(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113015216 B
(45)授权公告日 2022.05.10

(21)申请号 202110174619.4

(22)申请日 2021.02.05

(65)同一申请的已公布的文献号
申请公布号 CN 113015216 A

(43)申请公布日 2021.06.22

(73)专利权人 浙江大学
地址 310013 浙江省杭州市西湖区余杭塘路866号
专利权人 杭州筑家易网络科技股份有限公司

(72)发明人 邓水光 张城 杨斌 尹建伟

(74)专利代理机构 杭州天勤知识产权代理有限公司 33224
专利代理师 王琛 胡红娟

(51)Int.Cl.
H04W 28/08(2009.01)
H04W 28/10(2009.01)
H04L 47/125(2022.01)
H04L 67/1008(2022.01)

(56)对比文件
CN 111756812 A,2020.10.09
CN 111953759 A,2020.11.17
CN 112039965 A,2020.12.04
CN 112004239 A,2020.11.27
CN 109951821 A,2019.06.28
CN 110662238 A,2020.01.07
CN 110557732 A,2019.12.10
EP 3734452 A1,2020.11.04
CN 112214261 A,2021.01.12
谢人超等.移动边缘计算卸载技术综述.《通信学报》.2018,(第11期),

审查员 夏凯茜

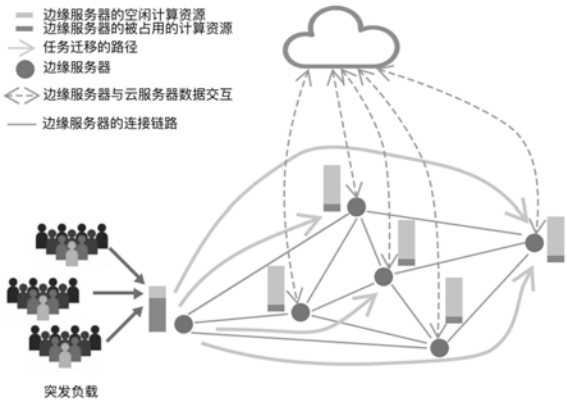
权利要求书2页 说明书7页 附图1页

(54)发明名称

一种面向边缘服务网络的突发任务卸载与调度方法

(57)摘要

本发明公开了一种面向边缘服务网络的突发任务卸载与调度方法,针对某一台拥塞服务器的突发任务,通过负载任务疏散和负载任务调度的策略同时执行,以达到任务突发负载疏散的优化过程。本发明方法可以弥补现有通信协议的不足,在不改变网络架构的基础上,使得边缘网络环境具备疏散任务和调度任务计算的能力。而且,本发明结合了边缘计算经典的网络结构即边缘服务器和云服务器两者各自的作用,进行任务计算的高效调度,使得突发的用户任务能够在较短的时间内完成,从而也保障了整个系统的服务质量。



CN 113015216 B

1.一种面向边缘服务网络的突发任务卸载与调度方法,包括如下步骤:

(1)检测边缘服务网络中存在任务过载的边缘服务器,对于任一过载的边缘服务器,确定该服务器需要疏散的任务;

(2)监测并记录边缘服务网络的链路和路由状态以及各边缘服务器的有效资源状况;

(3)枚举边缘服务网络中过载服务器到其他边缘服务器之间的所有无环疏散路径组成疏散路径集合,其中每一条疏散路径即对应了一组链路集合;进而按单个任务在疏散路径上的传输时间进行升序排列,并记录集合中各疏散路径的序号;

(4)以疏散完成时间最短为目标,为待疏散的任务分配疏散路径,具体实现过程如下:

首先,初始化m个解,每个解包含了N条疏散路径即表示从疏散路径集合中为每一个待疏散任务随机分配一条疏散路径,同时计算每个解所对应的疏散完成时间,m为自定义的初始化种群规模;

然后,从初始化m个解中取疏散完成时间最小的解记作x_t,其对应的疏散完成时间为y_t,x_t为一组N维向量且向量中的每一元素值对应为各待疏散任务所分配的疏散路径序号;

最后,自定义一个大小为W的窗口,利用该窗口对x_t进行循环扫描,记录每一窗口扫描所得到的局部最优解,比较所有的局部最优解,取疏散完成时间最小的局部最优解作为全局最优解并依此为待疏散的任务分配疏散路径;利用窗口对x_t进行循环扫描的具体过程如下:

4.1首先定义任务序号集合I={1,2…,N},并将I赋予集合I_v,同时建立两个空集I_f和V_t,设定参数γ以及阈值α∈0～1,然后从x_t的第一个元素值开始利用窗口截取W个元素值;

