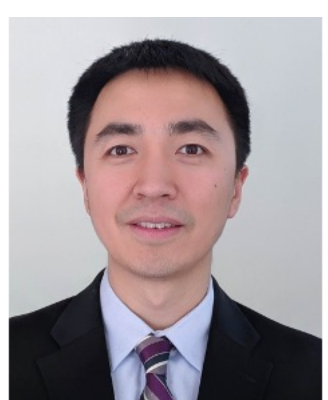
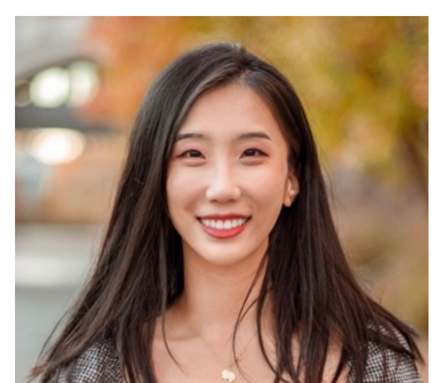
A New Primal-Dual Policy for Dynamic Matching

Jiaming Xu

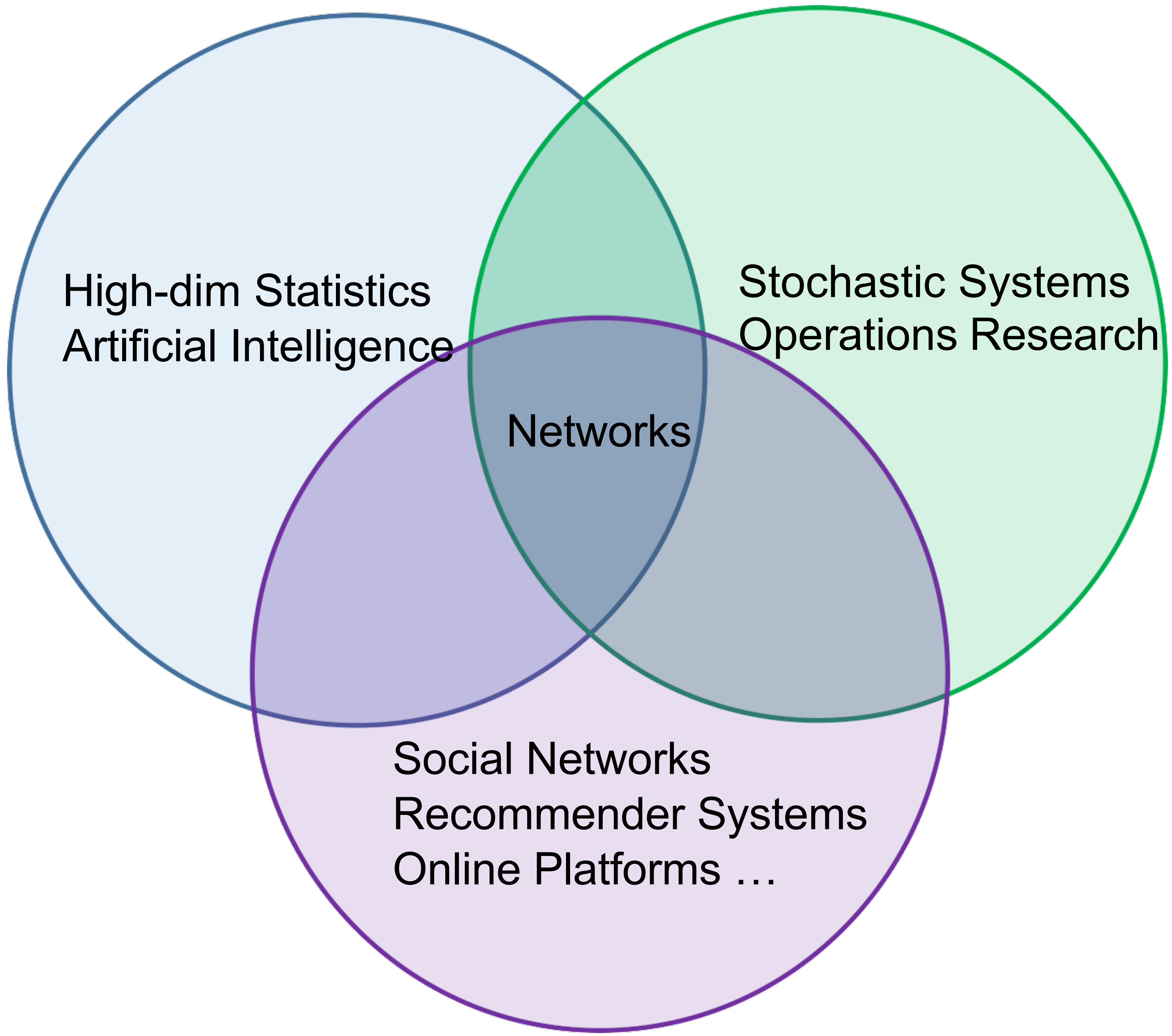
Decision Sciences Area, Duke Fuqua

Joint work: Yehua Wei (Duke) and Sophie H. Yu (Wharton)





Overview of my research and teaching interests

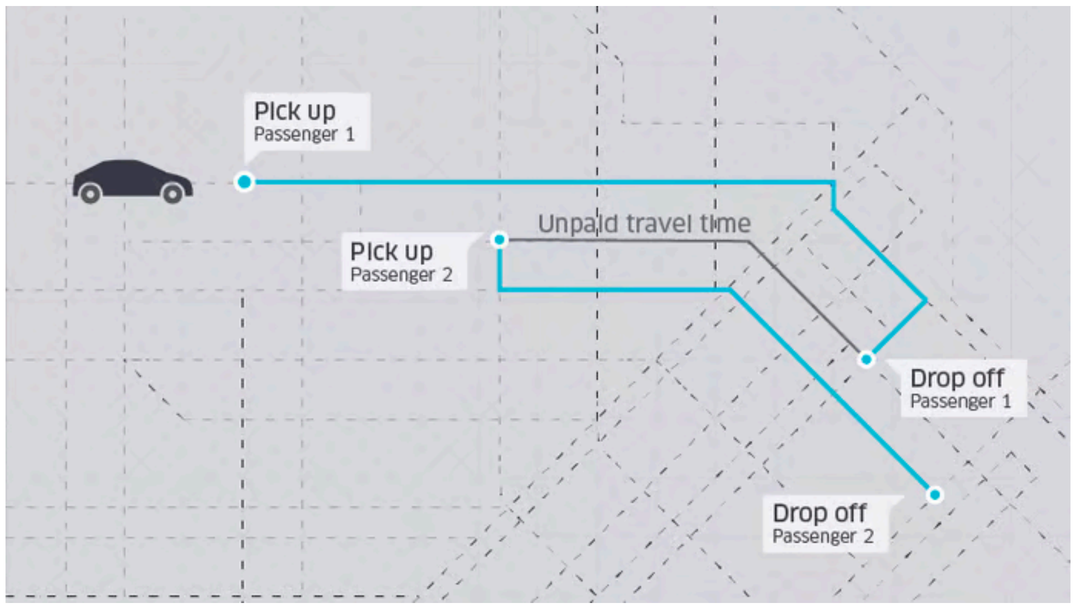


Motivating example: On-demand car-pooling

- Rides requesting a ride along a similar route will share the ride and split the cost

Motivating example: On-demand car-pooling

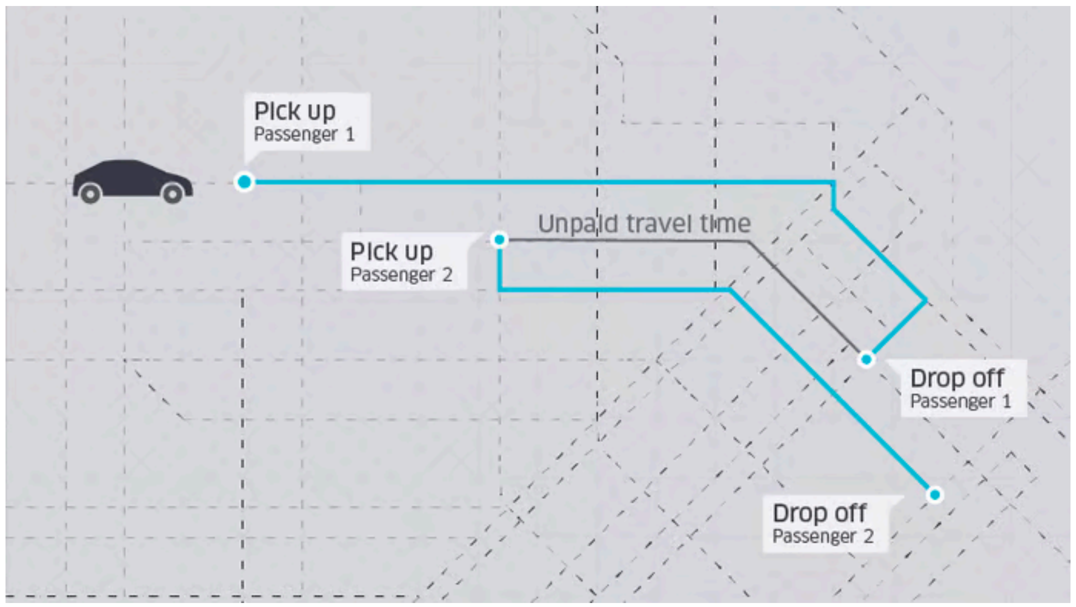
- Rides requesting a ride along a similar route will share the ride and split the cost



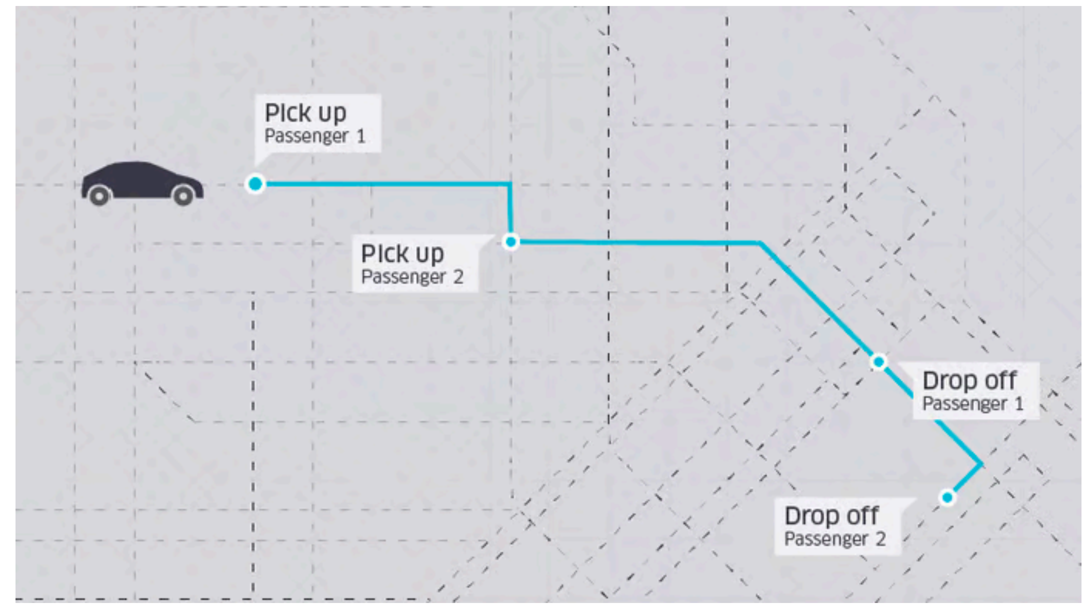
Without sharing, passengers pay higher rates and drivers don't earn money in between trips

Motivating example: On-demand car-pooling

- Rides requesting a ride along a similar route will share the ride and split the cost



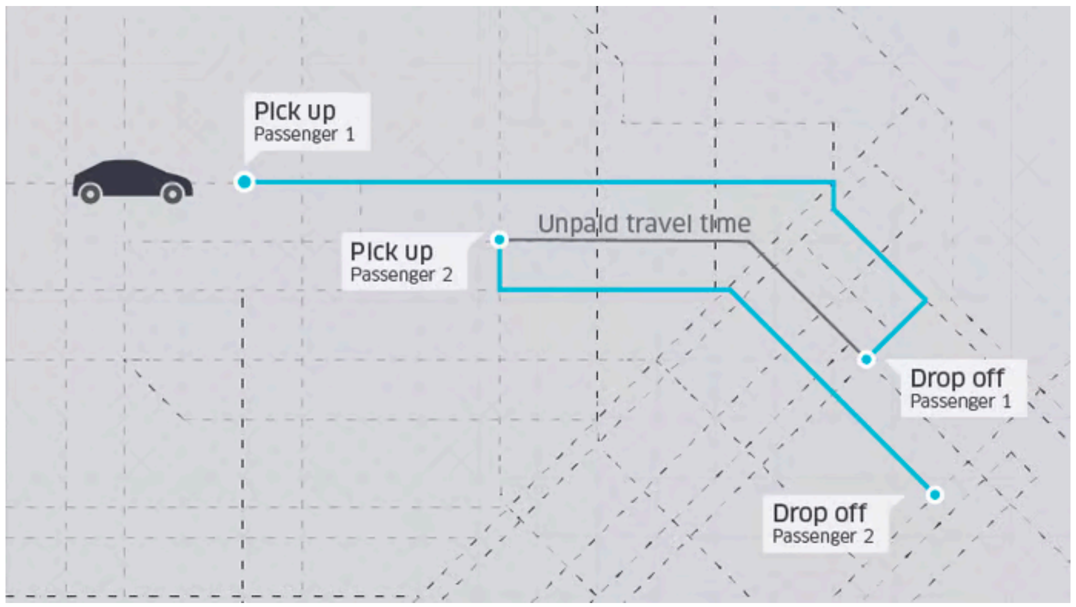
Without sharing, passengers pay higher rates and drivers don’t earn money in between trips



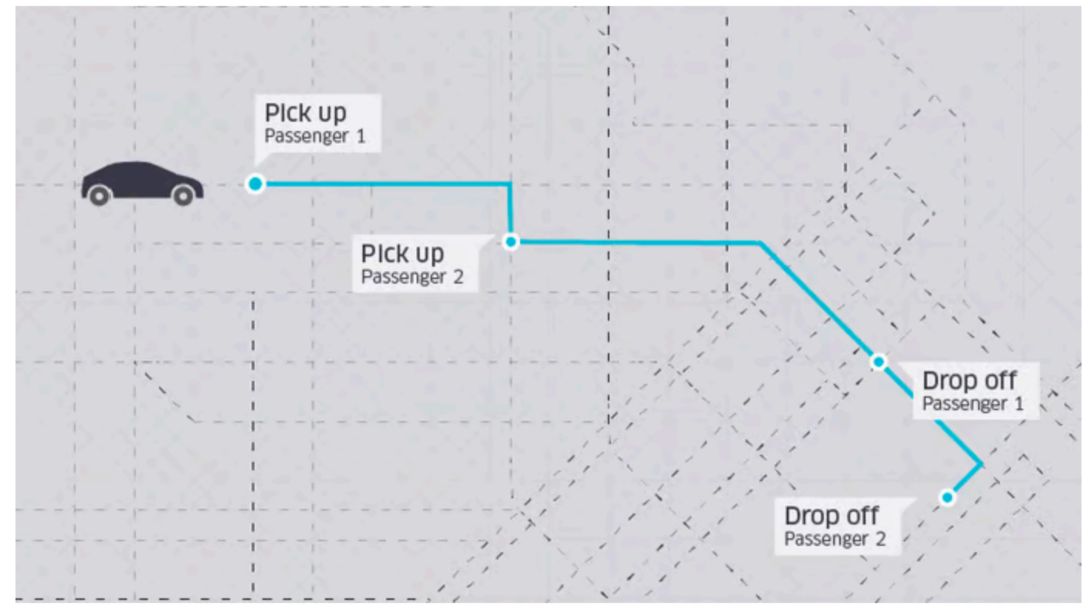
With sharing, passengers pay low rates and drivers earn money in between trips

Motivating example: On-demand car-pooling

- Rides requesting a ride along a similar route will share the ride and split the cost



Without sharing, passengers pay higher rates and drivers don't earn money in between trips



With sharing, passengers pay low rates and drivers earn money in between trips

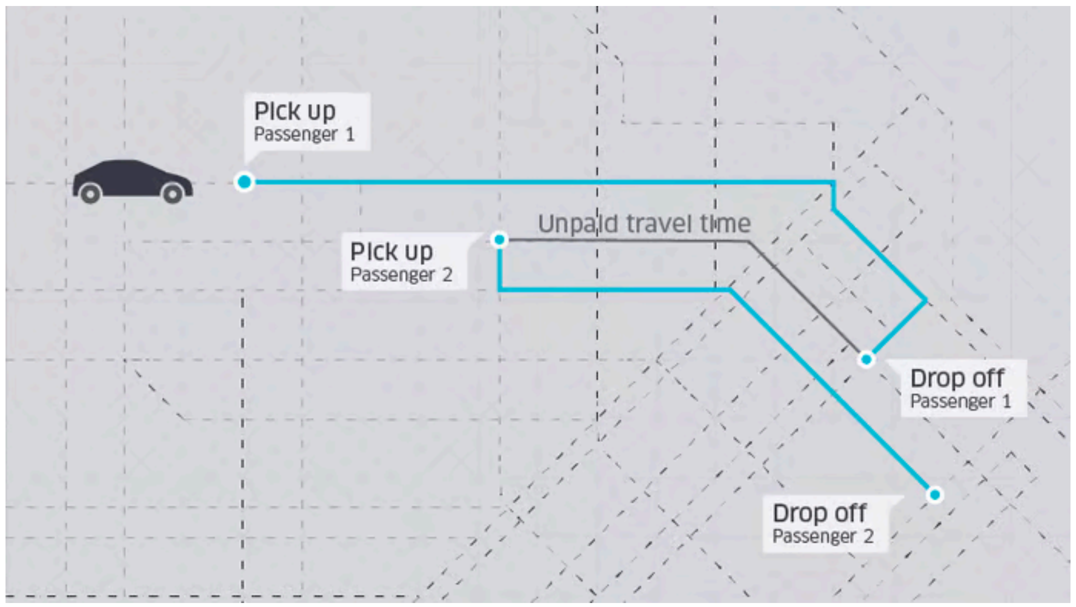
- Uber introduced Uber Pool in 2014
- Lyft introduced Share Saver in 2014

uberPOOL

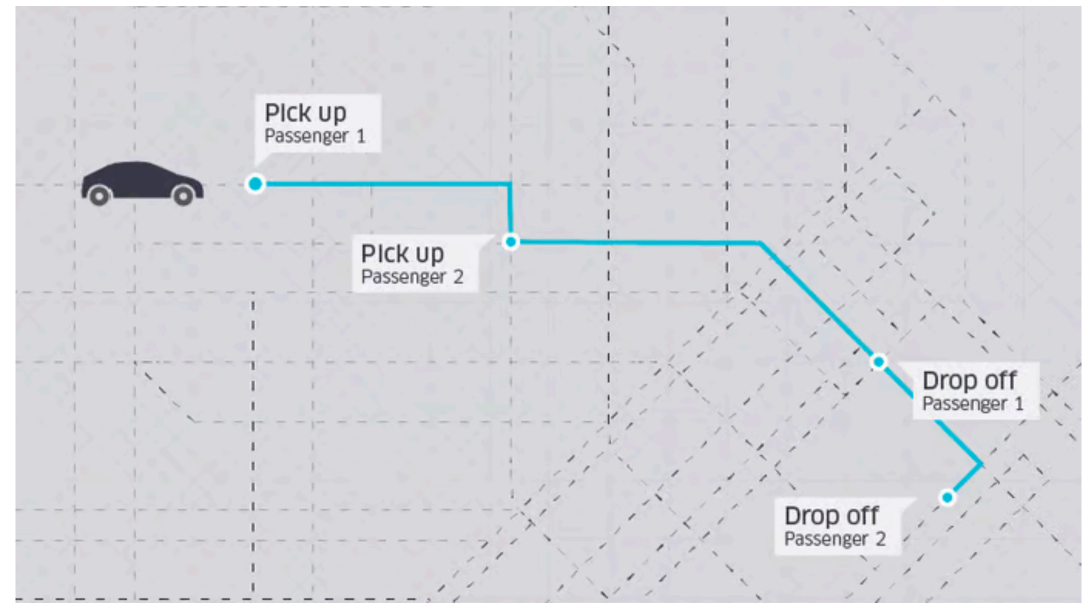
lyft
Shared rides

Motivating example: On-demand car-pooling

- Rides requesting a ride along a similar route will share the ride and split the cost



