Toward Efficient Online Scheduling for Distributed Machine Learning Systems

Abstract

- 考虑资源分配和PS和worker位置的分析模型
- 将PS和Worker转化为一个混合填充覆盖整数优化问题
- 提出随机舍入的近似算法来解决该优化问题

Related Work

1. 现有云系统的任务调度算法并不适用于ML

   因为MapReduce是单向流量，而ML的流量之间存在数据依赖，完成程度依赖于ML作业的收敛程度。

2. 资源调度问题

   - 静态调度

     worker和PS的数量是固定的，无法应对不同的资源需求

   - 基于深度强化学习的调度

     通过设置并行级别和执行顺序，启发式地训练基于图的并行任务调度策略

   - 在线原始对偶近似算法

     先前的研究中PS和worker严格隔离在两台机器上，简化了模型。如果可以将worker和PS放置在同一位置，可以避免高昂的通信开销

     同时由于任务调度的在线性，无法预测任务到达的时间，提升了调度的难度。

Modeling

采用BSP，因此下一次迭代前，每个Worker都是同步的

label	description
I/T	任务数/系统时间跨度
$\tilde{t}_i/a_i$	任务i完成的时间/任务i抵达的时间
$\mu_i(\cdot)/K_i$	任务i/任务i中的样本数
R/H	样本数量/机器数量
E_i/F_i	迭代次数/任务i的全局批处理大小
x_i	任务i是否被接受
C_h^r	服务器h上类型r的资源容量
α_i^r	任务i中一个worker需要种类r资源的数量
β_r^i	任务i种一个PS需要种类r资源的数量
$w_{ih}[t]$	t时刻Server h上任务i的worker的数量
$s_{if}[t]$	t时刻Server h上任务i的PS的数量
$b_i(h,p)$	Server h和p之间的带宽，如果h和p相同，则是总线带宽，不同则是链路带宽
τ_i	训练任务i一个样本的时间
g_i	任务i的梯度和参数量
$W_i[t]$	t时刻包含运行任务i的worker的物理机集合
$P_i[t]$	t时刻包含运行任务i的PS的物理机集合
x_{π_i}	任务i是否选择分配策略π
$\tilde{t}_{\pi_i}$	任务i使用分配策略π_i时任务完成时间

$w_{ht}^{\pi_i}$	分配策略中任务i的worker的数量
$s_{ht}^{\pi_i}$	分配策略中任务i的PS的数量
Π_i	任务i的可行分配策略
γ_i	任务i中Worker和PS的比例
$\rho_h^r[t]$	t时刻分配资源类型r给机器h
$Q_h^r(\cdot)$	机器h上资源类型r的价格函数

Modeling of learning jobs

处理一个样本的时间开销：

$$\underbrace{\tau_i}_{\substack{\text{Training time}\\\text{per sample}}} + \underbrace{\left(\frac{2g_i/\sum_{h'\in\mathcal{H}} s_{ih'}[t]}{\min_{p\in\mathcal{P}_i[t]} b_i(h,p)}\right)\Big/\left(\frac{F_i}{\sum_{h'\in\mathcal{H}} w_{ih'}[t]}\right)}_{\text{Communication time per sample}}.$$

worker和PS的比例，一般为1：1： $\gamma_i \triangleq \frac{\sum_{h'\in\mathcal{H}} w_{ih'}[t]}{\sum_{h'\in\mathcal{H}} s_{ih'}[t]}, \quad \forall i, t.$

得到每个时隙t的每个worker训练的样本数： $\frac{w_{ih}[t]}{\tau_i + \frac{\gamma_i}{F_i}\frac{2g_i}{\min_{p\in\mathcal{P}_i[t],h'\in\mathcal{W}_i[t]} b_i(h',p)}}.$

设K_i是样本数量，那么$K_i \gg F_i$：

保证任务i能够被训练完成，需要保证：

$$\sum_{t\in\mathcal{T}}\sum_{h\in\mathcal{H}} \frac{w_{ih}[t]}{\tau_i + \frac{\gamma_i}{F_i}\frac{2g_i}{\min_{p\in\mathcal{P}_i[t],h'\in\mathcal{W}_i[t]} b_i(h',p)}} \geq x_i E_i K_i, \forall i \in \mathcal{I}. \quad (3)$$

同时为了防止在训练过程中出现闲置资源，一般有：

$$\sum_{h\in\mathcal{H}} w_{ih}[t] \leq x_i F_i, \quad \forall i \in \mathcal{I}, a_i \leq t \leq T. \quad (4)$$

Resource Constraint Modeling

1. 资源的种类分为：CPU、GPU等

   为了保证资源足够，有以下约束：

   $$\sum_{i \in \mathcal{I}} (\alpha_i^r w_{ih}[t] + \beta_i^r s_{ih}[t]) \leq C_h^r, \forall t \in \mathcal{T}, r \in \mathcal{R}, h \in \mathcal{H}. \quad (5)$$

