述第一时间窗口的第一运行时长，并获取所述进程在所述第 二时间窗口的第二运行时长，基于所述第一运行时长、所述第二运行时长以及 所述第一时间窗口对应的窗口时长，统计所述进程在目标窗口的负载，所述目 标窗口为所述第一时间窗口或者所述第二时间窗口。即使进程跨窗口运行，通 过结合跨越的多个时间窗口的运行时间计算在某一个时间窗口的负载，而并非 只依据某一个时间窗口的运行时间统计负载，可以提高负载计算的准确性，使 得统计的负载更接近真实负载。

附图说明

[0015] 为了更清楚地说明本申请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 中的附图仅仅是本申请的一些实施例，对于本领域普通技术人员来讲，在不付 出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0016] 图1是本申请实施例提供的一种进程在时间窗口内的运行时长的举例示意 图；

[0017] 图2是本申请实施例提供的一种负载统计方法的流程示意图；

[0018] 图3是本申请实施例提供的一种负载统计方法的流程示意图；

[0019] 图4是本申请实施例提供的一种进程在时间窗口内的运行时长的举例示意 图；

[0020] 图5是本申请实施例提供的一种进程在时间窗口内的运行时长的举例示意 图；

[0021] 图6是本申请实施例提供的一种负载统计方法的流程示意图；

[0022] 图7是本申请实施例提供的一种进程在时间窗口内的运行时长的举例示意 图；

[0023] 图8是本申请实施例提供的一种负载统计装置的结构示意图；

[0024] 图9是本申请实施例提供的一种负载统计装置的结构示意图；

[0025] 图10是本申请实施例提供的一种电子设备的结构示意图。

具体实施方式

[0026] 为使本申请的目的、技术方案和优点更加清楚，下面将结合附图对本申请 实施例方式作进一步地详细描述。

[0027] 下面的描述涉及附图时，除非另有表示，不同附图中的相同数字表示相同 或相似的要素。以下示例性实施例中所描述的实施方式并不代表与本申请相一 致的所有实施方式。相反，它们仅是如所附权利要求书中所详述的、本申请的 一些方面相一致的装置和方法的例子。

[0028] 在本申请的描述中，需要理解的是，术语“第一”、“第二”等仅用于描述 目的，而不能理解为指示或暗示相对重要性。对于本领域的普通技术人员而言， 可以具体情况理解上述术语在本申请中的具体含义。此外，在本申请的描述中， 除非另有说明，“多个”是指两个或两个以上。“和/或”，描述关联对象的关联关 系，表示可以存在三种关系，例如，A和/或B，可以表示：单独存在A，同时 存在A和B，单独存在B这三种情况。字符“/”一般表示前后关联对象是一种 “或”的关系。

[0029] 下面将结合附图2-附图7，对本申请实施例提供的负载统计方法进行详细介 绍。该方法可依赖于计算机程序实现，可运行于基于冯诺依曼体系的负载统计 装置上。该计算

机程序可集成在应用中，也可作为独立的工具类应用运行。其 中，本申请实施例中的负载统计装置可以为用户终端，所述用户终端包括但不 限于：智能手机、个人电脑、平板电脑、手持设备、可穿戴设备、计算设备或 连接到无线调制解调器的其它处理设备等。

[0030] 请参见图2，为本申请实施例提供的一种负载统计方法的流程示意图。如图 2所示，本申请实施例的所述方法可以包括以下步骤：

[0031] S101，在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进 程在所述第一时间窗口的第一运行时长，并获取所述进程在所述第二时间窗口 的第二运行时长；

[0032] 进程常常被定义为程序的执行。可以把一个进程看成是一个独立的程序，在内存中有其完备的数据空间和代码空间。

[0033] 进程是表示资源分配的基本单位，又是调度运行的基本单位。例如，用户 运行自己的程序，系统就创建一个进程，并为它分配资源，包括各种表格、内 存空间、磁盘空间、I/O设备等。然后，将该进程添加到进程的就绪队列中。进程调度程序选中它，为它分配CPU以及其它有关资源，该进程才真正运行。

[0034] 在本申请实施例中，采用WALT(window asisted load tracking)的负载追踪 统计方法统计各运行进程的负载，其主体思想是将时间分为以window_size(如 20ms)大小的多个时间窗口。在每个时间窗口内统计每一个任务的运行时间 runtime。那么该任务的负载为：Load＝runtime/window_size*100％。

[0035] 需要说明的是，对于进程负载的统计是以时间窗口为单位，也就是统计进 程在某个时间窗口内的负载大小。

[0036] 但对于本申请实施例中提及的进程为跨窗口进程，也就是该进程跨越第一 时间窗口和第二时间窗口持续运行，在第一时间窗口内有一段运行时长，在第 二时间窗口内也有一段运行时长，并且第一时间窗口和第二时间窗口为相邻时 间窗口。

[0037] 具体的，记录该进程在第一时间窗口运行的时间起点，并记录该进程在第 二时间窗口运行的时间终点，将时间起点到两窗口之间的分割点之间的时间作 为第一运行时长，该分割点到时间终点之间的时间作为第二运行时长。当然，所述进程可以为一个或多个，对于多个运行进程均可采用上述相同的处理方式。

[0038] 可选的，在系统中包含多个进程，每个进程的功能不同，因此进程对CPU 的资源的要求也不同，所以对进程的调度就有了优先级，进程的优先级决定以 一个进程的CPU资源的优先级分配权。CPU资源分配的先后顺序，就是指进程 的优先权。

[0039] 预先对系统所包含的全部进程进行分类，从而生成不同的进程组。具体分 类可基于每个进程的属性，如进程名称、应用程序类型、应用程序标识、环境 数据、用户数据、优先级、对CPU频率的需求程度或对响应速度的需求程度，等。所述应用程序类型是指选择在启动队列接收到触发器消息时启动的应用程 序的类型。不同的进程组对应不同的负载跟踪策略，以便保持系统性能与功耗 的平衡。

[0040] 例如，进程组可包括前台进程组、后台进程组、顶级进程组。前台进程组 包括系统的前台进程，后台进程组包括系统的后台进程，顶级进程组包括系统 的顶级进程。前台进程组、后台进程组以及顶级进程组构成系统的全部进程。

[0041] 其中，顶级进程更偏向性能，那么所对应的负载跟踪策略可以为历史多个 负载跟踪记录的最大值；后台进程对性能需求不高，那么所对应的负载跟踪策 略就是历史多个负

