Online Matchings with Delays and Stochastic Arrival Times

Michał Pawłowski

University of Warsaw, IDEAS NCBR

December 2, 2022

Plan

online matchings with delays

Plan

adversarial model

stochastic model

online matchings with delays

Plan

adversarial
model

stochastic
model

online matchings with delays

1

motivation

Plan

adversarial model

stochastic model

online matchings with delays

1 motivation

2 problem statement

Plan

adversarial model

stochastic model

online matchings with delays

1 motivation

2 problem statement

3 previous results

Plan

adversarial model

stochastic model

online matchings with delays

1. motivation
2. problem statement
3. previous results
4. stochastic analysis

Plan

adversarial model

stochastic model

online matchings with delays

1 motivation

2 problem statement

3 previous results

4 stochastic analysis

5 questions

Motivation

Online chess gaming platforms

- players log into an **online** platform

Motivation

Online chess gaming platforms

- players log into an **online** platform

Alice

$t = 10$

Motivation

Online chess gaming platforms

- players log into an **online** platform

Alice

$t = 10$

Bob

$t = 20$

Motivation

Online chess gaming platforms

- players log into an **online** platform

Alice

$t = 10$

Bob

$t = 20$

Carol

$t = 25$

Motivation

Online chess gaming platforms

- players log into an **online** platform

Alice

$t = 10$

Bob

$t = 20$

Carol

$t = 25$

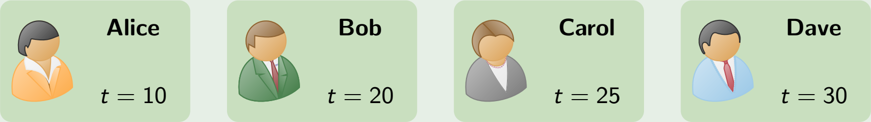
Dave

$t = 30$

Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other



Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other
- matching should **maximize** the overall satisfaction from the game

Alice

$t = 10$

Bob

$t = 20$

Carol

$t = 25$

Dave

$t = 30$

Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other
- matching should **maximize** the overall satisfaction from the game
- player prefers to be paired with someone of **similar gaming skills**

Alice

$t = 10$

Bob

$t = 20$

Carol

$t = 25$

Dave

$t = 30$

Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other
- matching should **maximize** the overall satisfaction from the game
- player prefers to be paired with someone of **similar gaming skills**

Alice

$l = 100$

$t = 10$

Bob

$l = 50$

$t = 20$

Carol

$l = 60$

$t = 25$

Dave

$l = 110$

$t = 30$

Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other
- matching should **maximize** the overall satisfaction from the game
- player prefers to be paired with someone of **similar gaming skills**
- first measure: **connection cost** = experience gap

Alice

$l = 100$

$t = 10$

Bob

$l = 50$

$t = 20$

Carol

$l = 60$

$t = 25$

Dave

$l = 110$

$t = 30$

Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other
- matching should **maximize** the overall satisfaction from the game
- player prefers to be paired with someone of **similar gaming skills**
- first measure: **connection cost** = experience gap
- player prefers to be **matched** as **fast** as possible

Alice
$l = 100$
$t = 10$

Bob
$l = 50$
$t = 20$

Carol
$l = 60$
$t = 25$

Dave
$l = 110$
$t = 30$

Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other
- matching should **maximize** the overall satisfaction from the game
- player prefers to be paired with someone of **similar gaming skills**
- first measure: **connection cost** = experience gap
- player prefers to be **matched** as **fast** as possible
- second measure: **delay cost** = waiting time

Alice
$l = 100$
$t = 10$

Bob
$l = 50$
$t = 20$

Carol
$l = 60$
$t = 25$

Dave
$l = 110$
$t = 30$

Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other
- matching should **maximize** the overall satisfaction from the game
- player prefers to be paired with someone of **similar gaming skills**
- first measure: **connection cost** = experience gap
- player prefers to be **matched** as **fast** as possible
- second measure: **delay cost** = waiting time

Alice
$l = 100$
$t = 10$

Bob
$l = 50$
$t = 20$

Carol
$l = 60$
$t = 25$

Dave
$l = 110$
$t = 30$

optimal:

Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other
- matching should **maximize** the overall satisfaction from the game
- player prefers to be paired with someone of **similar gaming skills**
- first measure: **connection cost** = experience gap
- player prefers to be **matched** as **fast** as possible
- second measure: **delay cost** = waiting time

Alice
$l = 100$
$t = 10$

Bob
$l = 50$
$t = 20$

Carol
$l = 60$
$t = 25$

Dave
$l = 110$
$t = 30$

optimal:

Motivation

Online chess gaming platforms

- players log into an **online** platform to be **paired** with each other
- matching should **maximize** the overall satisfaction from the game
- player prefers to be paired with someone of **similar gaming skills**
- first measure: **connection cost** = experience gap
- player prefers to be **matched** as **fast** as possible
- second measure: **delay cost** = waiting time

Alice
$l = 100$
$t = 10$

Bob
$l = 50$
$t = 20$

Carol
$l = 60$
$t = 25$

Dave
$l = 110$
$t = 30$

optimal: $20 + 10 + 5 + 10 = 45$

Problem statement

Min-cost Perfect Matching with Delays

Problem statement

Min-cost Perfect Matching with Delays

- **metric space** $\mathcal{M} = (\mathcal{X}, d)$ equipped with a distance function d

Min-cost Perfect Matching with Delays

- **metric space** $\mathcal{M} = (\mathcal{X}, d)$ equipped with a distance function d
- m requests arriving at **arbitrary times** at points of $\mathcal{X}$ assume $2|m$

