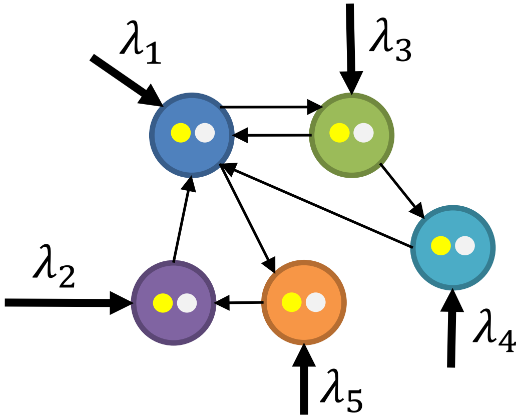
Greedy Algorithm for Multiway Matching with Bounded Regret

Varun Gupta

Motivation: Kidney Exchange



Online (Poisson) arrivals of (donor, recipient) pairs

Directed edge = donor of source pair compatible with the recipient of destination pair

State = queue lengths $(Q_1, Q_2, \ldots, Q_n)$ of unmatched pairs

Decision = which match (directed cycle) to execute now

Goal = maximize match quality, keep queues small

A high dimensional MDP!

Results of (Kerimov, Ashlagi, Gurvich. 2021)

- A periodic max weight matching policy keeps queues bounded
- If all feasible matches are of size 2, then a simple “Greedy” policy keeps queues bounded. “Greedy” fails for matches of size > 2

This paper: A different “Greedy” policy (greedy commitment) works; no tuning parameters

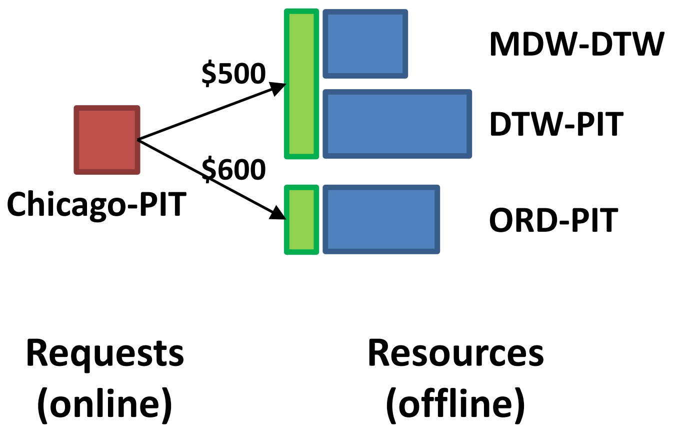
Main Contributions

1. A unified framework for many online resource allocation problems, e.g.,
   - Organ transplant matching
   - Assemble-to-Order systems
   - Network revenue management
   - Multiclass-multiserver queueing systems
   - Stochastic bin packing
   - Assortment under dynamic inventory

2. A universal greedy algorithm with small ($O(1)$ or $O(\log T)$) regret under a mild robustness of basis condition

3. Technique: Combination of Lypaunov Drift with Amortized Analysis to handle non-stationarity

Airline Network Revenue Management (NRM)



Finite horizon T

Offline Resources = capacities on legs

Online arrivals (*i.i.d.*) = origin-dest. pairs

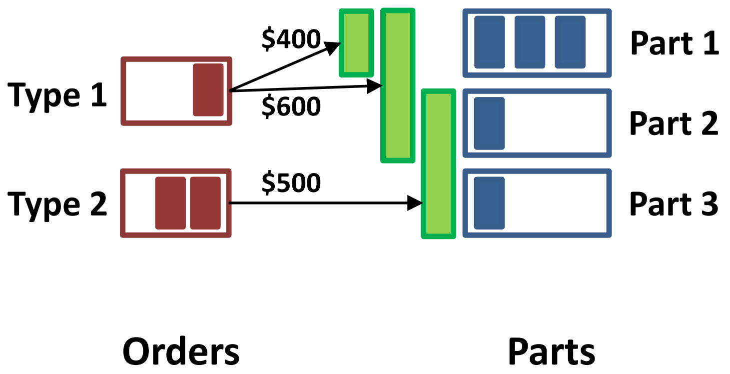
Decision = a feasible origin-dest. route

Goal = maximize total revenue

Extension: offer a *(route, price)*

Gallego and van Ryzin (1994), Reiman and Wang (2008), Jasin and Kumar (2012), Wu et al. (2015), Bumpensanti and He (2018), Vera et al. (2018)

Assemble-to-Order (ATO) Systems



Online arrivals (*i.i.d.*) of orders, can be queued or discarded on arrival

Parts/Components used to assemble orders

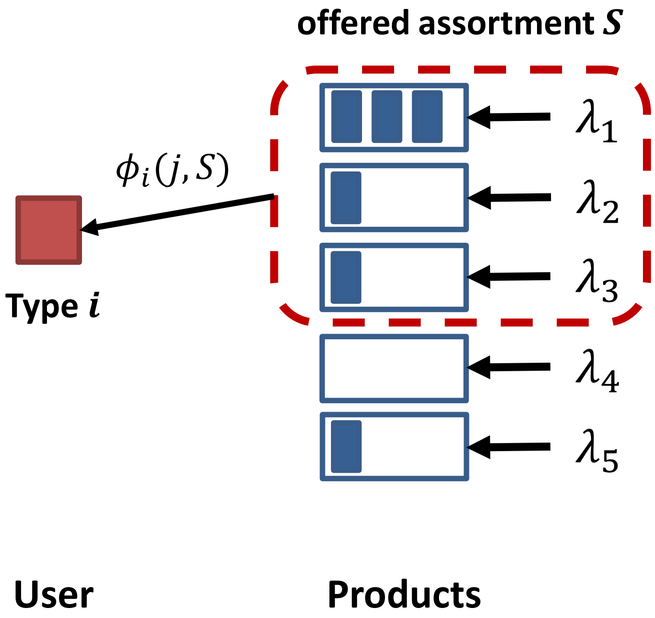
Decisions

1. Replenishment: when/how to purchase parts (assume *i.i.d.* exogenous arrivals)
2. Assembly: if/when/how to fulfill orders

Goal = maximize total revenue, keeping queues small

Plambeck and Ward (2008), Gurvich and Ward (2014), Nazari and Stolyar (2016)

Assortments under Dynamic Inventory

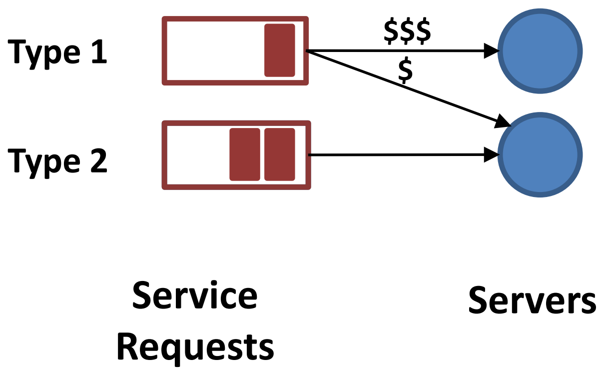


Online replenishment of products and arrivals of users

Decisions: which assortment to offer subject to current inventory

Goal = maximize revenue

Multiclass-Multiserver Queueing Systems



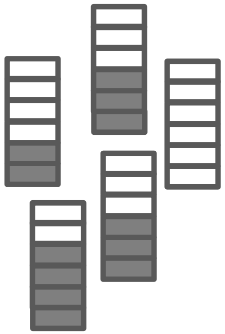
Online arrivals of service requests

Service requests can be queued or discarded on arrival

Decision = which request to serve when a server idles

Goal = maximize match quality, keep queues small

Stochastic (static) Bin Packing



Bins

Items (online)

Infinite collection of bins of integer size B

Online arrivals (*i.i.d.*) = items with size $\leq B$

Decision = irrevocably assign items on arrival to a feasible bin

Goal = minimize number of bins used

Csirik et al. (2006), Gupta and Radovanovic (2020), Banerjee and Freund (2021)

