

一种任务的调度方法、计算设备及存储介质

本申请要求 2020 年 05 月 29 日递交的申请号为 202010475800.4、发明名称为“一种任务的调度方法、计算设备及存储介质”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。

5 技术领域

本申请涉及计算机技术领域，尤其涉及一种任务的调度方法、计算设备及存储介质。
背景技术

随着互联网高速发展，云服务相比于传统的计算服务，具有廉价、易扩展、可伸缩、易管理、高可用等多项优势，因此在信息互联网时代占据着越来越关键的地位。而容器
10 技术由于它所具有的特性，被广泛的应用在云服务领域。

但由于人工智能的迅猛发展，异构计算成为了云服务的新兴热点。由此，云服务器中可以包含多个种类的硬件资源，如多个种类的物理处理设备等等。而如何高效而又便捷的调度这些硬件资源，是调度资源的关键。

发明内容

15 本申请的多个方面提供一种任务的调度方法、计算设备及存储介质，用以能够实现高效而又便捷的调度硬件资源。

本申请实施例提供一种任务的调度方法，包括：获取本次待执行任务的任务信息，所述任务信息描述了所述本次待执行任务；根据所述任务信息，确定所述本次待执行任务的实际执行时间；将所述实际执行时间与所述本次待执行任务对应的预置调度时间进行对比，根据对比结果，对下一次待执行任务对应的实际调度时间进行调节，以确定是否对下一次待执行任务进行调度。
20

本申请实施例还提供一种容器任务的调度方法，包括：接收任一容器发送的本次待执行任务，获取所述本次待执行任务的任务信息，所述任务信息描述了所述本次待执行任务；根据所述任务信息，确定所述本次待执行任务的实际执行时间；将所述实际执行时间与所述本次待执行任务对应的预置调度时间进行对比，根据对比结果，对同一容器的下一次待执行任务对应的实际调度时间进行调节，以确定是否对同一容器的下一次待执行任务进行调度。
25

本申请实施例还提供一种任务的调度装置，包括：获取模块，用于获取本次待执行任务的任务信息，所述任务信息描述了所述本次待执行任务；确定模块，用于根据所述任务信息，确定所述本次待执行任务的实际执行时间；调节模块，用于将所述实际执行时间与所述本次待执行任务对应的预置调度时间进行对比，根据对比结果，对下一次待执行任务对应的实际调度时间进行调节，以确定是否对下一次待执行任务进行调度。
30

本申请实施例还提供一种容器任务的调度装置，包括：接收模块，用于接收任一容

器发送的本次待执行任务，获取所述本次待执行任务的任务信息，所述任务信息描述了所述本次待执行任务；确定模块，用于根据所述任务信息，确定所述本次待执行任务的实际执行时间；调节模块，用于将所述实际执行时间与所述本次待执行任务对应的预置调度时间进行对比，根据对比结果，对同一容器的下一次待执行任务对应的实际调度时间进行调节，以确定是否对同一容器的下一次待执行任务进行调度。

5 本申请实施例还提供一种计算设备，包括：存储器以及处理器；所述存储器，用于存储计算机程序；所述处理器，用于执行所述计算机程序，以用于：获取本次待执行任务的任务信息，所述任务信息描述了所述本次待执行任务；根据所述任务信息，确定所述本次待执行任务的实际执行时间；将所述实际执行时间与所述本次待执行任务对应的
10 预置调度时间进行对比，根据对比结果，对下一次待执行任务对应的实际调度时间进行调节，以确定是否对下一次待执行任务进行调度。

本申请实施例还提供一种计算设备，包括：存储器以及处理器；所述存储器，用于存储计算机程序；所述处理器，用于执行所述计算机程序，以用于：接收任一容器发送的本次待执行任务，获取所述本次待执行任务的任务信息，所述任务信息描述了所述本次待执行任务；根据所述任务信息，确定所述本次待执行任务的实际执行时间；将所述
15 实际执行时间与所述本次待执行任务对应的预置调度时间进行对比，根据对比结果，对同一容器的下一次待执行任务对应的实际调度时间进行调节，以确定是否对同一容器的下一次待执行任务进行调度。

本申请实施例还提供一种存储有计算机程序的计算机可读存储介质，计算机程序被
20 一个或多个处理器执行时，致使所述一个或多个处理器实现上述方法中的步骤。

在本申请实施例中，获取本次待执行任务的任务信息，任务信息描述了本次待执行任务；根据任务信息，确定本次待执行任务的实际执行时间；将实际执行时间与本次待执行任务对应的预置调度时间进行对比，根据对比结果，对下一次待执行任务对应的实际调度时间进行调节，以确定是否对下一次待执行任务进行调度。由于在确定出实际执行
25 行时间后，与预置调度时间对比，就可以对下一次待执行任务对应的实际调度时间进行调节，决定是否进行任务调度，从而快速且方便地实现了任务的调度，充分利用硬件资源。同时，由于对实际调度时间进行了调节，则可以准确高效地分配各个任务的实际调度时间，平衡各个任务之间的调度时间，减少本次待执行任务过度占用其它任务的调度时间的情况发生。

30 此外，由于没有通过设备驱动来获取驱动方式来进行任务调度，所以对于未开源的设备驱动的而言，是非常友好的，无需再去关心设备驱动是否开源（即可以是开源的，也可以是未开源的），也可以减少设备驱动的开发。

附图说明

此处所说明的附图用来提供对本申请的进一步理解，构成本申请的一部分，本申请

的示意性实施例及其说明用于解释本申请，并不构成对本申请的不当限定。在附图中：

图 1 为本申请一示例性实施例的任务的调度系统的结构示意图；

图 2 为本申请一示例性实施例的任务的调度方法的流程示意图；

图 3 为本申请又一示例性实施的任务的调度方法的流程示意图；

5 图 4 为本申请又一示例性实施的容器任务的调度方法的流程示意图；

图 5 为本申请又一示例性实施例提供的任务的调度装置的结构示意图；

图 6 为本申请又一示例性实施例提供的容器任务的调度装置的结构示意图；

图 7 为本申请一示例性实施例提供的计算设备的结构示意图；

图 8 为本申请又一示例性实施例提供的计算设备的结构示意图。

10 具体实施方式

为使本申请的目的、技术方案和优点更加清楚，下面将结合本申请具体实施例及相应的附图对本申请技术方案进行清楚、完整地描述。显然，所描述的实施例仅是本申请一部分实施例，而不是全部的实施例。基于本申请中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本申请保护的范围。

15 容器技术是一种将软件打包成标准化单元的技术，以用于开发、交付和部署。它确保了应用运行环境一致性，能够更快速的启动，具有：隔离性、可扩展性、迁移方便、可持续交付和部署等特点。基于以上特性，容器技术被广泛的应用在云服务领域。由于人工智能的迅猛发展，异构计算成为了云服务的新兴热点。

