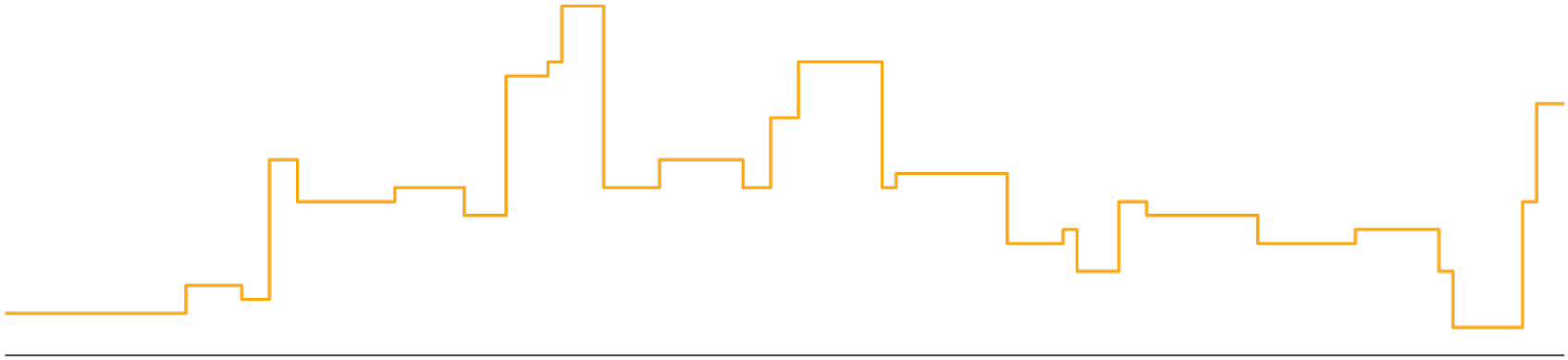
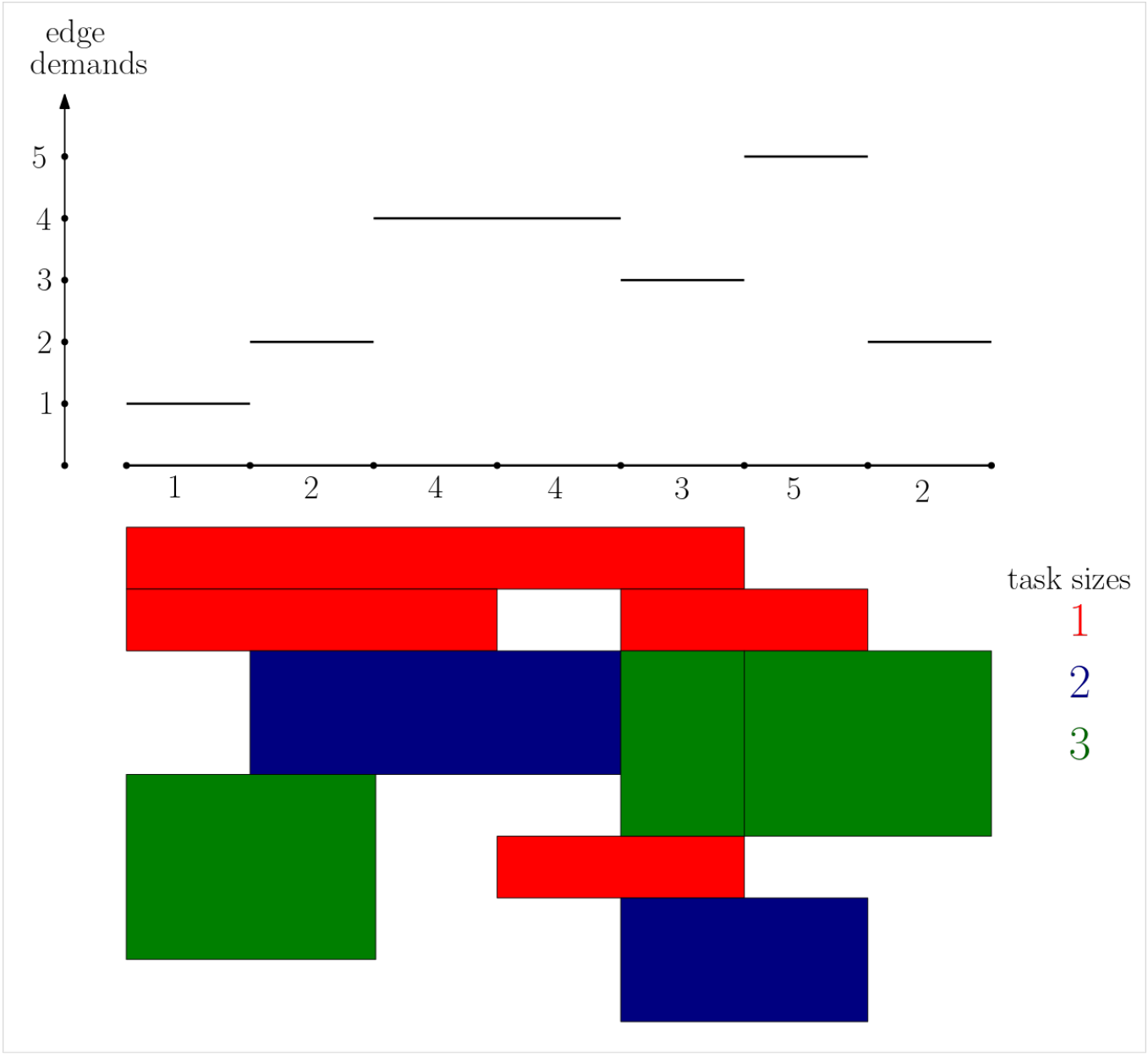


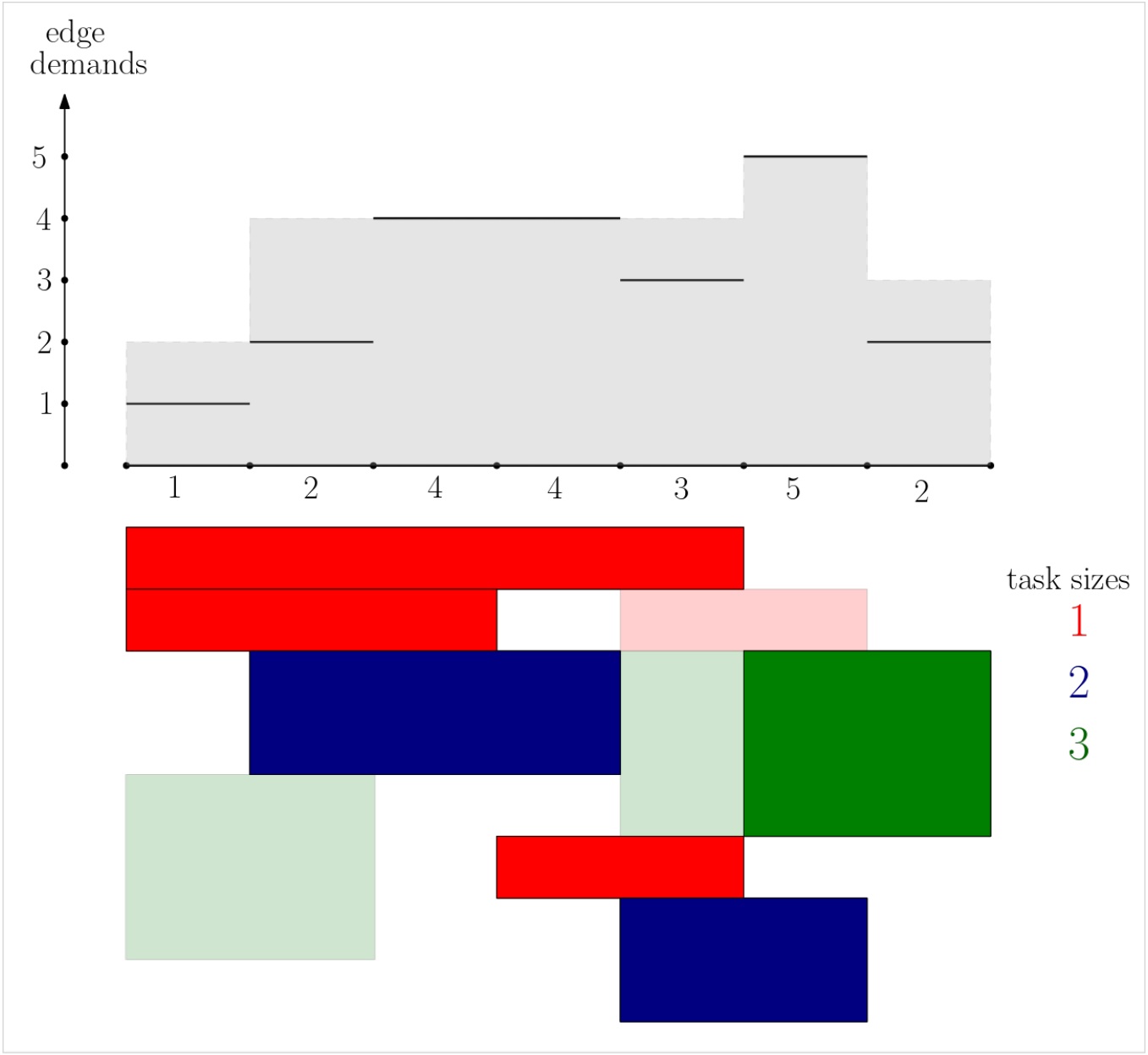
Fixed-parameter algorithms for Unsplittable Flow Cover.

Andrés Cristi, Mathieu Mari, Andreas Wiese

Unsplittable Flow cover on a Path



Unsplittable Flow cover on a Path



What is known for UFP-cover

- Strongly NP-hard

  Reduction to Caching in the fault model

- (polynomial time) 4-approximation algorithm

  [Bar-Noy, Bar-Yehuda, Freund, Naor, Schieber, 2000]

- QPTAS: $(1+\epsilon)$-approximation in time $2^{O_\epsilon(poly(\log n))}$

  [Höln, Mestre, Wiese, 2014]

Fixed-Parameter Tractable algorithms

"a certain parameter is small"

Definition

An algorithm is an **FPT-α-approximation algorithm** for a (minimisation) problem (and a specific parameter) if

- its running time is $f(k)n^{O(1)}$ for all inputs of size n
- it returns a solution of size $\leq \alpha \cdot OPT$ or attests that the parameter is $> k$.