2. 任务完成时，可能会有某几个worker还未彻底结束

   $$\tilde{t}_i = \arg\max_{t \in \mathcal{T}} \left\{ \sum_{h \in \mathcal{H}} w_{ih}[t] > 0 \right\}, \quad \forall i \in \mathcal{I}. \quad (6)$$

3. 在任务到达前，分配的资源为0：

   $$w_{ih}[t] = s_{ih}[t] = 0, \quad \forall i \in \mathcal{I}, h \in \mathcal{H}, t < a_i. \quad (7)$$

Object Function

建立关于任务i的效用函数$u_i(\tilde{t}_i - a_i)$,

定义DML resource scheduling问题：

$$\textbf{DMLRS:} \ \text{Maximize}_{\mathbf{x},\mathbf{w},\mathbf{s}} \sum_{i \in \mathcal{I}} x_i u_i(\tilde{t}_i - a_i)$$
$$\text{subject to Constraints (3) – (7).}$$

约束(3)和(6)存在非确定性，并且$\{\forall i, a_i\}$未知，因此算法是强制在线的

Online Scheduling Algorithm Design

为了解决非确定因素带来的问题，需要重新改写约束条件

定义$\pi i \in \Pi i$，Πi表示对于任务i，每个能够满足(3)(4)约束的可行分配；π_i表示分配的worker和PS数量，即$\pi_i = \{w_{ht}^{\pi_i}, s_{ht}^{\pi i}, \forall t \in T, h \in H\}$，是一个定值

R-DMLRS:

$$\underset{\mathbf{x}}{\text{Maximize}} \sum_{i\in\mathcal{I}} \sum_{\pi_i\in\Pi_i} x_{\pi_i} u_i(\tilde{t}_{\pi_i} - a_i)$$

$$\text{subject to} \sum_{i\in\mathcal{I}} \sum_{\pi_i\in\Gamma(t,h)} (\alpha_i^r w_{ht}^{\pi_i} + \beta_i^r s_{ht}^{\pi_i}) x_{\pi_i} \le C_h^r, \quad \forall t\in\mathcal{T}, r\in\mathcal{R}, h\in\mathcal{H}, \tag{8}$$

$$\sum_{\pi_i\in\Pi_i} x_{\pi_i} \le 1, \quad \forall i\in\mathcal{I}, \tag{9}$$

$$x_{\pi_i}\in\{0,1\}, \quad \forall i\in\mathcal{I}, \pi_i\in\Pi_i,$$

通过引入可行解的集合，可以消去不确定性约束，但是由于可行解的空间是指数级，所以$R-DMLRS$仍然是NPC的。

An Online Primal-Dual Framework for R-DMLRS

为了解决可行解空间的指数级

D-R-DMLRS:

$$\underset{\boldsymbol{\lambda},\mathbf{p}}{\text{Minimize}} \quad \sum_{i\in\mathcal{I}} \lambda_i + \sum_{t\in\mathcal{T}} \sum_{h\in\mathcal{H}} \sum_{r\in\mathcal{R}} p_h^r[t] C_h^r \tag{10}$$

$$\text{subject to} \quad \lambda_i \ge u_i(\tilde{t}_{\pi_i} - a_i) - \sum_{t\in\mathcal{T}(\pi_i)} \sum_{h\in\mathcal{H}(\pi_i[t])} \sum_{r\in\mathcal{R}} (\alpha_i^r w_{ht}^{\pi_i} + \beta_i^r s_{ht}^{\pi_i}) p_h^r[t], \quad \forall i\in\mathcal{I}, \pi_i\in\Pi_i, \tag{11}$$

$$p_h^r[t] \ge 0, \quad \forall t\in\mathcal{T}, h\in\mathcal{H}, r\in\mathcal{R},$$

$$\lambda_i \ge 0, \quad \forall i\in\mathcal{I},$$

其中$p_h^r[t]$是t时刻资源类型r的价格

为了最小化(10)，即最小化资源的费用，那么就要降低(11)，当λ_i下降到一定程度时，可以得到下式

$$\lambda_i^* = u_i(\tilde{t}_{\pi_i^*} - a_i) - \sum_{t\in\mathcal{T}(\pi_i^*)} \sum_{h\in\mathcal{H}(\pi_i^*[t])} \sum_{r\in\mathcal{R}} (\alpha_i^r w_{ht}^{\pi_i^*} + \beta_i^r s_{ht}^{\pi_i^*}) p_h^{r*}[t].$$

相当于找到一个π_i^* 和对偶价格$p_h^{r*}[t]$最大化(11)的右边

Algorithm 1: Primal-Dual Online Resource Scheduling.

Initialization:

1. Let $w_{ih}[t]=0$, $s_{ih}[t]=0$, $\forall i,t,h$. Let $\rho_h^r[t]=0$, $\forall h,r,t$. Choose some appropriate initial values for $p_h^r[0]$.

