Learning to Schedule Multi-Server Jobs with Fluctuated Processing Speeds

Hailiang ZHAO @ ZJU-CS

http://hliangzhao.me

May 14, 2023

2023 CCF 16th International Conference on Service Science (ICSS)

Outline

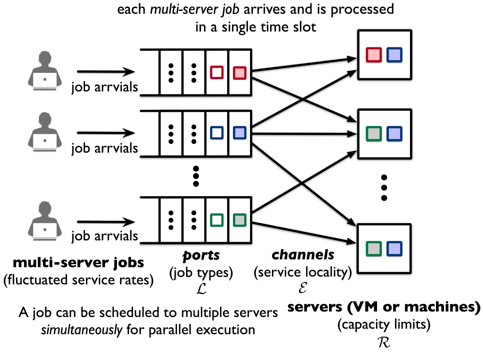
Non-Clairvoyant Online Job Scheduling

It is difficult for the cluster scheduler to allocate an appropriate number of computing devices to each multi-server job with a high system efficiency.

- *Service locality*. Could by described by a bipartite graph.
- *Unknown arrival patterns of jobs.* We don't know a job will arrive or not at some time t.
- *Unknown processing speeds (fluctuated around a certain value).* This falls into the non-clairvoyant job scheduling scenarios. Existing works cannot be applied.

Modeling with Bipartite Graph

We use the bipartite graph $(\mathcal{L}, \mathcal{R}, \mathcal{E})$ to model service locality.



Time is slotted, at each time $t \in \mathcal{T} := \{1, 2, ..., T\}$, a job is yielded from port $l \in \mathcal{L}$ with prob. $\rho_l(t)$. There are K types of computing devices in the cluster, including CPUs, GPUs, NPUs, and FPGAs.

Utility Formulation

The number of type-k devices is c_k. Each type-l job requests $a_k^{(l,r)} \in \mathbb{N}^+$ type-k devices. The decision variables are:

$$\boldsymbol{x}(t) := \left[x_{(l,r)}(t)\right]^{\mathrm{T}}_{(l,r)\in\mathcal{E}} \in \mathcal{X} := \left\{0, 1\right\}^{|\mathcal{E}|}. \tag{1}$$

$\forall r \in \mathcal{R}_l, x_{(l,r)}(t) = 0$ if $1_l(t) = 0$.

Formulate the utility of the type-l job at time t:

$$U_l(t) := \underbrace{\sum_{r\in\mathcal{R}_l} x_{(l,r)}(t) Z_{(l,r)}(t)}_{\text{gain}} - \underbrace{\sum_k \sum_{r\in\mathcal{R}_l} f_k(a_k^{(l,r)})(t) x_{(l,r)}(t)}_{\text{operating cost}}, \tag{2}$$

where $Z_{(l,r)}(t)$ is a stochastic variable following an underlying distribution with the expectation of $v_{(l,r)}$.

Scheduling without Knowing the Processing Speeds

$Z_{(l,r)}(t)$ captures the processing speed experienced by type-l job at time t. We don't know the value of $Z_{(l,r)}(t)$ until time t elapses. Correspondingly, $v_{(l,r)}$ can never be known, but can *be approximated* through learning.

Our goal is to maximize the expectation of job utilities:

$$\mathcal{P}_1: \quad \max_{\forall t \in \mathcal{T}: \boldsymbol{x}(t) \in \mathcal{X}} \lim_{T \to \infty} \sum_{t=1}^{T} \mathbb{E}\left[\sum_{l \in \mathcal{L}} U_l(t) \right]$$

$$s.t. \sum_{(l,r) \in \mathcal{E}} a_k^{(l,r)} x_{(l,r)}(t) \leq c_k, \forall k \in \mathcal{K}, t \in \mathcal{T}, \tag{3}$$

$$\sum_{r \in \mathcal{R}_l} x_{(l,r)}(t) = 0 \ \mathit{if} \ \mathbb{1}_l(t) = 0, \forall l \in \mathcal{L}, t \in \mathcal{T}. \tag{4}$$

Outline

1. System Model & Problem Formulation
   - Non-Clairvoyant Online Job Scheduling
   - Modeling with Bipartite Graph
   - Problem Formulation
2. Algorithm Design
   - Scheduling with Evolving Statistics
   - A Series of Budgeted IPs
   - Solving Subproblems with DP
   - The ESDP Framework

