

基于深度强化学习的任务调度网络训练方法

本申请要求于 2022 年 12 月 26 日提交中国专利局、申请号为 202211676872.0、申请名称为“基于深度强化学习的任务调度网络训练方法”的中国专利申请的优先权，其全部
5 内容通过引用结合在本申请中。

技术领域

本发明涉及数据中心任务调度领域，尤其涉及一种基于深度强化学习的任务调度网络训练方法。

背景技术

10 数据中心所包含的服务器集群及辅助冷却系统等是不间断运行的高能耗设备。2021 年全国数据中心能源消耗约为 2166 亿千瓦时，占全国总用电量的 2.6%，因用电产生的间接温室气体排放约达 1.35 亿吨，占全国碳排放量的 1.14%左右。并且过去十年间，我国各个数据中心行业整体用电量以年均超 10%的速度逐年增加，数据中心能耗问题正越来越得到人们的关注。2021 年 11 月 17 日发布的《“十四五”信息通信行业发展规划》中明确指出，
15 要加大对数据中心等重点领域的绿色化改造力度。因此，数据中心的低碳、经济运行，正成为数据中心改造的一个重要方向。

数据中心包括若干的服务器，数据中心在工作过程中，接收到的作业内容也会包括若干的任务，对这些任务进行合理的调度，从而将其分配给合适的服务器进行处理，对降低数据中心能耗、提高数据中心对任务处理的及时性等方面具有重要的意义。然而现有任务
20 调度算法通常忽略数据中心低碳经济运行的要求，无法做到在保证任务处理及时性的基础上实现数据中心低碳经济运行，因此数据中心的运行成本居高不下。

因此，相关技术中的任务调度算法无法在保证任务处理及时性的基础上实现数据中心低碳经济运行。针对上述的问题，目前尚未提出有效的解决方案。

在背景技术部分中公开的以上信息只是用来加强对本文所描述技术的背景技术的理解。因此，背景技术中可能包含某些信息，这些信息对于本领域技术人员来说并未形成在
25 已知的现有技术。

发明内容

本发明实施例提供了一种基于深度强化学习的任务调度网络训练方法，以至少解决相关技术中的任务调度算法无法在保证任务处理及时性的基础上实现数据中心低碳经济运

行的技术问题。

根据本发明实施例的第一个方面，提供了一种基于深度强化学习的任务调度网络训练方法，包括：获取待执行作业，待执行作业包括多个任务；采用任务调度网络对多个任务进行调度；根据任务调度网络的调度结果，确定多个时间步的奖励，各个时间步的奖励均
5 包括碳奖励、电费奖励和任务延迟奖励，碳奖励表征数据中心在时间步的碳排放成本，电费奖励表征数据中心在时间步的电费成本，任务延迟奖励表征数据中心在时间步的任务延迟情况；根据多个时间步的奖励，计算回报 G_t ，回报 G_t 为从时间步 t 到调度过程结束的累积奖励；至少根据回报 G_t ，基于梯度算法对任务调度网络的参数进行调整。

进一步地，根据任务调度网络的调度结果，确定多个时间步的奖励包括：对于任意的
10 时间步 t ，根据公式 $r_t = -r_t^{carbon} - r_t^{cost} - r_t^{Qos}$ 计算时间步 t 的奖励；其中， r_t^{carbon} 为时间步 t 的碳奖励， r_t^{cost} 为时间步 t 的电费奖励， r_t^{Qos} 为时间步 t 的任务延迟奖励。

进一步地， $r_t^{carbon} = -P_t^{DC} \cdot e_t \cdot p^{carbon}$ ， $r_t^{cost} = -P_t^{DC} \cdot q_t$ ， $r_t^{Qos} = -\sum_{i=1}^N (T_i^{cpu} \cdot p^{cpu} + T_i^{MEM} \cdot p^{MEM})$ ；其中， $P_t^{DC} = F(M) + V_t(U_t^{cpu}, M)$ ， $F(M) = M(P_{idle} + (PUE - 1) \cdot P_{peak})$ ， $V_t(U_t^{cpu}, M) = \sum_{i=1}^M (P_{peak} - P_{idle}) [2U_{i,t}^{cpu} - (U_{i,t}^{cpu})^r]$ ， P_t^{DC} 为数据中心在时间步 t 的功率，
15 e_t 为系统碳排放率， p^{carbon} 为碳排放价格， q_t 为时间步 t 的电价， N 为大于1的整数， T_i^{cpu} 为任务 i 对 CPU 资源的占用率， p^{cpu} 为 CPU 使用的价格， T_i^{MEM} 为任务 i 对内存资源的占用率， p^{MEM} 为内存资源使用的价格， M 为数据中心的服务器数量， $F(M)$ 为静态功率， $V_t(U_t^{cpu}, M)$ 为可变功率， P_{idle} 为服务器空闲时的额定功率， P_{peak} 为服务器完全占用时的额定功率，指数 r 为常数，PUE 为数据中心能效指标。

进一步地，根据多个时间步的奖励，计算回报 G_t 包括：对多个时间步的奖励进行放大或缩小，得到多个缩放奖励，其中，多个缩放奖励所属数值区间的范围窄于多个奖励所属
20 数值区间的范围；根据多个缩放奖励，计算回报 G_t 。

进一步地，对多个时间步的奖励进行放大或缩小，得到多个缩放奖励包括：对于任意的
时间步 t ，根据公式 $r_t' = (r_t - \mu) / (\sigma + \epsilon)$ 计算时间步 t 的缩放奖励 r_t' ，其中， r_t 为时间
25 步 t 的奖励， μ 通过公式 $\mu \leftarrow \mu' + (r - \mu) / k$ 随着 k 的值的增加而更新， σ 通过公式 $\sigma \leftarrow \sqrt{s/k}$ 随着 k 的值的增加而更新， s 通过公式 $s \leftarrow \sigma + (r - \mu') \cdot (r - \mu)$ 计算更新，其中， $k = n - 1$ ， n 为时间步 t 的编号，在 $k=0$ 的情况下， $\mu = r$ ， $\sigma = r$ ， r 为第一个时间步获得的奖励，在 k 不为 0 的情况下， $\mu = \mu'$ 。

进一步地，根据回报 G_t ，基于梯度算法对任务调度网络的参数进行调整包括：确定当
30 前状态价值的估计；根据回报 G_t 和当前状态价值的估计，基于梯度算法对任务调度网络的参数进行调整。

进一步地，根据回报 G_t 和当前状态价值的估计，基于梯度算法对任务调度网络的参数
进行调整，包括：采用方程 $\theta \leftarrow \theta + \alpha^\theta \gamma^t \delta \frac{\nabla_{\theta} \pi_{\theta}(a_t | s_t, \theta_t)}{\pi_{\theta}(a_t | s_t, \theta_t)}$ 迭代更新任务调度网络的参数 θ ，
其中， α^θ 为参数 θ 的学习率， δ 为回报 G_t 与当前状态价值的估计之间的差值， γ^t 为时间步 t
35 的折扣因子， $\pi_{\theta}(a_t | s_t, \theta_t)$ 为时间步 t 的 s 状态下，任务调度网络采取 a 动作的概率， $\nabla_{\theta} \pi_{\theta}(a_t | s_t, \theta_t)$ 为 $\pi_{\theta}(a_t | s_t, \theta_t)$ 对 θ 的导数。