Without sharing, passengers pay higher rates and drivers don't earn money in between trips



With sharing, passengers pay low rates and drivers earn money in between trips

- Uber introduced Uber Pool in 2014
- Lyft introduced Share Saver in 2014

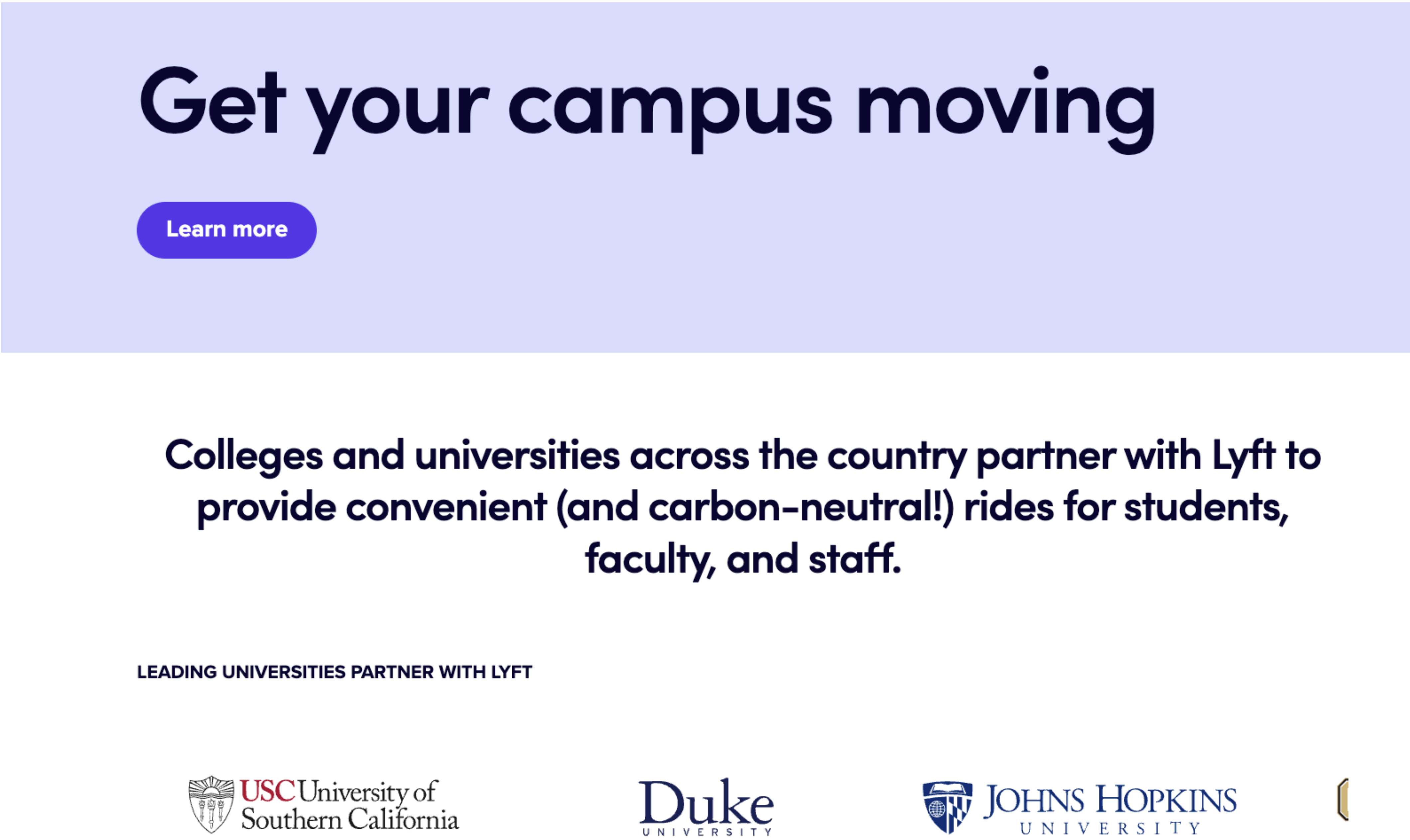
uberPOOL

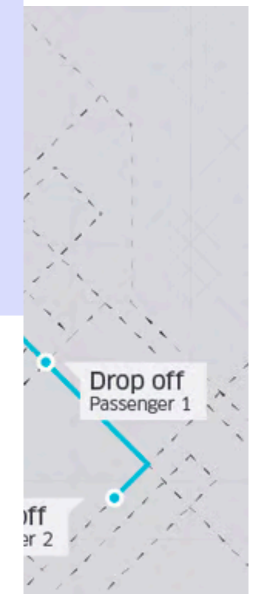
lyft Shared rides

- Great way to reduce congestion and emission while making ride-hailing affordable to more people

Motivating example: On-demand car-pooling

he cost





nd drivers

- Great way to reduce congestion and emission while making ride-hailing affordable to more people

Motivating example: On-demand car-pooling

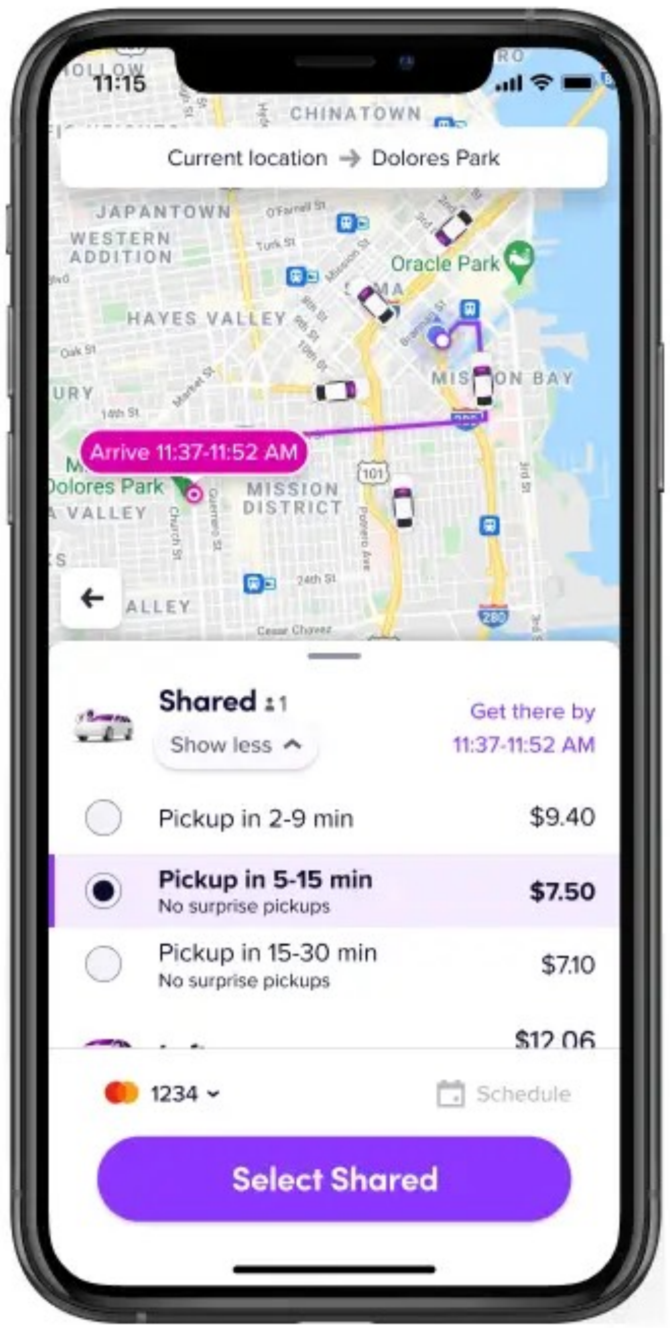
Society

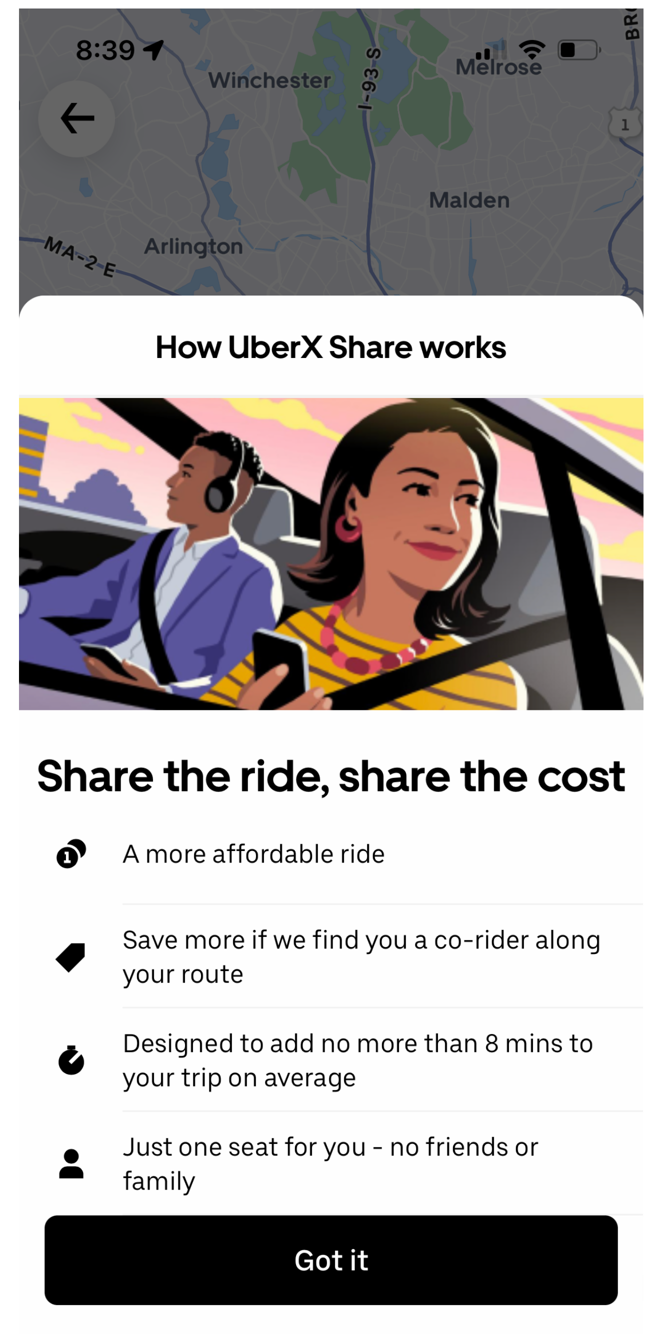
How a Shared Ride Led to Shared Love

Rev video - Jan 12, 2023

Motivating example: On-demand car-pooling

- After pandemic pause, Uber and Lyft revamped the car-pooling services





Motivating example: On-demand car-pooling

- **Face significant operational challenges: which to match and when?**

Motivating example: On-demand car-pooling

- **Face significant operational challenges: which to match and when?**
- Lyft discontinued shared rides in May 2023

"The problem with shared trips is that they take people out of their way"

- David Risher, CEO of Lyft

Motivating example: On-demand car-pooling

- **Face significant operational challenges: which to match and when?**
- Lyft discontinued shared rides in May 2023

“The problem with shared trips is that they take people out of their way”

- David Risher, CEO of Lyft

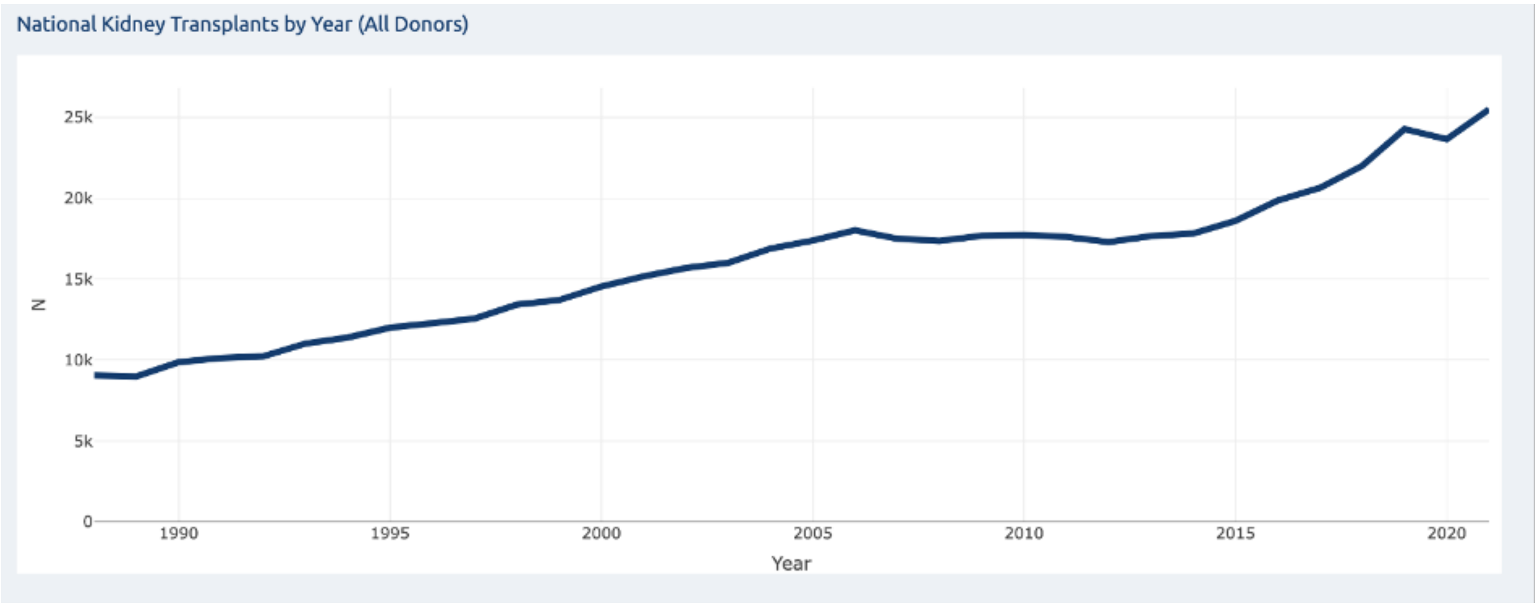
- Uber continues to expand UberX Share, as part of effort to achieve zero-emissions by 2030

“UberX share is deigned to operate with positive unit economics for Uber, while offering a lower-cost alternative to consumers… our newest billion-dollar gross bookings product in the coming quarters.”

- Dara Khosrowshahi, CEO of Uber

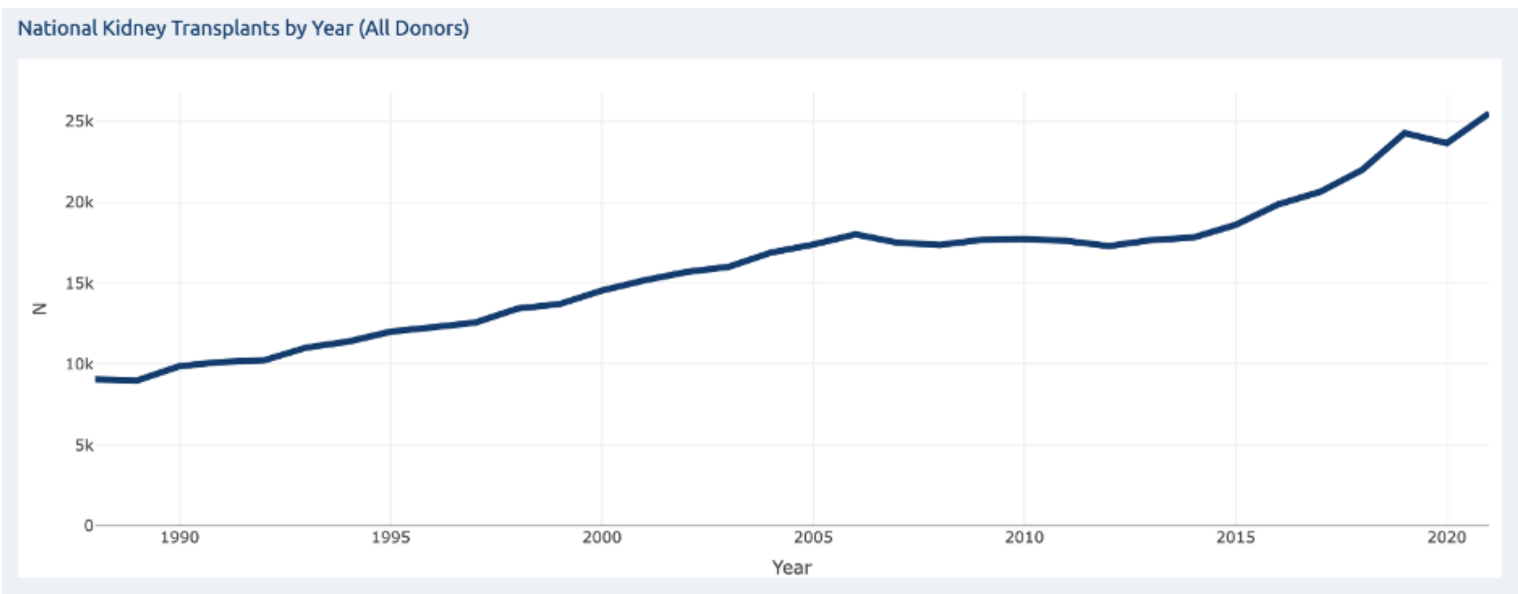
Motivating example: Kidney exchange

- Many people suffer from terminal kidney failure and are in dire need of a transplant. Some numbers from United States:
  - 100,000 patients needing a life-saving organ transplant (>50% is for kidney)
  - The median waiting time for a transplant is 5+ years.