在包含有 GPU（Graphics Processing Unit，图像处理器）、NPU（neural-network
20 processing units，嵌入式神经网络处理器）等多个处理设备以及容器的云服务器中，如何高效而又便捷的调度 GPU 硬件资源，使其在多个容器内可以无缝共享，同时调度算法自身的开销要保证足够小，是关键核心。

本申请实施例提出了一种方法，可以满足以上的要求，对于未开源的黑盒 GPU 设备驱动而言，也可以很好的进行调度。

25 以下结合附图，详细说明本申请各实施例提供的技术方案。

图 1 为本申请一示例性实施例提供的一种任务的调度系统的结构示意图。如图 1 所示，该系统 100 可以包括：第一设备 101 以及第二设备 102。

其中，第一设备 101 可以是有一定计算能力的设备，可以实现向第二设备 102 发送数据的功能，并从第二设备 102 获取到响应数据。第一设备 101 的基本结构可以包括：
30 至少一个处理器。处理器的数量可以取决于具有一定计算能力装置的配置和类型。具有一定计算能力装置也可以包括存储器，该存储器可以为易失性的，例如 RAM，也可以为非易失性的，例如只读存储器（Read-Only Memory，ROM）、闪存等，或者也可以同时包括两种类型。存储器内通常存储有操作系统（Operating System，OS）、一个或多个应用程序，也可以存储有程序数据等。除了处理单元和存储器之外，具有一定计算能力装

置还包括一些基本配置，例如网卡芯片、IO 总线、显示组件以及一些外围设备等。可选地，一些外围设备可以包括，例如键盘、输入笔等。其它外围设备在本领域中是众所周知的，在此不做赘述。可选地，第一设备 101 可以为智能终端，例如，手机、台式电脑、笔记本、平板电脑等。

5 第二设备 102 是指可以在网络虚拟环境中提供计算处理服务的设备，可以是指利用网络进行数据处理且进行任务调度的设备。在物理实现上，第二设备 102 可以是任何能够提供计算服务，响应服务请求，并返回数据处理结果，例如可以是云服务器、云主机、虚拟中心、常规服务器等等。第二设备 102 的构成主要包括处理器、硬盘、内存、系统总线等，和通用的计算机架构类似。

10 在本申请实施例中，第一设备 101，发送数据至第二设备 102。第二设备 102 接收到该数据，将该数据发送至对应的虚拟设备中，如容器。在第二设备 102 中，包括至少一个虚拟设备，如容器。同时，第二设备 102 中还可以包括设备驱动（也可以称为设备驱动层，或者设备驱动程序）以及中间驱动层。在该中间驱动层中可以设置有任务调度器，用于进行任务调度。中间驱动层与设备驱动通过通信接口进行通信。当第二设备 102 中的虚拟设备接收到该数据后，需要对该数据处理，将该数据处理生成任务（即待执行任务），
15 发送至中间驱动层中的任务调度器。任务调度器收到后，获取本次待执行任务的执行信息，该任务信息描述了本次待执行任务；根据任务信息，确定待执行任务的实际执行时间；将实际执行时间与本次待执行任务对应的预置调度时间进行对比，根据对比结果，对下一次待执行任务对应的实际调度时间进行调节，以确定是否下一次待执行任务（可以是同
20 一起来源的，也可以是不同来源的）进行调度。同时，任务调度器将该待执行任务通过设备驱动发送至对应的处理设备，如 GPU，以使其来处理该任务，对应的处理设备处理该任务后，将任务结果返回给对应的虚拟设备。再由该虚拟设备返回任务结果至第一设备 101。

25 需要说明的是，虚拟设备将任务结果返回至第一设备 101 的过程，可以是一个第二设备 102 中设备驱动执行的过程，具体的执行过程与前述的内容，相似，此处就不再说明。

此外，在驱动设备属于开源的前提下，也可以将该任务调度器设置在驱动设备中。无需在另部署一个中间驱动层。

30 在本申请实施例的应用场景中，用户通过第一设备 101，如电脑，向第二设备 102，如云服务器，发送数据，如用于账号登录请求的请求，该请求可以携带账号和密码。用户可以通过第一设备 101 安装的浏览器发送数据。第二设备 102 中的虚拟设备，如容器 1 接收该数据，并获取到账号以及密码。容器 1，向第二设备 102 的中间驱动层中的任务调度器发送任务，该任务可以为确认该账号和密码是否正确的执行指令。任务调度器接收到该任务后，根据该任务携带的存储地址，获取到上述执行指令，可以将该执行指令

输入至预置模型中，进行实际执行时间的预测。同时，任务调度器可以将该任务通过设备驱动发送至对应的处理设备，如 GPU。GPU 接收到该任务后，处理该任务。当处理结果时账号以及密码正确，则可以确定结果正确，可以登录。容器 1 获取到该处理结果，并通过第二设备 102 返回至用户的电脑，显示登录成功。

- 5 此外，任务调度器在预测了实际执行时间后，和对应的预置调度时间进行对比，当大于预置调度时间，确定超时时间。并将容器 1 的下一待执行任务的实际调度时间进行减少，减少的时间大小为超时时间的大小，然后将更新后的实际调度时间作为后续待执行任务的实际调度时间。直至容器 1 后续待执行任务的实际调度时间被更新减少到为 0，则可以在对该后续待执行任务进行调度的时候，不进行调度，等待下一次可进行任务调度的时候，如下一次预置调度时间到来后，在对该待执行任务进行调度。
- 10

若小于预置调度时间，确定空闲时间。此时，任务调度器可以不做处理，或者，将该空闲时间补偿给下一待执行任务（可以是同一个容器的任务，也可不是同一个容器的任务）的实际调度时间。任务调度器执行完本次任务的实际调度时间的调节以及任务调度后，进行下一次任务的调度，即重复上述内容。

- 15 在上述本实施例中，第一设备 101 以及第二设备 102 进行网络连接，该网络连接可以是无线连接。若第一设备 101 以及第二设备 102 是通信连接，该移动网络的网络制式可以为 2G（GSM）、2.5G（GPRS）、3G（WCDMA、TD-SCDMA、CDMA2000、UTMS）、4G（LTE）、4G+（LTE+）、WiMax、5G 等中的任意一种。

下面结合方法实施例，针对任务的调度过程进行详细说明。

- 20 图 2 为本申请一示例性实施例的一种任务的调度方法的流程示意图。本申请实施例提供的该方法 200 由计算设备执行，如，云服务器，更具体的可以是云服务器中的任务调度器。该方法 200 包括以下步骤：

201：获取本次待执行任务的任務信息，任务信息描述了本次待执行任务。

202：根据任务信息，确定本次待执行任务的实际执行时间。

- 25 203：将实际执行时间与本次待执行任务对应的预置调度时间进行对比，根据对比结果，对下一次待执行任务对应的实际调度时间进行调节，以确定是否对下一次待执行任务进行调度。

