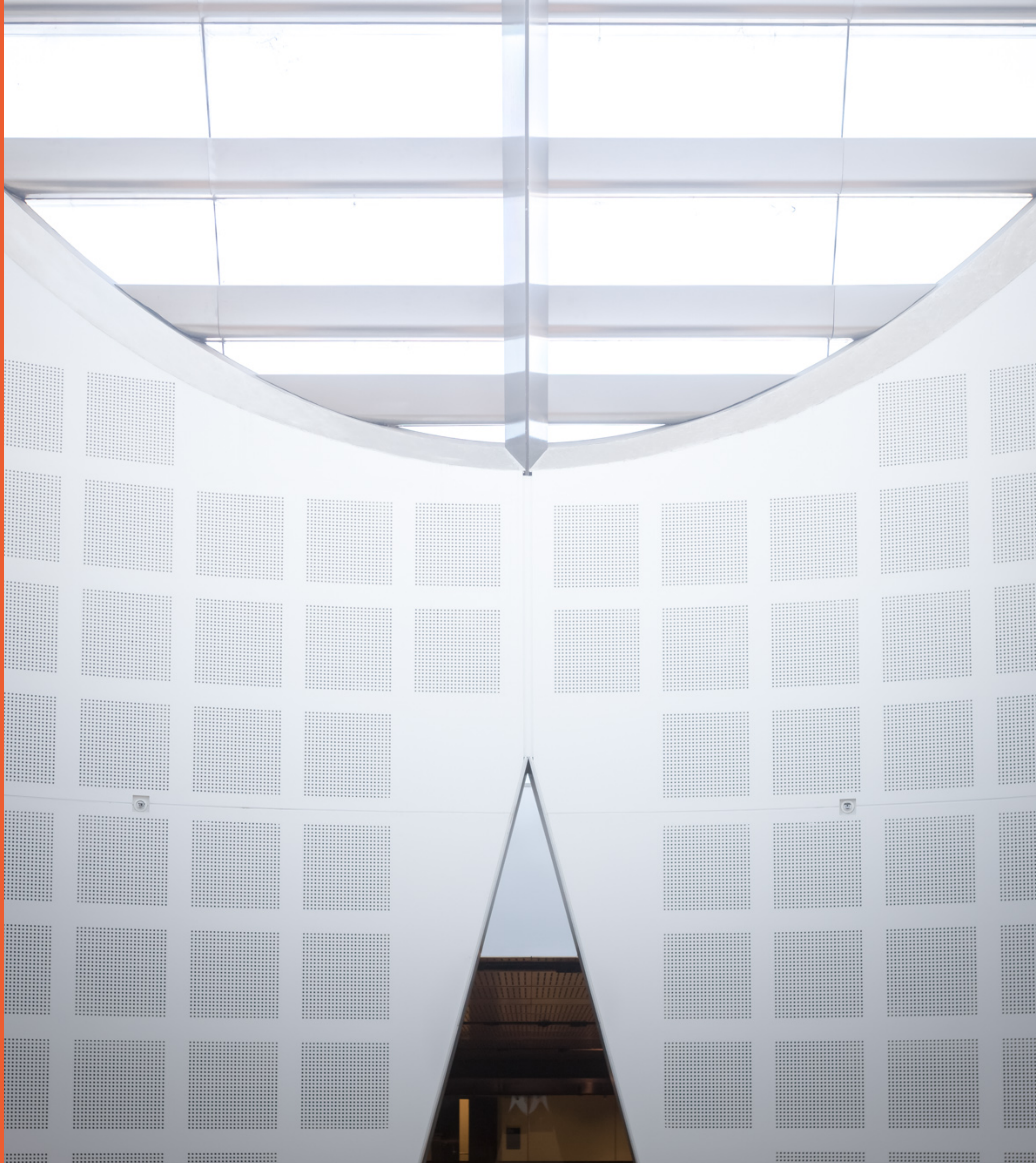
Non-clairvoyant problems with Deadlines or Delay

Presented by: Mai Le

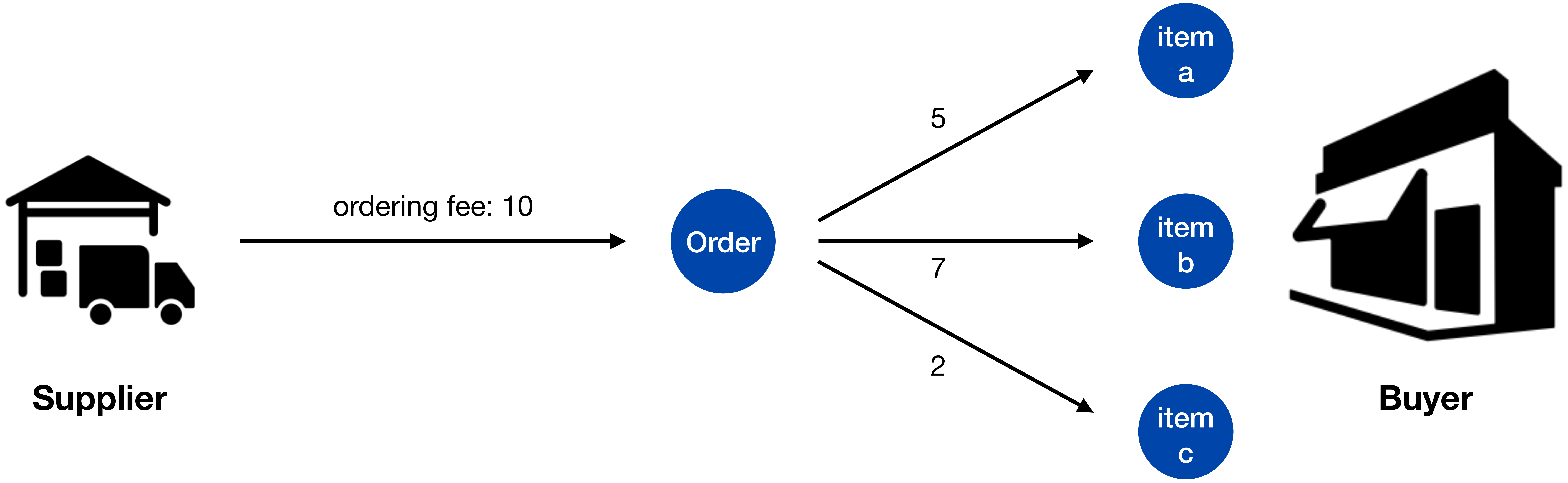
Supervised by: Dr. Seeun William Umboh

School of Computer Science

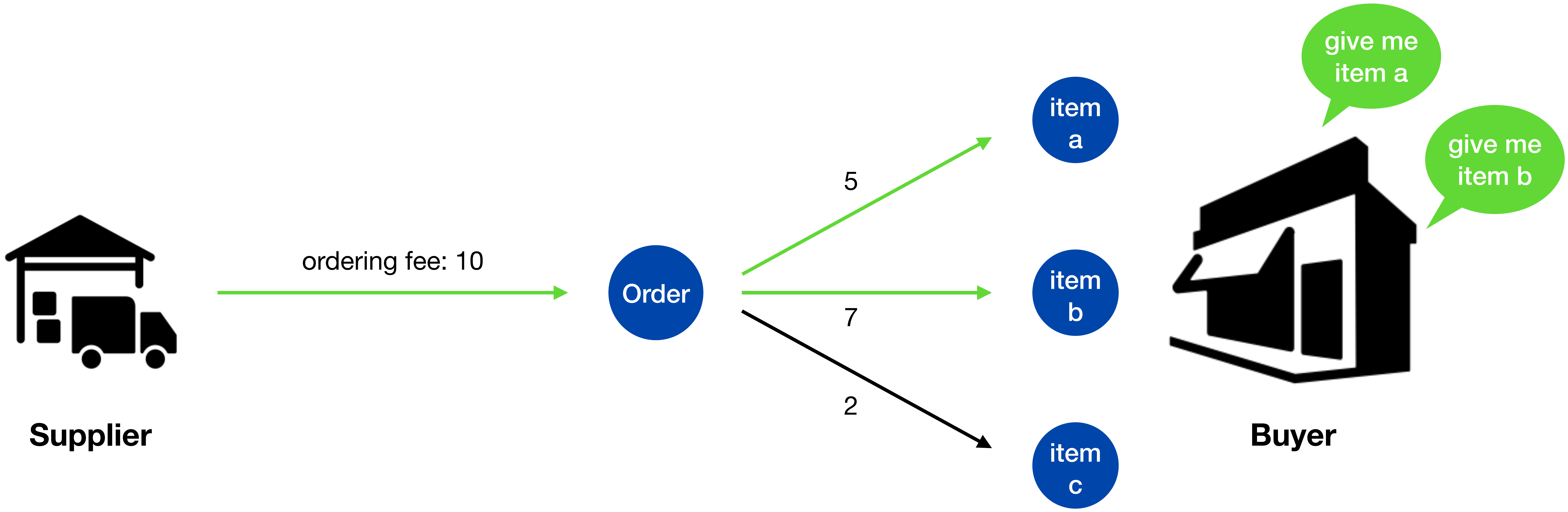
THE UNIVERSITY OF SYDNEY



Joint replenishment problem (JRP)

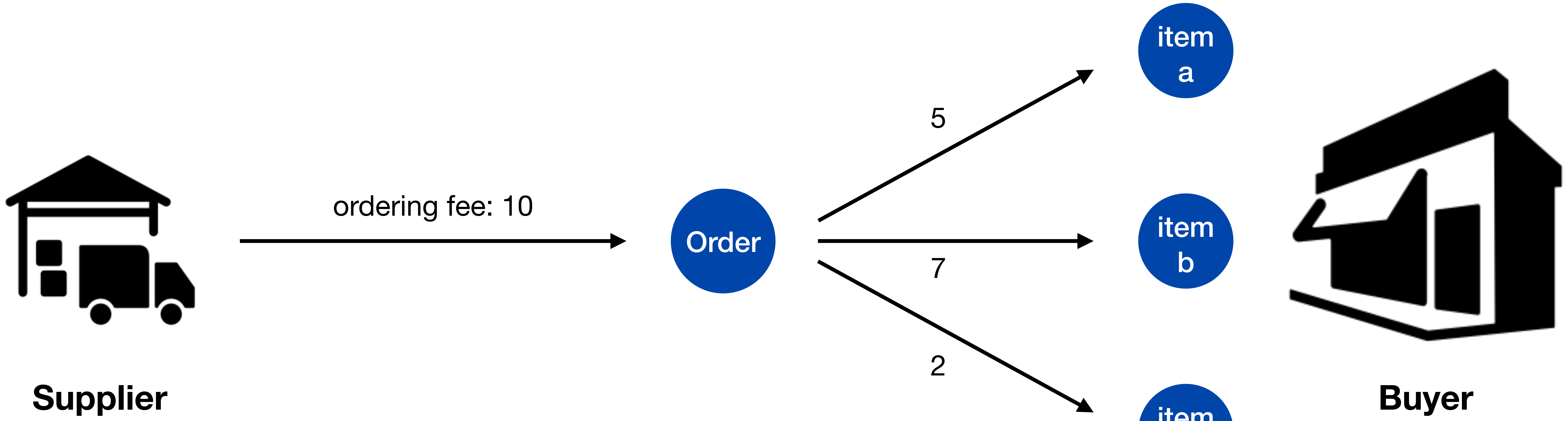


Joint replenishment problem



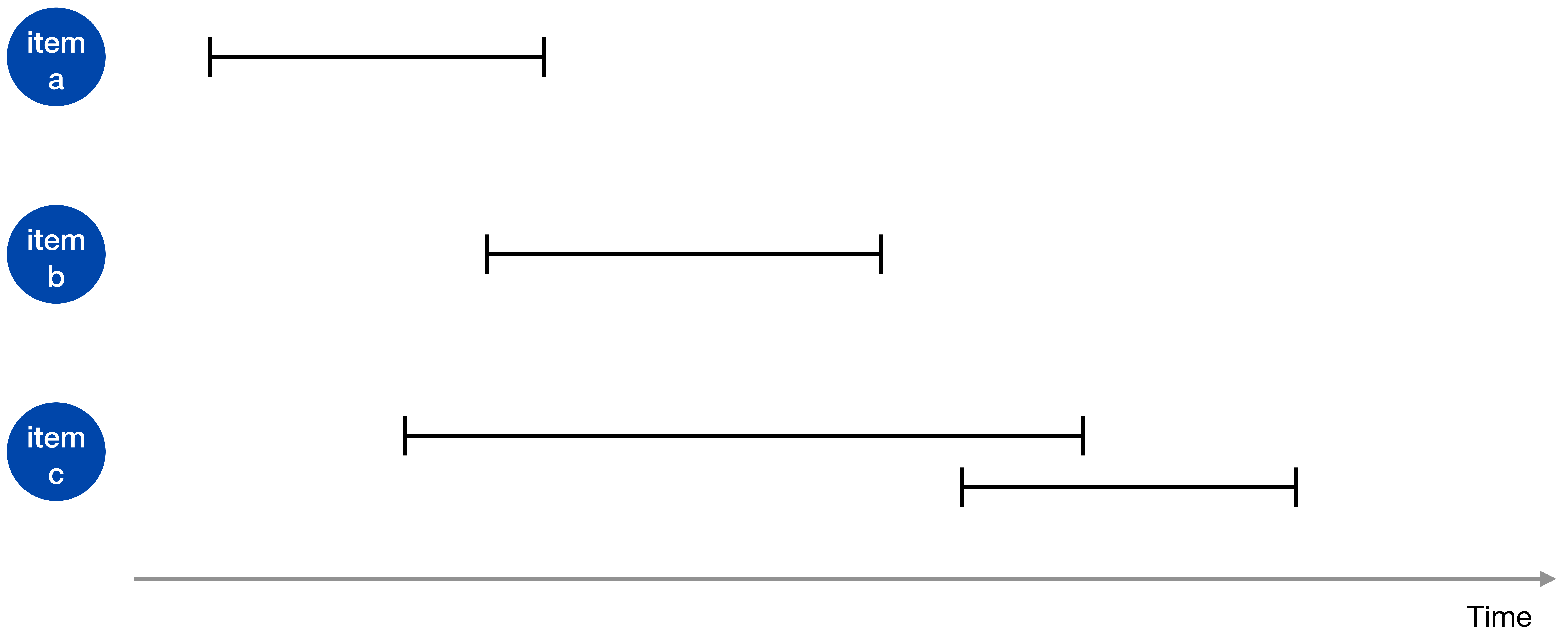
- Each request is on an item
- A transmission is a set of items delivered to the buyer
- A request is served if the item it occurs on is transmitted
- Transmitting a and b costs ordering fee + cost of a + cost of b = 10 + 5 + 7 = 22

Joint replenishment problem

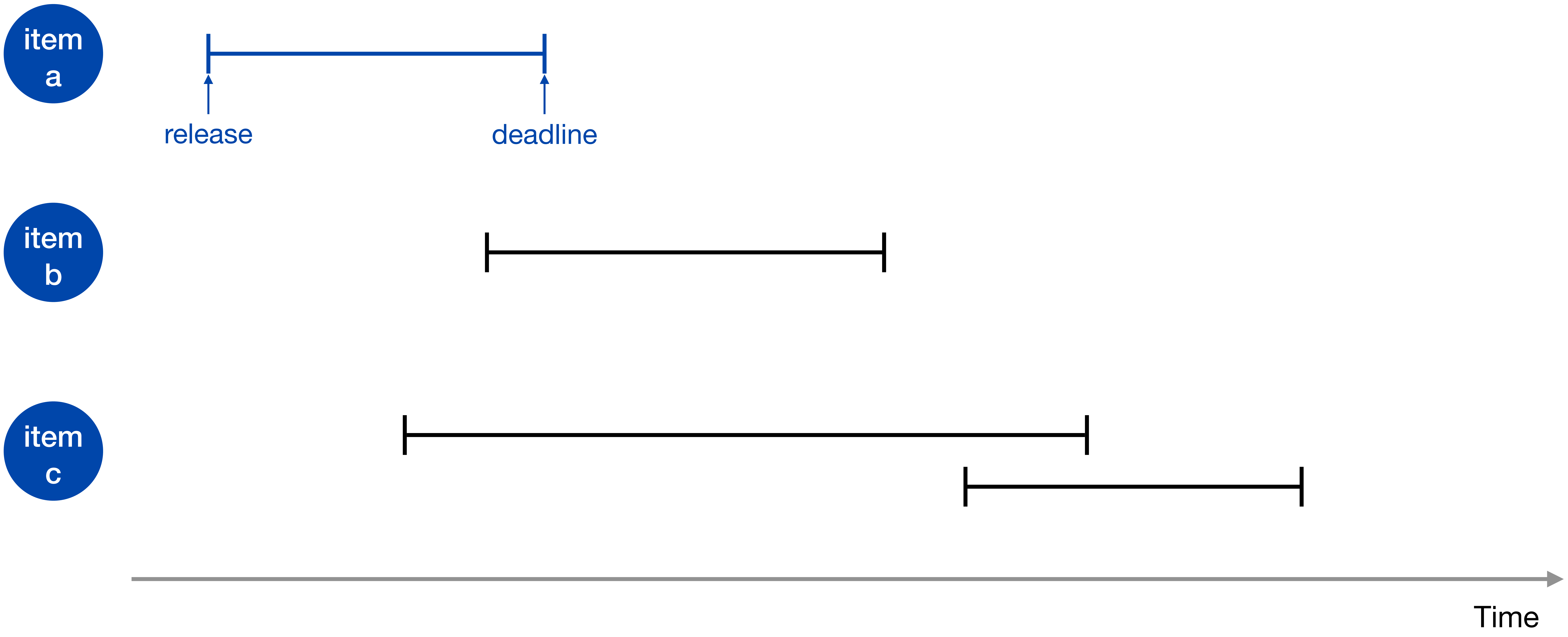


- Each request is on an item
- A transmission is a set of items delivered to the buyer
- A request is served if the item it occurs on is transmitted
- Transmitting a and b costs 10 + 5 + 7 = 22