Main Loop:

2. Upon the arrival of job i, determine a schedule π_i^* to maximize the RHS of (11) and its corresponding payoff λ_i using **Algorithm 2** (*to be specified*).
3. If $\lambda_i>0$, set $x_i=1$. Set $w_{ih}[t]$ and $s_{ih}[t]$ according to schedule π_i^*, $\forall t\in\mathcal{T}(\pi_i^*)$, $h\in\mathcal{H}(\pi_i^*[t])$.
   - Update $\rho_h^r[t]\leftarrow\rho_h^r[t]+\alpha_i^r w_{ih}[t]+\beta_i^r s_{ih}[t]$, $\forall t\in\mathcal{T}(\pi_i^*)$, $h\in\mathcal{H}(\pi_i^*[t])$, $r\in\mathcal{R}$.
   - Update $p_h^r[t]=Q_h^r(\rho_h^r[t])$, $\forall t\in\mathcal{T}(\pi_i^*)$, $h\in\mathcal{H}(\pi_i^*[t])$, $r\in\mathcal{R}$. Schedule job i based on π_i^* and go to Step 2.
4. If $\lambda_i\le 0$, set $x_i=0$ and reject job i and go to Step 2.

利用KKT的互补松弛条件，如果对偶变量$\lambda_i>0$时，原始的约束(9)就是紧致的；如果$\lambda_i=0$，那么约束(11)就是<0的，在最有调度π^*下的效用较低，因此需要拒绝任务i。

$Q_h^r(\cdot)$的定义：

$$Q_h^r(\rho_h^r[t])=L(U^r/L)^{\frac{\rho_h^r[t]}{C_h^r}}, \tag{12}$$

其中

$$U^r\triangleq\max_{i\in\mathcal{I}}\frac{u_i(\lceil\frac{E_iK_i}{F_i}(\tau_i+2g_i\gamma_i/(b_i^{(i)}F_i))\rceil-a_i)}{\alpha_i^r+\beta_i^r},\forall r\in\mathcal{R}, \tag{13}$$

$$L\triangleq\min_{i\in\mathcal{I}}\frac{1/(2\mu)u_i(T-a_i)}{\sum_{r\in\mathcal{R}}\lceil E_iK_i(\tau_i+2g_i\gamma_i/(b_i^{(e)}F_i)\rceil(\alpha_i^r+\beta_i^r)}. \tag{14}$$

U^r是最大的单位资源任务效用放置资源类型r的worker和PS所能达到的最大效用，$u_i(\lceil\frac{E_iK_i}{F_i}(\tau_i+2g_i\gamma_i/(b_i^{(i)}F_i))\rceil-a_i)$是任务i通过最大数量的worker和PS所能达到的最大效用，同时使得完成时间尽可能早

L是所有任务中最小单位时间、单位资源的效用

Determining Schedule π_i^* in Step 2 of Algorithm 1

首先，对于任务i，每个调度都会有一个唯一的完成时间$\tilde{t}_i$，因此寻找最大效用对应的π_i^*：

$$\operatorname*{Max}_{\tilde{t}_i} \left\{ \begin{array}{l} \operatorname*{Max}_{\mathbf{w},\mathbf{s}} \ u_i(\tilde{t}_i - a_i) - \sum_{t\in\mathcal{T}} \sum_{h\in\mathcal{H}} \sum_{r\in\mathcal{R}} p_h^r[t] \\ \qquad\qquad \times(\alpha_i^r w_{ih}[t] + \beta_i^r s_{ih}[t]) \\ \text{s.t.} \ \ \alpha_i^r w_{ih}[t] + \beta_i^r s_{ih}[t] \le \hat{C}_h^r[t], \\ \qquad\qquad \forall t \in \mathcal{T}, r \in \mathcal{R}, h \in \mathcal{H}, \\ \quad \text{Constraints (3)(4)(7) for } x_i = 1, \end{array} \right\}, \quad (15)$$

其中$\hat{C}_h^r[t] = C_h^r[t] - \rho_h^r[t]$

给定$\tilde{t}_i$时，$u(\tilde{t}_i - a_i)$就是定值了，所以问题可以简化成

$$\operatorname*{Minimize}_{\mathbf{w},\mathbf{s}} \ \sum_{t\in[a_i,\tilde{t}_i]} \sum_{h\in\mathcal{H}} \sum_{r\in\mathcal{R}} p_h^r[t](\alpha_i^r w_{ih}[t] + \beta_i^r s_{ih}[t]) \quad (16)$$

$$\text{subject to} \ \sum_{t\in[a_i,\tilde{t}_i]} \sum_{h\in\mathcal{H}} \frac{w_{ih}[t]}{\tau_i + \frac{\gamma_i}{F_i} \frac{2g_i}{\min_{p\in\mathcal{P}_i[t], h'\in\mathcal{W}_i[t]} b_i(h',p)}} \ge V_i, \quad (17)$$

$$\alpha_i^r w_{ih}[t] + \beta_i^r s_{ih}[t] \le \hat{C}_h^r[t], \forall r, h, \forall t \in [a_i, \tilde{t}_i], \quad (18)$$

$$\text{Constraint (4) for all } t \in [a_i, \tilde{t}_i],$$

其中$V_i = E_i * K_i$代表训练的样本总数。(16)中唯一的耦合约束是(17)，因此可以用动态规划来解决问题(16)