- 需要说明的是，根据前文可知，任务调度器可以设置在云服务器中的设备驱动中，也可以设置在中间驱动层中，该中间驱动层可以部署在设备驱动之上，即中间驱动层与设备驱动进行通信连接。任务调度器部署在中间驱动层中时，可以不用在意设备驱动是否开源了，是否属于三方。
- 30

以下针对上述步骤进行详细阐述：

201：获取本次待执行任务的任務信息，任务信息描述了本次待执行任务。

其中，待执行任务（以下也可以简称为待执行任务）是指由物理设备中的硬件资源

(也可以称为硬件设备)进行处理的任务,如由云服务器中的 GPU 处理的任务,该任务可以是指明至少一个处理设备进行确认密码和用户名是否正确的任务。本次待执行任务则是当下任务调度器正在准备进行调度的任务。

5 硬件资源还可以包括,如 CPU、NPU、内存、磁盘、网络等,除去处理设备外,其它硬件设备可以辅助处理设备进行任务的处理等,从而也可以均衡对其它硬件设备的利用。

获取本次待执行任务的方式可以包括:通过通信接口,接收虚拟设备发送的本次待执行任务,每个虚拟设备对应一个独立的通信接口。即每个虚拟设备与通信接口具有一一对应关系。且每个虚拟设备和通信接口都有自己的标识,如 ID 等。所以根据该对应关系,云服务器中的任务调度器在接收到本次待执行任务时,可以根据用于传输该待执行任务的通信接口的标识,知道该本次待执行任务所属的虚拟设备。

其中,该通信接口可以通过创建节点文件来实现通信的。

15 虚拟设备是指直接基于物理设备的操作系统运行的独立设备,如容器,沙箱等。其中物理设备可以是指物理主机,如云服务器主机。应理解的是,对于虚拟机而言,它不是直接基于物理设备的操作系统运行的,而是间接基于该操作系统运行的,因为对于虚拟机而言,它具有自己的虚拟操作系统,以及还具有虚拟硬件资源等,如虚拟 CPU 等,但这并真正不属于物理设备的硬件资源,是对该硬件资源进行了虚拟化。

此外,能够实现本申请实施例的虚拟设备都属于本申请实施例的保护范围,不仅限于容器。

20 还需要说明的是,对于虚拟设备而言,是已经创建在了物理设备上的了。或者说是,物理设备可以根据需求,对虚拟设备进行创建。由于创建过程属于现有技术,此处不再赘述。应理解,本申请实施例是基于物理设备已经创建好了多个虚拟设备而言的,或者是至少一个虚拟设备而言的。其中,虚拟设备可以创建在物理设备中的用户态,用户态是指操作系统(如, Linux 操作系统)中的一种权限级别,在用户态中,不允许其中的程序进行处理器(如, CPU、GPU 等)中要求特权态的操作,以避免操作系统崩溃。

25 具体的,获取本次待执行任务的任务信息,包括:根据本次待执行任务携带的存储地址,获取任务信息。

其中,任务信息描述了本次待执行任务,如,本次待执行任务的执行指令。

30 存储地址是指用于存储任务信息的地址,如缓冲区的缓冲地址或者缓存区的缓存地址。

需要说明的是,待执行任务除了可以携带存储地址外,还可以具有其它的信息,如指明哪一个处理设备进行任务处理(携带处理设备的标识)等等。

例如,云服务器中的任务调度器接收到本次待执行任务 a(以下也可以简称为任务 a)后,读取该本次待执行任务 a,从中获取其携带的命令缓冲区的缓冲地址。任务调度器

读取该缓冲地址，读取对应的执行指令。即如图 3 所示，执行了步骤 301：读取命令缓冲区中的信息。

需要说明的是，该缓冲地址可以是对应缓冲区的起始地址以及缓冲区长度，从而使得任务调度器到对应缓冲区中的对应缓冲位置读取执行指令。也可以是缓冲区的头尾指针，从而使得任务调度器到对应缓冲区中的对应缓冲位置读取执行指令。在存在多个缓冲区时，还可以携带该缓冲区标识等信息，使得任务调度器能够查找到对应的该缓冲区。

202：根据任务信息，确定本次待执行任务的实际执行时间。

其中，实际执行时间是指执行该待执行任务的执行时间，特别是针对处理设备执行该待执行任务的实际执行时间。

10 根据任务信息，确定本次待执行任务的实际执行时间，包括：确定执行任务信息的实际执行时间，作为本次待执行任务的实际执行时间。即，通过执行该任务信息得到的实际执行时间，作为执行该待执行任务的时间执行时间。

其中，确定实际执行时间的方式可以为：

1)、确定执行任务信息的实际执行时间，包括：将任务信息输入至预置模型中，确定预置模型的输出，作为实际执行时间。

其中，预置模型是用于预测任务信息的实际执行时间的模型。该模型可以是神经网络模型。其可以通过训练生成。在训练前，可以获取对应的训练数据，如多个任务信息（如，多个执行指令）以及对应处理设备的实际执行时间。在训练的过程中，通过该任务信息进行分析，获取其任务信息特征，然后根据对应的实际执行时间进行训练，生成该预置模型。

需要说明的是，由于处理设备可以不同种类，如 CPU、GPU 以及 NPU 等。所以根据不同的处理设备种类，可以生成多个对应的预置模型，用于分别针对一个处理设备进行实际执行时间的预测。

当模型生成后，就可以进行预测了。云服务器中的任务调度器可以将任务信息，输入至对应的预置模型中，如执行指令是针对 GPU 处理设备的，则将该执行指令输入至 GPU 对应的预置模型中，得到该预置模型的输出，作为 GPU 执行该执行指令的实际执行时间。即如图 3 所示，实现了步骤 302：确定处理设备的实际执行时间。

2)、根据任务信息，确定本次待执行任务的实际执行时间，包括：将任务信息与多个预置任务信息进行匹配，根据匹配结果，选择与匹配到的预置任务信息对应的预置执行时间作为实际执行时间。

此外，云服务器也可以获取到多个执行指令以及不同处理设备对应的实际执行时间，并将这些信息存储至本地预置存储区域，或者一个数据库服务器中，在云服务器中的任务调度器获取到任务信息后，如执行指令，将该执行指令与获取到的多个执行指令进行对比或匹配，确定相似度在一定阈值内的，或者相同的执行指令。并根据确定的执行指

令，获取对应的实际执行时间。

应理解，对于实际执行时间而言，不同处理设备可能是不同的，所以一个执行指令可以存储有多个实际执行时间，不同处理设备对应不同的实际执行时间，根据任务信息需要的处理设备，确定对应的实际执行时间给任务调度器。

- 5 203: 将实际执行时间与本次待执行任务对应的预置调度时间进行对比，根据对比结果，对下一次待执行任务对应的实际调度时间进行调节，以确定是否对下一次待执行任务进行调度。

