Online Multi-Level Aggregation with Delays and Stochastic Arrival Times

Michał Pawłowski

University of Warsaw, IDEAS NCBR

November 10, 2023

Plan

online multi-level aggregation with delays

Plan

adversarial model

stochastic model

online multi-level aggregation with delays

Plan

adversarial model

stochastic model

online multi-level aggregation with delays

1

motivation example

Plan

adversarial model

stochastic model

online multi-level aggregation with delays

1 motivation example

2 problem statement

Plan

adversarial model

stochastic model

online multi-level aggregation with delays

1 motivation example

2 problem statement

3 previous results

Plan

adversarial model

stochastic model

online multi-level aggregation with delays

1. motivation example
2. problem statement
3. previous results
4. simple cases

Plan

adversarial model

stochastic model

online multi-level aggregation with delays

1. motivation example
2. problem statement
3. previous results
4. simple cases
5. general instance

Motivation

Supply chain management

Motivation

Supply chain management

- manager of a factory needs to **deliver** demanded products to the shops

Motivation

Supply chain management

- manager of a factory needs to **deliver** demanded products to the shops
- shop owners inform the manager about out-of-stock products and **wait** for the restocking

Motivation

Supply chain management

- manager of a factory needs to **deliver** demanded products to the shops
- shop owners inform the manager about out-of-stock products and **wait** for the restocking

$$\textbf{delay cost} = \text{waiting time}$$

Motivation

Supply chain management

- manager of a factory needs to **deliver** demanded products to the shops
- shop owners inform the manager about out-of-stock products and **wait** for the restocking

  delay cost = waiting time

- manager needs to **order a truck** to deliver products

Motivation

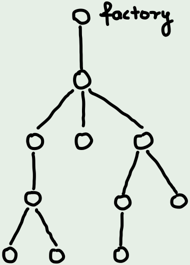
Supply chain management

- manager of a factory needs to **deliver** demanded products to the shops
- shop owners inform the manager about out-of-stock products and **wait** for the restocking

  delay cost = waiting time

- manager needs to **order a truck** to deliver products

  service cost = distance travelled



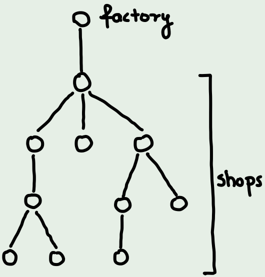
Motivation

Supply chain management

- manager of a factory needs to **deliver** demanded products to the shops
- shop owners inform the manager about out-of-stock products and **wait** for the restocking

  delay cost = waiting time
- manager needs to **order a truck** to deliver products

  service cost = distance travelled



Motivation

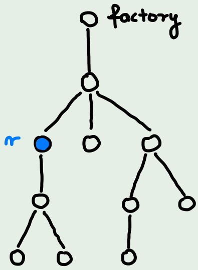
Supply chain management

- manager of a factory needs to **deliver** demanded products to the shops
- shop owners inform the manager about out-of-stock products and **wait** for the restocking

$$\textbf{delay cost} = \text{waiting time}$$

- manager needs to **order a truck** to deliver products

$$\textbf{service cost} = \text{distance travelled}$$



Motivation

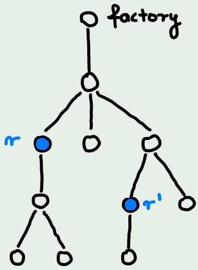
Supply chain management

- manager of a factory needs to **deliver** demanded products to the shops
- shop owners inform the manager about out-of-stock products and **wait** for the restocking

$$\textbf{delay cost} = \text{waiting time}$$

- manager needs to **order a truck** to deliver products

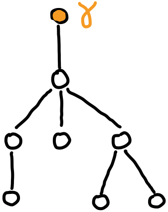
$$\textbf{service cost} = \text{distance travelled}$$



Problem statement

Multi-Level Aggregation with Delays

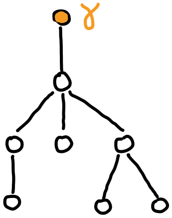
- **tree** T **rooted** at some node γ and a **weight** function $w : E(T) \to \mathbb{R}_+$



Problem statement

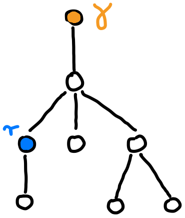
Multi-Level Aggregation with Delays

- **tree** T **rooted** at some node γ and a **weight** function $w : E(T) \to \mathbb{R}_+$
- requests arrive at **arbitrary times** at nodes $V(T)$



Multi-Level Aggregation with Delays

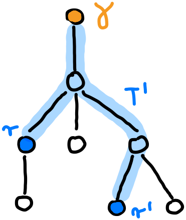
- **tree** T **rooted** at some node γ and a **weight** function $w : E(T) \to \mathbb{R}_+$
- requests arrive at **arbitrary times** at nodes $V(T)$
- each request r is characterized by its
  - **location** $\ell(r) \in V(T)$ and
  - **arrival time** $t(r) \in \mathbb{R}_+$



Problem statement

Multi-Level Aggregation with Delays

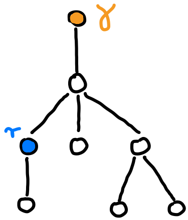
- **tree** T **rooted** at some node γ and a **weight** function $w : E(T) \to \mathbb{R}_+$
- requests arrive at **arbitrary times** at nodes $V(T)$
- each request r is characterized by its
  - **location** $\ell(r) \in V(T)$ and
  - **arrival time** $t(r) \in \mathbb{R}_+$