进一步地，确定当前状态价值的估计包括：计算参数向量 w 与特征向量 $x(s_t)$ 的内积，

得到当前状态价值的估计 $\hat{v}(s_t, w)$ ；其中， $x(s_t)$ 为时间步 t 的状态的特征向量。

进一步地，参数向量 w 通过公式 $w \leftarrow w + \alpha^w [G_t - \hat{v}(s_t, w)] \nabla \hat{v}(s_t, w)$ 迭代更新；其中， α^w 为 $\hat{v}(s_t, w)$ 的学习率， $\nabla \hat{v}(s_t, w)$ 为 $\hat{v}(s_t, w)$ 对 w 的导数。

进一步地，采用任务调度网络对多个任务进行调度包括：确定多个可执行的动作，动作用于指示将任意一个任务调度至任意一个服务器执行；将多个可执行的动作输入任务调度网络，输出各个可执行的动作被选中的概率；根据各个可执行的动作被选中的概率，从多个可执行的动作中确定多个调度动作；根据多个调度动作，对多个任务进行调度。

进一步地，确定多个可执行的动作包括：从待执行作业包括的多个任务中，确定一个或多个就绪任务，就绪任务包括不存在父任务的任务，和/或，就绪任务包括父任务执行完毕的任务，父任务为就绪任务执行时依赖的任务；针对每一个就绪任务，确定一个或多个目标服务器，目标服务器为满足就绪任务所需的计算资源的服务器；确定将任意一个就绪任务调度至任意一个目标服务器的动作为一个可执行的动作。

根据本发明实施例的第二个方面，还提供了一种基于深度强化学习的数据中心任务调度方法，包括：获取待调度作业，所述待调度作业包括多个待调度任务；采用任务调度网络对多个所述待调度任务进行调度，所述任务调度网络采用上述的基于深度强化学习的任务调度网络训练方法训练得到。

根据本发明实施例的第三个方面，还提供了一种可读存储介质，其上存储有计算机指令，其中，计算机指令被处理器执行时实现上述的基于深度强化学习的任务调度网络训练方法；或者，计算机指令被处理器执行时实现上述的基于深度强化学习的数据中心任务调度方法。

根据本发明实施例的第四个方面，还提供了一种终端设备，包括存储器和处理器，存储器上存储有计算机指令，其中，计算机指令被处理器执行时实现上述的基于深度强化学习的任务调度网络训练方法；或者，计算机指令被处理器执行时实现上述的基于深度强化学习的数据中心任务调度方法。

本发明实施例的基于深度强化学习的任务调度网络训练方法包括：获取待执行作业，待执行作业包括多个任务；采用任务调度网络对多个任务进行调度；根据任务调度网络的调度结果，确定多个时间步的奖励，各个时间步的奖励均包括碳奖励、电费奖励和任务延迟奖励，碳奖励表征数据中心在时间步的碳排放成本，电费奖励表征数据中心在时间步的电费成本，任务延迟奖励表征数据中心在时间步的任务延迟情况；根据多个时间步的奖励，计算回报 G_t ，回报 G_t 为从时间步 t 到调度过程结束的累积奖励；至少根据回报 G_t ，基于梯度算法对任务调度网络的参数进行调整。在进行任务调度网络的深度强化训练的过程中，在调度网络执行了对多个任务的调度操作之后，会计算多个时间步的奖励，进而计算出整个调度过程的回报 G_t ，然后根据回报 G_t 基于梯度算法（例如梯度上升或梯度下降）对任务调度网络进行参数调整。在进行任务调度网络训练的过程中，会对每个时间步建立相应的奖励，而每个时间步的奖励构成包括碳奖励、电费奖励和任务延迟奖励三项。然后计算多

个时间步的累积奖励,得到回报 G_t ,并根据回报 G_t 基于梯度算法对任务调度网络进行训练。由于在进行任务调度网络训练的过程中,针对每个时间步的奖励引入碳奖励、电费奖励和任务延迟奖励三个奖励项,因此在训练过程中便会考虑整个任务执行过程的碳排放成本、电费成本以及任务延迟情况,使得数据中心调度过程朝着低碳、经济、低延迟的方向去优化,进而使得训练出的任务调度网络可使数据中心在保证低延迟的基础上实现低碳经济运行,解决了相关技术中的任务调度算法无法在保证任务处理及时性的基础上实现数据中心低碳经济运行的技术问题。

附图说明

此处所说明的附图用来提供对本发明的进一步理解,构成本申请的一部分,本发明的示意性实施例及其说明用于解释本发明,并不构成对本发明的不当限定。在附图中:

图1为本发明实施例提供的一种基于深度强化学习的任务调度网络训练方法的流程图;

图2为本发明实施例提供的一种基于深度强化学习的任务调度网络训练装置的示意图;

图3为本发明实施例的基于深度强化学习的任务调度网络训练方法在对多个时间步的奖励进行放大或缩小从而得到多个缩放奖励的流程示意图;

图4为采用本发明实施例的基于深度强化学习的任务调度网络训练方法进行任务调度网络训练的过程示意图。

具体实施方式

为了使本技术领域的人员更好地理解本发明方案,下面将结合本发明实施例中的附图,对本发明实施例中的技术方案进行清楚、完整地描述,显然,所描述的实施例仅仅是本发明一部分的实施例,而不是全部的实施例。基于本发明中的实施例,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,都应当属于本发明保护的范围。

需要说明的是,本发明的说明书和权利要求书及附图中的术语“第一”、“第二”等是用于区别不同对象,而不是用于限定特定顺序。

图1是根据本发明实施例的基于深度强化学习的任务调度网络训练方法,任务调度网络用于对数据中心的任务进行调度,调度即将任务分配给某一个服务器去处理,当某一个任务被任务调度网络调度至某一个服务器后,服务器对该任务进行计算处理。本发明实施例的基于深度强化学习的任务调度网络训练方法用于对上述的任务调度网络进行训练,如图1所示,该方法包括如下步骤:

步骤S102,获取待执行作业,待执行作业包括多个任务;

步骤S104,采用任务调度网络对多个任务进行调度;