Problem #1: JRP with deadlines

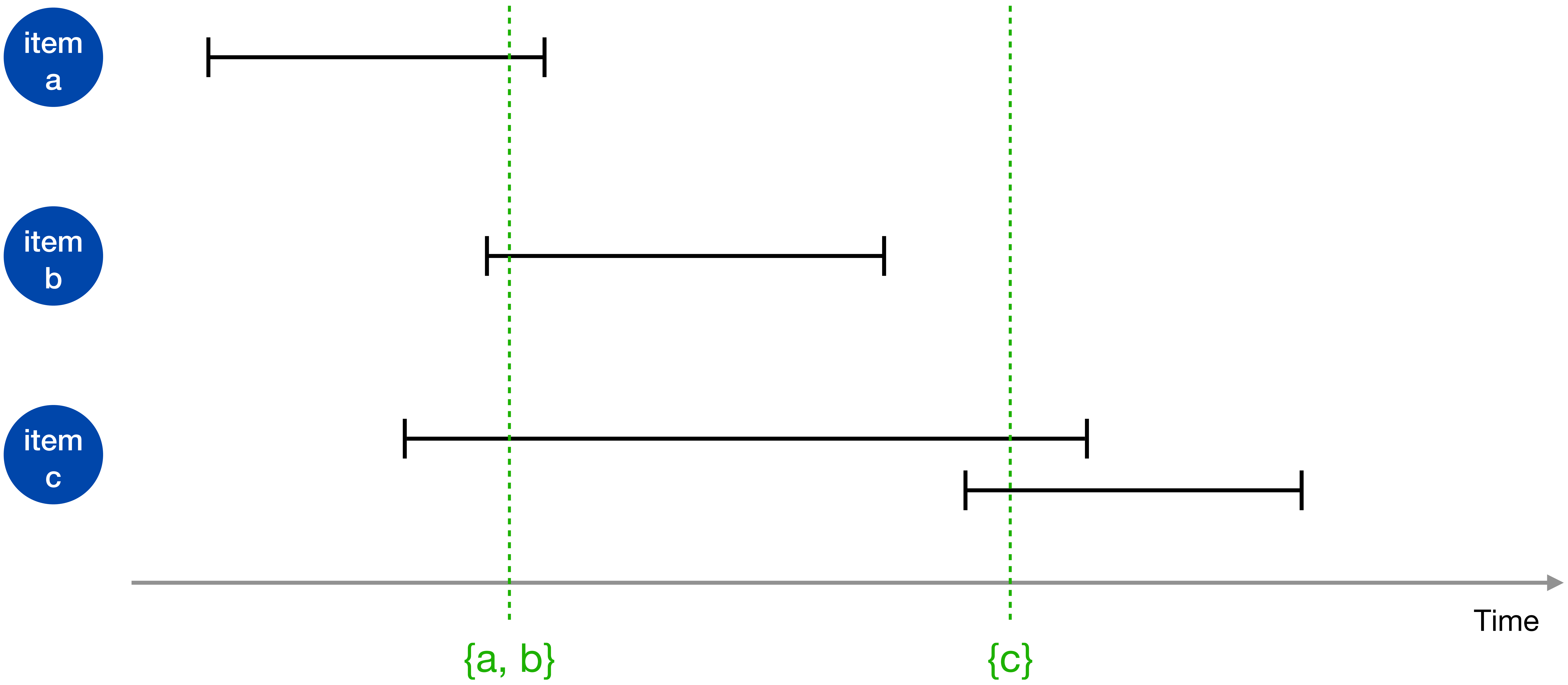


Problem #1: JRP with deadlines



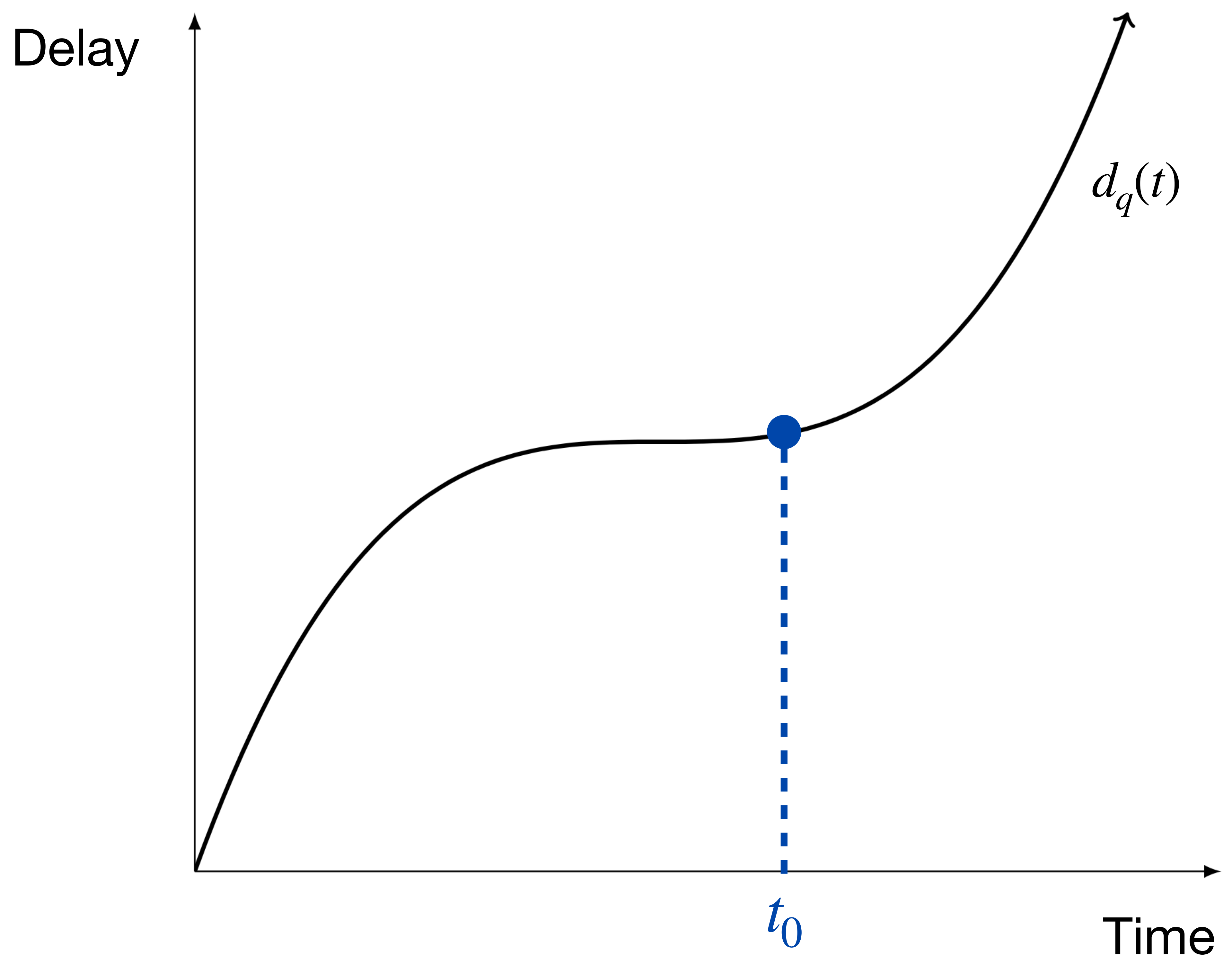
- Each request must be served after it's released and before its deadline

Problem #1: JRP with deadlines



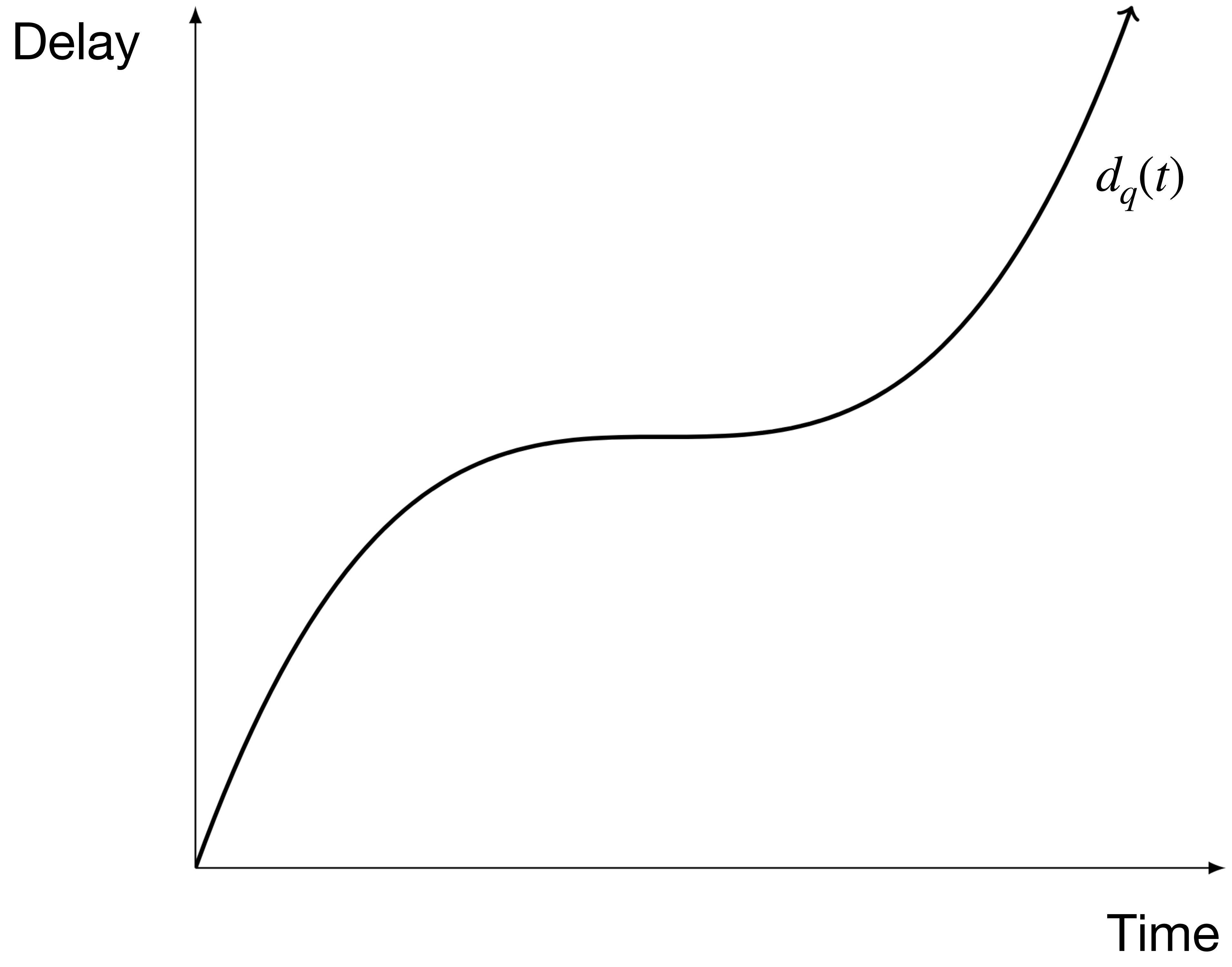
Goal: minimise total cost of all transmissions

Problem #2: JRP with delay



- Each request has a non-decreasing, continuous delay function $d_q(t)$
- We pay $d_q(t_0)$ if q is served at time t_0

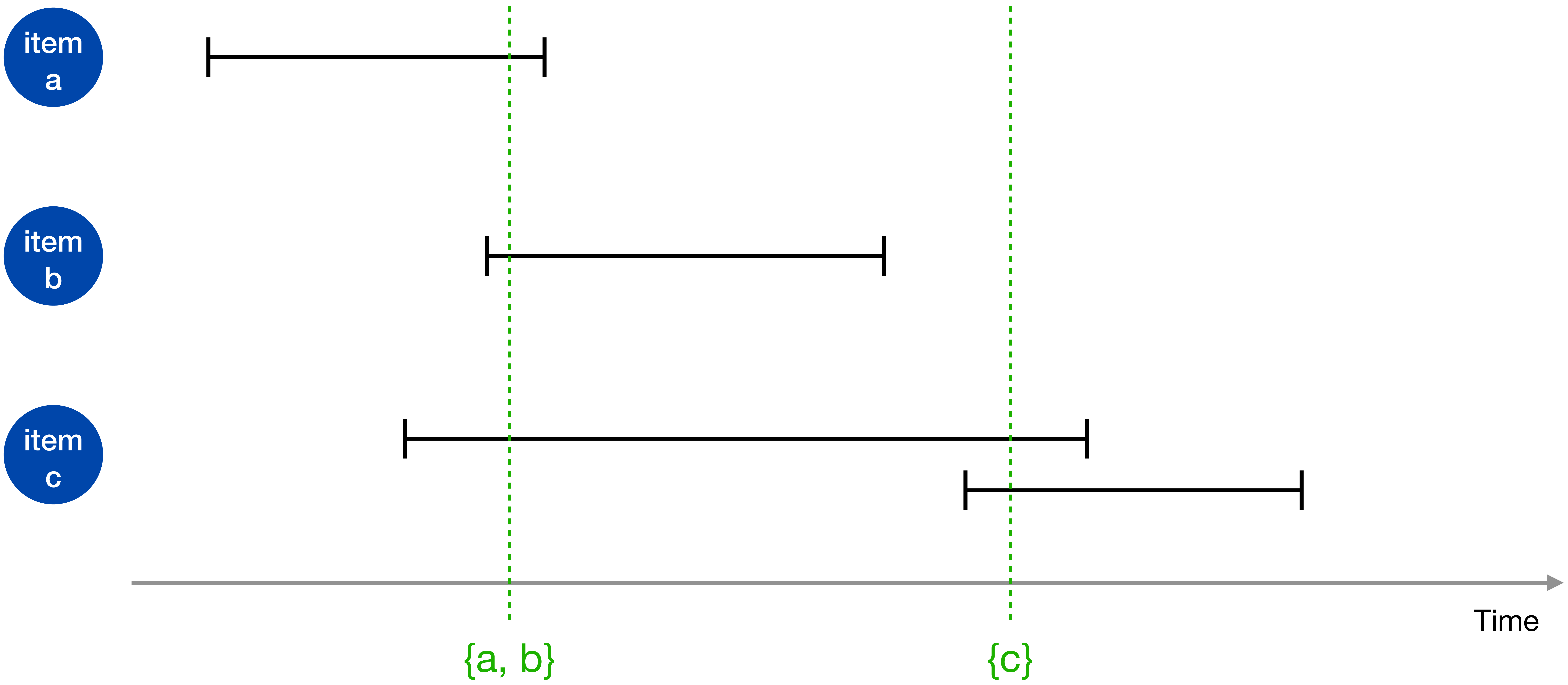
Problem #2: JRP with delay



- Each request has a non-decreasing, continuous delay function $d_q(t)$

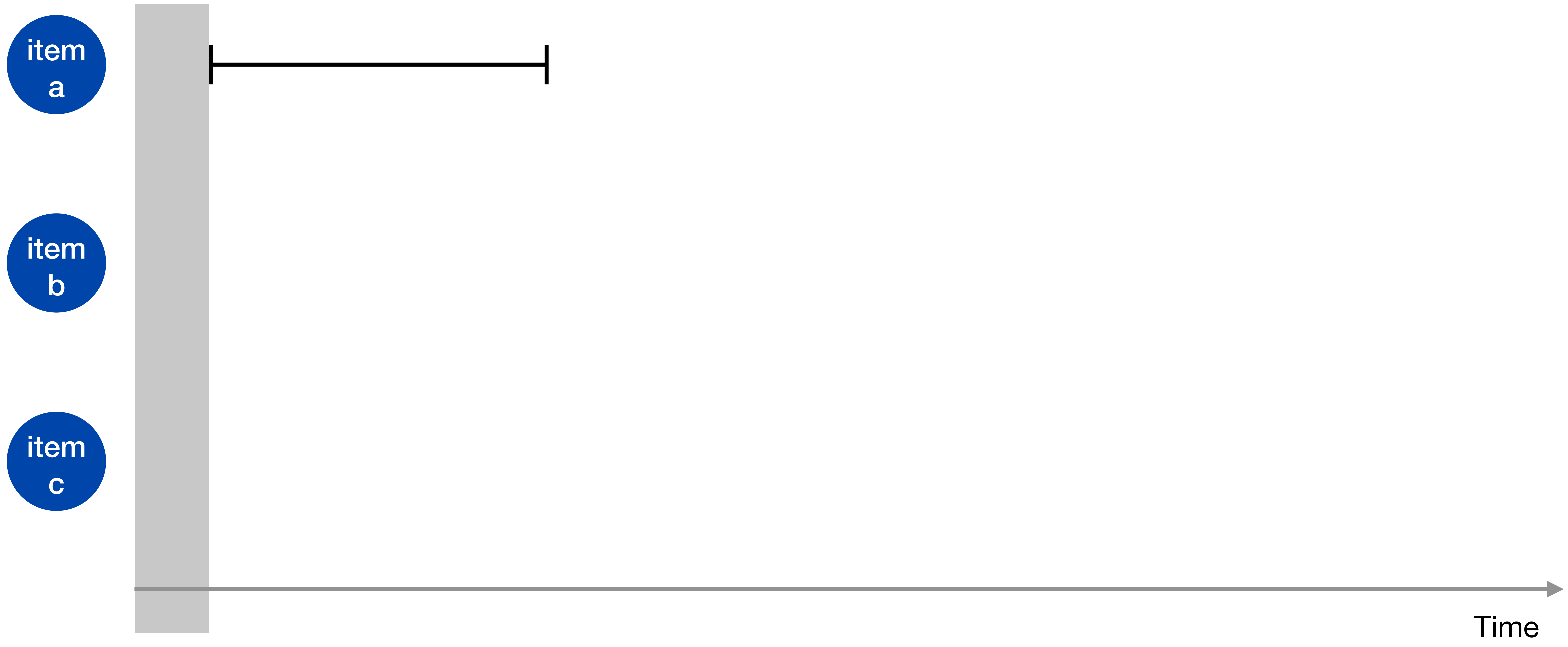
Goal: minimise total cost of all transmissions + delay costs

Offline Setting



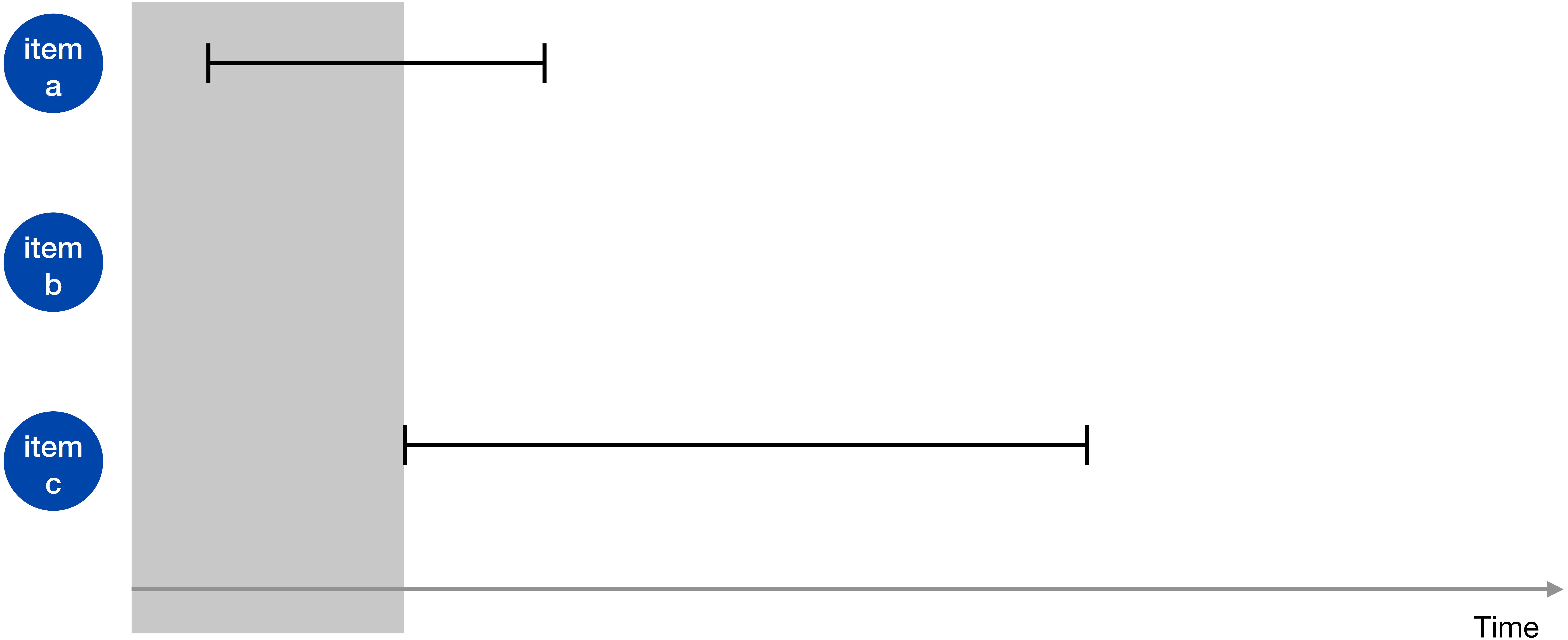
In the offline setting, all requests are known at the beginning