4.2对于窗口中任一元素值p,任取0～1之间的随机数λ,比较λ是否小于α:若是,则更新该元素值$\tilde{p} = \lfloor p/\gamma \rfloor$,同时将集合$I_v$中该元素值对应的任务序号转移至集合$I_f$中;若否,则从1～L中任取一个序号赋予更新该元素值$\tilde{p}$,L为疏散路径集合中路径数量;

完成上述操作后即x_t得到了更新,其对应更新后的解记作$\tilde{x}_t$,则$\tilde{x}_t$对应的疏散完成时间记作$\tilde{y}_t$,若$\tilde{y}_t < y_t$,则保持当前集合I_v和I_f不变,否则从集合I中随机任取相应数量的任务序号替换掉集合I_f中的所有任务序号,集合I中剩余的任务序号组成新的I_v;

4.3根据集合I_f中的任务序号,从$\tilde{x}_t$中找到这些任务序号所对应的元素值固定保持不变,对于其他元素值则从1～L中随机抽取并重复执行m次,从而得到m个新的解并纳入集合V_t中;

将x_t、$\tilde{x}_t$与V_t并集,取其中疏散完成时间最小的解作为局部最优解并保存,从而完成一次窗口扫描,进而平移窗口并返回执行步骤4.2;

(5)以边缘服务器与云服务器分布式并行协同处理的方式对边缘服务器本地任务执行时序进行在线调度,具体实现方式如下:

对于任一本地任务,边缘服务器会对其先进行预处理,然后将转交由云服务器进行协助处理,最后再返回给边缘服务器完成最终处理;

对于任一边缘服务器,其先执行完前n个本地任务的预处理工作后停止,直至云服务器完成所有边缘服务器交由其协助处理的n_1个任务后,边缘服务器才会继续执行完成n_2个任务的最终处理工作,n、n_1、n_2均为预设定的自然数且$n > n_1 > n_2$,所述的n_1个任务为n个本地任

务中的前n_1个任务，所述的n_2个任务为n_1个任务中的前n_2个任务；

定义四类时间节点：T1为边缘服务器执行完一个本地任务预处理工作的时刻，T2为边缘服务器执行完一个本地任务最终处理工作的时刻，T3为云服务器处理网络中第1号边缘服务器每个任务的开始时刻，T4为云服务器处理其他边缘服务器每个任务的结束时刻；

当T3触发，边缘服务器可从缓存中开始执行新的本地任务；

当T4触发，云服务器从缓存中立即开始执行新的任务；

当T1或T2触发，若云服务器处理完所有边缘服务器上传的第j号任务，各边缘服务器才能执行第$j+n-n_1$号本地任务的预处理工作或者第$j-n_1+n_2$号本地任务的最终处理工作，j为大于n_1的自然数。

2.根据权利要求1所述的突发任务卸载与调度方法，其特征在于：对于边缘服务网络中的任一条链路l_k，其任务传输的并发量上限$C_k^{\max} \geq \sum_{i=1}^{N} O_i^k(t)$，若待疏散的第i个任务在t时刻占用了链路$l_k$，则$O_i^k(t) = 1$，否则$O_i^k(t) = 0$，N为待疏散的任务数量。

3.根据权利要求1所述的突发任务卸载与调度方法，其特征在于：所述疏散完成时间的起始时刻为过载服务器做出疏散决策的时刻，终止时刻为最后一个任务完成疏散的时刻。

一种面向边缘服务网络的突发任务卸载与调度方法

技术领域

[0001] 本发明属于边缘计算网络负载调度技术领域,具体涉及一种面向边缘服务网络的突发任务卸载与调度方法。

背景技术

[0002] 在5G的普及下,借助于边缘计算,大量的智能设备在用户端就直接进行任务的卸载处理,边缘计算是为应用者和服务提供商在靠近用户边缘侧提供云服务器和各种网络应用环境的技术,其目标是在靠近用户数据产生的物理位置侧就提供给用户计算,存储和网络宽带的服务。边缘计算将算力了从远端的云服务器下放到了用户边缘侧,这种方式有效地增强了用户侧设备的处理任务的能力,缓解了由于用户侧设备的计算资源有限的瓶颈,并且因此诞生出许多独特的边缘卸载特点的应用场景。

[0003] 但是,通过边缘计算系统与远端云服务器系统一调度管理,一旦边缘网络中的某一台服务器出现任务拥塞现象,而不能及时处理这些任务时,将任务高效地迁移到别的空闲服务器上去执行,并且保证用户的QoS响应等需求变得越来越迫切和重要。边缘系统和远端服务器系统同事调度具有独特的系统特点,当前针对任务卸载和任务迁移已经提出了很多方法,但是很少考虑到边缘网络本身的特殊拓扑结构。当任务从一台服务器迁移到别的服务器上的同时,网络中的任务在不间断的执行着,两者同时进行,大多数算法考虑了静态的疏散,而没有考虑到任务的动态的疏散和执行。

