Karlsruhe Institute of Technology

Lifetime Maximization of Monitoring Sensor Networks

Dennis Schieferdecker – *schieferdecker@kit.edu*
Peter Sanders – *sanders@kit.edu*

Institute for Theoretical Informatics - Algorithms II

Introduction

Motivation

- fixed area
- stationary sensor nodes with
  - circular monitoring areas,
  - limited power supply

→ monitor entire region
as long as possible
→ schedule node activation

Introduction
Motivation

- fixed area
- stationary sensor nodes with
  - circular monitoring areas,
  - limited power supply

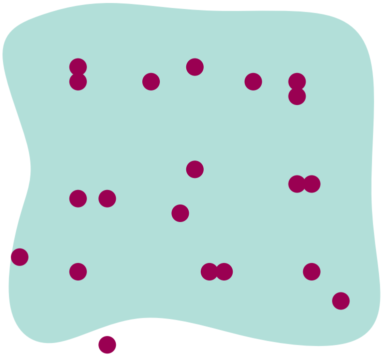
→ monitor entire region
as long as possible
→ schedule node activation

Introduction

Motivation

- fixed area
- stationary sensor nodes with
  - circular monitoring areas,
  - limited power supply

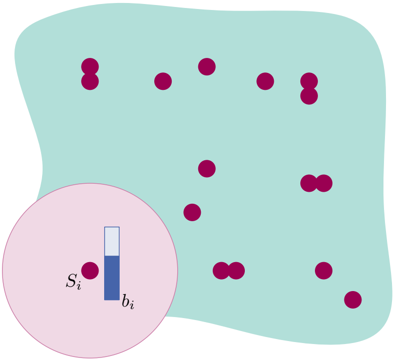
→ monitor entire region
as long as possible
→ schedule node activation



Introduction
Motivation

- fixed area
- stationary sensor nodes with
  - circular monitoring areas,
  - limited power supply

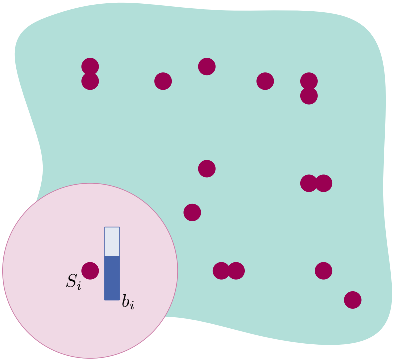
→ monitor entire region
as long as possible
→ schedule node activation



Introduction
Motivation

- fixed area
- stationary sensor nodes with
  - circular monitoring areas,
  - limited power supply

→ monitor entire region
as long as possible
→ schedule node activation

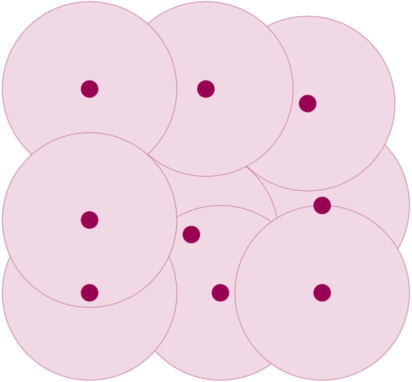


Introduction

Motivation

- fixed area
- stationary sensor nodes with
  - circular monitoring areas,
  - limited power supply

→ monitor entire region
as long as possible
→ schedule node activation

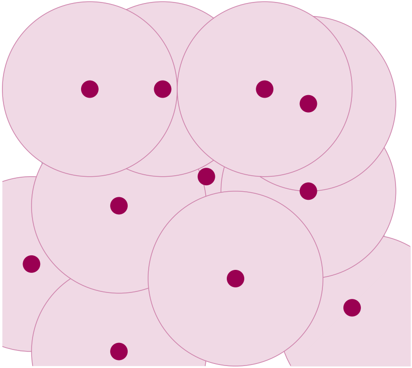


Introduction

Motivation

- fixed area
- stationary sensor nodes with
  - circular monitoring areas,
  - limited power supply

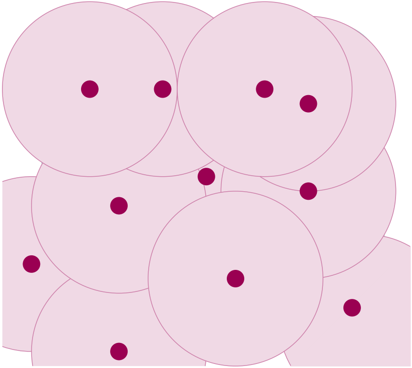
→ monitor entire region
as long as possible
→ schedule node activation



Introduction
Motivation

- fixed area
- stationary sensor nodes with
  - circular monitoring areas,
  - limited power supply

→ monitor entire region
as long as possible
→ schedule node activation



Introduction
Related Work

Extensive Previous Work

[Cardei, Wu 05]	area monitoring equals target monitoring
[Slijepcevic, P. 05]	target monitoring, uniform energy, disjoint sets
[Cardei, Wu 06]	non-disjoint sets, superlinear approximation algorithm
[Berman et al. 06]	general problem, log approximation in superlinear time
[Dhawan et al. 06]	extension to variable sensing ranges
[Luo et al. 09]	exact solver using column generation

Our Contribution

- pseudo-linear time dual approximation scheme
- proof of NP-completeness (see paper)

Introduction

Related Work

Extensive Previous Work

[Cardei, Wu 05]	area monitoring equals target monitoring
[Slijepcevic, P. 05]	target monitoring, uniform energy, disjoint sets
[Cardei, Wu 06]	non-disjoint sets, superlinear approximation algorithm
[Berman et al. 06]	general problem, log approximation in superlinear time
[Dhawan et al. 06]	extension to variable sensing ranges
[Luo et al. 09]	exact solver using column generation

Our Contribution

- pseudo-linear time dual approximation scheme
- proof of NP-completeness (see paper)

Introduction

Related Work

Extensive Previous Work

[Cardei, Wu 05]	area monitoring equals target monitoring
[Slijepcevic, P. 05]	target monitoring, uniform energy, disjoint sets
[Cardei, Wu 06]	non-disjoint sets, superlinear approximation algorithm
[Berman et al. 06]	general problem, log approximation in superlinear time
[Dhawan et al. 06]	extension to variable sensing ranges
[Luo et al. 09]	exact solver using column generation