载跟踪记录的平均值;前台进程其性能需求介于顶级进程和 后台进程之间,那么所对应的负载跟踪策略就是在历史多个负载跟踪记录的平 均值与最近一个记录之中较大的一个。这样通过应用不同的负载跟踪策略可以 实现进一步的细分控制,使得对性能需求高的进程能够更好的满足其性能,进 而提升应用的用户体验,对于性能需求不高的应用通过应用计算的负载偏低的 负载跟踪策略,以节省系统的功耗,延长系统的续航。

[0042] 如表1所示为所缓存的进程组集合中各进程组与负载跟踪策略的对应关系 表。例如,若目标负载跟踪策略为历史五个负载跟踪记录的最大值,那么获取 该进程在历史时刻随机对应的五个负载值,将其中的最大值确定为该进程的当 前负载。

[0043] 表1

[0044]

进程组	负载跟踪策略
前台进程组	在历史五个负载跟踪记录的平均值与最近一个记录之中较大的一个
后台进程组	历史五个负载跟踪记录的平均值
顶级进程组	历史五个负载跟踪记录的最大值

[0045] 在本申请实施例中,采用源自控制组群(control groups,cgroups)策略为 不同进程组分配不同负载跟踪策略。其中,cgroups,是Linux内核的一个功能, 用来限制、控制与分离一个进程组的资源(如CPU、内存、磁盘输入输出等)。其最初的目标是为资源管理提供的一个统一的框架,既整合现有的cpuset等子 系统,也为未来开发新的子系统提供接口。现在的cgroups适用于多种应用场景, 从单个进程的资源控制,到实现操作系统层次的虚拟化(OS Level Virtualization)。

[0046] 也就是说,cgroups作为进程的管理容器,可以对每个进程组的各进程进行 管理。

[0047] 又例如,对于前台进程包括UI线程、Render线程、以及所有实时跟UI线 程或Render线程通信的线程。其中,UI线程和Render线程属于UX线程,那 么所对应的负载跟踪策略可以为历史对个负载跟踪记录的最大值,而对于非UX 线程,则所对应的负载跟踪策略就是历史多个负载跟踪记录的平均值或最小值。

[0048] 其中,线程则是某一进程中一路单独运行的程序。也就是说,线程存在于 进程之中,通常在一个进程中可以包含至少一个线程。线程可以利用进程所拥 有的资源,在引入线程的操作系统中,通常都是把进程作为分配资源的基本单 位,而把线程作为独立运行和独立调度的基本单位。由于线程比进程更小,基 本上不拥有系统资源,故对它的调度所付出的开销就会小得多,能更高效的提 高系统多个程序间并发执行的程度。

[0049] S102,基于所述第一运行时长、所述第二运行时长以及所述第一时间窗口 对应的窗口时长,统计所述进程在目标窗口的负载,所述目标窗口为所述第一 时间窗口或者所述第二时间窗口。

[0050] 可以理解的是,所述负载即为进程在时间窗口所占用的中央处理器 (Central Processing Unit,CPU)负载。CPU是一个计算机的运算核心和控制核 心,是信息处理、程序运行的最终执行单元。CPU包含运算逻辑部件、寄存器 部件和控制部件等,并具有处理指令、执行操作、控制时间、处理数据等功能。

[0051] CPU的性能主要体现在其运行程序的速度上。影响运行速度的性能指标包 括CPU

的工作频率、Cache容量、指令系统和逻辑结构等参数。CPU负载就是 CPU使用率，当CPU使用率较大时，导致CPU运行速度慢，可通过CPU调频 限制CPU的使用。

[0052] 每个时间窗口对应的窗口时长相同，也就是第一时间窗口对应的窗口时长 与第二时间窗口对应的窗口时长相同。

[0053] 具体的，比较所述第一运行时长和所述第二运行时长，确定其中的较大时 长，再计算所述第一运行时长T1与所述第二运行时长T2的和，得到总运行时 长T1+T2；计算所述总运行时长T1+T2与所述第一时间窗口对应的窗口时长 windowsize的商(T1+T2)/windowsize，将所述商(T1+T2)/windowsize作为 所述进程在所述较大时长对应的目标窗口的负载。其中，当所述第一运行时长 T1大于所述第二运行时T2时，将所述商作为所述进程在所述第一时间窗W1 的负载；当所述第一运行时长小于所述第二运行时长时，将所述商作为所述进 程在所述第二时间窗口W2的负载；当所述第一运行时长等于所述第二运行时长时，将所述商作为所述进程在所述第一时间窗口W1或者所述第二时间窗口 W2的负载。

[0054] 可选的，按照上述方式可得到目标时间窗口内所有进程的负载，再根据动 态电压频率调整策略(Dynamic voltage and frequency scaling，DVFS)以及目标 时间窗口内各进程的负载，对所述CPU进行调频处理。Linux内核中的DVFS 策略是关系到操作系统响应速度、功耗表现、运行时间的关键策略，DVFS的具 体调频算法与负载跟踪机制密不可分，正常情况下，DVFS会根据负载跟踪的结 果驱动CPU的调频处理。

[0055] 在本申请实施例中，即使进程跨窗口运行，通过结合跨越的多个时间窗口 的运行时间计算在某一个时间窗口的负载，而并非只依据某一个时间窗口的运 行时间统计负载，可以提高负载计算的准确性，使得统计的负载更接近真实负 载，进而可优化统计的负载偏小导致的卡顿问题。此外，通过预先对系统的进 程分类为不同的进程组，为不同的进程组配置不用的负载跟踪策略，采用相适 配的负载跟踪策略统计运行的进程在时间窗口内的负载，可以有效平衡系统性 能与功耗，如对性能需求高的进程能够更好的满足其性能，进而提升应用的用 户体验，对于性能需求不高的应用通过应用计算的负载偏低的负载跟踪策略，以节省系统的功耗，延长系统的续航。

[0056] 请参见图3，为本申请实施例提供的一种负载统计方法的流程示意图。图3 所示实施例相对于图2所示实施例的不同之处在于，结合附图描述了进程跨越 时间窗口的负载计算过程，该负载统计方法可以包括以下步骤：

[0057] S201，在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进 程在所述第一时间窗口的第一运行时长，并获取所述进程在所述第二时间窗口 的第二运行时长；

[0058] 在本申请实施例中，采用WALT的负载追踪统计方法统计各运行进程的负 载，其主体思想是将时间分为以window_size大小的多个时间窗口，如图4所示，(n-1)ws～n*ws为第一时间窗口，n*ws～(n+1)ws为第二时间窗口，两窗口 相邻且对应的窗口时长均为window_size(如20ms)。当然，时间窗口为周期窗口，包括多个。