步骤S106,根据任务调度网络的调度结果,确定多个时间步的奖励,各个时间步的奖励均包括碳奖励、电费奖励和任务延迟奖励,碳奖励表征数据中心在时间步的碳排放成本,

电费奖励表征数据中心在时间步的电费成本，任务延迟奖励表征数据中心在时间步的任务延迟情况；

步骤 S108，根据多个时间步的奖励，计算回报 G_t ，回报 G_t 为从时间步 t 到调度过程结束的累积奖励；

5 步骤 S110，至少根据回报 G_t ，基于梯度算法对任务调度网络的参数进行调整。

本发明实施例的基于深度强化学习的任务调度网络训练方法包括：获取待执行作业，待执行作业包括多个任务；采用任务调度网络对多个任务进行调度；根据任务调度网络的调度结果，确定多个时间步的奖励，各个时间步的奖励均包括碳奖励、电费奖励和任务延迟奖励，碳奖励表征数据中心在时间步的碳排放成本，电费奖励表征数据中心在时间步的电费成本，任务延迟奖励表征数据中心在时间步的任务延迟情况，其中，时间步即某一个时间点/时刻；根据多个时间步的奖励，计算回报 G_t ，回报 G_t 为从时间步 t 到调度过程结束的累积奖励；至少根据回报 G_t ，基于梯度算法对任务调度网络的参数进行调整。在进行任务调度网络的深度强化训练的过程中，调度网络执行了对多个任务的调度操作之后，会计算多个时间步的奖励，进而计算出整个调度过程的回报 G_t ，然后根据回报 G_t 基于梯度算法
10 （例如梯度上升或梯度下降）对任务调度网络进行参数调整。也就是说，在进行任务调度网络训练的过程中，会对每个时间步建立相应的奖励，而每个时间步的奖励构成包括碳奖励、电费奖励和任务延迟奖励三项。然后计算多个时间步的累积奖励，得到回报 G_t ，并根据回报 G_t 基于梯度算法对任务调度网络进行训练。由于在进行任务调度网络训练的过程中，针对每个时间步的奖励引入碳奖励、电费奖励和任务延迟奖励三个奖励项，因此在训练过
15 程中便会考虑整个任务执行过程的碳排放成本、电费成本以及任务延迟情况，使得数据中心调度过程朝着低碳、经济、低延迟的方向去优化，进而使得训练出的任务调度网络可使数据中心在保证低延迟的基础上实现低碳经济运行，解决了相关技术中的任务调度算法无法在保证任务处理及时性的基础上实现数据中心低碳经济运行的技术问题。

也就是说，上述调度网络训练过程中，会将各个时间步的奖励记为三部分：碳奖励、
25 电费奖励和任务延迟奖励。碳奖励表征数据中心在该时间步的碳排放成本，数据中心电能消耗会产生碳排放成本，而碳奖励即用于对其进行表征的奖励项。同样的，数据中心在运行时会产生电费成本，而电费奖励即用于对其进行表征的奖励项。另外，本发明实施例还引入了任务延迟奖励的奖励项，任务延迟奖励用于表征数据中心在相应时间步的任务延迟情况，数据中心在处理某一个时间步的任务时产生的延迟越大，则任务延迟奖励也越大。
30 而通过将这三者纳入奖励计算以及总回报 G_t 的计算，可以使得任务调度网络训练过程中优化数据中心的上述三部分运行成本，有利于使得数据中心在保证服务质量（Quality Of Service，简称 QOS）的基础上降低电费支出和碳排放支出，实现数据中心低碳经济运行的目标。

通过将深度强化学习应用于任务调度网络的训练，任务调度网络可基于特定的调度动
35 作与周围环境进行交互，在动作和环境的作用下，调度网络所处的状态会发生变化，同时

环境会给出一个回报（奖励）。如此循环下去，任务调度网络与环境不断地交互从而产生很多数据。强化学习算法利用产生的数据修改任务调度网络自身的参数，再与环境交互，产生新的数据，并利用新的数据进一步改善自身的行为，经过数次迭代学习后，调度策略网络便可学到较好的任务调度策略，实现任务的智能调度。

5 其中，回报 G_t 为时间步 t 到调度过程结束的累积奖励，其由整个任务调度过程中的奖励累积得到，在实际对多个时间步的奖励进行累积的过程中，不同时间步的奖励可有不同的权重，例如可通过公式 $G_t = r_t + \gamma r_{t+1} + \dots + \gamma^{T-1-t} r_T$ 来计算回报 G_t 。其中， $\gamma \in [0,1]$ ， γ 为衡量当前和未来奖励重要性的折现因子。在一个优选的实施例中，当前奖励和未来奖励同样重要，此时 $\gamma = 1$ 。

10 在一个具体的实施例中，根据任务调度网络的调度结果，确定多个时间步的奖励包括：对于任意的时间步 t ，根据公式 $r_t = -r_t^{carbon} - r_t^{cost} - r_t^{Qos}$ 计算时间步 t 的奖励；其中， r_t^{carbon} 为时间步 t 的碳奖励， r_t^{cost} 为时间步 t 的电费奖励， r_t^{Qos} 为时间步 t 的任务延迟奖励。

在本实施例中，对于任意的时间步 t ，其奖励由三部分构成，即 r_t^{carbon} 、 r_t^{cost} 和 r_t^{Qos} ，
15 奖励即成本，由于成本越低越好，因此是采用负的奖励，最终的时间步 t 的奖励计算形式为上述三部分奖励的负值之和，其中， r_t^{carbon} 为所述时间步 t 的碳排放成本， r_t^{cost} 为所述时间步 t 的电费成本， r_t^{Qos} 为所述时间步 t 的任务延迟情况，所述任务延迟情况表征所述时间步 t 的至少一个所述任务的延迟情况。通过采用这种奖励计算形式，将碳排放成本、
20 电费成本以及任务延迟情况纳入奖励范围，在后续基于梯度算法进行任务调度网络训练的过程中，可以使得任务调度网络向着运行成本最小的方向优化，而此处的运行成本考虑到了碳排放成本、电费成本和任务延迟情况。最终训练好的任务调度网络在进行任务调度的过程中，可以在保证较小的任务延迟的情况下，减小碳排放成本和电费成本。

为了能够更加准确地计算上述的三部分奖励，本实施例中， $r_t^{carbon} = -P_t^{DC} \cdot e_t \cdot p^{carbon}$ ， $r_t^{cost} = -P_t^{DC} \cdot q_t$ ， $r_t^{Qos} = -\sum_{i=1}^N (T_i^{cpu} \cdot p^{cpu} + T_i^{MEM} \cdot p^{MEM})$ ；其中， $P_t^{DC} = F(M) + V_t(U_t^{cpu}, M)$ ， $F(M) = M(P_{idle} + (PUE - 1) \cdot P_{peak})$ ， $V_t(U_t^{cpu}, M) = \sum_{i=1}^M (P_{peak} - P_{idle}) [2U_{i,t}^{cpu} - (U_{i,t}^{cpu})^r]$ 。
25

其中， P_t^{DC} 为数据中心在时间步 t 的功率， e_t 为系统碳排放率，煤电、风电、光伏发电等等发电形式的碳排放量是不同的，因此电力系统实时消耗的电能中各种发电形式的电能占比的不同，系统碳排放率也是不同的，例如系统碳排放率的单位可以为 tCO_2/MWh ，
30 p^{carbon} 为碳排放价格，单位可以为元/ tCO_2 ， q_t 为时间步 t 的电价，单位可以为元/Mwh， N 为大于1的整数， T_i^{cpu} 为任务 i 对 CPU 资源的占用率， p^{cpu} 为 CPU 使用的价格，例如用户使用数据中心的服务器，根据服务器 CPU 的使用情况来收费，CPU 使用的价格的单位可以为元/秒， T_i^{MEM} 为任务 i 对内存资源的占用率， p^{MEM} 为内存资源使用的价格，单位可以为元/秒， M 为数据中心的服务器数量， $F(M)$ 为静态功率， $V_t(U_t^{cpu}, M)$ 为可变功率， P_{idle} 为
35 服务器空闲时的额定功率， P_{peak} 为服务器完全占用时的额定功率，指数 r 为常数，PUE 为