Our Contribution

- pseudo-linear time dual approximation scheme
- proof of NP-completeness (see paper)

Introduction
Related Work

Extensive Previous Work

[Cardei, Wu 05]	area monitoring equals target monitoring
[Slijepcevic, P. 05]	target monitoring, uniform energy, disjoint sets
[Cardei, Wu 06]	non-disjoint sets, superlinear approximation algorithm
[Berman et al. 06]	general problem, log approximation in superlinear time
[Dhawan et al. 06]	extension to variable sensing ranges
[Luo et al. 09]	exact solver using column generation

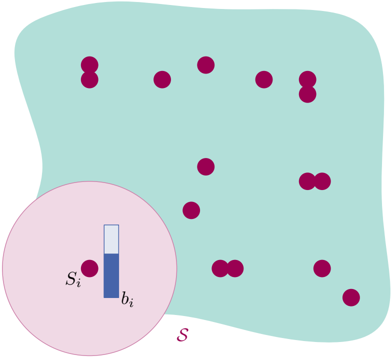
Our Contribution

- pseudo-linear time dual approximation scheme
- proof of NP-completeness (see paper)

Model and Problem Definition

Sensor Network Model

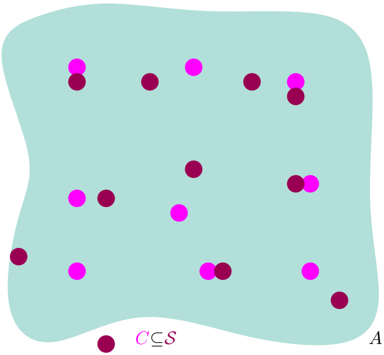
- sensor network $\mathcal{S} = \{S_1, \ldots, S_n\}$ with $S_i = (x_i, y_i, b_i)$
  - (x_i, y_i): coordinates
  - b_i: battery capacity
- $C \subseteq \mathcal{S}$ is a cover of area A,
  - if the union of disks centered at each $S \in C$ contains area A
  - disk radii equal sensing range R
- $\mathcal{C}$ set of all possible covers of A



Model and Problem Definition

Sensor Network Model

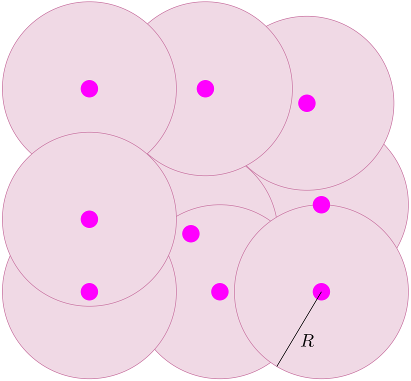
- sensor network $\mathcal{S} = \{S_1, \ldots, S_n\}$ with $S_i = (x_i, y_i, b_i)$
  - (x_i, y_i): coordinates
  - b_i: battery capacity
- $C \subseteq \mathcal{S}$ is a cover of area A,
  - if the union of disks centered at each $S \in C$ contains area A
  - disk radii equal sensing range R
- $\mathcal{C}$ set of all possible covers of A



Model and Problem Definition

Sensor Network Model

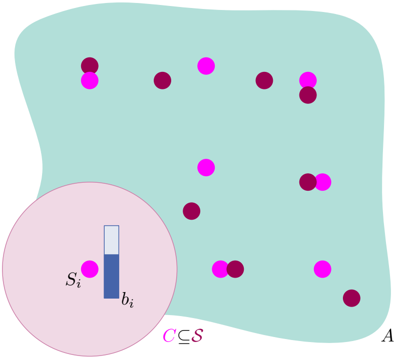
- sensor network $\mathcal{S} = \{S_1, \ldots, S_n\}$ with $S_i = (x_i, y_i, b_i)$
  - (x_i, y_i): coordinates
  - b_i: battery capacity
- $C \subseteq \mathcal{S}$ is a cover of area A,
  - if the union of disks centered at each $S \in C$ contains area A
  - disk radii equal sensing range R
- $\mathcal{C}$ set of all possible covers of A



Model and Problem Definition

Problem Definition

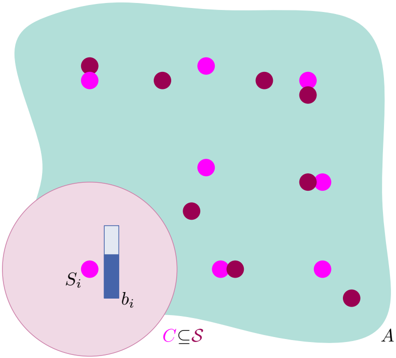
- find schedule $(\underline{C}, \underline{t})$,
  - covers $\underline{C} = \{C_1, \ldots, C_m\} \subseteq \mathcal{C}$
  - durations $\underline{t} = \{t_1, \ldots, t_m\}$
- maximizing lifetime $T = \sum_{j=1}^{m} t_j$
  - s.t. $\sum_{i:S_j \in C_i} t_i \leq b_j \; \forall \; S_j \in \mathcal{S}$ (1)
  - i.e. node S_j cannot consume more than b_j units of energy
- problem instance $(\mathcal{S}, A, R)$
  - solution is any schedule $(\underline{C}, \underline{t})$
  - solution $(\underline{C}, \underline{t})$ feasible if (1) holds
  - lifetime of a solution $T\langle \mathcal{S}, A, R \rangle$



Model and Problem Definition

Problem Definition

- find schedule $(\underline{C}, \underline{t})$,
  - covers $\underline{C} = \{C_1, \ldots, C_m\} \subseteq \mathcal{C}$
  - durations $\underline{t} = \{t_1, \ldots, t_m\}$
- maximizing lifetime $T = \sum_{j=1}^{m} t_j$
  - s.t. $\sum_{i:S_j \in C_i} t_i \leq b_j \ \forall \ S_j \in \mathcal{S}$ (1)
  - i.e. node S_j cannot consume more than b_j units of energy