Online Setting

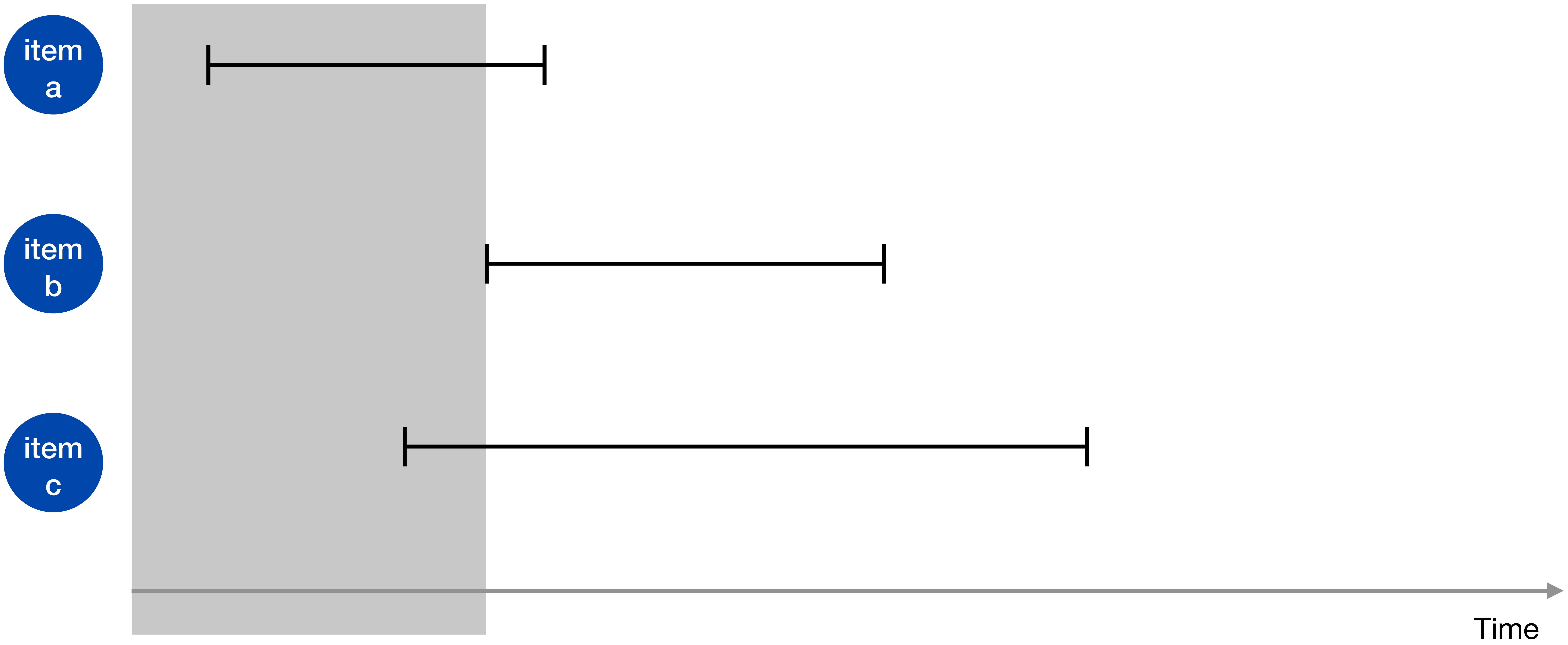


In the online setting, requests are released online

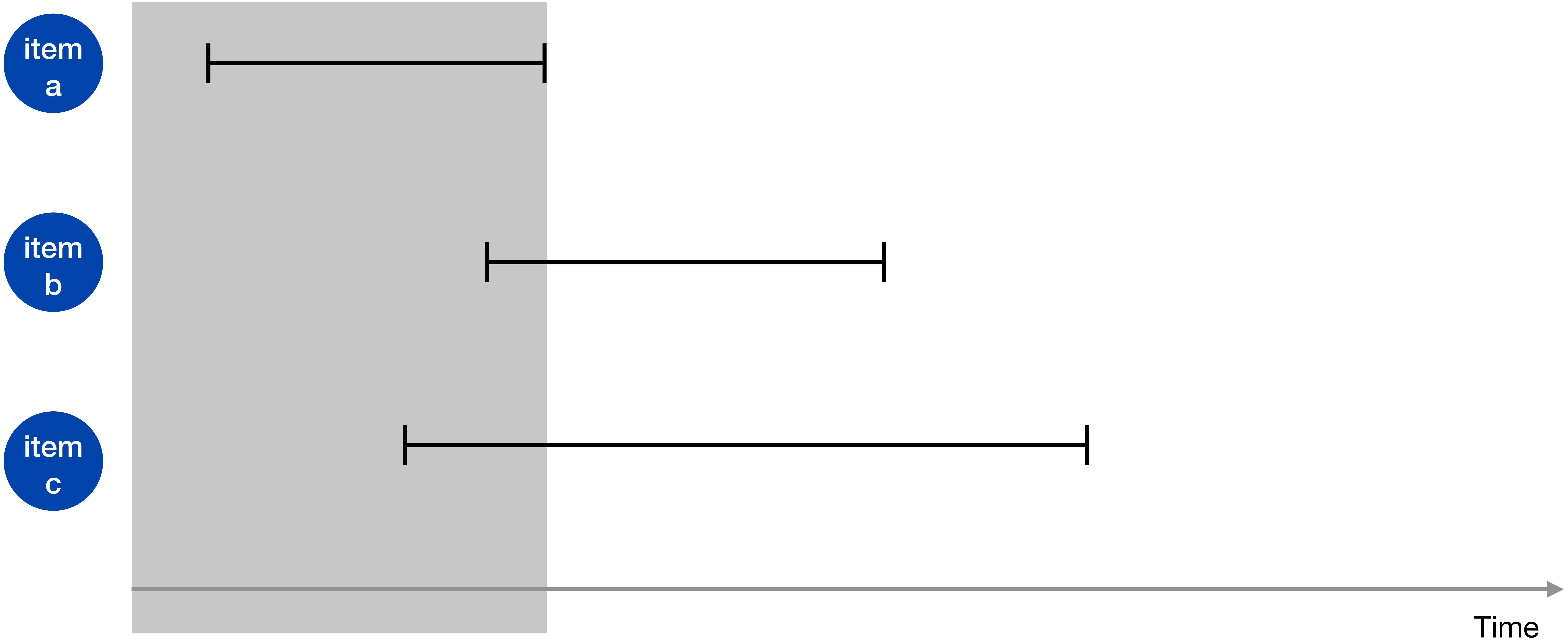
Online Setting



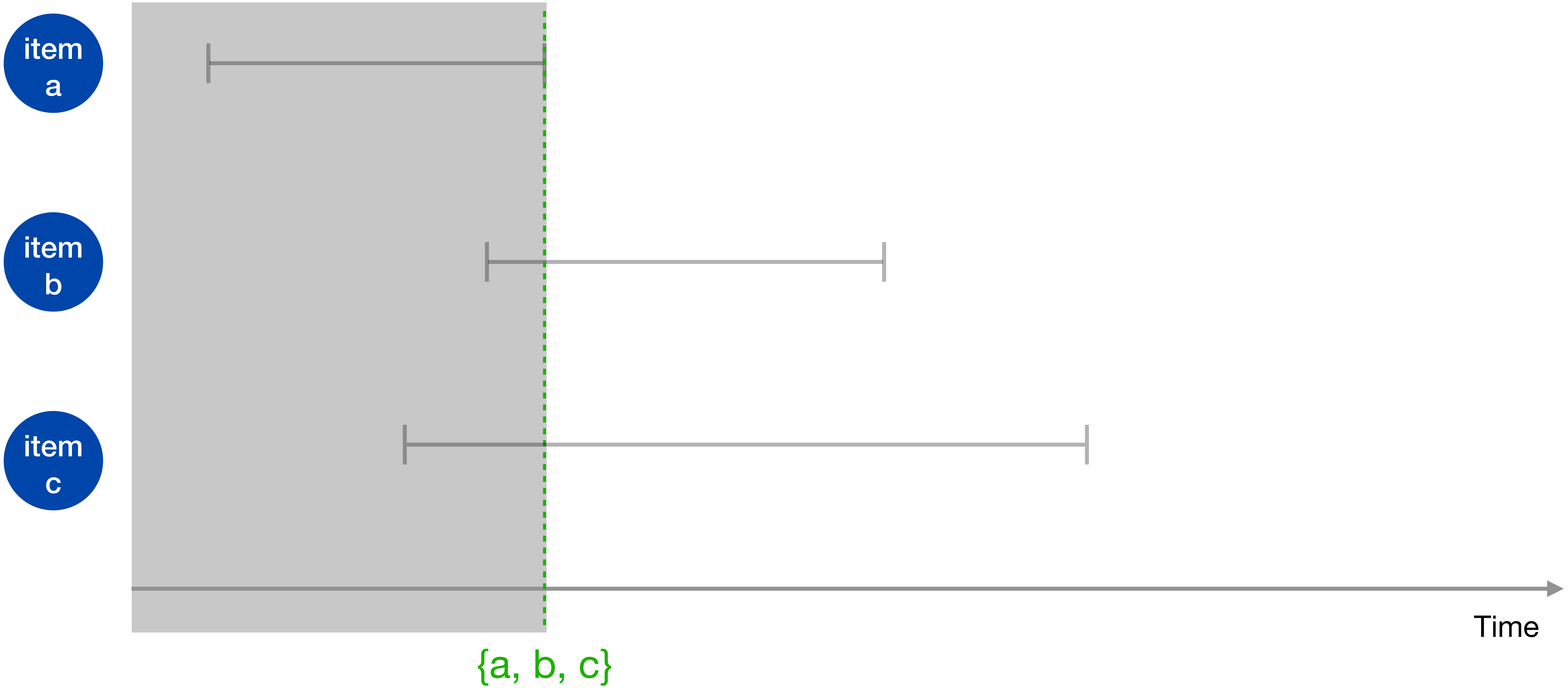
Online Setting



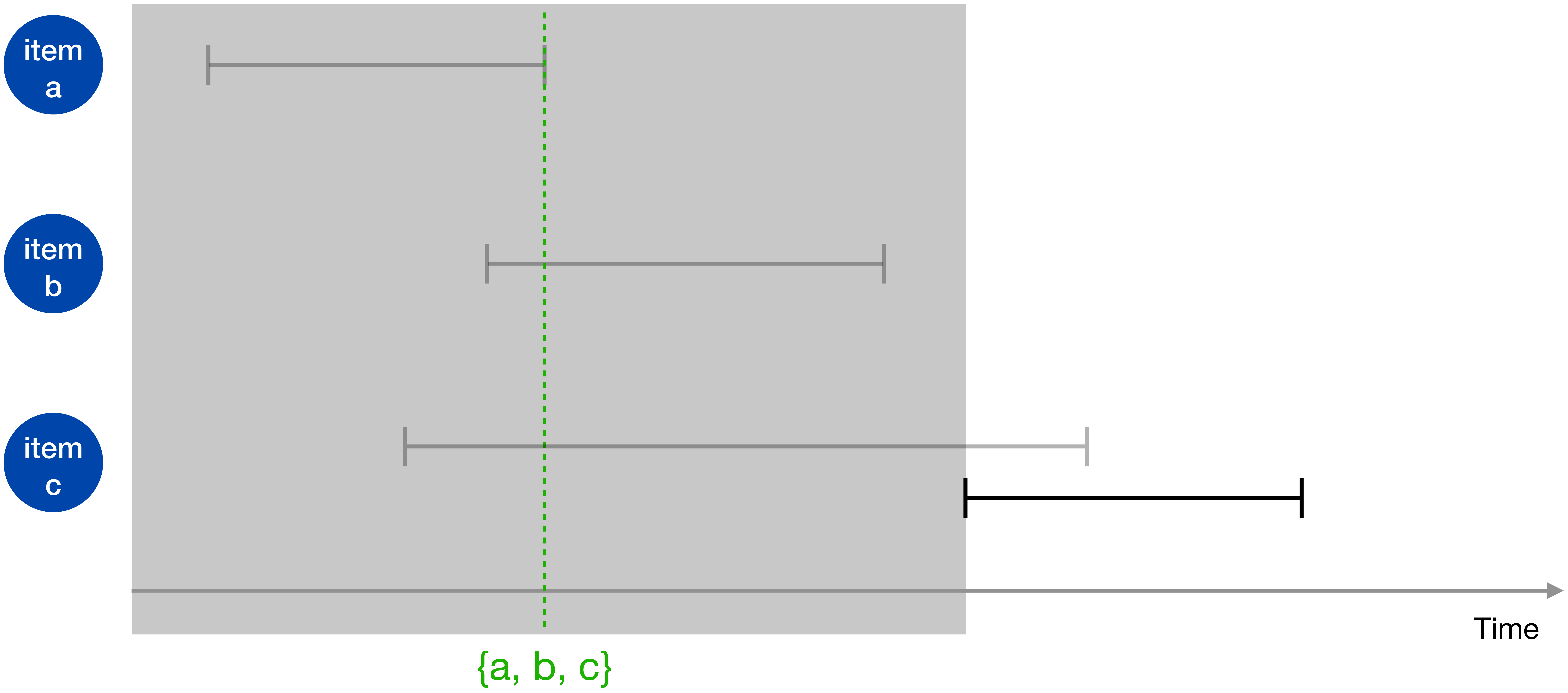
Online Setting



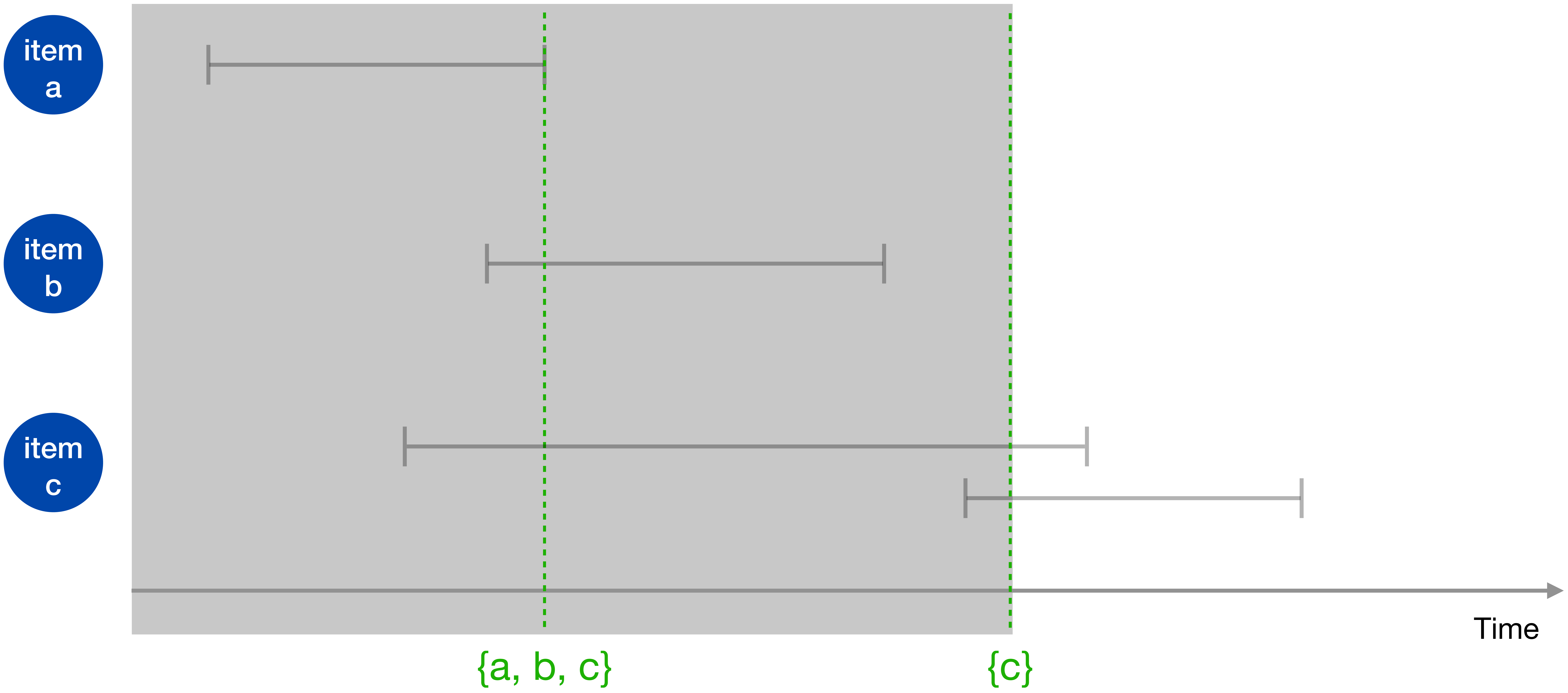
Online Setting



Online Setting



Online Setting



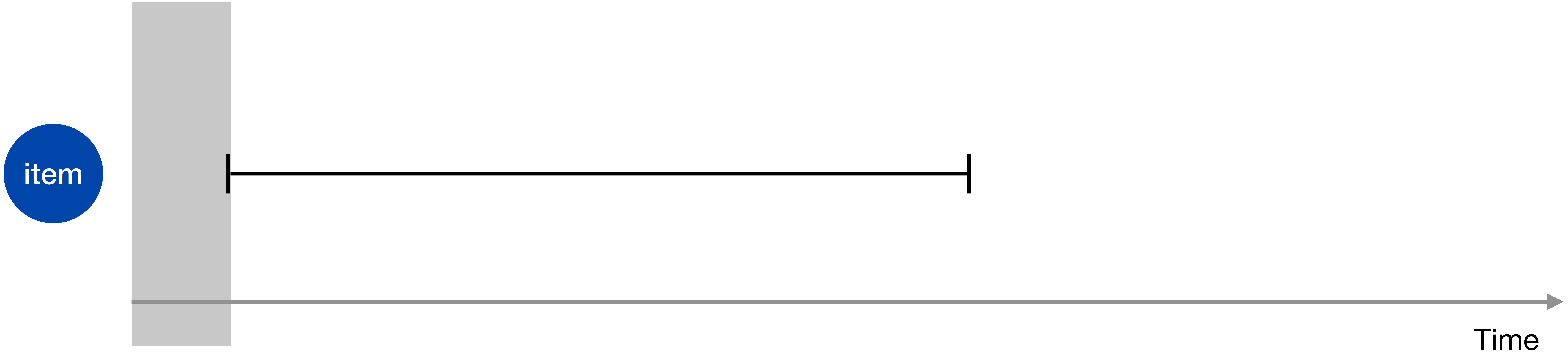
Measuring Performance

- OPT := cost of offline optimal solution
- ALG := cost of online algorithm

$$\textbf{Competitive ratio} := \frac{\textbf{ALG}}{\textbf{OPT}} \textbf{ in the worst case}$$