- to **serve** any set of requests R at time t, a **subtree** T' **containing the tree root** and locations of all the requests in R needs to be bought at a service cost equal to the total weight of edges in T'

Multi-Level Aggregation with Delays

- **tree** T **rooted** at some node γ and a **weight** function $w : E(T) \to \mathbb{R}_+$
- requests arrive at **arbitrary times** at nodes $V(T)$
- each request r is characterized by its
  - **location** $\ell(r) \in V(T)$ and
  - **arrival time** $t(r) \in \mathbb{R}_+$



- to **serve** any set of requests R at time t, a **subtree** T' **containing the tree root** and locations of all the requests in R needs to be bought at a service cost equal to the total weight of edges in T'
- the target is to **minimize the total cost** produced by the online algorithm for serving all the requests

Adversarial model

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

Adversarial model

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

online setting

Adversarial model

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

online setting

- the **current best** competitiveness is $O(d^2)$ (where d denotes the depth of the tree) [Azar, Touitou, FOCS'19]

Adversarial model

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

online setting

- the **current best** competitiveness is $O(d^2)$ (where d denotes the depth of the tree) [Azar, Touitou, FOCS'19]
- **no** online algorithm can achieve a competitive ratio **better than** $2 + \phi \approx 3.618$ [Bieńkowski et al., WADS'13]

Adversarial model

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

online setting

- the **current best** competitiveness is $O(d^2)$ (where d denotes the depth of the tree) [Azar, Touitou, FOCS'19]
- **no** online algorithm can achieve a competitive ratio **better than** $2 + \phi \approx 3.618$ [Bieńkowski et al., WADS'13]

offline setting

Adversarial model

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

online setting

- the **current best** competitiveness is $O(d^2)$ (where d denotes the depth of the tree) [Azar, Touitou, FOCS'19]
- **no** online algorithm can achieve a competitive ratio **better than** $2 + \phi \approx 3.618$ [Bieńkowski et al., WADS'13]

offline setting

- the problem is **NP-hard** in both deadline and delay versions [Arkin et al., Becchetti et al.]

Adversarial model

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

online setting

- the **current best** competitiveness is $O(d^2)$ (where d denotes the depth of the tree) [Azar, Touitou, FOCS'19]
- **no** online algorithm can achieve a competitive ratio **better than** $2 + \phi \approx 3.618$ [Bieńkowski et al., WADS'13]

offline setting

- the problem is **NP-hard** in both deadline and delay versions [Arkin et al., Becchetti et al.]
- **2-approximation** algorithm was proposed by Becchetti et al.

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
- factory owners have all the **historical data** and can **estimate the frequency** of the requests from a given shop

Stochastic model

- for many applications it is **too pessimistic** to assume that **no stochastic information** on the input is available
- factory owners have all the **historical data** and can **estimate the frequency** of the requests from a given shop
- 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
- factory owners have all the **historical data** and can **estimate the frequency** of the requests from a given shop
- 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
- factory owners have all the **historical data** and can **estimate the frequency** of the requests from a given shop
- we can assume that **requests follow a stochastic distribution**

Poisson arrival process

- **waiting time** between any two consecutive requests arriving at any node v follows an **exponential distribution** $\mathrm{Exp}(\lambda_v)$

How to compare performance?

Ratio of expectations

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

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

How to compare performance?

Ratio of expectations

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

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

where $\mathbb{E}_\sigma^\tau[\mathrm{ALG}(\sigma)]$ (resp. $\mathbb{E}_\sigma^\tau[\mathrm{OPT}(\sigma)]$) denotes the **expected cost** generated by ALG (resp. an **optimal offline solution**) on the **random request sequence** σ generated by the Poisson arrival process within the time interval $[0, \tau]$.

How to compare performance?

Ratio of expectations

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

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

where $\mathbb{E}^{\tau}_{\sigma}[\mathrm{ALG}(\sigma)]$ (resp. $\mathbb{E}^{\tau}_{\sigma}[\mathrm{OPT}(\sigma)]$) denotes the **expected cost** generated by ALG (resp. an **optimal offline solution**) on the **random request sequence** σ generated by the Poisson arrival process within the time interval $[0, \tau]$.

Main theorem

For MLA with linear delays in the Poisson arrival model, there exists a deterministic algorithm that achieves a **constant ratio of expectations**.

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 $Z := \min\{X_1, X_2, \ldots, X_n\}$ 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.

Poisson arrival process

distributed version

- waiting time between any two consecutive requests arriving at any **node** v follows an exponential distribution $\mathrm{Exp}(\lambda_v)$

Poisson arrival process

distributed version

- waiting time between any two consecutive requests arriving at any **node** v follows an exponential distribution $\mathrm{Exp}(\lambda_v)$

centralized version

- waiting time between any two consecutive requests in the given **tree** T follows an exponential distribution with parameter $\lambda(V(T)) := \sum_{v \in V(T)} \lambda_v$
- each time a request arrives, the probability of it appearing at node v equals $\lambda_v / \lambda(V(T))$

One edge — two approaches

edge $e = (\gamma, u)$: **weight** w, **arrival rate** of node u equal λ

One edge — two approaches

$X \sim \mathrm{Exp}(\lambda)$ then $\mathbb{E}[X] = 1/\lambda$

edge $e = (\gamma, u)$: **weight** w, **arrival rate** of node u equal λ

One edge — two approaches

$X \sim \mathrm{Exp}(\lambda)$ then $\mathbb{E}[X] = 1/\lambda$

edge $e = (\gamma, u)$: **weight** w, **arrival rate** of node u equal λ

$w < 1/\lambda$

$w \geq 1/\lambda$

One edge — two approaches

$X \sim \mathrm{Exp}(\lambda)$ then $\mathbb{E}[X] = 1/\lambda$

edge $e = (\gamma, u)$: **weight** w, **arrival rate** of node u equal λ

$w < 1/\lambda$

- **not worth waiting** for the next request to serve both of them together

$w \geq 1/\lambda$

One edge — two approaches

$$X \sim \mathrm{Exp}(\lambda) \text{ then } \mathbb{E}[X] = 1/\lambda$$

edge $e = (\gamma, u)$: **weight** w, **arrival rate** of node u equal λ

$w < 1/\lambda$

- **not worth waiting** for the next request to serve both of them together
- ALG: serve all the requests **immediately** at the moment of their arrival