In this talk: $|OPT| \leq k$

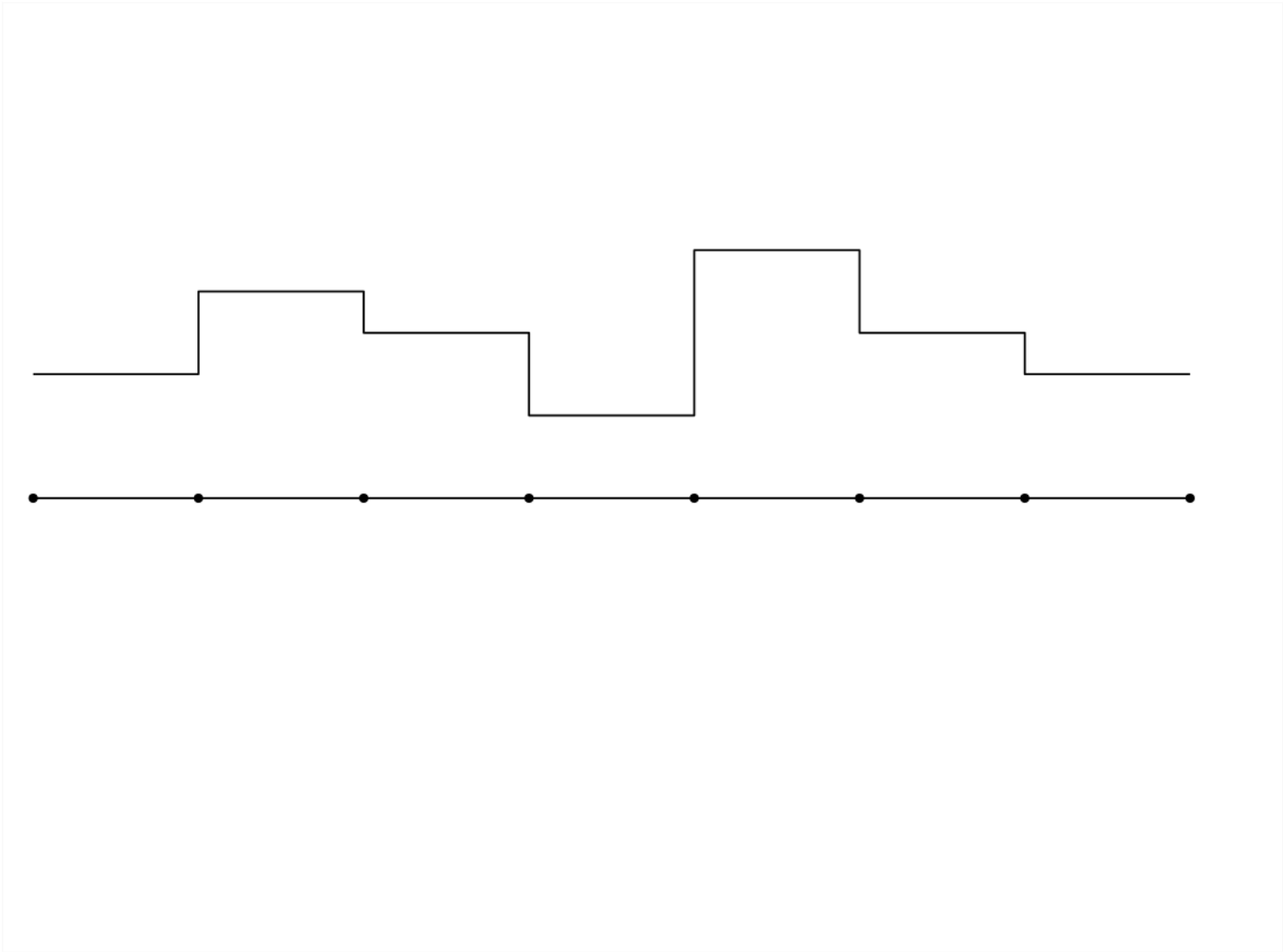
Our results for UFP-cover

1. W[1]-hard
2. exact FPT-algorithm when parameterized by $k + \#$**task sizes** *
3. exact FPT-algorithm using **resource augmentation** *
4. FPT-2-approximation *
5. **FPT-$(1+\epsilon)$-approximation**

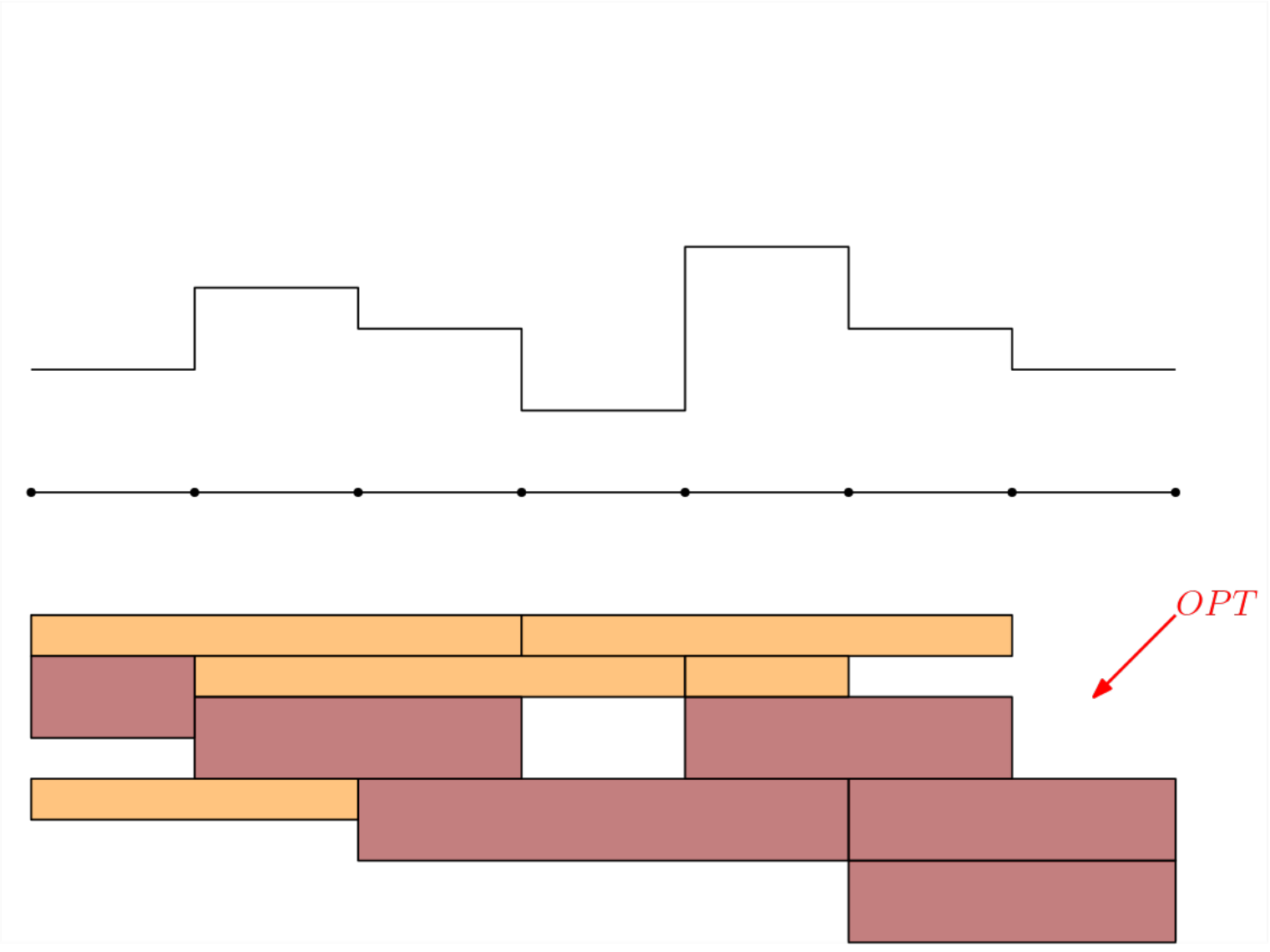
2. $\longrightarrow$ 3. $\longrightarrow$ 4.

* also works with task costs (loosing a factor $(1+\epsilon)$)