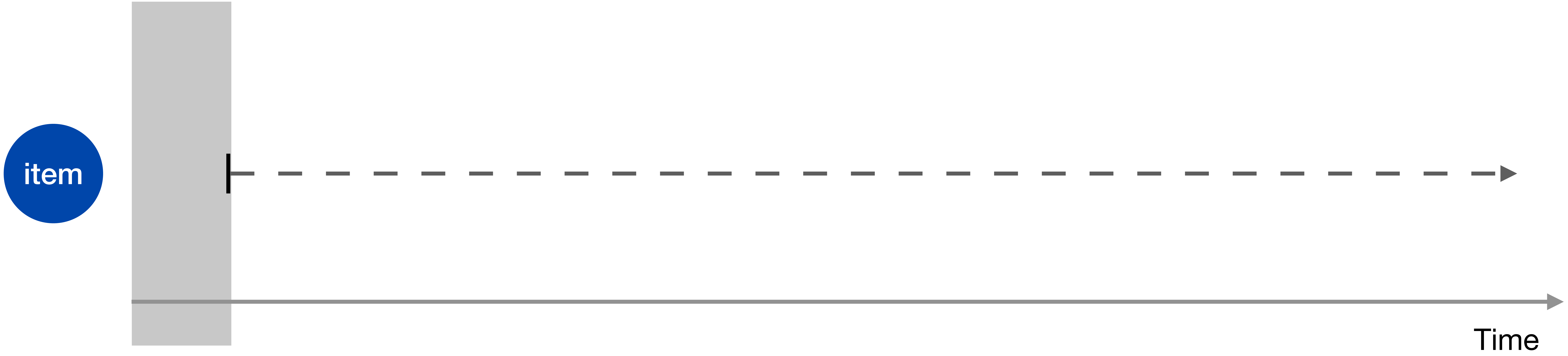
Clairvoyant Online Setting

- When a request arrives, its deadline is revealed to the algorithm



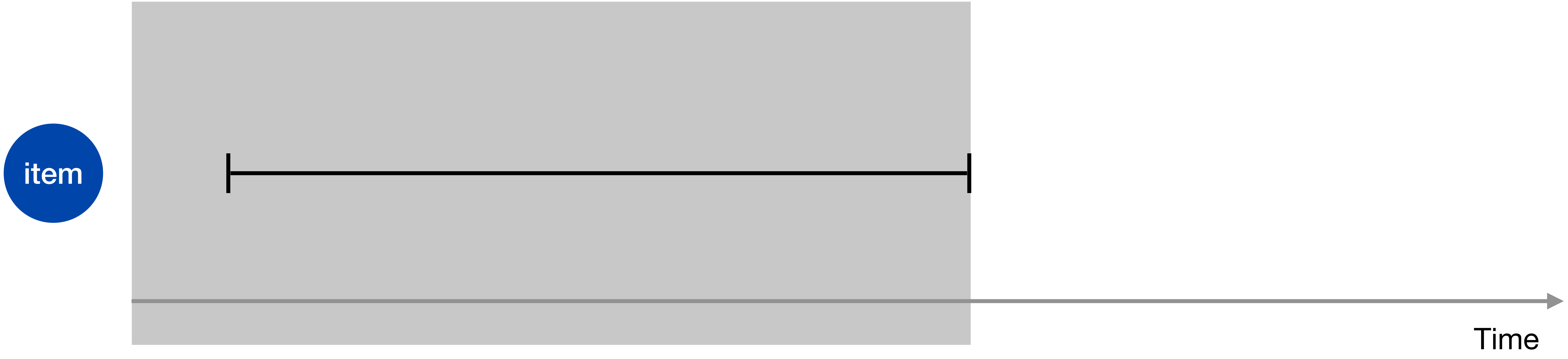
Non-clairvoyant Online Setting

- When a request arrives, its deadline is not known



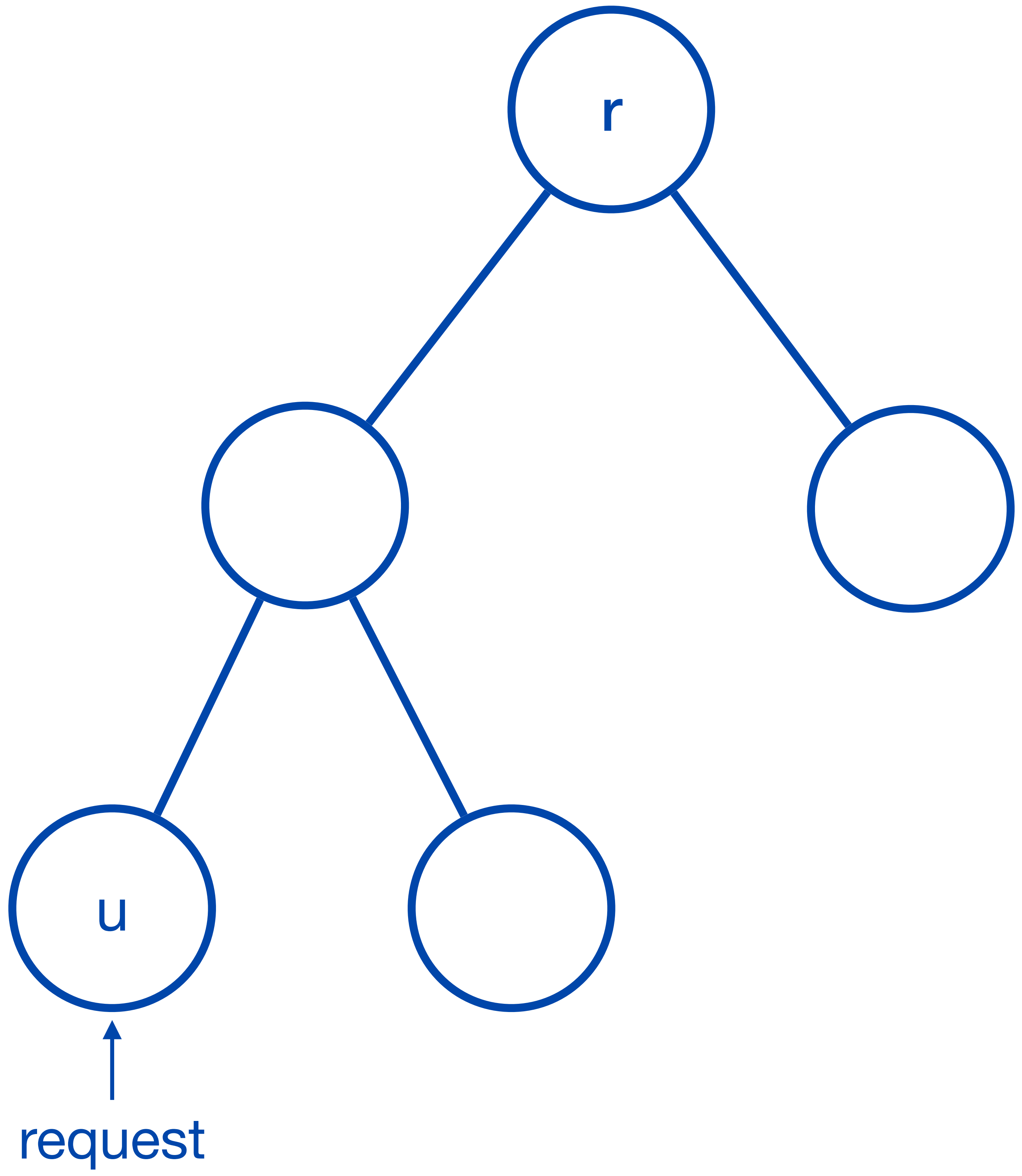
Non-clairvoyant Online Setting

- When a request arrives, its deadline is not known
- When a pending request reaches its deadline, the algorithm is informed and must serve it immediately



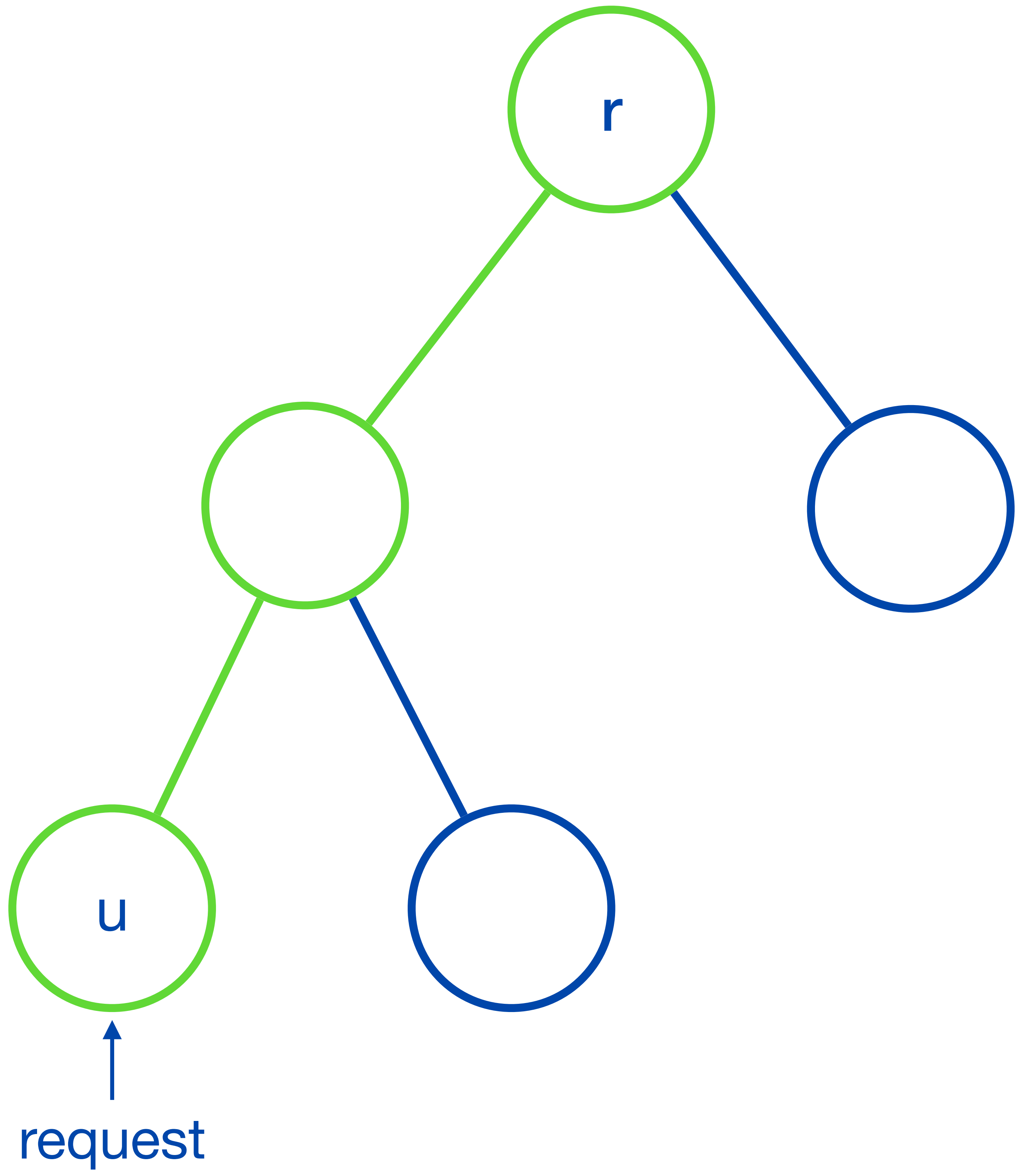
Multi-level aggregation with deadlines (MLAP)

- Given: node-weighted tree
- Each request is a node



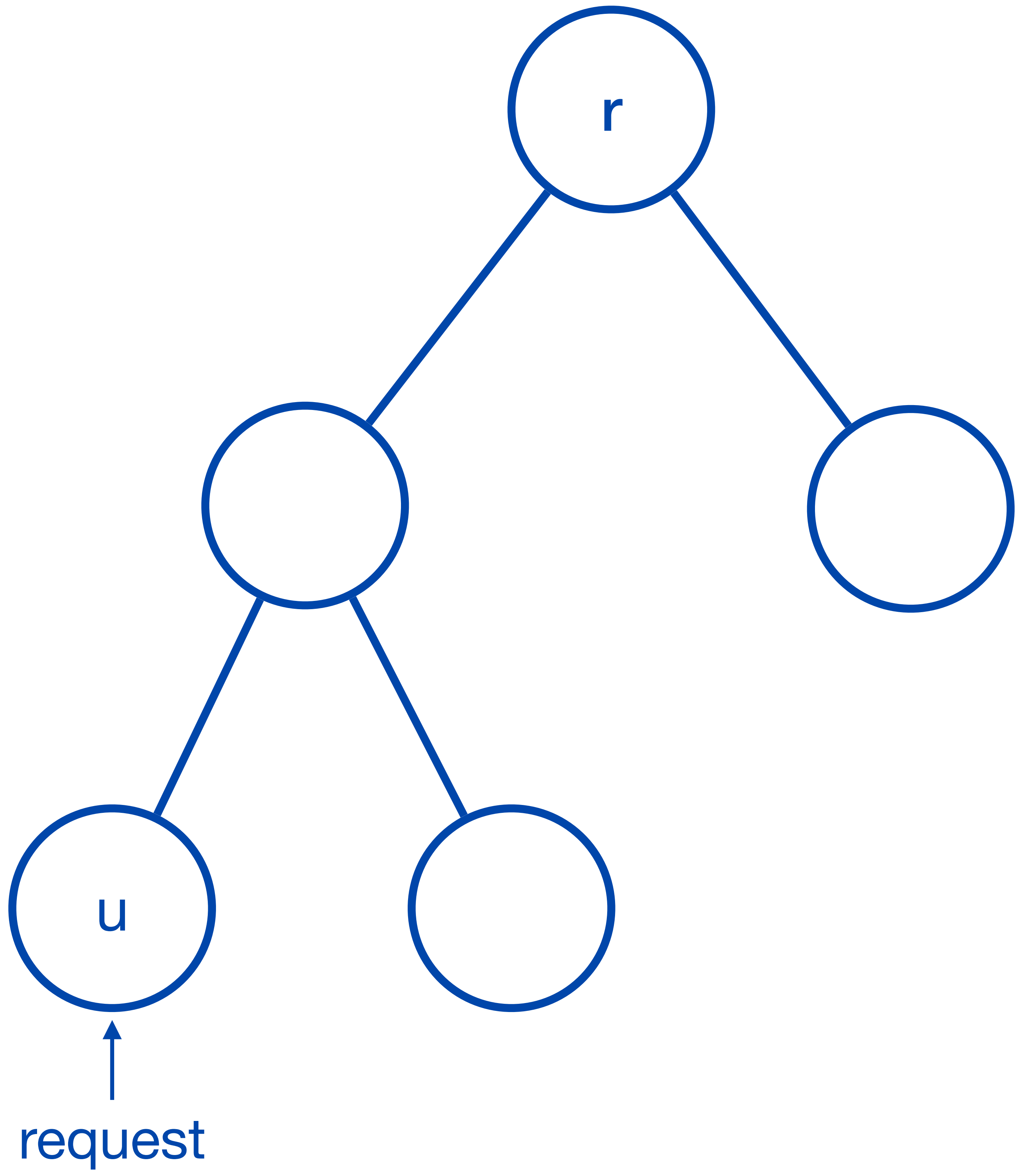
Multi-level aggregation with deadlines

- Given: node-weighted tree
- Each request is a node
- A request is served by transmitting all nodes from the root to the node it occurs on



Multi-level aggregation with deadlines

- Given: node-weighted tree
- Each request is a node
- A request is served by transmitting all nodes from the root to the node it occurs on
- JRP with deadlines is a special case where the tree has depth 1



Previous Results

	Clairvoyant	Non-clairvoyant
JRP with deadlines	2 (Bienkowski et al., 2014)	
JRP with delay	3 (Buchbinder et al., 2013)	
MLAP with deadlines	O(depth) (Buchbinder et al., 2017)	

Previous Results

	Clairvoyant	Non-clairvoyant
JRP with deadlines	2 (Bienkowski et al., 2014)	
JRP with delay	3 (Buchbinder et al., 2013)	
MLAP with deadlines	O(depth) (Buchbinder et al., 2017)	
Set cover with delay	O(log N) (Carrasco et al., 2018)	randomised O(log n log k) (Azar et al., 2020)

Our Results

	Clairvoyant	**Non-clairvoyant**
JRP with deadlines	2 (Bienkowski et al., 2014)	lower bound: $\Omega(\sqrt{n})$
JRP with delay	3 (Buchbinder et al., 2013)	
MLAP with deadlines	O(depth) (Buchbinder et al., 2017)	
Set cover with delay	O(log N) (Carrasco et al., 2018)	randomised O(log n log k) (Azar et al., 2020)

Our Results

	Clairvoyant	**Non-clairvoyant**
JRP with deadlines	2 (Bienkowski et al., 2014)	lower bound: $\Omega(\sqrt{n})$ upper bound: $O(\sqrt{n})$
JRP with delay	3 (Buchbinder et al., 2013)	$O(\sqrt{n})$
MLAP with deadlines	O(depth) (Buchbinder et al., 2017)	$O(\sqrt{n} + \text{depth})$
Set cover with delay	O(log N) (Carrasco et al., 2018)	randomised O(log n log k) (Azar et al., 2020)