将问题改写成每个时隙的形式，

$$\operatorname*{Minimize}_{w_{ih}[t], s_{ih}[t], \forall h} \ \sum_{h\in\mathcal{H}} \sum_{r\in\mathcal{R}} p_h^r[t](\alpha_i^r w_{ih}[t] + \beta_i^r s_{ih}[t]) \quad (19)$$

$$\text{subject to} \ \sum_{h\in\mathcal{H}} \frac{w_{ih}[t]}{\tau_i + \frac{\gamma_i}{F_i} \frac{2g_i}{\min_{p\in\mathcal{P}_i[t], h'\in\mathcal{W}_i[t]} b_i(h',p)}} \ge V_i[t], \quad (20)$$

$$\text{Constraints (4)(18) for the given } t.$$

$\Theta(\tilde{t}_i, V_i)$是问题(16)的最优值，$\theta(t, V_i[t])$是问题(19)的最优值，那么问题(16)可以通过以下动态规划解决；

$$\Theta(\tilde{t}_i, V_i) = \min_{v \in [0, V_i]} \left\{ \theta(\tilde{t}_i, v) + \Theta(\tilde{t}_i - 1, V_i - v) \right\}. \quad (21)$$

通过枚举$[0, E_i K_i]$找到在时隙$\tilde{t}$中完成的最佳样本数，即训练负载

Algorithm 2: Determine π_i^* in Step 2 of Algorithm 1.

Initialization:

1. Let $\tilde{t}_i = a_i$. Let $\lambda_i = 0$, $\pi_i^* = \varnothing$, $w_{ih}[t] = s_{ih}[t] = 0$, $\forall i, t, h$.

Main Loop:

2. Compute $\Theta(\tilde{t}_i, V_i)$ in (21) using **Algorithm 3**. Denote the resulted schedule as π_i. Let $\lambda_i' = u_i(\tilde{t}_i - a_i) - \Theta(\tilde{t}_i, V_i)$. If $\lambda_i' > \lambda_i$, let $\lambda_i \leftarrow \lambda_i'$ and $\pi_i^* \leftarrow \pi_i$.
3. Let $\tilde{t}_i \leftarrow \tilde{t}_i + 1$. If $\tilde{t}_i > T$, stop; otherwise, go to Step 2.

Algorithm 3: Solving $\Theta(\tilde{t}_i, V_i)$ by Dynamic Programming.

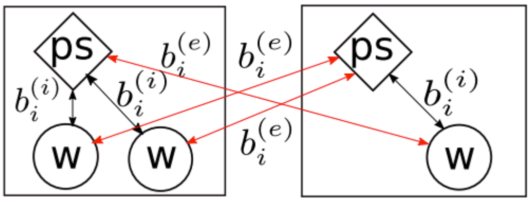
Initialization:

1. Let *cost-min* = ∞, $\pi_i = \varnothing$, and $v = 0$.

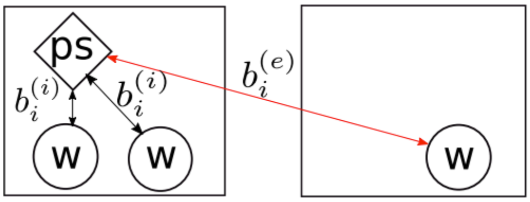
Main Loop:

2. Compute $\theta(\tilde{t}_i, v)$ using **Algorithm 4** (*to be specified*). Denote the resulted cost and schedule as *cost-v* and $\hat{\pi}_i$.
3. Compute $\Theta(\tilde{t}_i - 1, V_i - v)$ by calling **Algorithm 3** itself. Denote the resulted cost and schedule as *cost-rest* and $\tilde{\pi}_i$.
4. If *cost-min* $>$ *cost-v* $+$ *cost-rest* then *cost-min* $=$ *cost-v* $+$ *cost-rest* and let $\pi_i \leftarrow \hat{\pi}_i \cup \tilde{\pi}_i$.
5. Let $v \leftarrow v + 1$. If $v > V_i$ stop; otherwise go to Step 2.

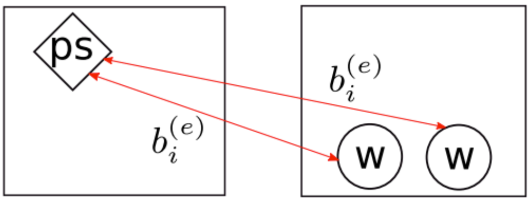
但是其中$\theta(t, v)$还是未知的，并且受到(20)中的非确定约束。因此下面进行分类讨论：



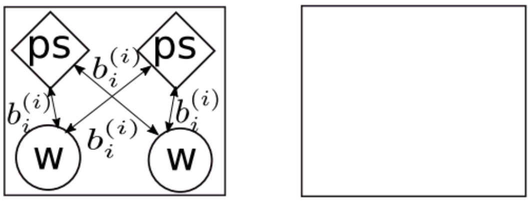
(a) $|\mathcal{P}_i[t]| \neq 1$.



(b) $|\mathcal{P}_i[t]| = 1, |\mathcal{W}_i[t]| \neq 1$.