[0059] 进程跨越第一时间窗口和第二时间窗口持续运行，即在时间上保持连续，总运行时长runtime＝e1-s1，在每个时间窗口内统计每一个任务的运行时间，即 该进程在第一时间窗口的第一运行时长为T1，在第二时间窗口的第二运行时长 为T2。

[0060] S202，获取所述第一运行时长和所述第二运行时长中的较大时长；

[0061] 比较T1和T2的大小，确定其中的较大值。包括T1>T2、T1<T2、T1＝T2 三种结果。

[0062] 若T1>T2,则将第一时间窗口确定为进程的运行窗口;若T1<T2,则将第 二时间窗口确定为进程的运行窗口,若T1＝T2,则将第一时间窗口或第二时间 窗口确定为进程的运行窗口。

[0063] S203,计算所述第一运行时长与所述第二运行时长的和,得到总运行时长;

[0064] 总运行时长runtime＝e1-s1＝T1+T2。

[0065] S204,当所述总运行时长大于所述窗口时长时,将所述总运行时长设置为 所述窗口时长;

[0066] 如果T1+T2>window_size,则设置runtime＝window_size,这是为了避免所 计算的负载大于100％。

[0067] S205,计算所述总运行时长与所述第一时间窗口对应的窗口时长的商,将 所述商作为所述进程在所述较大时长对应的目标窗口的负载,所述目标窗口为 所述第一时间窗口或者所述第二时间窗口。

[0068] 具体的,load＝(T1+T2)/window_size*100％,当T1+T2<window_size时, load<100％,当T1+T2>window_size或者T1+T2＝window_size,load＝100％。

[0069] 因此,当所述第一运行时长T1大于所述第二运行时长T2时,将load作为 所述进程在所述第一时间窗口(n-1)ws～n*ws的负载;当所述第一运行时长小 于所述第二运行时长时,将load作为所述进程在所述第二时间窗口n*ws～(n+1) ws的负载;当所述第一运行时长T1等于所述第二运行时长T2时,将load作为 所述进程在所述第一时间窗口(n-1)ws～n*ws或者所述第二时间窗口n*ws～(n+1) ws的负载。

[0070] 可选的,当该进程跨越大于两个时间窗口持续运行时,如图5所示,进程 跨越第一时间窗口(n-1)ws～n*ws、第二时间窗口n*ws～(n+1)ws以及第三 时间窗口(n+1)*ws～(n+2)ws,对应的运行时间分别为T1、T2和T3,T2>T1 且T2>T3,T2＝window_size,因此,可理解为该进程在第二时间窗口的负载为 100％。

[0071] 在本申请实施例中,即使进程跨窗口运行,通过计算跨越的多个时间窗口 的运行时间的总和来统计在运行时长较长的时间窗口的负载,而并非只依据该 时间窗口内的运行时间统计负载,采用与进程不跨窗口运行相同的方式计算负 载,区别在于确定进程所对应的运行时间窗口,采用上述方式可以提高负载计 算的准确性,使得统计的负载更接近真实负载,进而可优化统计的负载偏小导 致的卡顿问题。

[0072] 请参见图6,为本申请实施例提供的一种负载统计方法的流程示意图。图6 所示实施例相对于图3所示实施例的不同之处在于,强调的是进程在跨窗口运 行的情况下还包括非持续运行的情况,该负载统计方法可以包括以下步骤:

[0073] S301,在进程跨越第一时间窗口和第二时间窗口持续运行时,获取所述进 程在所述第一时间窗口的第一运行时长,并获取所述进程在所述第二时间窗口 的第二运行时长;

[0074] S302,获取所述第一运行时长和所述第二运行时长中的较大时长;

[0075] S301～S302具体可参见S201～S202,此处不再赘述。

[0076] S303,获取所述较大时长对应的目标窗口内所述进程非持续运行的第三运 行时长;

[0077] 若T2>T1,且在n*ws～(n+1)ws内该进程还运行持续时间为s2～e2的负载, 如图7所示,则第三运行时长T3＝e2-s2。

[0078] S304,计算所述第一运行时长、所述第二运行时长以及所述第三运行时长 的和,得到总运行时长;

[0079] 总运行时长runtime＝T1+T2+T3＝(e1-s1)+(e2-s2)

[0080] S305,当所述总运行时长大于所述窗口时长时,将所述总运行时长设置为 所述窗口时长;

[0081] 如果(e1-s1)+(e2-s2)>window size的话,则设置runtime＝window size。

[0082] S306,计算所述总运行时长与所述第一时间窗口对应的窗口时长的商,将 所述商作为所述进程在所述较大时长对应的目标窗口的负载,所述目标窗口为 所述第一时间窗口或者所述第二时间窗口。

[0083] 若T2>T1,且在n*ws～(n+1)ws内该进程还运行持续时间为s2～e2的负载, 如图7所示,当(e1-s1)+(e2-s2)<window size时,load＝((e1-s1)+(e2-s2))/ window_size*100%,当(e1-s1)+(e2-s2)>window size时,load＝window size/ window_size*100%,将所计算的load确定为该进程在第二时间窗口对应的负载。

[0084] 若T2<T1,但在n*ws～(n+1)ws内该进程还运行持续时间为s2～e2的负载, 则runtime＝e1-s1＝T1+T2,当e1-s1<window size时,load＝(e1-s1)/window_size *100%,当(e1-s1)>window size时,load＝window size/window_size*100%, 将所计算的load确定为该进程在第一时间窗口对应的负载。

[0085] 在本申请实施例中,即使进程跨窗口运行,通过计算跨越的多个时间窗口 的运行时间的总和来统计在某一个时间窗口的负载,而并非只依据某一个时间 窗口的运行时间统计负载,可以提高负载计算的准确性,此外,当该进程在运 行时间窗口内还包括非持续运行的负载时,也一并计入该进程的负载统计结果 中,可以更进一步提高负载计算的准确性。因此,采用WALT的负载跟踪策略, 通过一个时间窗口即可统计出进程的负载,在保证负载统计快的情况下还可准 确统计跨时间窗口运行的进程的负载,使得统计的负载更接近真实负载,进而 可优化统计的负载偏小导致的卡顿问题。