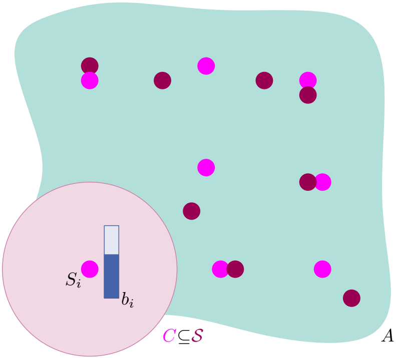


- problem instance $(\mathcal{S}, A, R)$
  - solution is any schedule $(\underline{C}, \underline{t})$
  - solution $(\underline{C}, \underline{t})$ feasible if (1) holds
  - lifetime of a solution $T\langle \mathcal{S}, A, R\rangle$

Model and Problem Definition

Problem Definition

- find schedule $(\underline{C}, \underline{t})$,
  - covers $\underline{C} = \{C_1, \dots, C_m\} \subseteq \mathcal{C}$
  - durations $\underline{t} = \{t_1, \dots, t_m\}$
- maximizing lifetime $T = \sum_{j=1}^{m} t_j$
  - s.t. $\sum_{i:S_j \in C_i} t_i \leq b_j \; \forall \; S_j \in \mathcal{S}$ (1)
  - i.e. node S_j cannot consume more than b_j units of energy



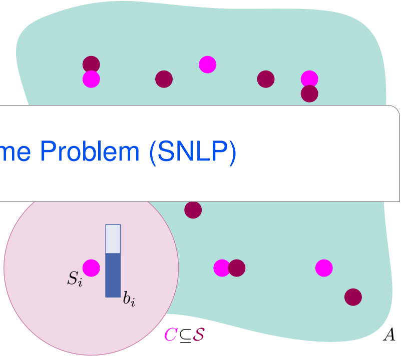
- problem instance $(\mathcal{S}, A, 1)$
  - solution is any schedule $(\underline{C}, \underline{t})$
  - solution $(\underline{C}, \underline{t})$ feasible if (1) holds
  - lifetime of a solution $T\langle \mathcal{S}, A, 1 \rangle$

Model and Problem Definition

Problem Definition

- find schedule $(\underline{C}, \underline{t})$,
  - covers $\underline{C} = \{C_1, \ldots, C_m\} \subseteq \mathcal{C}$
  - durations $\underline{t} = \{t_1, \ldots, t_m\}$
- maximizing lifetime $T = \sum_{j=1}^{m} t_j$

Sensor Network Lifetime Problem (SNLP)

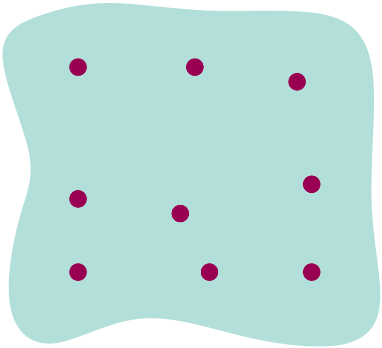


- problem instance $(\mathcal{S}, A, 1)$
  - solution is any schedule $(\underline{C}, \underline{t})$
  - solution $(\underline{C}, \underline{t})$ feasible if (1) holds
  - lifetime of a solution $T\langle \mathcal{S}, A, 1\rangle$

Model and Problem Definition

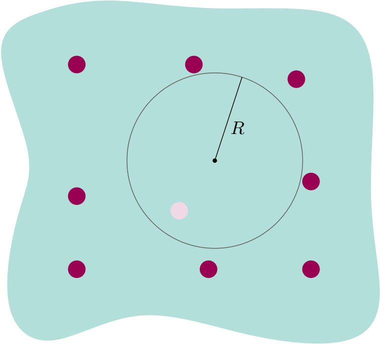
Further Considerations

- problem reinterpretation
  - area monitoring with min. guaranteed resolution
  - resolution
    ≙ max. distance of any point in the area to one sensor
    ≙ sensing range
- central algorithm sufficient
  - stationary problem
  - providing upper bounds for distributed algorithms



Model and Problem Definition
Further Considerations

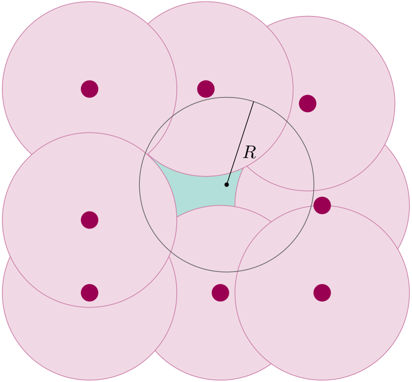
- problem reinterpretation
  - area monitoring with min. guaranteed resolution
  - resolution
    $\hat{=}$ max. distance of any point in the area to one sensor
    $\hat{=}$ sensing range
- central algorithm sufficient
  - stationary problem
  - providing upper bounds for distributed algorithms



Model and Problem Definition

Further Considerations

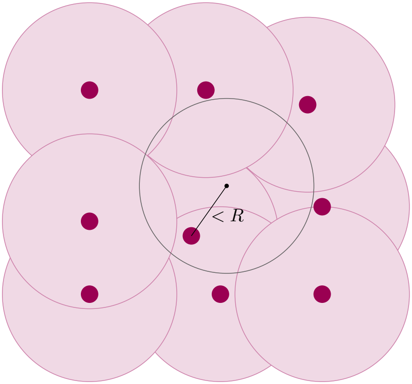
- problem reinterpretation
  - area monitoring with min. guaranteed resolution
  - resolution
    $\hat{=}$ max. distance of any point in the area to one sensor
    $\hat{=}$ sensing range
- central algorithm sufficient
  - stationary problem
  - providing upper bounds for distributed algorithms



Model and Problem Definition

Further Considerations

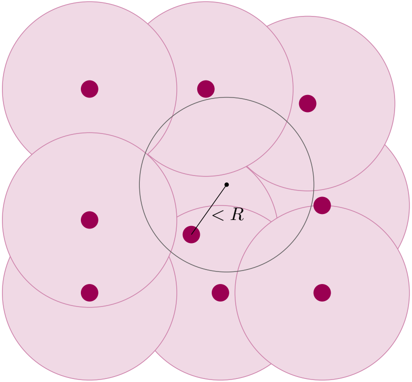
- problem reinterpretation
  - area monitoring with min. guaranteed resolution
  - resolution
    $\hat{=}$ max. distance of any point in the area to one sensor
    $\hat{=}$ sensing range
- central algorithm sufficient
  - stationary problem
  - providing upper bounds for distributed algorithms