Motivating example: Kidney exchange

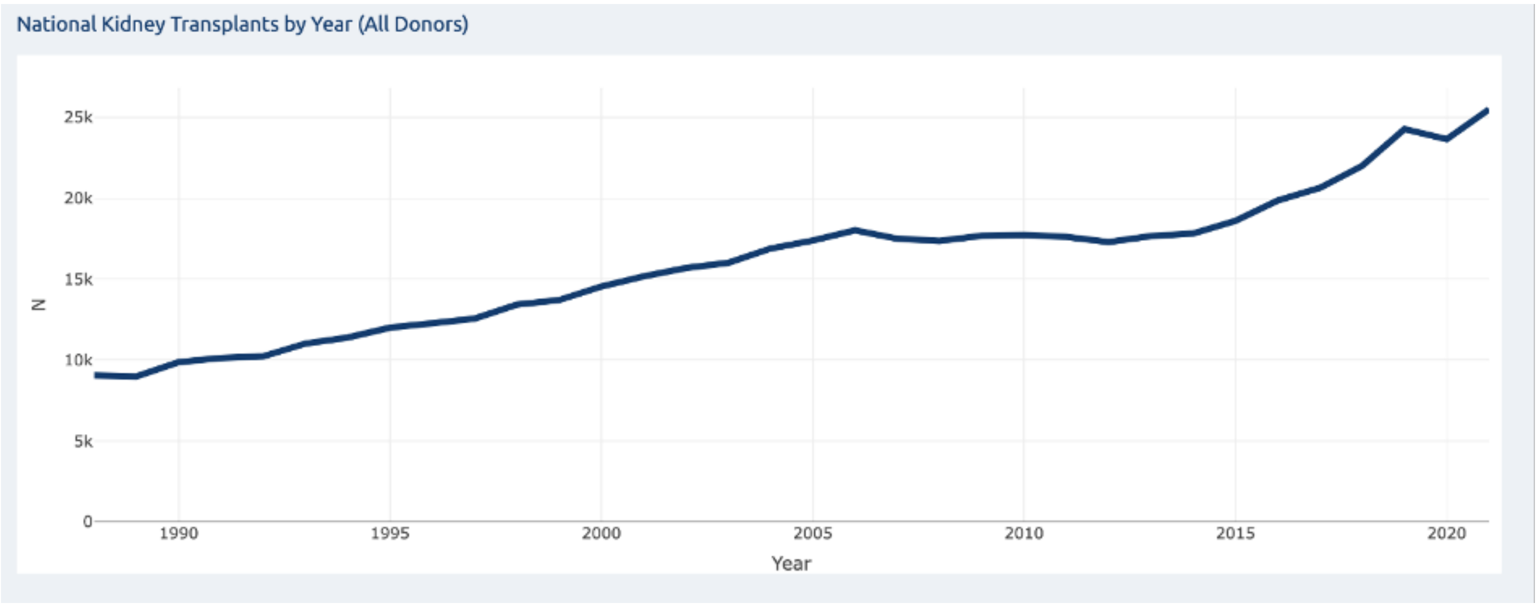
- Many people suffer from terminal kidney failure and are in dire need of a transplant. Some numbers from United States:
  - 100,000 patients needing a life-saving organ transplant (>50% is for kidney)
  - The median waiting time for a transplant is 5+ years.

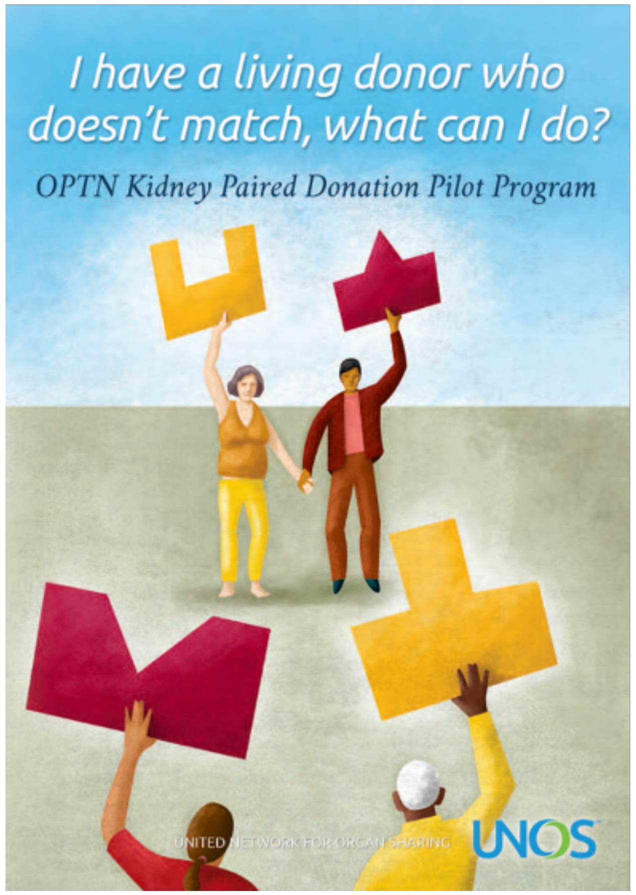


- Numerous patients have willing and able living donors (e.g., a spouse or sibling) but are incompatible with them, either because of blood type or tissue type (Human Leukocyte Antigens or "HLA") incompatibilities.

Motivating example: Kidney exchange

- Many people suffer from terminal kidney failure and are in dire need of a transplant. Some numbers from United States:
  - 100,000 patients needing a life-saving organ transplant (>50% is for kidney)
  - The median waiting time for a transplant is 5+ years.





- Numerous patients have willing and able living donors (e.g., a spouse or sibling) but are incompatible with them, either because of blood type or tissue type (Human Leukocyte Antigens or "HLA") incompatibilities.

- **Key operational challenge: which to match and when?**

Central challenge in dynamic matching

Matching efficiency vs. Waiting time

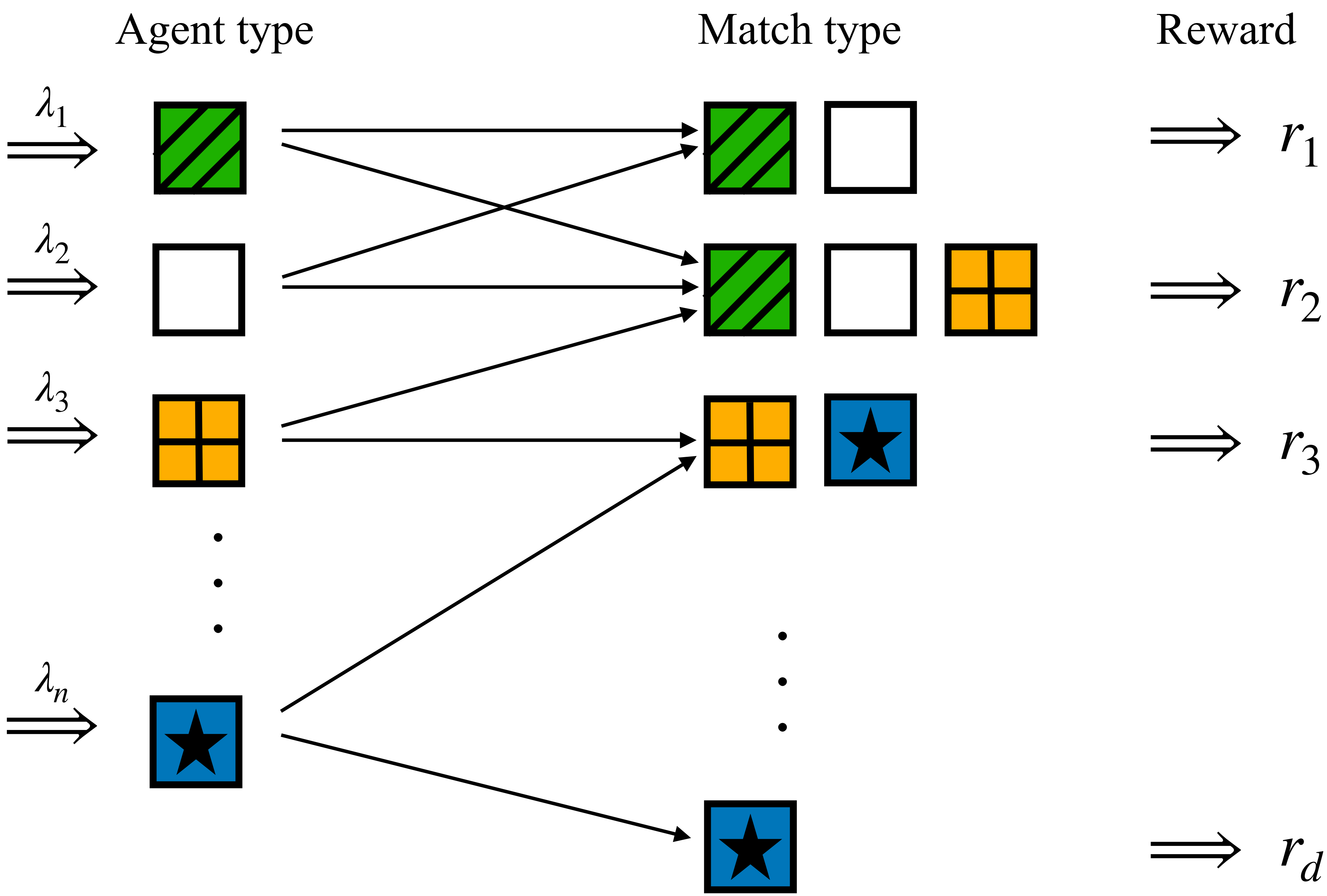
Central challenge in dynamic matching

Matching efficiency vs. Waiting time

Question: How to design an algorithm that attains high matching efficiency while incurring small waiting time?

- Practically simple
- Interpretable
- Provably optimal

Model setup on multi-way dynamic matching



At each time slot t
$\Longrightarrow$ One agent of a type arrives in the system
$\Longrightarrow$ Realize zero or one match type

Agents who arrived in the system only leave the system after they are matched.

Static Planning (Fluid) Solution

- Consider the static planning:

$$
\begin{aligned}
\max_{x} \ & \sum_{m} r_m x_m \\
\text{s.t.} \ & Mx = \lambda \\
& x_m \geq 0, \forall m
\end{aligned}
$$

Static Planning (Fluid) Solution

- Consider the static planning:

r_m is the reward of match m

$$\max_x \sum_m r_m x_m$$
$$\text{s.t. } Mx = \lambda$$
$$x_m \geq 0, \forall m$$

Static Planning (Fluid) Solution

- Consider the static planning:

$$\max_x \sum_m r_m x_m$$
$$\text{s.t. } Mx = \lambda$$
$$x_m \geq 0, \forall m$$

r_m is the reward of match m

where M denotes the matching matrix s.t. $M_{i,m} = 1$ if agent i belongs to match m

λ is the arrival rate vector

Static Planning (Fluid) Solution

- Consider the static planning:

r_m is the reward of match m

where M denotes the matching matrix s.t. $M_{i,m} = 1$ if agent i belongs to match m

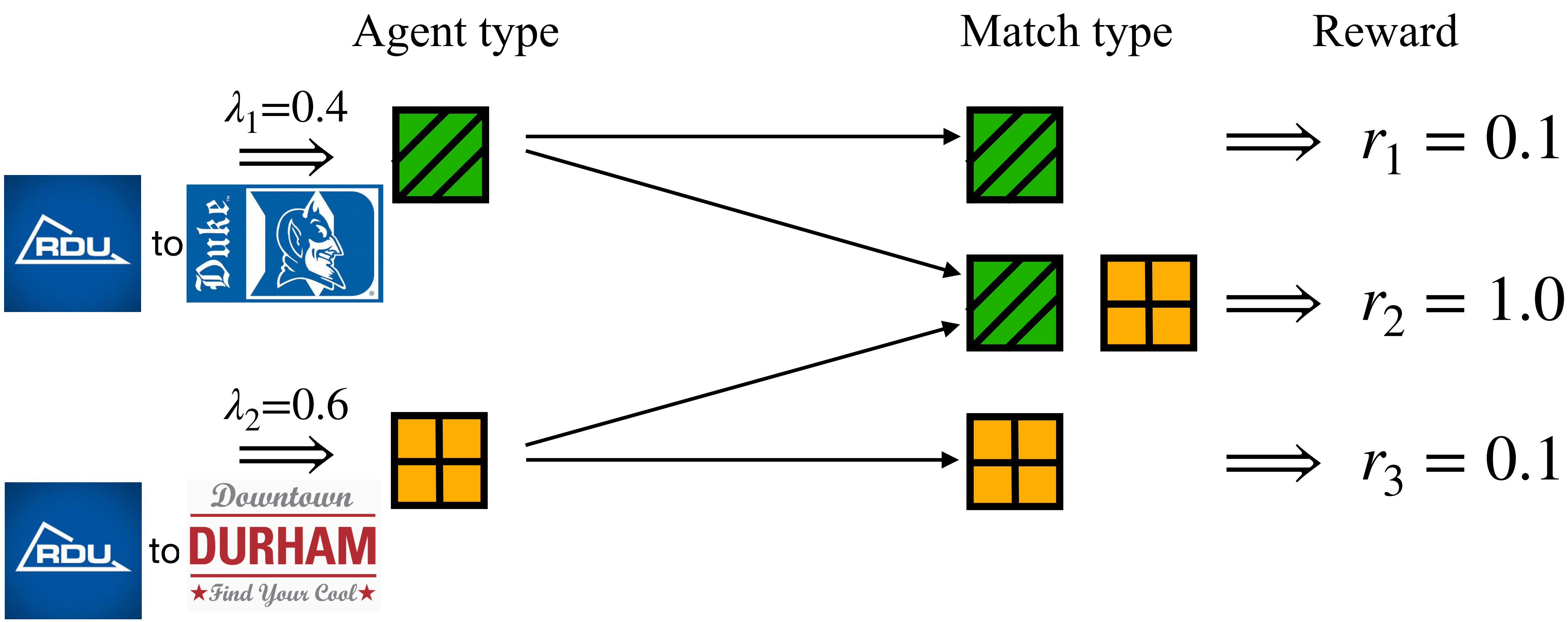
$$\max_x \sum_m r_m x_m$$
$$\text{s.t. } Mx = \lambda$$
$$x_m \geq 0, \forall m$$

λ is the arrival rate vector

- Let x^* denote the static planning solution
- Can we just match based on x^*?

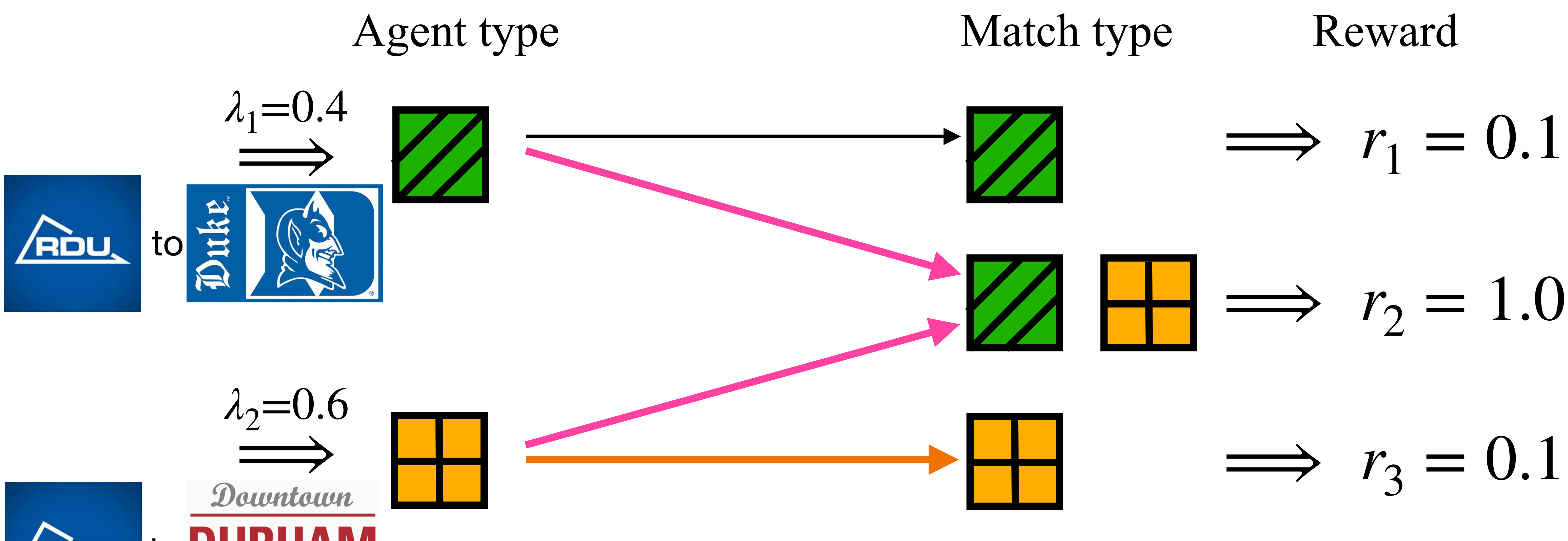
Failure of Static Planning with dynamic arrivals