Outline

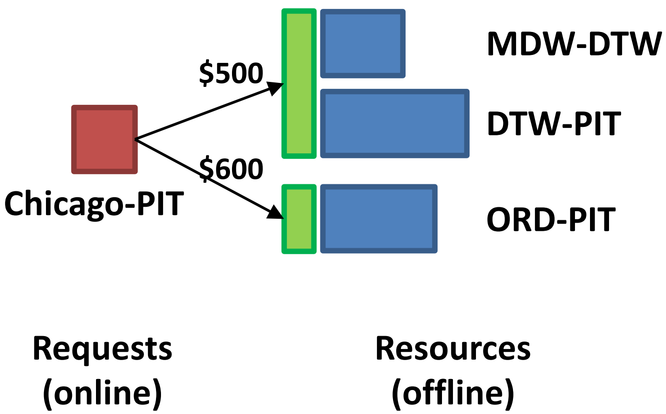
- A unified framework for online multiway matching

- Special case : online-queueable resources, *i.i.d.* arrivals
  - Definitions and Result
  - Algorithm
  - Analysis ideas

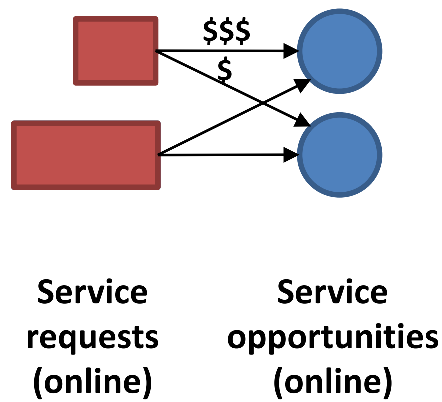
- General case : offline and nonqueueable resources, non-stationary arrivals
  - Definitions and Results
  - Algorithm

A unified online matching framework

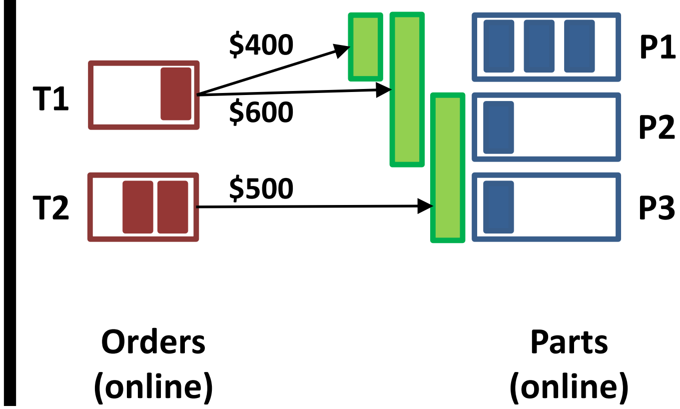
Airline network revenue management

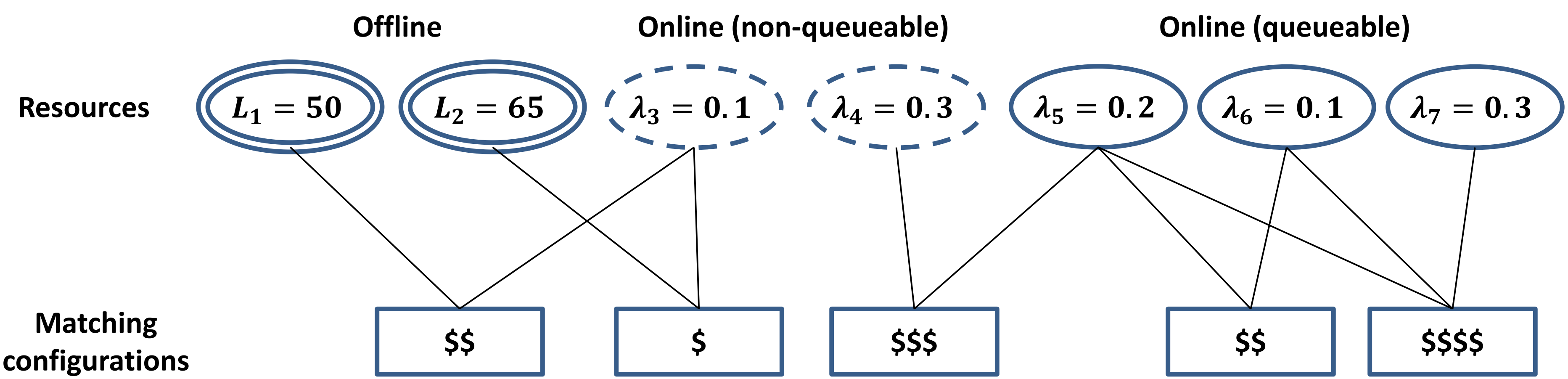


Multiclass-multiserver queueing



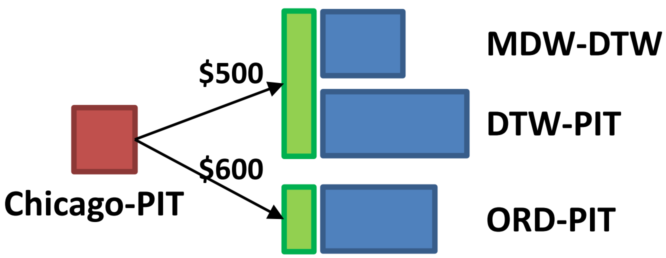
Assemble-to-Order





A unified online matching framework

Airline network revenue management



Requests (online)

Resources (offline)

Multiclass-multiserver queueing

Service requests (online)

Service opportunities (online)

Assemble-to-Order

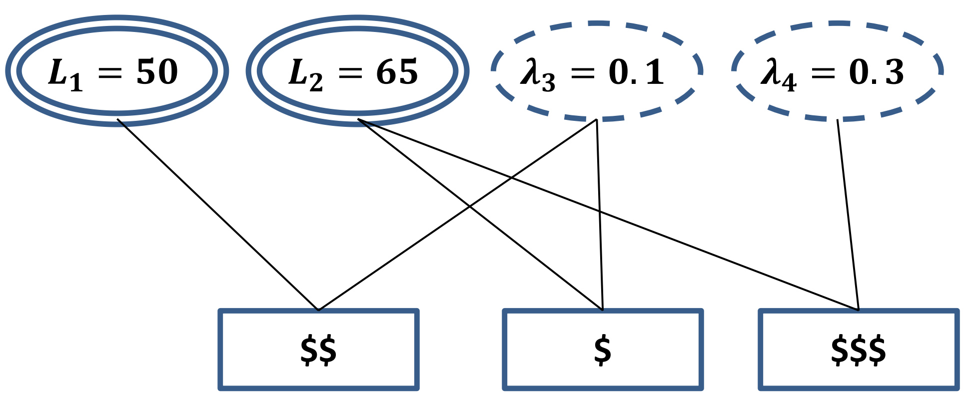
Orders (online)

Parts (online)

Offline

Online (non-queueable)

Resources



Matching configurations

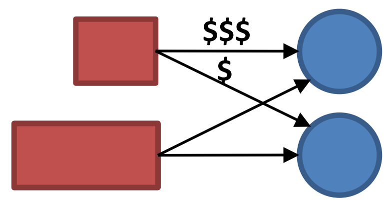
A unified online matching framework

Airline network revenue management

Requests (online)

Resources (offline)

Multiclass-multiserver queueing



Service requests (online)

Service opportunities (online)

Assemble-to-Order

Orders (online)