Model and Problem Definition

Further Considerations

- problem reinterpretation
  - area monitoring with min. guaranteed resolution
  - resolution
    $\widehat{=}$ max. distance of any point in the area to one sensor
    $\widehat{=}$ sensing range



- central algorithm sufficient
  - stationary problem
  - providing upper bounds for distributed algorithms

Algorithm Outline

- combination of two approximation techniques
  - consider simpler problem instances
  - assume f-approximate algorithm $\mathcal{A}$ exists to solve these instances
  - transform problem instances so that $\mathcal{A}$ can solve them
  - solutions are feasible for the original instance and near-optimal

- discretizing positions
  - nodes restricted to points on a grid

- area partitioning
  - area is divided into squared areas
  - subproblems restrained to these squares are solved and combined

Algorithm Outline

- combination of two approximation techniques
  - consider simpler problem instances
  - assume f-approximate algorithm $\mathcal{A}$ exists to solve these instances
  - transform problem instances so that $\mathcal{A}$ can solve them
  - solutions are feasible for the original instance and near-optimal
- discretizing positions
  - nodes restricted to points on a grid
- area partitioning
  - area is divided into squared areas
  - subproblems restrained to these squares are solved and combined

Algorithm Outline

- combination of two approximation techniques
  - consider simpler problem instances
  - assume f-approximate algorithm $\mathcal{A}$ exists to solve these instances
  - transform problem instances so that $\mathcal{A}$ can solve them
  - solutions are feasible for the original instance and near-optimal
- discretizing positions
  - nodes restricted to points on a grid
- area partitioning
  - area is divided into squared areas
  - subproblems restrained to these squares are solved and combined

Discretizing Positions

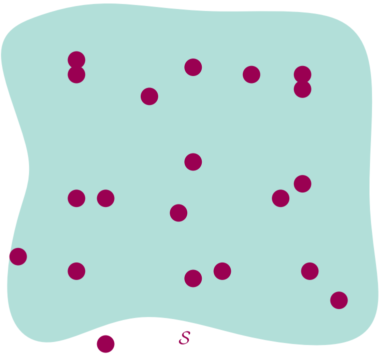
Procedure

Algorithm 1

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\delta \in [0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define grid of width $\delta/2$.
- Move every node in $\mathcal{S}$ to the closest point on the grid $\rightarrow \tilde{\mathcal{S}}$.
- Solve $(\tilde{\mathcal{S}}, 1 + \delta/2) \rightarrow (\underline{C}, \underline{t})$. (using algorithm $\mathcal{A}$)
- Return $(\underline{C}, \underline{t})$.



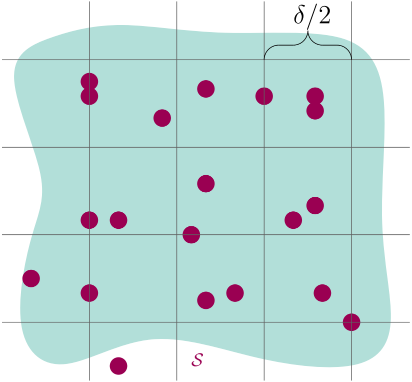
Discretizing Positions

Procedure

Algorithm 1

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\delta \in [0, 1]$
out: schedule $(\underline{C}, \underline{t})$

- Define grid of width $\delta/2$.
- Move every node in $\mathcal{S}$ to the closest point on the grid $\rightarrow \tilde{\mathcal{S}}$.
- Solve $(\tilde{\mathcal{S}}, 1 + \delta/2) \rightarrow (\underline{C}, \underline{t})$. (using algorithm $\mathcal{A}$)
- Return $(\underline{C}, \underline{t})$.



Discretizing Positions

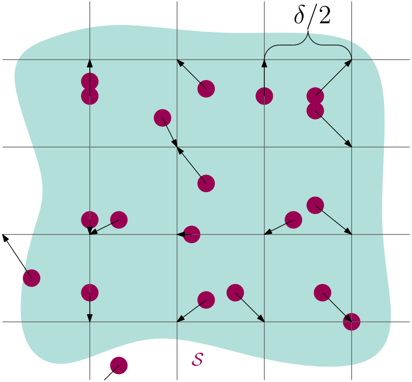
Procedure

Algorithm 1

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\delta \in [0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define grid of width $\delta/2$.
- Move every node in $\mathcal{S}$ to the closest point on the grid $\rightarrow \tilde{\mathcal{S}}$.
- Solve $(\tilde{\mathcal{S}}, 1 + \delta/2) \rightarrow (\underline{C}, \underline{t})$. (using algorithm $\mathcal{A}$)
- Return $(\underline{C}, \underline{t})$.



Discretizing Positions

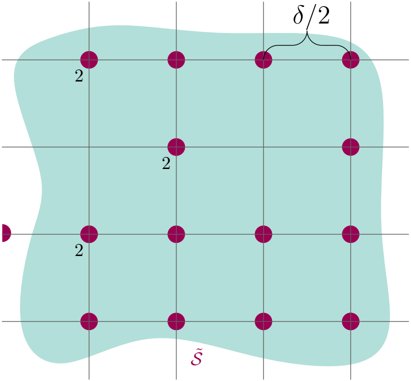
Procedure

Algorithm 1

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\delta \in [0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define grid of width $\delta/2$.
- Move every node in $\mathcal{S}$ to the closest point on the grid $\to \tilde{\mathcal{S}}$.
- Solve $(\tilde{\mathcal{S}}, 1 + \delta/2) \to (\underline{C}, \underline{t})$. (using algorithm $\mathcal{A}$)
- Return $(\underline{C}, \underline{t})$.