- Consider the following simple example



Failure of Static Planning with dynamic arrivals

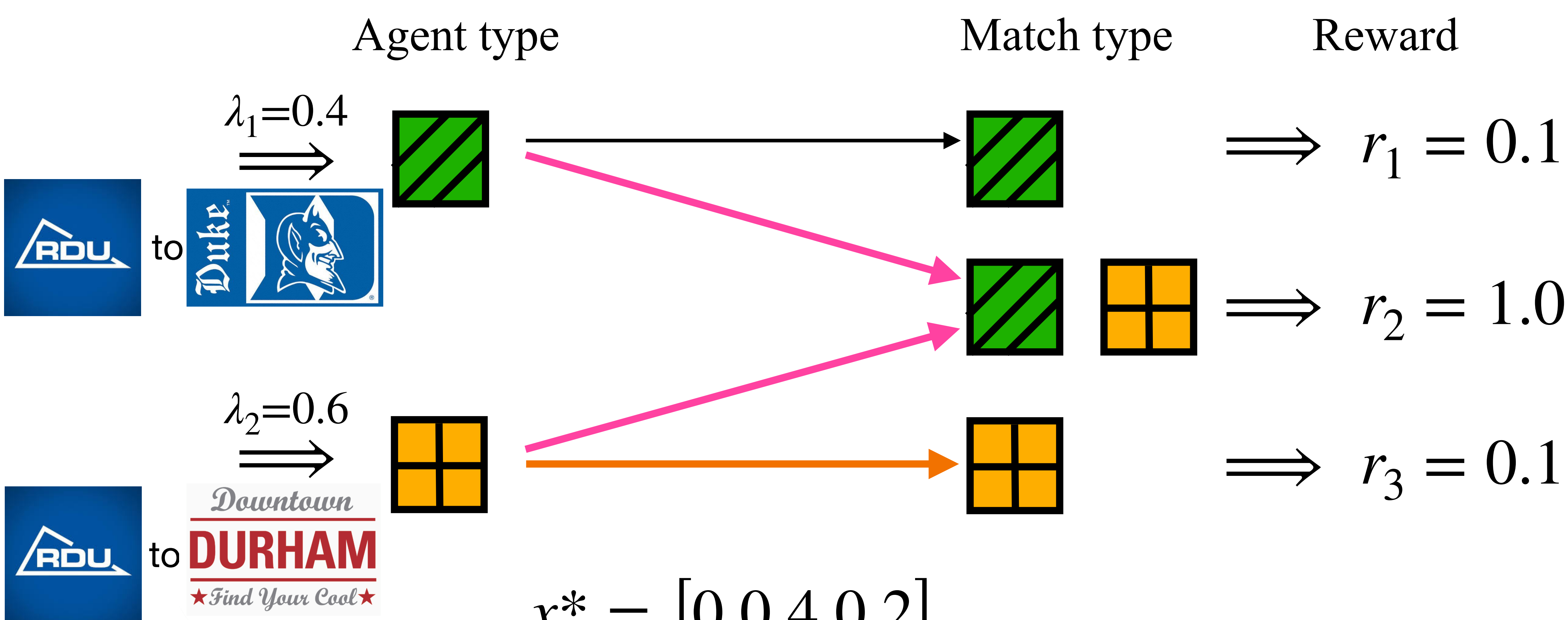
- Consider the following simple example



$$x^* = [0, 0.4, 0.2]$$

Failure of Static Planning with dynamic arrivals

- Consider the following simple example

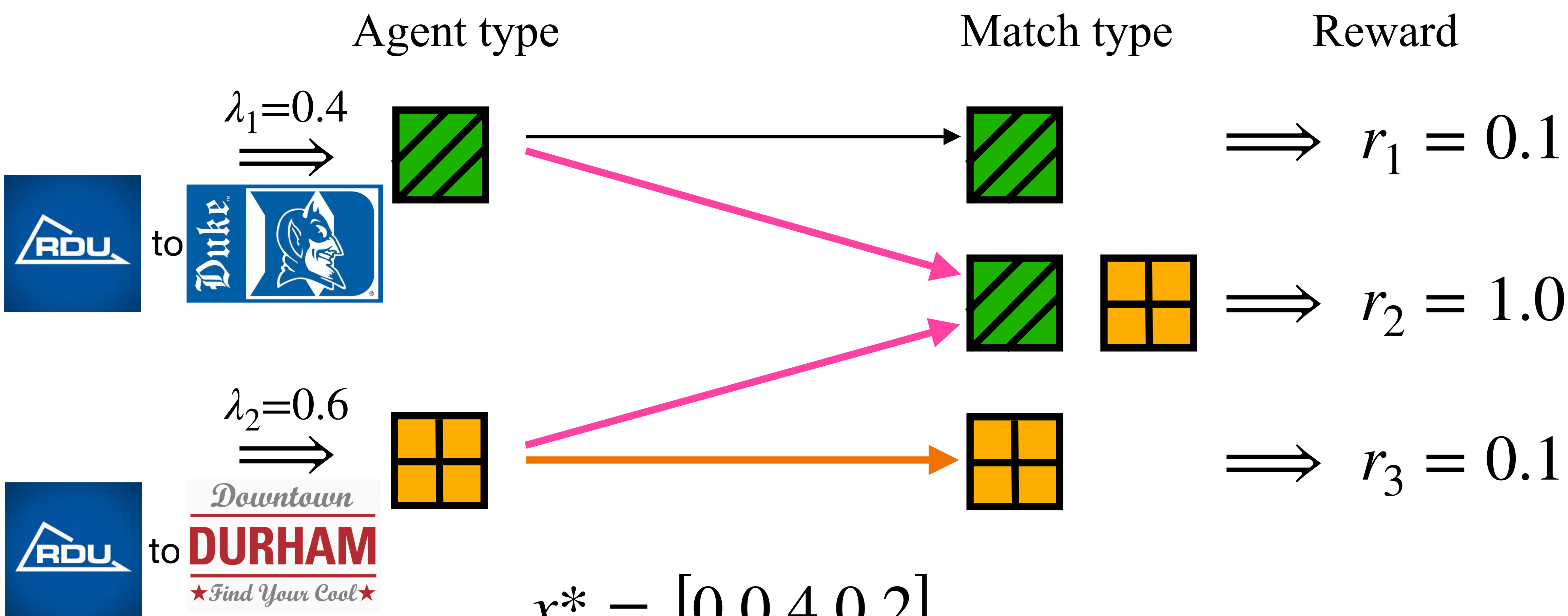


$$x^* = [0, 0.4, 0.2]$$

- Type-1 agents only get served with rate 0.4

Failure of Static Planning with dynamic arrivals

- Consider the following simple example

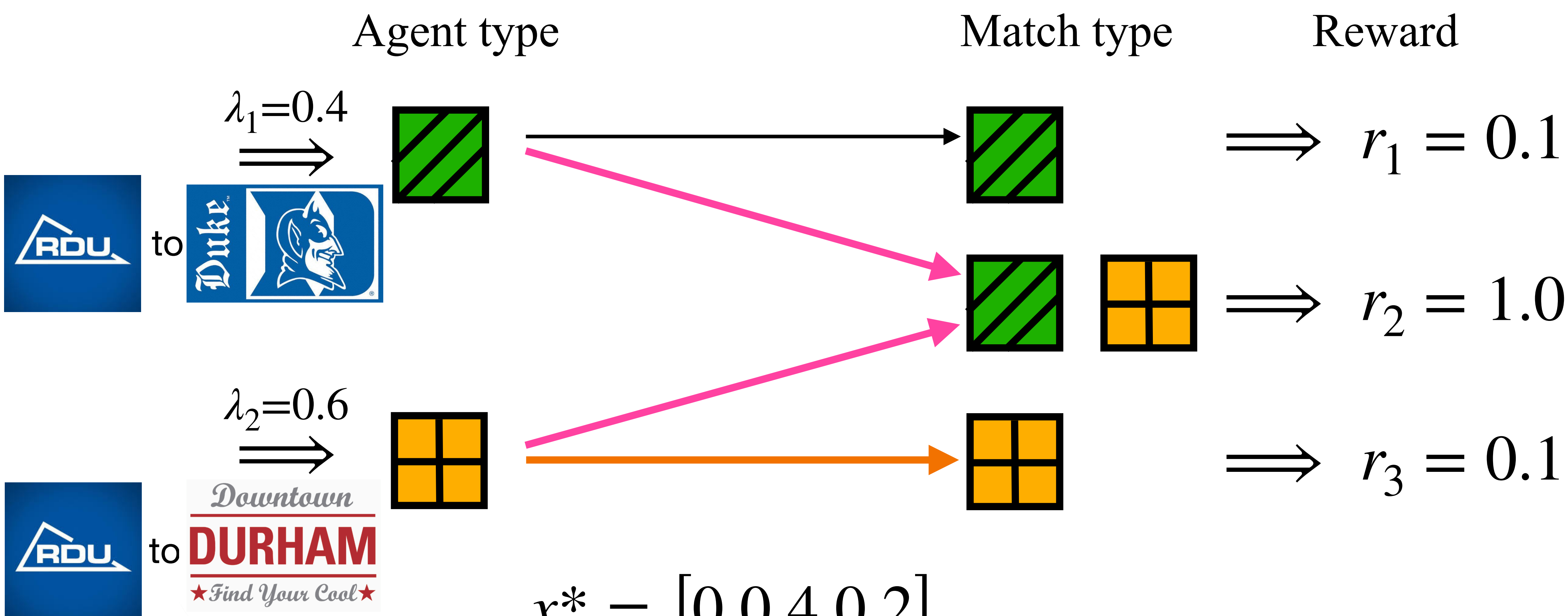


$$x^* = [0, 0.4, 0.2]$$

- Type-1 agents only get served with rate 0.4 $\Longrightarrow$ # of waiting type-1 agents grow to infinity

Failure of Static Planning with dynamic arrivals

- Consider the following simple example



$$x^* = [0,0.4,0.2]$$

- Type-1 agents only get served with rate 0.4 $\Longrightarrow$ # of waiting type-1 agents grow to infinity
- Flaw of Averages: "plans based on average conditions are wrong on average"

  matching are independent of states and cannot adapt to stochastic arrivals

Our solution: use a “signal” to guide matching

Shadow price as a signal

Primal problem

$$\max_x \sum_m r_m x_m$$

$$\text{s.t. } Mx = \lambda$$

$$x_m \geq 0, \forall m$$

Dual problem

$$\min_p \sum_i \lambda_i p_i$$

$$\text{s.t. } (M^\top p)_m \geq r_m, \forall m$$

p_i is the shadow price for type-i agent

sum of shadow prices for agent types in match m

Shadow price as a signal

Primal problem

$$\max_{x} \sum_{m} r_m x_m$$
$$\text{s.t. } Mx = \lambda$$
$$x_m \geq 0, \forall m$$

Dual problem

$$\min_{p} \sum_{i} \lambda_i p_i$$
$$\text{s.t. } (M^\top p)_m \geq r_m, \forall m$$

p_i is the shadow price for type-i agent

sum of shadow prices for agent types in match m

- Let p^* denote the optimal dual solution, also known as **shadow price** [Kantorovich, Koopmans 40's, …]
-

Shadow price as a signal

Primal problem

$$\max_x \sum_m r_m x_m$$
$$\text{s.t. } Mx = \lambda$$
$$x_m \geq 0, \forall m$$

Dual problem

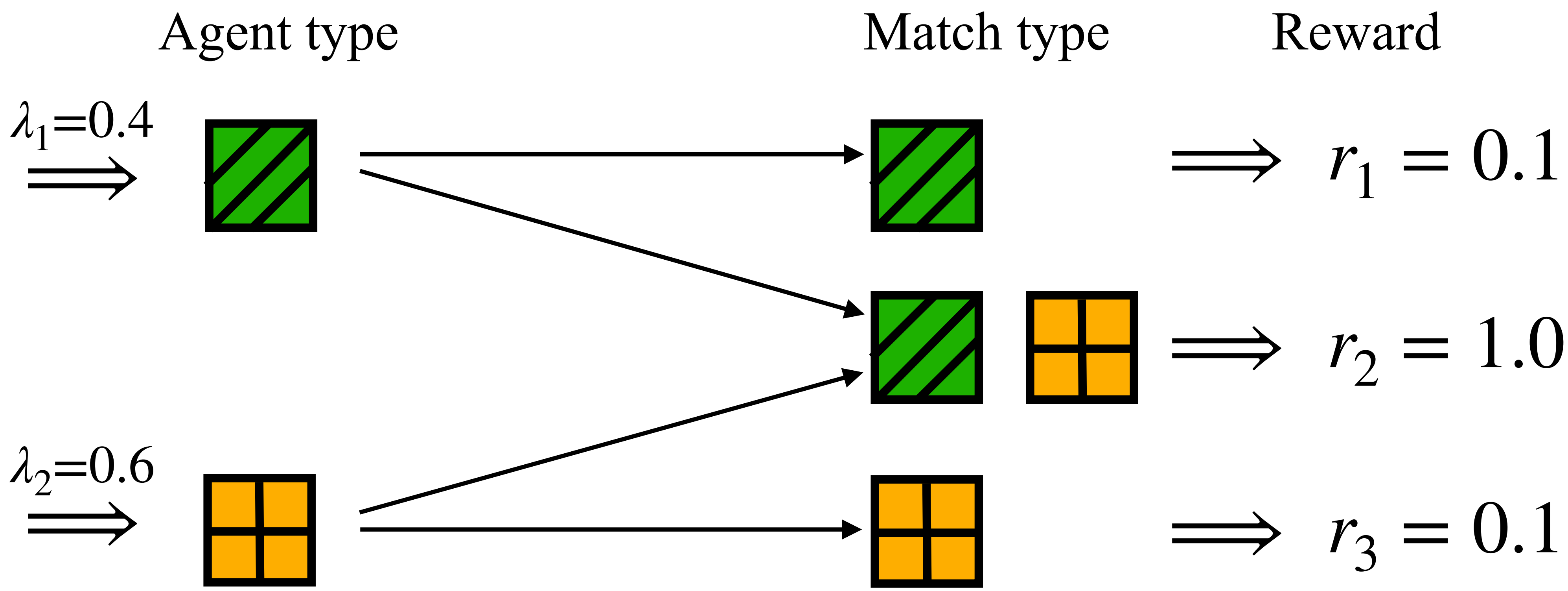
$$\min_p \sum_i \lambda_i p_i$$
$$\text{s.t. } (M^\top p)_m \geq r_m, \forall m$$

p_i is the shadow price for type-i agent

sum of shadow prices for agent types in match m

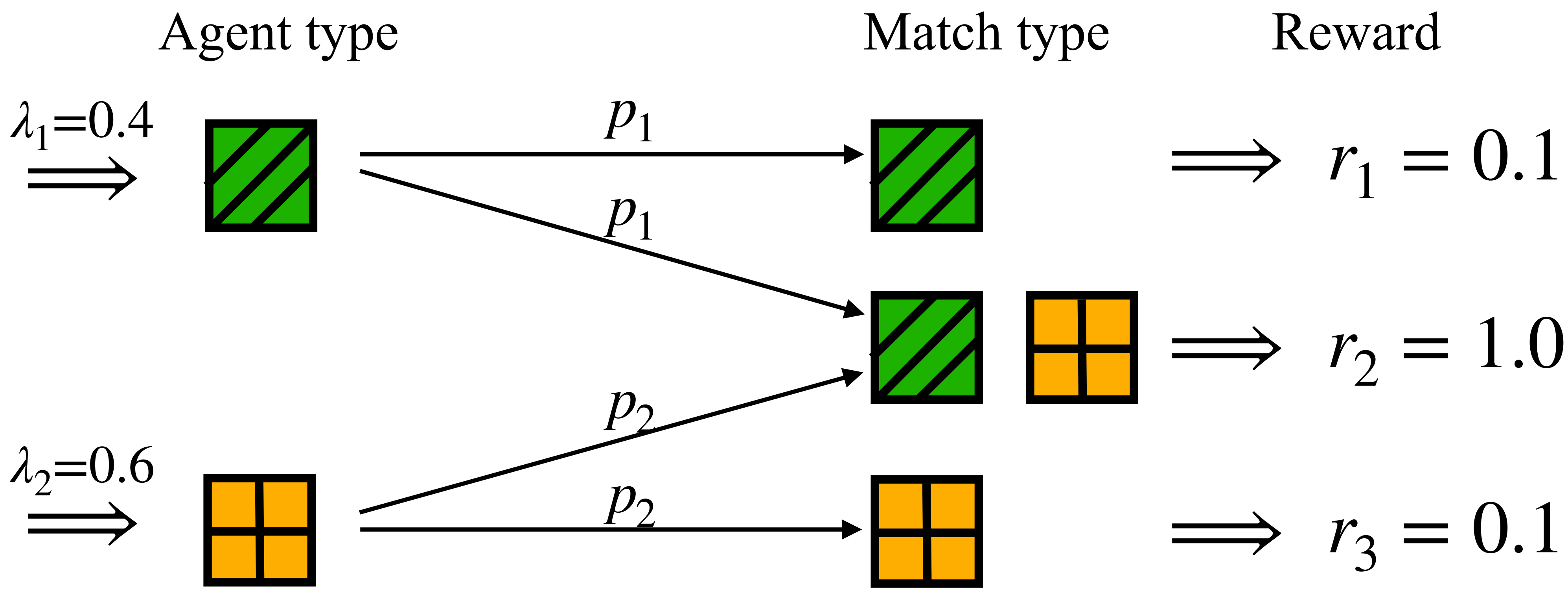
- Let p^* denote the optimal dual solution, also known as **shadow price** [Kantorovich, Koopmans 40's, …]
- A higher shadow price p_i^* means type-i agent resource is scarcer

Back to the example



$$x^* = [0,0.4,0.2]$$

Back to the example

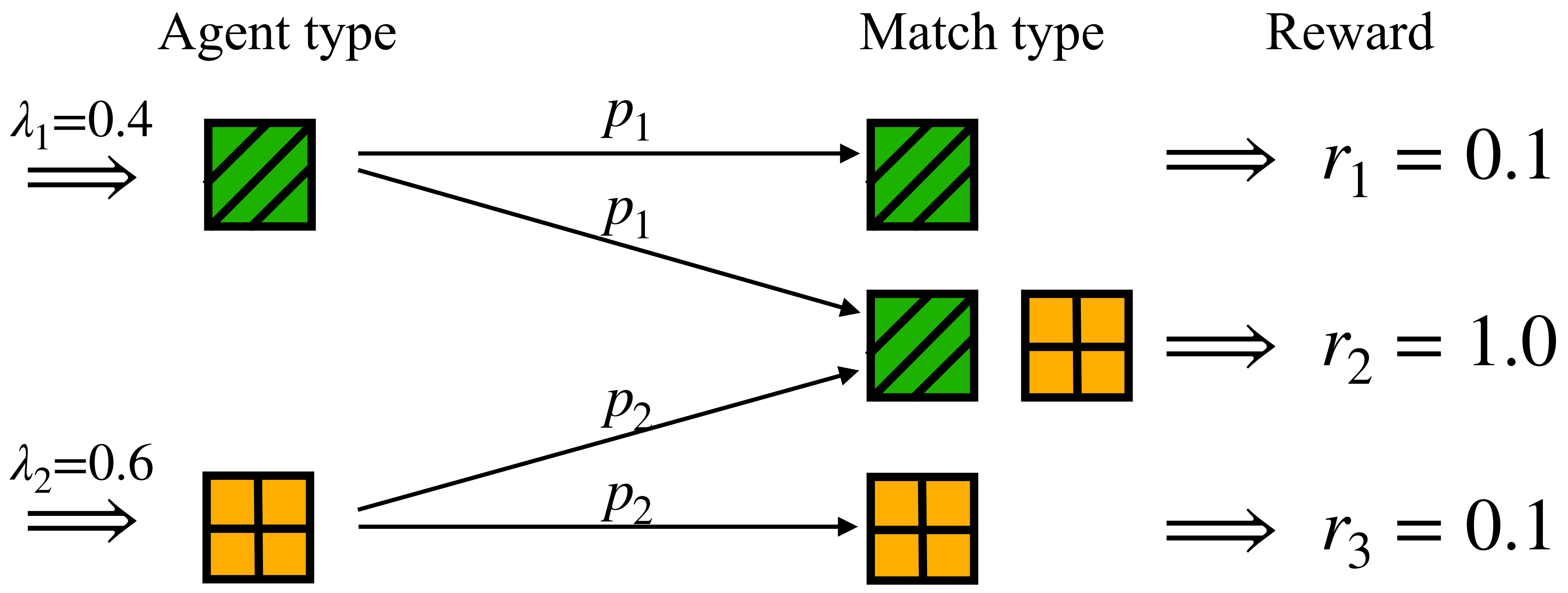


$$x^* = [0,0.4,0.2]$$

Dual

$$\begin{aligned} \min_p \; & 0.4p_1 + 0.6p_2 \\ \text{s.t.} \; & p_1 \geq 0.1 \\ & p_1 + p_2 \geq 1.0 \\ & p_2 \geq 0.1 \end{aligned}$$

