Configuration Balancing for Stochastic Request

Franziska Eberle, Anupam Gupta, Nicole Megow
Benjamin Moseley, Rudy Zhou
Stochastic Load Balancing

• $m$ unrelated machines
• $n$ jobs with stochastic sizes such that job $j$ has size $X_{ij} \sim \text{on machine } i$
Stochastic Load Balancing

- \( m \) unrelated machines
- \( n \) jobs with stochastic sizes such that job \( j \) has size \( X_{ij} \sim \) on machine \( i \)
Stochastic Load Balancing

• $m$ unrelated machines
• $n$ jobs with **stochastic sizes** such that job $j$ has size $X_{ij} \sim \text{stochastic distribution}$ on machine $i$
Stochastic Load Balancing

• $m$ unrelated machines
• $n$ jobs with stochastic sizes such that job $j$ has size $X_{ij} \sim \text{on machine } i$
Stochastic Load Balancing

• $m$ unrelated machines
• $n$ jobs with stochastic sizes such that job $j$ has size $X_{ij} \sim \text{on machine } i$
Stochastic Load Balancing

• $m$ unrelated machines
• $n$ jobs with **stochastic sizes** such that job $j$ has size $X_{ij} \sim \text{on machine } i$
Stochastic Load Balancing

- $m$ unrelated machines
- $n$ jobs with stochastic sizes such that job $j$ has size $X_{ij} \sim \text{on machine } i$

Minimize expected max load ...compared to optimal adaptive policy
Stochastic Load Balancing

- $m$ unrelated machines
- $n$ jobs with stochastic sizes such that job $j$ has size $X_{ij}$ on machine $i$

Max load

Minimize expected max load

...compared to optimal adaptive policy
Related Work

- Deterministic setting well-studied
  - 2-approximation offline (LP rounding) [Lenstra, Shmoys, Tardos, Math. Prog. 1990]
  - $O(\log m)$-competitive online (potential function) [Aspnes, Azar, Fiat, Plotkin, Waarts, J. ACM 1997]
  - Variety of generalizations (multidimensional, norm objective, etc.)

- Stochastic setting focused on non-adaptive policies
  - Non-adaptive algorithm that $O(1)$-approximates optimal non-adaptive policy (LP rounding + effective size) [Gupta, Kumar, Nagarajan, Shen, Math. Oper. Res. 2021]
Related Work

- Deterministic setting well-studied
  - 2-approximation offline (LP rounding) [Lenstra, Shmoys, Tardos, Math. Prog. 1990]
  - $O(\log m)$-competitive online (potential function) [Aspnes, Azar, Fiat, Plotkin, Waarts, J. ACM 1997]
  - Variety of generalizations (multidimensional, norm objective, etc.)

- Stochastic setting focused on non-adaptive policies
  - Non-adaptive algorithm that $O(1)$-approximates optimal non-adaptive policy (LP rounding + effective size) [Gupta, Kumar, Nagarajan, Shen, Math. Oper. Res. 2021]
  - Adaptivity gap is $\Omega\left(\frac{\log m}{\log \log m}\right)$
Our Results

Theorem: There exists an efficient algorithm for stochastic load balancing on unrelated machines that $O\left(\frac{\log m}{\log \log m}\right)$-approximates the optimal adaptive policy. Further, the algorithm is non-adaptive.

- Also give $O(1)$-approximate adaptive policy for related machines
Our Results

**Theorem:** There exists an efficient algorithm for stochastic load balancing on unrelated machines that $O\left(\frac{\log m}{\log \log m}\right)$-approximates the optimal adaptive policy. Further, the algorithm is non-adaptive.

- Also give $O(1)$-approximate adaptive policy for related machines
- First general result for stochastic load balancing compared to optimal adaptive policy
- Gives tight upper bound on adaptivity gap
- Can be generalized to variety of other resource allocation problems and online setting
Our Results

**Theorem:** There exists an efficient algorithm for stochastic load balancing on unrelated machines that $O\left(\frac{\log m}{\log \log m}\right)$-approximates the optimal adaptive policy. Further, the algorithm is non-adaptive.