数据中心能效指标。在本实施例中，采用公式 $r_t^{carbon} = -p_t^{DC} \cdot e_t \cdot p^{carbon}$ 计算碳排放成本，其中， p_t^{DC} 为数据中心在时间步 t 的功率， e_t 为系统碳排放率， p^{carbon} 为碳排放价格。采用公式 $r_t^{cost} = -p_t^{DC} \cdot q_t$ 计算电费成本，其中 q_t 为时间步 t 的电价，采用公式 $r_t^{Qos} = -\sum_{i=1}^N (T_i^{cpu} \cdot p^{cpu} + T_i^{MEM} \cdot p^{MEM})$ 任务延迟情况。这种计算方式可以实现对这三部分成本的客观、准确的计算，从而有利于提高后续的任务调度网络的训练效果。

在基于深度强化学习方式进行任务调度网络训练的过程中，存在任务调度网络不收敛或收敛速度慢的问题，为了解决这一技术问题。在本实施例中，根据多个时间步的奖励，计算回报 G_t 包括：对多个时间步的奖励进行放大或缩小，得到多个缩放奖励，其中，多个缩放奖励所属数值区间的范围窄于多个奖励所属数值区间的范围；根据多个缩放奖励，计算回报 G_t ，回报为从时间步 t 到调度过程结束的累积奖励。申请人研究后发现，在基于深度强化学习的方式进行任务调度网络训练的过程中之所以出现收敛慢或不收敛的情况，是由于奖励计算过程中的波动较大，对训练过程的收敛造成了影响。本实施例中，在确定多个时间步的奖励之后，会对这些奖励进行放大或缩小，从而得到多个缩放奖励，且多个缩放奖励所属数值区间的范围窄于多个奖励所属数值区间的范围。然后根据多个缩放奖励计算回报 G_t ，再根据具体的训练需求，至少根据回报 G_t 基于梯度算法对任务调度网络的参数进行调整。采用这种训练方法在对任务调度网络进行训练的过程中，并不会直接根据各个时间步的奖励进行任务调度网络的参数调整，而是先对多个奖励进行放大或缩小，得到多个缩放奖励，再根据缩放奖励计算回报 G_t ，进而基于梯度算法来进行参数调整。在这个过程中，通过对多个时间步的奖励进行放大或缩小，使得到的缩放奖励分布在更小的数值区间范围内，这提高了奖励分布的平稳性，能够减少甚至避免出现部分时间步奖励相对其他时间步波动较大的情况，从而加快任务调度网络的收敛速度，提高其训练效果，解决任务调度网络训练过程中不收敛或收敛速度慢的问题。

在本实施例中，为了提高奖励缩放的效果，使得缩放奖励能够更加可靠、有效地用于任务调度网络的训练，缩放后的缩放奖励 r'_t 的计算方法是用奖励 r_t 除以 r_t 的滚动折扣总和的标准差。即对多个时间步的奖励进行放大或缩小，得到多个缩放奖励包括：对于任意的时间步 t ，根据公式 $r'_t = (r_t - \mu) / (\sigma + \epsilon)$ 计算时间步 t 的缩放奖励 r'_t ，其中， r_t 为时间步 t 的奖励， μ 通过公式 $\mu \leftarrow \mu' + (r - \mu) / k$ 随着 k 的值的增加而更新， σ 通过公式 $\sigma \leftarrow \text{sqrt}(s / k)$ 随着 k 的值的增加而更新， s 通过公式 $s \leftarrow \sigma + (r - \mu') * (r - \mu)$ 计算更新，其中， $k = n - 1$ ， n 为时间步 t 的编号，在 $k=0$ 的情况下， $\mu = r$ ， $\sigma = r$ ， r 为第一个时间步获得的奖励，在 k 不为 0 的情况下， $\mu = \mu'$ 。这样，即消除了部分时间步的奖励过大或过小的问题，且缩放后得到的缩放奖励 r'_t 也能够客观准确地表示奖励情况，从而为模型优化提供依据，有利于提高任务调度网络的训练效果和加快训练过程收敛。以下结合图 3 对奖励缩放过程进行更清晰的说明，图 3 为本发明实施例的基于深度强化学习的任务调度网络训练方法在对多个时间步的奖励进行放大或缩小从而得到多个缩放奖励的流程示意图。如图 3 所示，初始状态下， k 取值为 0，在获取到当前时间步 t 的奖励 r_t 后，会采用公式 $\mu \leftarrow \mu' + (r - \mu) / k$ 更

新平均值 μ , 采用公式 $\sigma \leftarrow \text{sqrt}(s/k)$ 更新奖励的标准差 σ , 其中 s 通过公式 $s \leftarrow \sigma + (r - \mu') * (r - \mu)$ 计算更新, 然后 k 的取值加1, 根据公式 $r'_t = (r_t - \mu)/(\sigma + \epsilon)$ 计算并返回缩放奖励 r'_t , 在待执行作业的所有任务调度完成的情况下, 上述的缩放奖励计算过程也随之结束。

另外, 申请人还发现, 在实际对任务调度网络进行训练的过程中, 如果回报 G_t 的数量级较大, 且好的回报和坏的回报之间的差别又较小, 则在任务调度网络参数更新的过程中, 这种差别将难以分辨, 从而对训练过程带来不利影响, 导致不收敛或收敛慢。例如 G_t 为百万量级的数值, 好的回报和坏的回报分别对应的是100万加5和100万减5, 则这两个 G_t 之间的差别就不容易区分出来, 在参数更新的过程中, 在对这两个 G_t 进行一些运算后, 更新后的参数的变化也会微乎其微, 也会导致训练过程不收敛或收敛慢。

为了解决这一技术问题, 根据回报 G_t , 基于梯度算法对任务调度网络的参数进行调整包括: 确定当前状态价值的估计; 根据回报 G_t 和当前状态价值的估计, 基于梯度算法对任务调度网络的参数进行调整。本实施例通过引入当前状态价值的估计, 并结合回报 G_t 和当前状态价值的估计来进行策略网络参数调整, 此时, 当前状态价值的估计可以作为一个参考基准, 来辅助进行任务调度网络参数的调整, 能够起到加快训练过程收敛的作用。

在一个具体的实施例中, 根据回报 G_t 和当前状态价值的估计, 基于梯度算法对任务调度网络的参数进行调整, 包括:

采用方程 $\theta \leftarrow \theta + \alpha^\theta \gamma^t \delta \frac{\nabla_{\theta} \pi_{\theta}(a_t | s_t, \theta_t)}{\pi_{\theta}(a_t | s_t, \theta_t)}$ 迭代更新任务调度网络的参数 θ 。