$w \geq 1/\lambda$

One edge — two approaches

$X \sim \mathrm{Exp}(\lambda)$ then $\mathbb{E}[X] = 1/\lambda$

edge $e = (\gamma, u)$: **weight** w, **arrival rate** of node u equal λ

$w < 1/\lambda$

- **not worth waiting** for the next request to serve both of them together
- ALG: serve all the requests **immediately** at the moment of their arrival
- cost: $\lambda\tau \cdot w$

$w \geq 1/\lambda$

One edge — two approaches

$X \sim \mathrm{Exp}(\lambda)$ then $\mathbb{E}[X] = 1/\lambda$

edge $e = (\gamma, u)$: **weight** w, **arrival rate** of node u equal λ

$w < 1/\lambda$

- **not worth waiting** for the next request to serve both of them together
- ALG: serve all the requests **immediately** at the moment of their arrival
- cost: $\lambda\tau \cdot w$

$w \geq 1/\lambda$

- **delaying** service in this case is profitable

One edge — two approaches

$X \sim \mathrm{Exp}(\lambda)$ then $\mathbb{E}[X] = 1/\lambda$

edge $e = (\gamma, u)$: **weight** w, **arrival rate** of node u equal λ

$w < 1/\lambda$

- **not worth waiting** for the next request to serve both of them together
- ALG: serve all the requests **immediately** at the moment of their arrival
- cost: $\lambda\tau \cdot w$

$w \geq 1/\lambda$

- **delaying** service in this case is profitable
- ALG: **wait until** the total expected delay cost equals w, i.e., for $p = \sqrt{2w/\lambda}$, serve all the request then

One edge — two approaches

$X \sim \mathrm{Exp}(\lambda)$ then $\mathbb{E}[X] = 1/\lambda$

edge $e = (\gamma, u)$: **weight** w, **arrival rate** of node u equal λ

$w < 1/\lambda$

- **not worth waiting** for the next request to serve both of them together
- ALG: serve all the requests **immediately** at the moment of their arrival
- cost: $\lambda\tau \cdot w$

$w \geq 1/\lambda$

- **delaying** service in this case is profitable
- ALG: **wait until** the total expected delay cost equals w, i.e., for $p = \sqrt{2w/\lambda}$, serve all the request then
- cost: $(\lambda p^2/2 + w) \cdot \tau/p$

One edge — light case

$$w < 1/\lambda$$

One edge — light case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w < 1/\lambda$$

One edge — light case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w < 1/\lambda$$

- **divide timeline** into intervals of length $1/\lambda$

One edge — light case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w < 1/\lambda$$

- **divide timeline** into intervals of length $1/\lambda$
- let us analyse the **cost of OPT** within one interval

One edge — light case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w < 1/\lambda$$

- **divide timeline** into intervals of length $1/\lambda$
- let us analyse the **cost of OPT** within one interval
- probability that ≥ 1 request arrives equals $1 - e^{-1}$

One edge — light case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w < 1/\lambda$$

- **divide timeline** into intervals of length $1/\lambda$
- let us analyse the **cost of OPT** within one interval
- probability that ≥ 1 request arrives equals $1 - e^{-1}$
- assume that **at least one request arrives** at node u

One edge — light case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w < 1/\lambda$$

- **divide timeline** into intervals of length $1/\lambda$
- let us analyse the **cost of OPT** within one interval
- probability that ≥ 1 request arrives equals $1 - e^{-1}$
- assume that **at least one request arrives** at node u
- then OPT pays at least $\min(D, w)$, $D \sim \mathrm{U}(1/\lambda)$

One edge — light case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w < 1/\lambda$$

- **divide timeline** into intervals of length $1/\lambda$
- let us analyse the **cost of OPT** within one interval
- probability that ≥ 1 request arrives equals $1 - e^{-1}$
- assume that **at least one request arrives** at node u
- then OPT pays at least $\min(D, w)$, $D \sim \mathrm{U}(1/\lambda)$
- thus, in expectation OPT pays at least $w/2$

One edge — light case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w < 1/\lambda$$

- **divide timeline** into intervals of length $1/\lambda$
- let us analyse the **cost of OPT** within one interval
- probability that ≥ 1 request arrives equals $1 - e^{-1}$
- assume that **at least one request arrives** at node u
- then OPT pays at least $\min(D, w)$, $D \sim \mathrm{U}(1/\lambda)$
- thus, in expectation OPT pays at least $w/2$
- in total it gives us $\lambda\tau \cdot (1 - e^{-1})w/2$

One edge — heavy case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w \geq 1/\lambda$$

One edge — heavy case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w \geq 1/\lambda$$

- **divide timeline** into intervals of length $p = \sqrt{2w/\lambda}$

One edge — heavy case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w \geq 1/\lambda$$

- **divide timeline** into intervals of length $p = \sqrt{2w/\lambda}$
- let us analyse the **cost of OPT** within a given interval

One edge — heavy case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w \geq 1/\lambda$$