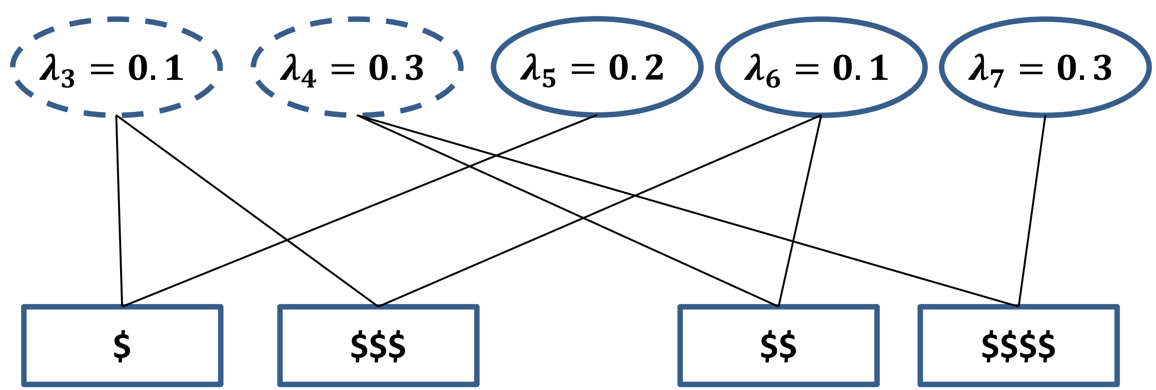
Parts (online)

Online (non-queueable)

Online (queueable)

Resources

Matching configurations



A unified online matching framework

Airline network revenue management

Requests (online)

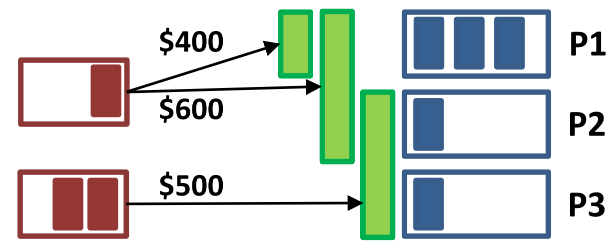
Resources (offline)

Multiclass-multiserver queueing

Service requests (online)

Service opportunities (online)

Assemble-to-Order



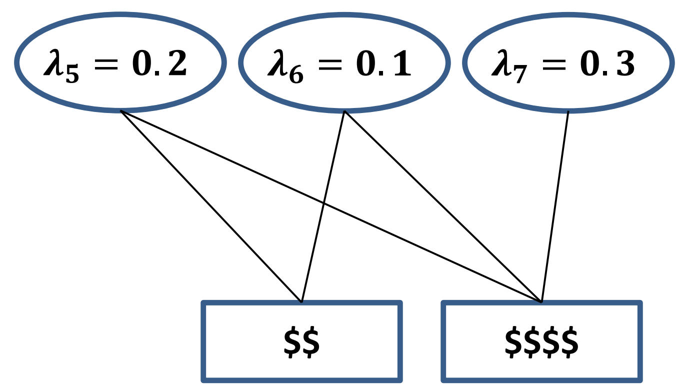
Orders (online)

Parts (online)

Resources

Matching configurations

Online (queueable)



A unified online matching framework

- Horizon T
- Resource/Item types $\mathcal{I} = \mathcal{I}^{\text{off}} \cup \mathcal{I}^{\text{on-q}} \cup \mathcal{I}^{\text{on-nq}} = \{1, \dots, n\}$
- Matching configurations $\mathcal{M} = \{1, \dots, d\}$, matrix $M \in \mathbb{R}_+^{n \times d}$
  - M_{im} = average number of type i resources consumed by m
  - reward r_m
  - $\mathcal{M}$ includes singletons to model discarding of resources
- Known stationary demand distribution $\boldsymbol{\lambda}$
- Offline resources: inventory $L_i = \lambda_i T$ for $\mathrm{i} \in \mathcal{I}^{\text{off}}$
- Online resources: at each time $t \in [T]$ item $\mathrm{i} \in \mathcal{I}^{\text{on-q}} \cup \mathcal{I}^{\text{on-nq}}$ arrives w. prob. λ_i
- $\mathcal{I}^{\text{on-nq}}$ are lost if not matched on arrival; $\mathcal{I}^{\text{on-q}}$ can be queued indefinitely
- **Goal:** Minimize anytime regret

$$\max_t \mathbb{E}[(\text{Reward of } OPT^t) - (\text{Reward of } ALG^t)]$$

Outline

- A unified framework for online multiway matching
- Special case : online-queueable resources, *i.i.d.* arrivals
  - Definitions and Result
  - Algorithm
  - Analysis ideas

> - Assemble-to-Order with exogenous replenishments
> - Kidney exchange

- General case : offline and nonqueueable resources, non-stationary arrivals
  - Definitions and Results
  - Algorithm

Definitions: Static Planning Problem (SPP), General Position Gap (GPG)

Static Planning Problem ($SPP(\lambda)$):

$$\max_{\{x_m\}\geq \mathbf{0}} \sum_m r_m \cdot x_m$$

subject to

$$\forall i \in \mathcal{I} : \sum_m M_{im} \cdot x_m = \boldsymbol{\lambda}_i$$

Definition (GPG_ϵ): Let $\mathcal{M}_+$ be an optimal basis for $(SPP(\boldsymbol{\lambda}))$ with optimal solution x^*.

We say that λ satisfies GPG_ϵ if for any $\hat{\boldsymbol{\lambda}} \in \mathcal{B}_{\epsilon,\mathrm{TV}}(\boldsymbol{\lambda})$, there exists an optimal solution to $SPP(\hat{\boldsymbol{\lambda}})$ with basis $\mathcal{M}_+$

Definitions (contd.)

Definition' (GPG_ϵ): $\boldsymbol{\lambda}$ satisfies GPG_ϵ if

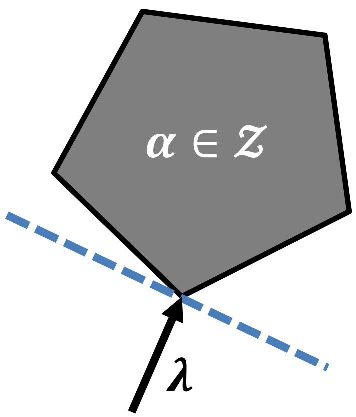
1. the solution $\boldsymbol{\alpha}^*$ to the dual of $SPP(\boldsymbol{\lambda})$ is unique
2. $\boldsymbol{\alpha}^*$ is the solution to dual of $SPP(\hat{\boldsymbol{\lambda}})$ for any $\hat{\boldsymbol{\lambda}} \in \mathcal{B}_{\epsilon,\mathrm{TV}}(\boldsymbol{\lambda})$

Dual Static Planning Problem ($DSPP(\boldsymbol{\lambda})$):

$$\min_{\{\alpha_i\}} \sum_{i \in \mathcal{I}} \lambda_i \cdot \alpha_i$$

subject to

$$\forall m \in \mathcal{M}: \sum_{i \in \mathcal{I}} M_{im} \cdot \alpha_i \geq r_m$$

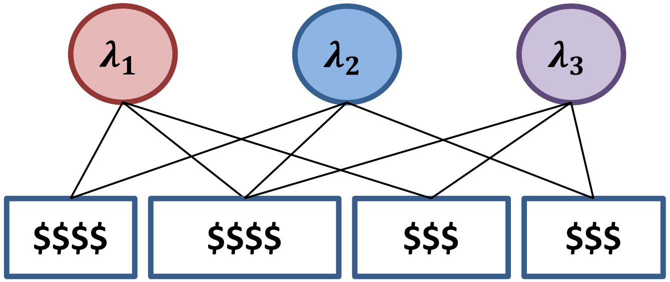


Theorem 1: Under GPG_ϵ, Greedy algorithm gets anytime regret

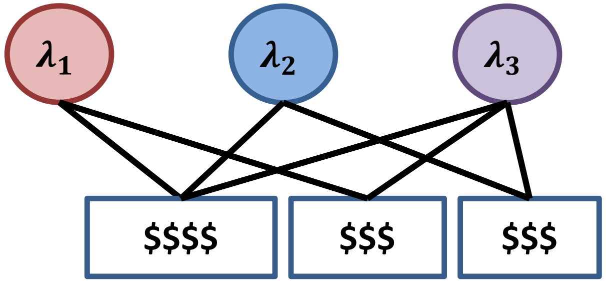
$$\text{Regret} \leq O\left({n^3}/{\epsilon}\right).$$