[0004] 公开号为CN110536358A的中国专利提出了一种面向无依赖子任务的边缘计算负载均衡方法,在保证了能耗均衡的同时,使得用户产生的任务都能最优被执行,即选择本地计算或者卸载到边缘服务器上计算,但是这种方案没有考虑到边缘计算中的路径选择和远端服务器的协同处理。

[0005] 公开号为CN111597025A的中国专利提出了一种边缘计算调度算法及系统,其考虑了用队列来缓存用户的任务,并且用到了任务的优先级来调度任务的在不同边缘服务器上最优选择从而实现任务的执行目的;这种方法虽然实现了任务在不同的边缘服务器间的调度,但是局限于任务的优先级调度,从而忽略了任务在边缘服务器间迁移传输的开销以及远端云服务器的协助执行的调度,一旦发生了任务的拥塞或者用户流量的突发,任务在边缘服务器网络间的传输和云服务器间的调度将变得尤为重要。

[0006] 一方面,在传统的5G通信的架构下已经有许多研究表明,无线网络环境中比如(C-RAN)为基础设施架构的环境存在着一个具有挑战性的问题,即突发网络流量负载的疏散和迁移的问题。边缘计算网络往往都是构建在这种通信架构的基础上的,边缘计算必然要面个挑战,但是在现有的边缘服务器网络架构中,服务器之间的流量往往是通过http、mqtt、amqp等协议实现任务的通信,而这些协议没有提出有效的方法去解决突发流量的疏散与任务计算的调度。另一方面,随着IoT与各种移动设备的兴起,可以预计到边缘网络环境下,接入网络的设备数量越来越多,各种多样的应用服务随之产生,这也使得在边缘计算面临突发负载的重要挑战。

发明内容

[0007] 鉴于上述,本发明提供了一种面向边缘服务网络的突发任务卸载与调度方法,针对某一台拥塞服务器的突发任务,通过负载任务疏散和负载任务调度的策略同时执行,以达到任务突发负载疏散的优化过程。

[0008] 一种面向边缘服务网络的突发任务卸载与调度方法,包括如下步骤:

[0009] (1)检测边缘服务网络中存在任务过载的边缘服务器,对于任一过载的边缘服务器,确定该服务器需要疏散的任务;

[0010] (2)监测并记录边缘服务网络的链路和路由状态以及各边缘服务器的有效资源状况;

[0011] (3)枚举边缘服务网络中过载服务器到其他边缘服务器之间的所有无环疏散路径组成疏散路径集合,其中每一条疏散路径即对应了一组链路集合;进而按单个任务在疏散路径上的传输时间进行升序排列,并记录集合中各疏散路径的序号;

[0012] (4)以疏散完成时间最短为目标,为待疏散的任务分配疏散路径;

[0013] (5)以边缘服务器与云服务器分布式并行协同处理的方式对边缘服务器本地任务执行时序进行在线调度。

[0014] 进一步地,对于边缘服务网络中的任一条链路l_k,其任务传输的并发量上限$C_k^{\max} \geq \sum_{i=1}^{N} O_i^k(t)$,若待疏散的第i个任务在t时刻占用了链路$l_k$,则$O_i^k(t)=1$,否则$O_i^k(t)=0$,N为待疏散的任务数量。

[0015] 进一步地,所述疏散完成时间的起始时刻为过载服务器做出疏散决策的时刻,终止时刻为最后一个任务完成疏散的时刻。

[0016] 进一步地,所述步骤(4)的具体实现过程如下:

[0017] 首先,初始化m个解,每个解包含了N条疏散路径即表示从疏散路径集合中为每一个待疏散任务随机分配一条疏散路径,同时计算每个解所对应的疏散完成时间,m为自定义的初始化种群规模;

[0018] 然后,从初始化m个解中取疏散完成时间最小的解记作x_t,其对应的疏散完成时间为y_t,x_t为一组N维向量且向量中的每一元素值对应为各待疏散任务所分配的疏散路径序号;

[0019] 最后,自定义一个大小为W的窗口,利用该窗口对x_t进行循环扫描,记录每一窗口扫描所得到的局部最优解,比较所有的局部最优解,取疏散完成时间最小的局部最优解作为全局最优解并依此为待疏散的任务分配疏散路径。

[0020] 进一步地,利用窗口对x_t进行循环扫描的具体过程如下:

[0021] 4.1首先定义任务序号集合I={1,2…,N},并将I赋予集合I_v,同时建立两个空集I_f和V_t,设定参数γ以及阈值α∈0～1,然后从x_t的第一个元素值开始利用窗口截取W个元素值;

[0022] 4.2对于窗口中任一元素值p,任取0～1之间的随机数λ,比较λ是否小于α:若是,则更新该元素值$\tilde{p}=\lceil p/\gamma \rceil$,同时将集合$I_v$中该元素值对应的任务序号转移至集合$I_f$中;若否,则从1～L中任取一个序号赋予更新该元素值$\tilde{p}$,L为疏散路径集合中路径数量;

[0023] 完成上述操作后即x_t得到了更新，其对应更新后的解记作$\tilde{\mathbf{x}}_t$，则$\tilde{\mathbf{x}}_t$对应的疏散完成时间记作$\tilde{\mathbf{y}}_t$，若$\tilde{\mathbf{y}}_t < y_t$，则保持当前集合$I_v$和$I_f$不变，否则从集合I中随机任取相应数量的任务序号替换掉集合I_f中的所有任务序号，集合I中剩余的任务序号组成新的I_v；

[0024] 4.3根据集合I_f中的任务序号，从$\tilde{\mathbf{x}}_t$中找到这些任务序号所对应的元素值固定保持不变，对于其他元素值则从1～L中随机抽取并重复执行m次，从而得到m个新的解并纳入集合V_t中；

[0025] 将x_t、$\tilde{\mathbf{x}}_t$与V_t并集，取其中疏散完成时间最小的解作为局部最优解并保存，从而完成一次窗口扫描，进而平移窗口并返回执行步骤4.2。

[0026] 进一步地，所述步骤(5)的具体实现方式如下：

[0027] 对于任一本地任务，边缘服务器会对其先进行预处理，然后将转交由云服务器进行协助处理，最后再返回给边缘服务器完成最终处理；

[0028] 对于任一边缘服务器，其先执行完前n个本地任务的预处理工作后停止，直至云服务器完成所有边缘服务器交由其协助处理的n_1个任务后，边缘服务器才会继续执行完成n_2个任务的最终处理工作，n、n_1、n_2均为预设定的自然数且$n > n_1 > n_2$，所述的n_1个任务为n个本地任务中的前n_1个任务，所述的n_2个任务为n_1个任务中的前n_2个任务；

[0029] 定义四类时间节点：T1为边缘服务器执行完一个本地任务预处理工作的时刻，T2为边缘服务器执行完一个本地任务最终处理工作的时刻，T3为云服务器处理网络中第1号边缘服务器每个任务的开始时刻，T4为云服务器处理其他边缘服务器每个任务的结束时刻；

[0030] 当T3触发，边缘服务器可从缓存中开始执行新的本地任务；

[0031] 当T4触发，云服务器从缓存中立即开始执行新的任务；

[0032] 当T1或T2触发，若云服务器处理完所有边缘服务器上传的第j号任务，各边缘服务器才能执行第$j+n-n_1$号本地任务的预处理工作或者第$j-n_1+n_2$号本地任务的最终处理工作，j为大于n_1的自然数。

[0033] 本发明针对边缘环境下的IoT设备各种移动设备、车载设备等智能设备，根据边缘网络中边缘服务器的资源情况和网络情况，对该区域网络内的边缘服务器产生的突发任务负载进行疏散路径的规划，从而将突发负载的任务分散到边缘网络中空闲的服务器上进行执行。每一个智能网关根据任务的数据依赖，负责向云端数据存储服务器发送请求，获取边缘服务器所需执行任务的输入数据，以智能网关包含的所有边缘服务器最短时间内执行完所有任务为目标，提出了一种面向边缘服务网络的突发任务卸载疏散和任务调度执行的方法。

[0034] 本发明提出的疏散路径规划策略可以应对某一台边缘服务器上产生的突发负载，通过路径疏散管理和决策，决定疏散任务在哪台服务器执行以及任务从什么链路迁移到这台服务器。于是，给定了任务的信息，我们可以根据当前网络中的链路繁忙状态和边缘服务器的可用资源状况计算出任务进行迁移的开始时间以及任务在网络中迁移的时间开销，从而提出了最优任务疏散路径；一旦任务疏散到一台边缘服务器，那么其连接的多台边缘服务器就可以实时不间断的对任务进行调度执行，当任务在云服务器协助处理的时间开销不确定的情况，对所有任务进行最优的调度执行，提出了基于在线任务调度方法，从而获取近