Min-cost Perfect Matching with Delays

- **metric space** $\mathcal{M} = (\mathcal{X}, d)$ equipped with a distance function d
- m requests arriving at **arbitrary times** at points of $\mathcal{X}$ assume $2|m$
- each request r is characterized by its **location** $\ell(r) \in \mathcal{X}$ and **arrival time** $t(r) \in \mathbb{R}^+$

Min-cost Perfect Matching with Delays

- **metric space** $\mathcal{M} = (\mathcal{X}, d)$ equipped with a distance function d
- m requests arriving at **arbitrary times** at points of $\mathcal{X}$ assume $2|m$
- each request r is characterized by its **location** $\ell(r) \in \mathcal{X}$ and **arrival time** $t(r) \in \mathbb{R}^+$
- when two requests r and r' are **matched** into a pair at time $t \geq \max\{t(r), t(r')\}$, a connection cost of $d(\ell(r), \ell(r'))$ plus a delay cost $(t - t(r)) + (t - t(r'))$ is incurred

Min-cost Perfect Matching with Delays

- **metric space** $\mathcal{M} = (\mathcal{X}, d)$ equipped with a distance function d
- m requests arriving at **arbitrary times** at points of $\mathcal{X}$ assume $2|m$
- each request r is characterized by its **location** $\ell(r) \in \mathcal{X}$ and **arrival time** $t(r) \in \mathbb{R}^+$
- when two requests r and r' are **matched** into a pair at time $t \geq \max\{t(r), t(r')\}$, a connection cost of $d(\ell(r), \ell(r'))$ plus a delay cost $(t - t(r)) + (t - t(r'))$ is incurred
- the target is to **minimize the total cost** produced by the online algorithm for matching all the requests into pairs

Adversarial model

- **adversary determines** the arrival times and locations of requests

Adversarial model

- **adversary determines** the arrival times and locations of requests

known metric

Adversarial model

- **adversary determines** the arrival times and locations of requests

known metric

- the **current best** competitiveness is $O(\log n)$ (where n denotes the number of points in the metric) [Azar at al., SODA'17]

Adversarial model

- **adversary determines** the arrival times and locations of requests

known metric

- the **current best** competitiveness is $O(\log n)$ (where n denotes the number of points in the metric) [Azar at al., SODA'17]
- **no** online algorithm can achieve competitive ratio **better than** $\Omega(\log n / \log \log n)$ [Ashlagi et al., APPROX/RANDOM'17]

Adversarial model

- **adversary determines** the arrival times and locations of requests

known metric

- the **current best** competitiveness is $O(\log n)$ (where n denotes the number of points in the metric) [Azar at al., SODA'17]
- **no** online algorithm can achieve competitive ratio **better than** $\Omega(\log n / \log\log n)$ [Ashlagi et al., APPROX/RANDOM'17]

unknown metric

Adversarial model

- **adversary determines** the arrival times and locations of requests

known metric

- the **current best** competitiveness is $O(\log n)$ (where n denotes the number of points in the metric) [Azar at al., SODA'17]
- **no** online algorithm can achieve competitive ratio **better than** $\Omega(\log n / \log \log n)$ [Ashlagi et al., APPROX/RANDOM'17]

unknown metric

- the **current best** competitiveness is $O(m^{\log \frac{3}{2} + \varepsilon})$ (with $\varepsilon > 0$ arbitrarily small) [Azar et al., TOCS'20]

Stochastic model

- for many applications it is **too pessimistic** to assume that **no stochastic information** on the input is available

Stochastic model

- for many applications it is **too pessimistic** to assume that **no stochastic information** on the input is available
- online gaming platform has all the **historical data** and can **estimate the arrival frequency** of the players with each particular skill level

Stochastic model

- for many applications it is **too pessimistic** to assume that **no stochastic information** on the input is available
- online gaming platform has all the **historical data** and can **estimate the arrival frequency** of the players with each particular skill level
- we can assume that **requests follow a stochastic distribution**

Stochastic model

- for many applications it is **too pessimistic** to assume that **no stochastic information** on the input is available
- online gaming platform has all the **historical data** and can **estimate the arrival frequency** of the players with each particular skill level
- we can assume that **requests follow a stochastic distribution**

Poisson arrival process

Stochastic model

- for many applications it is **too pessimistic** to assume that **no stochastic information** on the input is available
- online gaming platform has all the **historical data** and can **estimate the arrival frequency** of the players with each particular skill level
- we can assume that **requests follow a stochastic distribution**

Poisson arrival process

- **waiting time** between any two consecutive requests arriving at any metrical point x follows an **exponential distribution** $\mathrm{Exp}(\lambda_x)$

Stochastic model

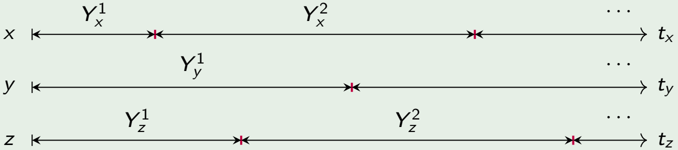
- for many applications it is **too pessimistic** to assume that **no stochastic information** on the input is available
- online gaming platform has all the **historical data** and can **estimate the arrival frequency** of the players with each particular skill level
- we can assume that **requests follow a stochastic distribution**

Poisson arrival process

- **waiting time** between any two consecutive requests arriving at any metrical point x follows an **exponential distribution** $\mathrm{Exp}(\lambda_x)$
  (we will refer to this model as *distributed Poisson arrival model*)