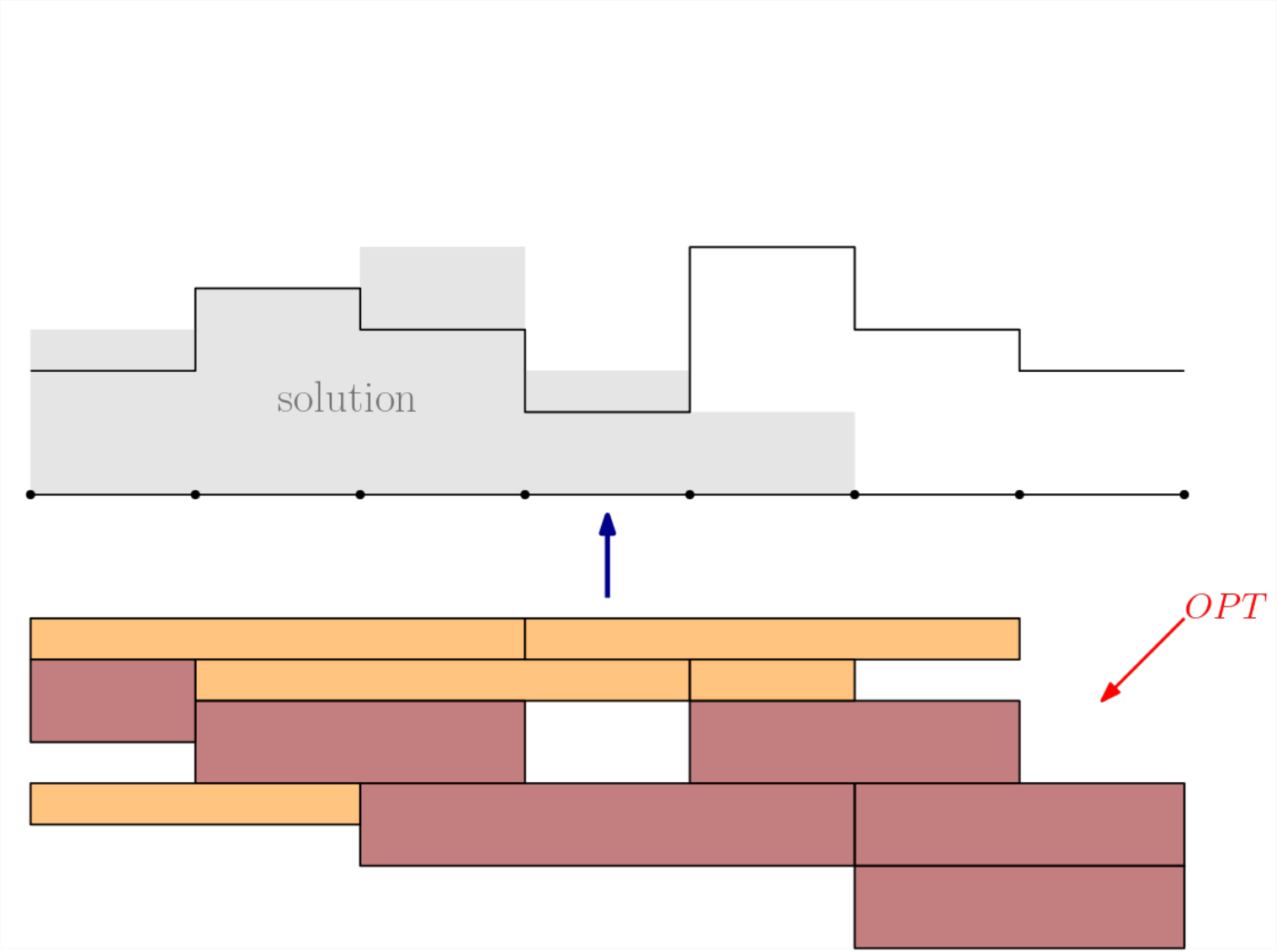
Tasks of size 1 or size 2



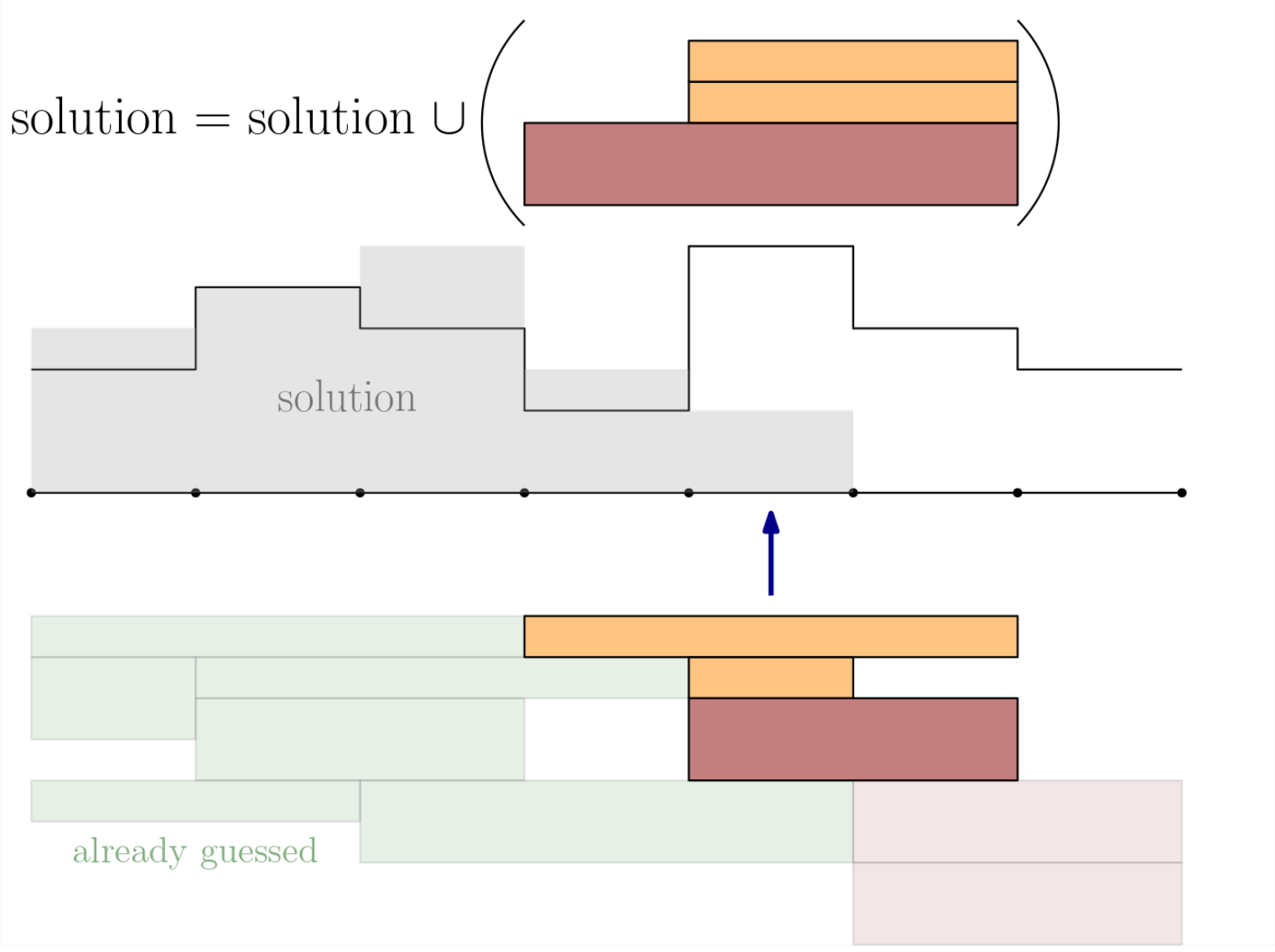
Tasks of size 1 or size 2



Tasks of size 1 or size 2



Tasks of size 1 or size 2



Few different task sizes

Theorem

There is an algorithm that solves UFP-cover in time $k^{O(k' \cdot k)} n^{O(1)}$ when the number of task sizes is at most k'.

solution $= \emptyset$

For each **edge** e from left to right:

For each **size** s:

1. if e is uncovered, *guess* the number m of tasks of size s in *OPT* that cover e, not yet guessed.
2. solution $\leftarrow$ m such tasks with **rightmost endpoints**

Few different task sizes

Theorem

There is an algorithm that solves UFP-cover in time $k^{O(k' \cdot k)} n^{O(1)}$ when the number of task sizes is at most k'.

solution $= \emptyset$

For each **edge** e from left to right: $(k+1)^{k'k}$ guesses

  For each **size** s: $(k+1)^{k'}$ guesses

1. if e is uncovered, *guess* the number m of tasks of size s in *OPT* that cover e, not yet guessed.
2. solution $\leftarrow$ m such tasks with **rightmost endpoints**

Few sizes $\rightarrow$ Arbitrary sizes

Theorem (Resource augmentation)

There is an algorithm with running time $f(k, \delta) \cdot n^{O(1)}$ that either:

- outputs a solution of size k, that satisfies all the $\frac{\text{demand}(e)}{1+\delta}$
- or, attests that there is no solution of size k for the original demands

1. round down tasks sizes to $(1+\delta)^{\ell}$.
2. guess which groups are used by OPT
3. **apply the previous algorithm**

here, we assume: max demand $\leq$ poly(n) but the theorem is true in general.

Few sizes $\rightarrow$ Arbitrary sizes

Theorem (Resource augmentation)

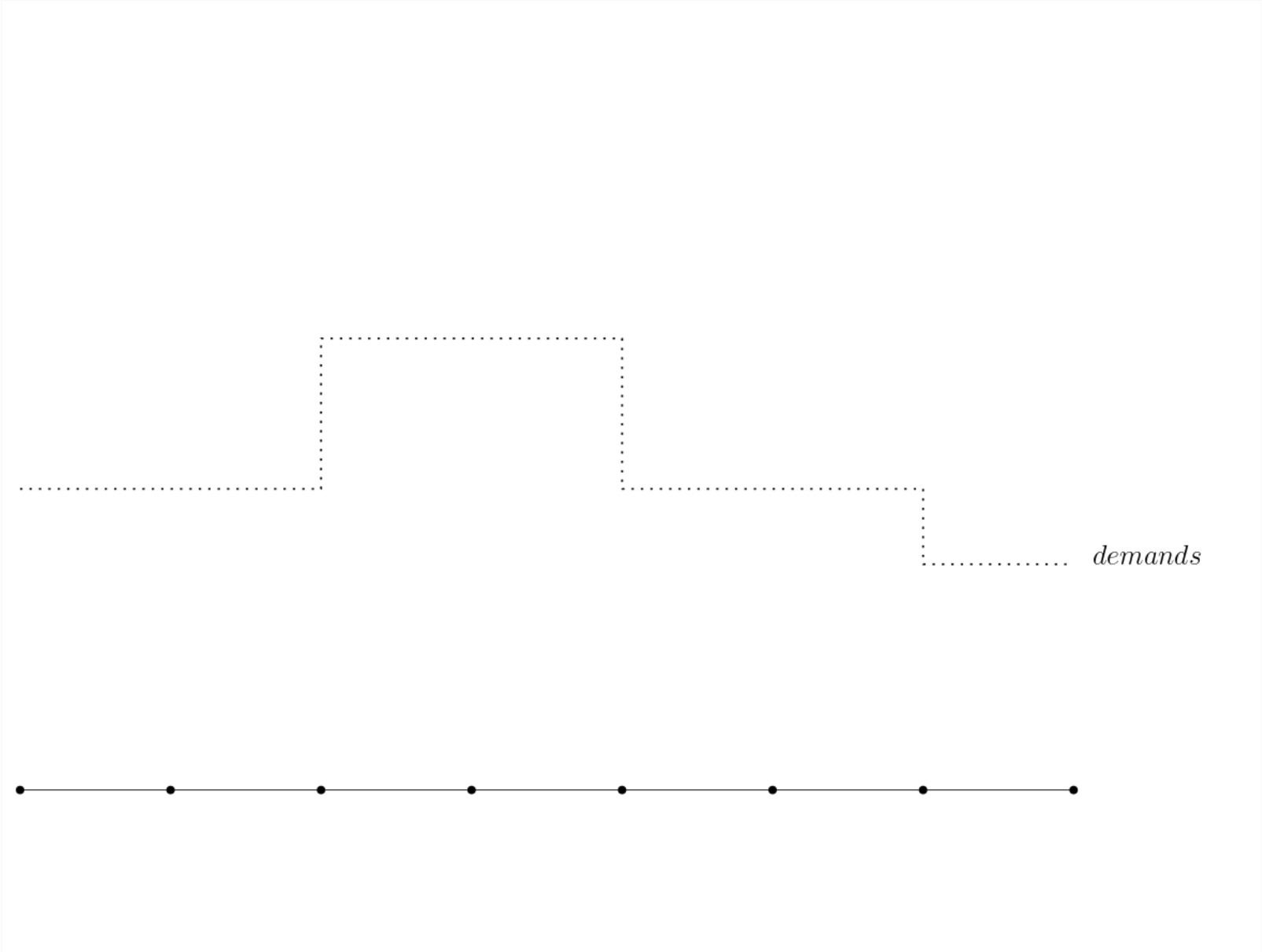
There is an algorithm with running time $f(k, \delta) \cdot n^{O(1)}$ that either:

- outputs a solution of size k, that satisfies all the $\frac{\text{demand}(e)}{1+\delta}$
- or, attests that there is no solution of size k for the original demands

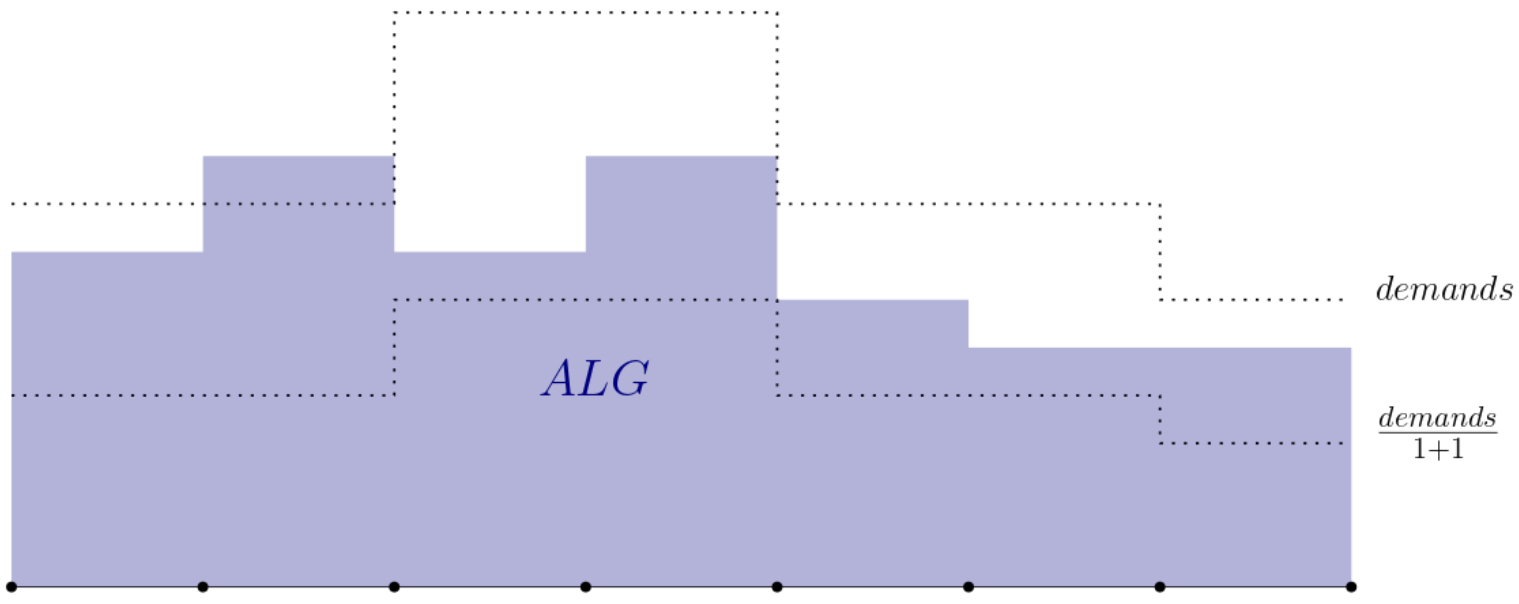
1. round down tasks sizes to $(1+\delta)^{\ell}$.
   at most $O(\log_{1+\delta} n)$ rounded sizes
2. guess which groups are used by OPT
   $\binom{O(\log_{1+\delta} n)}{k} \leq (\frac{1}{\delta})^{O(k)}(n + k^{O(k)})$
3. **apply the previous algorithm**

here, we assume: max demand $\leq$ poly(n) but the theorem is true in general.

Resource augmentation $\rightarrow$ FPT-2-approximation

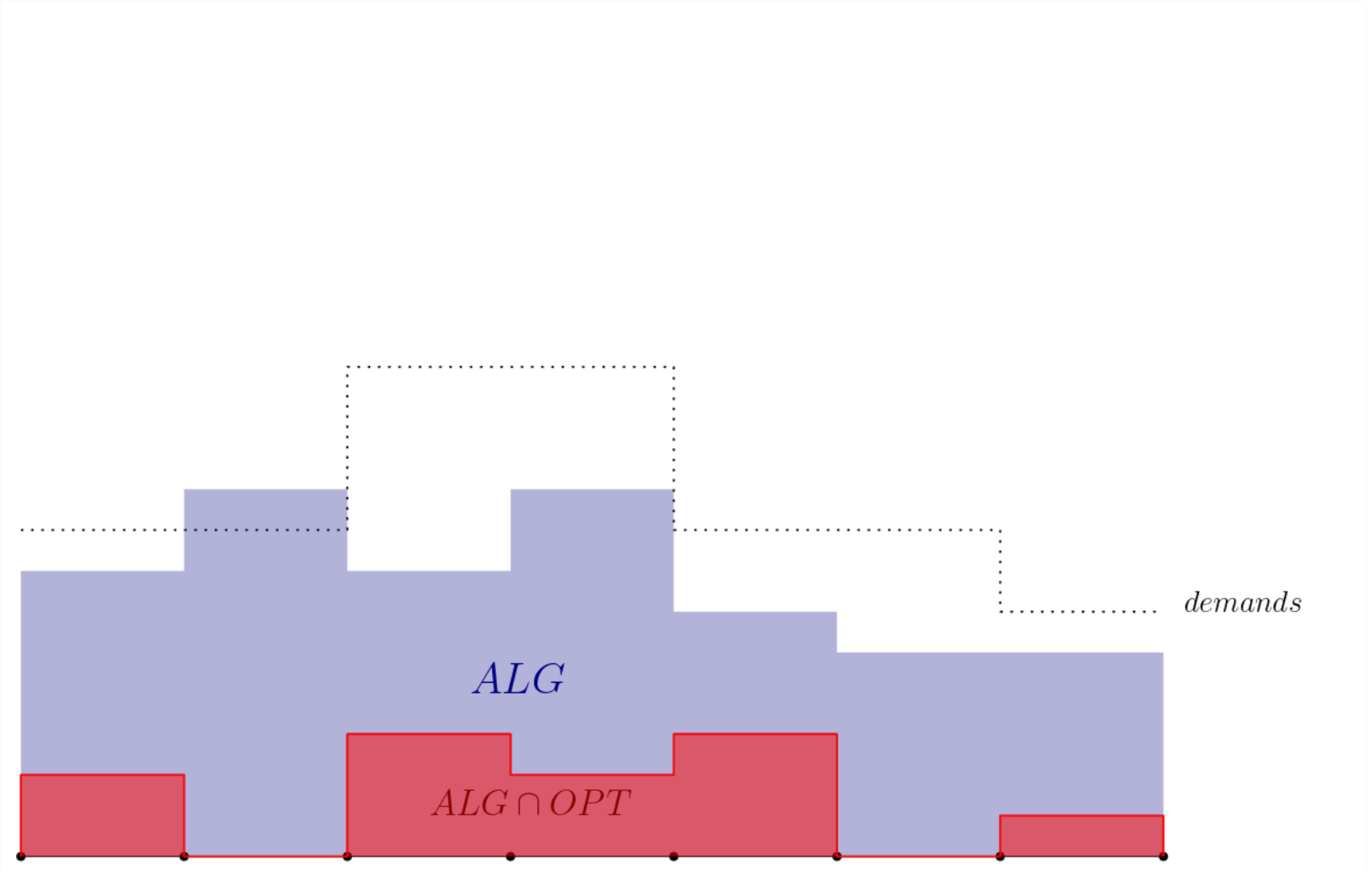


Resource augmentation $\rightarrow$ FPT-2-approximation



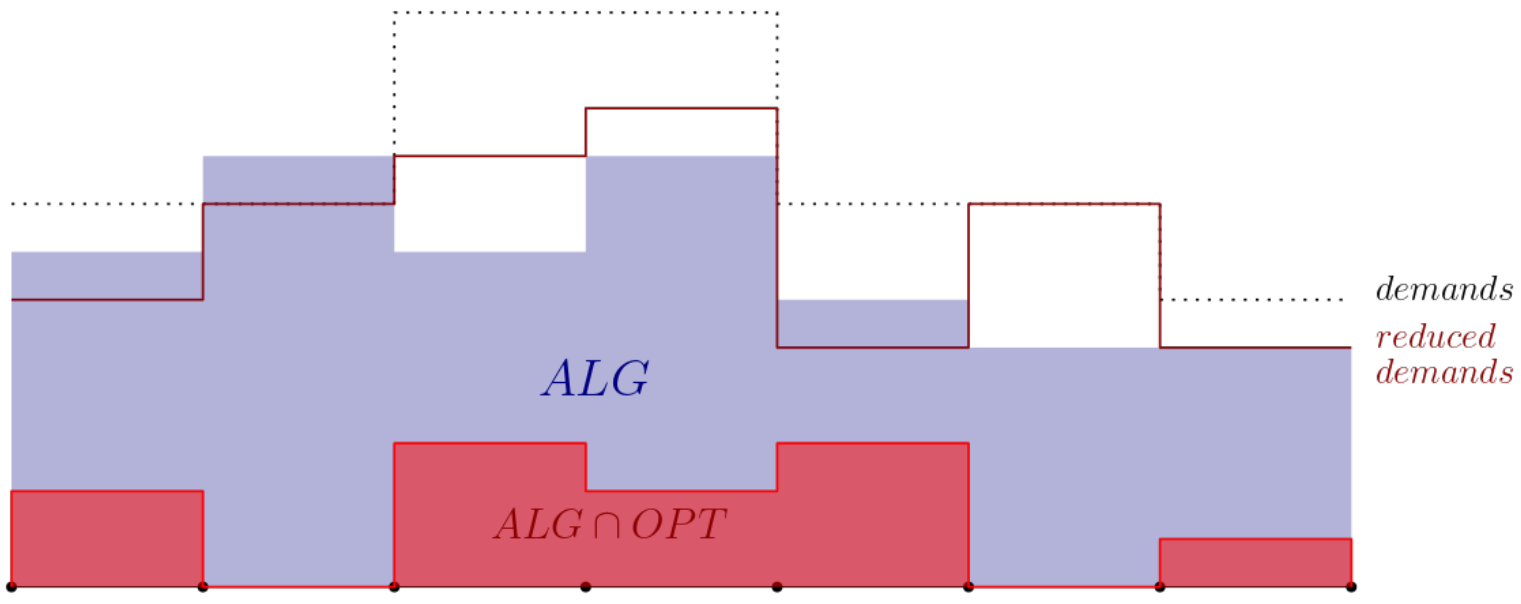
step 1: compute $ALG \leftarrow$ resource augmentation with $\delta = 1$