似最优的计算任务执行策略。

[0035] 本发明方法可以弥补现有通信协议的不足，在不改变网络架构的基础上，使得边缘网络环境具备疏散任务和调度任务计算的能力。而且，本发明结合了边缘计算经典的网络结构即边缘服务器和云服务器两者各自的作用，进行任务计算的高效调度，使得突发的用户任务能够在较短的时间内完成，从而也保障了整个系统的服务质量。

附图说明

[0036] 图1为本发明基于边缘服务器拓扑结构的系统模型示意图。

[0037] 图2为本发明分布式并行协同处理边缘服务器本地任务执行时序示意图。

具体实施方式

[0038] 为了更为具体地描述本发明，下面结合附图及具体实施方式对本发明的技术方案进行详细说明。

[0039] 本发明针对如图1所示的边缘服务器拓扑结构，提出了一种突发任务卸载与调度方法，其具体实现主要包含两个部分：

[0040] (1)路径疏散管理和决策中心，其负责：①网络资源监控；记录边缘网络链路和路由状态，它能实时的检测链路中的繁忙状态，通过智能网关及时记录边缘服务器的有效资源状况。②网络任务疏散引擎；规划网络中突发负载的任务疏散到哪个边缘服务器，决定了每一个任务在何时通过具体某条链路疏散等等。

[0041] 边缘计算网络的每一条链路被划分为$l_1,\cdots,l_K$，每一条传输速率为v_k，任务i的传输时间为$t_i^{trans}=size_i/v_k$，每一条链路具有任务传输的并发数量上限为c_k^{max}，网络资源监控可以实时获取这些网络链路的可用资源，为每一个需要疏散的任务进行网络上链路的规划。任务疏散引擎对网络突发负载任务进行疏散的时候，首先遍历每一个需要疏散的任务，任务i的疏散路径从起始流经的链路开始，依次定义为$(l_1^i,\ldots,l_s^i)$，l_s^i为任务i最后流经的网络链路。为该任务寻找一条无环路径记作$path_i=(l_1^i,\ldots,l_s^i)$，每一条链路的资源使用情况的约束定义为$c_k^{max}\geq\sum_{i\in N}o_i^k(t)$，$o_i^k(t)=1$表示t时刻任务i占用了链路$l_k$，否则为0。

[0042] 假设在一台发生任务拥塞的边缘服务器上一共有N个任务需要疏散，遍历所有任务，当一个任务确定了其疏散路径后，最终会疏散到某一台边缘服务器，N个任务的疏散路径定义这样的疏散路径为一组可行解。任务i完全通过一条疏散路径所需的延迟为$trans_i=\sum_{k\in K}t_i^{trans}$，计算得到所有任务路径规划的总延迟作为疏散路径的一个解，初始化m个解作为集合$\{(path_{n1},\cdots,path_{N1}),\cdots,(path_{nm},\cdots,path_{Nm})\}$，对应的疏散完成时间的集合为$\{y_1,\cdots,y_m\}$，定义$x_i=(path_{ni},\cdots,path_{Ni})$。

[0043] 在一个现有的边缘环境下，服务器链路拓扑是确定的，假设已经得到了任务拥塞服务器到其余各台服务器之间的无环路径$\{path_1,path_2,\ldots,path_L\}$，L表示所有路径数量，该路径集合是用单个任务通过时间大小的升序排序而成；进一步，我们将上述的路径集合用链路对应的序号表示即{1,...,L}。

[0044] 疏散路径算法中定义了一个窗口，其大小为W，设定路径选择参数为$\gamma\in R$，阈值α

∈[0,1]。首先初始化m个可行解为$S_0=\{(x_1,y_0),\cdots,(x_m,y_m)\}$，那么先计算当前m个解中的最小值对应的解，定义$(x_t,y_t)=(x_{min},y_{min})$，$V_t=\emptyset$。将以下过程迭代T次，定义t为该循环内的指示序号，使得t＝1到t＝T。在每一轮t的遍历中，使i定义在范围[(t-1)W,tW]，遍历i，从而使i＝(t-1)W到i＝tW。

[0045] 每一次遍历i，系统生成一个随机数λ，如果小于α，那么$\tilde{x}_t^i=\lceil x_t^i/\gamma \rceil$，$I_v=I\backslash\{i\}$，$I_f=I_f\cup\{i\}$，否则$\tilde{x}_t^i=rand(1,L)$，rand(1,L)表示从1到L中随机取一个值。其中$\tilde{x}_t^i$表示任务i对应的疏散路径序号，是由x_t^i的序号除以参数γ，并且向上取整而得到。而I是一个集合，每一个元素对应一个疏散任务的序号，由所有需要疏散的任务随机排列生成。I_f表示已经为任务安排好疏散路径的那些任务对应的序号集合，I_v表示未被安排好疏散路径的任务的集合。