Discretizing Positions

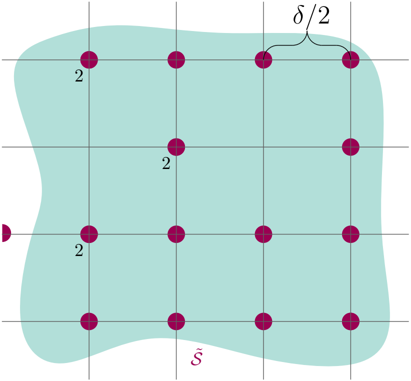
Procedure

Algorithm 1

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\delta \in [0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define grid of width $\delta/2$.
- Move every node in $\mathcal{S}$ to the closest point on the grid $\rightarrow \tilde{\mathcal{S}}$.
- Solve $(\tilde{\mathcal{S}}, 1 + \delta/2) \rightarrow (\underline{C}, \underline{t})$. (using algorithm $\mathcal{A}$)
- Return $(\underline{C}, \underline{t})$.



Discretizing Positions

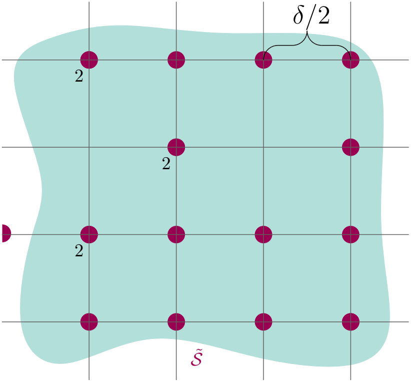
Procedure

Algorithm 1

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\delta \in [0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define grid of width $\delta/2$.
- Move every node in $\mathcal{S}$ to the closest point on the grid $\rightarrow \tilde{\mathcal{S}}$.
- Solve $(\tilde{\mathcal{S}}, 1 + \delta/2) \rightarrow (\underline{C}, \underline{t})$. (using algorithm $\mathcal{A}$)
- Return $(\underline{C}, \underline{t})$.



Discretizing Positions

Proofs

Lemma 1

Let $\delta \in [0, 1]$. *Algorithm 1* yields a feasible solution to $(\mathcal{S}, 1 + \delta)$ with lifetime $T\langle \mathcal{S}, 1 + \delta \rangle \geq f \cdot T_{\text{opt}}\langle \mathcal{S}, 1 \rangle$.

Correctness

- solution to $(S, 1)$ is solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$
  $\rightarrow T_{\text{opt}}\langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq T_{\text{opt}}\langle \mathcal{S}, 1 \rangle$
- solving $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ yields $T\langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle$
  $\rightarrow T\langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq f \cdot T_{\text{opt}}\langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq f \cdot T_{\text{opt}}\langle \mathcal{S}, 1 \rangle$
- solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ is solution to $(\mathcal{S}, 1 + \delta)$
  $\rightarrow T\langle \mathcal{S}, 1 + \delta \rangle \geq T\langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq f \cdot T_{\text{opt}}\langle \mathcal{S}, 1 \rangle$

Discretizing Positions

Proofs

Lemma 1

Let $\delta \in [0, 1]$. *Algorithm 1* yields a feasible solution to $(\mathcal{S}, 1 + \delta)$ with lifetime $T\langle\mathcal{S}, 1 + \delta\rangle \geq f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$.

Correctness

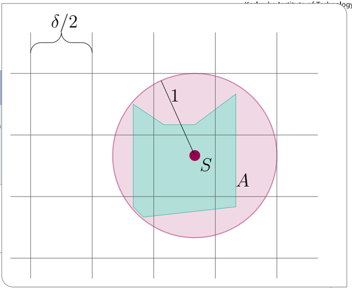
- solution to $(S, 1)$ is solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$
  $\rightarrow T_{\text{opt}}\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle \geq T_{\text{opt}}\langle\mathcal{S}, 1\rangle$
- solving $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ yields $T\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle$
  $\rightarrow T\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle \geq f \cdot T_{\text{opt}}\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle \geq f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$
- solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ is solution to $(\mathcal{S}, 1 + \delta)$
  $\rightarrow T\langle\mathcal{S}, 1 + \delta\rangle \geq T\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle \geq f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$

Discretizing Positions

Proofs

Lemma 1

Let $\delta \in [0,1]$. *Algorithm 1* yields a feasible
lifetime $T\langle \mathcal{S}, 1+\delta \rangle \geq f \cdot T_{\mathrm{opt}}\langle \mathcal{S}, 1 \rangle$.



Correctness

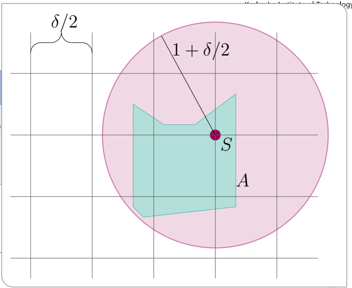
- solution to $(S, 1)$ is solution to $(\tilde{\mathcal{S}}, 1+\frac{\delta}{2})$
  $\rightarrow T_{\mathrm{opt}}\langle \tilde{\mathcal{S}}, 1+\frac{\delta}{2} \rangle \geq T_{\mathrm{opt}}\langle \mathcal{S}, 1 \rangle$
- solving $(\tilde{\mathcal{S}}, 1+\frac{\delta}{2})$ yields $T\langle \tilde{\mathcal{S}}, 1+\frac{\delta}{2} \rangle$
  $\rightarrow T\langle \tilde{\mathcal{S}}, 1+\frac{\delta}{2} \rangle \geq f \cdot T_{\mathrm{opt}}\langle \tilde{\mathcal{S}}, 1+\frac{\delta}{2} \rangle \geq f \cdot T_{\mathrm{opt}}\langle \mathcal{S}, 1 \rangle$
- solution to $(\tilde{\mathcal{S}}, 1+\frac{\delta}{2})$ is solution to $(\mathcal{S}, 1+\delta)$
  $\rightarrow T\langle \mathcal{S}, 1+\delta \rangle \geq T\langle \tilde{\mathcal{S}}, 1+\frac{\delta}{2} \rangle \geq f \cdot T_{\mathrm{opt}}\langle \mathcal{S}, 1 \rangle$

Discretizing Positions

Proofs

Lemma 1

Let $\delta \in [0, 1]$. *Algorithm 1* yields a feasible
lifetime $T\langle\mathcal{S}, 1+\delta\rangle \geq f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$.