- Also give $O(1)$-approximate adaptive policy for related machines
- First general result for stochastic load balancing compared to optimal adaptive policy
- Gives tight upper bound on adaptivity gap
- Can be generalized to variety of other resource allocation problems and online setting
- **New Idea:** Show that there exists near-optimal adaptive policy that behaves similarly to a non-adaptive policy
Warm Up: Small jobs

- Assume all jobs are small: $X_{ij} \in [0, E\ Opt]$ for all $i, j$
- Suffices to control expected load on each machine

Max expected load

$E[\ ] + \cdots + E[\ ]$

$E[\ ] + \cdots + E[\ ]$

$\vdots$

$E[\ ] + \cdots + E[\ ]$

$O(E\ Opt)$
Warm Up: Small jobs

- Assume all jobs are small: $X_{ij} \in [0, \mathbb{E} Opt]$ for all $i, j$
- Suffices to control expected load on each machine \((\text{Concentration + Union})\)

\[
\begin{align*}
&\text{Max expected load} & \text{Expected max load} \\
&O(\mathbb{E} Opt) & O(\log m) \mathbb{E} Opt
\end{align*}
\]
Warm Up: Small jobs

• Assume all jobs are small: $X_{ij} \in [0, \mathbb{E} Opt]$ for all $i, j$
• Suffices to control expected load on each machine (Concentration + Union)

Max expected load  |  Expected max load

Main Challenge: How to handle jobs that aren’t reasonably bounded?
Truncation

• Problem is easy if jobs are small ⇒ make jobs small and deal with big jobs separately
Truncation

- Problem is easy if jobs are small ⇒ make jobs small and deal with big jobs separately.
- Given truncation threshold $\tau$,
  - the truncated part of $X_{ij}$ is: $X_{ij}^T = X_{ij} \cdot 1_{X_{ij} \leq \tau}$
  - the exceptional part is: $X_{ij}^E = X_{ij} \cdot 1_{X_{ij} > \tau}$
Truncation

• Problem is easy if jobs are small ⇒ make jobs small and deal with big jobs separately
• Given truncation threshold \( \tau \),
  • the **truncated part** of \( X_{ij} \) is: \( X_{ij}^T = X_{ij} \cdot 1_{X_{ij} \leq \tau} \)
  • the **exceptional part** is: \( X_{ij}^E = X_{ij} \cdot 1_{X_{ij} > \tau} \)
• ⇒ handle truncated parts by controlling max expected load

**Question:** How to control expected max load of exceptional parts?
Exceptional parts

• Bound contribution of exceptional parts: $$\mathbb{E} \left[ \max_i \sum_{j \to i} X^E_{ij} \right]$$

• Only have trivial upper bound:

$$\mathbb{E} \left[ \max_i \sum_j X^E_{ij} \cdot 1_{j \to i} \right] \leq \sum_i \sum_j \mathbb{E}[X^E_{ij} \cdot 1_{j \to i}]$$
Exceptional parts

• Bound contribution of exceptional parts: \( \mathbb{E} \left[ \max_i \sum_{j \rightarrow i} X_{ij}^E \right] \)

• Only have trivial upper bound:

\[
\mathbb{E} \left[ \max_i \sum_{j} X_{ij}^E \cdot 1_{j \rightarrow i} \right] \leq \sum_i \sum_j \mathbb{E}[X_{ij}^E \cdot 1_{j \rightarrow i}] \]

**Algorithm goal:** assign jobs to machines non adaptively such that each machine has expected truncated load \( O(\mathbb{E} \text{ Opt}) \) and the total expected exceptional load \( O(\mathbb{E} \text{ Opt}) \)