- **divide timeline** into intervals of length $p = \sqrt{2w/\lambda}$
- let us analyse the **cost of OPT** within a given interval
- probability that $\geq n_0 = \lceil \lambda p \rceil$ requests arrive is at least $1/2$

One edge — heavy case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w \geq 1/\lambda$$

- **divide timeline** into intervals of length $p = \sqrt{2w/\lambda}$
- let us analyse the **cost of OPT** within a given interval
- probability that $\geq n_0 = \lceil \lambda p \rceil$ requests arrive is at least $1/2$
- assume that **at least n_0 requests arrive** at node u

One edge — heavy case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w \geq 1/\lambda$$

- **divide timeline** into intervals of length $p = \sqrt{2w/\lambda}$
- let us analyse the **cost of OPT** within a given interval
- probability that $\geq n_0 = \lceil \lambda p \rceil$ requests arrive is at least $1/2$
- assume that **at least** n_0 **requests arrive** at node u
- then OPT pays at least $\min(w, \sum_{j=1}^{n_0} U_j)$, $U_j \sim \mathrm{U(p)}$

One edge — heavy case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w \geq 1/\lambda$$

- **divide timeline** into intervals of length $p = \sqrt{2w/\lambda}$
- let us analyse the **cost of OPT** within a given interval
- probability that $\geq n_0 = \lceil \lambda p \rceil$ requests arrive is at least $1/2$
- assume that **at least** n_0 **requests arrive** at node u
- then OPT pays at least $\min(w, \sum_{j=1}^{n_0} U_j)$, $U_j \sim \mathrm{U(p)}$
- we can show that in expectation OPT pays at least $3/4w$

One edge — heavy case

in a Poisson process with parameter λ over $[0, \tau]$ **conditioned** on n requests arriving, all the arrival times follow a **uniform** distribution $\mathrm{U}(\tau)$

$$w \geq 1/\lambda$$

- **divide timeline** into intervals of length $p = \sqrt{2w/\lambda}$
- let us analyse the **cost of OPT** within a given interval
- probability that $\geq n_0 = \lceil \lambda p \rceil$ requests arrive is at least $1/2$
- assume that **at least** n_0 **requests arrive** at node u
- then OPT pays at least $\min(w, \sum_{j=1}^{n_0} U_j)$, $U_j \sim \mathrm{U(p)}$
- we can show that in expectation OPT pays at least $3/4w$
- in total it gives us $3/8w \cdot \tau/p$

Heavy instances

let us consider an **edge-weighted tree** T

Heavy instances

let us consider an **edge-weighted tree** T

assume that **for each edge** $e = (u, v)$ in T, $w(e) \geq 1/\lambda(v)$?

Heavy instances

let us consider an **edge-weighted tree** T

assume that **for each edge** $e = (u, v)$ in T, $w(e) \geq 1/\lambda(v)$?
should we serve this tree **periodically**?

Heavy instances

in a Poisson process with parameter λ the expected **total waiting time** generated by requests within the time interval $[0, \tau]$ is $\lambda\tau^2/2$

let us consider an **edge-weighted tree** T

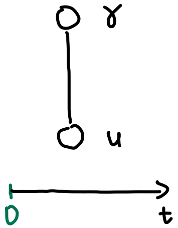
assume that **for each edge** $e = (u, v)$ in T, $w(e) \geq 1/\lambda(v)$?
should we serve this tree **periodically**?

Heavy instances

in a Poisson process with parameter λ the expected **total waiting time** generated by requests within the time interval $[0, \tau]$ is $\lambda\tau^2/2$

let us consider an **edge-weighted tree** T

assume that **for each edge** $e = (u, v)$ in T, $w(e) \geq 1/\lambda(v)$?
should we serve this tree **periodically**?

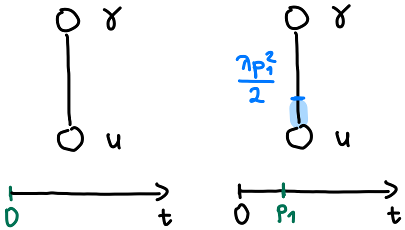


Heavy instances

in a Poisson process with parameter λ the expected **total waiting time** generated by requests within the time interval $[0, \tau]$ is $\lambda\tau^2/2$

let us consider an **edge-weighted tree** T

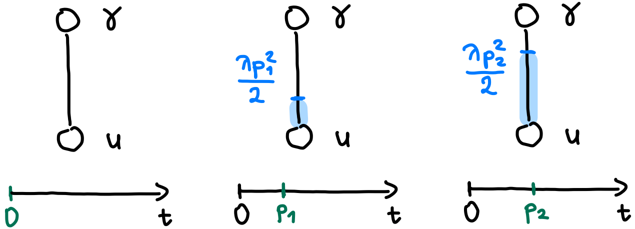
assume that **for each edge** $e = (u, v)$ in T, $w(e) \geq 1/\lambda(v)$?
should we serve this tree **periodically**?



in a Poisson process with parameter λ the expected **total waiting time** generated by requests within the time interval $[0, \tau]$ is $\lambda\tau^2/2$

let us consider an **edge-weighted tree** T

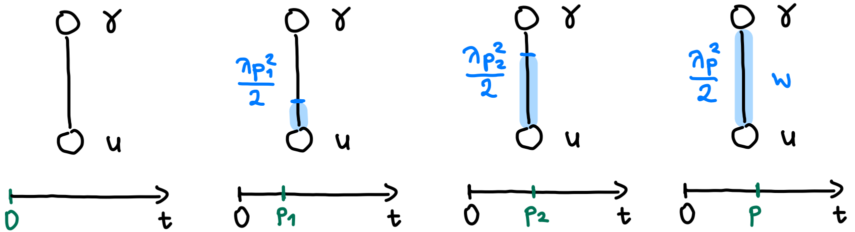
assume that **for each edge** $e = (u, v)$ in T, $w(e) \geq 1/\lambda(v)$?
should we serve this tree **periodically**?