其中, α^θ 为参数 θ 的学习率, δ 为回报 G_t 与当前状态价值的估计之间的差值, γ^t 为时间步 t 的折扣因子, $\pi_{\theta}(a_t | s_t, \theta_t)$ 为时间步 t 的 s 状态下, 任务调度网络采取 a 动作的概率, $\nabla_{\theta} \pi_{\theta}(a_t | s_t, \theta_t)$ 为 $\pi_{\theta}(a_t | s_t, \theta_t)$ 对 θ 的导数。本实施例中, 采用上述方程进行任务调度网络的参数 θ , 其中 δ 为所述回报 G_t 与所述当前状态价值的估计之间的差值, 通过采用 δ 来代替 G_t , δ 可以更清晰地代表调度结果比之前的平均数好多少或差多少, 从而帮助任务调度网络进行参数优化。为了方便理解, 此处可通俗理解为: 发现了一个效果很好的策略的方向, 便在这个方向上多放大一下, 即好的东西增加可能性, 不好的东西降低可能性, 从而解决任务调度网络训练过程不收敛或收敛慢的问题。如上所述, 在训练过程中, G_t 的变化可能非常大, 例如 G_t 为百万量级的数, 一个比较好的调度策略产生的 G_t 为100万加5, 比较差的调度策略产生的 G_t 为100万减5, 在这个数量级上差别很小(1000005与999995差别较小), 再乘上学习率之后将难以分辨两者的优劣, 而通过引入状态价值的估计, 采用回报 G_t 与所述当前状态价值的估计之间的差值, 可增加两者之间的辨识度。例如两者均减去100万, 得到5和-5, 两者差别更明显, 对训练过程更有利, 帮助策略网络学习调度策略。

本实施例中, 确定当前状态价值的估计包括: 计算参数向量 w 与特征向量 $x(s_t)$ 的内积, 得到当前状态价值的估计 $\hat{v}(s_t, w)$; 其中, $x(s_t)$ 为时间步 t 的状态的特征向量。采用值函数近似 $\hat{v}(s_t, w)$ 作为当前状态价值, 可以方便当前状态价值的估计, 具体地, 当前状态价值的估计 $\hat{v}(s_t, w)$ 可以通过参数向量 w 与特征向量 $x(s_t)$ 的内积来计算, 其中, 针对每一个

时间步 t 的状态 s_t , 都有一个特征向量。在一个具体的实施例中, 特征向量 $x(s_t) = (x_1(s_t), x_2(s_t) \dots, x_{M+m+1}(s_t))$ 。

在本实施例中, 为了保证当前状态价值估计的准确性, 参数向量 w 会随着 θ 一起更新, 具体地, 参数向量 w 通过公式 $w \leftarrow w + \alpha^w [G_t - \hat{v}(s_t, w)] \nabla \hat{v}(s_t, w)$ 迭代更新; 其中, α^w 为 $\hat{v}(s_t, w)$ 的学习率, $\nabla \hat{v}(s_t, w)$ 为 $\hat{v}(s_t, w)$ 对 w 的导数。

采用任务调度网络对多个任务进行调度包括: 确定多个可执行的动作, 动作用于指示将任意一个任务调度至任意一个服务器执行; 将多个可执行的动作输入任务调度网络, 输出各个可执行的动作被选中的概率; 根据各个可执行的动作被选中的概率, 从多个可执行的动作中确定多个调度动作; 根据多个调度动作, 对多个任务进行调度。在采用任务调度网络对多个任务进行调度的过程中, 会先确定多个可执行的动作, 其中, 动作即在某一个时间步将某个任务分配给某个服务器, 用于指示将任意一个任务调度至任意一个服务器执行, 例如动作 $a_t(\varphi_i, s_j)$ 是一个可执行的动作, 该动作即将任务 φ_i 分配给服务器 s_j 处理。在确定多个可执行动作后, 会将多个可执行动作输入任务调度网络, 从而输出得到各个可执行动作被选中的概率, 也就是说, 任务调度网络输出的是概率, 概率越大说明相应的任务调度网络越倾向于选中该动作, 基于任务调度网络输出的概率, 可从上述的可执行的动作中选出调度动作, 并根据调度动作进行任务调度。

在本实施例中, 对如何确定多个可执行的动作进行说明, 确定多个可执行的动作包括: 从待执行作业包括的多个任务中, 确定一个或多个就绪任务, 就绪任务包括不存在父任务的, 和/或, 就绪任务包括父任务执行完毕的任务, 父任务为就绪任务执行时依赖的任务; 针对每一个就绪任务, 确定一个或多个目标服务器, 目标服务器为满足就绪任务所需的计算资源的服务器; 确定将任意一个就绪任务调度至任意一个目标服务器的动作为一个可执行的动作。

在上述的确定多个可执行的动作的过程中, 会基于任务间依赖关系和任务对计算资源的需求来确定, 具体地, 会先从多个任务中确定一个或多个就绪任务, 其中, 就绪任务包括不存在父任务的所述任务, 和/或, 所述就绪任务包括父任务执行完毕的所述任务, 所述父任务为所述就绪任务执行时依赖的任务。然后, 针对每一个就绪任务, 会确定与其对应的目标服务器, 目标服务器需要满足该就绪任务对计算资源的需求, 例如一个就绪任务需求的计算资源有 10 个服务器能够满足, 则这 10 个服务器均为目标服务器, 将该就绪任务调度至这 10 个服务器中的任意一个服务器的动作均是一个可执行的动作。这样, 在进行任务调度的过程这能够充分考虑了任务间依赖关系和任务对计算资源的需求, 有利于提高任务调度的合理性。

在一些实施场景中, 任务调度网络的输入维度是固定的, 而到达的任务数量是位置的, 即任务的数量在变化, 此时, 为了顺利实现任务调度网络的训练, 在本实施例中, 从待执行作业包括的多个任务中, 确定一个或多个就绪任务包括: 按照任务排列顺序, 从待执行作业包括的多个任务中, 确定前预设数量的就绪任务。也就是说, 会按照任务排列顺序,

从待执行作业包括的多个任务中，确定前预设数量的为就绪任务，也就是说，任务调度网络在每次调度过程中，仅观察固定数量的就绪任务，从而保证输入维度的固定，节省计算开销。

以下结合图 4 对本发明实施例的基于深度强化学习的任务调度网络训练方法进行说明，图 4 为采用本发明实施例的基于深度强化学习的任务调度网络训练方法进行任务调度网络训练的过程示意图，如图 4 所示，在训练开始时，会初始化一个策略网络 $\pi(a|s, \theta)$ 以及状态价值的估计 $\hat{v}(s_t, w)$ ，其中策略网络即上述的任务调度网络，并初始化 N_i 、 N_e 、 m 、 α 、 α^w 等参数，其中， N_i 为任务调度网络训练过程中的迭代次数， N_e 为迭代的回合次数， m 为待执行作业中观察的任务的数量， α 为参数 θ 的学习率， α^w 为 $\hat{v}(s_t, w)$ 的学习率，并向任务调度网络中输入一些相关的数据，例如待执行作业的任务信息、电价、碳排放率等。之后，会从等待任务中获取就绪任务，如上所述，就绪任务包括不存在父任务的任务，和/或，就绪任务包括父任务执行完毕的任务。针对每个就绪任务，会确定出与其对应的可执行动作（即可行动作），然后任务调度网络计算各个可执行动作被选中的概率，从而根据概率来输出更优的动作，做出任务调度决策，直到最终没有可执行动作或者更优的动作
15 为无动作，则整个任务调度过程执行完毕，此时便可获取各个时间步 t 的奖励 r_t ，并对其
进行奖励缩放，得到多个缩放奖励 r'_t ，在所有时间步的缩放奖励计算完毕后，便可根据多个时间步的缩放奖励来计算回报 G_t ，在模拟调度过程的次数达到 N_e 时，便会进行迭代优化，更新状态价值估计 $\hat{v}(s_t, w)$ 和策略网络 $\pi(a|s, \theta)$ 的参数 θ 。