Correctness

- solution to $(S, 1)$ is solution to $(\tilde{\mathcal{S}}, 1+\frac{\delta}{2})$
  $\rightarrow T_{\text{opt}}\langle\tilde{\mathcal{S}}, 1+\frac{\delta}{2}\rangle \geq T_{\text{opt}}\langle\mathcal{S}, 1\rangle$
- solving $(\tilde{\mathcal{S}}, 1+\frac{\delta}{2})$ yields $T\langle\tilde{\mathcal{S}}, 1+\frac{\delta}{2}\rangle$
  $\rightarrow T\langle\tilde{\mathcal{S}}, 1+\frac{\delta}{2}\rangle \geq f \cdot T_{\text{opt}}\langle\tilde{\mathcal{S}}, 1+\frac{\delta}{2}\rangle \geq f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$
- solution to $(\tilde{\mathcal{S}}, 1+\frac{\delta}{2})$ is solution to $(\mathcal{S}, 1+\delta)$
  $\rightarrow T\langle\mathcal{S}, 1+\delta\rangle \geq T\langle\tilde{\mathcal{S}}, 1+\frac{\delta}{2}\rangle \geq f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$

Discretizing Positions

Proofs

Lemma 1

Let $\delta \in [0, 1]$. *Algorithm 1* yields a feasible solution to $(\mathcal{S}, 1 + \delta)$ with lifetime $T\langle \mathcal{S}, 1 + \delta \rangle \geq f \cdot T_{\text{opt}} \langle \mathcal{S}, 1 \rangle$.

Correctness

- solution to $(S, 1)$ is solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$
  $\rightarrow T_{\text{opt}} \langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq T_{\text{opt}} \langle \mathcal{S}, 1 \rangle$
- solving $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ yields $T\langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle$
  $\rightarrow T\langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq f \cdot T_{\text{opt}} \langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq f \cdot T_{\text{opt}} \langle \mathcal{S}, 1 \rangle$
- solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ is solution to $(\mathcal{S}, 1 + \delta)$
  $\rightarrow T\langle \mathcal{S}, 1 + \delta \rangle \geq T\langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq f \cdot T_{\text{opt}} \langle \mathcal{S}, 1 \rangle$

Discretizing Positions

Proofs

Lemma 1

Let $\delta \in [0, 1]$. *Algorithm 1* yields a feasible solution to $(\mathcal{S}, 1 + \delta)$ with lifetime $T\langle \mathcal{S}, 1 + \delta \rangle \geq f \cdot T_{\mathrm{opt}} \langle \mathcal{S}, 1 \rangle$.

Correctness

- solution to $(S, 1)$ is solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$
  $\rightarrow T_{\mathrm{opt}} \langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq T_{\mathrm{opt}} \langle \mathcal{S}, 1 \rangle$
- solving $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ yields $T \langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle$
  $\rightarrow T \langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq f \cdot T_{\mathrm{opt}} \langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq f \cdot T_{\mathrm{opt}} \langle \mathcal{S}, 1 \rangle$
- solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ is solution to $(\mathcal{S}, 1 + \delta)$
  $\rightarrow T \langle \mathcal{S}, 1 + \delta \rangle \geq T \langle \tilde{\mathcal{S}}, 1 + \frac{\delta}{2} \rangle \geq f \cdot T_{\mathrm{opt}} \langle \mathcal{S}, 1 \rangle$

Discretizing Positions

Proofs

Lemma 1

Let $\delta \in [0, 1]$. *Algorithm 1* yields a feasible solution to $(\mathcal{S}, 1 + \delta)$ with lifetime $T\langle\mathcal{S}, 1 + \delta\rangle \geq f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$.

Correctness

- solution to $(S, 1)$ is solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$
  $\rightarrow T_{\text{opt}}\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle \geq T_{\text{opt}}\langle\mathcal{S}, 1\rangle$
- solving $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ yields $T\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle$
  $\rightarrow T\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle \geq f \cdot T_{\text{opt}}\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle \geq f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$
- solution to $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ is solution to $(\mathcal{S}, 1 + \delta)$
  $\rightarrow T\langle\mathcal{S}, 1 + \delta\rangle \geq T\langle\tilde{\mathcal{S}}, 1 + \frac{\delta}{2}\rangle \geq f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$

Area Partitioning

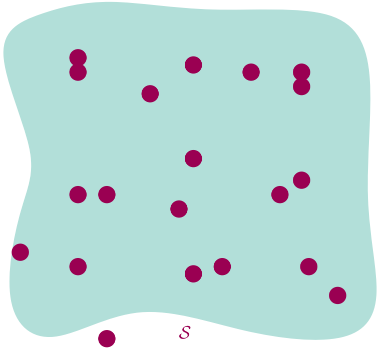
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

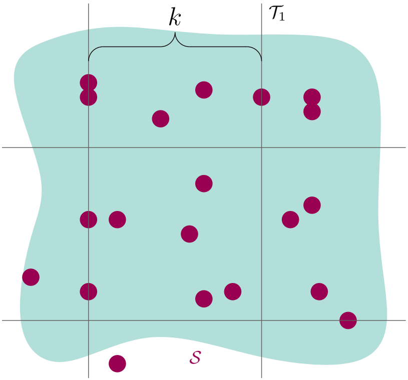
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$,
parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$,
  $k = \lceil \frac{10}{\epsilon} \rceil$, $i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return
  $$(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right).$$



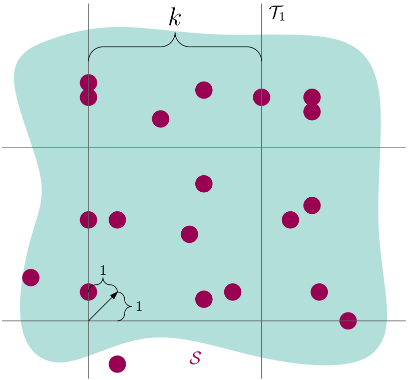
Area Partitioning

Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$
out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil$, $i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