[0086] 下述为本申请装置实施例,可以用于执行本申请方法实施例。对于本申请 装置实施例中未披露的细节,请参照本申请方法实施例。

[0087] 请参见图8,其示出了本申请一个示例性实施例提供的负载统计装置的结构 示意图。该负载统计装置可以通过软件、硬件或者两者的结合实现成为用户终 端的全部或一部分。该装置1包括运行时长获取模块10和负载统计模块20。

[0088] 运行时长获取模块10,用于在进程跨越第一时间窗口和第二时间窗口持续 运行时,获取所述进程在所述第一时间窗口的第一运行时长,并获取所述进程 在所述第二时间窗口的第二运行时长;

[0089] 负载统计模块20,用于基于所述第一运行时长、所述第二运行时长以及所 述第一时间窗口对应的窗口时长,统计所述进程在目标窗口的负载,所述目标 窗口为所述第一时间窗口或者所述第二时间窗口。

[0090] 可选的,如图9所示,所述负载统计模块20,包括:

[0091] 较大时长获取单元21,用于获取所述第一运行时长和所述第二运行时长中 的较大时长;

[0092] 总时长计算单元22,用于计算所述第一运行时长与所述第二运行时长的和, 得到

总运行时长；

[0093] 负载计算单元23，用于计算所述总运行时长与所述第一时间窗口对应的窗 口时长的商，将所述商作为所述进程在所述较大时长对应的目标窗口的负载。

[0094] 可选的，所述负载计算单元23，具体用于：

[0095] 当所述第一运行时长大于所述第二运行时长时，将所述商作为所述进程在 所述第一时间窗口的负载；

[0096] 当所述第一运行时长小于所述第二运行时长时，将所述商作为所述进程在 所述第二时间窗口的负载；

[0097] 当所述第一运行时长等于所述第二运行时长时，将所述商作为所述进程在 所述第一时间窗口或者所述第二时间窗口的负载。

[0098] 可选的，所述总时长计算单元22，具体用于：

[0099] 获取所述较大时长对应的目标窗口内所述进程非持续运行的第三运行时长；

[0100] 计算所述第一运行时长、所述第二运行时长以及所述第三运行时长的和，得到总运行时长。

[0101] 可选的，如图9所示，所述负载统计模块20，还包括时长设置单元24，用 于当所述总运行时长大于所述窗口时长时，将所述总运行时长设置为所述窗口 时长。

[0102] 需要说明的是，上述实施例提供的负载统计装置在执行负载统计方法时，仅以上述各功能模块的划分进行举例说明，实际应用中，可以根据需要而将上 述功能分配由不同的功能模块完成，即将设备的内部结构划分成不同的功能模 块，以完成以上描述的全部或者部分功能。另外，上述实施例提供的负载统计 装置与负载统计方法实施例属于同一构思，其体现实现过程详见方法实施例，这里不再赘述。

[0103] 上述本申请实施例序号仅仅为了描述，不代表实施例的优劣。

[0104] 在本申请实施例中，在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进程在所述第一时间窗口的第一运行时长，并获取所述进程在所述第 二时间窗口的第二运行时长，基于所述第一运行时长、所述第二运行时长以及 所述第一时间窗口对应的窗口时长，统计所述进程在目标窗口的负载，所述目 标窗口为所述第一时间窗口或者所述第二时间窗口。即使进程跨窗口运行，通 过计算跨越的多个时间窗口的运行时间的总和来统计在某一个时间窗口的负载，而并非只依据某一个时间窗口的运行时间统计负载，可以提高负载计算的准确 性，此外，当该进程在运行时间窗口内还包括非持续运行的负载时，也一并计 入该进程的负载统计结果中，可以更进一步提高负载计算的准确性。因此，采 用WALT的负载跟踪策略，通过一个时间窗口即可统计出进程的负载，在保证负 载统计快的情况下还可准确统计跨时间窗口运行的进程的负载，使得统计的负 载更接近真实负载，进而可优化统计的负载偏小导致的卡顿问题。

[0105] 本申请实施例还提供了一种计算机存储介质，所述计算机存储介质可以存 储有多条指令，所述指令适于由处理器加载并执行如上述图2-图7所示实施例 的方法步骤，具体执行过程可以参见图2-图7所示实施例的具体说明，在此不 进行赘述。

[0106] 请参见图10，为本申请实施例提供了一种电子设备的结构示意图。如图10 所示，所述电子设备1000可以包括：至少一个处理器1001，至少一个网络接口 1004，用户接口1003，存储器1005，至少一个通信总线1002。

[0107] 其中,通信总线1002用于实现这些组件之间的连接通信。

[0108] 其中,用户接口1003可以包括显示屏(Display)、摄像头(Camera),可选 用户接口1003还可以包括标准的有线接口、无线接口。

[0109] 其中,网络接口1004可选的可以包括标准的有线接口、无线接口(如WI-FI 接口)。

[0110] 其中,处理器1001可以包括一个或者多个处理核心。处理器1001利用各 种借口和线路连接整个电子设备1000内的各个部分,通过运行或执行存储在存 储器1005内的指令、程序、代码集或指令集,以及调用存储在存储器1005内 的数据,执行电子设备1000的各种功能和处理数据。可选的,处理器1001可 以采用数字信号处理(Digital Signal Processing,DSP)、现场可编程门阵列 (Field-Programmable Gate Array,FPGA)、可编程逻辑阵列(Programmable Logic Array,PLA)中的至少一种硬件形式来实现。处理器1001可集成中央处理器 (Central Processing Unit,CPU)、图像处理器(Graphics Processing Unit,GPU) 和调制解调器等中的一种或几种的组合。其中,CPU主要处理操作系统、用户 界面和应用程序等;GPU用于负责显示屏所需要显示的内容的渲染和绘制;调 制解调器用于处理无线通信。可以理解的是,上述调制解调器也可以不集成到 处理器1001中,单独通过一块芯片进行实现。

[0111] 其中,存储器1005可以包括随机存储器(Random Access Memory,RAM), 也可以包括只读存储器(Read-Only Memory)。可选的,该存储器1005包括非 瞬时性计算机可读介质(non-transitory computer-readable storage medium)。存储 器1005可用于存储指令、程序、代码、代码集或指令集。存储器1005可包括 存储程序区和存储数据区,其中,存储程序区可存储用于实现操作系统的指令、用于至少一个功能的指令(比如触控功能、声音播放功能、图像播放功能等)、用于实现上述各个方法实施例的指令等;存储数据区可存储上面各个方法实施 例中涉及到的数据等。存储器1005可选的还可以是至少一个位于远离前述处理 器1001的存储装置。如图10所示,作为一种计算机存储介质的存储器1005中 可以包括操作系统、网络通信模块、用户接口模块以及负载统计应用程序。