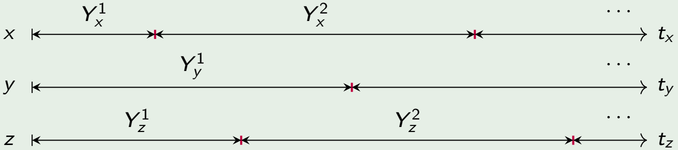
Ratio of expectations

Distributed Poisson arrival model



Ratio of expectations

Distributed Poisson arrival model



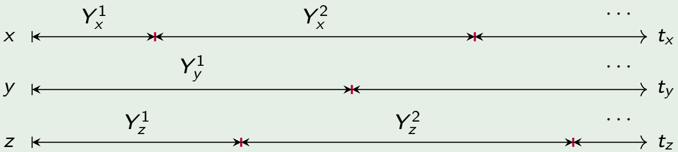
Ratio of expectations

Algorithm ALG for MPMD has a **ratio-of-expectations** $C \geq 1$, if

$$\varlimsup_{m \to \infty} \frac{\mathbb{E}_\sigma^m[\mathrm{ALG}(\sigma)]}{\mathbb{E}_\sigma^m[\mathrm{OPT}(\sigma)]} \leq C$$

Ratio of expectations

Distributed Poisson arrival model



Ratio of expectations

Algorithm ALG for MPMD has a **ratio-of-expectations** $C \geq 1$, if

$$\overline{\lim_{m \to \infty}} \frac{\mathbb{E}_\sigma^m[\mathrm{ALG}(\sigma)]}{\mathbb{E}_\sigma^m[\mathrm{OPT}(\sigma)]} \leq C,$$

where $\mathbb{E}_\sigma^m[\mathrm{ALG}(\sigma)]$ (resp. $\mathbb{E}_\sigma^m[\mathrm{OPT}(\sigma)]$) denotes the **expected cost** generated by ALG (resp. an **optimal offline solution**) on the **random request sequence** σ, $|\sigma| = m$, generated by the Poisson arrival process.

Exponential variables recap

Memoryless property

If X is an exponential variable with parameter λ, then for all $s, t \geq 0$, we have

$$\mathbb{P}(X > s + t \mid X > s) = \mathbb{P}(X > t) = e^{-\lambda t}.$$

Exponential variables recap

Memoryless property

If X is an exponential variable with parameter λ, then for all $s, t \geq 0$, we have

$$\mathbb{P}\left(X > s + t \mid X > s\right) = \mathbb{P}(X > t) = e^{-\lambda t}.$$

Minimum of exponential variables

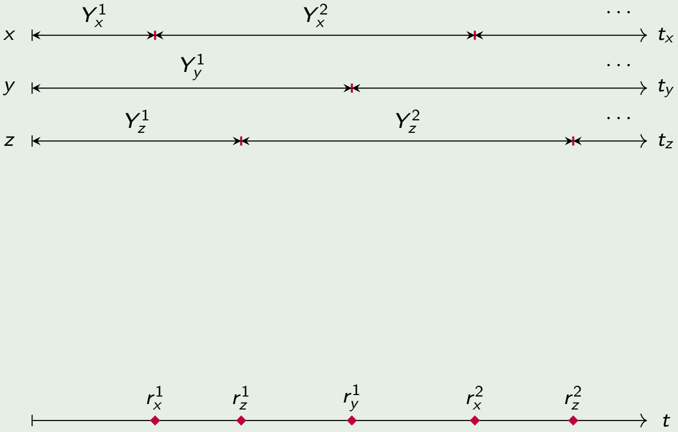
Given n independent exponential variables $X_i \sim \mathrm{Exp}(\lambda_i)$ for $1 \leq i \leq n$, let $Z := \min\{X_1, X_2, \dots, X_n\}$ and let $\lambda := \sum_{i=1}^{n} \lambda_i$. It holds that

$$Z \sim \mathrm{Exp}(\lambda), \qquad \mathbb{P}(Z = X_i) = \lambda_i/\lambda, \qquad Z \perp \{Z = X_i\},$$

where $\perp$ denotes independence.