其次，如图 2 所示，本发明实施例还提供了一种基于深度强化学习的任务调度网络训练装置，包括：获取单元，用于获取待执行作业，待执行作业包括多个任务；调度单元，
20 用于采用任务调度网络对多个任务进行调度；确定单元，用于根据任务调度网络的调度结果，确定多个时间步的奖励，各个时间步的奖励均包括碳奖励、电费奖励和任务延迟奖励，碳奖励表征数据中心在时间步的碳排放成本，电费奖励表征数据中心在时间步的电费成本，任务延迟奖励表征数据中心在时间步的任务延迟情况；计算单元，用于根据多个时间步的
25 奖励，计算回报 G_t ，回报 G_t 为从时间步 t 到调度过程结束的累积奖励；调整单元，用于至少根据回报 G_t ，基于梯度算法对任务调度网络的参数进行调整。

采用这种设置方式的基于深度强化学习的任务调度网络训练装置，在获取单元获取待执行作业之后，待执行作业包括多个任务；调度单元会采用任务调度网络对多个任务进行调度；然后确定单元根据任务调度网络的调度结果，确定多个时间步的奖励，各个时间步
30 的奖励均包括碳奖励、电费奖励和任务延迟奖励，碳奖励表征数据中心在时间步的碳排放成本，电费奖励表征数据中心在时间步的电费成本，任务延迟奖励表征数据中心在时间步的任务延迟情况；之后，计算单元根据多个时间步的奖励，计算回报 G_t ，回报 G_t 为从时间步 t 到调度过程结束的累积奖励；最后，调整单元至少根据回报 G_t ，基于梯度算法对任务调度网络的参数进行调整。在进行任务调度网络的深度强化训练的过程中，在调度网络执行了
35 了对多个任务的调度操作之后，会计算多个时间步的奖励，进而计算出整个调度过程的

回报 G_t ，然后根据回报 G_t 基于梯度算法（例如梯度上升或梯度下降）对任务调度网络进行参数调整。其中，每个时间步的奖励构成包括碳奖励、电费奖励和任务延迟奖励。这样，在进行任务调度网络训练的过程中，便会考虑整个任务执行过程的碳排放成本、电费成本以及任务延迟情况，使得数据中心调度过程朝着低碳、经济、低延迟的方向去优化，解决了相关技术中的数据中心运行成本高的技术问题。

在一个优选的实施例中，确定单元用于：对于任意的时间步 t ，根据公式 $r_t = -r_t^{carbon} - r_t^{cost} - r_t^{Qos}$ 计算时间步 t 的奖励；其中， r_t^{carbon} 为时间步 t 的碳奖励， r_t^{cost} 为时间步 t 的电费奖励， r_t^{Qos} 为时间步 t 的任务延迟奖励；

$$r_t^{carbon} = -P_t^{DC} \cdot e_t \cdot p^{carbon}, r_t^{cost} = -P_t^{DC} \cdot q_t, r_t^{Qos} = -\sum_{i=1}^N (T_i^{cpu} \cdot p^{cpu} + T_i^{MEM} \cdot p^{MEM}); \text{其中, } P_t^{DC} = F(M) + V_t(U_t^{cpu}, M), F(M) = M(P_{idle} + (PUE - 1) \cdot P_{peak}), V_t(U_t^{cpu}, M) = \sum_{i=1}^M (P_{peak} - P_{idle}) [2U_{i,t}^{cpu} - (U_{i,t}^{cpu})^r].$$

P_t^{DC} 为数据中心在时间步 t 的功率， e_t 为系统碳排放率， p^{carbon} 为碳排放价格， q_t 为时间步 t 的电价， N 为大于1的整数， T_i^{cpu} 为任务 i 对CPU资源的占用率， p^{cpu} 为CPU使用的价格， T_i^{MEM} 为任务 i 对内存资源的占用率， p^{MEM} 为内存资源使用的价格， M 为数据中心的服务器数量， $F(M)$ 为静态功率， $V_t(U_t^{cpu}, M)$ 为可变功率， P_{idle} 为服务器空闲时的额定功率， P_{peak} 为服务器完全占用时的额定功率，指数 r 为常数，PUE为数据中心能效指标；

计算单元包括：缩放模块，用于对多个时间步的奖励进行放大或缩小，得到多个缩放奖励，其中，多个缩放奖励所属数值区间的范围窄于多个奖励所属数值区间的范围；计算模块，用于根据多个缩放奖励，计算回报 G_t ，回报为从时间步 t 到调度过程结束的累积奖励；

缩放模块用于：对于任意的时间步 t ，根据公式 $r'_t = (r_t - \mu) / (\sigma + \epsilon)$ 计算时间步 t 的缩放奖励 r'_t ，其中， r_t 为时间步 t 的奖励， μ 通过公式 $\mu \leftarrow \mu' + (r - \mu) / k$ 随着 k 的值的增加而更新， σ 通过公式 $\sigma \leftarrow \sqrt{s / k}$ 随着 k 的值的增加而更新， s 通过公式 $s \leftarrow \sigma + (r - \mu) * (r - \mu)$ 计算更新，其中， $k = n - 1$ ， n 为时间步 t 的编号，在 $k=0$ 的情况下， $\mu = r$ ， $\sigma = r$ ， r 为第一个时间步获得的奖励，在 k 不为0的情况下， $\mu = \mu'$ ；

调整单元包括：第一确定模块，用于确定当前状态价值的估计；调整模块，用于根据回报 G_t 和当前状态价值的估计，基于梯度算法对任务调度网络的参数进行调整；

调整模块用于：采用方程 $\theta \leftarrow \theta + \alpha^{\theta} \gamma^t \delta \frac{\nabla_{\theta} \pi_{\theta}(a_t | s_t, \theta_t)}{\pi_{\theta}(a_t | s_t, \theta_t)}$ 迭代更新更新任务调度网络的参数 θ ；

其中， α^{θ} 为参数 θ 的学习率， δ 为回报 G_t 与当前状态价值的估计之间的差值， γ^t 为时间步 t 的折扣因子， $\pi_{\theta}(a_t | s_t, \theta_t)$ 为时间步 t 的 s 状态下，任务调度网络采取 a 动作的概率， $\nabla_{\theta} \pi_{\theta}(a_t | s_t, \theta_t)$ 为 $\pi_{\theta}(a_t | s_t, \theta_t)$ 对 θ 的导数；

第一确定模块用于：计算参数向量 w 与特征向量 $x(s_t)$ 的内积，得到当前状态价值的估计 $\hat{v}(s_t, w)$ ；其中， $x(s_t)$ 为时间步 t 的状态的特征向量；