Resource augmentation $\rightarrow$ FPT-2-approximation



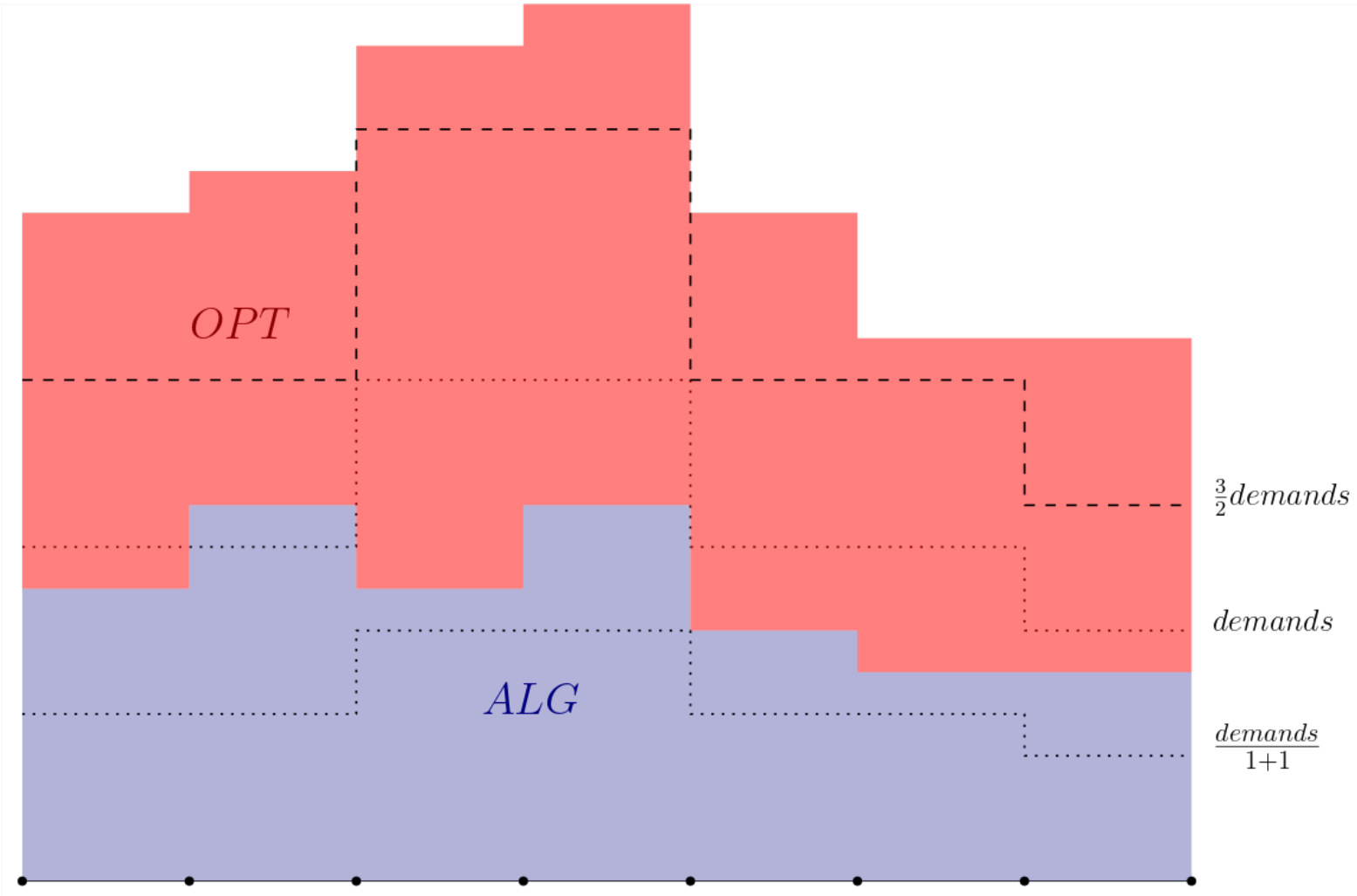
step 2: guess $ALG \cap OPT$

Resource augmentation $\rightarrow$ FPT-2-approximation



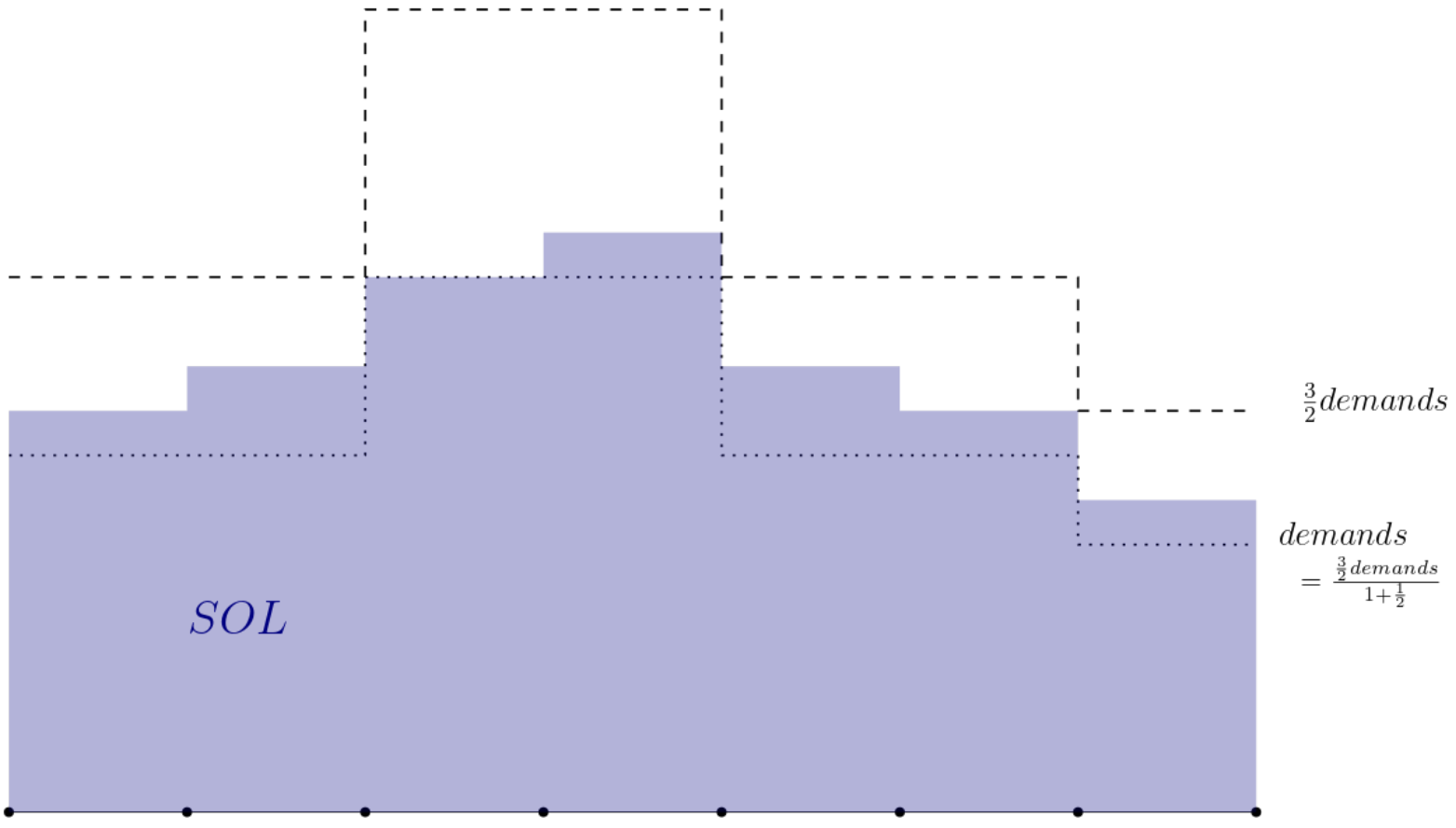
step 2: guess $ALG \cap OPT$
- if $ALG \cap OPT \neq \emptyset$: recurse with the *reduced demands* and $k \leftarrow k - |ALG \cap OPT|$

Resource augmentation $\rightarrow$ FPT-2-approximation



step 2: guess $ALG \cap OPT$
- if $ALG \cap OPT = \emptyset$: there exists a solution of size $\leq 2k$ that covers $\geq \frac{3}{2}demands$

Resource augmentation $\rightarrow$ FPT-2-approximation



step 2: guess $ALG \cap OPT$
- if $ALG \cap OPT = \emptyset$: $SOL \leftarrow$ resource augmentation algorithm with $new\ demands = \frac{3}{2}demands$, $k' = 2k$ and $\delta = \frac{1}{2}$

FPT-2-approximation

Theorem

UFP-cover has a FPT-**2**-approximation

1. $ALG \leftarrow$ resource augmentation algorithm with $\delta = 1$
2. guess $ALG \cap OPT$, and *recurse* if $ALG \cap OPT \neq \emptyset$
3. run the resource augmentation algorithm with: **new demands** $= \frac{3}{2} \cdot$ demands *and* $\delta = \frac{1}{2}$