其中，预置调度时间，也可以称为预置时间片，是指用于占用硬件资源的时间，通常是指一段时间，如 10ms（毫秒）。以处理器为例进行说明，如对于 CPU 而言，每个
10 CPU 可以为划分出多个时间片，并分配给不同的容器，使得容器的待处理任务就可以根据时间片，使得 CPU 对待处理任务进行处理，从而实现 CPU 在一段时间内处理多个任务的情况。此外，对于多核 CPU 而言，每个核心可以划分出来多个时间片，并分配给不同的容器，使得容器的待处理任务可以根据时间片，使得 CPU 核心对任务进行处理，实现 CPU 在一段时间内处理多个任务的情况。对于 GPU 以及 NPU 而言也是相似的，此处
15 就不再一一赘述了。其它硬件资源也可以以此为依据进行划分。故，云服务器可以对不同的虚拟设备，如容器，划分时间片。同一个容器的预置时间片是相同的，即预置调度时间是相同的。那么，来自同一个容器的不同待执行任务的预置时间片也是相同的。不同容器的时间片可以是不同的，可以是相同的。该预置调度时间可以由任务调度器预先划分好的。

- 20 与预置调度时间相对的还有实际调度时间。实际调度时间是指，在具有预置调度时间的前提下，对于处理设备而言，待执行任务实际可以具有多少时间可以让处理设备进行任务处理。

虽然为每个容器都设置了对应的预置调度时间，但是待执行任务在硬件资源，如处理设备，中的执行时间，有可能会超过预置调度时间，导致后面提交的待执行任务的预
25 置调度时间被挤占。例如，对于 GPU1 处理设备而言，任务调度器可以先调度容器 1 的任务 1 给 GPU1 进行任务处理，但由于 GPU1 处理该任务 1 的时间过长，超过了预置调度时间，使得后续任务的处理时间压缩，比如，此时，排在任务 1 后等待 GPU1 处理的任务时容器 2 的任务 2。此时，由于任务 2 的调度时间被占用，没有完全利用到属于自己的预置调度时间，进而会影响到容器 2 的调度质量（Quality of Service, QoS）。

- 30 故，这里设置了实际调度时间，用来量化出来每个容器中的各个任务到底可占用处理设备的时间，即各个任务到底还有多少时间可以调度。

例如，根据前文所述，云服务器中的任务调度器在接收到容器 A 发送的任务 a 后，并确定出该任务 a 是针对于 GPU1 的，同时确定出通过 GPU 进行处理任务 a 的实际执行时间。任务调度器将实际执行时间 15ms 与容器 A 对应的预置调度时间 10ms 进行对比，

实际执行时间大于预置调度时间，则减少容器 A 的下一次待执行任务的实际调度时间。即如图 3 所示，执行了步骤 303：减少下一次实际调度时间。以使得，利用容器 A 的下次待执行任务的调度时间，补偿给其它容器的待处理任务的调度时间。

此外，当任务调度器将实际执行时间 7ms 与容器 A 对应的预置调度时间 10ms 进行对比，实际执行时间小于预置调度时间，此时，可以使得对应的处理设备空闲，或者，补偿给容器 A 的下次待执行任务的实际调度时间。如图 3 所示，执行步骤 305：确定空闲时间或者补偿下一次实际调度时间。

需要说明的是，任务调度器将实际执行时间与容器 A 对应的预置调度时间相等时，则可以不进行调节。或者维持当前的已经被减少的调度时间（即实际调度时间）。

另，对于本次待执行任务而言，如果容器初始化阶段，即容器发送的本次待执行任务是该容器的第一次待执行任务，则实际调度时间就是预置调度时间，并且可以继续作为该容器的第二次待执行任务的实际调度时间，以进行后续的更新。此时，任务调度器可以在执行步骤 302 前，执行步骤 306：提交任务到处理设备，由处理设备进行任务处理。如任务调度器在接收到容器 A 的任务 a 后，获取到执行指令后，直接将任务 a 提交到对应的 GPU，进行任务处理。但是，如果容器已经在运行一段期间了，此时，任务调度器可以在接收到任务 a 后，可以先检测一下本次该任务 a 所属容器 A 的实际调度时间是否为 0，如果是，则不会进行任何处理，也不会执行步骤 306，需要等待下一次预置调度时间的到来，再去执行步骤 306 以及实际调度时间的调节的步骤。

对于本次待执行任务而言，在调节完其同一起来源的下次待执行任务对应的实际调度时间后，任务调度器可以继续对该下次待执行任务进行调度以及实际调度时间的调节。即执行步骤 304：进行下次任务调度。

步骤 203 的具体的实现方式如下：

其中，根据对比结果，对下次待执行任务对应的实际调度时间进行调节，包括：当比较结果为实际执行时间大于预置调度时间，确定超时时间；根据超时时间，对同一起来源的下次待执行任务对应的实际调度时间进行调节，得到更新后的实际调度时间，并继续作为同一起来源的后续待执行任务的实际调度时间，直至后续更新后的实际调度时间减少到为 0；在本次待执行任务为同一起来源的初始待执行任务时，本次待执行任务对应的实际调度时间与同一起来源的下次待执行任务对应的实际调度时间相同，为预置调度时间。

例如，根据前文所述，任务调度器接收到容器 A 发送的任务 a（即本次待执行任务），并获取到其执行指令，从而确定对应的实际执行时间为 15ms，与容器 A 的预置调度时间 10ms 进行对比，确定超时时间为 5ms，那么任务调度器下次再接收到容器 A 发送的任务时，如任务 b，那么其实际调度时间为 $10-5=5\text{ms}$ ，即由原来的 10ms（此时，可以将预置调度时间视为没有超时时间情况下的实际调度时间，如任务 a 是容器 A 的第一次任务，

或者容器 A 之前的任务都没有超时以及空闲，则任务 a 的实际调度实际可以为预置调度时间并且作为任务 b 的实际调度时间) 变为了 5ms，此时任务 b 的实际调度时间进行了更新为 5ms，并作为容器 A 下一次任务 c 的实际调度时间。但是，由于任务 a 对应的实际调度时间为预置调度时间，且大于 0，所以任务调度器在计算任务 b 的实际调度时间前，会将任务 a 发送至对应的处理设备，如 GPU1，GPU1 可以从任务 a 中的存储地址获取到执行指令，进行任务处理。由于任务 b 的实际调度时间更新为 5ms，但也大于 0，所以任务 b 也可以被任务调度器进行调度，由对应处理设备进行处理。若任务 b 的实际执行时间也为 15ms，则大于预置调度时间 10ms，超时时间也是 5ms。那么对于容器 A 的再下一次待执行任务，如任务 c 而言，其实际调度时间将由 5ms 更新为 0ms，即 $5-5=0$ 。此时任务调度器在接收到任务 c 后，当其预置调度时间来到时，不对该任务 c 进行调度，需要等待。