Scheduling with Evolving Statistics

We denote by $\tilde{\boldsymbol{Z}}(t)$ the column vector

$$\left[Z_{(l,r)}(t) - \sum_{k\in\mathcal{K}} f_k(a_k^{(l,r)})\right]^{\mathrm{T}}_{\forall(l,r)\in\mathcal{E}}$$

and normalize it into $[0,1]^{|\mathcal{E}|}$. We further introduce

$$\begin{cases} \tilde{\boldsymbol{v}} := \left[v_{(l,r)} - \sum_{k\in\mathcal{K}} f_k\left(a_k^{(l,r)}\right)\right]^{\mathrm{T}}_{\forall(l,r)\in\mathcal{E}} \in [0,1]^{|\mathcal{E}|} \\ \boldsymbol{x}^*(t) := \operatorname{argmax}_{\boldsymbol{x}(t)\in\Omega(t)} \left\{\tilde{\boldsymbol{v}}^{\mathrm{T}}\boldsymbol{x}(t)\right\} \\ \Omega(t) := \left\{\boldsymbol{x}(t)\in\mathcal{X} \mid \text{(3) \& (4) hold at time } t\right\}. \end{cases} \tag{5}$$

Then, $\mathcal{P}_1$ can be written as $\min_{\boldsymbol{x}(t)\in\Omega(t)} \sum_t \mathbb{E}\left[\tilde{\boldsymbol{Z}}(t)^{\mathrm{T}}\boldsymbol{x}(t)\right]$.

Scheduling with Evolving Statistics

At each time t, we define

$$n_{(l,r)}(t) := \sum_{t'=1}^{t} x_{(l,r)}(t') \tag{6}$$

as the *cumulative quantity* of channel $(l, r) \in \mathcal{E}$ been used up to time t. Based on it, we introduce the following statistics:

$$\hat{\upsilon}_{(l,r)}(t) := \begin{cases} \frac{\sum_{t'=1}^{t} x_{(l,r)}(t')\tilde{Z}_{(l,r)}(t')}{n_{(l,r)}(t)} & n_{(l,r)}(t) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$\hat{\sigma}^2_{(l,r)}(t) := \begin{cases} \frac{g(t)}{2n_{(l,r)}(t)} & n_{(l,r)}(t) > 0 \\ +\infty & \text{otherwise,} \end{cases} \tag{8}$$

where $g(t) := \ln t + 4\ln(\ln t + 1) \cdot \max_{t' \in \mathcal{T}} \left\{ \max_{\boldsymbol{x} \in \Omega(t')} \|\boldsymbol{x}\|_1 \right\}$ is designed to modeling the variance of the estimate.

Scheduling with Evolving Statistics

With the statistics, we introduce the following *deterministic* problem $\mathcal{P}_3(t)$:

$$\mathcal{P}_3(t): \max_{\boldsymbol{x}(t)\in\Omega(t)} \tilde{\mathrm{U}}(\boldsymbol{x}(t)) := \delta(t) + \underbrace{\hat{\boldsymbol{v}}(t)^{\mathrm{T}}\boldsymbol{x}(t)}_{\text{mean}} + \underbrace{\sqrt{\hat{\boldsymbol{\sigma}}^2(t)^{\mathrm{T}}\boldsymbol{x}(t)}}_{\text{standard deviation}}$$

$$\begin{aligned} s.t. \quad & (3), \\ & \delta(t) > 0, \lim_{t\to\infty} \delta(t) = 0, \end{aligned} \tag{9}$$

where $\hat{\boldsymbol{v}}(t) := [\hat{v}_{(l,r)}(t)]^{\mathrm{T}}_{(l,r)\in\mathcal{E}}$, and $\hat{\boldsymbol{\sigma}}^2(t) := [\hat{\sigma}^2_{(l,r)}(t)]^{\mathrm{T}}_{(l,r)\in\mathcal{E}}$. Note that (4) is not considered, temporarily.

$\{\delta(t)\}_{t\in\mathcal{T}}$ could be any sequence converges to zero. For instance,

$$\delta(t) := \frac{1}{\ln\big(\ln t + 1\big) + 1}. \tag{10}$$

Scaling Up

At each time t, based on $\delta(t)$, we define the following scale-up statistics for $\hat{\upsilon}_{(l,r)}(t)$ and $\hat{\sigma}^2_{(l,r)}(t)$ respectively:

$$\hat{\Upsilon}_{(l,r)}(t) := \left\lceil \xi(t)\hat{\upsilon}_{(l,r)}(t) \right\rceil \tag{11}$$

$$\hat{\Sigma}^2_{(l,r)}(t) := \left\lceil \xi^2(t)\hat{\sigma}^2_{(l,r)}(t) \right\rceil, \tag{12}$$

where

$$\xi(t) := \left\lceil \frac{\max_{t' \in \mathcal{T}} \left\{ \max_{\boldsymbol{x} \in \Omega(t')} \|\boldsymbol{x}\|_1 \right\}}{\delta(t)} \right\rceil \tag{13}$$

is the scaling size at time t.