Greedy Algorithm

1. Solve for x^* and an optimal basis $\mathcal{M}_+$

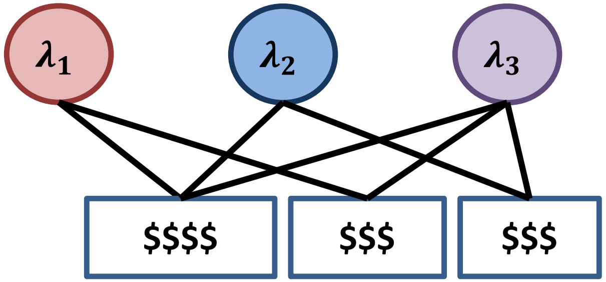


Greedy Algorithm



1. Solve for x^* and an optimal basis $\mathcal{M}_+$

Greedy Algorithm

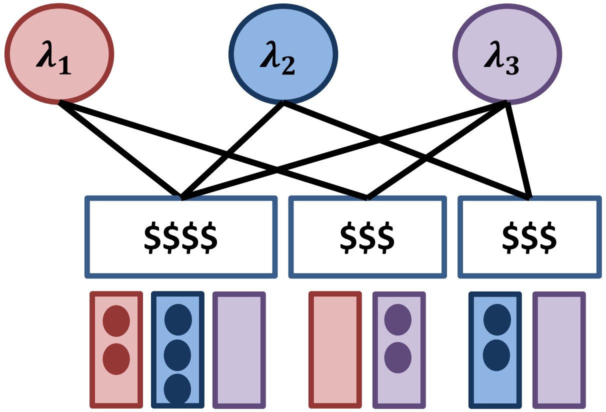


1. Solve for x^* and an optimal basis $\mathcal{M}_+$

Idea: commit an arrival to some configuration $m \in \mathcal{M}_+$ greedily

2. Maintain queues for each $m \in \mathcal{M}_+$ and $i \in m$

Greedy Algorithm



1. Solve for x^* and an optimal basis $\mathcal{M}_+$

Idea: commit an arrival to some configuration $m \in \mathcal{M}_+$ greedily

2. Maintain queues for each $m \in \mathcal{M}_+$ and $i \in m$

Q_{im} = number of unmatched type i items committed to m

Greedy Algorithm

1. Solve for x^* and an optimal basis $\mathcal{M}_+$

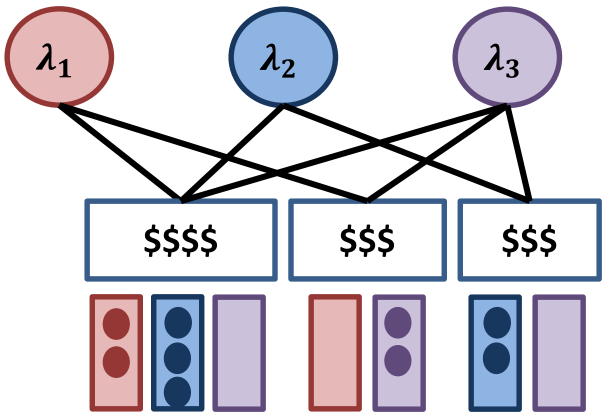
Idea: commit an arrival to some configuration $m \in \mathcal{M}_+$ greedily

2. Maintain queues for each $m \in \mathcal{M}_+$ and $i \in m$

Q_{im} = number of unmatched type i items committed to m

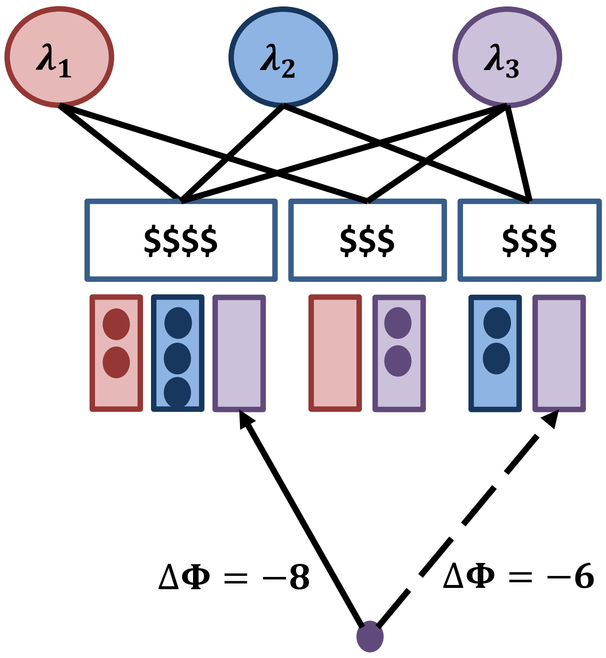
3. Cyclic SS potential function for config. m:

$$\Phi_m(\boldsymbol{Q_m}) = \left(\frac{Q_{i_1 m}}{M_{i_1 m}} - \frac{Q_{i_2 m}}{M_{i_2 m}}\right)^2 + \left(\frac{Q_{i_2 m}}{M_{i_2 m}} - \frac{Q_{i_3 m}}{M_{i_3 m}}\right)^2 + \cdots + \left(\frac{Q_{i_m m}}{M_{i_m m}} - \frac{Q_{i_1 m}}{M_{i_1 m}}\right)^2$$



$$\Phi_m = (2-3)^2 + (3-0)^2 + (0-2)^2$$

Greedy Algorithm



1. Solve for x^* and an optimal basis $\mathcal{M}_+$

Idea: commit an arrival to some configuration $m \in \mathcal{M}_+$ greedily

2. Maintain queues for each $m \in \mathcal{M}_+$ and $i \in m$

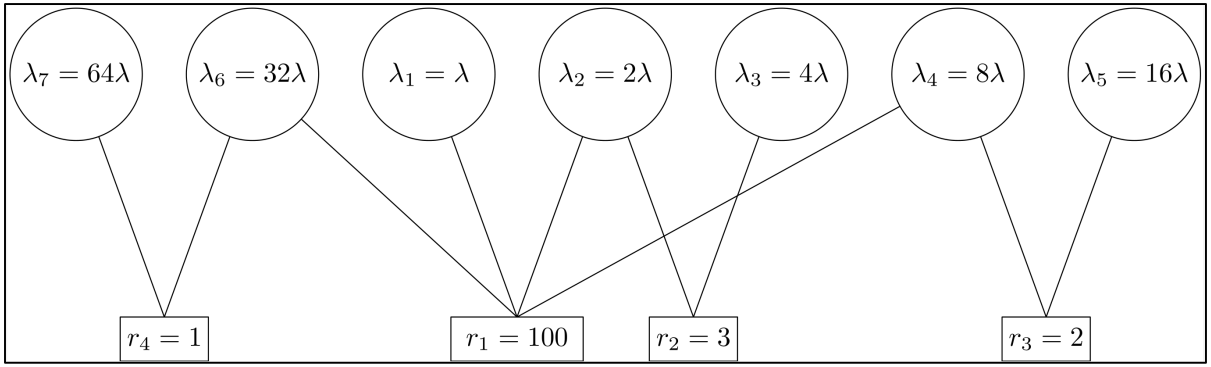
Q_{im} = number of unmatched type i items committed to m