应理解，后续待执行任务可以是指本次待执行任务之后的任务，也可以是下一次待执行任务之后的任务。

需要说明的是，对实际调度时间的调节使得其它虚拟设备，如其它容器的调度时间不会被强行挤占，保证了调度的公平性，满足了服务质量 (QoS)。同时也避免了某些程序恶意的占用其它虚拟设备，如其它容器的时间片 (即预置调度时间)，保证了容器调度的安全性。

基于超时时间，对实际调度时间进行调节，具体过程如下：

具体的，该方法 200 还包括：针对本次待执行任务，将超时时间减少至本次待执行任务对应的实际调度时间，以对对应的实际调度时间进行更新，并将更新后的实际调度时间作为下一次待执行任务的实际调度时间。

由于前文已经阐述过了，此处就不再赘述。

上述等待的过程具体可以为：该方法 200 还包括：当后续更新后的实际调度时间为 0，则在后续待执行任务对应的预置调度时间达到后，确定不对该后续待执行任务进行调度。

例如，根据前文所述，任务调度器在接收到任务 c 后，其实际调度时间被更新为 0 了，则任务调度器等到任务 c 所属容器 A 对应的预置调度时间到达后，不再对该任务 c 进行调度，即不会通过设备驱动发送任务 c 给对应的处理设备，如 GPU1 进行任务处理。

但如果任务 c 的实际调度时间更新后不为 0，或者说后续任务的实际调度时间更新后不为 0，则可以进行调度：

具体的，该方法 200 还包括：当后续待执行任务对应的实际调度时间更新后不为 0，则在对应的预置调度时间达到后，确定根据实际执行时间，对该后续待执行任务进行调度。

例如，根据前文所述，任务调度器在接收到任务 b 后，由于实际调度时间更新为 5ms，

那么，任务调度器在接收到任务 b 后，在等到该容器 A 的预置调度时间到来后，将任务 b 通过云服务器的驱动设备发送至对应的处理设备，如 GPU1 进行任务处理。同时，GPU1 处理任务 b 的实际执行时间为上述预测的 15ms。

需要说明的是，由于任务 b 的实际执行时间为 15ms，那么任务调度器可以在 15ms 前，通过轮询的方式访问处理设备，确定其是否已经处理完任务 b，当确定处理完后，进行下一次任务的发送。正常的情况下，处理设备是可以在 15ms 时刻处理完该任务，或者，在针对 15ms 的一个误差范围内处理完该任务。

在本次待执行任务调度完成后，可以确定出下一次待执行任务是否进行调度，在确定进行调度后，进行任务调度。

具体的，该方法 200 还包括：在确定对下一次待执行任务进行调度后，接收下一次待执行任务，并等待对应的预置调度时间到达后，将下一次待执行任务发送至对应的处理设备，以根据对应的实际执行时间进行任务处理。

由于前文已经阐述过此处，就不再赘述。当后续待执行任务不进行调度，可以将其预置调度时间补偿给其它来源的任务，从而使得各个任务的调度时间能够得到平衡。

具体的，该方法 200 还包括：接收后续待执行任务，在对应的预置调度时间到达后，发送其它来源的待执行任务至对应的处理设备，以根据对应的实际执行时间进行任务处理。

例如，根据前文所述，任务调度器没有在对应的预置调度时间到达后，调度任务 c。但是可以调度其它任务，如排在任务 c 后面的其它容器的任务，如容器 B 的任务 m。

从而使得容器 A 的下次预置调度时间到达之前，将后续的任务进行调度，将容器 A 当前的调度时间分享给其它容器进行使用。

由于任务调度器对其它来源的待执行任务的调度与前文阐述的调度过程相同，此处就不再赘述。仅说明，无论哪个容器的任务，都可以利用本申请实施例的调度方式进行调度。其它容器的任务也有实际调度时间更新的情况，以及等待任务调度的情况，就不再展开详述。

此外，由于云服务器中存在多个容器，任务调度器可以接收到多个待执行任务，这些任务需要进行排队进行处理，在其所属的容器的预置调度时间到来后，按照这个排队顺序，依次进行任务处理。

当任务没有被任务调度器进行及时调度，而是确定为等待调度的情况下，在等到可以进行调度了，其具体的调度过程可以为：该方法 200 还包括：等待对应的下次预置调度时间到达后，发送后续待执行任务至对应的处理设备，以根据对应的实际执行时间进行任务处理；在对应的下次预置调度时间到达后，后续待执行任务对应的实际调度时间等于预置调度时间。

例如，根据前文所述，任务调度器未调度任务 c，而将任务 c 确定为等待状态后，等

待其所属容器 A 的下一预置调度时间到来

需要说明的是，在本申请实施例中，虽然容器中的任务可能会占用其它容器的任务，但是其它容器的调度时间会得到补偿，但是在占用到其它容器的调度时间后，其它容器的任务可能不能完全准时地被调到对应处理设备，即在其它容器的任务的预置调度时间到来后，不能马上进行调度，会有一些时间的延迟，但是还是要在对应预置调度时间到来，才能进行该任务的调度，只是要等到任务调度器轮询到对应处理设备已经完成了当前任务，即可调度该任务过去给对应处理设备。

除了实际执行时间大于预置调度时间外，还有可能是实际执行时间小于预置调度时间，在这种情况下，实际调度时间的调节过程如下：

- 10 具体的，根据对比结果，对下一次待执行任务对应的实际调度时间进行调节，包括：当比较结果为实际执行时间小于预置调度时间，确定空闲时间；根据空闲时间，对同一来源的下一预置调度时间对应的实际调度时间进行调节，得到更新后的实际调度时间，并继续作为同一来源的后续待执行任务的实际调度时间。

- 15 例如，根据前文所述，任务调度器接收到容器 A 发送的任务 a（即本次待执行任务），并获取到其执行指令，从而确定对应的实际执行时间为 7ms，与容器 A 的预置调度时间 10ms 进行对比，确定空闲时间为 3ms，那么任务调度器下一次再接收到容器 A 发送的任务时，如任务 b，那么其实际调度时间为 $10+3=13\text{ms}$ ，即由原来的 10ms（此时，可以将预置调度时间视为没有超时时间情况下的实际调度时间，如任务 a 是容器 A 的第一次任务，或者容器 A 之前的任务都没有超时以及空闲，则任务 a 的实际调度实际可以为预置调度时间并且作为任务 b 的实际调度时间）变为了 13ms，此时任务 b 的实际调度时间进行了更新为 13ms，并作为容器 A 下一次任务 c 的实际调度时间。

需要说明的是，除了可以给相同容器的任务其空闲时间外，还可以将该空闲时间直接空闲下来，使得对应处理设备能够空闲下来。此外，还可以将空闲下来的空闲时间分给其它容器的任务，以进行补偿或者帮助等等。