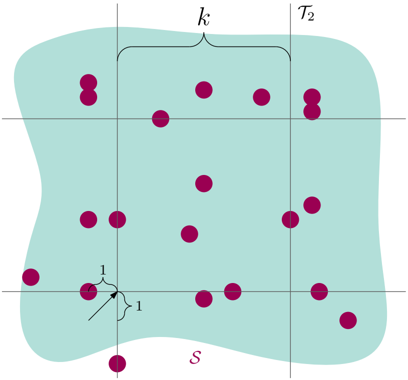
Area Partitioning

Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$
out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\to (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

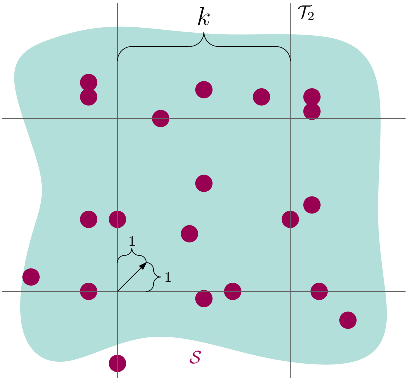
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

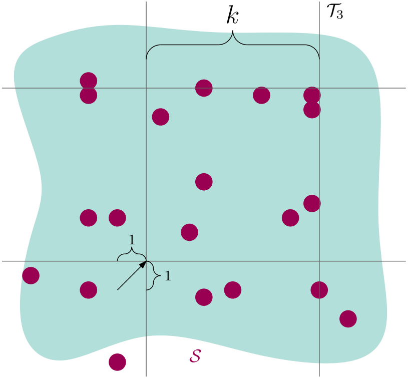
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

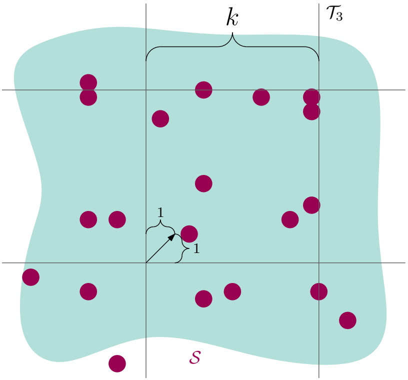
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

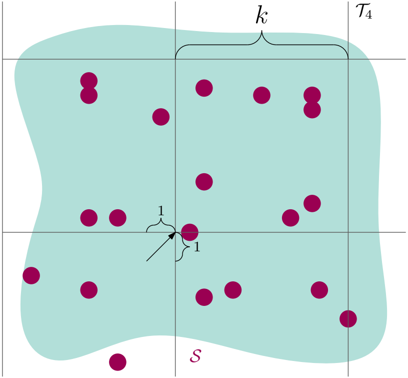
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$,
parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



Area Partitioning

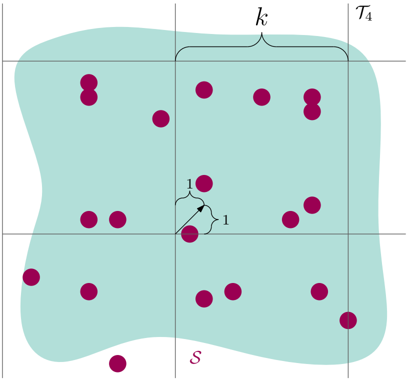
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\to (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

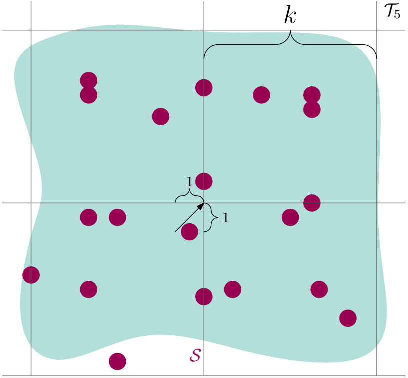
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\to (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



Area Partitioning

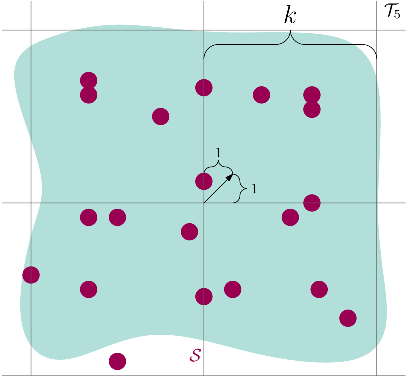
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\to (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



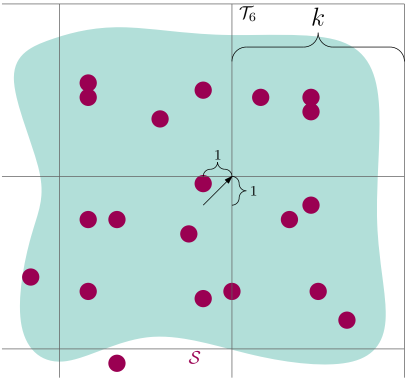
Area Partitioning

Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$
out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



Area Partitioning

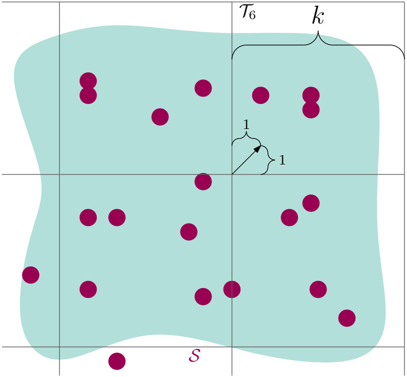
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

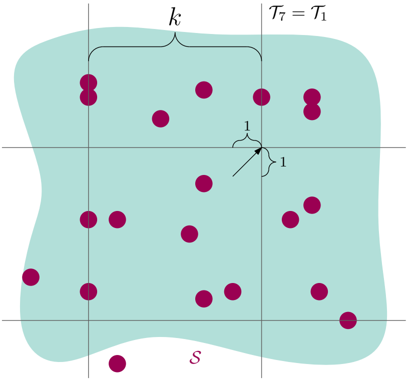
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