[0112] 在图10所示的电子设备1000中,用户接口1003主要用于为用户提供输入 的接口,获取用户输入的数据;而处理器1001可以用于调用存储器1005中存 储的负载统计应用程序,并具体执行以下操作:

[0113] 在进程跨越第一时间窗口和第二时间窗口持续运行时,获取所述进程在所 述第一时间窗口的第一运行时长,并获取所述进程在所述第二时间窗口的第二 运行时长;

[0114] 基于所述第一运行时长、所述第二运行时长以及所述第一时间窗口对应的 窗口时长,统计所述进程在目标窗口的负载,所述目标窗口为所述第一时间窗 口或者所述第二时间窗口。

[0115] 在一个实施例中,所述处理器1001在执行基于所述第一运行时长、所述第 二运行时长以及所述第一时间窗口对应的窗口时长,统计所述进程在目标窗口 的负载时,具体执行以下操作:

[0116] 获取所述第一运行时长和所述第二运行时长中的较大时长;

[0117] 计算所述第一运行时长与所述第二运行时长的和,得到总运行时长;

[0118] 计算所述总运行时长与所述第一时间窗口对应的窗口时长的商,将所述商 作为所述进程在所述较大时长对应的目标窗口的负载。

[0119] 在一个实施例中，所述处理器1001在执行将所述商作为所述进程在所述较 大时长对应的目标窗口的负载时，具体执行以下操作：

[0120] 当所述第一运行时长大于所述第二运行时长时，将所述商作为所述进程在 所述第一时间窗口的负载；

[0121] 当所述第一运行时长小于所述第二运行时长时，将所述商作为所述进程在 所述第二时间窗口的负载；

[0122] 当所述第一运行时长等于所述第二运行时长时，将所述商作为所述进程在 所述第一时间窗口或者所述第二时间窗口的负载。

[0123] 在一个实施例中，当所述进程组为顶级进程组时，所述处理器1001在执行 计算所述第一运行时长与所述第二运行时长的和，得到总运行时长时，具体执 行以下操作：

[0124] 获取所述较大时长对应的目标窗口内所述进程非持续运行的第三运行时长；

[0125] 计算所述第一运行时长、所述第二运行时长以及所述第三运行时长的和，得到总运行时长。

[0126] 在一个实施例中，所述处理器1001在执行得到总运行时长之后，还执行以 下操作：

[0127] 当所述总运行时长大于所述窗口时长时，将所述总运行时长设置为所述窗 口时长。

[0128] 在本申请实施例中，在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进程在所述第一时间窗口的第一运行时长，并获取所述进程在所述第 二时间窗口的第二运行时长，基于所述第一运行时长、所述第二运行时长以及 所述第一时间窗口对应的窗口时长，统计所述进程在目标窗口的负载，所述目 标窗口为所述第一时间窗口或者所述第二时间窗口。即使进程跨窗口运行，通 过计算跨越的多个时间窗口的运行时间的总和来统计在某一个时间窗口的负载，而并非只依据某一个时间窗口的运行时间统计负载，可以提高负载计算的准确 性，此外，当该进程在运行时间窗口内还包括非持续运行的负载时，也一并计 入该进程的负载统计结果中，可以更进一步提高负载计算的准确性。因此，采 用WALT的负载跟踪策略，通过一个时间窗口即可统计出进程的负载，在保证 负载统计快的情况下还可准确统计跨时间窗口运行的进程的负载，使得统计的 负载更接近真实负载，进而可优化统计的负载偏小导致的卡顿问题。

[0129] 本领域普通技术人员可以理解实现上述实施例方法中的全部或部分流程，是可以通过计算机程序来指令相关的硬件来完成，所述的程序可存储于一计算 机可读取存储介质中，该程序在执行时，可包括如上述各方法的实施例的流程。其中，所述的存储介质可为磁碟、光盘、只读存储记忆体或随机存储记忆体等。

[0130] 以上所揭露的仅为本申请较佳实施例而已，当然不能以此来限定本申请之 权利范围，因此依本申请权利要求所作的等同变化，仍属本申请所涵盖的范围。

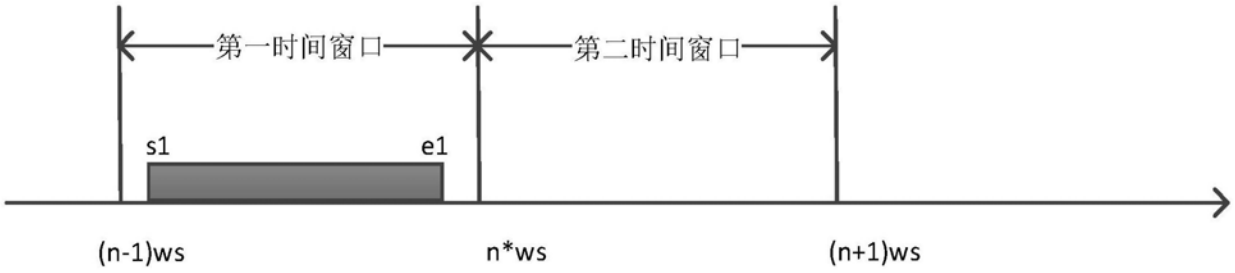

第一时间窗口
第二时间窗口
s1
e1
(n-1)ws
n*ws
(n+1)ws

图1

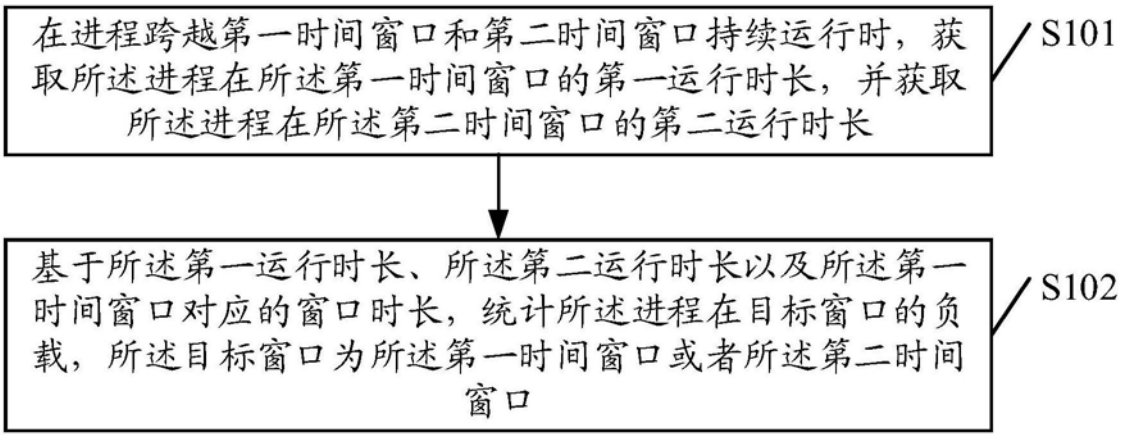

在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进程在所述第一时间窗口的第一运行时长，并获取所述进程在所述第二时间窗口的第二运行时长
S101
基于所述第一运行时长、所述第二运行时长以及所述第一时间窗口对应的窗口时长，统计所述进程在目标窗口的负载，所述目标窗口为所述第一时间窗口或者所述第二时间窗口
S102