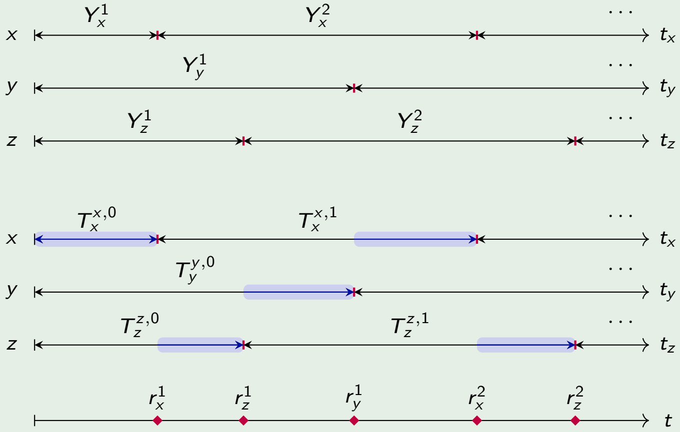
Alternative arrival models — part 1

Distributed Poisson arrival model

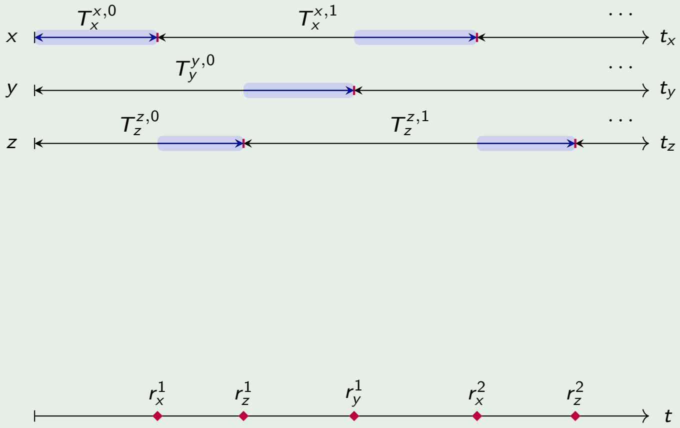


Alternative arrival models — part 1

Distributed Poisson arrival model = timers without resets

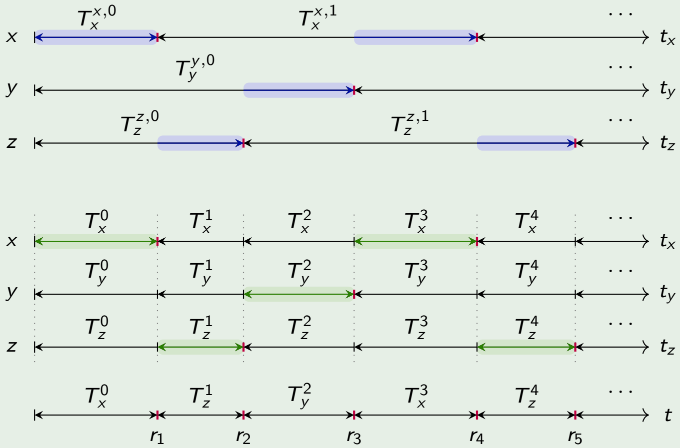


Timers without resets



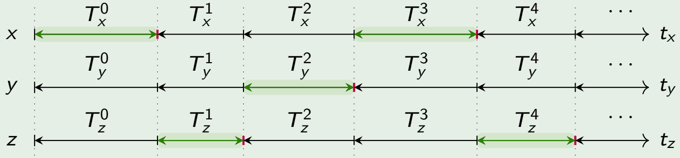
Alternative arrival models — part 2

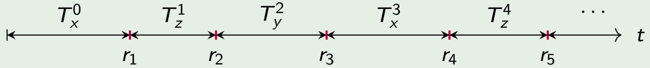
Timers without resets $=$ timers with resets



Alternative arrival models — part 3

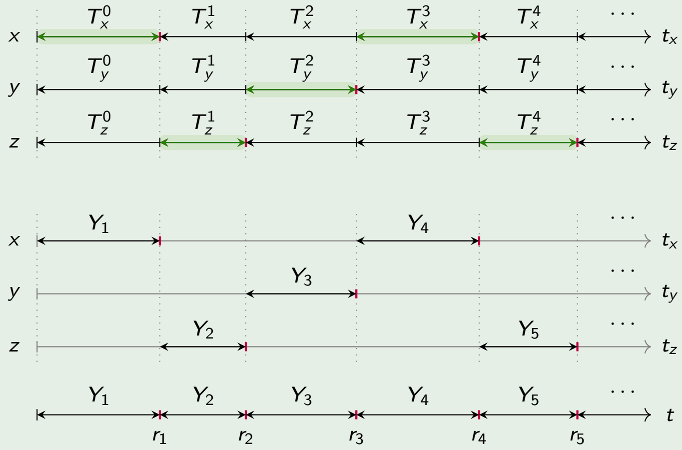
Timers with resets





Alternative arrival models — part 3

Timers with resets $=$ centralized Poisson arrival model



Poisson arrival process

distributed version

- waiting time between any two consecutive requests arriving at any **metrical point** x follows an exponential distribution $\mathrm{Exp}(\lambda_x)$

centralized version

- waiting time between any two consecutive requests in the given **metric space** follows an exponential distribution with parameter $\lambda(\mathcal{X}) := \sum_{x \in \mathcal{X}} \lambda_x$
- each time a request arrives, the probability of it appearing at point x equals $\lambda_x / \lambda(\mathcal{X})$

Lower bounding the optimal solution

- for a given sequence σ of requests denote the **minimum total cost** of r in σ as

$$c(\sigma, r) := \min_{r' \in \sigma, r' \neq r} \left\{ d(\ell(r), \ell(r')) + |t(r) - t(r')| \right\}$$

Lower bounding the optimal solution

- for a given sequence σ of requests denote the **minimum total cost** of r in σ as

$$c(\sigma, r) := \min_{r' \in \sigma, r' \neq r} \left\{ d(\ell(r), \ell(r')) + |t(r) - t(r')| \right\}$$

- given a pair of requests (r_1, r_2) paired by the OPT we have that $c(\sigma, r_i) \leq d(\ell(r_1), \ell(r_2)) + |t(r_1) - t(r_2)|$ for $i \in \{1, 2\}$

Lower bounding the optimal solution

- for a given sequence σ of requests denote the **minimum total cost** of r in σ as

$$c(\sigma, r) := \min_{r' \in \sigma, r' \neq r} \left\{ d(\ell(r), \ell(r')) + |t(r) - t(r')| \right\}$$

- given a pair of requests (r_1, r_2) paired by the OPT we have that $c(\sigma, r_i) \leq d(\ell(r_1), \ell(r_2)) + |t(r_1) - t(r_2)|$ for $i \in \{1, 2\}$