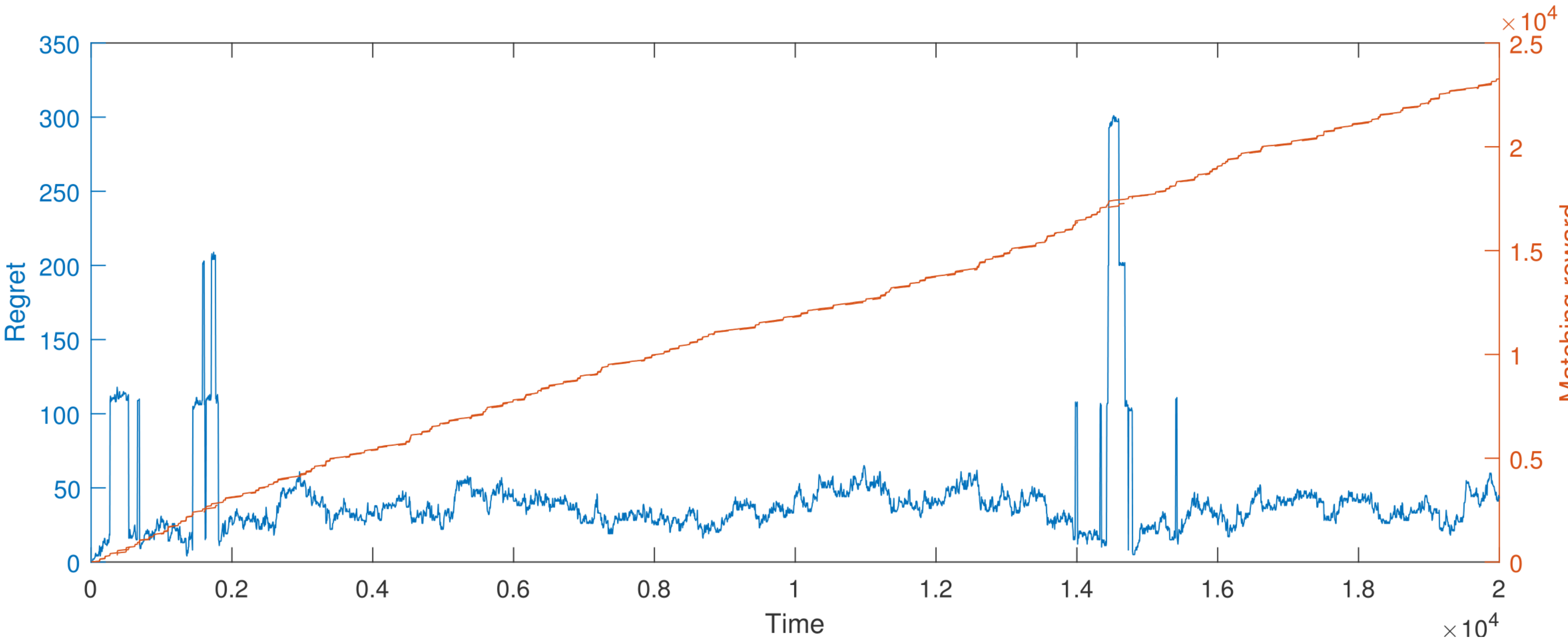
3. Cyclic SS potential function for config. m:

$$\Phi_m(\boldsymbol{Q_m}) = \left(\frac{Q_{i_1m}}{M_{i_1m}} - \frac{Q_{i_2m}}{M_{i_2m}}\right)^2 + \left(\frac{Q_{i_2m}}{M_{i_2m}} - \frac{Q_{i_3m}}{M_{i_3m}}\right)^2 + \cdots + \left(\frac{Q_{i_mm}}{M_{i_mm}} - \frac{Q_{i_1m}}{M_{i_1m}}\right)^2$$

4. Commit item to minimize $\sum_m \Phi_m(\boldsymbol{Q_m})$

Results





Example from Kerimov et al. (2021)

Analysis Ideas

Regret-Q lemma: $\text{Regret}^t \leq \boldsymbol{\alpha}^* \cdot \mathbb{E}[\boldsymbol{Q}^t]$

Lyapunov function: $\Psi^t = \sqrt{\Phi^t}$

Bounded increment: $|\Psi^{t+1} - \Psi^t| \leq \sqrt{2}$ almost surely

Drift lemma: If Ψ^t is large then there is some item $i \in \mathcal{I}$ such that

1. expected change in Ψ^t is large and negative when i arrives
2. expected change in Ψ^t is $\mathcal{O}(1)$ due to $\boldsymbol{\lambda} - \epsilon \cdot \boldsymbol{e_i}$

$$\Psi^t \text{ large} \implies \text{expected drift} \leq -{}^{\epsilon}/_{4n^2}$$

Random walk bound: Upper bound Ψ^t in increasing convex order by a random walk with *i.i.d.* increments of size $\pm\sqrt{2}$ and bias $\left(-{}^{\epsilon}/_{4n^2}\right)$

Outline

- A unified framework for online multiway matching

- Special case : online-queueable resources, *i.i.d.* arrivals
  - Definitions and Result
  - Algorithm
  - Analysis ideas

- General case : offline and nonqueueable resources, non-stationary arrivals
  - Definitions and Results
  - Algorithm

What changes?

Obstacle 1 (Algorithm): Dealing with offline resources

Solution: Treat them like online resources by simulating an arrival process (need knowledge of T), i.e., the item arriving at time t is

- either an $\mathcal{J}^{\mathrm{off}}$ item with probability 1
- or, an $\mathcal{J}^{\mathrm{on-q}} \cup \mathcal{J}^{\mathrm{on-nq}}$ item sampled from $\boldsymbol{\lambda}$

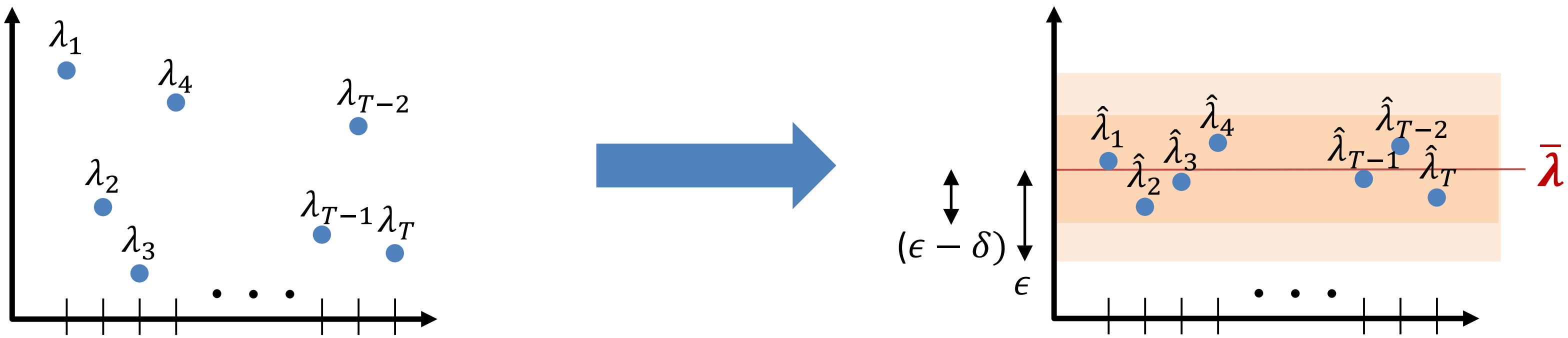
Obstacle 2 (Analysis): This gives a non-stationary arrival process

Solution:

- Relax GPG_{ϵ} to smoothed $GPG_{\delta,\tau}$
- combine amortized analysis with Lyapunov drift analysis

Smoothed $GPG_{\delta,\tau}$ condition

$(\lambda_1, \lambda_2, \dots, \lambda_T)$: a *previsible* stochastic process w.r.t the filtration of arrivals



There exists a process $\hat{\boldsymbol{\lambda}}^t$ that is :

(i) Close to $\bar{\boldsymbol{\lambda}}$: $\hat{\boldsymbol{\lambda}}^t \in \mathcal{B}_{\epsilon-\delta,\mathrm{TV}}(\bar{\boldsymbol{\lambda}})$