图2

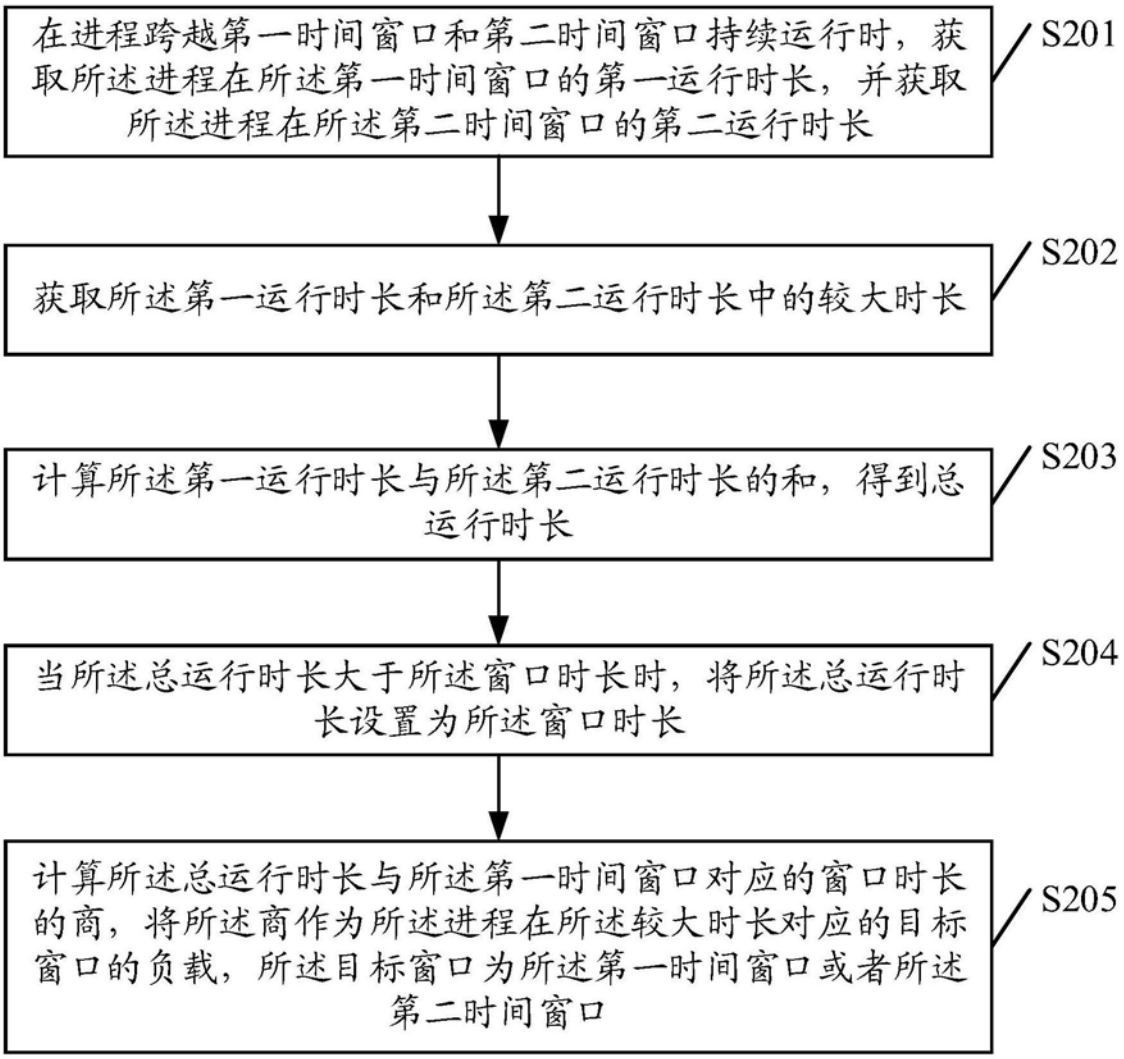

在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进程在所述第一时间窗口的第一运行时长，并获取所述进程在所述第二时间窗口的第二运行时长
S201
获取所述第一运行时长和所述第二运行时长中的较大时长
S202
计算所述第一运行时长与所述第二运行时长的和，得到总运行时长
S203
当所述总运行时长大于所述窗口时长时，将所述总运行时长设置为所述窗口时长
S204
计算所述总运行时长与所述第一时间窗口对应的窗口时长的商，将所述商作为所述进程在所述较大时长对应的目标窗口的负载，所述目标窗口为所述第一时间窗口或者所述第二时间窗口
S205