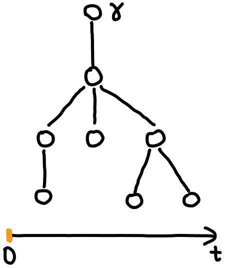
in a Poisson process with parameter λ the expected **total waiting time** generated by requests within the time interval $[0, \tau]$ is $\lambda\tau^2/2$

let us consider an **edge-weighted tree** T

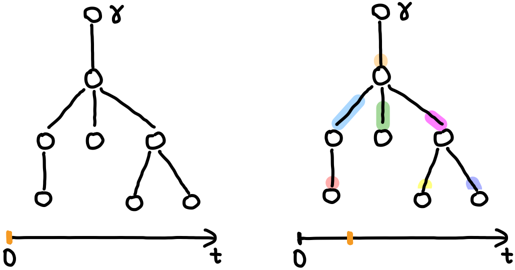
assume that **for each edge** $e = (u, v)$ in T, $w(e) \geq 1/\lambda(v)$?
should we serve this tree **periodically**?



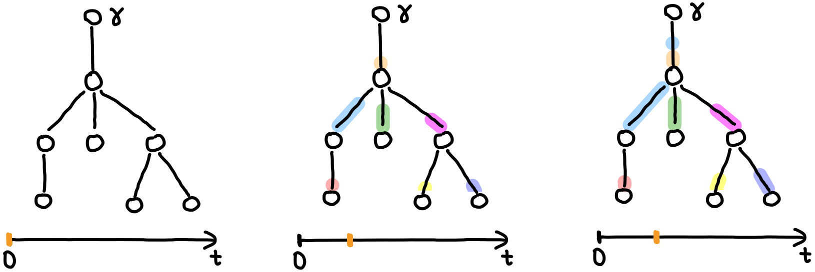
Heavy instances — clusters



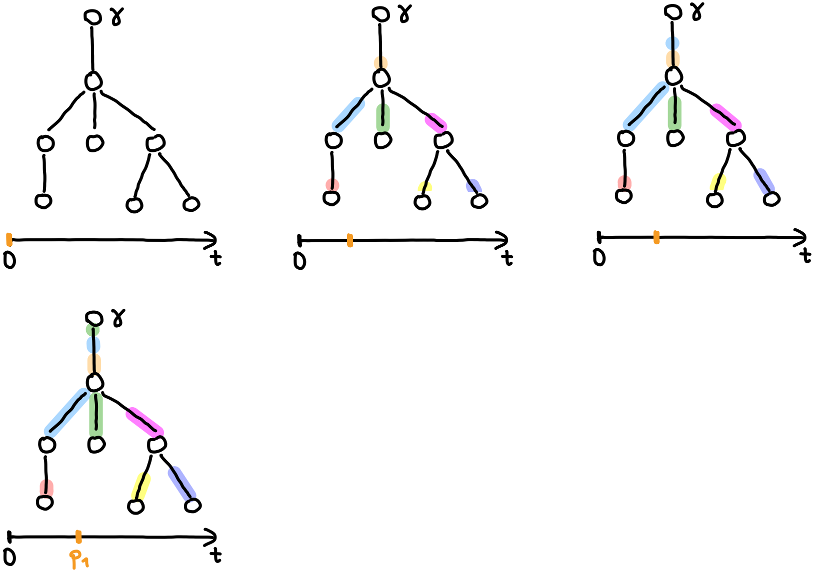
Heavy instances — clusters



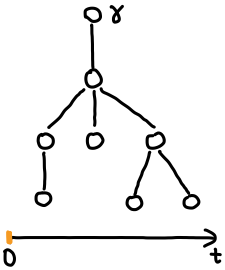
Heavy instances — clusters

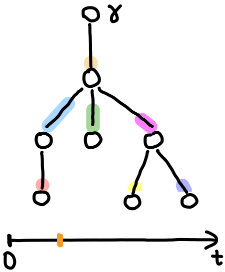


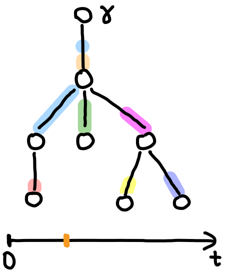
Heavy instances — clusters

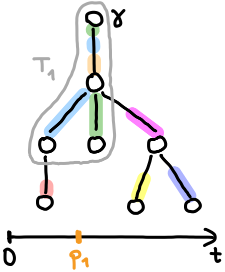


Heavy instances — clusters

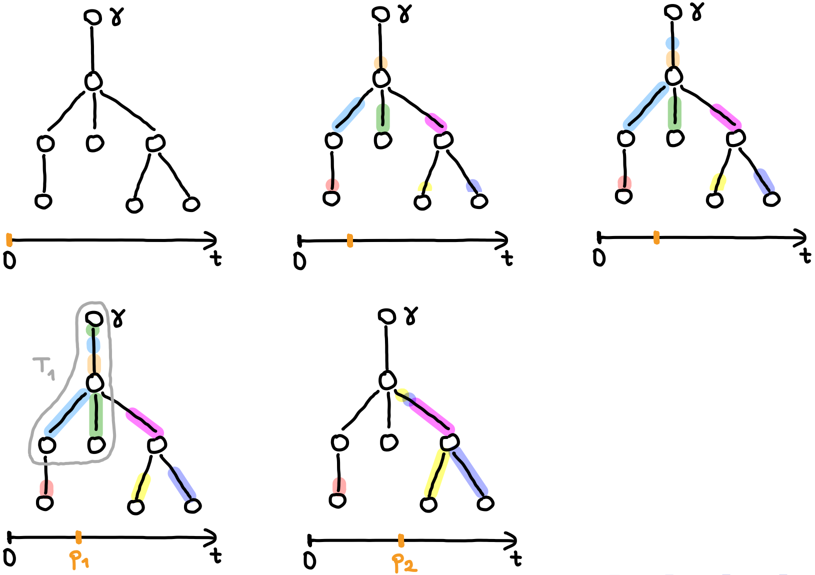




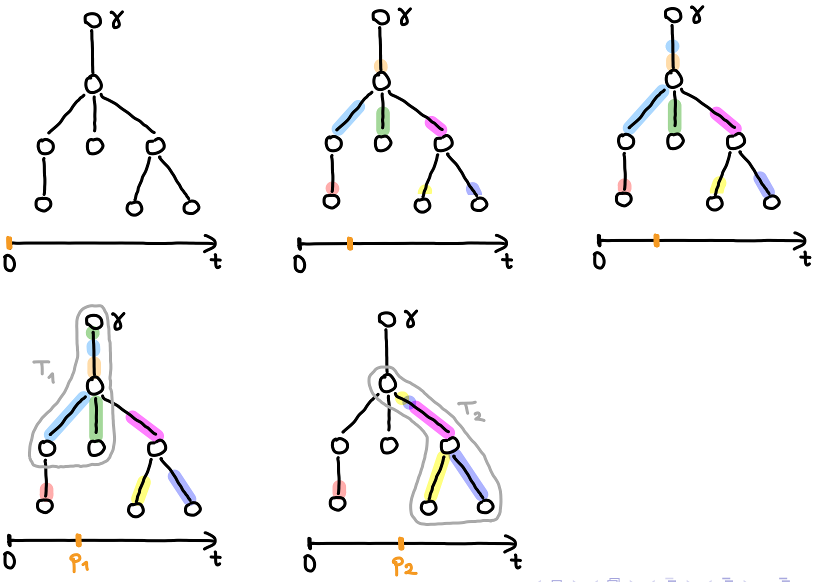




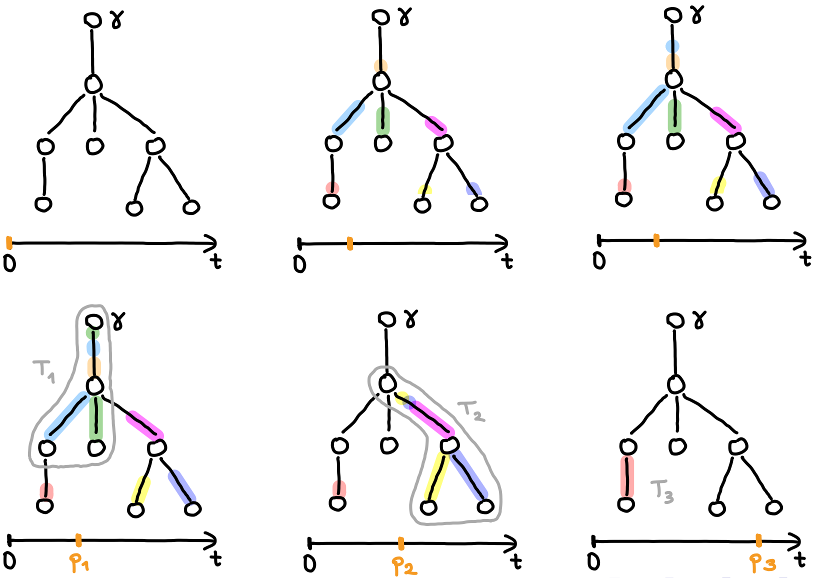
Heavy instances — clusters



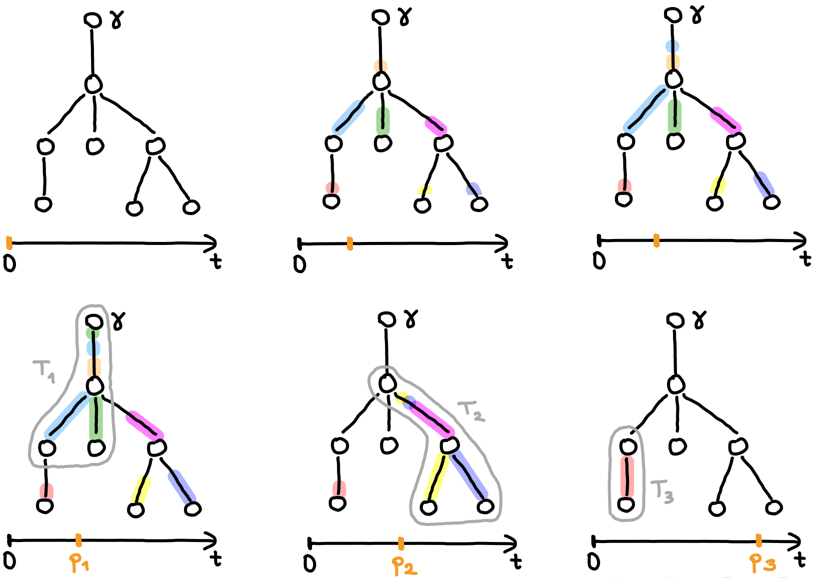
Heavy instances — clusters



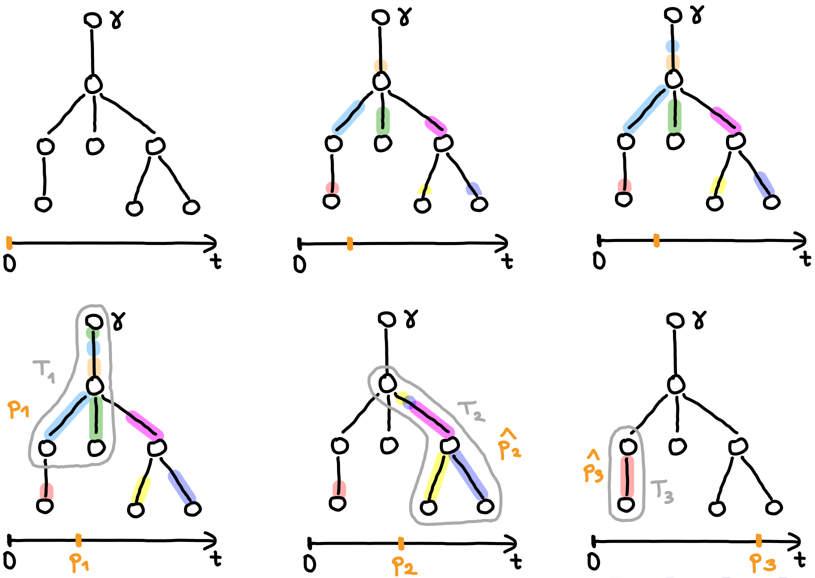
Heavy instances — clusters



Heavy instances — clusters



Heavy instances — clusters



Heavy instances — analysis

assume that a **heavy tree** T is assigned **period** p

Heavy instances — analysis