[0046] 然后在T次迭代结束后，计算$\bar{x}_t$这个解所对应的所有任务疏散的时间值，即$\bar{y}_t=c(\bar{x}_t)$，$y_t=c(x_t)$，c()表示计算疏散任务所需时间的函数。

[0047] 如果$\bar{y}_t<y_t$，保持I_v、I_f不变，否则从疏散任务的索引集合I中随机选择数量等于$|I_f|$的任务组成一个新的集合替换I_f，然后使得$I_v=I\backslash I_f$。

[0048] 最后重新构建m个解，我们对I_v集合中的任务依次随机从{1,...,L}选取一条疏散路径序号，而保持I_f集合中的任务对应的疏散路径序号不发生变化，得到一个新的解(x_h,y_h)。重复上述步骤m次，使得$V_t=V_t\cup(x_h,y_h)$，V_t表示新构建的m个解。$y_{min}=\min V_t\cup\{y_t,\bar{y}_t\}$，开始新一轮t+1的迭代；T轮迭代结束后，$(x_{min},y_{min})$即为所要求的为所有疏散任务所安排的最优疏散路径以及疏散所需要的总时间。

[0049] (2)边缘网络调度器，主要用于调度突发负载的任务，并且以最优的策略在边缘服务器上执行。每一台边缘服务器执行的任务都需要向云服务器请求任务处理协助，边缘服务器向云服务器请求协助处理任务，由于每台边缘服务器执行任务所需要的时间和传输时间未知的原因，多台边缘服务器进行数据请求和任务的执行只能进行在线的调度。

[0050] 该调度算法以到达边缘服务器的任务全部都执行完的时间最短为目标，设计了一种基于边缘服务器与云服务器分布式并行协同处理的在线调度优化算法。每一台边缘服务器都要向请求协助执行任务，在调度策略中，任务执行都需要获取所需的相应的数据信息，或者需要云服务器的授权，也可能需要云服务器的计算处理，这些过程统称为云服务器的协作处理，考虑到边缘边服务器往往数量较多，而远端只有一个中心云服务器的场景，往往中心云服务器是整个任务疏散过程的瓶颈所在。

[0051] 在一个边缘网络环境下的几台边缘服务器以流水线的方式执行迁移到自身的任务，然后每一台边缘服务器执行任务的一部分计算任务后，都需要向云服务器请求任务协助处理，每次协助处理云服务器将任务传回边缘服务器的时间是未知的，每一台边缘服务器顺序地执行到达他们的任务。每一个任务分为了三个步骤，分为任务预处理阶段、云服务协助处理阶段和任务最终处理阶段。

[0052] 基于边缘服务器与云服务器分布式并行协同处理的在线调度优化算法分为初始化和并行任务调度两部分。

[0053] 初始化部分策略为:边缘服务器数量为M,设定缓存参数为$|buf_j|$和$|buf_{re}|$,前者表示第j台边缘服务器处理初试时,首先处理完$|buf_j|$个任务后就等待系统决策信号,才能继续执行后续任务;同理,后者表示云服务器初始时,如图2所示,在对应的$step_2$,$step_5$,...,$step_{M*3-1}$阶段时候需要执行完的任务数量,然后就进入等待系统决策信号的触发才能继续执行后续任务。$step_1$、$step_2$、$step_3$表示任务预处理阶段,云服务协助处理阶段和任务最终处理阶段,其中$step_1$和$step_3$在边边缘服务器j执行,$step_2$在云服务器上执行,且$|buf_j|=3$,$|buf_{re}|=1$。

[0054] 一旦上述的初始化过程结束,系统就进入以下调度流程,首先定义:

[0055] 任务i一旦开始云服务器协助阶段的处理,那么定义该事件集合为$event_1=\{t=ts_{i,1}^{re}\}$,下标1表示云服务器对应的$step_2$。

[0056] 任务i一旦结束了云服务器协助阶段的处理,那么定义该事件集合为$event_2=\{t=tc_{i,3j-1}^{re}\}$,下标3j-1表示云服务对应的步骤,如图2所示从$step_5$开始及之后的步骤(j≥1),边缘服务器j的任务预处理阶段与云服务器3j-1步骤对应。

[0057] 任务i一旦结束了在边缘服务器j上的起始阶段的处理,那么定义该事件集合为$event_3=\{t=tc_{i,j}^{pr}\}$,下标j表示第j台边缘服务器。