**Question:** Does there exist such an assignment?
Benefit of Adaptivity

- One fast machine, $m - 1$ slow
- One Bernoulli job, $m - 1$ deterministic
Benefit of Adaptivity

- One fast machine, $m - 1$ slow
- One Bernoulli job, $m - 1$ deterministic

$\tau \sim \mathbb{E} \text{Opt}$
Benefit of Adaptivity

• One fast machine, $m - 1$ slow
• One Bernoulli job, $m - 1$ deterministic

$\tau \sim \mathbb{E} \text{ Opt}$
Benefit of Adaptivity

- One fast machine, $m - 1$ slow
- One Bernoulli job, $m - 1$ deterministic

$\tau \sim \mathbb{E} Opt$
Benefit of Adaptivity

• One \textbf{fast} machine, $m - 1$ \textbf{slow}
• One \textbf{Bernoulli} job, $m - 1$ \textbf{deterministic}

\textbf{Problem:} Optimal adaptive policy can have total expected exceptional load $\Omega(m) \cdot \mathbb{E} \text{Opt}$. 

\[ \tau \sim \mathbb{E} \text{Opt} \]
Structure Theorem

• For truncation threshold $\tau \sim \mathbb{E} \ Opt$, there exists an adaptive policy $\tilde{Opt}$ such that:
  • (near optimal) $\mathbb{E}[\tilde{Opt}] \leq 2 \cdot \mathbb{E}[Opt]$
  • (small total expected exceptional load) The total expected exceptional load of $\tilde{Opt}$ is at most $2 \cdot \mathbb{E}[Opt]$
Structure Theorem

• For truncation threshold $\tau \sim \mathbb{E} \text{Opt}$, there exists an adaptive policy $\tilde{\text{Opt}}$ such that:
  • (near optimal) $\mathbb{E}[\tilde{\text{Opt}}] \leq 2 \cdot \mathbb{E}[\text{Opt}]$
  • (small total expected exceptional load) The total expected exceptional load of $\tilde{\text{Opt}}$ is at most $2 \cdot \mathbb{E}[\text{Opt}]$

• $\Rightarrow$ natural assignment LP that ensures expected truncated load on each machine is $O(\mathbb{E} \text{Opt})$ and total expected exceptional load is $O(\mathbb{E} \text{Opt})$ is feasible
  • $\Rightarrow$ can round offline
  • $\Rightarrow$ can use potential function online
Proof

• Simulate $Opt$, but forget when we get unlucky

**(Existential) Algorithm:**
1. Given jobs $J$, follow optimal policy $Opt(J)$
2. If $Opt(J)$ assigns $j \rightarrow i$ such that $X_{ij}$ becomes exceptional ($X_{ij} > \tau \geq 2 \cdot \mathbb{E}[Opt(J)]$)
3. Forget all previously-accrued machine loads, and recurse on remaining jobs $R \subset J$
Proof

• Simulate $Opt$, but forget when we get unlucky

(Existential) Algorithm:
1. Given jobs $J$, follow optimal policy $Opt(J)$
2. If $Opt(J)$ assigns $j \rightarrow i$ such that $X_{ij}$ becomes exceptional ($X_{ij} > \tau \geq 2 \cdot \mathbb{E}[Opt(J)]$)
3. Forget all previously-accrued machine loads, and recurse on remaining jobs $R \subset J$
Proof

• Simulate $Opt$, but forget when we get unlucky

(Existential) Algorithm:
1. Given jobs $J$, follow optimal policy $Opt(J)$
2. If $Opt(J)$ assigns $j \rightarrow i$ such that $X_{ij}$ becomes exceptional ($X_{ij} > \tau \geq 2 \cdot \mathbb{E}[Opt(J)]$)
3. Forget all previously-accrued machine loads, and recurse on remaining jobs $R \subset J$
Proof