(c) $|\mathcal{P}_i[t]|{=}|\mathcal{W}_i[t]|{=}1, \mathcal{P}_i[t]{\neq}\mathcal{W}_i[t]$.



(d) $|\mathcal{P}_i[t]|{=}|\mathcal{W}_i[t]|{=}1, \mathcal{P}_i[t]{=}\mathcal{W}_i[t]$.

只有当worker和Server在单机时，选取$b^{(i)}$，否则都是选取$b^{(e)}$

> **Fact 1.** *The function* $(2g_i/\min_{p\in\mathcal{P}_i[t],h\in\mathcal{W}_i[t]} b_i(h,p)) = 2g_i/b_i^{(i)}$ *if and only if* $|\mathcal{P}_i[t]| = |\mathcal{W}_i[t]| = 1$ *and* $\mathcal{P}_i[t] = \mathcal{W}_i[t]$; *otherwise,* $(2g_i/\min_{p\in\mathcal{P}_i[t],h\in\mathcal{W}_i[t]} b_i(h,p)) = 2g_i/b_i^{(e)}$.

情况1： $b^{(i)}$

因为问题退化成单机问题，所以问题可以简化成如下：

$$\sum_{r\in\mathcal{R}} \left\{ \begin{array}{ll} \text{Min} & p_h^r[t]s_{ih}[t](\alpha_i^r\gamma_i + \beta_i^r) \\ \text{s.t.} & s_{ih}[t](\alpha_i^r\gamma_i + \beta_i^r) \leq \hat{C}_h^r[t], \\ & \text{Constraint (4) for given } r, h \end{array} \right\}, \qquad (22)$$

上式中有一个平凡解$\forall h \in H, w_{ih}[t] = s_{ih}[t] = 0$，但这明显违反了工作负载约束(20)。因此最优值应该在$h \in H, w_{ih}[t] > 0 \& s_{ih}[t] > 0$中取到。

> $$\underset{w_{ih}[t],s_{ih}[t],\forall h}{\text{Minimize}} \quad \sum_{h\in\mathcal{H}}\sum_{r\in\mathcal{R}} p_h^r[t](\alpha_i^r w_{ih}[t] + \beta_i^r s_{ih}[t]) \qquad (19)$$
>
> $$\text{subject to} \quad \sum_{h\in\mathcal{H}} \frac{w_{ih}[t]}{\tau_i + \frac{\gamma_i}{F_i}\frac{2g_i}{\min_{p\in\mathcal{P}_i[t],h'\in\mathcal{W}_i[t]} b_i(h',p)}} \geq V_i[t], \qquad (20)$$
>
> Constraints (4)(18) for the given t.

同时约束(20)简化为$\gamma_i s_{ih}[t] \geq V_i[t](\tau_i + \frac{2g_i\gamma_i}{b_i^{(i)}F_i})$，因此可以：根据$\sum_{r\in\mathcal{R}} p_h^r[t](\alpha_i^r\gamma_i + \beta_i^r)$对每台物理机进行排序，并根据计算出$s_{ih}[t] = V_i[t](\tau_i + \frac{2g_i\gamma_i}{b_i^{(i)}F_i})/\gamma_i$，就可以得到PS的数量。最后检查资源容量约束(18)和不出现闲置资源约束(4)

情况2： $b^{(e)}$

不满足情况1的时候，最低带宽选取为$b^{(e)}$

约束(20)简化为$\sum_{h\in H} w_{ih}[t] \geq V_i[t](\tau_i + \frac{2g_i\gamma_i}{b_i^{(e)}F_i})$

$$\underset{w_{ih}[t],s_{ih}[t],\forall h}{\text{Minimize}} \quad \sum_{h\in\mathcal{H}} p_h^w[t]w_{ih}[t] + p_h^s[t]s_{ih}[t] \tag{23}$$

$$\text{subject to} \quad \alpha_i^r w_{ih}[t] + \beta_i^r s_{ih}[t] \leq \hat{C}_h^r[t],\ \forall h, r, \tag{24}$$

$$\sum\nolimits_{h\in\mathcal{H}} w_{ih}[t] \leq F_i, \tag{25}$$

$$\sum\nolimits_{h\in\mathcal{H}} w_{ih}[t] \geq V_i[t]\left(\tau_i + \frac{2g_i\gamma_i}{b_i^{(e)}F_i}\right), \tag{26}$$

其中$p_h^w[t] = \sum_{r\in R} p_h^r[t]\alpha_i^r, p_h^s[t] = \sum_{r\in R} p_h^r[t]\beta_i^r$，表示为在时间$t$中机器$h$分配worker 和PS所有资源的费用总和。

但是问题(23)也是一个NP问题，难于求解

> In what follows, we will pursue an instance-dependent constant ratio approximation scheme to solve Problem (23) in this paper.To this end, we propose a randomized rounding scheme: First, we solve the linear programming relaxation of Problem (23).