- for **any input sequence** σ it holds that

$$\mathrm{OPT}(\sigma) \geq \frac{1}{2} \sum_{r \in \sigma} c(\sigma, r)$$

Minimum total cost of a request

- we want to **lower bound**

$$\mathbb{E}_{\sigma}^{m}\left[\frac{1}{2}\sum_{i=1}^{m} c(\sigma, r_i)\right]$$

Minimum total cost of a request

- we want to **lower bound**

$$\mathbb{E}_{\sigma}^{m}\left[\frac{1}{2}\sum_{i=1}^{m} c(\sigma, r_i)\right]$$

- by the **linearity of expectation**, it suffices to estimate

$$\mathbb{E}_{\sigma}^{m}\left[c(\sigma, r_i)\right]$$

Minimum total cost of a request

- we want to **lower bound**

$$\mathbb{E}_{\sigma}^{m}\left[\frac{1}{2}\sum_{i=1}^{m} c(\sigma, r_i)\right]$$

- by the **linearity of expectation**, it suffices to estimate

$$\mathbb{E}_{\sigma}^{m}\left[c(\sigma, r_i)\right]$$

- the definition of the **centralized** Poisson arrival **model** allows us to focus on bounding

$$\mathbb{E}_{\sigma}^{m}[c(\sigma, r_i) \mid \ell(r_i) = x]$$

Minimum total cost of a request

- we want to **lower bound**

$$\mathbb{E}_\sigma^m\left[\frac{1}{2}\sum_{i=1}^{m} c(\sigma, r_i)\right]$$

- by the **linearity of expectation**, it suffices to estimate

$$\mathbb{E}_\sigma^m\left[c(\sigma, r_i)\right] = \sum_{x\in\mathcal{X}} \mathbb{P}_\sigma(\ell(r_i) = x) \cdot \mathbb{E}_\sigma^m[c(\sigma, r_i) \mid \ell(r_i) = x]$$

- the definition of the **centralized** Poisson arrival **model** allows us to focus on bounding

$$\mathbb{E}_\sigma^m[c(\sigma, r_i) \mid \ell(r_i) = x]$$

Radius definition — part 1

Finding a trade-off between distance and delay cost

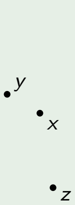
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



$d(x, y) = 1$ $\quad d(x, z) = 2$ $\quad d(x, v) = 4$

Radius definition — part 1

Finding a trade-off between distance and delay cost

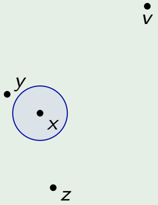
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



$d(x, y) = 1$ $\quad d(x, z) = 2$ $\quad d(x, v) = 4$

Radius definition — part 1

Finding a trade-off between distance and delay cost

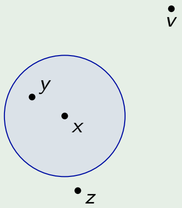
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



$d(x, y) = 1$ $d(x, z) = 2$ $d(x, v) = 4$

Radius definition — part 1

Finding a trade-off between distance and delay cost

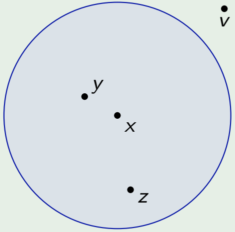
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



$d(x, y) = 1$ $d(x, z) = 2$ $d(x, v) = 4$

Radius definition — part 1

Finding a trade-off between distance and delay cost

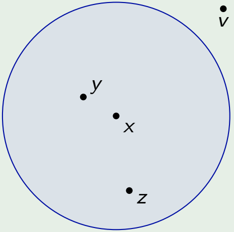
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



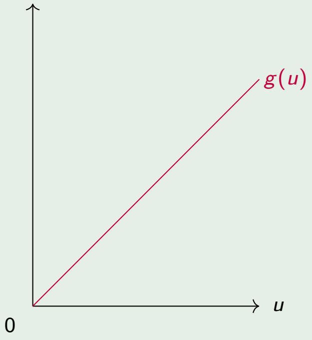
$d(x, y) = 1$ $d(x, z) = 2$ $d(x, v) = 4$

$f(u) = ?$ $g(u) = u$

Radius definition — part 1

Finding a trade-off between distance and delay cost

ball without border points



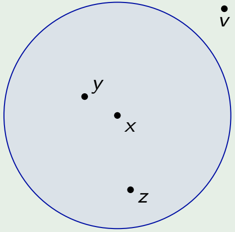
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$