assume that a **heavy tree** T is assigned **period** p
denote the **cost share** of node v by $\hat{w}_v := \lambda(v)/2 \cdot p^2$

Heavy instances — analysis

assume that a **heavy tree** T is assigned **period** p
denote the **cost share** of node v by $\hat{w}_v := \lambda(v)/2 \cdot p^2$

$$\sum_{v \in T} \frac{\lambda(v) \cdot p^2}{2} = \sum_{v \in T} \hat{w}_v = \sum_{v \in T} w_v \geq \sum_{v \in T} \frac{1}{\lambda(v)}.$$

Heavy instances — analysis

assume that a **heavy tree** T is assigned **period** p
denote the **cost share** of node v by $\hat{w}_v := \lambda(v)/2 \cdot p^2$

$$\sum_{v \in T} \frac{\lambda(v) \cdot p^2}{2} = \sum_{v \in T} \hat{w}_v = \sum_{v \in T} w_v \geq \sum_{v \in T} \frac{1}{\lambda(v)}.$$

let $L \subseteq T$ be the subset of nodes v for which $p \leq 1/\lambda(v)$

Heavy instances — analysis

assume that a **heavy tree** T is assigned **period** p
denote the **cost share** of node v by $\hat{w}_v := \lambda(v)/2 \cdot p^2$

$$\sum_{v \in T} \frac{\lambda(v) \cdot p^2}{2} = \sum_{v \in T} \hat{w}_v = \sum_{v \in T} w_v \geq \sum_{v \in T} \frac{1}{\lambda(v)}.$$

let $L \subseteq T$ be the subset of nodes v for which $p \leq 1/\lambda(v)$