设$\{\bar{w}_{ih}[t], \bar{s}_{ih}[t], \forall h, t\}$是最优值，引入$\delta \in (0,1]$，通过随机四舍五入，$w'_{ih}[t] = G_\delta * \bar{w}_{ih}[t], s'_{ih}[t] = G_\delta \bar{s}_{ih}[t]$，求解得到$w_{ih}[t]$和$s_{ih}[t]$

$$w_{ih}[t] = \begin{cases} \lceil w'_{ih}[t] \rceil, \text{ with probability } w'_{ih}[t] - \lfloor w'_{ih}[t] \rfloor, \\ \lfloor w'_{ih}[t] \rfloor, \text{ with probability } \lceil w'_{ih}[t] \rceil - w'_{ih}[t], \end{cases} \tag{27}$$

$$s_{ih}[t] = \begin{cases} \lceil s'_{ih}[t] \rceil, \text{ with probability } s'_{ih}[t] - \lfloor s'_{ih}[t] \rfloor, \\ \lfloor s'_{ih}[t] \rfloor, \text{ with probability } \lceil s'_{ih}[t] \rceil - s'_{ih}[t]. \end{cases} \tag{28}$$

得到求解$\theta(t, v)$的算法：

Algorithm 4: Solving $\theta(t, v)$ (i.e., Problem (19)).

Initialization:

1. Let $w_{ih}[t]=s_{ih}[t]=0, \forall h$. Let $h=1$. Pick some $\delta \in (0,1]$. Let G_δ be defined as in Eq. (31) or Eqn (32). Let $D = \lceil v(\tau_i + 2g_i\gamma_i/(b_i^{(i)} F_i))\rceil$. Let $h^* = \varnothing$. Let *cost-min*$= \infty$. Choose some integer $S \geq 1$. Let $iter \leftarrow 1$.

Handling Internal Communication:

2. Sort machines in $\mathcal{H}$ according to $\sum_{r\in\mathcal{R}} p_h^r[t](\alpha_i^r\gamma_i + \beta_i^r)$ in non-decreasing order into $h_1, h_2, ..., h_H$.
3. Calculate the minimum number of $s_{ih}[t] = V_i[t]\Big(\tau_i + \frac{2g_i\gamma_i}{b_i^{(i)} F_i}\Big)/\gamma_i$.
4. If Constraint (4) is not satisfied, go to Step 7.
5. If Constraint (24) is not satisfied, go to Step 7.
6. Return *cost-min*$\sum_{r\in\mathcal{R}} p_h^r[t]s_{ih}[t](\alpha_i^r\gamma_i + \beta_i^r)$ and $h^* = h$.
7. Let $h \leftarrow h+1$. If $h > H$, stop; otherwise, go to Step 2.

Handling External Communication:

8. Solve the linear programming relaxation of Problem (23). Let $\{\bar{w}_{ih}[t], \bar{s}_{ih}[t], \forall h, t\}$ be the fractional optimal solution.
9. Let $w'_{ih}[t] = G_\delta \bar{w}_{ih}[t], s'_{ih}[t] = G_\delta \bar{s}_{ih}[t], \forall h, t$.
10. Generate an integer solution $\{w_{ih}[t], s_{ih}[t], \forall h, t\}$ following the randomized rounding scheme in (27)–(28).
11. If $\{w_{ih}[t], s_{ih}[t], \forall h, t\}$ is infeasible or $iter < S$, then $iter \leftarrow iter + 1$, go to Step 10.

Final Step:

12. Compare the solutions between internal and external cases. Pick the one with the lowest cost among them and return the cost and the corresponding schedule $\{w_{ih}[t], s_{ih}[t], \forall h, t\}$.

NUMERICAL RESULTS

1. 任务的大小：

	取值大小
迭代次数E_i	[50,200]
任务样本数K_i	[20000,500000]
参数大小g_i	$[30, 575]$MB
样本处理时间τ_i	$[10^{-5}, 10^{-4}]$
worker和PS数量比γ_i	$[1, 10]$
任务的global batch sizeF_i	[1,200]

2. worker和PS的资源需求：

	Worker	PS
GPU	0-4	None
CPU	1-10	1-10
Memory	2-32GB	2-32GB
Storage	5-10GB	5-10GB

3. 效用函数

   选取sigmoid效用函数：$u(t - a_i) = \theta_1/(1 + e^{\theta_2(t-a_i-\theta_3)})$

   $\theta_1 \in [1, 100]$，表示任务的优先级

   $\theta_2 \in [0.01, 1]$，表示任务对时间的敏感程度

   $\theta_3 \in [1, 15]$，表示估计完成时间

---

1. 将PD-ORS与FIFO、DRF和Dorm从主机数量、任务数量

> FIFO：根据任务到达时间分配资源
>
> DRF：根据当前拥有的最大资源进行资源分配

> Dorm：根据MILP资源利用率最大化问题来分配资源。

Fig. 6中时间跨度=20，任务数=50

Fig. 7中时间跨度=20，主机数=100

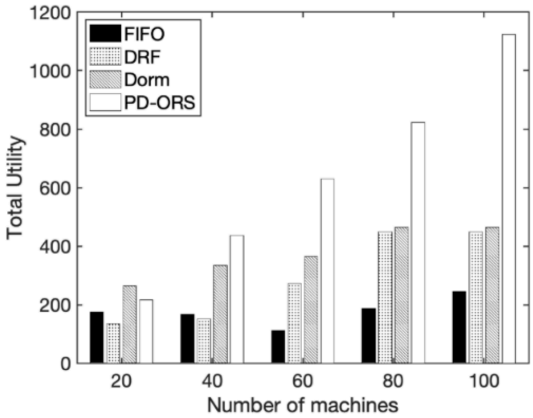


Fig. 6: PD-ORS vs. baselines (growing machine number).