参数向量 w 通过公式 $w \leftarrow w + \alpha^w [G_t - \hat{v}(s_t, w)] \nabla \hat{v}(s_t, w)$ 迭代更新；其中， α^w 为 $\hat{v}(s_t, w)$ 的学习率， $\nabla \hat{v}(s_t, w)$ 为 $\hat{v}(s_t, w)$ 对 w 的导数；

调度单元包括：第二确定模块，用于确定多个可执行的动作，动作用于指示将任意一个任务调度至任意一个服务器执行；输入模块，用于将多个可执行的动作输入任务调度网络，输出各个可执行的动作被选中的概率；第三确定模块，用于根据各个可执行的动作被选中的概率，从多个可执行的动作中确定多个调度动作；调度模块，用于根据多个调度动作，对多个任务进行调度；

第二确定模块包括：第一确定子模块，用于从待执行作业包括的多个任务中，确定一个或多个就绪任务，就绪任务包括不存在父任务的任務，和/或，就绪任务包括父任务执行完毕的任务，父任务为就绪任务执行时依赖的任务；第二确定子模块，用于针对每一个就绪任务，确定一个或多个目标服务器，目标服务器为满足就绪任务所需的计算资源的服务器；第三确定子模块，用于确定将任意一个就绪任务调度至任意一个目标服务器的动作作为一个可执行的动作；

第一确定子模块用于：按照任务排列顺序，从待执行作业包括的多个任务中，确定前预设数量的就绪任务。

另外，本发明实施例还提供了一种基于深度强化学习的数据中心任务调度方法，其包括：获取待调度作业，所述待调度作业包括多个待调度任务；采用任务调度网络对多个所述待调度任务进行调度，所述任务调度网络采用上述的基于深度强化学习的任务调度网络训练方法训练得到。由于任务调度网络采用上述的基于深度强化学习的任务调度网络训练方法训练得到，在训练过程中，基于各个时间步的碳奖励、电费奖励和任务延迟奖励计算得到的回报 G_t ，并基于回报 G_t 根据梯度算法来调整任务调度网络的参数，因此训练得到的任务调度网络便会考虑整个任务执行过程的碳排放成本、电费成本以及任务延迟情况，使得数据中心调度过程朝着低碳、经济、低延迟的方向去优化，解决了相关技术中的任务调度算法无法在保证任务处理及时性的基础上实现数据中心低碳经济运行的技术问题。

另外，本发明的实施例还提供了一种可读存储介质，其上存储有计算机指令，其中，计算机指令被处理器执行时实现上述的基于深度强化学习的任务调度网络训练方法；或者，计算机指令被处理器执行时实现上述的基于深度强化学习的数据中心任务调度方法。

最后，本发明的实施例还提供了一种终端设备，包括存储器和处理器，存储器上存储有计算机指令，其中，计算机指令被处理器执行时实现上述的基于深度强化学习的任务调度网络训练方法；或者，计算机指令被处理器执行时实现上述的基于深度强化学习的数据中心任务调度方法。

需要说明的是，本申请所涉及的用户信息（包括但不限于用户设备信息、用户个人信息等）和数据（包括但不限于用于分析的数据、存储的数据、展示的数据等），均为经用户授权或者经过各方充分授权的信息和数据，并且相关数据的收集、使用和处理需要遵守相关国家和地区的相关法律法规和标准，并提供有相应的操作入口，供用户选择授权或者

拒绝。

上述本发明实施例序号仅仅为了描述，不代表实施例的优劣。而且，在附图的流程图示出的步骤可以在诸如一组计算机可执行指令的计算机系统中执行，并且，虽然在流程图中示出了逻辑顺序，但是在某些情况下，可以以不同于此处的顺序执行所示出或描述的步骤。

在本发明的上述实施例中，对各个实施例的描述都各有侧重，某个实施例中没有详述的部分，可以参见其他实施例的相关描述。

在本申请所提供的几个实施例中，应该理解到，所揭露的技术内容，可通过其它的方式实现。其中，以上所描述的装置实施例仅仅是示意性的，例如所述单元的划分，可以为一种逻辑功能划分，实际实现时可以有另外的划分方式，例如多个单元或组件可以结合或者可以集成到另一个系统，或一些特征可以忽略，或不执行。另一点，所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口，单元或模块的间接耦合或通信连接，可以是电性或其它的形式。

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单元上。可以根据实际的需要选择其中的部分或者全部单元来实现本实施例方案的目的。

另外，在本发明各个实施例中的各功能单元可以集成在一个处理单元中，也可以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。上述集成的单元既可以采用硬件的形式实现，也可以采用软件功能单元的形式实现。

所述集成的单元如果以软件功能单元的形式实现并作为独立的产品销售或使用，可以存储在一个计算机可读取存储介质中。基于这样的理解，本发明的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的全部或部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质中，包括若干指令用以使得一台计算机设备（可为个人计算机、服务器或者网络设备等）执行本发明各个实施例所述方法的全部或部分步骤。而前述的存储介质包括：U 盘、只读存储器（ROM，Read-Only Memory）、随机存取存储器（RAM，Random Access Memory）、移动硬盘、磁碟或者光盘等各种可以存储程序代码的介质。

以上所述仅是本发明的优选实施方式，应当指出，对于本技术领域的普通技术人员来说，在不脱离本发明原理的前提下，还可以做出若干改进和润饰，这些改进和润饰也应视为本发明的保护范围。

权 利 要 求 书

1. 一种基于深度强化学习的任务调度网络训练方法，包括：

获取待执行作业，所述待执行作业包括多个任务；

采用任务调度网络对多个所述任务进行调度；

- 5 根据所述任务调度网络的调度结果，确定多个时间步的奖励，各个所述时间步的所述奖励均包括碳奖励、电费奖励和任务延迟奖励，所述碳奖励表征数据中心在所述时间步的碳排放成本，所述电费奖励表征所述数据中心在所述时间步的电费成本，所述任务延迟奖励表征所述数据中心在所述时间步的任务延迟情况；

10 根据多个时间步的所述奖励，计算回报 G_t ，所述回报 G_t 为从时间步 t 到调度过程结束的累积奖励；

至少根据所述回报 G_t ，基于梯度算法对所述任务调度网络的参数进行调整。

2. 根据权利要求 1 所述的基于深度强化学习的任务调度网络训练方法，其中，根据所述任务调度网络的调度结果，确定多个时间步的奖励包括：

15 对于任意的时间步 t ，根据公式 $r_t = -r_t^{carbon} - r_t^{cost} - r_t^{Qos}$ 计算所述时间步 t 的所述奖励；

其中， r_t^{carbon} 为所述时间步 t 的碳奖励， r_t^{cost} 为所述时间步 t 的电费奖励， r_t^{Qos} 为所述时间步 t 的任务延迟奖励。

3. 根据权利要求 2 所述的基于深度强化学习的任务调度网络训练方法，其中， $r_t^{carbon} = -P_t^{DC} \cdot e_t \cdot p^{carbon}$ ， $r_t^{cost} = -P_t^{DC} \cdot q_t$ ， $r_t^{Qos} = -\sum_{i=1}^N (T_i^{cpu} \cdot p^{cpu} + T_i^{MEM} \cdot p^{MEM})$ ；