Back to the example



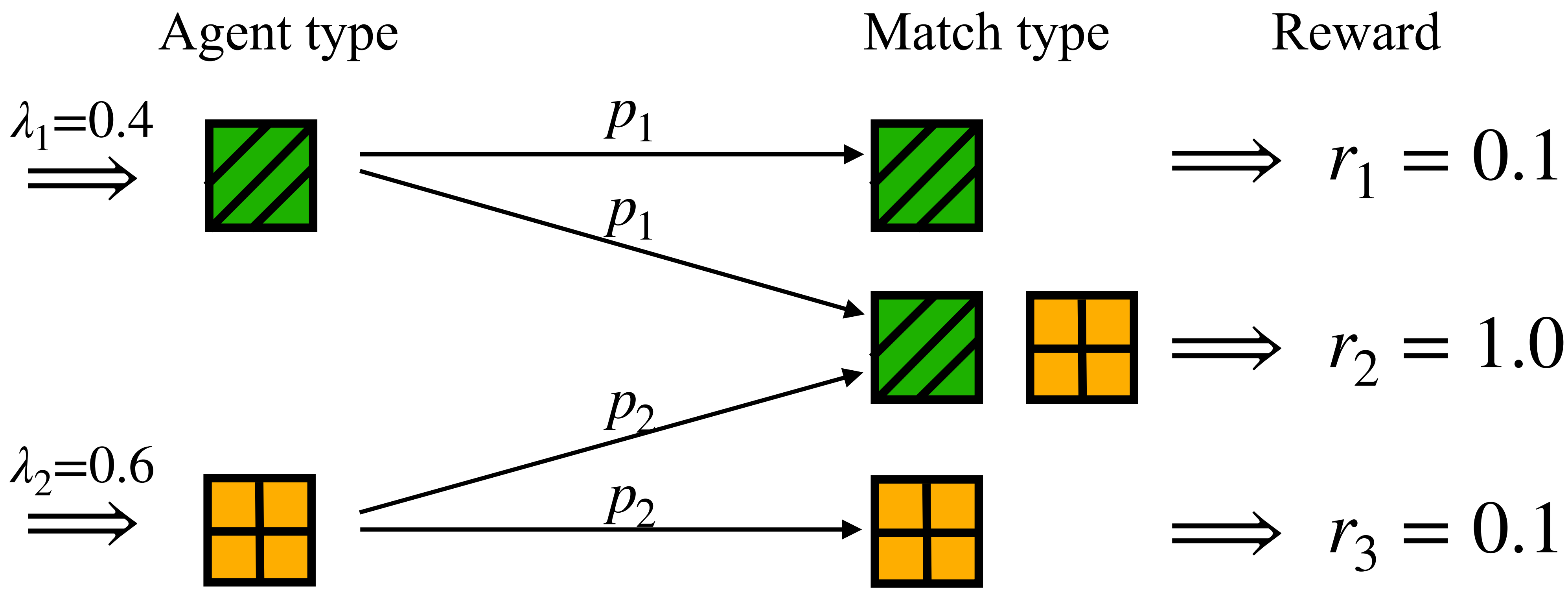
$$x^* = [0,0.4,0.2]$$

Dual

$$\begin{aligned} \min_{p} \; & 0.4p_1 + 0.6p_2 \\ \text{s.t.} \; & p_1 \geq 0.1 \\ & p_1 + p_2 \geq 1.0 \\ & p_2 \geq 0.1 \end{aligned}$$

$$\Longrightarrow \; p^* = [0.9,0.1]$$

Back to the example



$$x^* = [0, 0.4, 0.2]$$

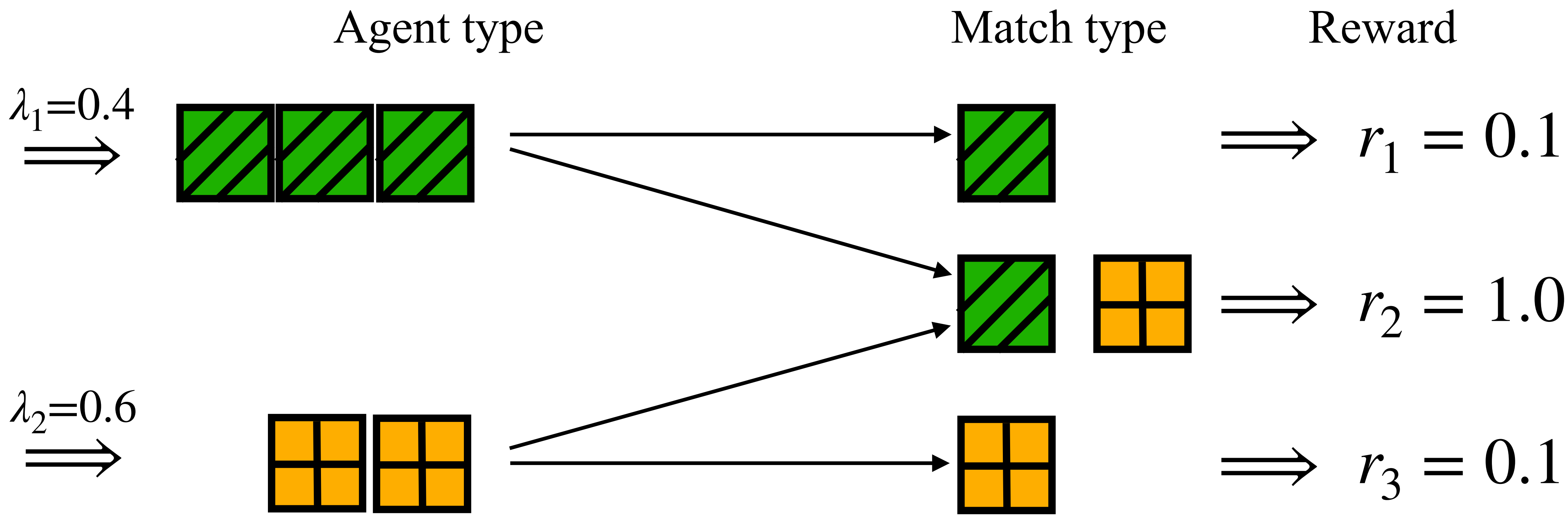
Dual

$$\begin{aligned} \min_{p} \; & 0.4p_1 + 0.6p_2 \\ \text{s.t.} \; & p_1 \geq 0.1 \\ & p_1 + p_2 \geq 1.0 \\ & p_2 \geq 0.1 \end{aligned}$$

Agent type 1 is more precious

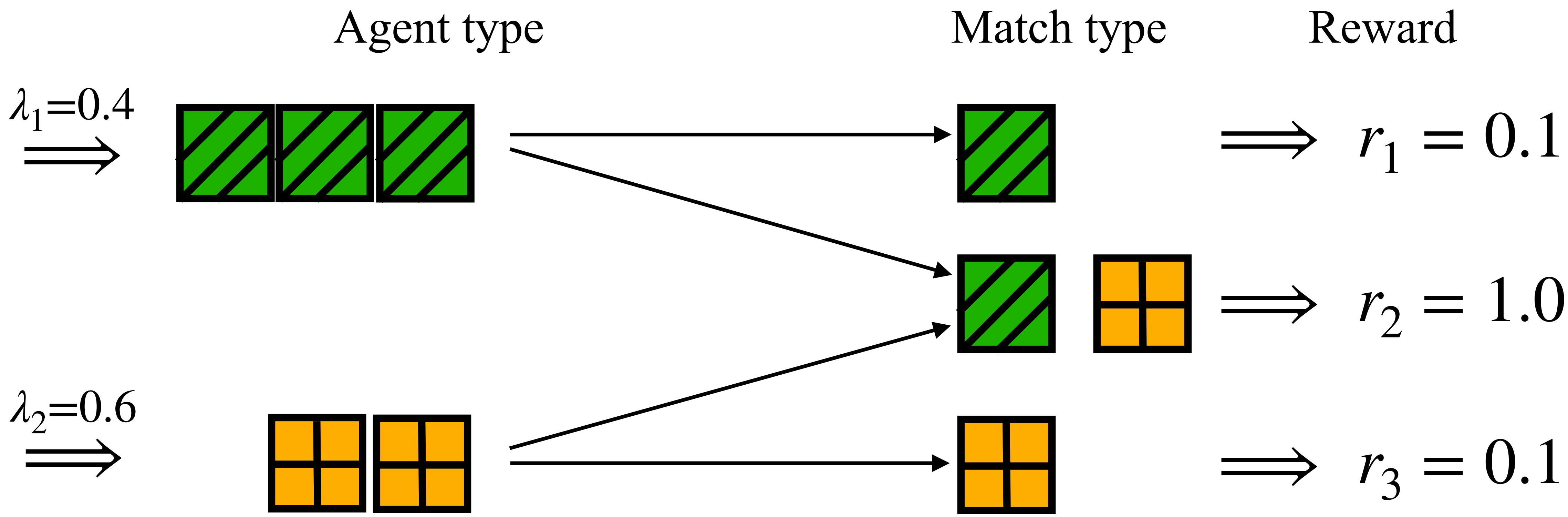
$$\Longrightarrow p^* = [0.9, 0.1]$$

How to dynamically adjust shadow price?



- If we have many type-1 agents waiting to be matched, it "costs" less to match them, so we should decrease its shadow price

How to dynamically adjust shadow price?

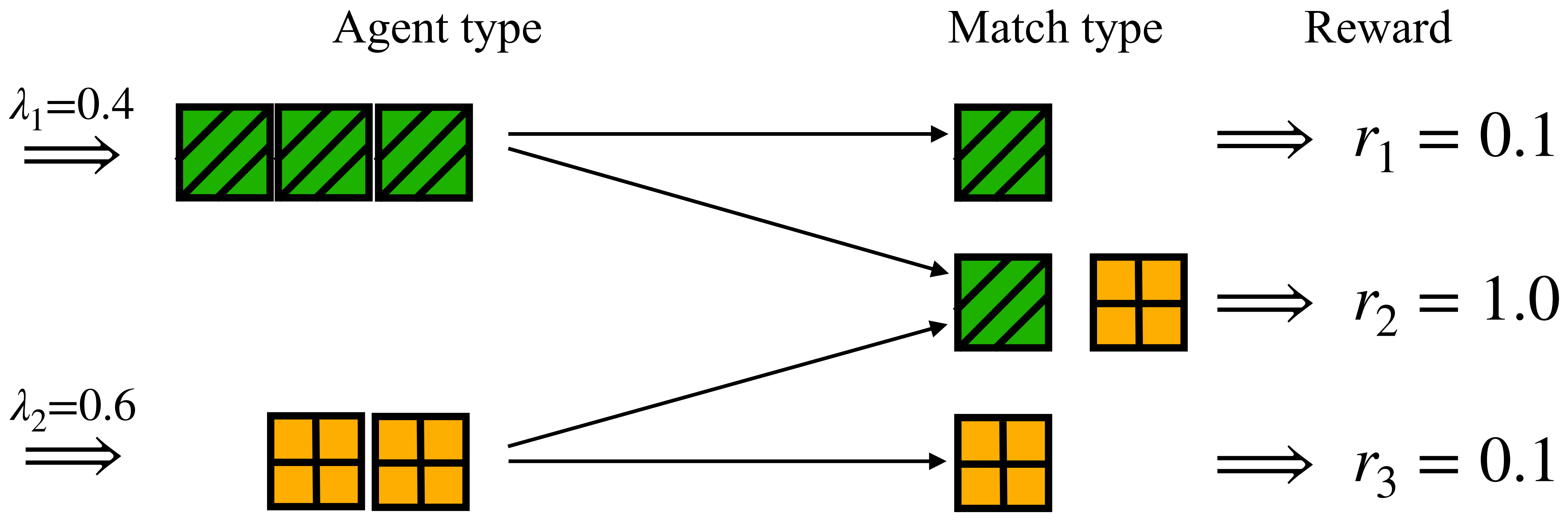


- If we have many type-1 agents waiting to be matched, it "costs" less to match them, so we should decrease its shadow price
- More formally, let $\delta_i(t)$ denote the # of type-i agents waiting to be matched

$$p_i(t) = p_i^* - \eta_t \times \delta_i(t)$$

where η_t is time-varying price elasticity.

How to dynamically adjust shadow price?



- If we have many type-1 agents waiting to be matched, it "costs" less to match them, so we should decrease its shadow price
- More formally, let $\delta_i(t)$ denote the # of type-i agents waiting to be matched

$$p_i(t) = p_i^* - \eta_t \times \delta_i(t)$$

where η_t is time-varying price elasticity.

- As time progresses, fluctuation of stochastic arrivals are amortized, and less adjustment needed $\Longrightarrow$ η_t decreases in t

Our policy (arrival rates λ are known)

- At each time slot t:
  - Find $$x(t) = \arg\max_{x \in \mathcal{X}} \sum_m r_m x_m - p(t)^T M x$$

$\Longleftrightarrow$ Schedule match m that has **maximal** **reduced reward**, i.e.,

$$\arg\max_m \left(r_m - \sum_{i \in \mathcal{A}(m)} p_i(t) \right)$$

Our policy (arrival rates λ are known)

- At each time slot t:
  - Find $x(t) = \arg\max_{x \in \mathcal{X}} \sum_m r_m x_m - p(t)^T M x$

    $\iff$ Schedule match m that has **maximal** **reduced reward**, i.e.,

    $$\arg\max_m \left(r_m - \sum_{i \in \mathcal{A}(m)} p_i(t) \right)$$

  - Update the inventory of each agent based on $x(t)$ and the arrival $A(t)$:

    $$\delta(t) = \delta(t-1) - Mx(t) + A(t)$$

Our policy (arrival rates λ are known)

- At each time slot t:
  - Find $$x(t) = \arg\max_{x \in \mathcal{X}} \sum_m r_m x_m - p(t)^T M x$$

    $\iff$ Schedule match m that has **maximal reduced reward**, i.e.,

    $$\arg\max_m \left(r_m - \sum_{i \in \mathcal{A}(m)} p_i(t) \right)$$

  - Update the inventory of each agent based on $x(t)$ and the arrival $A(t)$:

    $$\delta(t) = \delta(t-1) - Mx(t) + A(t)$$

  - Update the **shadow price for each agent type:**

    $$\boxed{p_i(t+1) = p_i^* - \eta_t \times \delta_i(t)}$$

p_i^* is the optimal dual solution

$\delta_i(t)$ is the inventory of agent i

η_t is time-varying price elasticity

Our policy (arrival rates λ are known)

Note: If the scheduled match m cannot be realized at time slot t, we will realize it in the future based on the first-come-first-serve policy

- At each time slot t:
  - Find $$x(t) = \arg\max_{x \in \mathcal{X}} \sum_m r_m x_m - p(t)^T M x$$

$\Longleftrightarrow$ Schedule match m that has **maximal reduced reward**, i.e.,

$$\arg\max_m \left(r_m - \sum_{i \in \mathcal{A}(m)} p_i(t) \right)$$

  - Update the inventory of each agent based on $x(t)$ and the arrival $A(t)$:

$$\delta(t) = \delta(t-1) - Mx(t) + A(t)$$

  - Update the **shadow price for each agent type:**

$$\boxed{p_i(t+1) = p_i^* - \eta_t \times \delta_i(t)}$$

p_i^* is the optimal dual solution

$\delta_i(t)$ is the inventory of agent i

η_t is time-varying price elasticity

What if arrival rates λ are unknown or changing?

- Plug in the estimate of p^* s.t.

$$\boxed{p_i(t+1) = p_i^* - \eta_t \times \delta_i(t)}$$

What if arrival rates λ are unknown or changing?

- Plug in the estimate of p^* s.t.

$$\boxed{p_i(t+1) = \hat{p}_i(t) - \eta_t \times \delta_i(t)}$$

where $\hat{p}(t)$ denotes the optimal dual under the empirical arrival rate $\hat{\lambda}(t)$:

$$\hat{p}(t) \in \arg\min_{p} \sum_i \hat{\lambda}_i(t) p_i$$
$$\text{s.t. } (M^\top p)_m \geq r_m, \forall m$$

What if arrival rates λ are unknown or changing?

- Plug in the estimate of p^* s.t.

$$\boxed{p_i(t+1) = \hat{p}_i(t) - \eta_t \times \delta_i(t)}$$

where $\hat{p}(t)$ denotes the optimal dual under the empirical arrival rate $\hat{\lambda}(t)$:

$$\hat{p}(t) \in \arg\min_{p} \sum_i \hat{\lambda}_i(t) p_i$$
$$\text{s.t. } (M^\top p)_m \geq r_m, \forall m$$

- The rest of the algorithm is exactly the same as before

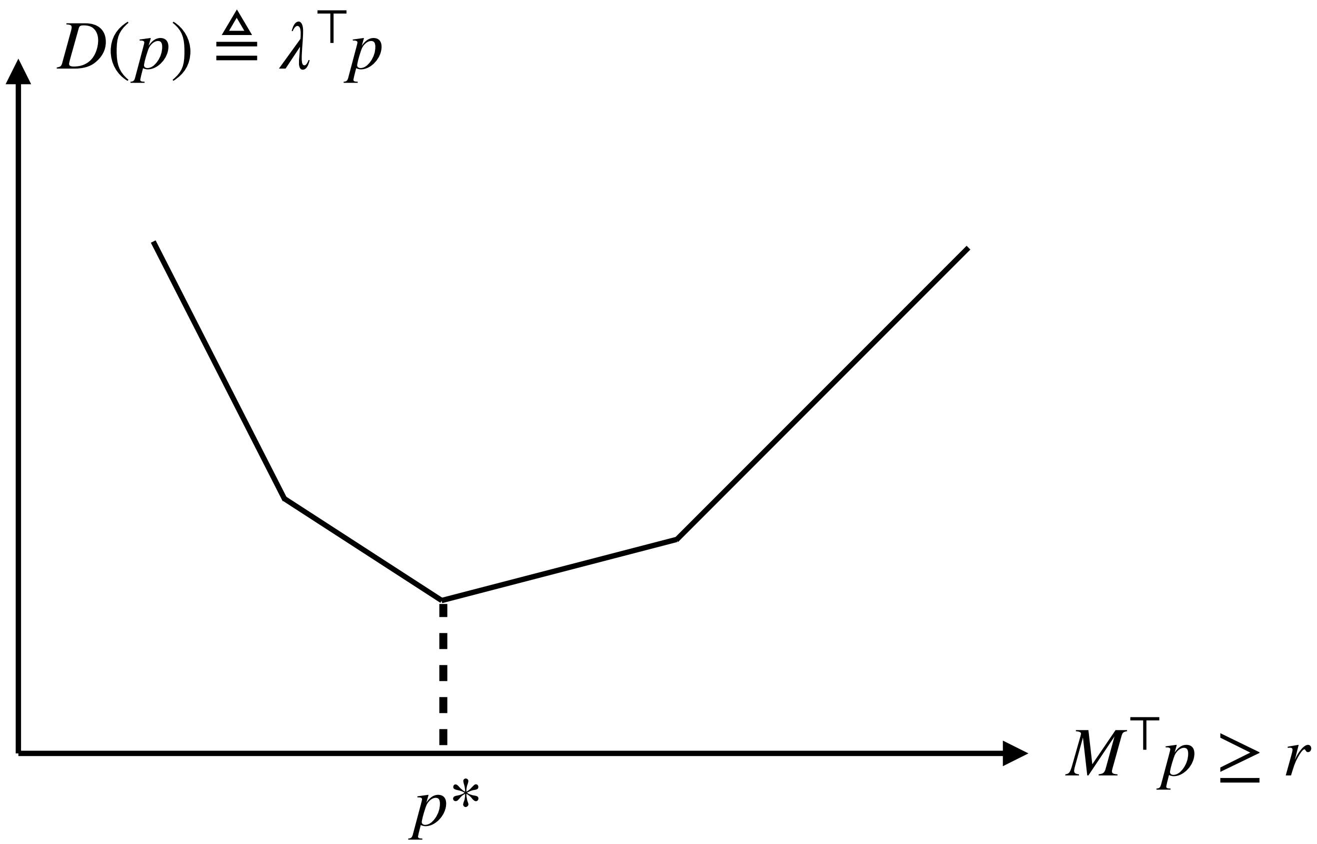
Our theory

Our policy achieves constant regret

Non-degeneracy assumption: For all $\|\hat{\lambda} - \lambda\|_2 \leq \epsilon$, optimal dual p^* **remains unchanged**