图3

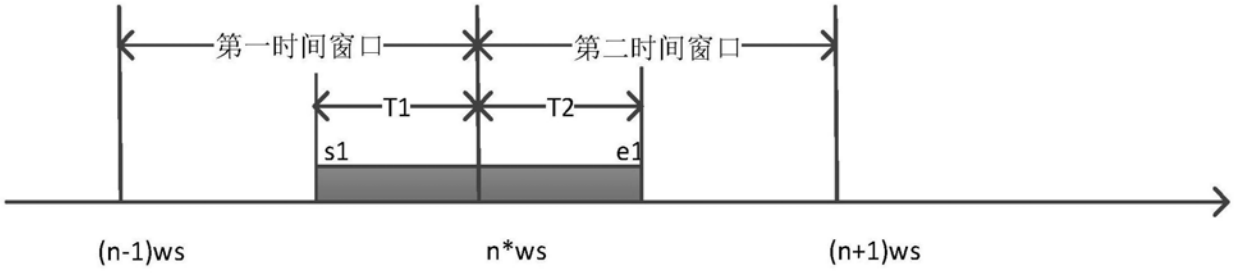

第一时间窗口
第二时间窗口
T1
T2
s1
e1
(n-1)ws
n*ws
(n+1)ws

图4

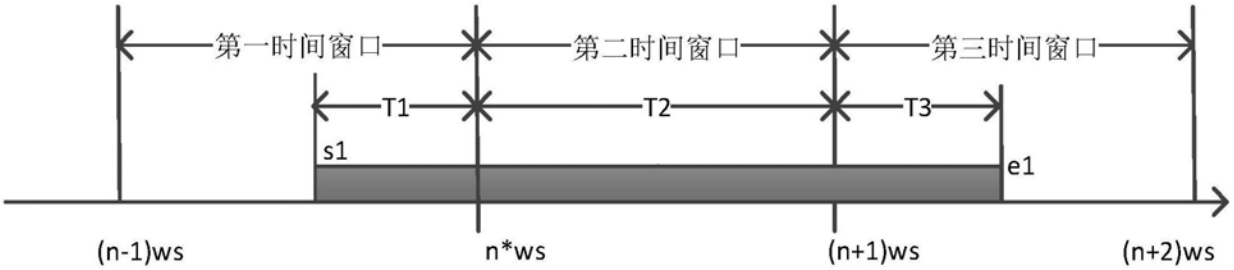

第一时间窗口
第二时间窗口
第三时间窗口
T1
T2
T3
s1
e1
(n-1)ws
n*ws
(n+1)ws
(n+2)ws

图5

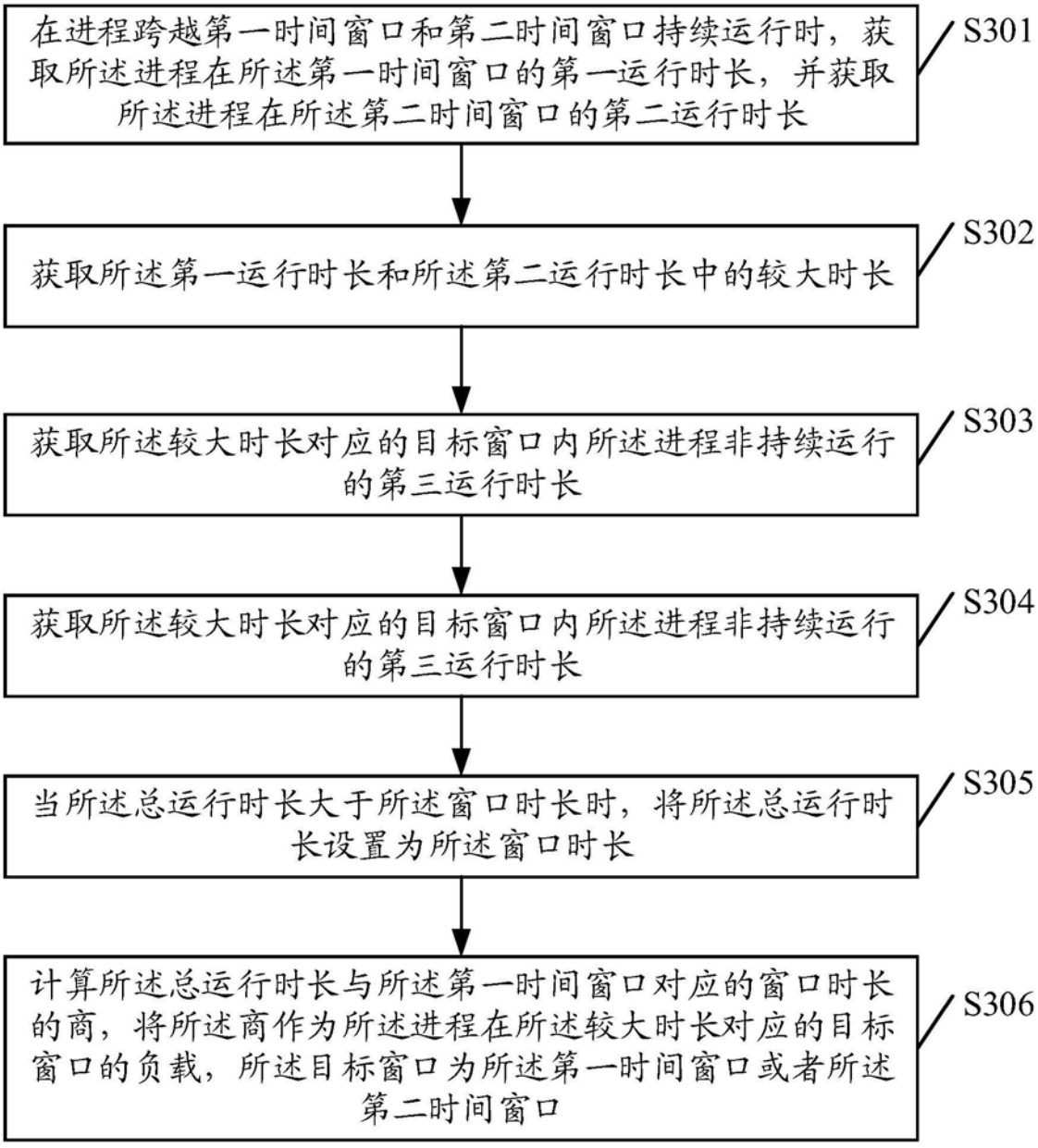

在进程跨越第一时间窗口和第二时间窗口持续运行时，获取所述进程在所述第一时间窗口的第一运行时长，并获取所述进程在所述第二时间窗口的第二运行时长
S301
获取所述第一运行时长和所述第二运行时长中的较大时长
S302
获取所述较大时长对应的目标窗口内所述进程非持续运行的第三运行时长
S303
获取所述较大时长对应的目标窗口内所述进程非持续运行的第三运行时长
S304
当所述总运行时长大于所述窗口时长时，将所述总运行时长设置为所述窗口时长
S305
计算所述总运行时长与所述第一时间窗口对应的窗口时长的商，将所述商作为所述进程在所述较大时长对应的目标窗口的负载，所述目标窗口为所述第一时间窗口或者所述第二时间窗口
S306