Our Results

	Clairvoyant	**Non-clairvoyant**
JRP with deadlines	2 (Bienkowski et al., 2014)	lower bound: $\Omega(\sqrt{n})$ upper bound: $O(\sqrt{n})$
JRP with delay	3 (Buchbinder et al., 2013)	$O(\sqrt{n})$
MLAP with deadlines	O(depth) (Buchbinder et al., 2017)	$O(\sqrt{n} + \text{depth})$
Set cover with delay	O(log N) (Carrasco et al., 2018)	deterministic O(log n log k) not discussed

Lower bound

JRP with deadlines

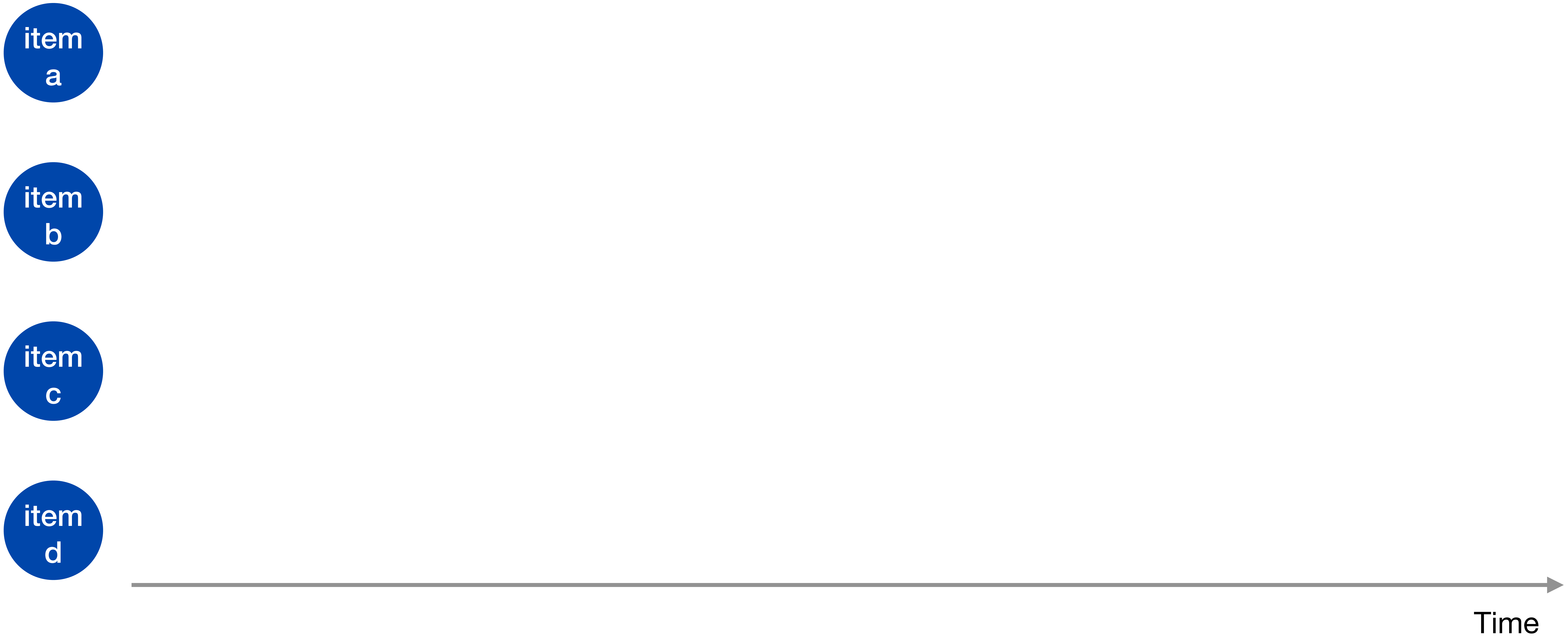
- *Theorem:* Any non-clairvoyant algorithm for JRP with deadlines is $\Omega(\sqrt{n})$ - competitive.

Lower bound

JRP with deadlines

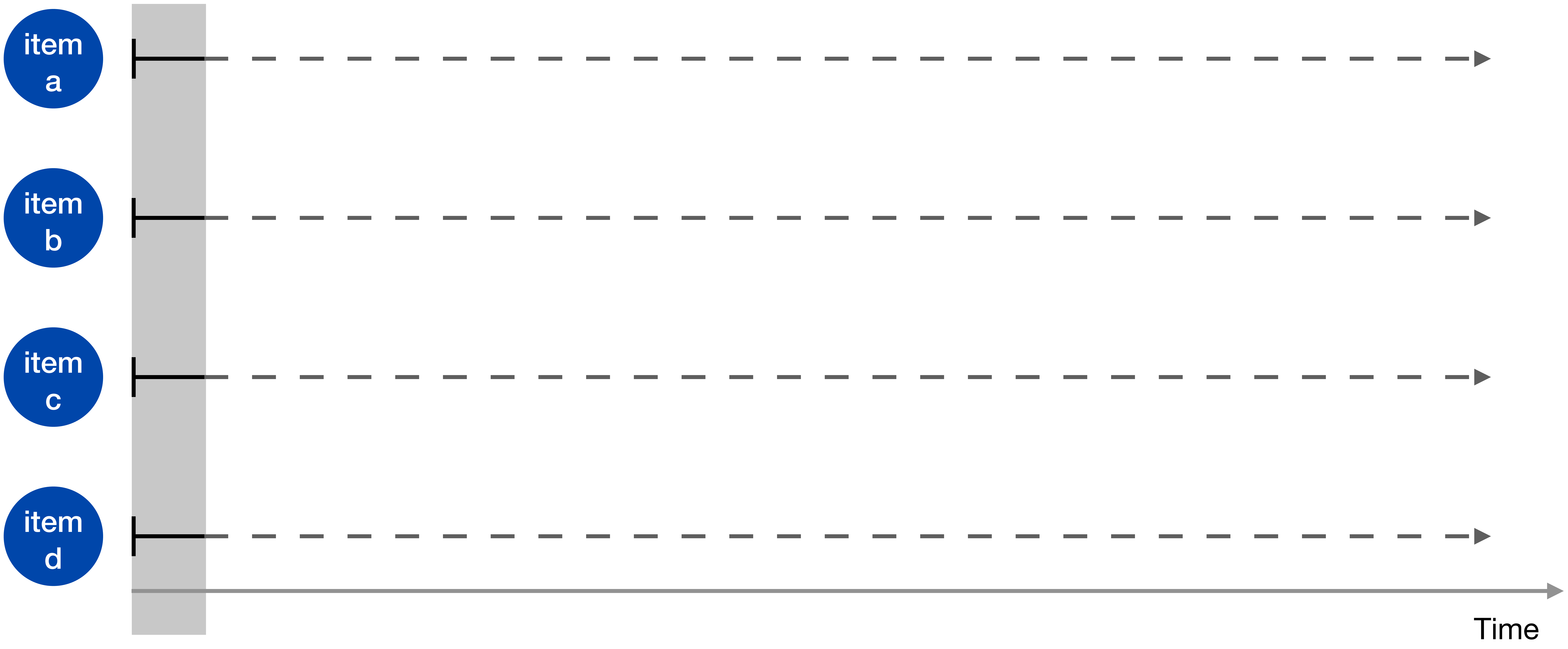
- n items
- each item costs 1
- ordering fee = $\sqrt{n}$

Lower bound



- Ordering fee = 2

Lower bound

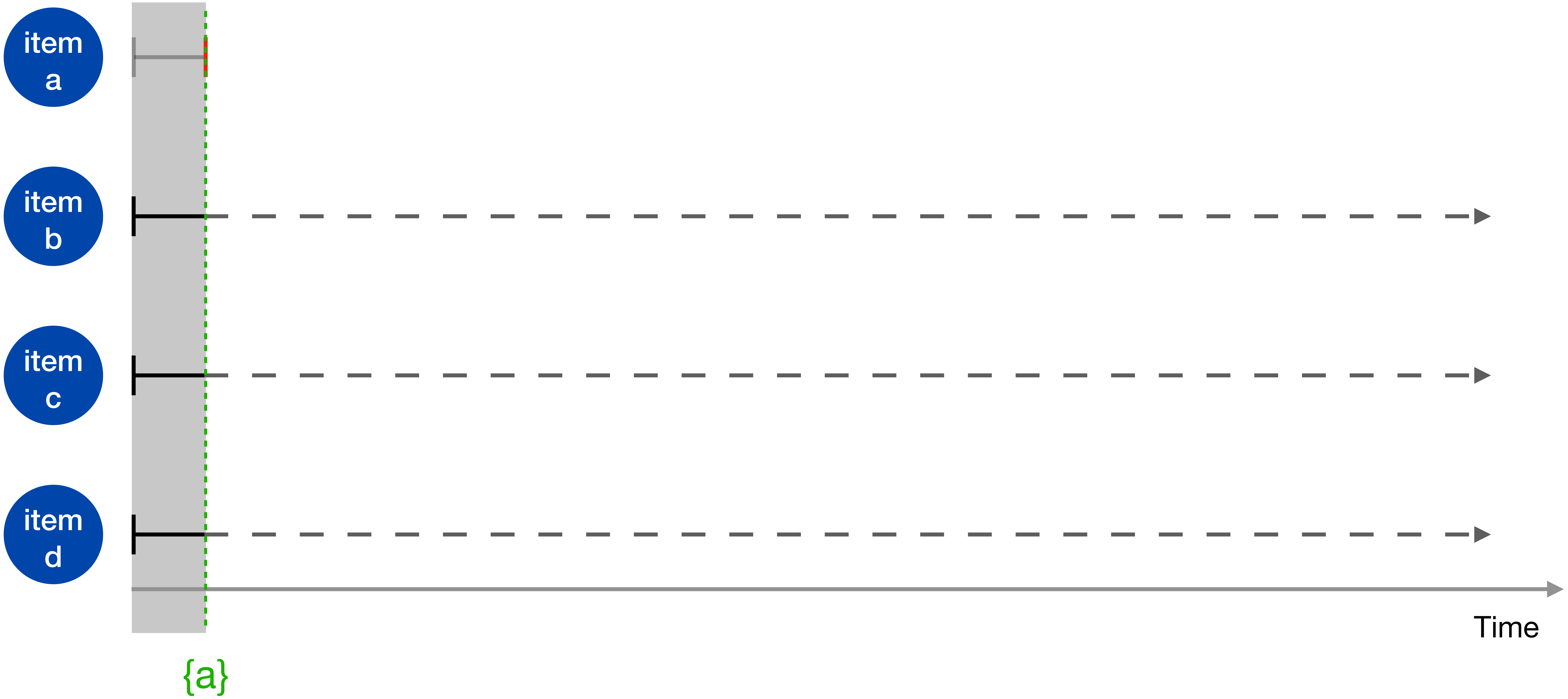


- Release requests on all items

Lower bound

- Pick a pending request and set its deadline

Lower bound



- Pick a pending request and set its deadline

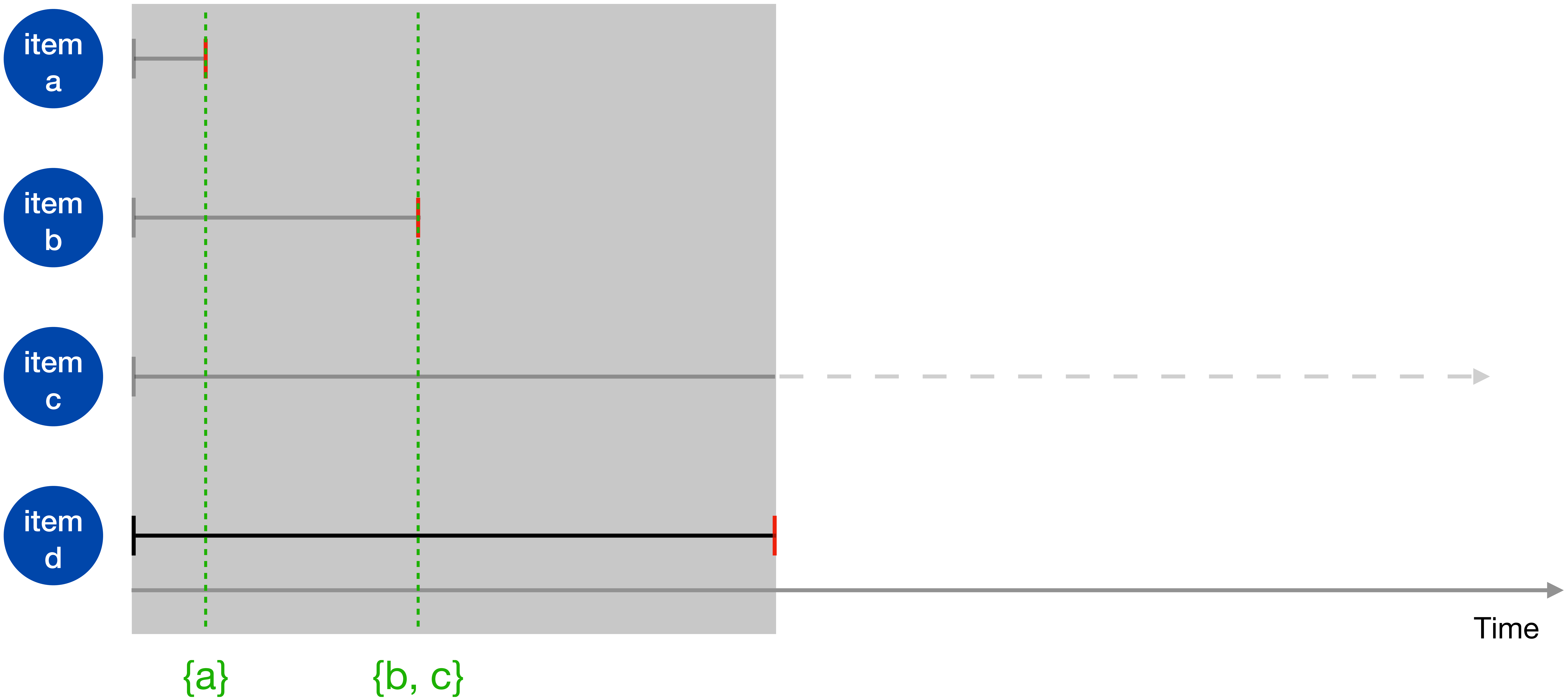
Lower bound

- As long as there remains pending requests, pick one and set its deadline

Lower bound

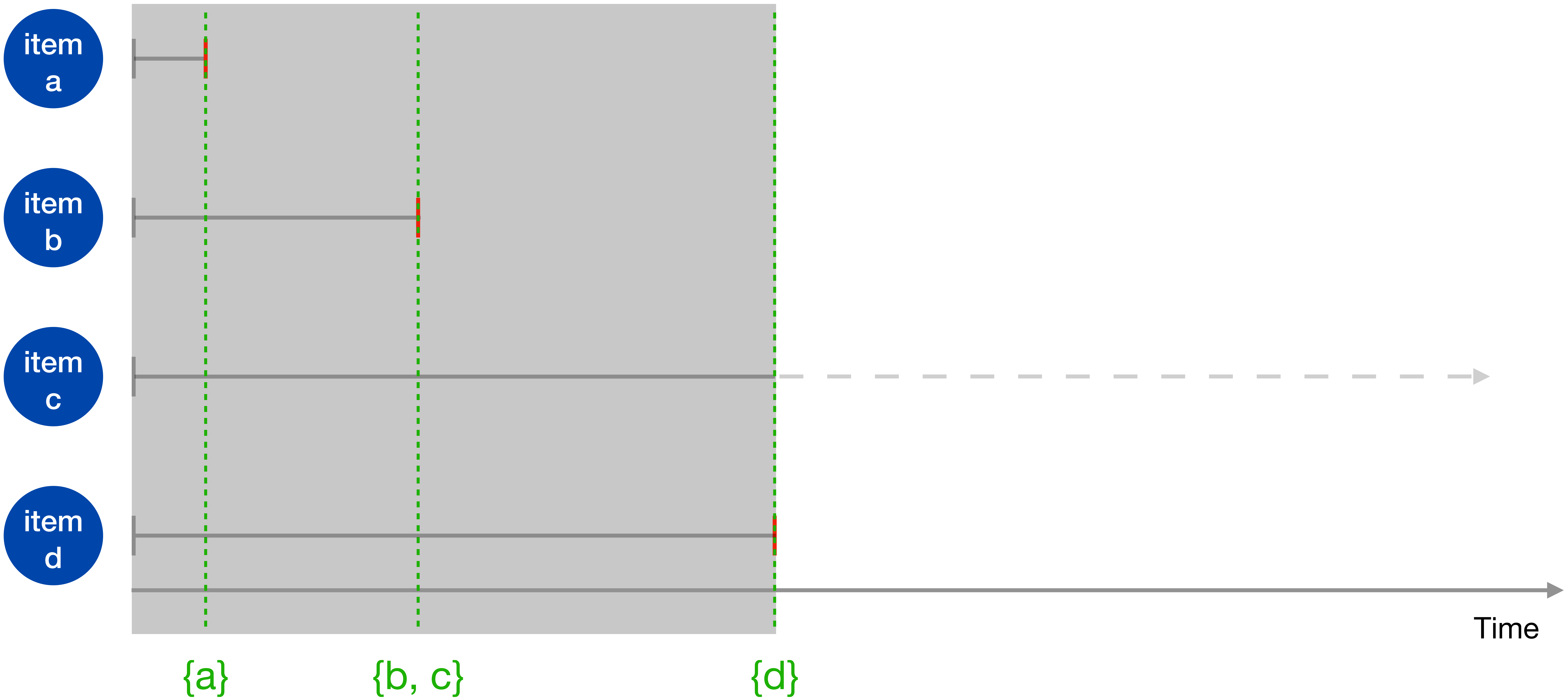
- As long as there remains pending requests, pick one and set its deadline