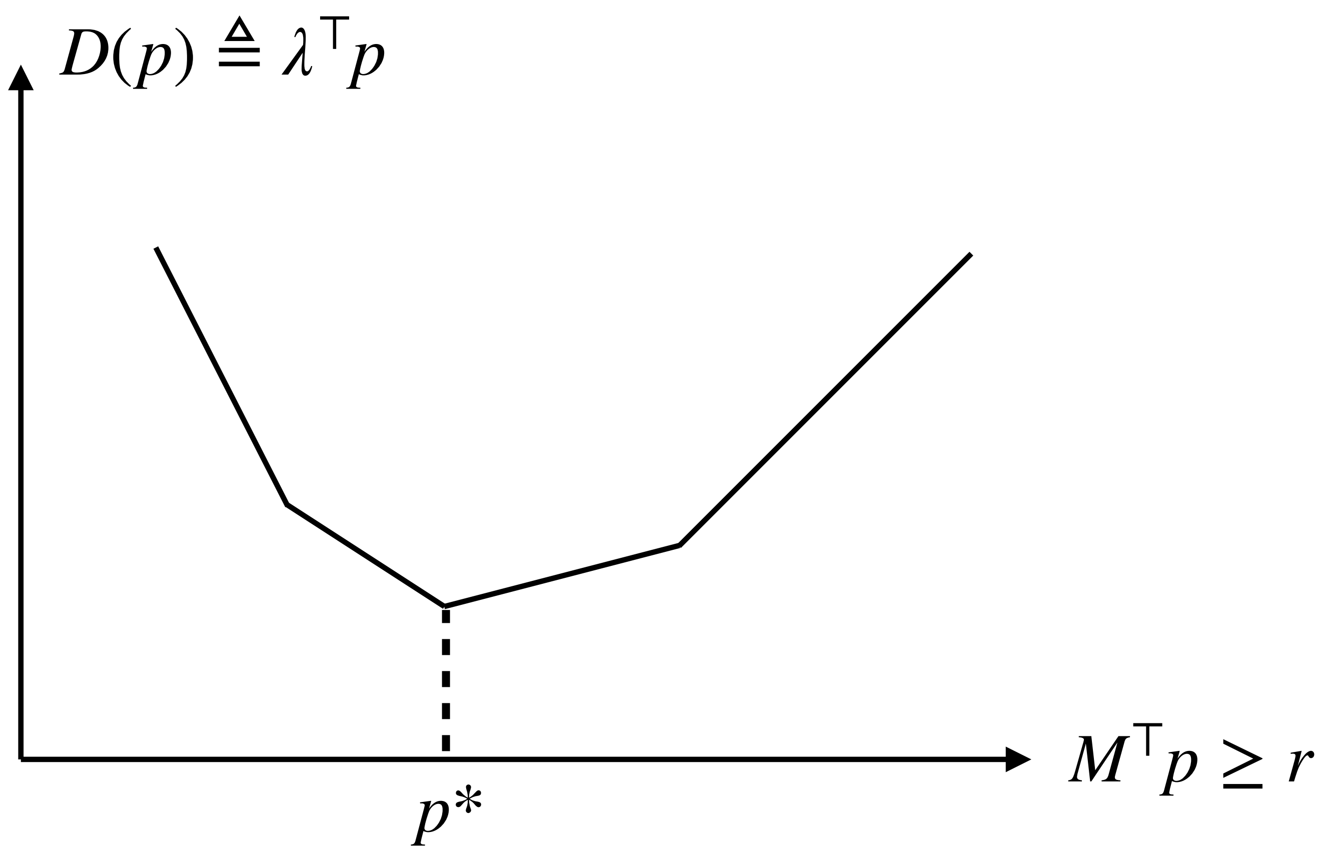
Our policy achieves constant regret

Non-degeneracy assumption: For all $\|\hat{\lambda} - \lambda\|_2 \leq \epsilon$, optimal dual p^* **remains unchanged**



Our policy achieves constant regret

Non-degeneracy assumption: For all $\|\hat{\lambda} - \lambda\|_2 \leq \epsilon$, optimal dual p^* **remains unchanged**



Local polyhedral property: slope at p^* is always bounded below by ϵ

$$D(p) - D(p^*) \geq \epsilon \|p - p^*\|_2$$

Our policy achieves constant regret

Assumption: For all $\|\hat{\lambda} - \lambda\|_2 \leq \epsilon$, the optimal dual p^* **remains unchanged**

Theorem [Wei-Xu-Yu '23]

Choose η_t to be decreasing so that $\sum_t \eta_t < \infty$. Then our primal-dual policy π achieves

$$\sup_{0 \leq t \leq T} \mathbb{E}\left[R_t^* - R_t^\pi\right] \leq \begin{cases} O\left(\frac{1}{\epsilon}\right) & \text{known } \lambda \\ O\left(\frac{1}{\epsilon^2}\right) & \text{unknown } \lambda \end{cases}$$

- R_t^* and R_t^π represent the cumulative rewards under the hindsight optimal policy and our policy, respectively

Our policy achieves constant regret

Assumption: For all $\|\hat{\lambda} - \lambda\|_2 \le \epsilon$, the optimal dual p^* **remains unchanged**

Theorem [Wei-Xu-Yu '23]

Choose η_t to be decreasing so that $\sum_t \eta_t < \infty$. Then our primal-dual policy π achieves

$$\sup_{0 \le t \le T} \mathbb{E}\left[R_t^* - R_t^\pi\right] \le \begin{cases} O\left(\frac{1}{\epsilon}\right) & \text{known } \lambda \\ O\left(\frac{1}{\epsilon^2}\right) & \text{unknown } \lambda \end{cases}$$

Regret does not grow with T

- R_t^* and R_t^π represent the cumulative rewards under the hindsight optimal policy and our policy, respectively

Our policy achieves constant regret

Assumption: For all $\|\hat{\lambda} - \lambda\|_2 \leq \epsilon$, the optimal dual p^* **remains unchanged**

Theorem [Wei-Xu-Yu '23]

Choose η_t to be decreasing so that $\sum_t \eta_t < \infty$. Then our primal-dual policy π achieves

$$\sup_{0 \leq t \leq T} \mathbb{E}\left[R_t^* - R_t^\pi\right] \leq \begin{cases} O\left(\frac{1}{\epsilon}\right) & \text{known } \lambda \\ O\left(\frac{1}{\epsilon^2}\right) & \text{unknown } \lambda \end{cases}$$

Regret does not grow with T

- R_t^* and R_t^π represent the cumulative rewards under the hindsight optimal policy and our policy, respectively
- We can also show the expected total waiting times satisfy the same bounds

Our policy achieves constant regret

Assumption: For all $\|\hat{\lambda} - \lambda\|_2 \leq \epsilon$, the optimal dual p^* **remains unchanged**

Theorem [Wei-Xu-Yu '23]

Choose η_t to be decreasing so that $\sum_t \eta_t < \infty$. Then our primal-dual policy π achieves

$$\sup_{0 \leq t \leq T} \mathbb{E}\left[R_t^* - R_t^\pi\right] \leq \begin{cases} O\left(\frac{1}{\epsilon}\right) & \text{known } \lambda \\ O\left(\frac{1}{\epsilon^2}\right) & \text{unknown } \lambda \end{cases}$$

Regret does not grow with T

- R_t^* and R_t^π represent the cumulative rewards under the hindsight optimal policy and our policy, respectively
- We can also show the expected total waiting times satisfy the same bounds
- $O(1/\epsilon)$ regret is the best possible [Kerimov-Ashlagi-Gurvich' 22a]

Our policy achieves constant regret

Assumption: For all $\|\hat{\lambda} - \lambda\|_2 \leq \epsilon$, the optimal dual p^* **remains unchanged**

Theorem [Wei-Xu-Yu '23]

Choose η_t to be decreasing so that $\sum_t \eta_t < \infty$. Then our primal-dual policy π achieves

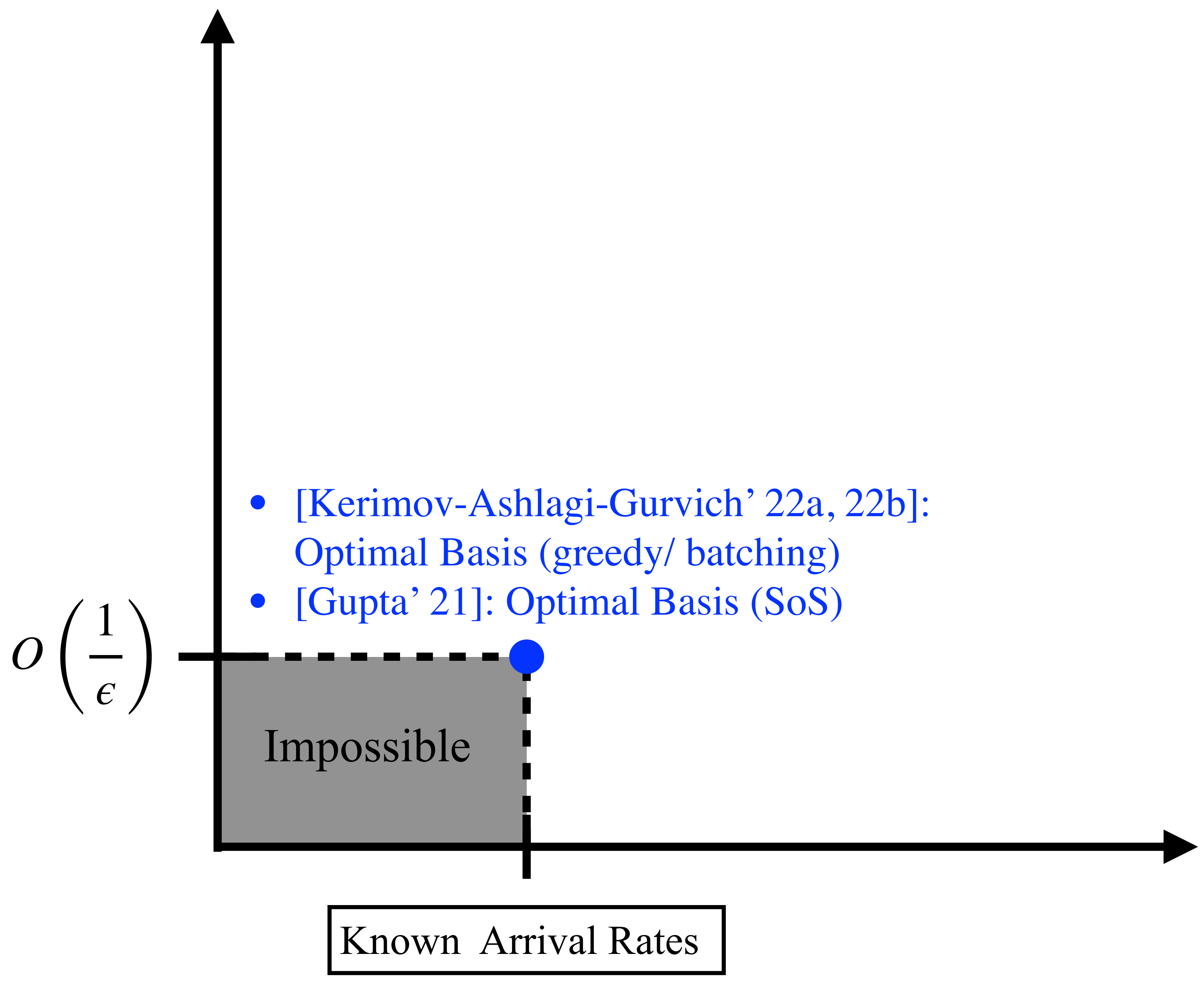
$$\sup_{0 \leq t \leq T} \mathbb{E}\left[R_t^* - R_t^\pi\right] \leq \begin{cases} O\left(\frac{1}{\epsilon}\right) & \text{known } \lambda \\ O\left(\frac{1}{\epsilon^2}\right) & \text{unknown } \lambda \end{cases}$$

Regret does not grow with T

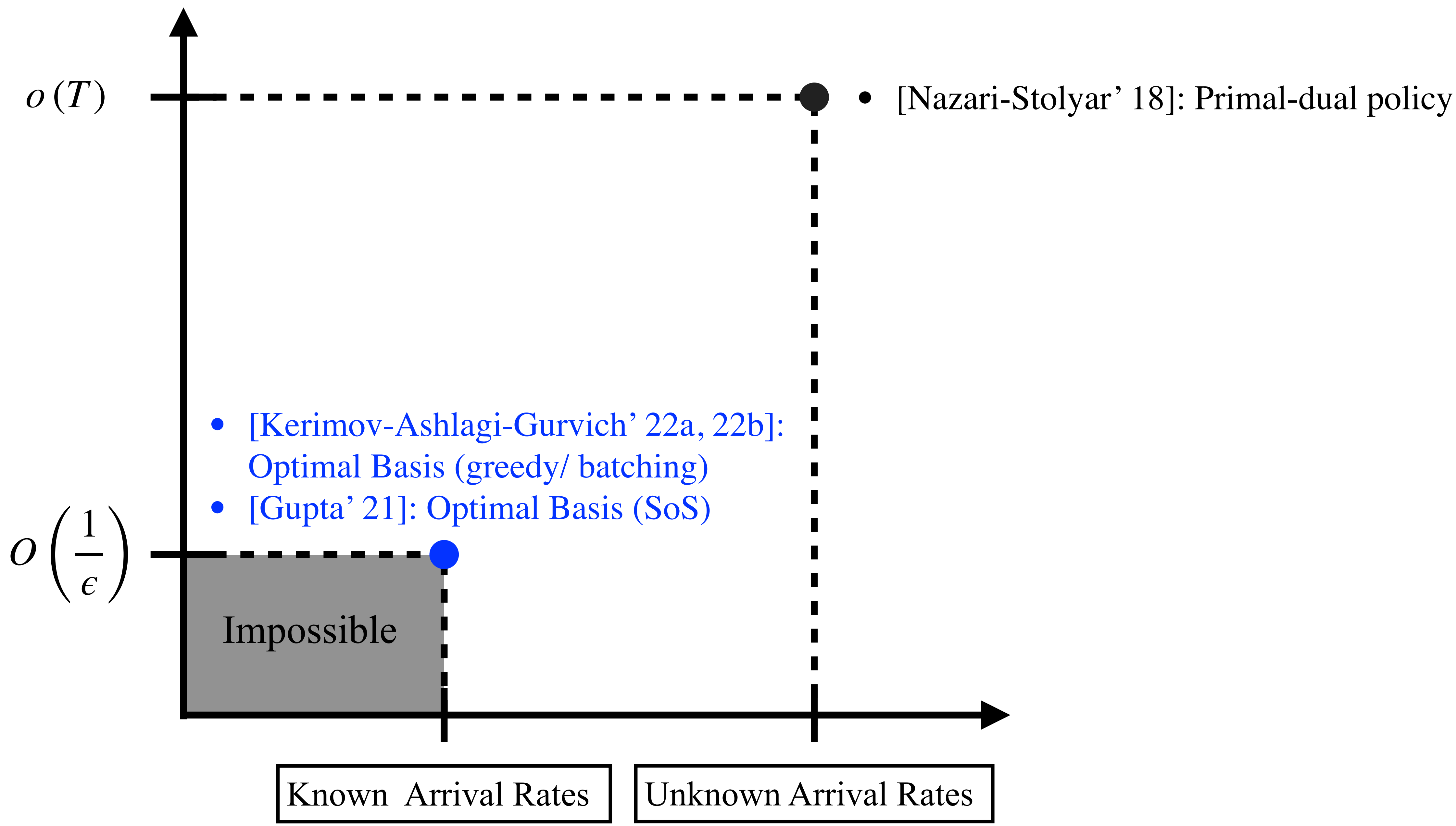
- R_t^* and R_t^π represent the cumulative rewards under the hindsight optimal policy and our policy, respectively
- We can also show the expected total waiting times satisfy the same bounds
- $O(1/\epsilon)$ regret is the best possible [Kerimov-Ashlagi-Gurvich' 22a]
- If $\epsilon = 0$ (optimal dual is not unique), the regret at all time is $O(\sqrt{T})$ [Wei-Xu-Yu' 23], where the lower bound is $\Omega(\sqrt{T})$[Kerimov-Ashlagi-Gurvich' 22a].