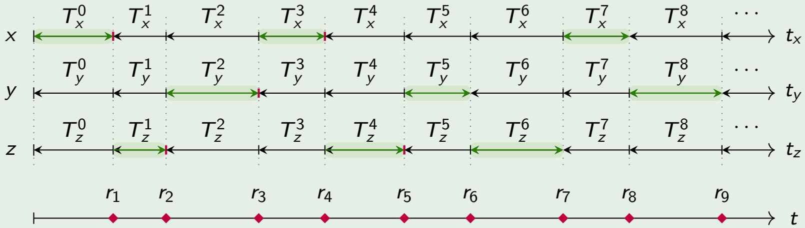


$d(x, y) = 1$ $d(x, z) = 2$ $d(x, v) = 4$

$f(u) = ?$ $g(u) = u$

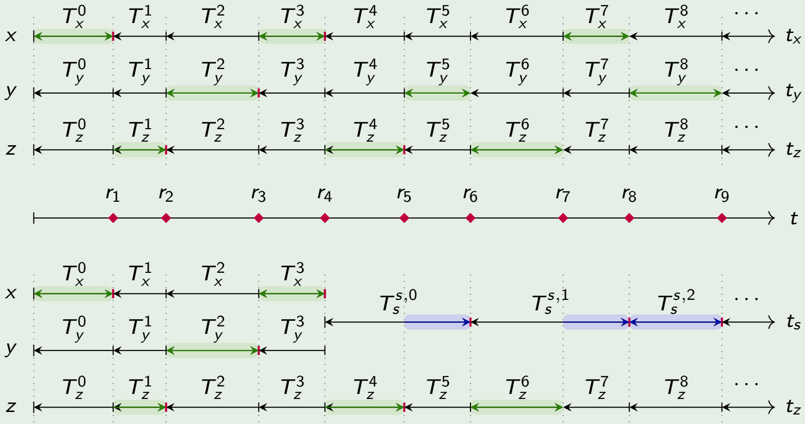
Radius definition — part 2

Arrival distribution for a subset



Radius definition — part 2

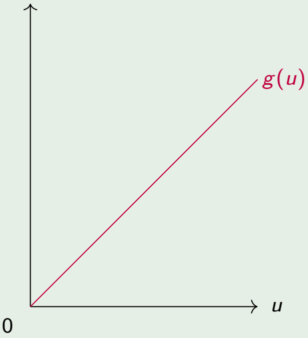
Arrival distribution for a subset



Radius definition — part 3

Finding a trade-off between distance and delay cost

ball without border points



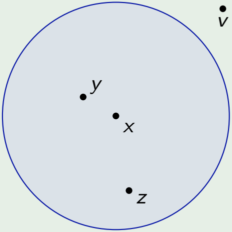
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



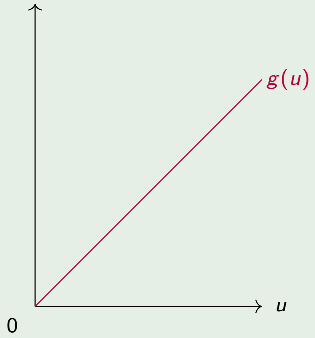
$d(x, y) = 1$ $d(x, z) = 2$ $d(x, v) = 4$

$f(u) = ?$ $g(u) = u$

Radius definition — part 3

Finding a trade-off between distance and delay cost

ball without border points



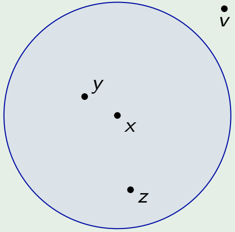
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



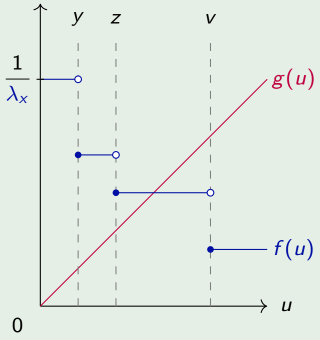
$d(x, y) = 1$ $\quad d(x, z) = 2$ $\quad d(x, v) = 4$

$$f(u) = \frac{1}{\lambda\left(\bar{B}(x, u)\right)} \qquad g(u) = u$$

Radius definition — part 3

Finding a trade-off between distance and delay cost

ball without border points



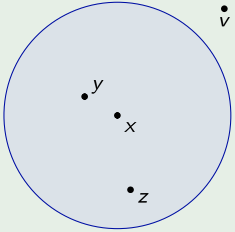
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



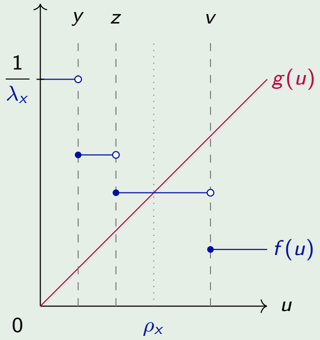
$d(x, y) = 1$ $\quad d(x, z) = 2$ $\quad d(x, v) = 4$

$$f(u) = \frac{1}{\lambda\left(\bar{B}(x, u)\right)} \qquad g(u) = u$$

Radius definition — part 3

Finding a trade-off between distance and delay cost

ball without border points



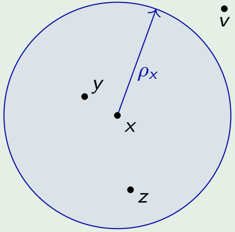
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



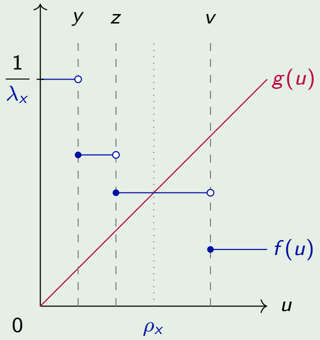
$d(x, y) = 1$ $\quad d(x, z) = 2$ $\quad d(x, v) = 4$

$$f(u) = \frac{1}{\lambda\left(\bar{B}(x, u)\right)} \qquad g(u) = u$$

Radius definition — part 3

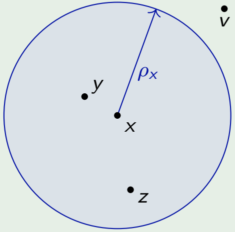
Finding a trade-off between distance and delay cost

ball without border points



metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$ $\lambda_y = \frac{1}{12}$ $\lambda_z = \frac{1}{12}$ $\lambda_v = \frac{1}{3}$



$\lambda_x = \frac{1}{6}$ $\lambda_y = \frac{1}{5}$ $\lambda_z = \frac{1}{3}$ $\lambda_v = \frac{1}{2}$