图6

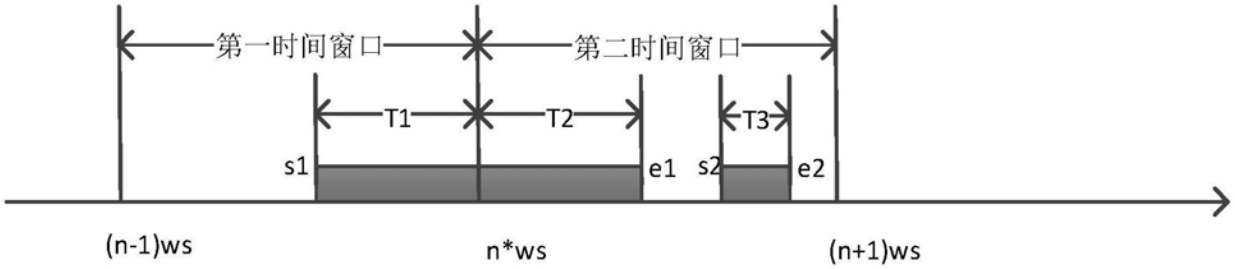

第一时间窗口
第二时间窗口
T1
T2
T3
s1
e1
s2
e2
(n-1)ws
n*ws
(n+1)ws

图7

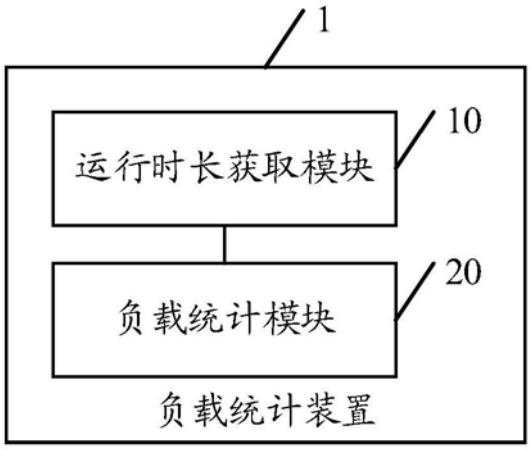

1
运行时长获取模块
10
负载统计模块
20
负载统计装置

图8

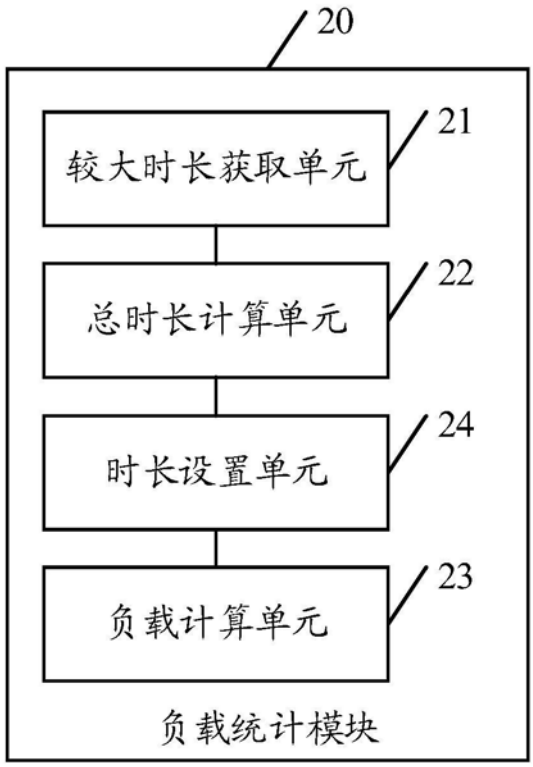

20
21
较大时长获取单元
22
总时长计算单元
24
时长设置单元
23
负载计算单元
负载统计模块

图9

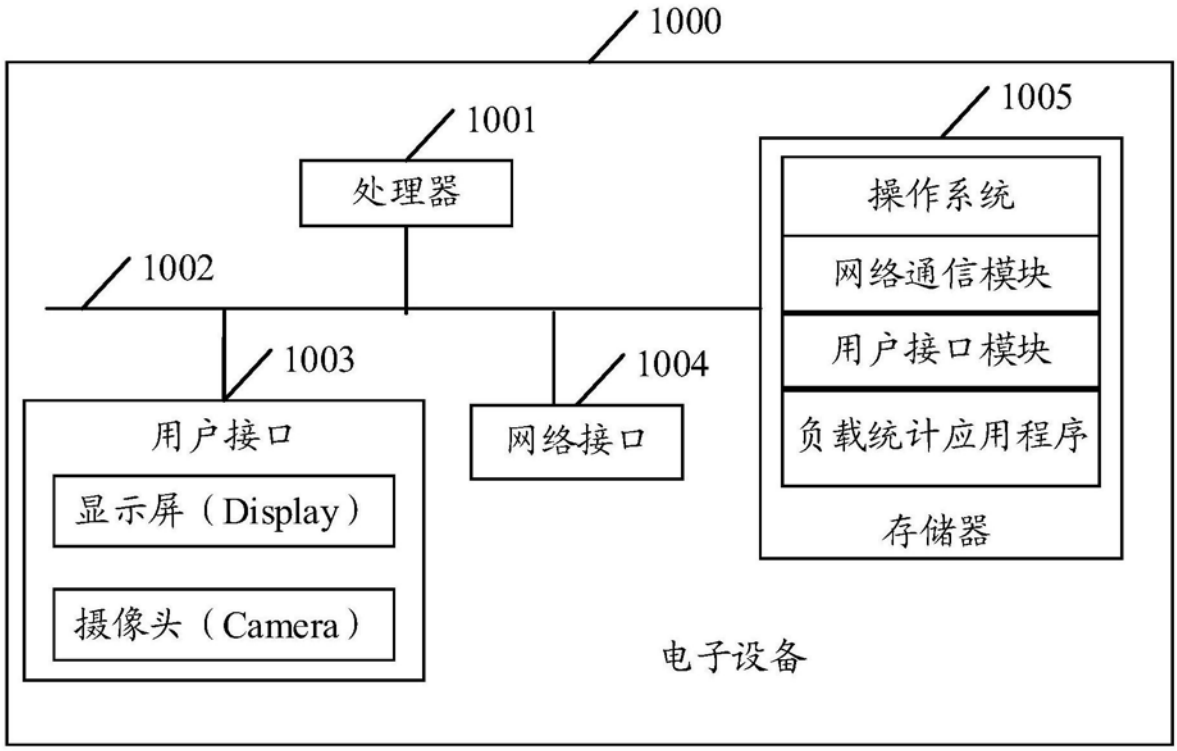

1000
1001
处理器
1005
操作系统
网络通信模块
用户接口模块
负载统计应用程序
存储器
1002
1003
用户接口
显示屏（Display）
摄像头（Camera）
1004
网络接口
电子设备

图10