Lower bound



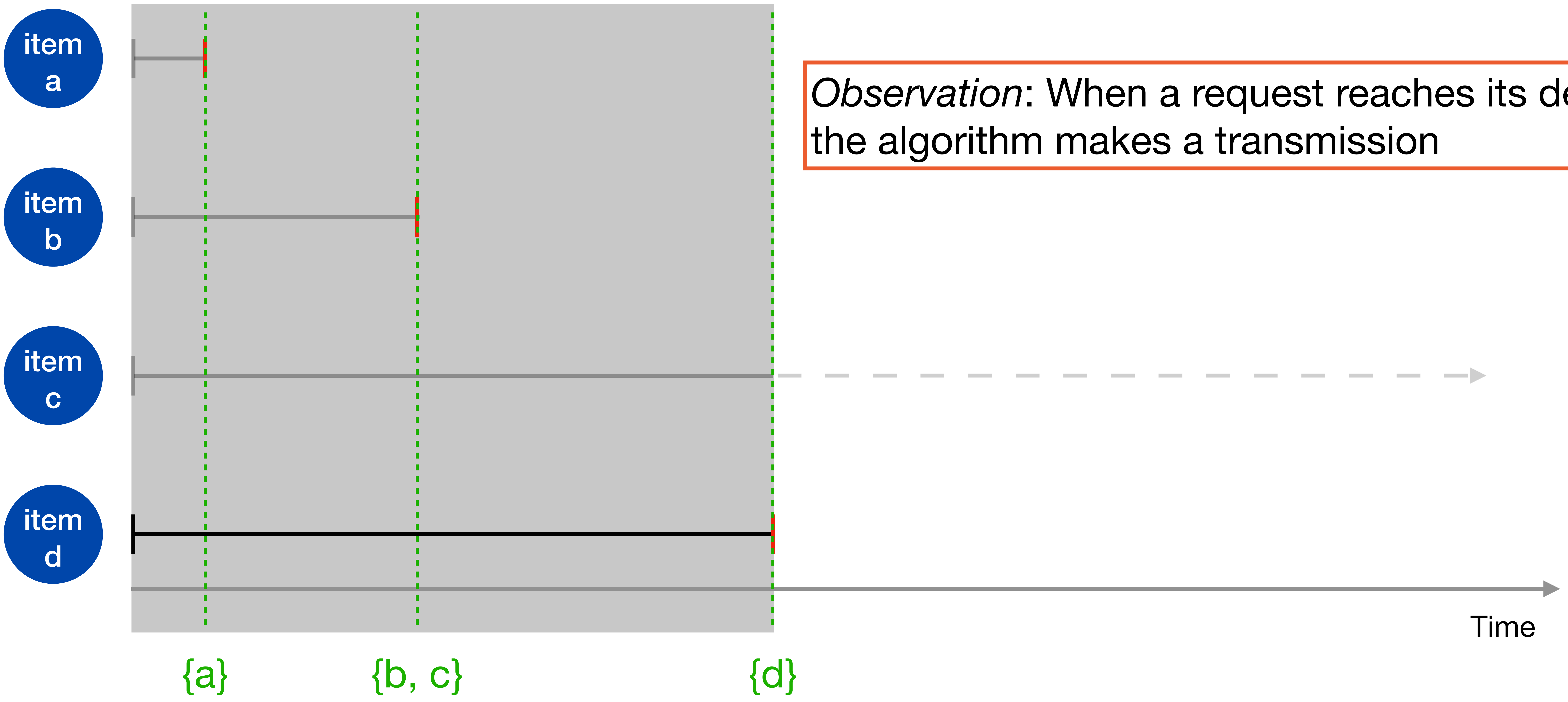
- As long as there remains pending requests, pick one and set its deadline

Lower bound



- As long as there remains pending requests, pick one and set its deadline

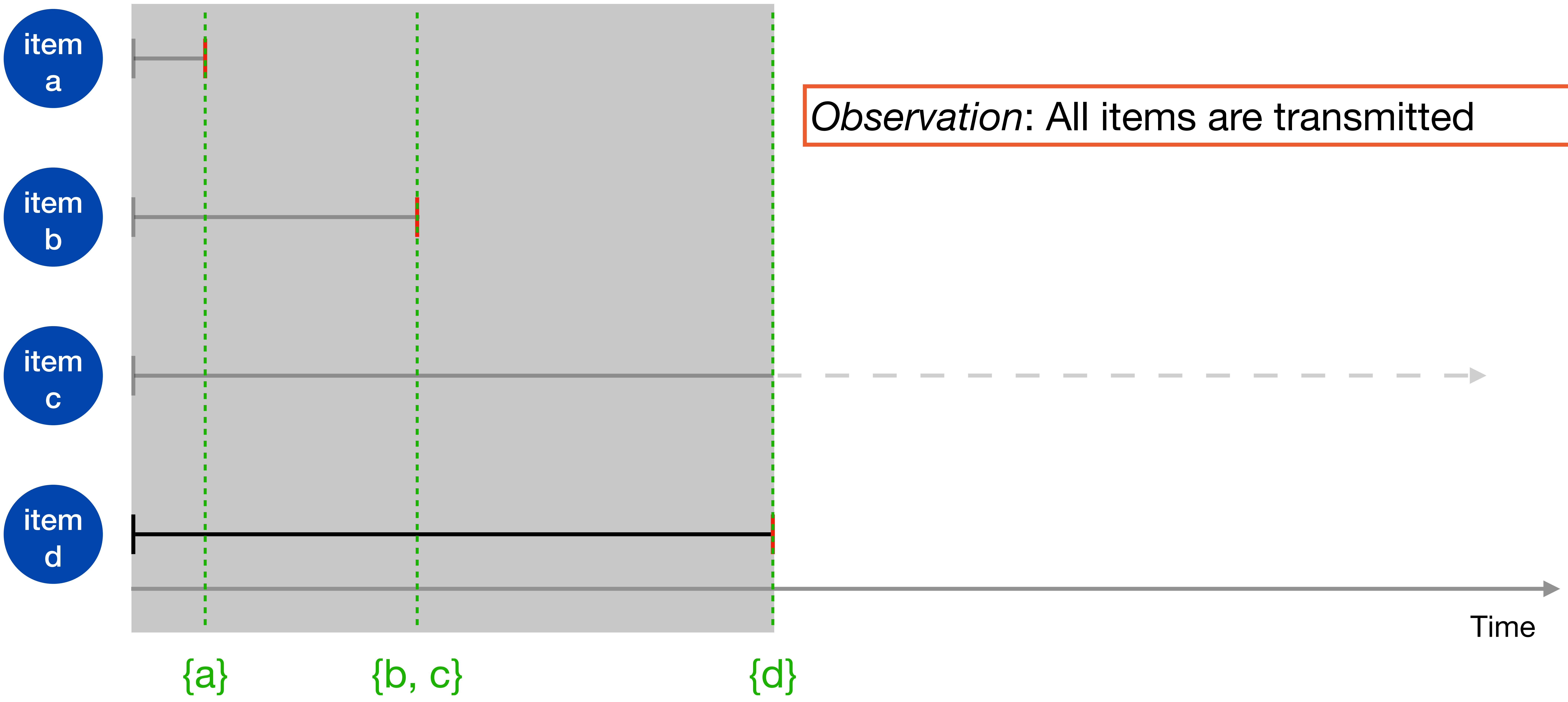
Lower bound



Observation: When a request reaches its deadline, the algorithm makes a transmission

- ALG = (#requests with finite deadlines) x $\sqrt{n}$

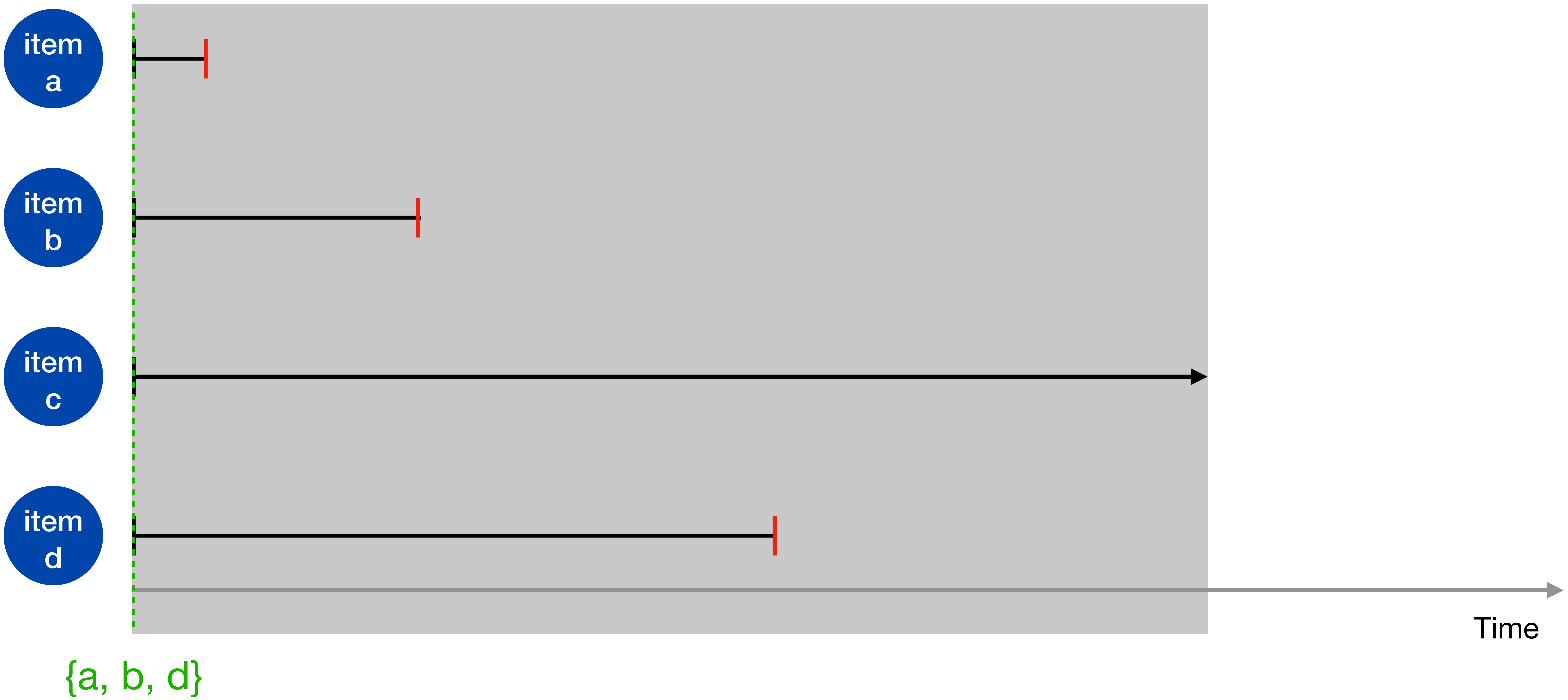
Lower bound



Observation: All items are transmitted

- ALG = (#requests with finite deadlines) x $\sqrt{n} + n$

Lower bound - Offline optimal



- OPT = $\sqrt{n}$ + (#requests with finite deadlines)

Lower bound

JRP with deadlines

- ALG = (#requests with finite deadlines) x $\sqrt{n} + n$
- OPT = $\sqrt{n}$ + (#requests with finite deadlines)

Lower bound

JRP with deadlines

- ALG = (#requests with finite deadlines) x $\sqrt{n} + n$
- OPT = $\sqrt{n}$ + (#requests with finite deadlines)

$$\text{Competitive ratio} = \frac{(\#\text{requests with finite deadlines}) \times \sqrt{n} + n}{(\#\text{requests with finite deadlines}) + \sqrt{n}}$$

Lower bound

JRP with deadlines

- ALG = (#requests with finite deadlines) x $\sqrt{n} + n$
- OPT = $\sqrt{n}$ + (#requests with finite deadlines)

$$\text{Competitive ratio} = \frac{(\#\text{requests with finite deadlines}) \times \sqrt{n} + n}{(\#\text{requests with finite deadlines}) + \sqrt{n}}$$

$$= \sqrt{n}\,\frac{(\#\text{requests with finite deadlines}) + \sqrt{n}}{(\#\text{requests with finite deadlines}) + \sqrt{n}} = \sqrt{n}$$

Insights

JRP with deadlines

- Algorithm does not know requests’ deadlines
    - serving requests in order of increasing deadlines is not possible

Insights

JRP with deadlines

- Algorithm does not know requests' deadlines
    - serving requests in order of increasing deadlines is not possible

- Prioritise requests based on the cost of the items they are on

Algorithm

JRP with deadlines

- *Theorem:* There exists a $O(\sqrt{n})$ - competitive non-clairvoyant algorithm for JRP with deadlines

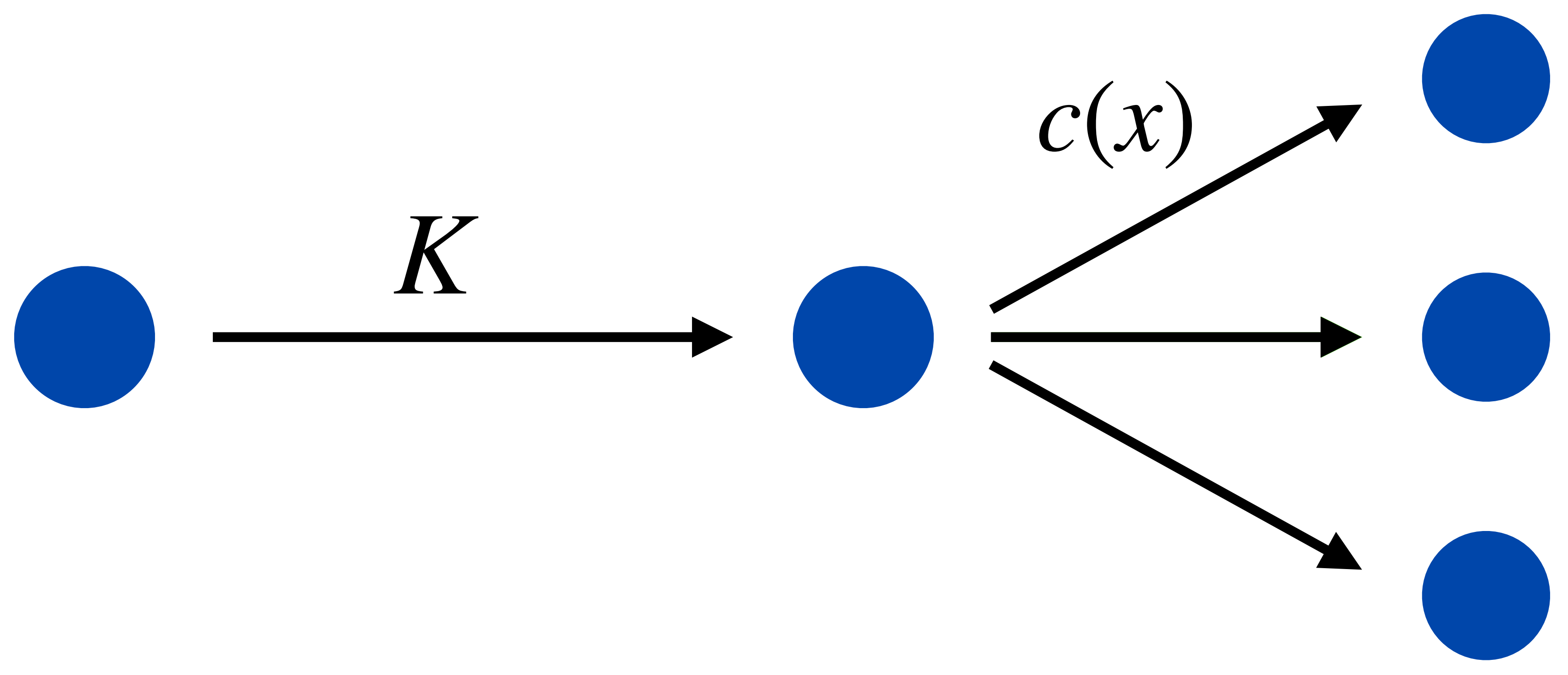
Algorithm

JRP with deadlines

- *Definition:*

$$x \text{ is } \text{cheap} \text{ if } c(x) \leq \frac{K}{\sqrt{n}}$$

$$x \text{ is } \text{expensive} \text{ if } c(x) > \frac{K}{\sqrt{n}}$$



K := ordering fee, $c(x)$:= cost of item x

Algorithm

JRP with deadlines

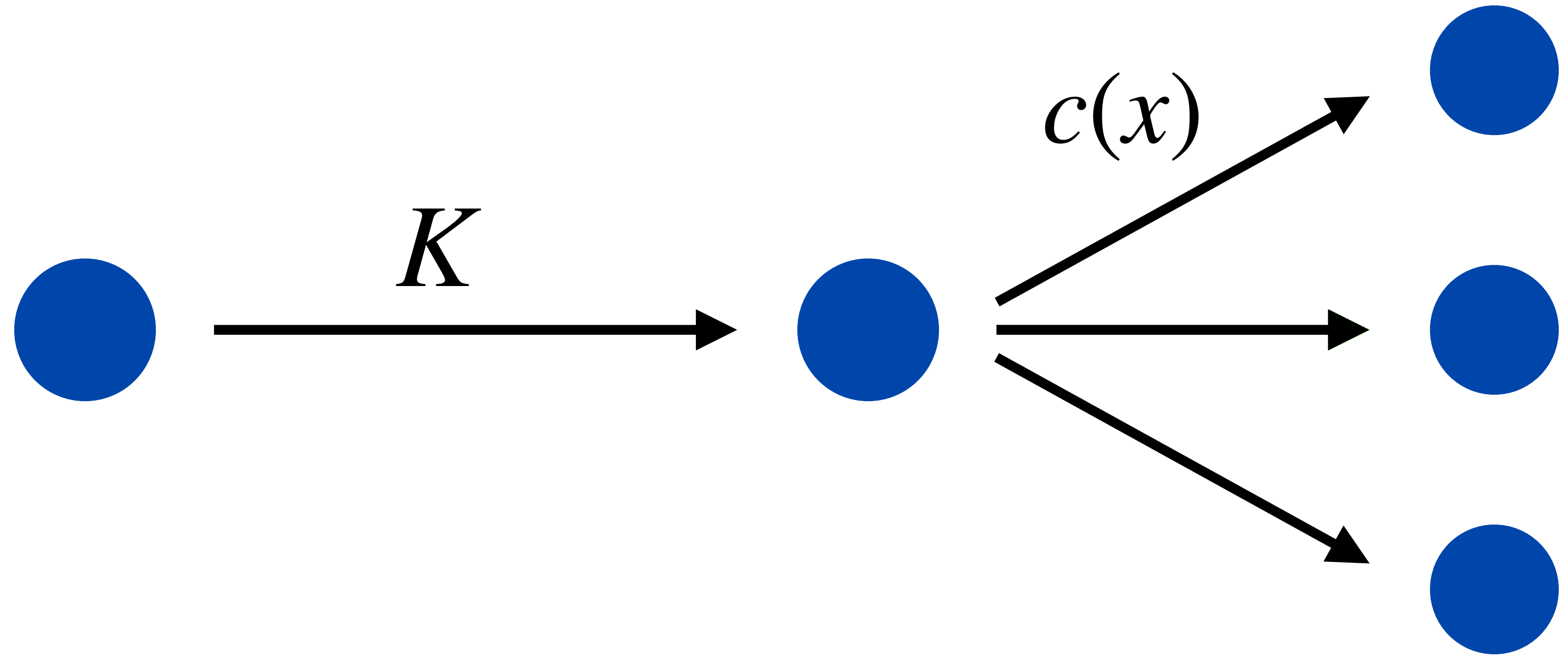
- *Definition:*

$$x \text{ is } \textcolor{orange}{\text{cheap}} \text{ if } c(x) \le \frac{K}{\sqrt{n}}$$

a request is cheap if it’s on a cheap item

$$x \text{ is } \textcolor{orange}{\text{expensive}} \text{ if } c(x) > \frac{K}{\sqrt{n}}$$