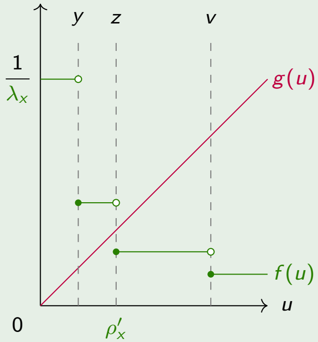
$d(x, y) = 1$ $d(x, z) = 2$ $d(x, v) = 4$

$$f(u) = \frac{1}{\lambda\left(\bar{B}(x, u)\right)} \qquad g(u) = u \qquad f(u) = \frac{1}{\lambda\left(\bar{B}(x, u)\right)}$$

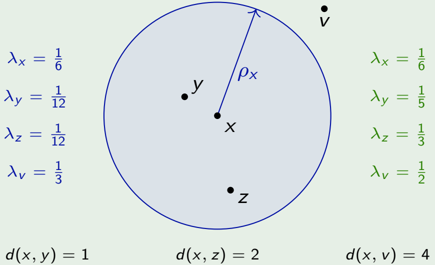
Radius definition — part 3

Finding a trade-off between distance and delay cost

ball with one border point



metric space and stochastic parameters

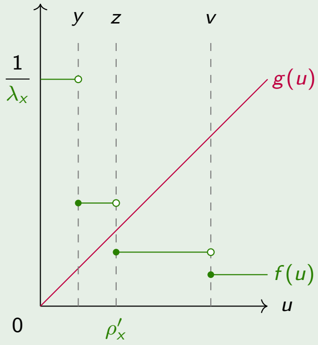


$f(u) = \dfrac{1}{\lambda\left(\bar{B}(x,u)\right)}$ $\quad g(u) = u \quad$ $f(u) = \dfrac{1}{\lambda\left(\bar{B}(x,u)\right)}$

Radius definition — part 3

Finding a trade-off between distance and delay cost

ball with one border point



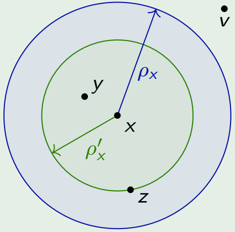
metric space and stochastic parameters

$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{12}$

$\lambda_z = \frac{1}{12}$

$\lambda_v = \frac{1}{3}$



$\lambda_x = \frac{1}{6}$

$\lambda_y = \frac{1}{5}$

$\lambda_z = \frac{1}{3}$

$\lambda_v = \frac{1}{2}$

$d(x, y) = 1$ $\quad d(x, z) = 2$ $\quad d(x, v) = 4$

$f(u) = \dfrac{1}{\lambda\left(\bar{B}(x, u)\right)}$ $\quad g(u) = u$ $\quad f(u) = \dfrac{1}{\lambda\left(\bar{B}(x, u)\right)}$

Bounding the minimum total cost of a request

- given a request r_i from the sequence σ, we want to lower bound

$$c(\sigma, r_i) := \min_{r' \in \sigma, r' \neq r_i} \left\{ d(\ell(r_i), \ell(r')) + |t(r_i) - t(r')| \right\}$$

Bounding the minimum total cost of a request

- given a request r_i from the sequence σ, we want to lower bound

$$c(\sigma, r_i) := \min_{r' \in \sigma, r' \neq r_i} \left\{ d(\ell(r_i), \ell(r')) + |t(r_i) - t(r')| \right\}$$

- we condition on $\ell(r_i) = x$ and define ρ_x to be the **radius** for x

Bounding the minimum total cost of a request

- given a request r_i from the sequence σ, we want to lower bound

$$c(\sigma, r_i) := \min_{r' \in \sigma, r' \neq r_i} \left\{ d(\ell(r_i), \ell(r')) + |t(r_i) - t(r')| \right\}$$

- we condition on $\ell(r_i) = x$ and define ρ_x to be the **radius** for x
- notice that r_i can be matched either with a request arriving **before**, or with a request arriving **after** its arrival time $t(r_i)$

Bounding the minimum total cost of a request

- given a request r_i from the sequence σ, we want to lower bound

$$c(\sigma, r_i) := \min_{r' \in \sigma, r' \neq r_i} \{d(\ell(r_i), \ell(r')) + |t(r_i) - t(r')|\}$$

- we condition on $\ell(r_i) = x$ and define ρ_x to be the **radius** for x
- notice that r_i can be matched either with a request arriving **before**, or with a request arriving **after** its arrival time $t(r_i)$
- then OPT pays the minimum of three costs:

Bounding the minimum total cost of a request

- given a request r_i from the sequence σ, we want to lower bound

$$c(\sigma, r_i) := \min_{r' \in \sigma, r' \neq r_i} \left\{ d(\ell(r_i), \ell(r')) + |t(r_i) - t(r')| \right\}$$

- we condition on $\ell(r_i) = x$ and define ρ_x to be the **radius** for x
- notice that r_i can be matched either with a request arriving **before**, or with a request arriving **after** its arrival time $t(r_i)$
- then OPT pays the minimum of three costs:
  - waiting time of the previous request generated within distance ρ_x

Bounding the minimum total cost of a request

- given a request r_i from the sequence σ, we want to lower bound

$$c(\sigma, r_i) := \min_{r' \in \sigma, r' \neq r_i} \left\{ d(\ell(r_i), \ell(r')) + |t(r_i) - t(r')| \right\}$$