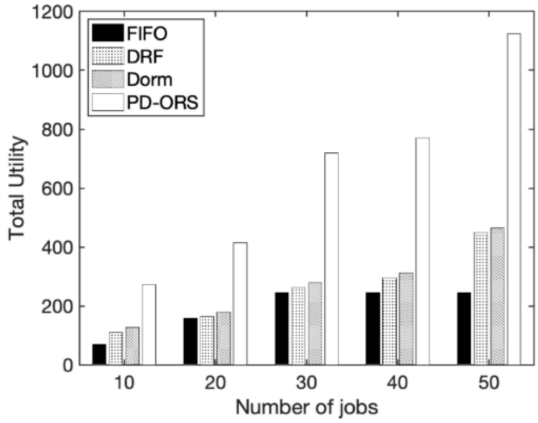


Fig. 7: PD-ORS vs. baselines (growing job number).

2. 与算法OASiS比较

   Fig. 8 主机数=50，时间跨度=20

   Fig 9 任务数=100，时间跨度=80

   OASiS算法中Worker和PS是严格隔离在不同的主机上。因此PD-ORS允许在同一主机上设置Worker和PS，可以增加效用。

Fig. 8: PD-ORS vs. OASiS (growing job number).

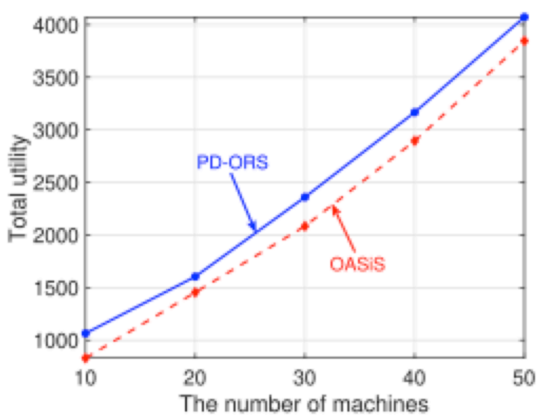


Fig. 9: PD-ORS vs. OASiS (growing machine number).

3. 模型实际训练时间

Fig. 10 主机数=30，时间跨度=80，任务数=100

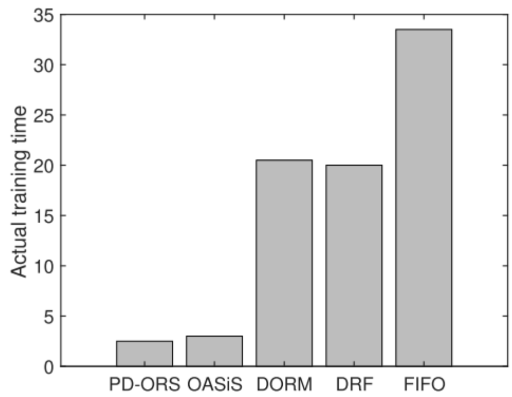


Fig. 10: Median of actual training time comparison.

Time Complexity Analysis

总的时间复杂度为$O(\sum_{i=1}^{I} TK_i^2E_i^2(H^3+S))$

其中

Algorithm 3：$O(TK_i^2E_i^2)$

Algorithm 4：内部通信：$O(HlogH)$ 外部通信：$O(H^3)$

> **Algorithm 1:** Primal-Dual Online Resource Scheduling.
>
> **Initialization:**
>
> 1. Let $w_{ih}[t]=0$, $s_{ih}[t]=0$, $\forall i,t,h$. Let $\rho_h^r[t]=0$, $\forall h,r,t$. Choose some appropriate initial values for $p_h^r[0]$.
>
> **Main Loop:**
>
> 2. Upon the arrival of job i, determine a schedule π_i^* to maximize the RHS of (11) and its corresponding payoff λ_i using **Algorithm 2** (*to be specified*).
> 3. If $\lambda_i>0$, set $x_i=1$. Set $w_{ih}[t]$ and $s_{ih}[t]$ according to schedule π_i^*, $\forall t\in\mathcal{T}(\pi_i^*)$, $h\in\mathcal{H}(\pi_i^*[t])$.
>   - Update $\rho_h^r[t]\leftarrow\rho_h^r[t]+\alpha_i^r w_{ih}[t]+\beta_i^r s_{ih}[t]$, $\forall t\in\mathcal{T}(\pi_i^*)$, $h\in\mathcal{H}(\pi_i^*[t])$, $r\in\mathcal{R}$.
>   - Update $p_h^r[t]=Q_h^r(\rho_h^r[t])$, $\forall t\in\mathcal{T}(\pi_i^*)$, $h\in\mathcal{H}(\pi_i^*[t])$, $r\in\mathcal{R}$. Schedule job i based on π_i^* and go to Step 2.
> 4. If $\lambda_i\leq 0$, set $x_i=0$ and reject job i and go to Step 2.