Comparison to state of the art

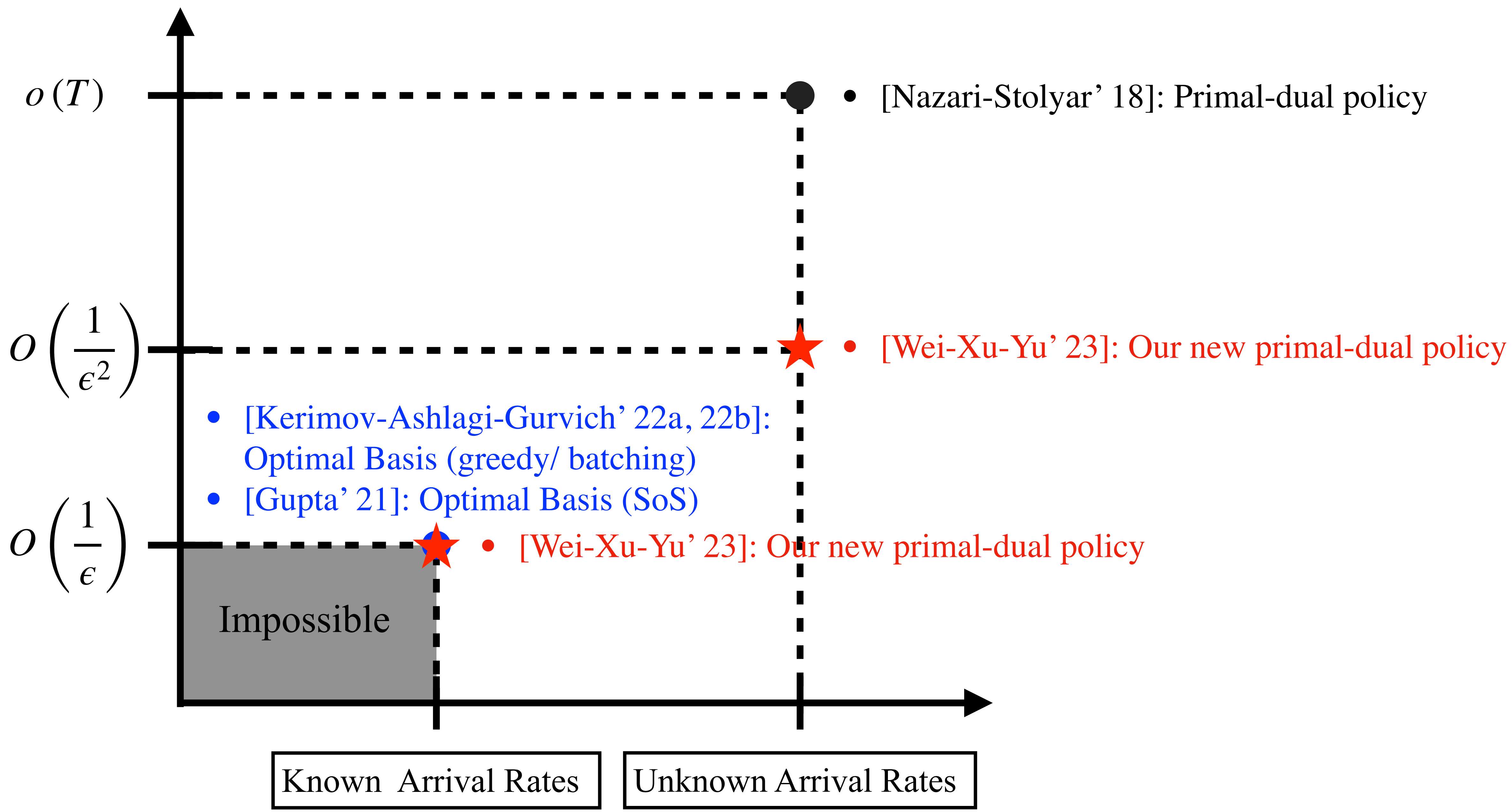
Comparison to state of the art



Comparison to state of the art

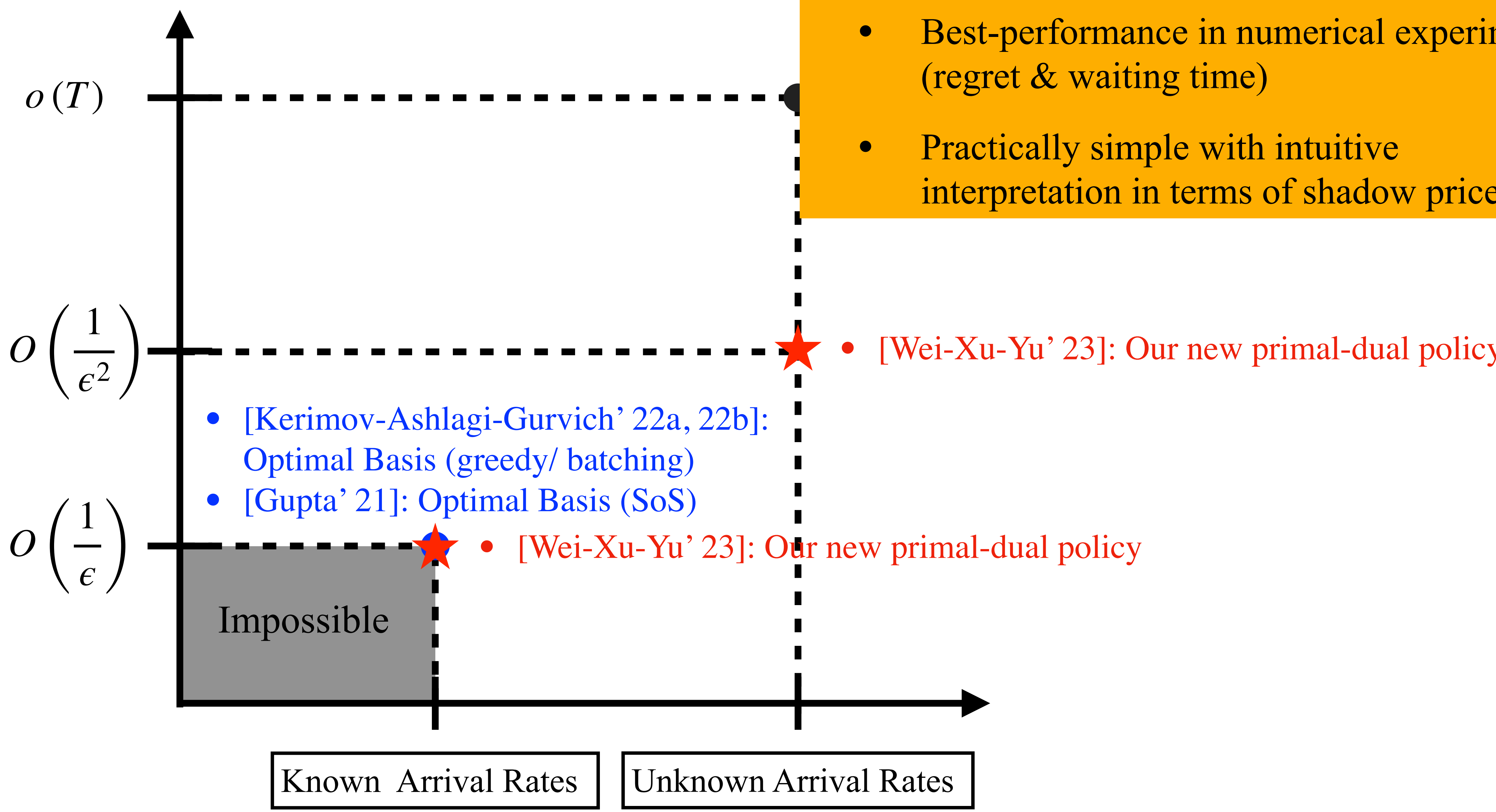


Comparison to state of the art



Comparison to state of the art

- Our new primal-dual policy:
  - **Fairness:** allows any match type to be used;
  - **Constant regret**: suitable for **unknown** λ
  - Best-performance in numerical experiments (regret & waiting time)
  - Practically simple with intuitive interpretation in terms of shadow prices



Proof ideas (Lyapunov drift method)

- Regret decomposition at time t:

$$\mathbb{E}[R_t^* - R_t^\pi] \lesssim \sum_{s=1}^{t} \eta_s + \mathbb{E}[\|\delta(t)\|] + \sum_{s=1}^{t} \mathbb{P}[\hat{p}_s \neq p^*]$$

Loss due to dynamic adjustment of shadow prices

Loss due to backlogged agents

Loss due to shadow price errors

Proof ideas (Lyapunov drift method)

- Regret decomposition at time t:

$$\mathbb{E}[R_t^* - R_t^\pi] \lesssim \underbrace{\sum_{s=1}^{t} \eta_s}_{\text{Loss due to dynamic adjustment of shadow prices}} + \underbrace{\mathbb{E}[\|\delta(t)\|]}_{\text{Loss due to backlogged agents}} + \underbrace{\sum_{s=1}^{t} \mathbb{P}[\hat{p}_s \neq p^*]}_{\text{Loss due to shadow price errors}}$$

- Choose $\eta_s = s^{-2}$ ensures that $\sum_{1 \le s \le t} \eta_s < \infty$

Proof ideas (Lyapunov drift method)

- Regret decomposition at time t:

$$\mathbb{E}[R_t^* - R_t^\pi] \lesssim \sum_{s=1}^{t} \eta_s + \mathbb{E}[\|\delta(t)\|] + \sum_{s=1}^{t} \mathbb{P}[\hat{p}_s \neq p^*]$$

Loss due to dynamic adjustment of shadow prices

Loss due to backlogged agents

Loss due to shadow price errors

- Choose $\eta_s = s^{-2}$ ensures that $\sum_{1 \le s \le t} \eta_s < \infty$

- Show $\mathbb{E}\left[\|\delta(t+1)\|_2 | \mathcal{F}_t\right] \le \|\delta(t)\|_2 - \frac{\epsilon}{2} \Longrightarrow \mathbb{E}[\|\delta(t)\|] = O\left(\epsilon^{-1}\right)$

Lyapunov drift analysis + local polyhedral property [Huang-Neely '09]

Coupling with bounded reflective random walk [Gupta' 21]

Proof ideas (Lyapunov drift method)

- Regret decomposition at time t:

$$\mathbb{E}[R_t^* - R_t^\pi] \lesssim \sum_{s=1}^{t} \eta_s + \mathbb{E}[\|\delta(t)\|] + \sum_{s=1}^{t} \mathbb{P}[\hat{p}_s \neq p^*]$$

Loss due to dynamic adjustment of shadow prices

Loss due to backlogged agents

Loss due to shadow price errors

- Choose $\eta_s = s^{-2}$ ensures that $\sum_{1 \le s \le t} \eta_s < \infty$

- Show $\mathbb{E}\left[\|\delta(t+1)\|_2 | \mathcal{F}_t\right] \le \|\delta(t)\|_2 - \frac{\epsilon}{2} \implies \mathbb{E}[\|\delta(t)\|] = O\left(\epsilon^{-1}\right)$

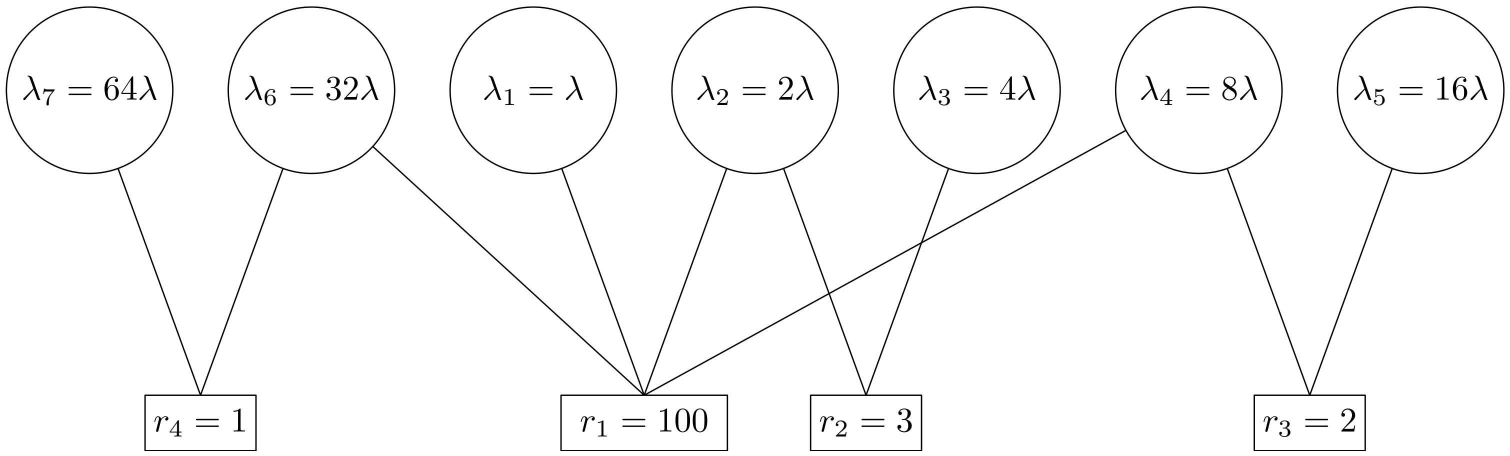
- $\sum_{s=1}^{t} \mathbb{P}[\hat{p}(s) \neq p^*] \le \sum_{s=1}^{t} \mathbb{P}[\|\hat{\lambda}(s) - \lambda\| \ge \epsilon] \le \sum_{s=1}^{t} \exp(-s\epsilon^2/8) \le O\left(\epsilon^{-2}\right)$

Non-degeneracy assumption

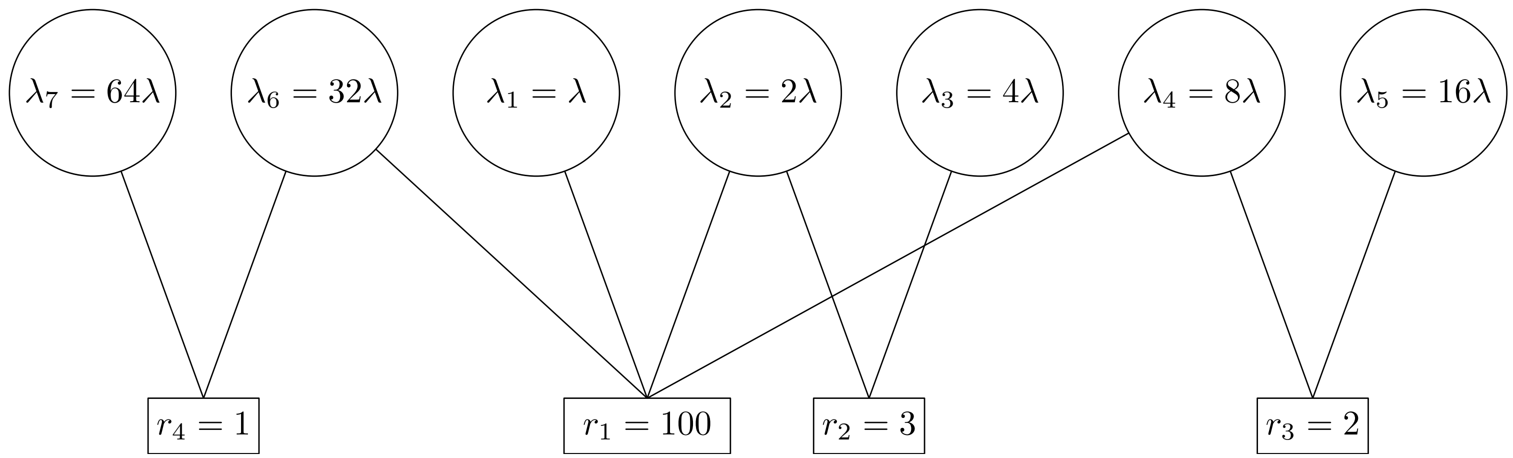
Hoeffding's inequality

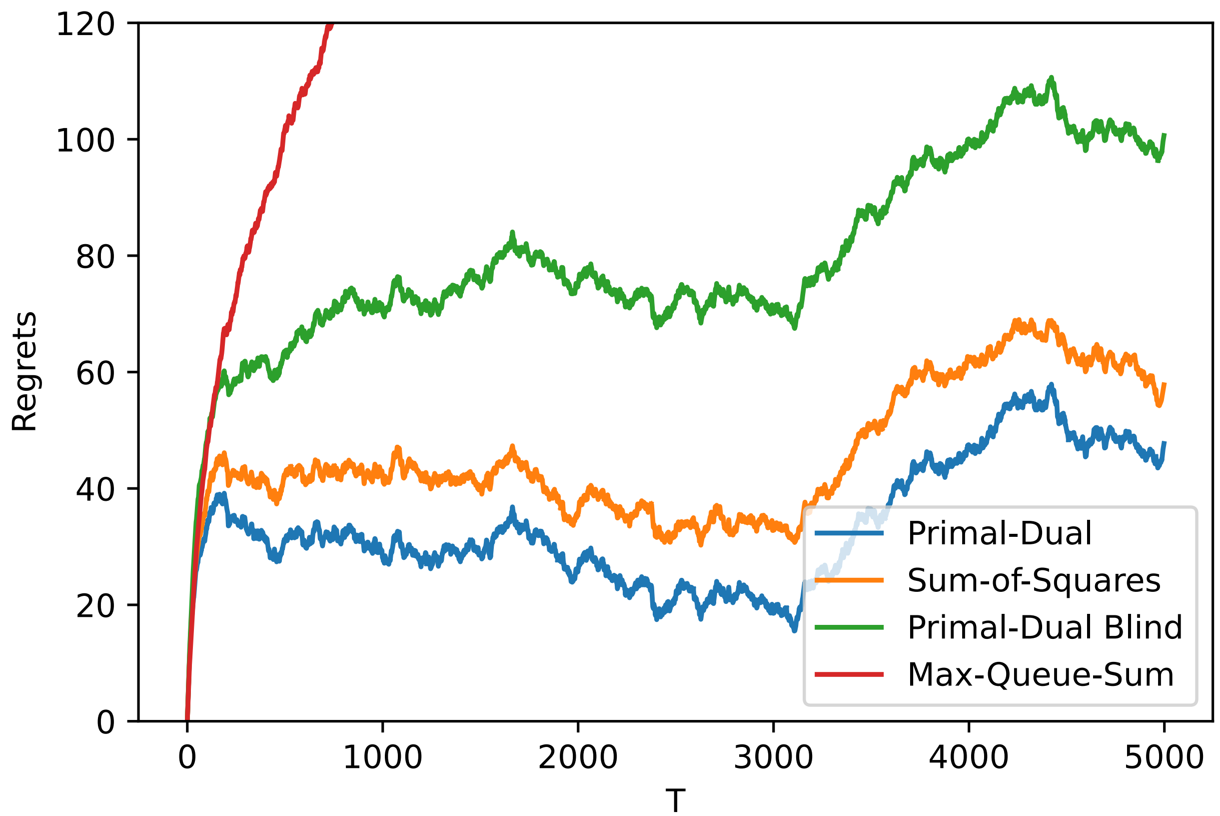
Numerical experiments

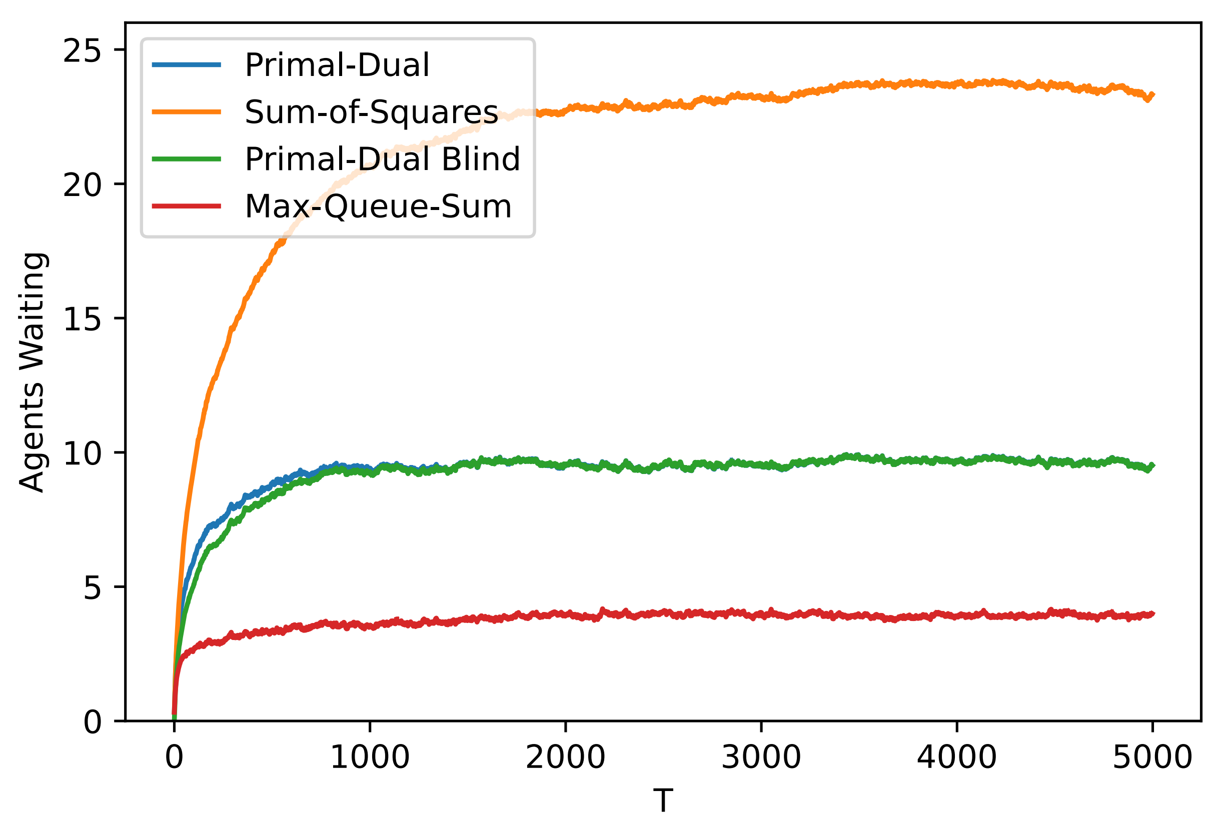
Better empirical rewards and waiting times



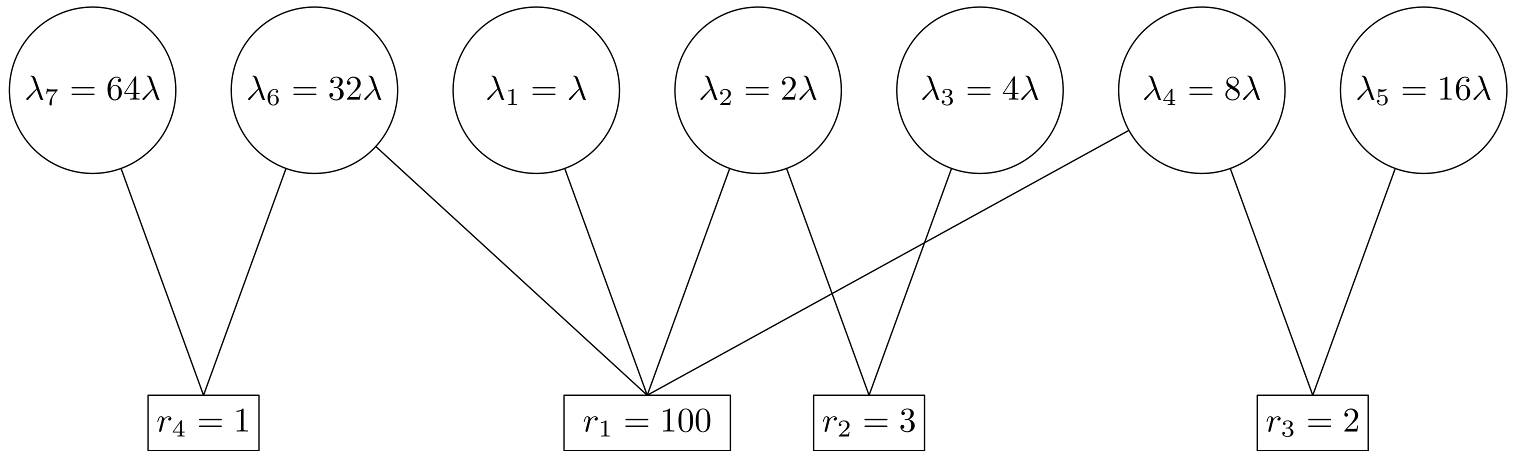
Better empirical rewards and waiting times