$$\begin{aligned}\sum_{v \in L} \hat{w}_v &= \sum_{v \in L} \frac{\lambda(v) \cdot p^2}{2} \leq \frac{1}{2} \sum_{v \in L} \frac{\lambda(v)}{\lambda^2(v)} \\ &= \frac{1}{2} \sum_{v \in L} \frac{1}{\lambda(v)} \leq \frac{1}{2} \sum_{v \in T} \frac{1}{\lambda(v)} \leq \frac{1}{2} \sum_{v \in T} w_v\end{aligned}$$

Light instances

Definition

A stochastic MLA instance (T, w, λ) is called light if

$$\sum_{v \in V(T)} \frac{\lambda(v)}{\lambda(T)} d(v, \gamma) \leq \frac{1}{\lambda(T)},$$

where d is the distance function based on w.

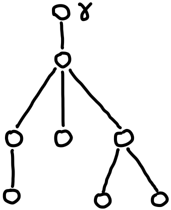
Light instances

Definition

A stochastic MLA instance (T, w, λ) is called light if

$$\sum_{v \in V(T)} \frac{\lambda(v)}{\lambda(T)} d(v, \gamma) \leq \frac{1}{\lambda(T)},$$

where d is the distance function based on w.



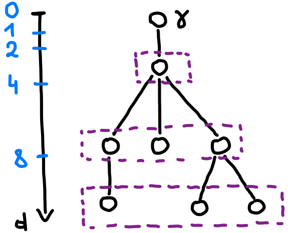
Light instances

Definition

A stochastic MLA instance (T, w, λ) is called light if

$$\sum_{v \in V(T)} \frac{\lambda(v)}{\lambda(T)} d(v, \gamma) \leq \frac{1}{\lambda(T)},$$

where d is the distance function based on w.



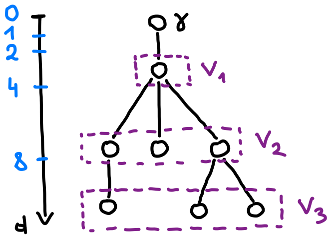
Light instances

Definition

A stochastic MLA instance (T, w, λ) is called light if

$$\sum_{v \in V(T)} \frac{\lambda(v)}{\lambda(T)} d(v, \gamma) \leq \frac{1}{\lambda(T)},$$

where d is the distance function based on w.