(ii) Tracks $\boldsymbol{\lambda}^t$: $\sum_{s=1}^{t-2\tau} \hat{\boldsymbol{\lambda}}^s \le \sum_{s=\tau+1}^{t} \boldsymbol{\lambda}^s \le \sum_{s=1}^{t} \hat{\boldsymbol{\lambda}}^s$

Smoothed $GPG_{\delta,\tau}$ $\approx$ the avg. distribution in any τ window is $\epsilon - \delta$ close to $\bar{\boldsymbol{\lambda}}$

Results

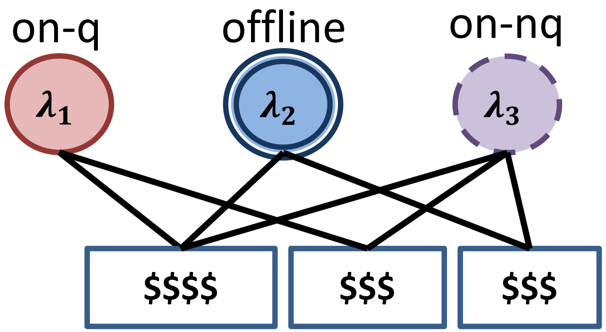
Theorem 2: If no configurations in $\mathcal{M}_+$ have items from both $\mathcal{J}^{\mathrm{on-q}}, \mathcal{J}^{\mathrm{on-nq}}$ then under $GPG_{\delta,\tau}$

$$\text{Regret} \leq O\left({}^{n^3\tau^2}/_{\delta}\right).$$

Theorem 3: If some configuration in $\mathcal{M}_+$ has items from both $\mathcal{J}^{\mathrm{on-q}}, \mathcal{J}^{\mathrm{on-nq}}$ then under $GPG_{\delta,\tau}$

$$\text{Regret} \leq O\left({}^{n^3\tau^4 \log t}/_{\delta}\right).$$

Greedy Algorithm

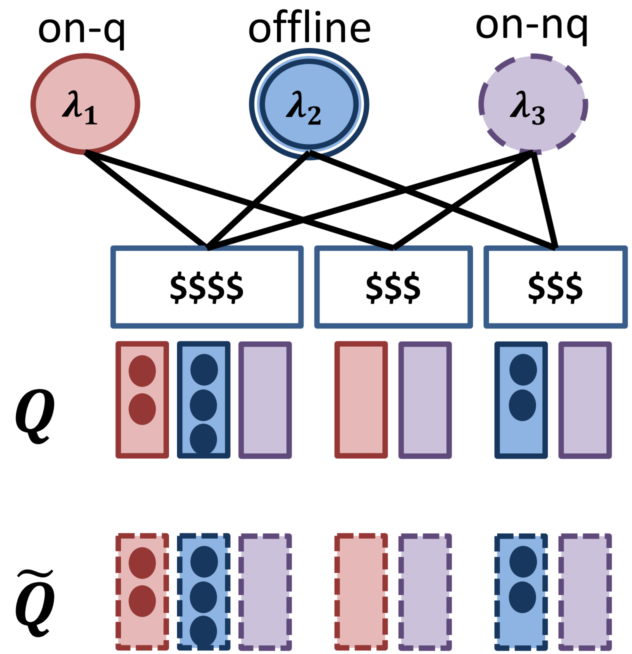


1. Solve for x^* and an optimal basis $\mathcal{M}_+$

Greedy Algorithm

1. Solve for x^* and an optimal basis $\mathcal{M}_+$

2. Maintain **physical** and **virtual** queues for each $m \in \mathcal{M}_+$ and $i \in m$



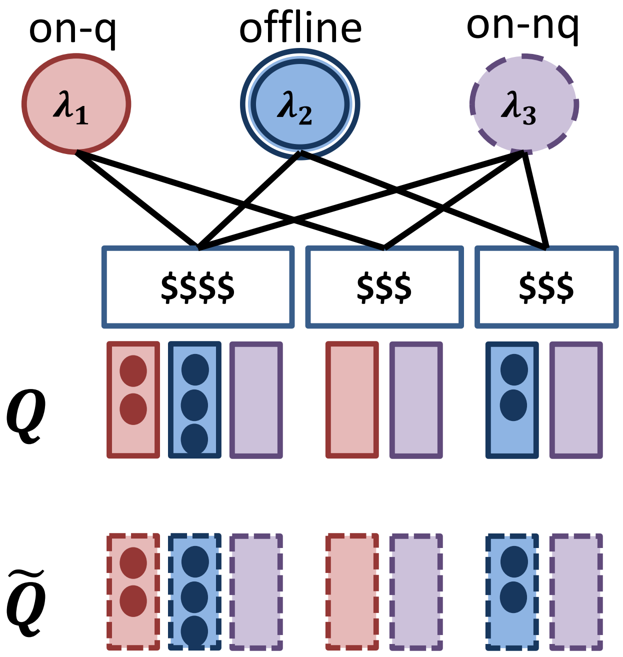
Greedy Algorithm

1. Solve for x^* and an optimal basis $\mathcal{M}_+$

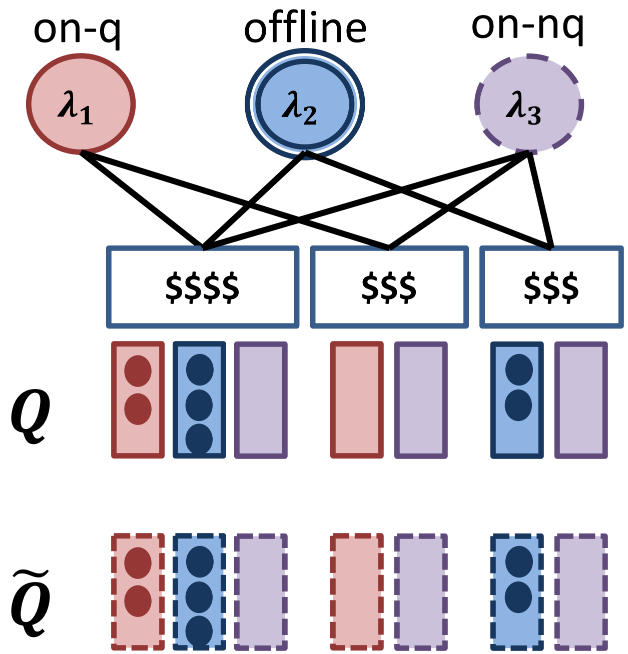
2. Maintain **physical** and **virtual** queues for each $m \in \mathcal{M}_+$ and $i \in m$

$\tilde{Q}_{im}$ = virtual queue length for (i, m)

3. Commit item to minimize $\sum_m \Phi_m(\tilde{\boldsymbol{Q}}_{\boldsymbol{m}})$



Greedy Algorithm



1. Solve for x^* and an optimal basis $\mathcal{M}_+$

2. Maintain **physical** and **virtual** queues for each $m \in \mathcal{M}_+$ and $i \in m$

$\tilde{Q}_{im}$ = virtual queue length for (i, m)

3. Commit item to minimize $\sum_m \Phi_m(\tilde{\boldsymbol{Q}}_m)$

4. **Virtual match:** executed if

- arrival is on-nq, or
- arrival is on-q or offline, and sufficient resources locally in $\tilde{\boldsymbol{Q}}$

Greedy Algorithm

1. Solve for x^* and an optimal basis $\mathcal{M}_+$

2. Maintain **physical** and **virtual** queues for each $m \in \mathcal{M}_+$ and $i \in m$

$\tilde{Q}_{im}$ = virtual queue length for (i, m)