• Simulate $Opt$, but forget when we get unlucky

**(Existential) Algorithm:**
1. Given jobs $J$, follow optimal policy $Opt(J)$
2. If $Opt(J)$ assigns $j \rightarrow i$ such that $X_{ij}$ becomes exceptional ($X_{ij} > \tau \geq 2 \cdot \mathbb{E}[Opt(J)]$)
3. Forget all previously-accrued machine loads, and recurse on remaining jobs $R \subset J$
Proof

• Simulate $Opt$, but forget when we get unlucky

(Existential) Algorithm:
1. Given jobs $J$, follow optimal policy $Opt(J)$
2. If $Opt(J)$ assigns $j \rightarrow i$ such that $X_{ij}$ becomes exceptional ($X_{ij} > \tau \geq 2 \cdot \mathbb{E}[Opt(J)]$)
3. Forget all previously-accrued machine loads, and recurse on remaining jobs $R \subset J$
Proof

• Simulate $Opt$, but forget when we get unlucky

(Existential) Algorithm:

1. Given jobs $J$, follow optimal policy $Opt(J)$
2. If $Opt(J)$ assigns $j \to i$ such that $X_{ij}$ becomes exceptional ($X_{ij} > \tau \geq 2 \cdot \mathbb{E}[Opt(J)]$)
3. Forget all previously-accrued machine loads, and recurse on remaining jobs $R \subset J$
Proof

• Simulate $Opt$, but forget when we get unlucky

(Existential) Algorithm:
1. Given jobs $J$, follow optimal policy $Opt(J)$
2. If $Opt(J)$ assigns $j \rightarrow i$ such that $X_{ij}$ becomes exceptional ($X_{ij} > \tau \geq 2 \cdot \mathbb{E}[Opt(J)]$)
3. Forget all previously-accrued machine loads, and recurse on remaining jobs $R \subset J$

• Makespan $\leq \mathbb{E}[Opt(J)]$
• $\leq 1$ exceptional job

Same but scaled by $P(\text{recurse}) \leq \frac{1}{2}$
Proof

• Simulate $Opt$, but forget when we get unlucky

(Existential) Algorithm:
1. Given jobs $J$, follow optimal policy $Opt(J)$
2. If $Opt(J)$ assigns $j \to i$ such that $X_{ij}$ becomes exceptional ($X_{ij} > \tau \geq 2 \cdot \mathbb{E}[Opt(J)]$)
3. Forget all previously-accrued machine loads, and recurse on remaining jobs $R \subset J$
Structure Theorem

• For truncation threshold $\tau \sim \mathbb{E} Opt$, there exists an adaptive policy $\tilde{Opt}$ such that:
  • (near optimal) $\mathbb{E} [\tilde{Opt}] \leq 2 \cdot \mathbb{E} [Opt]$
  • (small total expected exceptional load) The total expected exceptional load of $\tilde{Opt}$ is at most $2 \cdot \mathbb{E} [Opt]$

• $\Rightarrow$ natural assignment LP that ensures expected truncated load on each machine is $O(\mathbb{E} Opt)$ and total expected exceptional load is $O(\mathbb{E} Opt)$ is feasible
  • $\Rightarrow$ can round offline
  • $\Rightarrow$ can use potential function online

Idea: Forget when we get unlucky
Conclusion

Theorem: There exists an efficient algorithm for stochastic load balancing on unrelated machines that $O\left(\frac{\log m}{\log \log m}\right)$-approximates the optimal adaptive policy. Further, the algorithm is non-adaptive.

• Many extensions: online, adaptive policies for related machines, stochastic routing, etc.
• Structure theorem for near-optimal adaptive policies
Conclusion

**Theorem:** There exists an efficient algorithm for stochastic load balancing on unrelated machines that $O\left(\frac{\log m}{\log \log m}\right)$-approximates the optimal adaptive policy. Further, the algorithm is non-adaptive.

- Many extensions: online, adaptive policies for related machines, stochastic routing, etc.
- Structure theorem for near-optimal adaptive policies

**Questions:**
- Improve using adaptivity?
- Hardness of approximation?