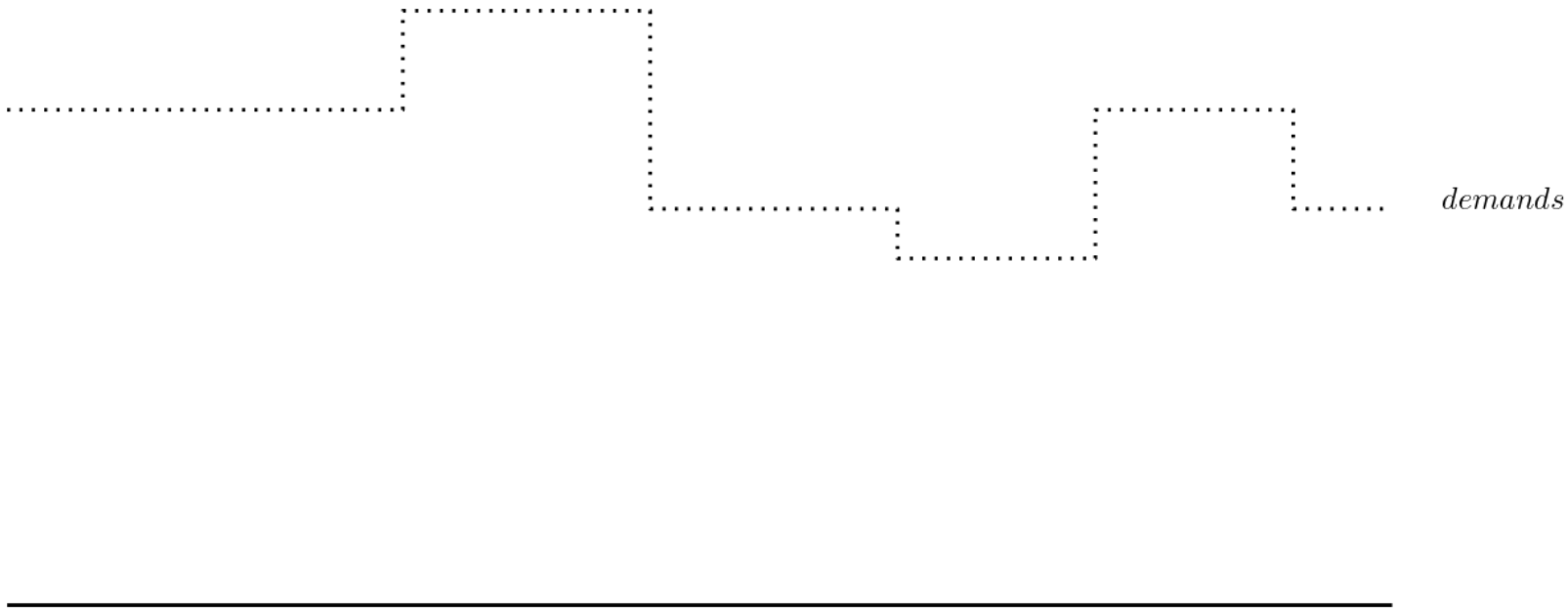
Parameterized Approximation Scheme

Theorem: PAS

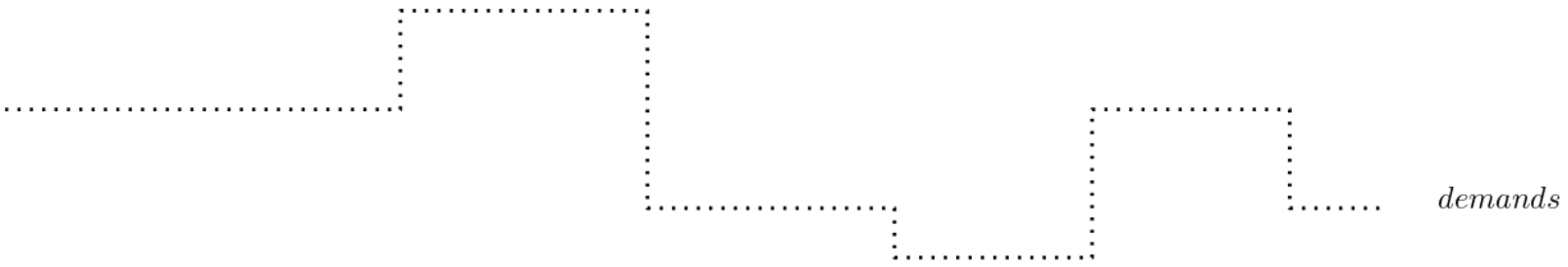
There is an algorithm for UFP-cover with running time $k^{O(k)}n^{(1/\epsilon)^{O(1/\epsilon)}}$ that either

- returns a solution of size $\leq (1+\epsilon)k$
- or, attests that there is no solution of size $\leq k$

Some flavors of the PAS



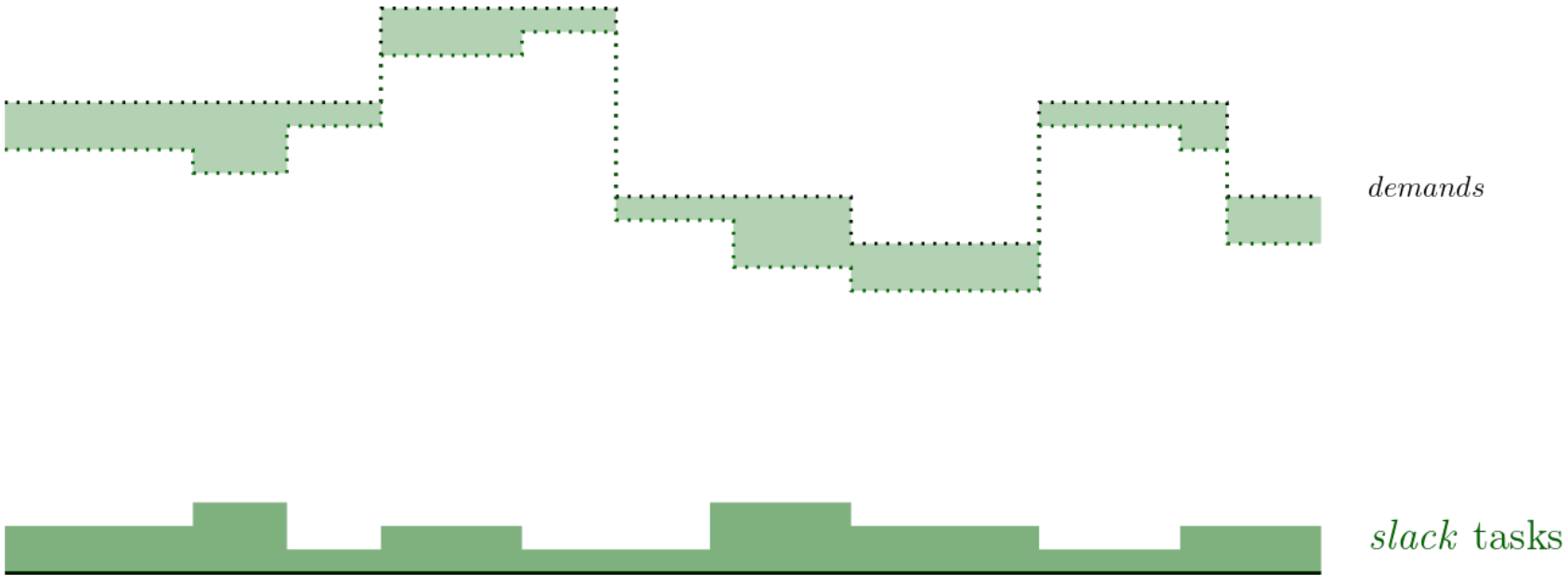
Some flavors of the PAS



Step 1: compute a set of $O(\epsilon k)$ *slack tasks*

based on a Lemma for (classic) UFP [Batra, Garg, Kumar, Mömke, Wiese, 2015]