[0058] 任务i一旦结束了在边缘服务器j上的最后阶段的处理,那么定义该事件集合为$event_4=\{t=tc_{i,j}^{fi}\}$。

[0059] 其中:tc表示任务完成的时刻,ts表示任务开始的时刻,上标re表示任务云服务协助处理阶段,上标pr任务初试阶段,上标fi表示任务最终处理阶段。

[0060] 并行任务调度策略为:并行的监测每一台边缘服务器上运行的任务状态和云服务器上运行的任务状态,一旦有属于$event_1$,$event_2$,$event_3$,$event_4$任意集合中的任意事件发生,系统立即进入该边缘服务器或者云服务器相应的处理流程里,所有事件都是并行执行,所以服务器都是并行执行各自可以执行的任务,且定义云服务器执行M次任务为一个cycle,定义M台边缘服务器序号为1,...,M,分别对应云服务器的$step_2$,$step_5$,...,$step_{M*3-1}$,如图2所示。初始化$cycle_r=0$,$cycle_j=0$,$j\in\{1,2,...,M\}$,startCycleEn=[0,0,...,0],维度为M,每一个分量表示边缘服务器j是否可以执行新一轮任务,1表示可以,0表示不可以。

[0061] 如果$event_1$发生了,那么立即将$cycle_r$的值自增1,定义此时空闲的边缘服务器索引为j,那么使得这些服务器从缓存中以FIFO的顺序启动一个任务。如果此时边缘服务器j处于工作状态或者缓存无可执行任务,那么startCycleEn[j]=1,表示边缘服务器j可以在空闲时候开始新的cycle,即新的任务可以执行。如果此时边缘服务器j处于空闲状态且缓存中有可执行的任务,那么若$cycle_r\geq cycle_j$,那么让边缘服务器j从$buf_{pr,j}\cup buf_{fi,j}$中以FIFO的顺序启动一个任务,$cycle_j$自增1,若自增后$cycle_r=cycle_j$,则startCycleEn[j]=0。

[0062] 如果$evetn_2$发生了,那么让云服务器从缓存中以FIFO的顺序启动一个任务,如果缓存空间为0,则不执行。

[0063] 如果$event_4$或$event_3$发生了并且满足startCycleEn[j]=1和$cycle_r\geq cycle_j$,那

么让边缘服务器j从$buf_{pr,j} \cup buf_{fi,j}$中以FIFO的顺序启动一个任务，$cycle_j$自增1，若自增后$cycle_r = cycle_j$，则startCycleEn[j]＝0。

[0064] 最后，所有任务都将以策略的优化后的最短时间内执行完毕。

[0065] 上述对实施例的描述是为便于本技术领域的普通技术人员能理解和应用本发明，熟悉本领域技术的人员显然可以容易地对上述实施例做出各种修改，并把在此说明的一般原理应用到其他实施例中而不必经过创造性的劳动。因此，本发明不限于上述实施例，本领域技术人员根据本发明的揭示，对于本发明做出的改进和修改都应该在本发明的保护范围之内。