- we condition on $\ell(r_i) = x$ and define ρ_x to be the **radius** for x
- notice that r_i can be matched either with a request arriving **before**, or with a request arriving **after** its arrival time $t(r_i)$
- then OPT pays the minimum of three costs:
  - waiting time of the previous request generated within distance ρ_x
  - waiting time for the next request generated within distance ρ_x

Bounding the minimum total cost of a request

- given a request r_i from the sequence σ, we want to lower bound

$$c(\sigma, r_i) := \min_{r' \in \sigma, r' \neq r_i} \{d(\ell(r_i), \ell(r')) + |t(r_i) - t(r')|\}$$

- we condition on $\ell(r_i) = x$ and define ρ_x to be the **radius** for x
- notice that r_i can be matched either with a request arriving **before**, or with a request arriving **after** its arrival time $t(r_i)$
- then OPT pays the minimum of three costs:
  - waiting time of the previous request generated within distance ρ_x
  - waiting time for the next request generated within distance ρ_x
  - distance to a request generated outside of ρ_x-circle

Bounding the minimum total cost of a request

- given a request r_i from the sequence σ, we want to lower bound

$$c(\sigma, r_i) := \min_{r' \in \sigma, r' \neq r_i} \left\{ d(\ell(r_i), \ell(r')) + |t(r_i) - t(r')| \right\}$$

- we condition on $\ell(r_i) = x$ and define ρ_x to be the **radius** for x
- notice that r_i can be matched either with a request arriving **before**, or with a request arriving **after** its arrival time $t(r_i)$
- then OPT pays the minimum of three costs:
  - waiting time of the previous request generated within distance ρ_x
  - waiting time for the next request generated within distance ρ_x
  - distance to a request generated outside of ρ_x-circle
- hence, it holds

$$c(\sigma, r_i) \geq \min\left(W_{prev}, W_{next}, \rho_x\right)$$

Taking the expectation

- we have that W_{prev} and W_{next} are **independent** and follow **the same distribution** $\mathrm{Exp}(1/\rho_x)$

Taking the expectation

- we have that W_{prev} and W_{next} are **independent** and follow **the same distribution** $\mathrm{Exp}(1/\rho_x)$
- one can calculate that

$$\mathbb{E}_{\sigma}^{m}[c(\sigma, r_i) \mid \ell(r_i) = x] \geq \mathbb{E}_{\sigma}^{m}\left[\min\left(W_{prev}, W_{next}, \rho_x\right)\right] = \frac{1 - e^{-2}}{2}\rho_x$$

Taking the expectation

- we have that W_{prev} and W_{next} are **independent** and follow **the same distribution** $\mathrm{Exp}(1/\rho_x)$
- one can calculate that

$$\mathbb{E}_\sigma^m[c(\sigma, r_i) \mid \ell(r_i) = x] \geq \mathbb{E}_\sigma^m\left[\min\left(W_{prev}, W_{next}, \rho_x\right)\right] = \frac{1 - e^{-2}}{2}\rho_x$$

- it is enough to upper bound $\mathbb{E}_\sigma^m[ALG(\sigma, r_i) \mid \ell(r_i) = x]$ by some $C \cdot \rho_x$

Greedy algorithm ALG

- at any time t if there exist pending requests r, r' such that $(t - t(r)) + (t - t(r')) \geq d(\ell(r), \ell(r'))$, match them into a pair with ties broken arbitrarily

Greedy algorithm ALG

- at any time t if there exist pending requests r, r' such that $(t - t(r)) + (t - t(r')) \geq d(\ell(r), \ell(r'))$, match them into a pair with ties broken arbitrarily

Analysis sketch

- **total cost** generated by ALG is **at most twice** its **delay cost**

Greedy algorithm

Greedy algorithm ALG

- at any time t if there exist pending requests r, r' such that $(t - t(r)) + (t - t(r')) \geq d(\ell(r), \ell(r'))$, match them into a pair with ties broken arbitrarily

Analysis sketch

- **total cost** generated by ALG is **at most twice** its **delay cost**
- for a request r_i arriving at x the expected **waiting time** to be matched by ALG is at most $\rho_x + \mathbb{E}[W_{next}] = 2\rho_x$

Greedy algorithm

Greedy algorithm ALG

- at any time t if there exist pending requests r, r' such that $(t - t(r)) + (t - t(r')) \geq d(\ell(r), \ell(r'))$, match them into a pair with ties broken arbitrarily

Analysis sketch

- **total cost** generated by ALG is **at most twice** its **delay cost**
- for a request r_i arriving at x the expected **waiting time** to be matched by ALG is at most $\rho_x + \mathbb{E}[W_{next}] = 2\rho_x$
- hence, the total cost of serving r_i does not exceed $4\rho_x$

Other results

1. second algorithm achieving a **better ratio** of expectations

Other results

1. second algorithm achieving a **better ratio** of expectations
2. analysis for a **general delay cost** described by an arbitrary **positive** and **non-decreasing** function

Other results

1. second algorithm achieving a **better ratio** of expectations
2. analysis for a **general delay cost** described by an arbitrary **positive** and **non-decreasing** function
3. variant with **penalties** to clear pending requests

Open problems

1. can we generalize this result to work for the bipartite case?

Open problems

1. can we generalize this result to work for the bipartite case?
2. is there a constant competitive algorithm for matching k-tuples?

Open problems

1. can we generalize this result to work for the bipartite case?
2. is there a constant competitive algorithm for matching k-tuples?
3. how to define and analyse similar problems (Facility Location, Online Service, Multi-Level Aggregation) in the stochastic environment?

Thank you!