- 25 在确定具有空闲时间后，就可以对该任务进行调度：

具体的，该方法 200 还包括：当比较结果为实际执行时间小于或者等于预置调度时间，则确定对下一次待执行任务进行调度。

例如，根据前文所述，任务调度器在确定任务 a 具有空闲时间 3ms，那么，任务 b 的实际调度时间更新为 13ms，大于 0，则可以在接收到任务 b 后，对任务 b 进行调度。

- 30 应理解的是，在任务 a 的实际执行时间为 10ms，等于预置调度时间，那么任务 b 的实际调度时间不会被更新，还维持在 10ms，或者其它大于 0 的数值（如果是等于 0 的话，那么任务 a 是不能进行调度的，所以这里的前提是任务 a 是可以进行调度的，没有处于等待），任务 b 也可以被进行调度。

其中，基于空闲时间，具体调节实际调度时间的过程为：

具体的，该方法 200 还包括：针对本次待执行任务，将空闲时间增加至本次待执行任务对应的实际调度时间，以对对应的实际调度时间进行更新，并将更新后的实际调度时间作为下一次待执行任务的实际调度时间。

由于前文已经阐述过，此处就不再赘述。

- 5 在实际调度时间被更新为 0 后，由于对应的任务暂时不能进行调度，需要等待，直到下一次预置调度时间到达后，才可以进行调度。这时，该任务就没有再欠其它容器的任务的调度时间了，所以其实际调度时间也可以被重置，为预置调度时间。具体过程如下：

10 具体的，该方法 200 还包括：当更新后的实际调度时间为 0，则在确定下一次待执行任务可进行调度的情况下，下一次待执行任务对应的实际调度时间等于预置调度时间。

 例如，根据前文所述，任务 c 的实际调度时间被更新为 0 后，等待下一次容器 A 的预置调度时间到达，到达后，任务 c 可以被调度了。则其实际调度时间也可以被重置为 10ms=预置调度时间，并作为容器 A 的下一次待执行任务的实际调度时间，如任务 d，以等后续任务调度器接收到该任务 d 时，在对该实际调度时间 10ms 进行更新，具体的更新过程参照前文。

15 需要说明的是，在本申请实施例中的出现的后续任务，如任务 b、c、d 等仅仅是为了能够清楚说明进行的举例，并不是说将这些任务进行了标识，只有符合这个标识才能执行本方案，这些标识只是表明是同一容器的后续任务而言。

20 此外，由于在实际调度时间进行调节的过程中，可能不能完全直接更新为 0，有可能从 5ms 更新到了-1ms，这时，调节的过程如下：

 具体的，该方法 200 还包括：当同一起来源的后续待执行任务对应的实际调度时间直接从正数减少到为负数，确定该负数与 0 的差距数值；将同一起来源的后续待执行任务对应的实际调度时间调整到为 0；接收同一起来源的后续待执行任务，在对应的预置调度时间到达后，发送其它来源的待执行任务至对应的处理设备；等待对应的下一次预置调度时间到达后，发送同一起来源的后续待执行任务至对应的处理设备，以根据对应的实际执行时间进行任务处理；在对应的下一次预置调度时间到达后，同一起来源的后续待执行任务对应的实际调度时间等于预置调度时间与差距数值之差。

30 例如，根据前文所述，任务调度器接收到容器 A 发送的任务 a（即本次待执行任务），并获取到其执行指令，从而确定对应的实际执行时间为 16ms，与容器 A 的预置调度时间 10ms 进行对比，确定超时时间为 6ms，那么任务调度器下一次再接收到容器 A 发送的任务时，如任务 b，那么其实际调度时间为 10-6=4ms，即由原来的 10ms（此时，可以将预置调度时间视为没有超时时间情况下的实际调度时间，如任务 a 是容器 A 的第一次任务，或者容器 A 之前的任务都没有超时以及空闲，则任务 a 的实际调度实际可以为预置调度时间并且作为任务 b 的实际调度时间）变为了 4ms，此时任务 b 的实际调度时间进行了

更新为 4ms, 并作为容器 A 下一次任务 c 的实际调度时间。任务调度器在接收到任务 b 后, 确定其实际执行时间为 16ms, 超时时间为 6ms, 那么更新其实际调度时间为 $4-6=-2\text{ms}$ 。此时任务调度器就可以直接把实际调度时间更新为 0ms, 并作为任务 c 的实际调度时间, 同时确定 2ms 作为差距数值。任务调度器可以调度任务 a 和任务 b, 但暂时不能调度任务 c, 需要等待。在对应的预置调度时间到达后, 暂时不调度任务 c, 可以调度后续其它容器的任务。等到下一次预置调度时间到达后, 再对任务 c 进行调度。由于此时, 任务 c 的实际调度时间为 0 但是在重新进行调度的情况下, 其实际调度时间可以被重置为 $10\text{ms}=\text{预置调度时间}$, 所以该容器 A 的下一任务 d 的实际调度时间可以更新为 $10\text{ms}-2=8\text{ms}$ 。

需要说明的是, 本申请实施例的任务调度与设备驱动是否开源无关, 可以容易的应用在开源或是闭源的设备驱动的处理设备上, 应用范围更为广泛。

基于上述相同的发明构思, 图 4 示出了本申请另一示例性实施例提供的一种容器任务的调度方法的流程示意图。本申请实施例提供的该方法 400 由上述云服务器执行, 更具体的可以是设置在云服务器的任务调度器, 如图 4 所示, 该方法 400 包括以下步骤:

401: 接收任一容器发送的本次待执行任务, 获取本次待执行任务的任务信息, 任务信息描述了本次待执行任务。

402: 根据任务信息, 确定本次待执行任务的实际执行时间。

403: 将实际执行时间与本次待执行任务对应的预置调度时间进行对比, 根据对比结果, 对同一容器的下一次待执行任务对应的实际调度时间进行调节, 以确定是否对同一容器的下一次待执行任务进行调度。

由于前文已经详细阐述过步骤 401-403 的具体实施方式, 此处就不再赘述。

其中, 接收任一容器发送的本次待执行任务, 包括: 根据与容器对应的通信接口接收本次待执行任务; 该方法 400 还包括: 根据通信接口确定容器标识, 以使得根据容器标识, 确定同一容器的待执行任务。

由于前文已经阐述过此处, 就不再赘述。

另, 本方法 400 未能详细描述的内容, 还可以参照上述方法 200 中的各个步骤。

图 5 为本申请一示例性实施例提供的一种任务的调度装置的结构框架示意图。该装置 500 可以应用于云服务器, 更具体的可以是云服务器中的任务调度器。该装置 500 包括获取模块 501、确定模块 502 以及调节模块 503; 以下针对各个模块的功能进行详细的阐述:

获取模块 501, 用于获取本次待执行任务的任务信息, 任务信息描述了本次待执行任务。

确定模块 502, 用于根据任务信息, 确定本次待执行任务的实际执行时间。

调节模块 503, 用于将实际执行时间与本次待执行任务对应的预置调度时间进行对

比, 根据对比结果, 对下一次待执行任务对应的实际调度时间进行调节, 以确定是否对下一次待执行任务进行调度。

具体的, 获取模块 501, 用于根据本次待执行任务携带的存储地址, 获取任务信息。

具体的, 确定模块 502, 用于确定执行任务信息的实际执行时间, 作为本次待执行任务的实际执行时间。

具体的, 确定模块 502, 用于将任务信息输入至预置模型中, 确定预置模型的输出, 作为实际执行时间。

具体的, 确定模块 502, 用于将任务信息与多个预置任务信息进行匹配, 根据匹配结果, 选择与匹配到的预置任务信息对应的预置执行时间作为实际执行时间。

具体的, 调节模块 503, 包括: 确定单元, 用于当比较结果为所述实际执行时间大于预置调度时间, 确定超时时间; 调节单元, 用于根据超时时间, 对同一来源的下一任务待执行任务对应的实际调度时间进行调节, 得到更新后的实际调度时间, 并继续作为同一来源的后续待执行任务的实际调度时间, 直至后续更新后的实际调度时间减少到为 0; 在本次待执行任务为同一来源的初始待执行任务时, 本次待执行任务对应的实际调度时间与同一来源的下一任务待执行任务对应的实际调度时间相同, 为预置调度时间。

此外, 该装置 500 还包括: 调度模块, 用于当后续更新后的实际调度时间为 0, 则在后续待执行任务对应的预置调度时间达到后, 确定不对该后续待执行任务进行调度。

此外, 调度模块, 还用于: 当后续待执行任务对应的实际调度时间更新后不为 0, 则在对应的预置调度时间达到后, 确定根据实际执行时间, 对该后续待执行任务进行调度。

此外, 该装置 500 还包括: 发送模块, 用于在确定对下一次待执行任务进行调度后, 接收下一次待执行任务, 并等待对应的预置调度时间到达后, 将下一次待执行任务发送至对应的处理设备, 以根据对应的实际执行时间进行任务处理。

此外, 发送模块, 还用于接收后续待执行任务, 在对应的预置调度时间到达后, 发送其它来源的待执行任务至对应的处理设备, 以根据对应的实际执行时间进行任务处理。

此外, 发送模块, 还用于等待对应的下一次预置调度时间到达后, 发送后续待执行任务至对应的处理设备, 以根据对应的实际执行时间进行任务处理; 在对应的下一次预置调度时间到达后, 后续待执行任务对应的实际调度时间等于预置调度时间。

此外, 确定单元, 还用于: 当比较结果为实际执行时间小于预置调度时间, 确定空闲时间; 调节单元, 还用于: 根据空闲时间, 对同一来源的下一任务待执行任务对应的实际调度时间进行调节, 得到更新后的实际调度时间, 并继续作为同一来源的后续待执行任务的实际调度时间。

此外, 确定单元, 还用于: 当比较结果为实际执行时间小于或者等于预置调度时间, 则确定对下一次待执行任务进行调度。

此外，调节单元，还用于：针对本次待执行任务，将空闲时间增加至本次待执行任务对应的实际调度时间，以对对应的实际调度时间进行更新，并将更新后的实际调度时间作为下一次待执行任务的实际调度时间。

此外，调节单元，还用于：针对本次待执行任务，将超时时间减少至本次待执行任务对应的实际调度时间，以对对应的实际调度时间进行更新，并将更新后的实际调度时间作为下一次待执行任务的实际调度时间。

此外，调节单元，还用于：当更新后的实际调度时间为 0，则在确定下一次待执行任务可进行调度的情况下，下一次待执行任务对应的实际调度时间等于预置调度时间。

其中，同一起来源的待执行任务对应的预置调度时间相同。

此外，确定模块 502，还用于：当同一起来源的后续待执行任务对应的实际调度时间直接从正数减少到为负数，确定该负数与 0 的差距数值；调节模块 503，还用于：将同一起来源的后续待执行任务对应的实际调度时间调整到为 0；发送模块，还用于：接收同一起来源的后续待执行任务，在对应的预置调度时间到达后，发送其它来源的待执行任务至对应的处理设备；发送模块，还用于：等待对应的下一次预置调度时间到达后，发送同一起来源的后续待执行任务至对应的处理设备，以根据对应的实际执行时间进行任务处理；在对应的下一次预置调度时间到达后，同一起来源的后续待执行任务对应的实际调度时间等于预置调度时间与差距数值之差。

具体的，获取模块 501，用于通过通信接口，接收虚拟设备发送的本次待执行任务。

图 6 为本申请一示例性实施例提供的一种容器任务的调度装置的结构框架示意图。

该装置 600 可以应用于云服务器，更具体的可以是设置在云服务器中的任务调度器。该装置 600 包括接收模块 601、确定模块 602 以及调节模块 603；以下针对各个模块的功能进行详细的阐述：

接收模块 601，用于接收任一容器发送的本次待执行任务，获取本次待执行任务的任务信息，任务信息描述了本次待执行任务。

确定模块 602，用于根据任务信息，确定本次待执行任务的实际执行时间。

调节模块 603，用于将实际执行时间与本次待执行任务对应的预置调度时间进行对比，根据对比结果，对同一容器的下一次待执行任务对应的实际调度时间进行调节，以确定是否对同一容器的下一次待执行任务进行调度。

具体的，接收模块 601，具体用于根据与容器对应的通信接口接收本次待执行任务；确定模块 602，用于根据通信接口确定容器标识，以使得根据容器标识，确定同一容器的待执行任务。

需要说明的是，对于装置 600 未能提及的部分内容，可以参照上述装置 500 的内容。

以上描述了图 5 所示的装置 500 的内部功能和结构，在一个可能的设计中，图 5 所示的装置 500 的结构可实现为计算设备，如，服务器。如图 7 所示，该设备 700 可以包

括：存储器 701 以及处理器 702；

存储器 701，用于存储计算机程序。

处理器 702，用于执行计算机程序，以用于：获取本次待执行任务的任务信息，任务信息描述了本次待执行任务。根据任务信息，确定本次待执行任务的实际执行时间。

- 5 将实际执行时间与本次待执行任务对应的预置调度时间进行对比，根据对比结果，对下一次待执行任务对应的实际调度时间进行调节，以确定是否对下一次待执行任务进行调度。

具体的，处理器 702，具体用于：根据本次待执行任务携带的存储地址，获取任务信息。

- 10 具体的，处理器 702，具体用于：确定执行任务信息的实际执行时间，作为本次待执行任务的实际执行时间。

具体的，处理器 702，具体用于：将任务信息输入至预置模型中，确定预置模型的输出，作为实际执行时间。

- 15 具体的，处理器 702，具体用于：将任务信息与多个预置任务信息进行匹配，根据匹配结果，选择对应的预置执行时间作为实际执行时间。

- 具体的，处理器 702，具体用于：当比较结果为所述实际执行时间大于预置调度时间，确定超时时间；根据超时时间，对同一来源的下一任务待执行任务对应的实际调度时间进行调节，得到更新后的实际调度时间，并继续作为同一来源的后续待执行任务的实际调度时间，直至后续更新后的实际调度时间减少到为 0；在本次待执行任务为同一来源的初始待执行任务时，本次待执行任务对应的实际调度时间与同一来源的下一任务待执行任务对应的实际调度时间相同，为预置调度时间。
- 20