a request is expensive if it’s on an expensive item



K := ordering fee, $c(x)$:= cost of item x

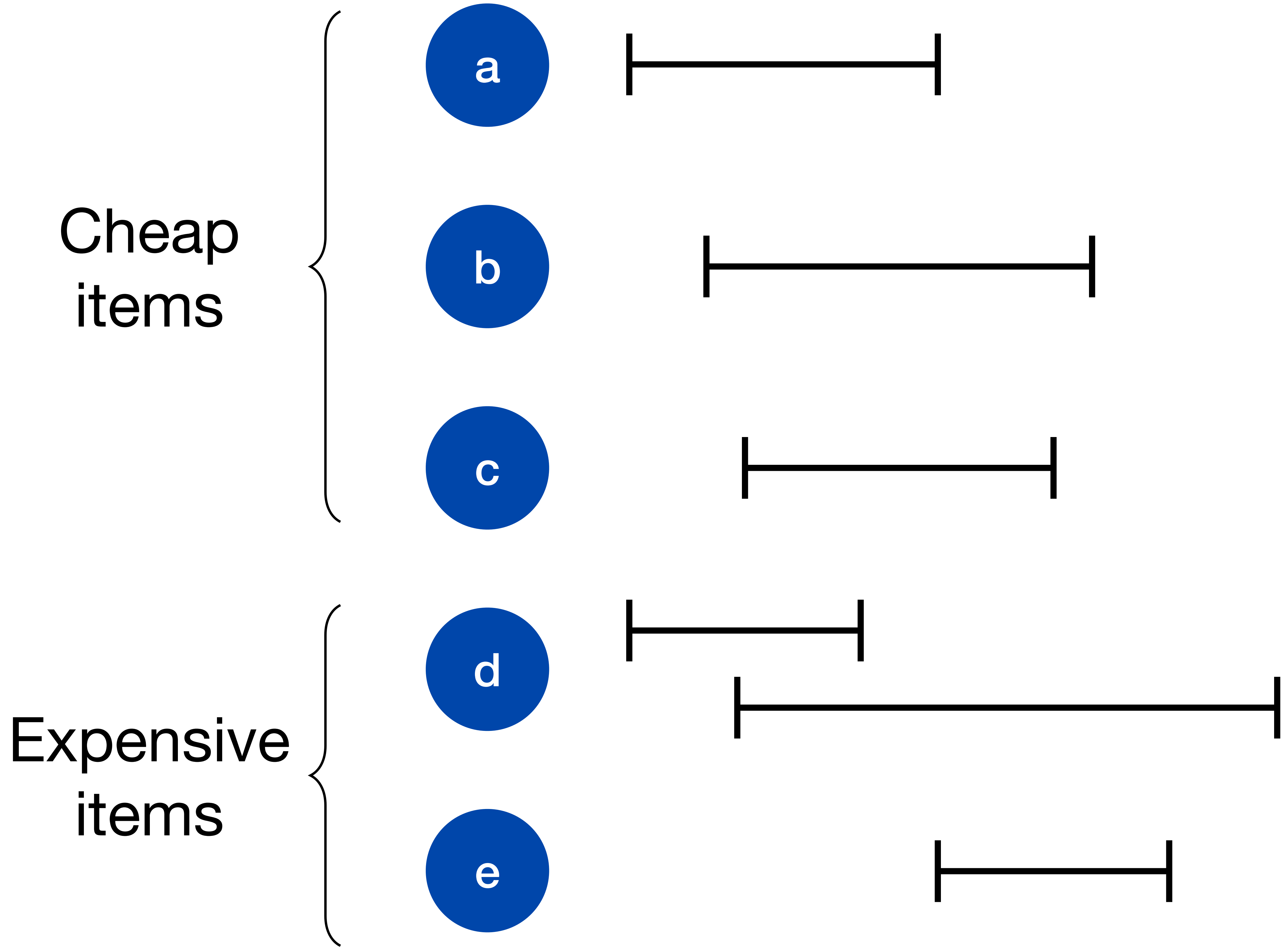
Algorithm

JRP with deadlines

UponDeadline(q):

if q is expensive:

Transmit item q is on



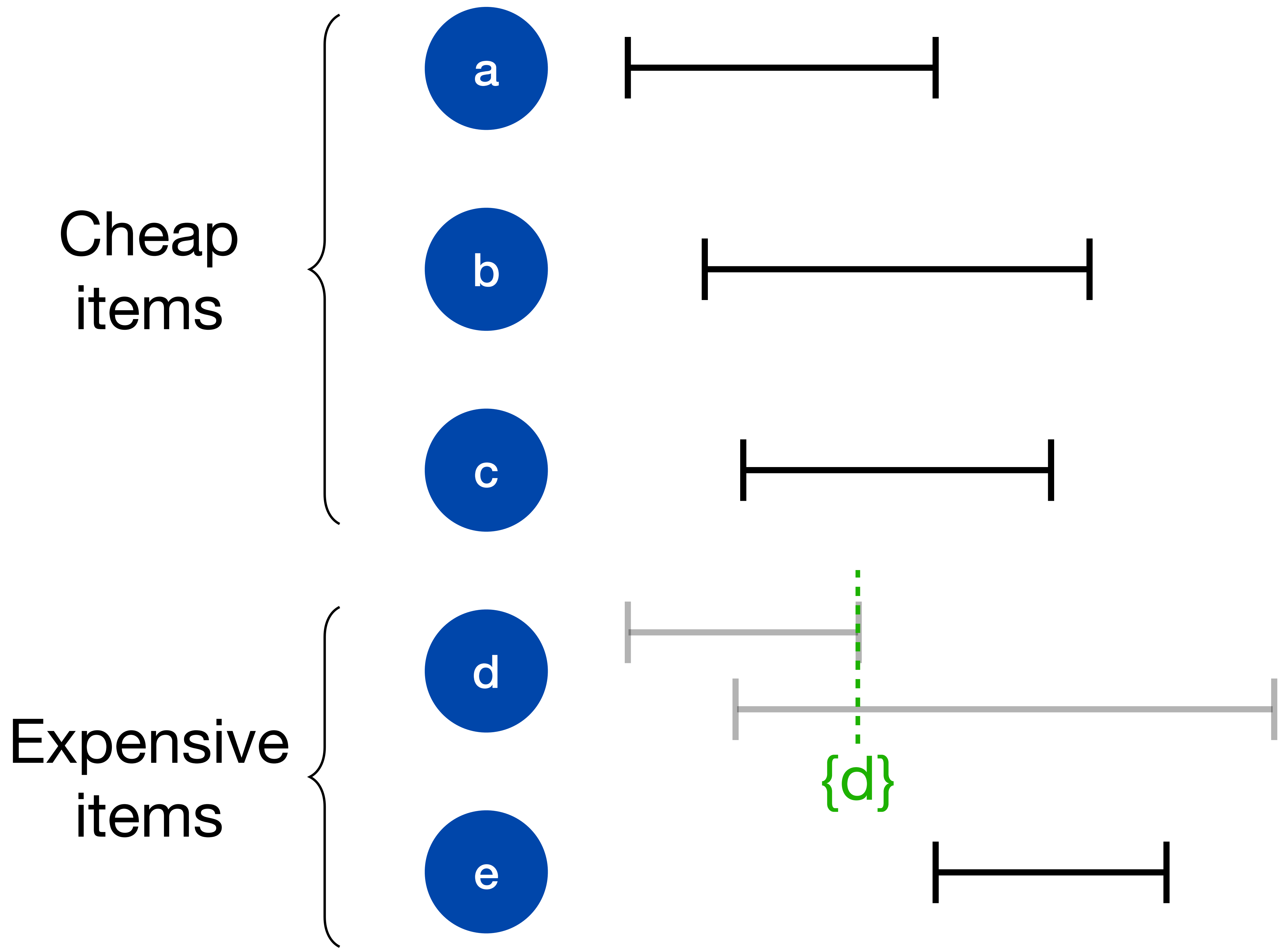
Algorithm

JRP with deadlines

UponDeadline(q):

if q is expensive:

Transmit item q is on



Algorithm

JRP with deadlines

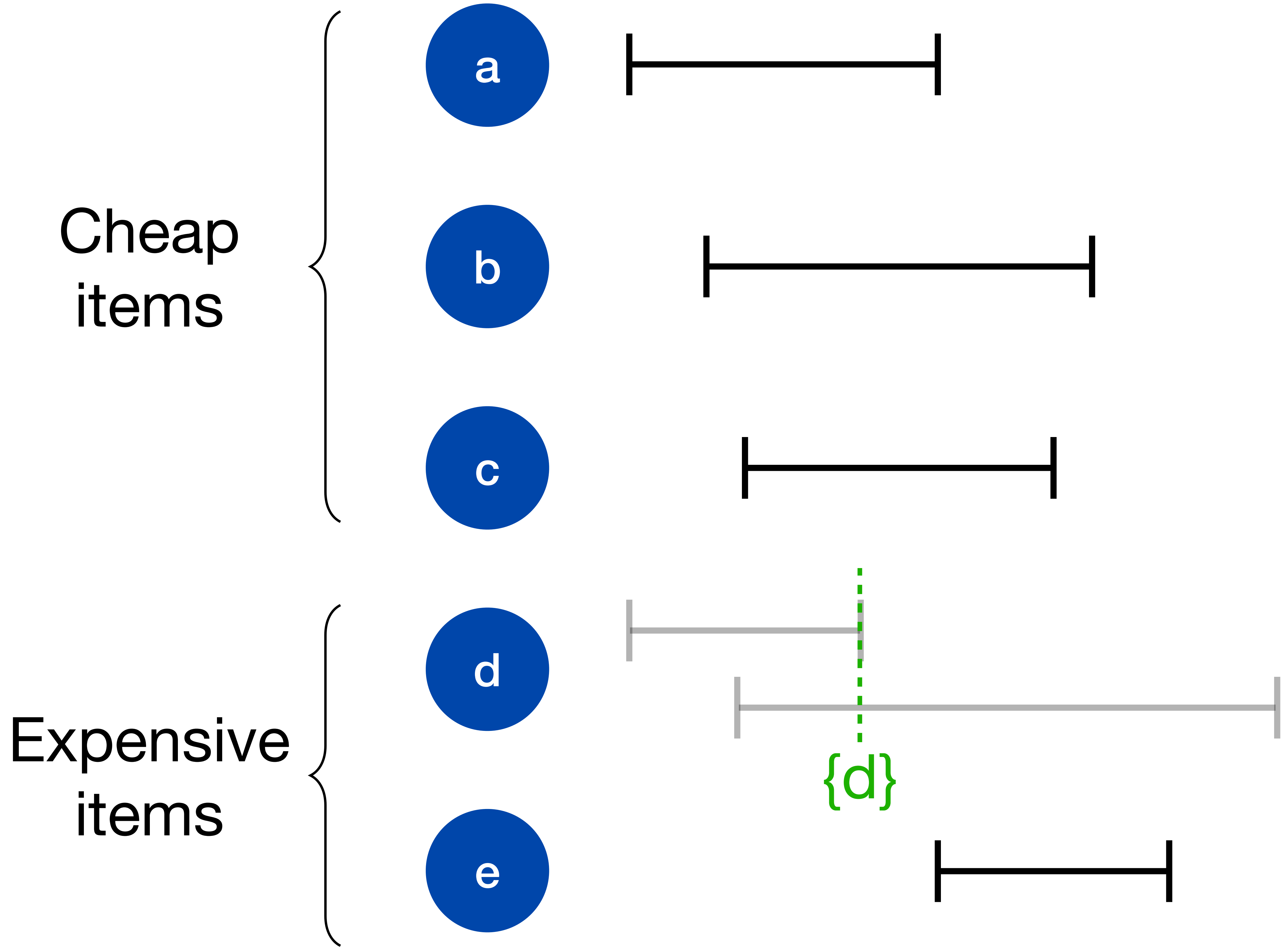
UponDeadline(q):

    if q is expensive:

        Transmit item q is on

    if q is cheap:

        Transmit all cheap items



Algorithm

JRP with deadlines

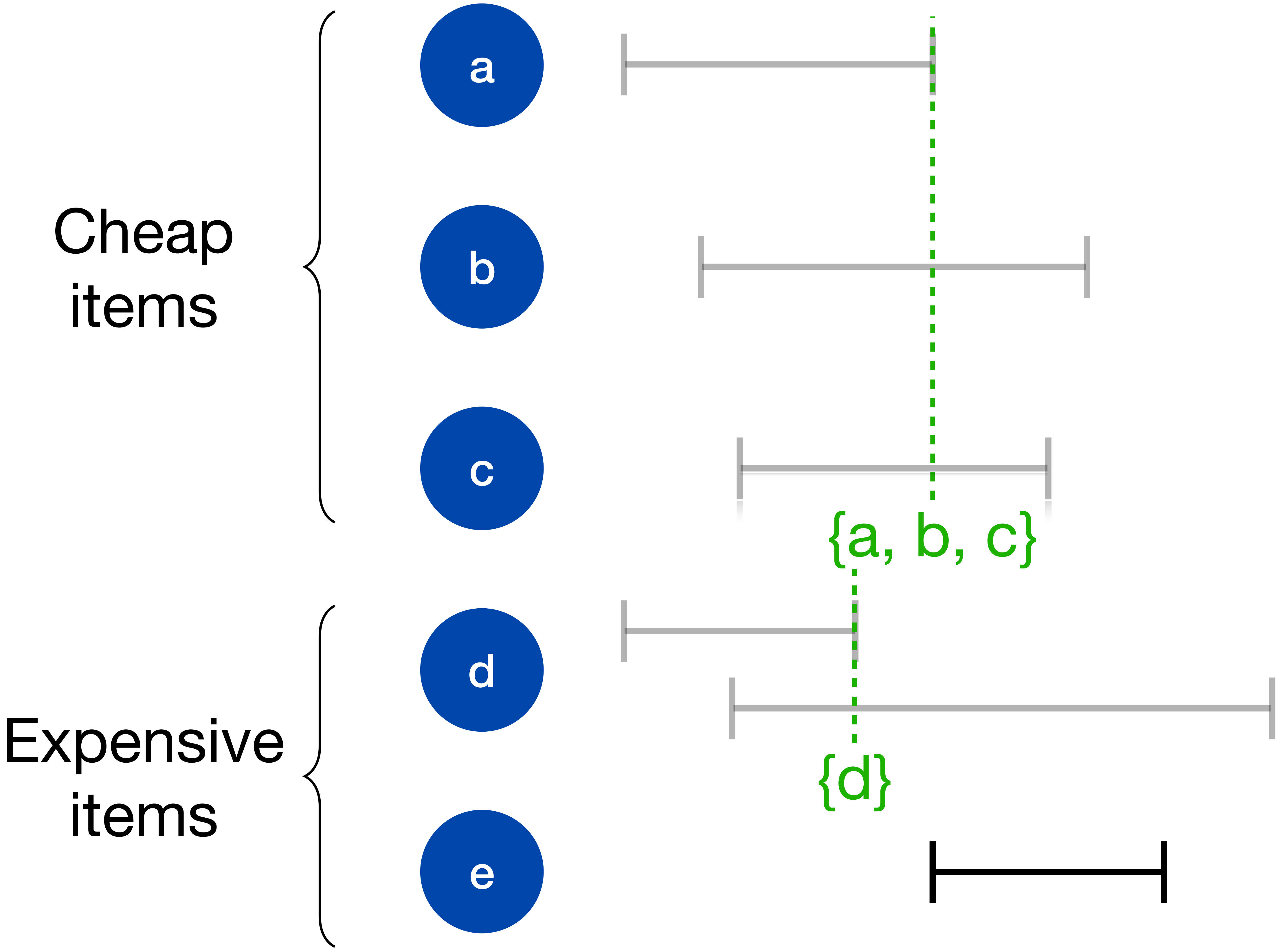
UponDeadline(q):

if q is expensive:

Transmit item q is on

if q is cheap:

Transmit all cheap items



Algorithm

JRP with deadlines

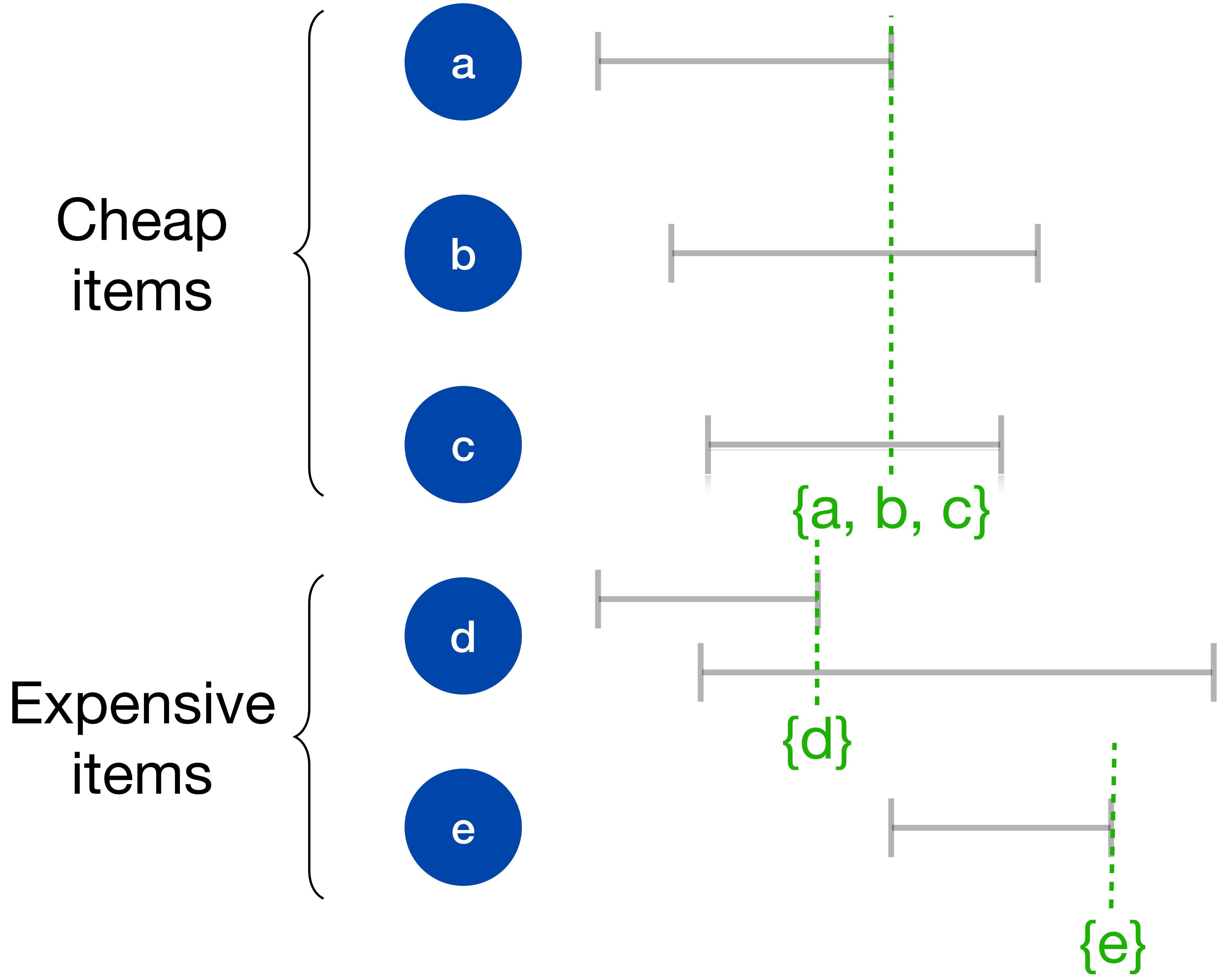
UponDeadline(q):

if q is expensive:

Transmit item q is on

if q is cheap:

Transmit all cheap items



Algorithm Analysis

JRP with deadlines

A transmission is either triggered by

1. a cheap request

OR

2. an expensive request

Algorithm Analysis

JRP with deadlines - Transmissions triggered by cheap requests

- x is cheap if $c(x) \leq K/\sqrt{n}$
- There are at most n cheap items
- Cost of all cheap items is at most $\dfrac{K}{\sqrt{n}} \times n = K\sqrt{n}$

Algorithm Analysis

JRP with deadlines - Transmissions triggered by cheap requests

When a transmission is triggered by a cheap request, all cheap items are transmitted

$\boxed{\text{Cost of transmission triggered by cheap request } q}$ $\leq K + K\sqrt{n}$

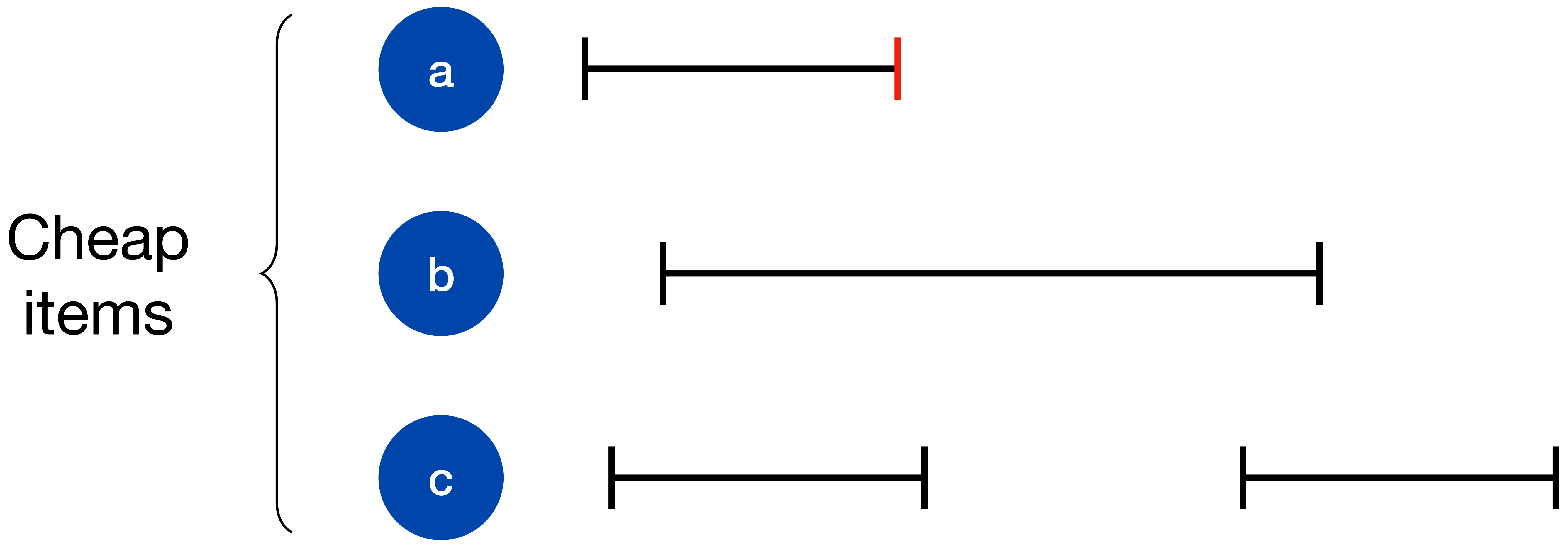
Algorithm Analysis

Charging optimal solution

Cost of transmission triggered by cheap request q	$\leq K + K\sqrt{n}$
$\downarrow K$	
Ordering fee cost of optimal transmission that served q	$= K$

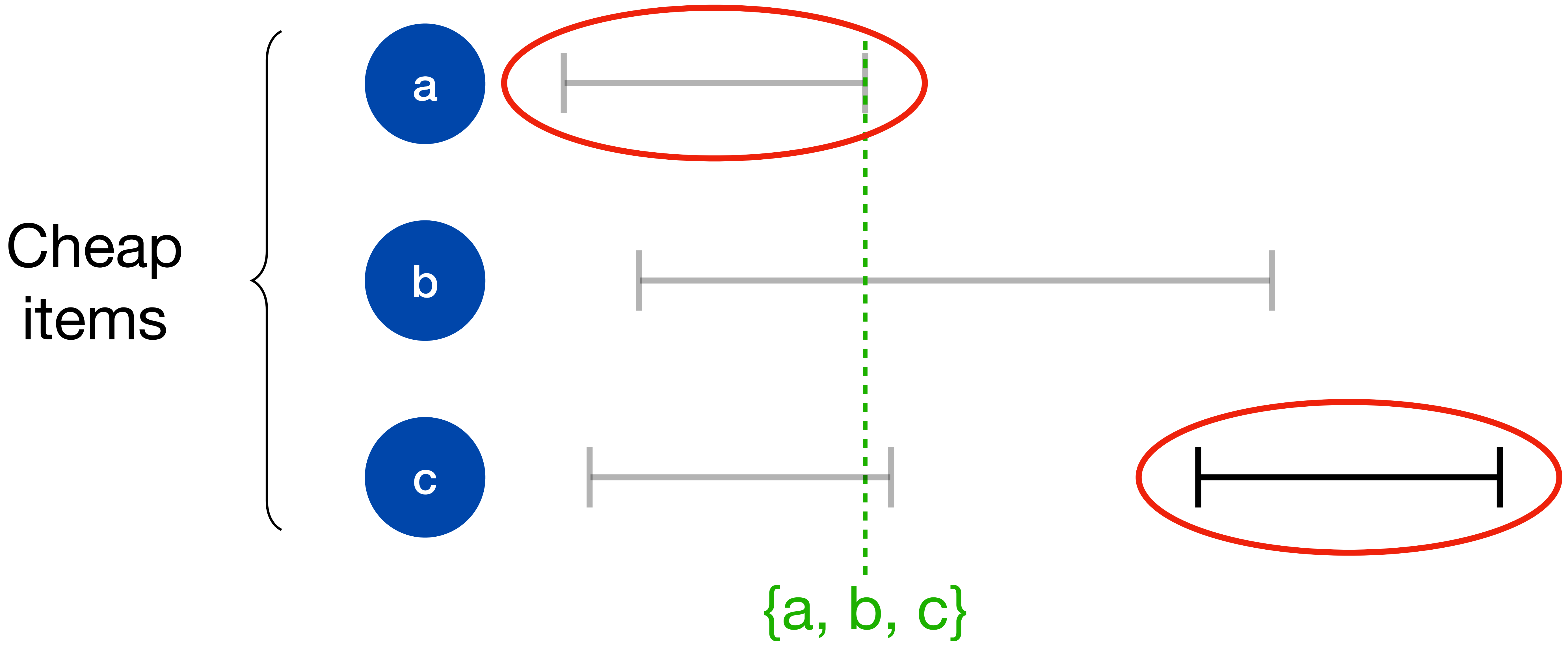
- Can we charge the same optimal transmission twice?

Algorithm Analysis



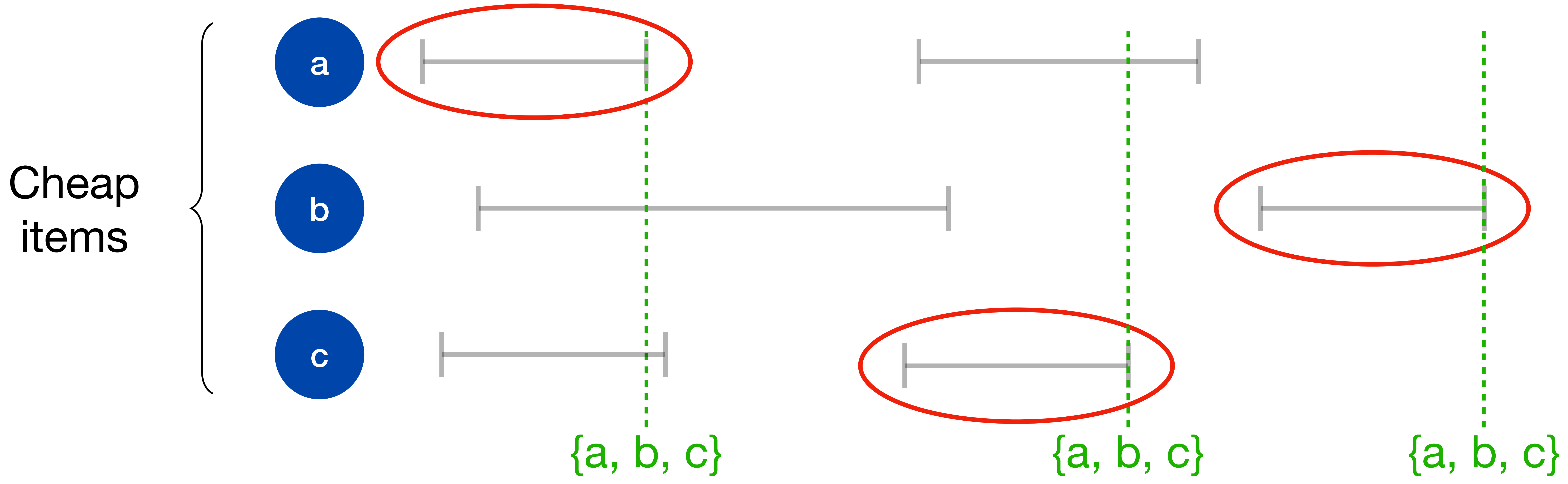
- Can we charge the same optimal transmission twice?

Algorithm Analysis



- Can we charge the same optimal transmission twice?

Algorithm Analysis



- Can we charge the same optimal transmission twice? **No**

Algorithm Analysis

JRP with deadlines - Transmissions triggered by expensive requests

x is expensive if $c(x) > K/\sqrt{n}$

Hence $K < \sqrt{n} \cdot c(x)$

Cost of a transmission is at most $K + c(x) < (\sqrt{n} + 1) \cdot c(x)$

Algorithm Analysis

Charging optimal solution

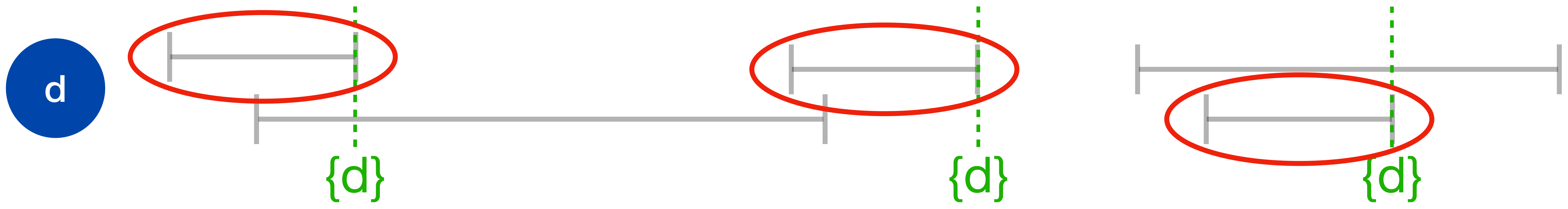
$$\boxed{\text{Cost of transmission triggered by expensive request q}} \quad \leq (\sqrt{n} + 1) \cdot c(x)$$

$$\big\downarrow c(x)$$

$$\boxed{\text{Cost of item } x \text{ in optimal transmission that served q}} \quad = c(x)$$

- Can we charge the **same item** in the same optimal transmission twice?

Algorithm Analysis



- Can we charge the **same item** in the same optimal transmission twice? **No**

Algorithm Analysis

Cost of transmission triggered by cheap request q $\leq K + K\sqrt{n}$

$(\sqrt{n} + 1)$ factor

Ordering fee cost of optimal transmission that served q $= K$

Cost of transmission triggered by expensive request q $\leq (\sqrt{n} + 1) \cdot c(x)$

$(\sqrt{n} + 1)$ factor

Cost of item x in optimal transmission that served q $= c(x)$

Competitive ratio = $\sqrt{\mathbf{n}} + 1$

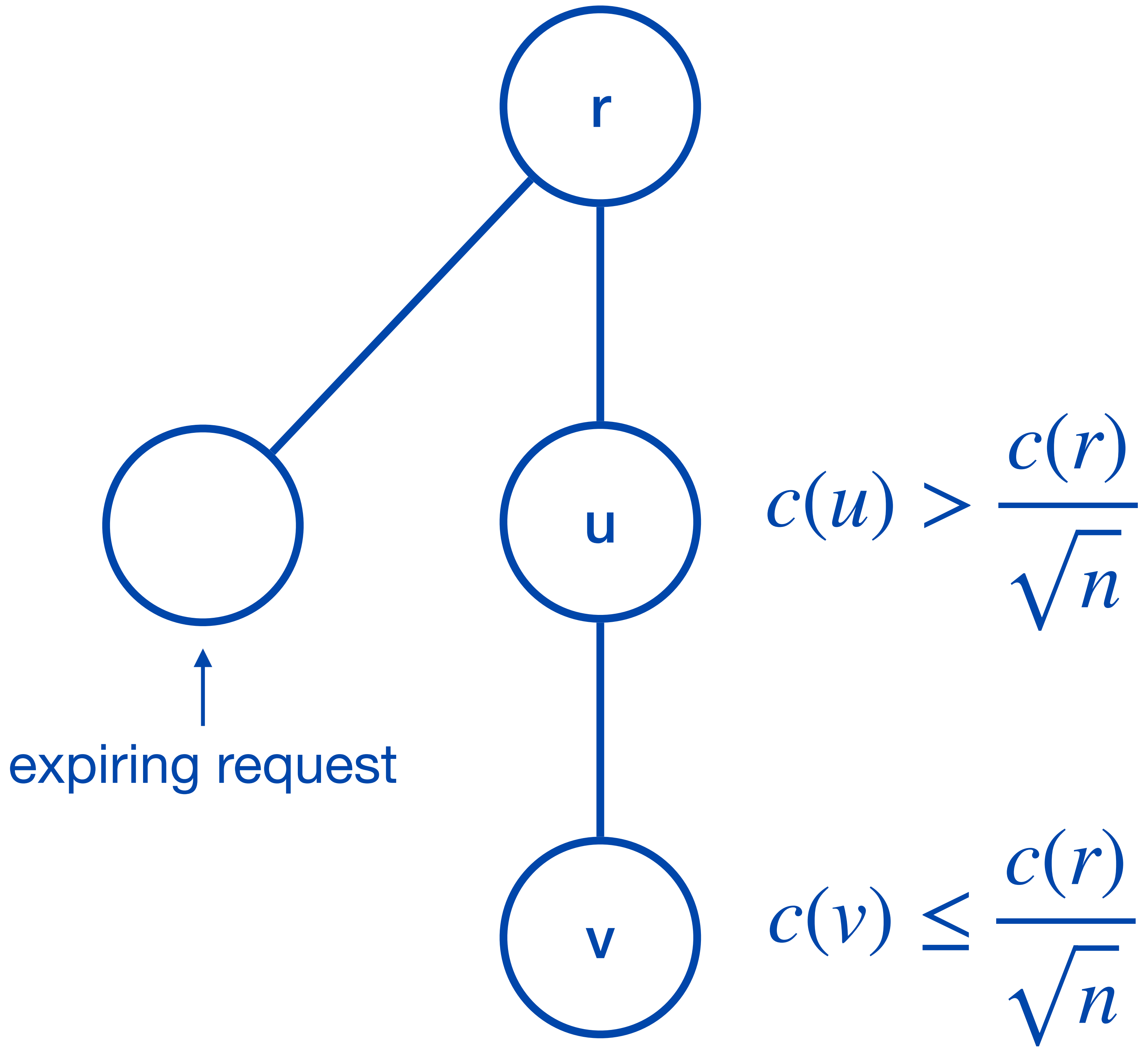
Algorithm

JRP with deadlines

- Can be generalised to the delay case, giving $O(\sqrt{n})$-competitive algorithm

Multi-level aggregation with deadlines

- Use a similar idea as in JRP
- What to do if we must go through an expensive node to reach a cheap node?
- Key idea: **Fractionally** pay ahead for nodes we can't transmit yet
- Theorem: There exists a $O(\sqrt{n} + \text{depth})$ - competitive non-clairvoyant algorithm for MLAP with deadlines



Summary

	Non-clairvoyant
JRP with deadlines	lower bound: $\Omega(\sqrt{n})$ upper bound: $O(\sqrt{n})$
JRP with delay	$O(\sqrt{n})$
MLAP with deadlines	$O(\sqrt{n}$ + depth)

Technique: Divide items into cheap and expensive relative to fixed ordering cost

Open Problems

- Extend algorithm for MLAP to the delay case

Open Problems

- Extend algorithm for MLAP to the delay case
- Study more general problems in the non-clairvoyant setting
    - Steiner-tree with deadlines/delay

References

Yossi Azar, Ashish Chiplunkar, Shay Kutten, & Noam Touitou (2020). Set Cover with Delay - Clairvoyance Is Not Required. In *28th Annual European Symposium on Algorithms (ESA 2020)* (pp. 8:1–8:21). Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

Rodrigo A. Carrasco and Kirk Pruhs and Cliff Stein and José Verschae (2018). The Online Set Aggregation Problem. In LATIN 2018: Theoretical Informatics - 13th Latin American Symposium, Buenos Aires, Argentina, April 16-19, 2018, Proceedings (pp. 245–259). Springer.

Marcin Bienkowski and Jaroslaw Byrka and Marek Chrobak and Lukasz Jez and Dorian Nogneng and Jir\'\i Sgall (2014). Better Approximation Bounds for the Joint Replenishment Problem. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014* (pp. 42–54). SIAM.

Niv Buchbinder and Tracy Kimbrel and Retsef Levi and Konstantin Makarychev and Maxim Sviridenko (2013). Online Make-to-Order Joint Replenishment Model: Primal-Dual Competitive Algorithms*. Oper. Res., 61(4), 1014–1029.*

Niv Buchbinder and Moran Feldman and Joseph (Seffi) Naor and Ohad Talmon (2017). O(depth)-Competitive Algorithm for Online Multi-level Aggregation. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19* (pp. 1235–1244). SIAM.