$V_j = \{ v \in V(T) : 2^j \leq d(v,\gamma) < 2^{j+1} \}$

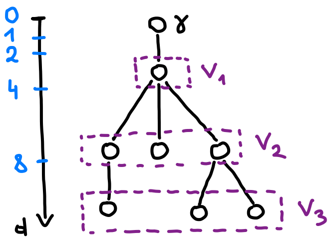
Light instances

Definition

A stochastic MLA instance (T, w, λ) is called light if

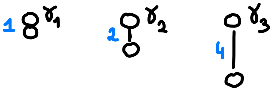
$$\sum_{v \in V(T)} \frac{\lambda(v)}{\lambda(T)} d(v, \gamma) \leq \frac{1}{\lambda(T)},$$

where d is the distance function based on w.



$V_j = \{v \in V(T) : 2^j \leq d(v,\gamma) < 2^{j+1}\}$

$w_j = 2^{j-1}$



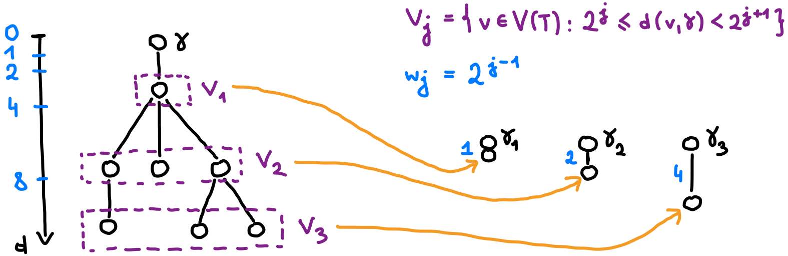
Light instances

Definition

A stochastic MLA instance (T, w, λ) is called light if

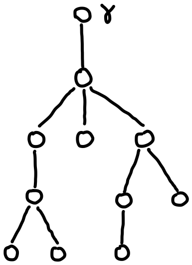
$$\sum_{v \in V(T)} \frac{\lambda(v)}{\lambda(T)} d(v, \gamma) \leq \frac{1}{\lambda(T)},$$

where d is the distance function based on w.

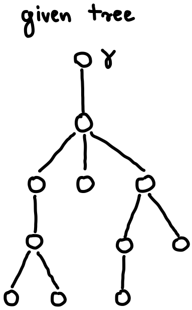


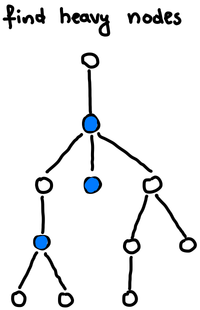
General case

given tree

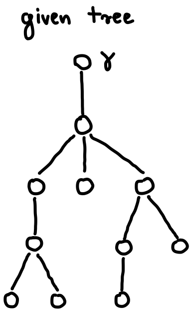


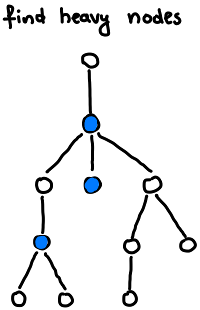
General case

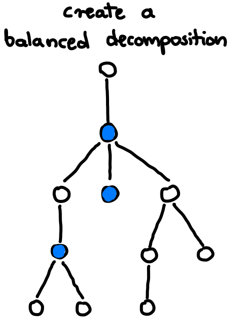




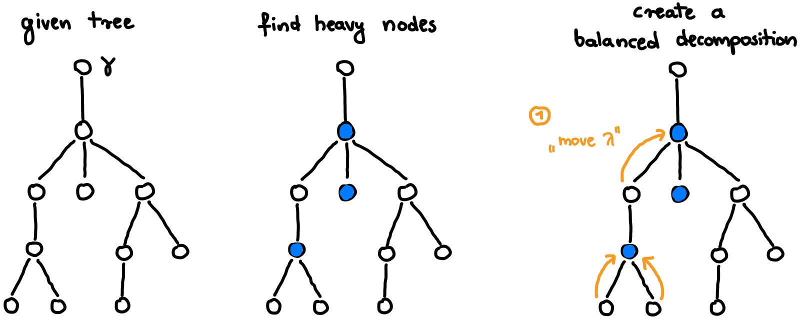
General case



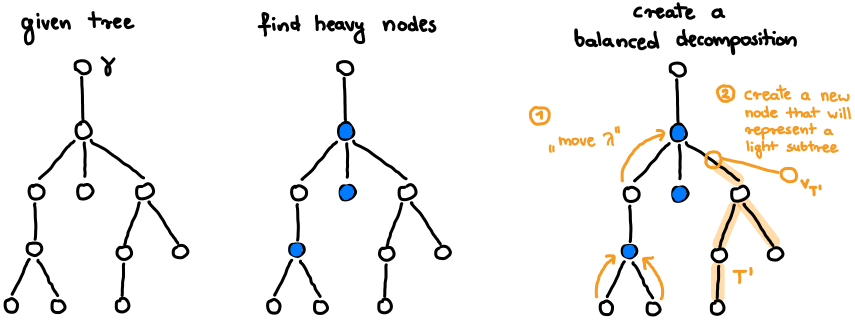




General case



General case



Open problems

1. does the **greedy** algorithm achieve a **constant ratio** of expectations?

Open problems

1. does the **greedy** algorithm achieve a **constant ratio** of expectations?
2. how to define and analyse **similar problems** (Facility Location, Online Service) in the stochastic environment?

Thank you!