A Series of Budgeted IPs

At each time t, we introduce several budgeted integer programming problems $\mathcal{P}_4(s,t)$ for each $s \in \mathcal{S}(t)$, where

$$\mathcal{S}(t) := \left\{0, 1, ..., \xi(t) \cdot \max_{t' \in \mathcal{T}} \max_{\boldsymbol{x} \in \Omega(t')} \|\boldsymbol{x}\|_1\right\}, \tag{14}$$

as follows:

$$\begin{aligned} \mathcal{P}_4(s,t): \quad & \max_{\boldsymbol{x}(t) \in \mathcal{X}} \hat{\boldsymbol{\Sigma}}^2(t)^{\mathrm{T}} \boldsymbol{x}(t) \\ s.t. \quad & (3), (9), \\ & \hat{\boldsymbol{\Upsilon}}(t)^{\mathrm{T}} \boldsymbol{x}(t) \geq s. \end{aligned} \tag{15}$$

In $\mathcal{P}_4(s,t)$, $\hat{\boldsymbol{\Sigma}}^2(t)$ and $\hat{\boldsymbol{\Upsilon}}(t)$ are the corresponding column vectors for (11) and (12), respectively. From $\mathcal{P}_3$ to $\mathcal{P}_4$, the $\mathcal{O}(\ln T)$-regret is guaranteed.

A Series of Budgeted IPs

Let us use $\boldsymbol{x}^*_{\mathcal{P}_4}(s, t)$ to denote the optimal solution for $\mathcal{P}_4(s, t)$. Then, the final solution to $\max\{\mathcal{P}_4(s, t)\}_{s\in\mathcal{S}(t)}$ at time t, denoted by $\boldsymbol{x}^*_{\mathcal{P}_4}(t)$, is set as some $\boldsymbol{x}^*_{\mathcal{P}_4}(s^\star, t)$ where $s^\star \in \mathcal{S}(t)$ staisfies

$$s^\star \in \operatorname*{argmax}_{s\in\mathcal{S}(t)} \left\{ s + \sqrt{\hat{\boldsymbol{\Sigma}}^2(t)^{\mathrm{T}} \boldsymbol{x}^*_{\mathcal{P}_4}(s, t)} \right\}. \tag{16}$$

That is, we select the optimal scaling indicator and the corresponding value as the optimal solution for the series of problems $\{\mathcal{P}_4(s, t)\}_{s\in\mathcal{S}(t)}$.

Solving Each $\mathcal{P}_4(s, t)$

At each time t, corresponding to each $\mathcal{P}_4(s, t)$, we bring in the problem $\mathcal{P}_5(s, t, \boldsymbol{c}, i)$ as follows.

$$\begin{aligned} \mathcal{P}_5(s, t, \boldsymbol{c}, i): \quad & \max_{\boldsymbol{x}(t) \in \mathcal{X}} \hat{\boldsymbol{\Sigma}}^2(t)^{\mathrm{T}} \boldsymbol{x}(t) \\ s.t. \quad & (3), (9), (15), \\ & \sum_{e=e_1}^{e_i} x_e(t) = 0, \end{aligned} \tag{17}$$

where $\boldsymbol{c} := [c_k]_{k \in \mathcal{K}}^{\mathrm{T}}$ is the capacity vector in (3), $e := (l, r) \in \mathcal{E}$ and e_i is the i-th edge (l, r) in $\mathcal{E}$. The new constraint (17) is used to set the first several scheduling decisions (until i) to 0 forcibly. Obviously, $\mathcal{P}_5(s, t, \boldsymbol{c}, 0)$ is equal to $\mathcal{P}_4(s, t)$ because (17) is not functioning when $i = 0$.

Solving Each $\mathcal{P}_5(s, t, \boldsymbol{c}, i)$ with DP

The optimal solution of $\mathcal{P}_5(s, t, \boldsymbol{c}, i)$ can be obtained by recursing over s, $\boldsymbol{c}$, and i. We use $\boldsymbol{x}^*(s, t, \boldsymbol{c}, i)$ to denote the optimal solution of $\mathcal{P}_5(s, t, \boldsymbol{c}, i)$, and use $V^*_{\mathcal{P}_5}(s, t, \boldsymbol{c}, i)$ to denote the corresponding objective.