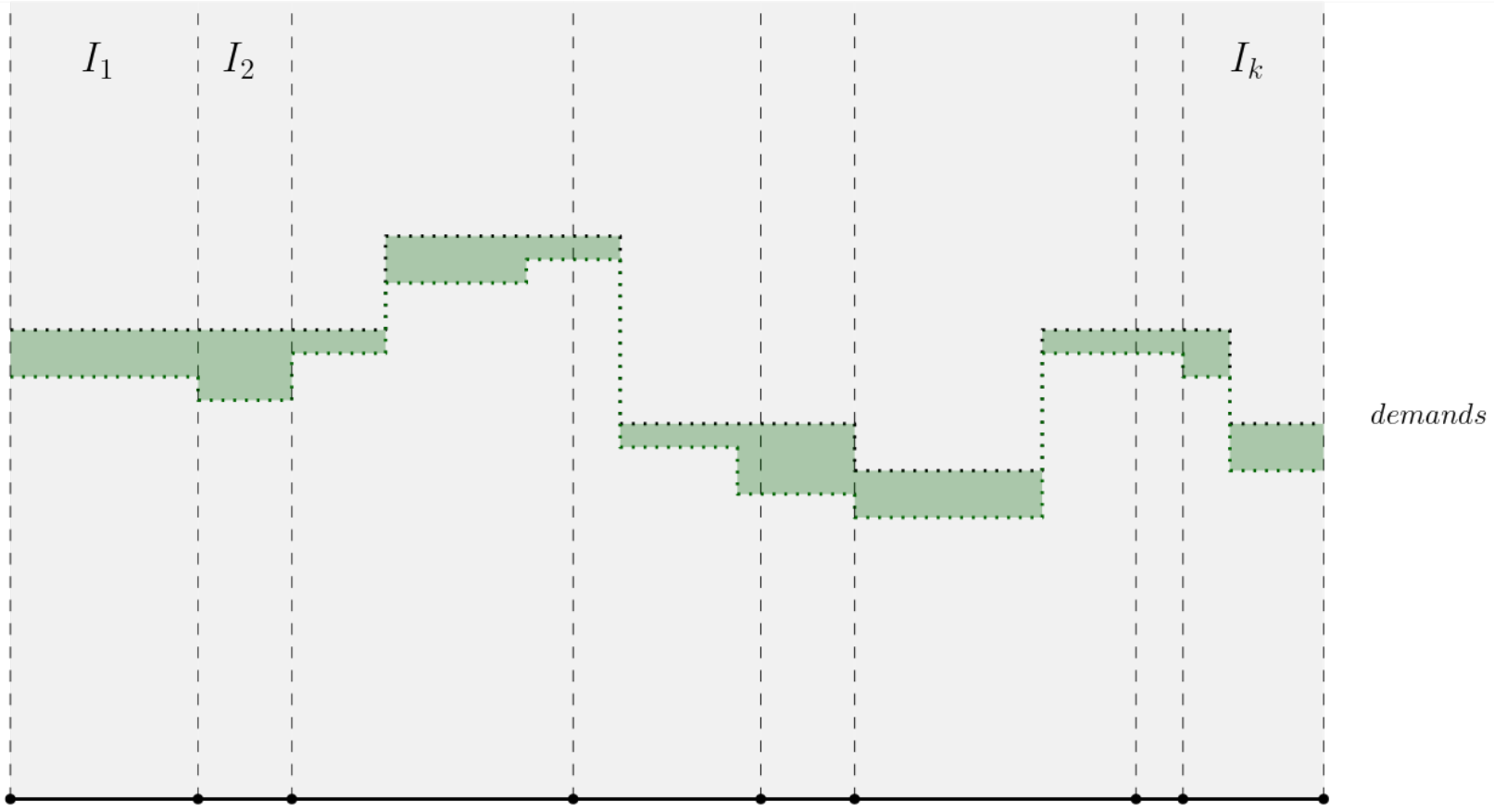
Some flavors of the PAS



Step 1: compute a set of $O(\epsilon k)$ *slack tasks*

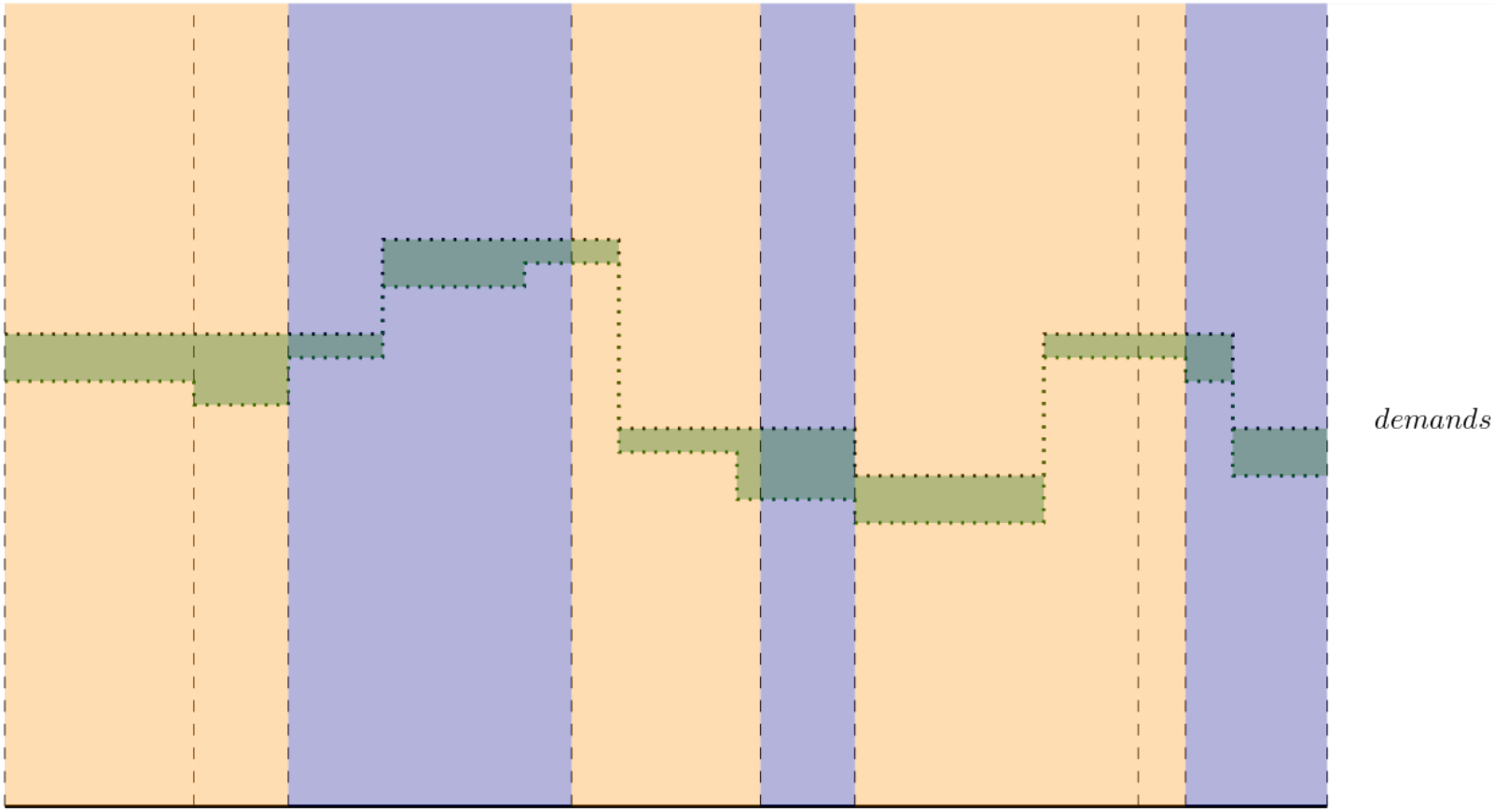
based on a Lemma for (classic) UFP [Batra, Garg, Kumar, Mömke, Wiese, 2015]

Some flavors of the PAS



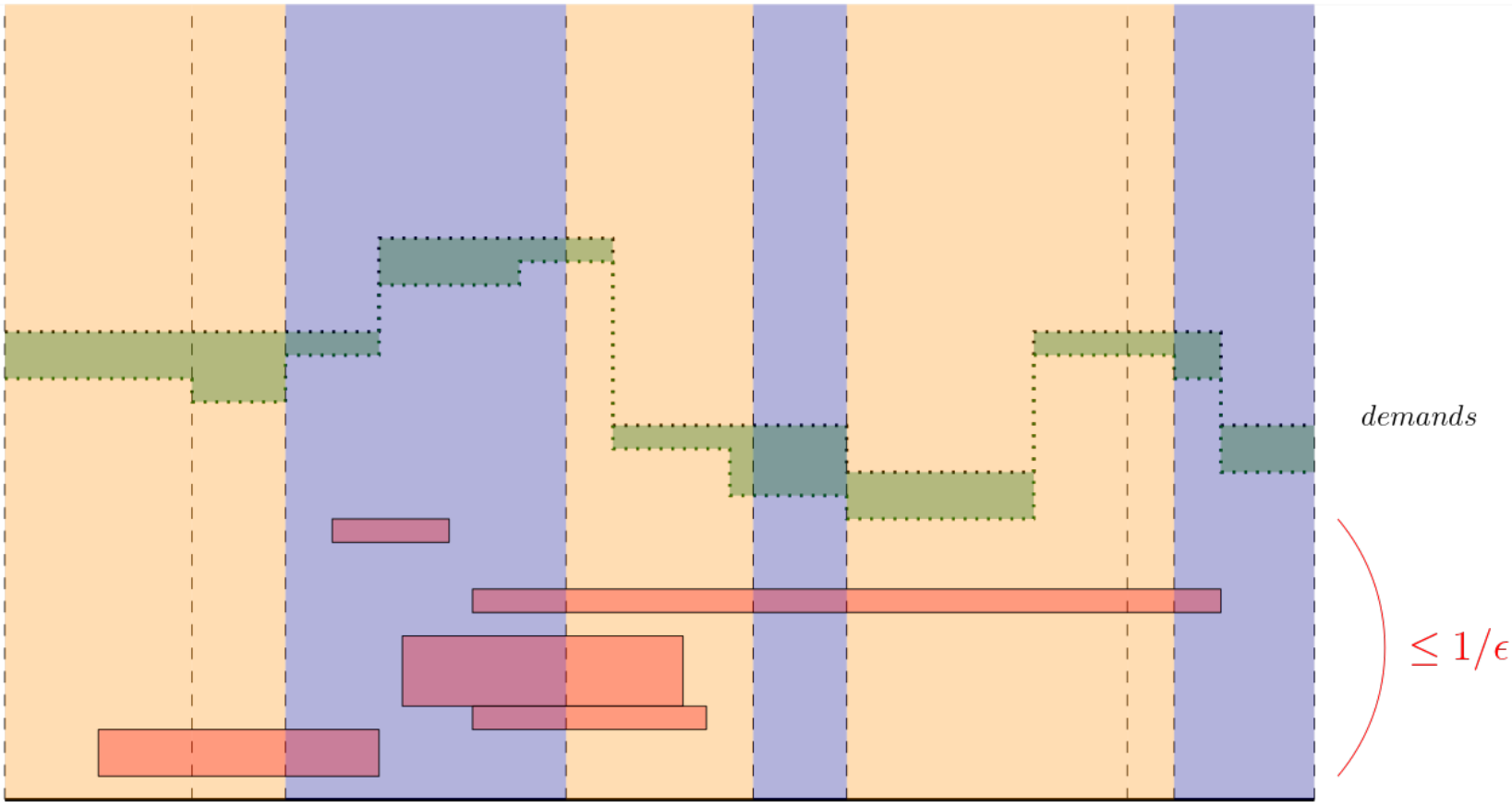
Step 2: partition the path into $O(k)$ sub-intervals