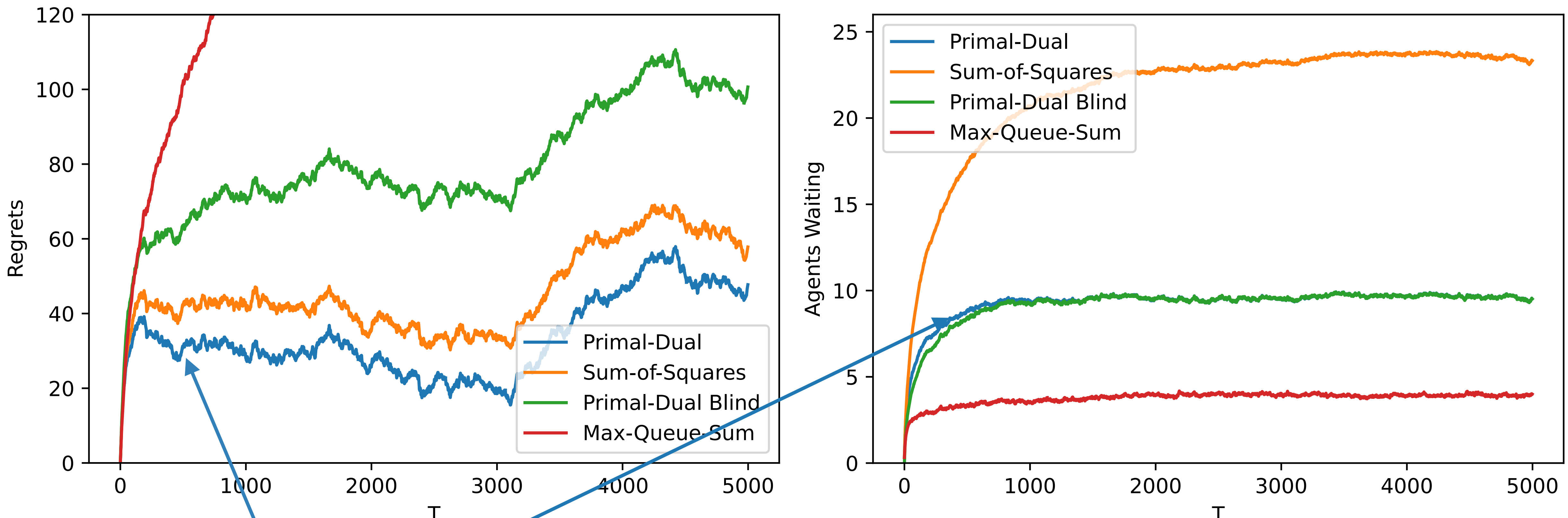






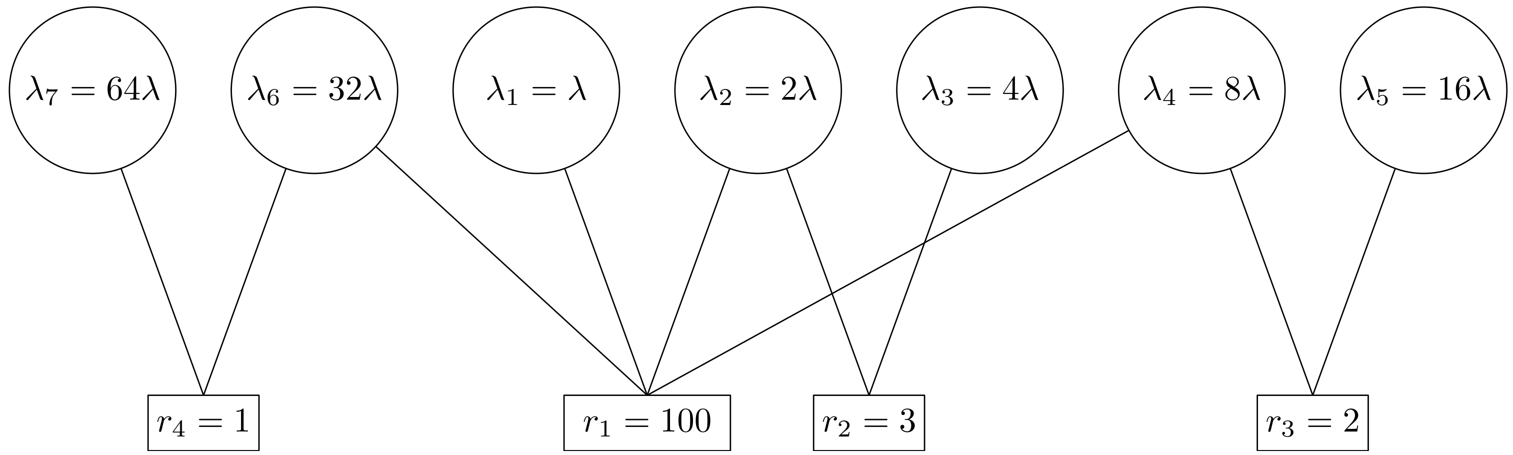
Better empirical rewards and waiting times

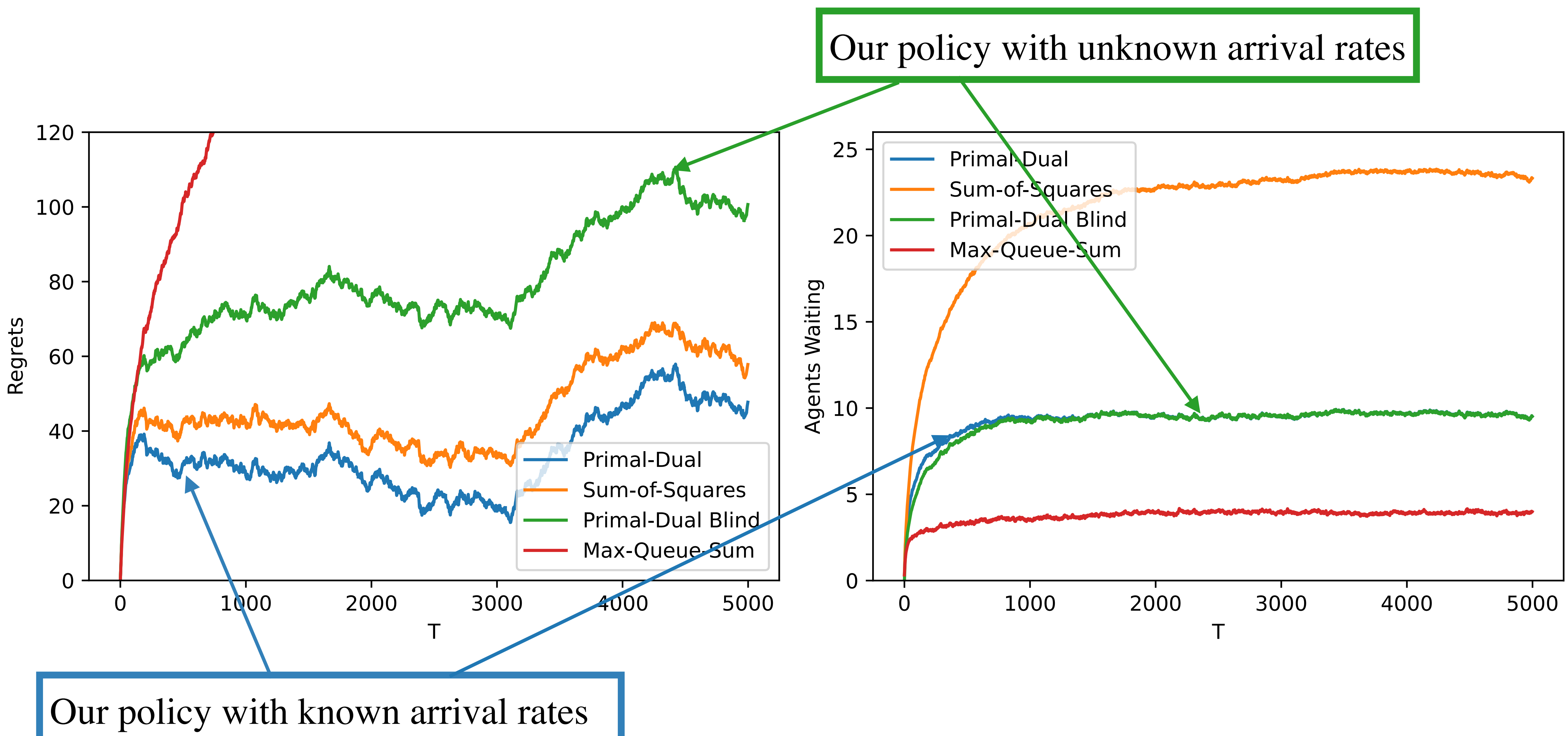




Our policy with known arrival rates

Better empirical rewards and waiting times





Summary

- **A new primal-dual policy for dynamic matching:**
  - First constant-regret policy with unknown λ
  - Does not confine matching decisions to the optimal basis
  - With known λ, our policy matches the regret bound in the literature
  - Best performance in numerical experiments and practically simple
  - Results can be extended to other dynamic resource allocation problems

Summary

- **A new primal-dual policy for dynamic matching:**
  - First constant-regret policy with unknown λ
  - Does not confine matching decisions to the optimal basis
  - With known λ, our policy matches the regret bound in the literature
  - Best performance in numerical experiments and practically simple
  - Results can be extended to other dynamic resource allocation problems
- **Some interesting problems:**
  - Matching models with infinite agent types/abandonment;
  - Matching models with possible agent declines;
  - Fairness in dynamic matching;
  - Primal-dual policies for other reward collection problems

Reference:

Y. Wei, J. Xu, & S. H. Yu, *Constant Regret Primal-Dual Policy for Multi-Way Dynamic Matching,* under revision, Management Science, SSRN.4357216

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;
- After one **agent** arrives, we need to immediately decides which **action** we use to serve the **agent**:

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;
- After one **agent** arrives, we need to immediately decides which **action** we use to serve the **agent**:
  - Each **action** could only serve one type of **agent,** consumes one or multiple types of offline **resources**, and generate r_a reward.

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;
- After one **agent** arrives, we need to immediately decides which **action** we use to serve the **agent**:
  - Each **action** could only serve one type of **agent,** consumes one or multiple types of offline **resources**, and generate r_a reward.
- Consider the fluid formulation:

$$
\begin{aligned}
\max_x \ & \sum_a r_a x_a \\
\text{s.t.} \ & Mx = \lambda \\
& Sx \leq B/T \\
& x_a \geq 0, \forall a
\end{aligned}
$$

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;
- After one **agent** arrives, we need to immediately decides which **action** we use to serve the **agent**:
  - Each **action** could only serve one type of **agent,** consumes one or multiple types of offline **resources**, and generate r_a reward.
- Consider the fluid formulation:

r_a is the reward of **action** a.

$$\begin{aligned} \max_x \ & \sum_a r_a x_a \\ \text{s.t.}\ & Mx = \lambda \\ & Sx \leq B/T \\ & x_a \geq 0, \forall a \end{aligned}$$

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;
- After one **agent** arrives, we need to immediately decides which **action** we use to serve the **agent**:
  - Each **action** could only serve one type of **agent,** consumes one or multiple types of offline **resources**, and generate r_a reward.
- Consider the fluid formulation:

r_a is the reward of **action** a.

$$
\begin{aligned}
\max_x \ & \sum_a r_a x_a \\
\text{s.t. } & Mx = \lambda \\
& Sx \leq B/T \\
& x_a \geq 0, \forall a
\end{aligned}
$$

where M denotes the **action** serving matrix s.t. $M_{i,a} = 1$ if **action** a serves **agent** i.

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;
- After one **agent** arrives, we need to immediately decides which **action** we use to serve the **agent**:
  - Each **action** could only serve one type of **agent,** consumes one or multiple types of offline **resources**, and generate r_a reward.
- Consider the fluid formulation:

$$
\begin{aligned}
\max_x \ & \sum_a r_a x_a \\
\text{s.t.} \ & Mx = \lambda \\
& Sx \le B/T \\
& x_a \ge 0, \forall a
\end{aligned}
$$

r_a is the reward of **action** a.

where M denotes the **action** serving matrix s.t. $M_{i,a} = 1$ if **action** a serves **agent** i.

λ is the **agents**' arrival rate vector.

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;
- After one **agent** arrives, we need to immediately decides which **action** we use to serve the **agent**:
  - Each **action** could only serve one type of **agent,** consumes one or multiple types of offline **resources**, and generate r_a reward.
- Consider the fluid formulation:

r_a is the reward of **action** a.

where M denotes the **action** serving matrix s.t. $M_{i,a} = 1$ if **action** a serves **agent** i.

$$\max_x \sum_a r_a x_a$$
$$\text{s.t. } Mx = \lambda$$
$$Sx \leq B/T$$
$$x_a \geq 0, \forall a$$

λ is the **agents**' arrival rate vector.

where S denotes the offline **resource** consumption matrix s.t. $S_{j,a} = 1$ if **action** a consumes **resource** j.

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;
- After one **agent** arrives, we need to immediately decides which **action** we use to serve the **agent**:
  - Each **action** could only serve one type of **agent,** consumes one or multiple types of offline **resources**, and generate r_a reward.
- Consider the fluid formulation:

$$
\begin{aligned}
\max_x \ & \sum_a r_a x_a \\
\text{s.t. } & Mx = \lambda \\
& Sx \le B/T \\
& x_a \ge 0, \forall a
\end{aligned}
$$

r_a is the reward of **action** a.

where M denotes the **action** serving matrix s.t. $M_{i,a} = 1$ if **action** a serves **agent** i.

λ is the **agents**' arrival rate vector.

B/T denotes the offline **resource** consumption per period on average.

where S denotes the offline **resource** consumption matrix s.t. $S_{j,a} = 1$ if **action** a consumes **resource** j.

Resource allocation model [Wei-Xu-Yu 24']

- Offline **resources** (available at the beginning) with budget B;
- Online **agents** arrive dynamically with arrival rate λ;
- After one **agent** arrives, we need to immediately decides which **action** we use to serve the **agent**:
  - Each **action** could only serve one type of **agent,** consumes one or multiple types of offline **resources**, and generate r_a reward.
- Consider the fluid formulation:

r_a is the reward of **action** a.

where M denotes the **action** serving matrix s.t. $M_{i,a} = 1$ if **action** a serves **agent** i.

$$\begin{aligned} \max_x \ & \sum_a r_a x_a \\ \text{s.t.} \ & Mx = \lambda \\ & \boxed{Sx \le B/T} \\ & x_a \ge 0, \forall a \end{aligned}$$

λ is the **agents**' arrival rate vector.

where S denotes the offline **resource** consumption matrix s.t. $S_{j,a} = 1$ if **action** a consumes **resource** j.

B/T denotes the offline **resource** consumption per period on average.

- Key: primal-dual algorithm by updating $p(t)$, and select the action has maximum reduced reward, where p^* is the associated dual variable of $Sx \le B/T$.

Bounding the inventory

Bounding the inventory

- **Negative drift:**

B is the maximum size of a match in the system.

$$\mathbb{E}\left[\|\delta(t+1)\|_2^2 \mid \mathscr{F}_t\right] \leq \|\delta(t)\|_2^2 + B - 2\eta_t^{-1}\left(D(p(t)) - D(p^*)\right)$$

Dual objective function

Bounding the inventory

- **Negative drift:**

B is the maximum size of a match in the system.

$$\mathbb{E}\left[\|\delta(t+1)\|_2^2 \mid \mathscr{F}_t\right] \le \|\delta(t)\|_2^2 + B - 2\eta_t^{-1}\left(D(p(t)) - D(p^*)\right)$$
$$\le \|\delta(t)\|_2^2 + B - 2\eta_t^{-1}\epsilon\|p(t) - p^*\|_2$$

By the GPG assumption $\Longleftrightarrow$ the local polyhedral property.

Bounding the inventory

- **Negative drift:**

B is the maximum size of a match in the system.

$$
\begin{aligned}
\mathbb{E}\left[\|\delta(t+1)\|_2^2 \mid \mathcal{F}_t\right] &\le \|\delta(t)\|_2^2 + B - 2\eta_t^{-1}\left(D(p(t)) - D(p^*)\right) \\
&\le \|\delta(t)\|_2^2 + B - 2\eta_t^{-1}\epsilon\|p(t) - p^*\|_2 \\
&= \|\delta(t)\|_2^2 + B - 2\epsilon\|\delta(t)\|_2
\end{aligned}
$$

By the GPG assumption $\Longleftrightarrow$ the local polyhedral property.

Bounding the inventory

- **Negative drift:**

B is the maximum size of a match in the system.

$$
\begin{aligned}
\mathbb{E}\left[\|\delta(t+1)\|_2^2 \mid \mathscr{F}_t\right] &\leq \|\delta(t)\|_2^2 + B - 2\eta_t^{-1}\left(D(p(t)) - D(p^*)\right) \\
&\leq \|\delta(t)\|_2^2 + B - 2\eta_t^{-1}\epsilon\|p(t) - p^*\|_2 \\
&= \|\delta(t)\|_2^2 + B - 2\epsilon\|\delta(t)\|_2
\end{aligned}
$$

By the GPG assumption $\Longleftrightarrow$ the local polyhedral property.

If $\|\delta(t)\| > \dfrac{3B}{\epsilon}$, we have

$$
\mathbb{E}\left[\|\delta(t+1)\|_2 | \mathscr{F}_t\right] \leq \|\delta(t)\|_2 - \frac{\epsilon}{2}.
$$

Bounding the inventory

- **Negative drift:**

B is the maximum size of a match in the system.

$$\mathbb{E}\left[\|\delta(t+1)\|_2^2 \mid \mathscr{F}_t\right] \le \|\delta(t)\|_2^2 + B - 2\eta_t^{-1}\left(D(p(t)) - D(p^*)\right)$$
$$\le \|\delta(t)\|_2^2 + B - 2\eta_t^{-1}\epsilon\|p(t) - p^*\|_2$$
$$= \|\delta(t)\|_2^2 + B - 2\epsilon\|\delta(t)\|_2$$

By the GPG assumption $\Longleftrightarrow$ the local polyhedral property.

If $\|\delta(t)\| > \dfrac{3B}{\epsilon}$, we have

$$\mathbb{E}\left[\|\delta(t+1)\|_2 \mid \mathscr{F}_t\right] \le \|\delta(t)\|_2 - \frac{\epsilon}{2}.$$

- By a coupling argument with bounded reflective random walk following [Gupta' 21],

$$\mathbb{E}\left[\|\delta(t)\|_2\right] = O\left(\epsilon^{-1}\right) \quad \forall t.$$