此外，处理器 702，还用于：当后续更新后的实际调度时间为 0，则在后续待执行任务对应的预置调度时间达到后，确定不对该后续待执行任务进行调度。

- 此外，处理器 702，还用于：当后续待执行任务对应的实际调度时间更新后不为 0，则在对应的预置调度时间达到后，确定根据实际执行时间，对该后续待执行任务进行调度。
- 25

此外，处理器 702，还用于：在确定对下一任务待执行任务进行调度后，接收下一任务待执行任务，并等待对应的预置调度时间到达后，将下一任务待执行任务发送至对应的处理设备，以根据对应的实际执行时间进行任务处理。

- 30 此外，处理器 702，还用于：接收后续待执行任务，在对应的预置调度时间到达后，发送其它来源的待执行任务至对应的处理设备，以根据对应的实际执行时间进行任务处理。

此外，处理器 702，还用于：等待对应的下一任务预置调度时间到达后，发送后续待执行任务至对应的处理设备，以根据对应的实际执行时间进行任务处理；在对应的下一

次预置调度时间到达后，后续待执行任务对应的实际调度时间等于预置调度时间。

此外，处理器 702，还用于：当比较结果为实际执行时间小于预置调度时间，确定空闲时间；根据空闲时间，对同一来源的下一任务待执行任务对应的实际调度时间进行调节，得到更新后的实际调度时间，并继续作为同一来源的后续待执行任务的实际调度时间。

此外，处理器 702，还用于：当比较结果为实际执行时间小于或者等于预置调度时间，则确定对下一任务待执行任务进行调度。

此外，处理器 702，还用于：针对本次待执行任务，将空闲时间增加至本次待执行任务对应的实际调度时间，以对对应的实际调度时间进行更新，并将更新后的实际调度时间作为下一任务待执行任务的实际调度时间。

此外，处理器 702，还用于：针对本次待执行任务，将超时时间减少至本次待执行任务对应的实际调度时间，以对对应的实际调度时间进行更新，并将更新后的实际调度时间作为下一任务待执行任务的实际调度时间。

此外，处理器 702，还用于：当更新后的实际调度时间为 0，则在确定下一任务待执行任务可进行调度的情况下，下一任务待执行任务对应的实际调度时间等于预置调度时间。

其中，同一来源的任务待执行任务对应的预置调度时间相同。

此外，处理器 702，还用于：当同一来源的后续待执行任务对应的实际调度时间直接从正数减少到为负数，确定该负数与 0 的差距数值；将同一来源的后续待执行任务对应的实际调度时间调整到为 0；接收同一来源的后续待执行任务，在对应的预置调度时间到达后，发送其它来源的任务待执行任务至对应的处理设备；发送模块，还用于：等待对应的下一任务预置调度时间到达后，发送同一来源的后续待执行任务至对应的处理设备，以根据对应的实际执行时间进行任务处理；在对应的下一任务预置调度时间到达后，同一来源的后续待执行任务对应的实际调度时间等于预置调度时间与差距数值之差。

具体的，处理器 702，具体用于：通过通信接口，接收虚拟设备发送的本次待执行任务。

另外，本发明实施例提供了一种计算机存储介质，计算机程序被一个或多个处理器执行时，致使一个或多个处理器实现图 2 方法实施例中一种任务的调度方法的步骤。

以上描述了图 6 所示的装置 600 的内部功能和结构，在一个可能的设计中，图 6 所示的装置 600 的结构可实现为计算设备，如，服务器。如图 8 所示，该设备 800 可以包括：存储器 801 以及处理器 802；

存储器 801，用于存储计算机程序；

处理器 802，用于执行计算机程序，以用于：接收任一容器发送的本次待执行任务，获取本次待执行任务的任务信息，任务信息描述了本次待执行任务。根据任务信息，确定本次待执行任务的实际执行时间。将实际执行时间与本次待执行任务对应的预置调度

时间进行对比，根据对比结果，对同一容器的下一次待执行任务对应的实际调度时间进行调节，以确定是否对同一容器的下一次待执行任务进行调度。

具体的，处理器 802，具体用于：根据与容器对应的通信接口接收本次待执行任务；处理器 802，还用于：根据通信接口确定容器标识，以使得根据容器标识，确定同一容器的待执行任务。

需要说明的是，对于该设备 800 未能提及的部分内容，可以参照上述设备 700 的内容。

另外，本发明实施例提供了一种计算机存储介质，计算机程序被一个或多个处理器执行时，致使一个或多个处理器实现图 4 方法实施例中一种容器任务的调度方法的步骤。

另外，在上述实施例及附图中的描述的一些流程中，包含了按照特定顺序出现的多个操作，但是应该清楚了解，这些操作可以不按照其在本文中出现的顺序来执行或并行执行，操作的序号如 201、202、203 等，仅仅是用于区分开各个不同的操作，序号本身不代表任何的执行顺序。另外，这些流程可以包括更多或更少的操作，并且这些操作可以按顺序执行或并行执行。需要说明的是，本文中的“第一”、“第二”等描述，是用于区分不同的消息、设备、模块等，不代表先后顺序，也不限定“第一”和“第二”是不同的类型。

以上所描述的装置实施例仅仅是示意性的，其中所述作为分离部件说明的单元可以是或者也可以不是物理上分开的，作为单元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实施例方案的目的。本领域普通技术人员在不付出创造性的劳动的情况下，即可以理解并实施。

通过以上的实施方式的描述，本领域的技术人员可以清楚地了解到各实施方式可借助加必需的通用硬件平台的方式来实现，当然也可以通过硬件和软件结合的方式来实现。基于这样的理解，上述技术方案本质上或者说对现有技术做出贡献的部分可以以计算机产品的形式体现出来，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、CD-ROM、光学存储器等）上实施的计算机程序产品的形式。

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多媒体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多媒体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。

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多媒体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。

- 5 这些计算机程序指令也可装载到计算机或其他可编程多媒体数据处理设备上，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。

- 10 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。

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。

- 15 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，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。

- 20 最后应说明的是：以上实施例仅用以说明本发明的技术方案，而非对其限制；尽管参照前述实施例对本发明进行了详细的说明，本领域的普通技术人员应当理解：其依然可以对前述各实施例所记载的技术方案进行修改，或者对其中部分技术特征进行等同替换；而这些修改或者替换，并不使相应技术方案的本质脱离本发明各实施例技术方案的精神和范围。

30