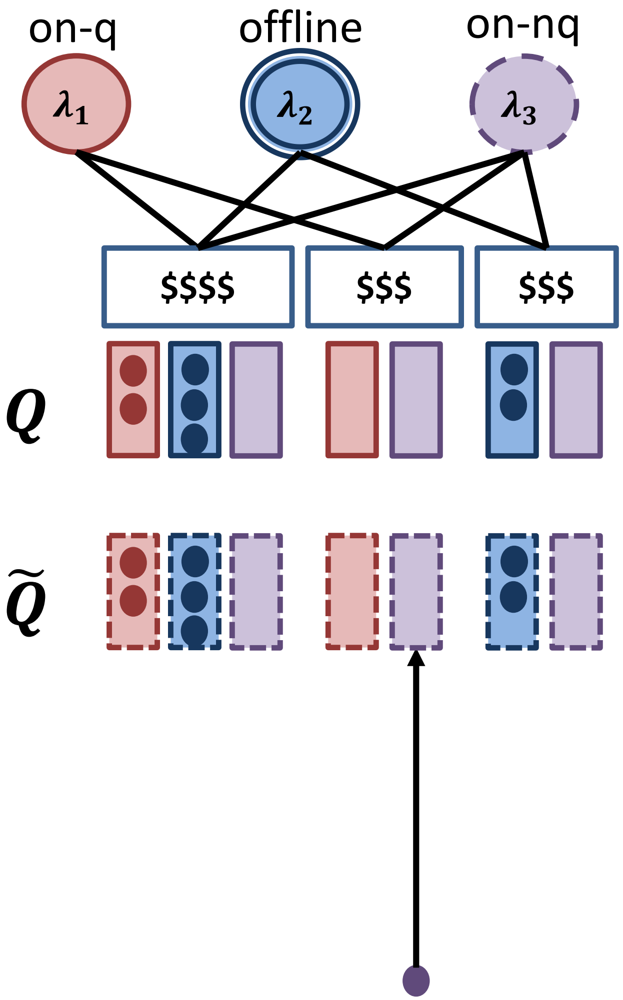
3. Commit item to minimize $\sum_m \Phi_m(\tilde{\boldsymbol{Q}}_{\boldsymbol{m}})$

4. **Virtual match:** executed if

- arrival is on-nq, or
- arrival is on-q or offline, and sufficient resources locally in $\tilde{\boldsymbol{Q}}$

Physical match: executed if

- virtual match happens, and
- on-q available locally in $\boldsymbol{Q}$, and
- offline available globally (possibly outside $\boldsymbol{Q}$)



Greedy Algorithm

1. Solve for x^* and an optimal basis $\mathcal{M}_+$

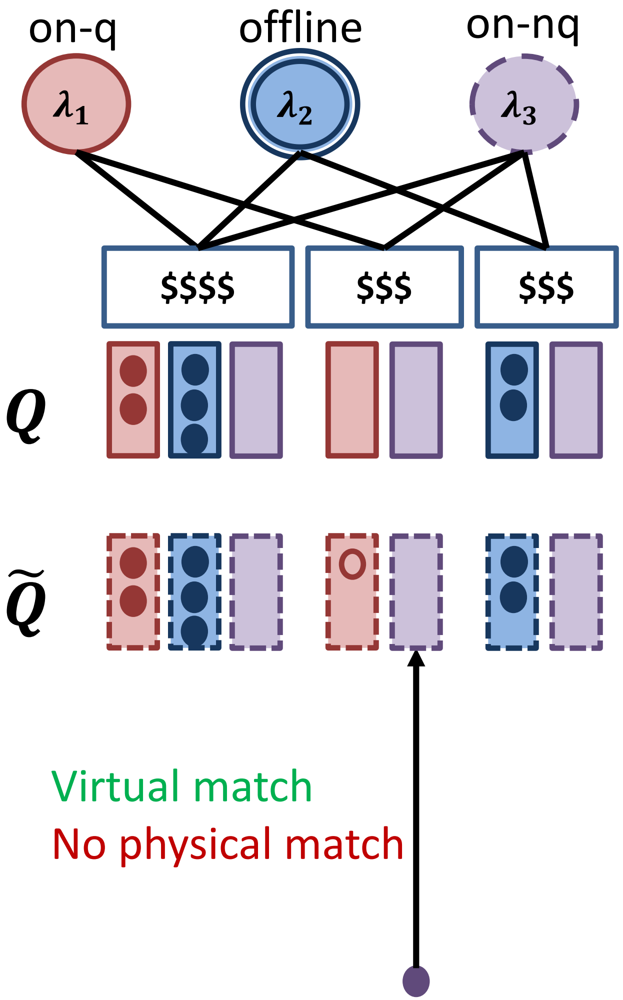
2. Maintain **physical** and **virtual** queues for each $m \in \mathcal{M}_+$ and $i \in m$

$\tilde{Q}_{im}$ = virtual queue length for (i, m)

3. Commit item to minimize $\sum_m \Phi_m(\tilde{\boldsymbol{Q}}_m)$

4. **Virtual match:** executed if
- arrival is on-nq, or
- arrival is on-q or offline, and sufficient resources locally in $\tilde{\boldsymbol{Q}}$

Physical match: executed if
- virtual match happens, and
- on-q available locally in $\boldsymbol{Q}$, and
- offline available globally (possibly outside $\boldsymbol{Q}$)



Greedy Algorithm

1. Solve for x^* and an optimal basis $\mathcal{M}_+$

2. Maintain **physical** and **virtual** queues for each $m \in \mathcal{M}_+$ and $i \in m$

$\tilde{Q}_{im}$ = virtual queue length for (i, m)

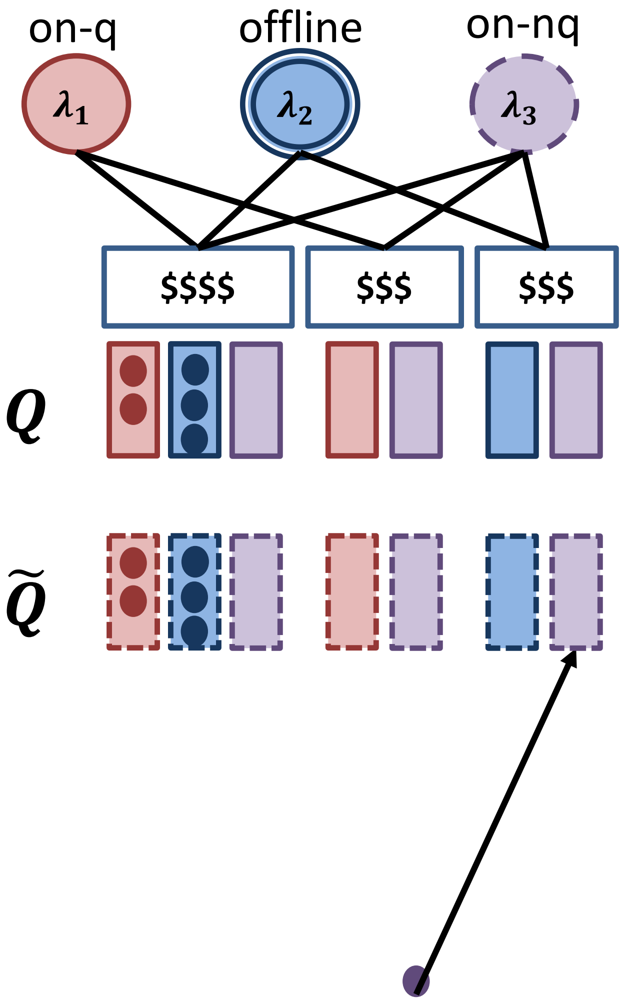
3. Commit item to minimize $\sum_m \Phi_m(\tilde{\boldsymbol{Q}}_m)$

4. **Virtual match:** executed if

- arrival is on-nq, or
- arrival is on-q or offline, and sufficient resources locally in $\tilde{\boldsymbol{Q}}$

Physical match: executed if

- virtual match happens, and
- on-q available locally in $\boldsymbol{Q}$, and
- offline available globally (possibly outside $\boldsymbol{Q}$)