Some flavors of the PAS



Step 3: guess the dense and sparse sub-intervals

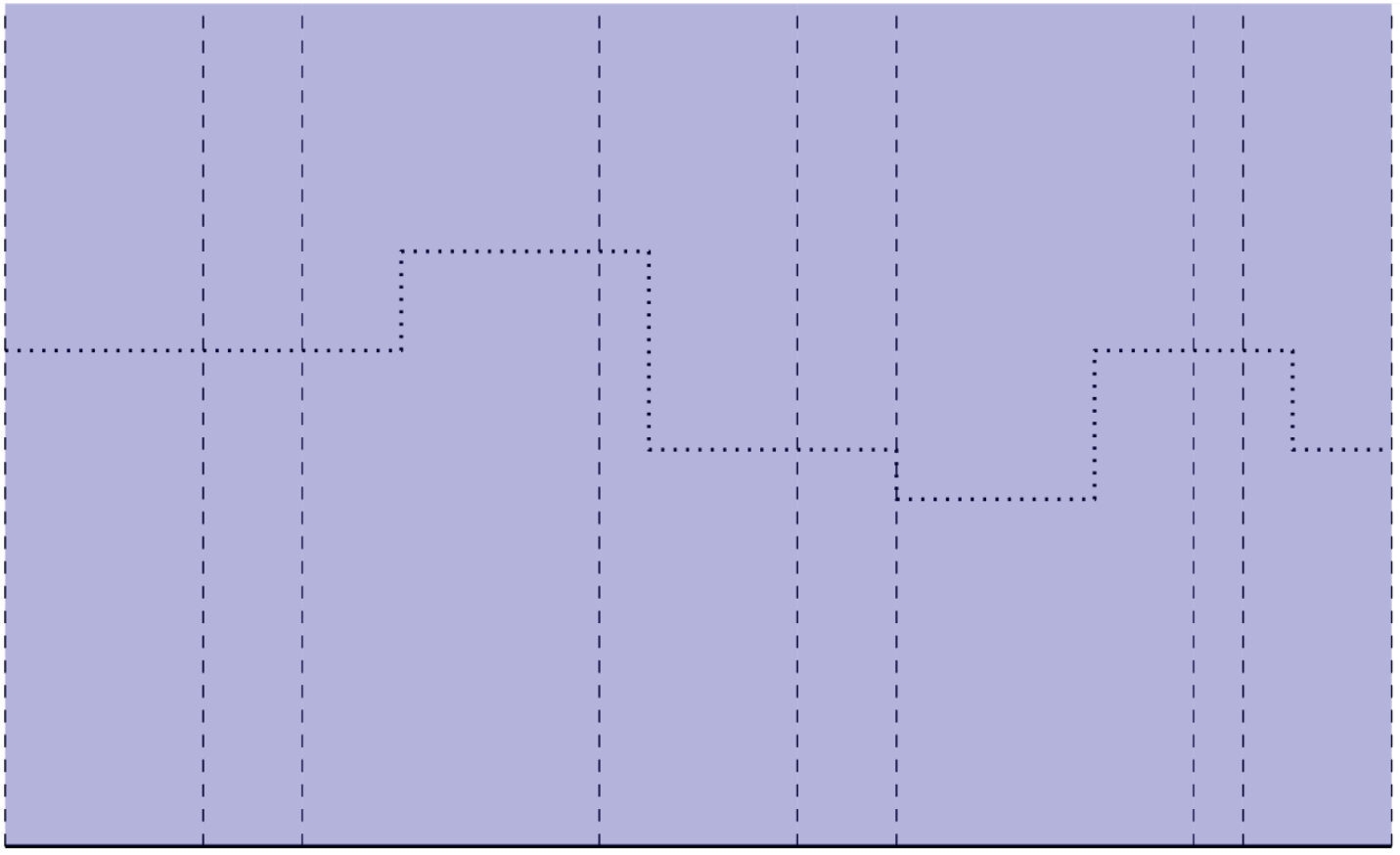
Some flavors of the PAS



Step 3: guess the dense and sparse sub-intervals

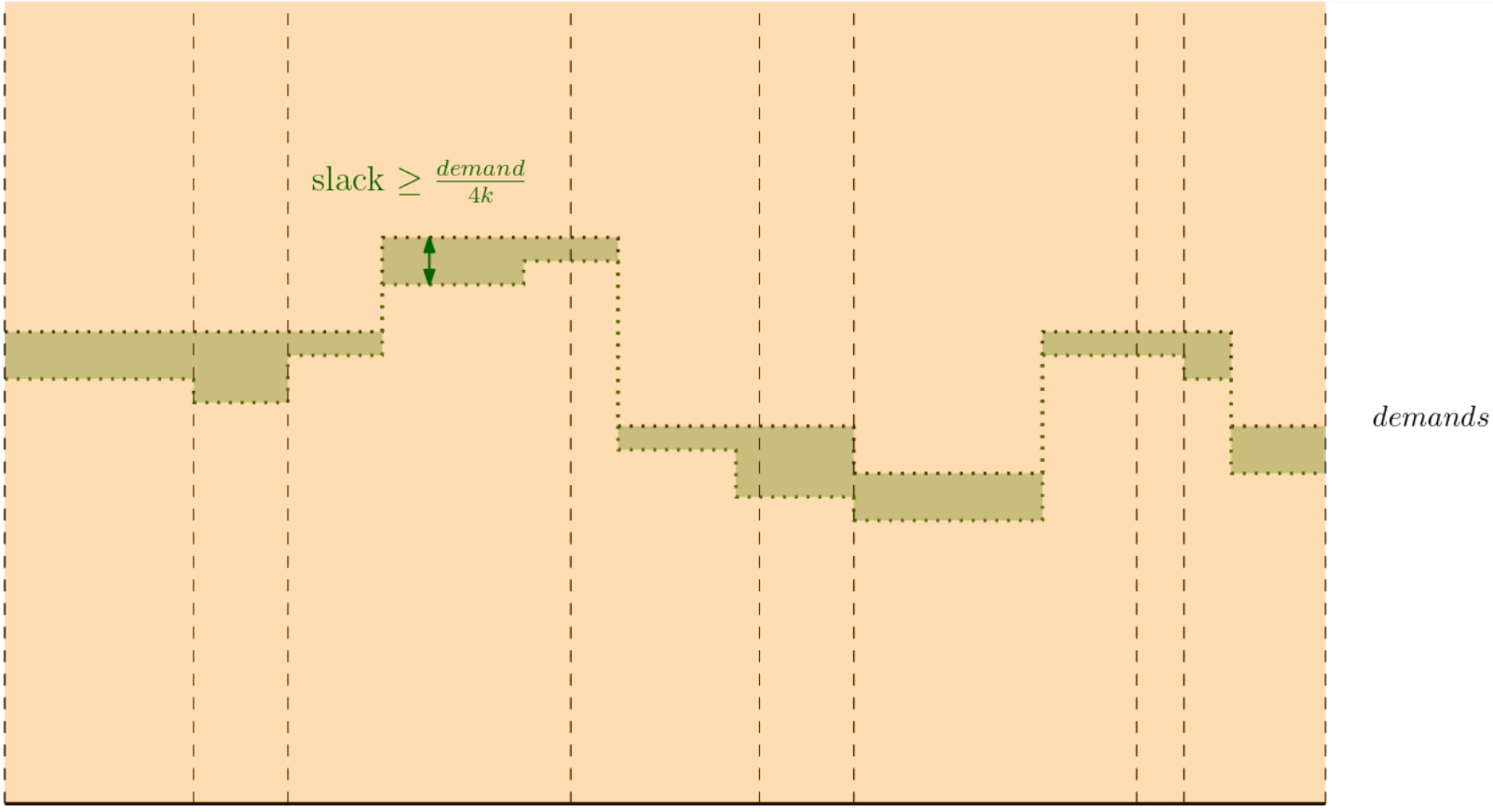
I is sparse: only few tasks of OPT start or end in I and dense otherwise

Some flavors of the PAS



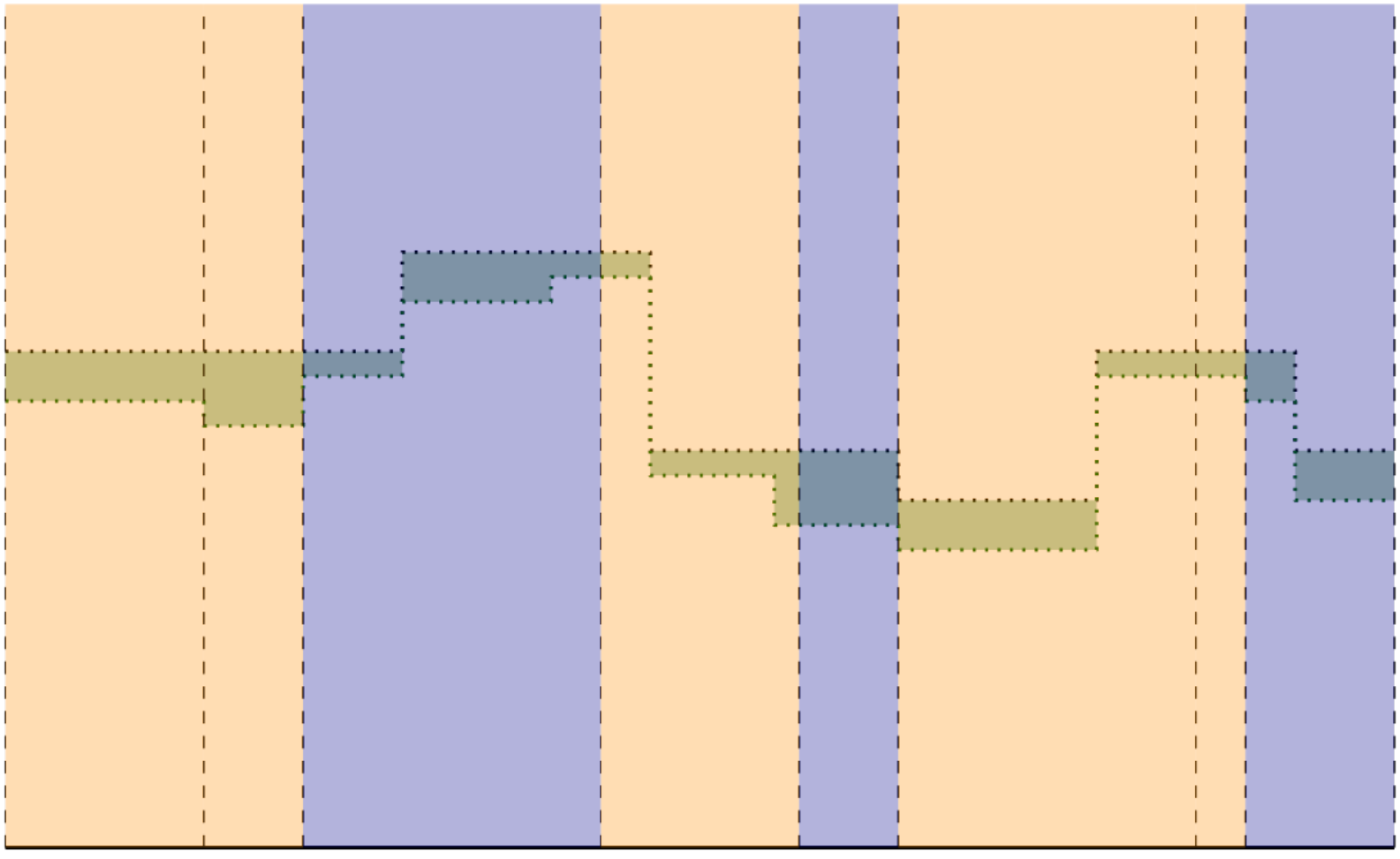
in the sparse case: **DP** that uses slack to "forget" some previously guessed tasks

Some flavors of the PAS



in the dense case: resource augmentation algorithm, with $\delta = \frac{1}{4k}$

Some flavors of the PAS



Real life: both dense and sparse intervals ! (read the paper)

Summary and future questions

Our upper bound: A FPT-$(1+\epsilon)$-approximation for UFP-cover with unit cost.

$\longrightarrow$ a FPT-$(1+\epsilon)$-approximation with task costs ?

Our lower bound: W[1]-hardness

$\longrightarrow$ UFP-cover is FPT if the ***input data*** $\leq$ poly(n) ?

Summary and future questions

Our upper bound: A FPT-$(1+\epsilon)$-approximation for UFP-cover with unit cost.

$\longrightarrow$ a FPT-$(1+\epsilon)$-approximation with task costs ?

Our lower bound: W[1]-hardness

$\longrightarrow$ UFP-cover is FPT if the ***input data*** $\leq$ poly(n) ?