20 其中， $P_t^{DC} = F(M) + V_t(U_t^{cpu}, M)$ ， $F(M) = M(P_{idle} + (PUE - 1) \cdot P_{peak})$ ， $V_t(U_t^{cpu}, M) = \sum_{i=1}^M (P_{peak} - P_{idle}) [2U_{i,t}^{cpu} - (U_{i,t}^{cpu})^r]$ ， P_t^{DC} 为数据中心在时间步 t 的功率， e_t 为系统碳排放率， p^{carbon} 为碳排放价格， q_t 为时间步 t 的电价， N 为大于 1 的整数， T_i^{cpu} 为任务 i 对 CPU 资源的占用率， p^{cpu} 为 CPU 使用的价格， T_i^{MEM} 为任务 i 对内存资源的占用率， p^{MEM} 为内存资源使用的价格， M 为数据中心的服务器数量， $F(M)$ 为静态功率， $V_t(U_t^{cpu}, M)$ 为可变功率， P_{idle} 为服务器空闲时的额定功率， P_{peak} 为服务器完全占用时的额定功率，指数 r 为常数，PUE 为数据中心能效指标。

4. 根据权利要求 1 所述的基于深度强化学习的任务调度网络训练方法，其中，根据多个时间步的所述奖励，计算回报 G_t 包括：

30 对多个所述时间步的所述奖励进行放大或缩小，得到多个缩放奖励，其中，多个所述缩放奖励所属数值区间的范围窄于多个所述奖励所属数值区间的范围；

根据多个所述缩放奖励，计算所述回报 G_t 。

5. 根据权利要求 4 所述的基于深度强化学习的任务调度网络训练方法，其中，对多个所述时间步的所述奖励进行放大或缩小，得到多个缩放奖励包括：

35 对于任意的时间步 t ，根据公式 $r_t' = (r_t - \mu) / (\sigma + \epsilon)$ 计算所述时间步 t 的所述缩放奖励 r_t' ，其中， r_t 为所述时间步 t 的所述奖励， μ 通过公式 $\mu \leftarrow \mu' + (r - \mu) / k$ 随着 k 的值的增加而更新， σ 通过公式 $\sigma \leftarrow \text{sqr}(s/k)$ 随着 k 的值的增加而更新， s 通过公式

$s \leftarrow \sigma + (r - \mu') * (r - \mu)$ 计算更新, 其中, $k = n - 1$, n 为所述时间步 t 的编号, 在 $k=0$ 的情况下, $\mu = r$, $\sigma = r$, r 为第一个所述时间步获得的奖励, 在 k 不为 0 的情况下, $\mu = \mu'$ 。

6. 根据权利要求 1 所述的基于深度强化学习的任务调度网络训练方法, 其中, 根据所述
5 回报 G_t , 基于梯度算法对所述任务调度网络的参数进行调整包括:

确定当前状态价值的估计;

根据所述回报 G_t 和所述当前状态价值的估计, 基于梯度算法对所述任务调度网络的参数进行调整。

7. 根据权利要求 6 所述的基于深度强化学习的任务调度网络训练方法, 其中, 根据所述
10 回报 G_t 和所述当前状态价值的估计, 基于梯度算法对所述任务调度网络的参数进行调整, 包括:

采用方程 $\theta \leftarrow \theta + \alpha^\theta \gamma^t \delta \frac{\nabla_\theta \pi_\theta(a_t | s_t, \theta_t)}{\pi_\theta(a_t | s_t, \theta_t)}$ 迭代更新所述任务调度网络的参数 θ , 其中, α^θ 为参数 θ 的学习率, δ 为所述回报 G_t 与所述当前状态价值的估计之间的差值, γ^t 为时间步 t 的折扣因子, $\pi_\theta(a_t | s_t, \theta_t)$ 为时间步 t 的 s 状态下, 所述任务调度网络采取 a
15 动作的概率, $\nabla_\theta \pi_\theta(a_t | s_t, \theta_t)$ 为 $\pi_\theta(a_t | s_t, \theta_t)$ 对 θ 的导数。

8. 根据权利要求 6 所述的基于深度强化学习的任务调度网络训练方法, 其中, 确定当前
状态价值的估计包括:

计算参数向量 w 与特征向量 $x(s_t)$ 的内积, 得到所述当前状态价值的估计 $\hat{v}(s_t, w)$;
其中, $x(s_t)$ 为时间步 t 的状态的特征向量。

9. 根据权利要求 8 所述的基于深度强化学习的任务调度网络训练方法, 其中, 所述参数
20 向量 w 通过公式 $w \leftarrow w + \alpha^w [G_t - \hat{v}(s_t, w)] \nabla \hat{v}(s_t, w)$ 迭代更新;

其中, α^w 为 $\hat{v}(s_t, w)$ 的学习率, $\nabla \hat{v}(s_t, w)$ 为 $\hat{v}(s_t, w)$ 对 w 的导数。

10. 根据权利要求 1 至 9 中任一项所述的基于深度强化学习的任务调度网络训练方法, 其
中, 采用任务调度网络对所述多个任务进行调度包括:

- 25 确定多个可执行的动作, 所述动作用于指示将任意一个所述任务调度至任意一个
服务器执行;

将多个所述可执行的动作输入所述任务调度网络, 输出各个所述可执行的动作被
选中的概率;

- 30 根据各个所述可执行的动作被选中的概率, 从多个所述可执行的动作中确定多个
调度动作;

根据多个所述调度动作, 对多个所述任务进行调度。

11. 根据权利要求 10 所述的基于深度强化学习的任务调度网络训练方法, 其中, 确定多
个可执行的动作包括:

- 35 从所述待执行作业包括的所述多个任务中, 确定一个或多个就绪任务, 所述就绪
任务包括不存在父任务的所述任务, 和/或, 所述就绪任务包括父任务执行完毕的所述

- 任务，所述父任务为所述就绪任务执行时依赖的任务；
- 针对每一个所述就绪任务，确定一个或多个目标服务器，所述目标服务器为满足所述就绪任务所需的计算资源的服务器；
- 确定将任意一个所述就绪任务调度至任意一个所述目标服务器的动作为一个所述可执行的动作。
- 5
12. 一种基于深度强化学习的数据中心任务调度方法，包括：
- 获取待调度作业，所述待调度作业包括多个待调度任务；
- 采用任务调度网络对多个所述待调度任务进行调度，所述任务调度网络采用权利要求 1 至 11 中任一项所述的基于深度强化学习的任务调度网络训练方法训练得到。
- 10 13. 一种可读存储介质，其上存储有计算机指令，其中，所述计算机指令被处理器执行时实现权利要求 1 至 11 中任一项所述的基于深度强化学习的任务调度网络训练方法；或者，所述计算机指令被处理器执行时实现权利要求 12 所述的基于深度强化学习的数据中心任务调度方法。
14. 一种终端设备，包括存储器和处理器，所述存储器上存储有计算机指令，其中，所述
- 15 计算机指令被所述处理器执行时实现权利要求 1 至 11 中任一项所述的基于深度强化学习的任务调度网络训练方法；或者，所述计算机指令被所述处理器执行时实现权利要求 12 所述的基于深度强化学习的数据中心任务调度方法。