Algorithm 2: Determine π_i^* in Step 2 of Algorithm 1.

Initialization:

1. Let $\tilde{t}_i = a_i$. Let $\lambda_i = 0$, $\pi_i^* = \varnothing$, $w_{ih}[t] = s_{ih}[t] = 0$, $\forall i, t, h$.

Main Loop:

2. Compute $\Theta(\tilde{t}_i, V_i)$ in (21) using **Algorithm 3**. Denote the resulted schedule as π_i. Let $\lambda_i' = u_i(\tilde{t}_i - a_i) - \Theta(\tilde{t}_i, V_i)$. If $\lambda_i' > \lambda_i$, let $\lambda_i \leftarrow \lambda_i'$ and $\pi_i^* \leftarrow \pi_i$.
3. Let $\tilde{t}_i \leftarrow \tilde{t}_i + 1$. If $\tilde{t}_i > T$, stop; otherwise, go to Step 2.

Algorithm 3: Solving $\Theta(\tilde{t}_i, V_i)$ by Dynamic Programming.

Initialization:

1. Let *cost-min* = ∞, $\pi_i = \varnothing$, and $v = 0$.

Main Loop:

2. Compute $\theta(\tilde{t}_i, v)$ using **Algorithm 4** (*to be specified*). Denote the resulted cost and schedule as *cost-v* and $\hat{\pi}_i$.
3. Compute $\Theta(\tilde{t}_i - 1, V_i - v)$ by calling **Algorithm 3** itself. Denote the resulted cost and schedule as *cost-rest* and $\tilde{\pi}_i$.
4. If *cost-min* $>$ *cost-v* $+$ *cost-rest* then *cost-min* $=$ *cost-v* $+$ *cost-rest* and let $\pi_i \leftarrow \hat{\pi}_i \cup \tilde{\pi}_i$.
5. Let $v \leftarrow v + 1$. If $v > V_i$ stop; otherwise go to Step 2.

Algorithm 4: Solving $\theta(t, v)$ (i.e., Problem (19)).

Initialization:

1. Let $w_{ih}[t] = s_{ih}[t] = 0, \forall h$. Let $h = 1$. Pick some $\delta \in (0, 1]$. Let G_δ be defined as in Eq. (31) or Eqn (32). Let $D = \lceil v(\tau_i + 2g_i\gamma_i/(b_i^{(i)}F_i)) \rceil$. Let $h^* = \varnothing$. Let *cost-min*$= \infty$. Choose some integer $S \geq 1$. Let $iter \leftarrow 1$.

Handling Internal Communication:

2. Sort machines in $\mathcal{H}$ according to $\sum_{r\in\mathcal{R}} p_h^r[t](\alpha_i^r\gamma_i + \beta_i^r)$ in non-decreasing order into $h_1, h_2, ..., h_H$.
3. Calculate the minimum number of $s_{ih}[t] = V_i[t]\Big(\tau_i + \frac{2g_i\gamma_i}{b_i^{(i)}F_i}\Big)/\gamma_i$.
4. If Constraint (4) is not satisfied, go to Step 7.
5. If Constraint (24) is not satisfied, go to Step 7.
6. Return *cost-min*$\sum_{r\in\mathcal{R}} p_h^r[t]s_{ih}[t](\alpha_i^r\gamma_i + \beta_i^r)$ and $h^* = h$.
7. Let $h \leftarrow h + 1$. If $h > H$, stop; otherwise, go to Step 2.

Handling External Communication:

8. Solve the linear programming relaxation of Problem (23). Let $\{\bar{w}_{ih}[t], \bar{s}_{ih}[t], \forall h, t\}$ be the fractional optimal solution.
9. Let $w'_{ih}[t] = G_\delta \bar{w}_{ih}[t], s'_{ih}[t] = G_\delta \bar{s}_{ih}[t], \forall h, t$.
10. Generate an integer solution $\{w_{ih}[t], s_{ih}[t], \forall h, t\}$ following the randomized rounding scheme in (27)–(28).
11. If $\{w_{ih}[t], s_{ih}[t], \forall h, t\}$ is infeasible or $iter < S$, then $iter \leftarrow iter + 1$, go to Step 10.

Final Step:

12. Compare the solutions between internal and external cases. Pick the one with the lowest cost among them and return the cost and the corresponding schedule $\{w_{ih}[t], s_{ih}[t], \forall h, t\}$.

CONCLUSION

1. 文章使用数学语言将服务器的资源分配建模成一个在线的优化问题，并给出了近似求解。

2. 资源分配中PS和Worker可以在分配在同一主机中，降低一部分通信开销。
3. 模型中还是做了一部分的简化，假定每个任务的完成时间都是不同的。
4. 没有考虑通信的可靠性。