Greedy Algorithm

1. Solve for x^* and an optimal basis $\mathcal{M}_+$

2. Maintain **physical** and **virtual** queues for each $m \in \mathcal{M}_+$ and $i \in m$

$\tilde{Q}_{im}$ = virtual queue length for (i, m)

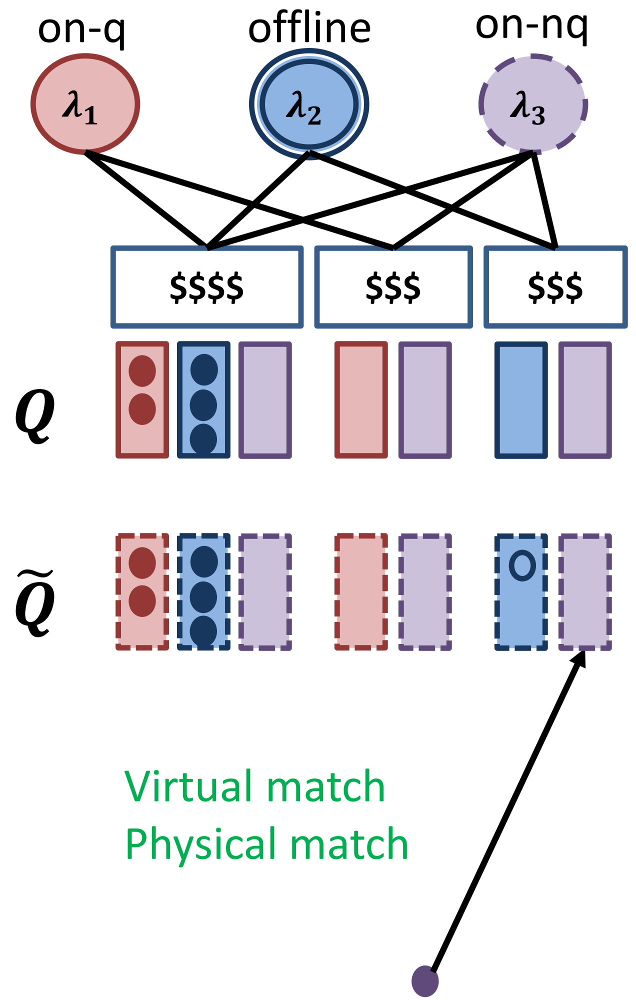
3. Commit item to minimize $\sum_m \Phi_m(\tilde{\boldsymbol{Q}}_m)$

4. **Virtual match:** executed if

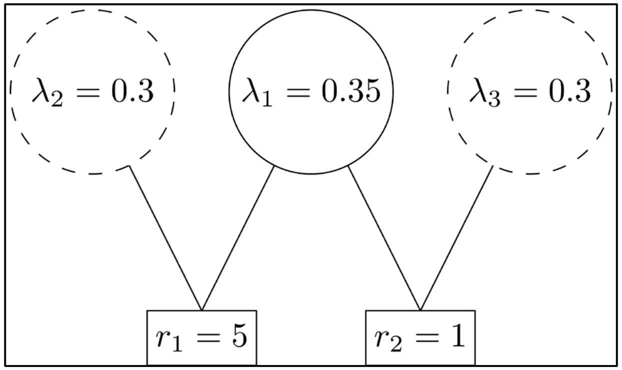
- arrival is on-nq, or
- arrival is on-q or offline, and sufficient resources locally in $\tilde{\boldsymbol{Q}}$

Physical match: executed if

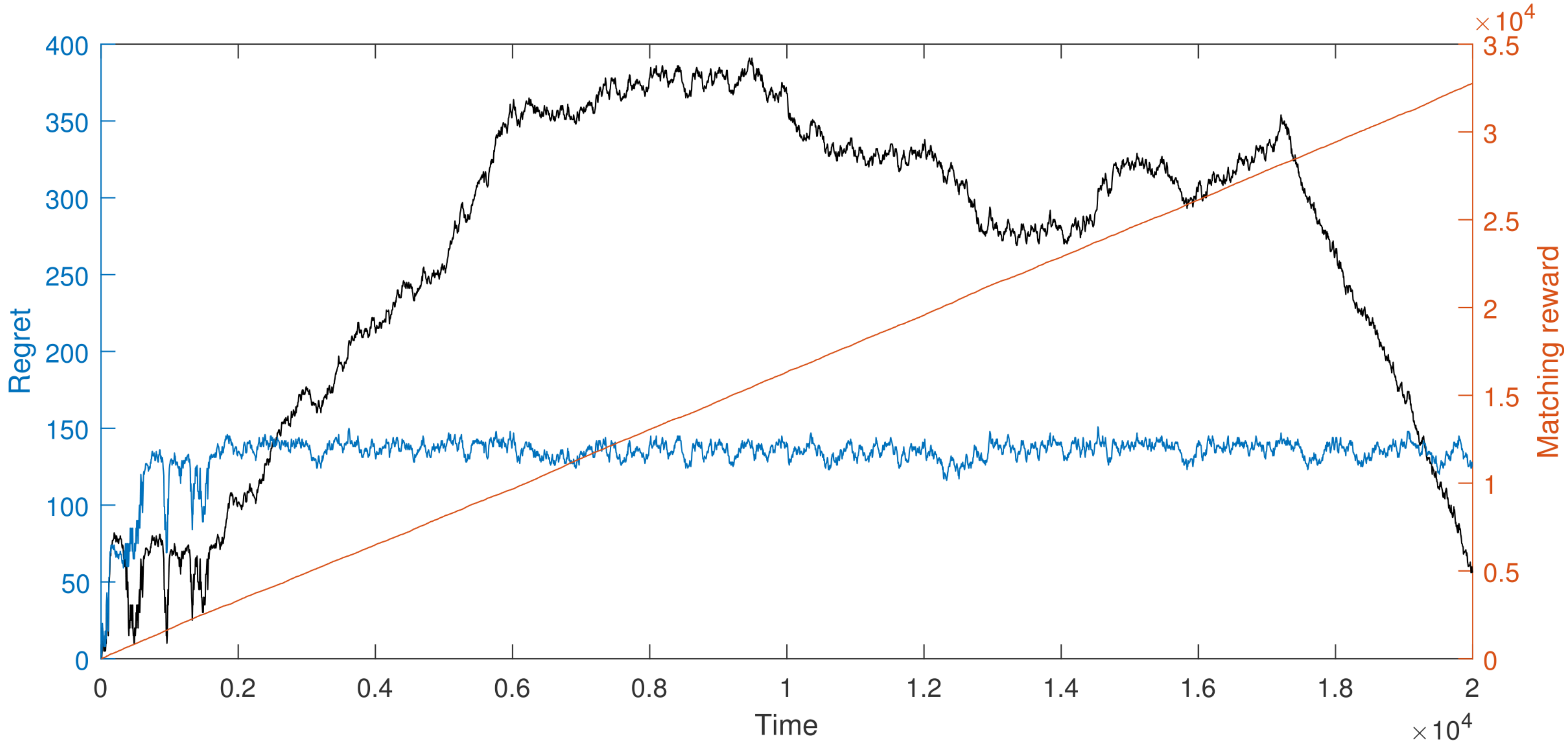
- virtual match happens, and
- on-q available locally in $\boldsymbol{Q}$, and
- offline available globally (possibly outside $\boldsymbol{Q}$)



Results



$\lambda_2 = 0.3$
$\lambda_1 = 0.35$
$\lambda_3 = 0.3$
$r_1 = 5$
$r_2 = 1$



Regret
400
350
300
250
200
150
100
50
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Time
$\times 10^4$
$\times 10^4$
3.5
3
2.5
2
1.5
1
0.5
0
Matching reward

Next steps

- Replenishment decisions for Assemble-to-Order systems

- Reusable resources

e.g., loss networks (Iyengar and Sigman, 2004)

- Extension to non-stationary demand

- Incorporating noisy forecasts

e.g., Gaussian process demand (Ciocan and Farias, 2012)

- Waiting costs / abandonments