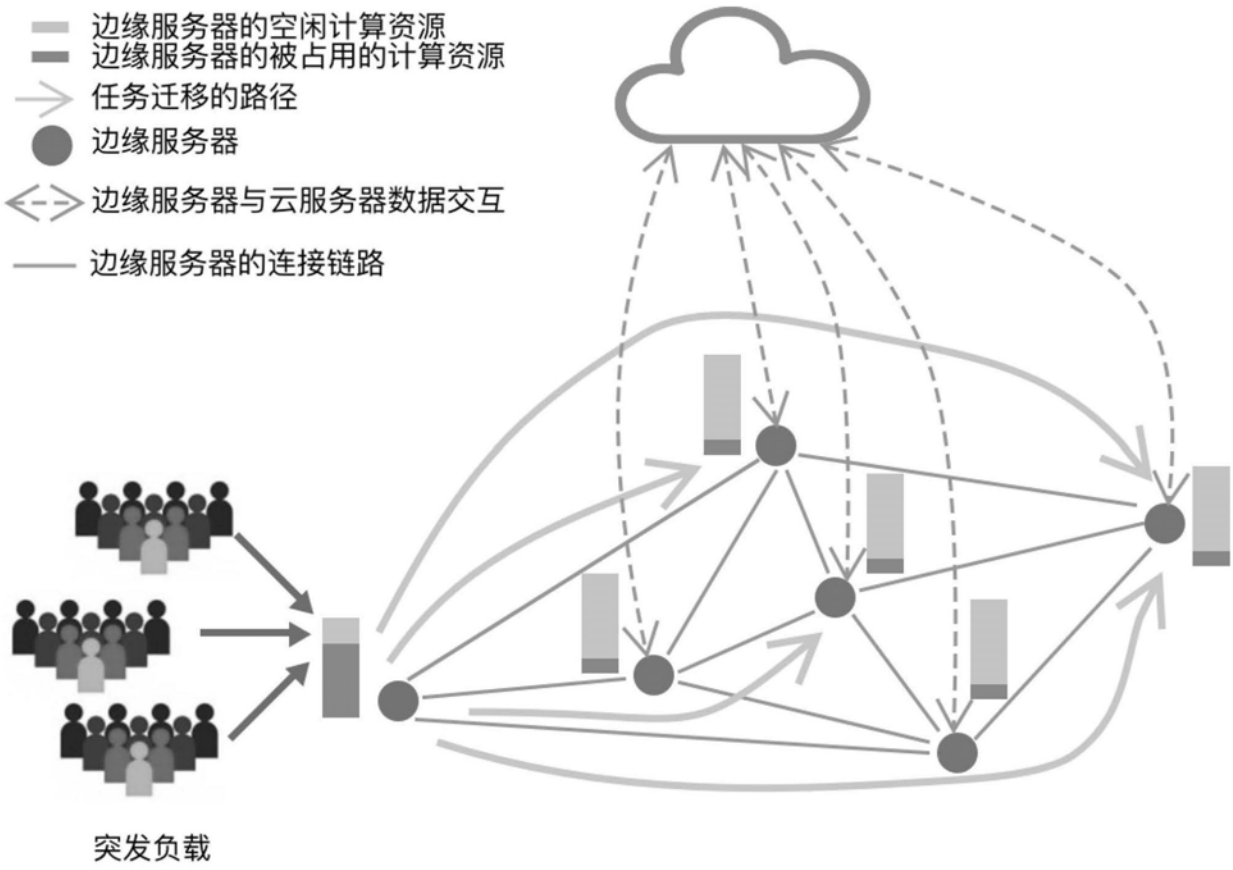

边缘服务器的空闲计算资源
边缘服务器的被占用的计算资源
任务迁移的路径
边缘服务器
边缘服务器与云服务器数据交互
边缘服务器的连接链路
突发负载

图1

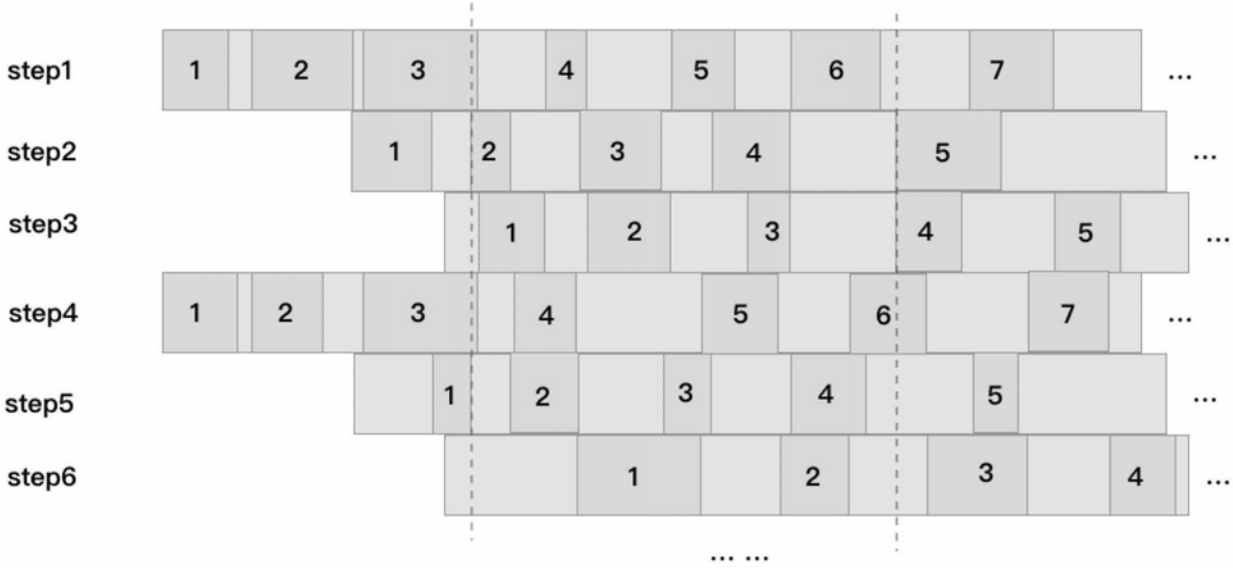

step1
step2
step3
step4
step5
step6

图2