- If $x^*_{e_{i+1}}(s, t, \boldsymbol{c}, i) = 0$, i.e., the $(i+1)$-element of $\boldsymbol{x}^*(s, t, \boldsymbol{c}, i)$ is 0, then (17) is not violated for $\mathcal{P}_5(s, t, \boldsymbol{c}, i+1)$. Thus, we have

$$\boldsymbol{x}^*(s, t, \boldsymbol{c}, i+1) = \boldsymbol{x}^*(s, t, \boldsymbol{c}, i) \tag{18}$$

  and

$$V^*_{\mathcal{P}_5}(s, t, \boldsymbol{c}, i+1) = V^*_{\mathcal{P}_5}(s, t, \boldsymbol{c}, i). \tag{19}$$

  The result means that $\boldsymbol{x}^*(s, t, \boldsymbol{c}, i)$ is also the optimal solution to $\mathcal{P}_5(s, t, \boldsymbol{c}, i+1)$.

Solving Each $\mathcal{P}_5(s, t, \boldsymbol{c}, i)$ with DP

- If $x^*_{e_{i+1}}(s, t, \boldsymbol{c}, i) = 1$, we define matrix $\mathbf{A}$ by

$$\mathbf{A} = \left[a_k^{(l,r)}\right]^{K \times |\mathcal{E}|}.$$

Then we have

$$\mathbf{A}\Big(\boldsymbol{x}^*(s, t, \boldsymbol{c}, i) - \boldsymbol{e}_{i+1}\Big) \leq \boldsymbol{c} - A_{:,i+1}, \tag{20}$$

where $\boldsymbol{e}_{i+1}$ is the $(i+1)$-th standard unit basis. Besides,

$$\hat{\boldsymbol{\Upsilon}}(t)^{\mathrm{T}}\Big(\boldsymbol{x}^*(s, t, \boldsymbol{c}, i) - \boldsymbol{e}_{i+1}\Big) \geq s - \hat{\Upsilon}_{e_{i+1}}(t) \tag{21}$$

and

$$\hat{\boldsymbol{\Sigma}}^2(t)^{\mathrm{T}}\big(\boldsymbol{x}^*(s, t, \boldsymbol{c}, i) - \boldsymbol{e}_{i+1}\big) = \hat{\boldsymbol{\Sigma}}^2(t)^{\mathrm{T}}\boldsymbol{x}^*(s, t, \boldsymbol{c}, i) - \hat{\Sigma}^2_{e_{i+1}}(t).$$

Solving Each $\mathcal{P}_5(s, t, \boldsymbol{c}, i)$ with DP

Combining the above formula with (20) and (21), we can get the following evolving optimal substructure:

$$\begin{aligned} V^*_{\mathcal{P}_5}(s, t, \boldsymbol{c}, i) =& V^*_{\mathcal{P}_5}\Big(\max\left\{ s - \hat{\Upsilon}_{e_{i+1}}(t), 0 \right\}, t, \\ & \max\{\boldsymbol{c} - A_{:,i+1}, 0\}, i+1 \Big) + \hat{\Sigma}^2_{e_{i+1}}(t). \end{aligned} \tag{22}$$

Thus, for every possible s, $\boldsymbol{c}$, and i, we can update the solution to $\mathcal{P}_5(s, t, \boldsymbol{c}, i)$ by

$$x^*_{e_{i+1}}(s, t, \boldsymbol{c}, i) = \begin{cases} 0 & V^*_{\mathcal{P}_5}(s, t, \boldsymbol{c}, i) = V^*_{\mathcal{P}_5}(s, t, \boldsymbol{c}, i+1) \\ 1 & \text{otherwise.} \end{cases}$$

The recursion starts from condition $s = 0$, $\boldsymbol{c} = \boldsymbol{0}$, and $i = |\mathcal{E}|$.

ESDP

The ESDP algorithm is finally demonstrated below.

while $t = 1, ..., T$ **do**
    Observe the job arrival status from each port $l \in \mathcal{L}$
    Update $\hat{\Upsilon}(t)$ and $\hat{\Sigma}^2(t)$ with (11) and (12) based on $\delta(t)$, respectively
    for *each* $s \in \mathcal{S}(t)$ **do**
        Solve $\mathcal{P}_4(s, t)$ and return $\boldsymbol{x}^*_{\mathcal{P}_4}(s, t)$
    end for
    $\boldsymbol{x}^*_{\mathcal{P}_4}(t) \leftarrow \boldsymbol{x}^*_{\mathcal{P}_4}(s^\star, t)$, where $s^\star$ staisfies (16)
    for *each* $l \in \mathcal{L}$ **do**
        if $\mathbb{1}_l(t) == 0$ **then**
            for *each* $r \in \mathcal{R}_l$ **do**
                Set the (l, r)-th element of $\boldsymbol{x}^*_{\mathcal{P}_4}(t)$ as 0
            end for
        end if
    end for
end while