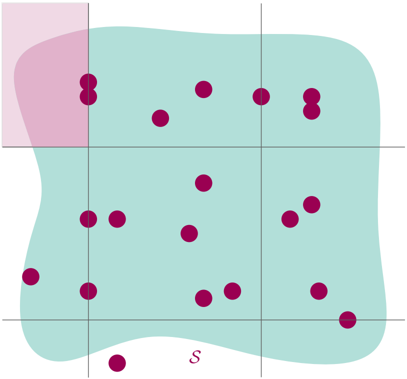
Area Partitioning
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$
out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return

$$(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right).$$



Area Partitioning

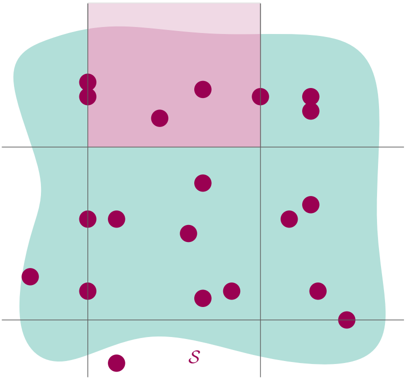
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil$, $i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

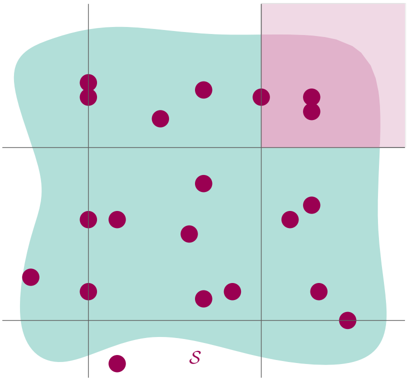
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$,
parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$,
  $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return
  $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



Area Partitioning

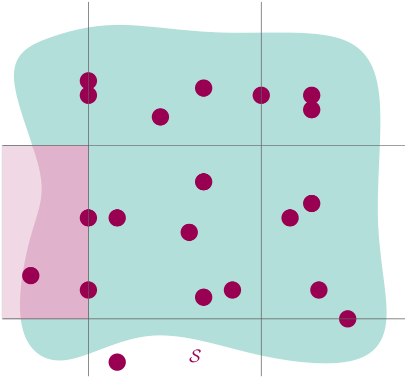
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

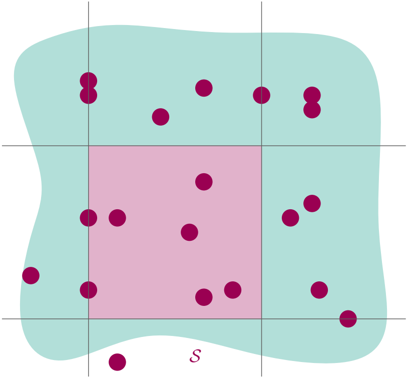
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



Area Partitioning

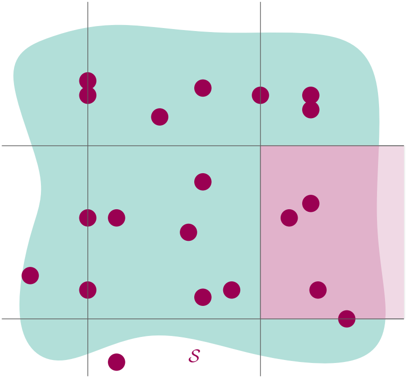
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



Area Partitioning

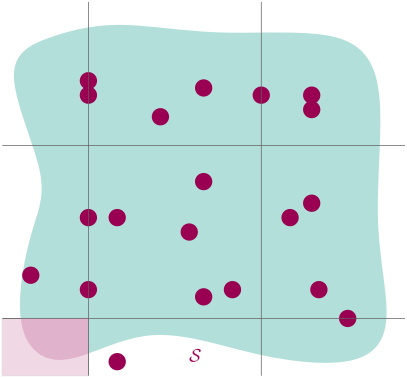
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

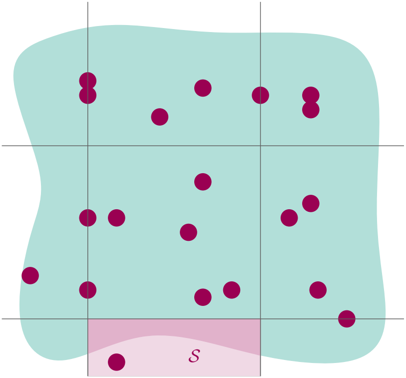
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



Area Partitioning

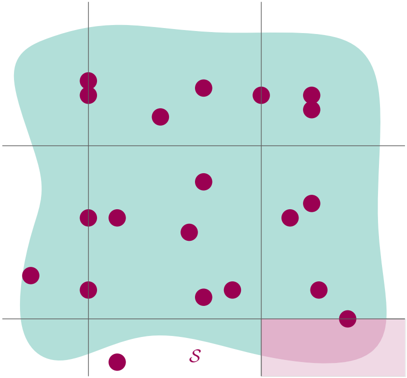
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

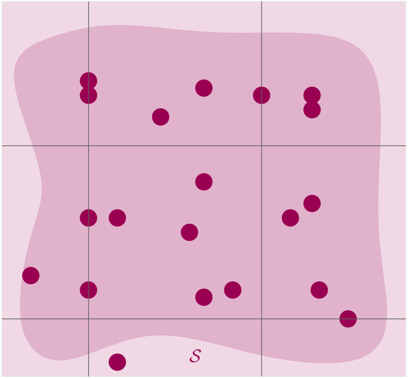
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.



Area Partitioning

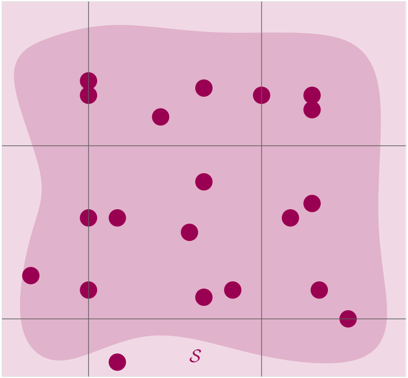
Procedure

Algorithm 2

in: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameter $\epsilon \in (0, 1]$

out: schedule $(\underline{C}, \underline{t})$

- Define k partitions $\mathcal{T}^i$, $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1)$ restricted to each square of $\mathcal{T}^i$, (with algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.



Area Partitioning

Proofs

Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solution to $(\mathcal{S}, 1)$ with lifetime $T\langle\mathcal{S}, 1\rangle \geq f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle\mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle\mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle = f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$$

Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

Area Partitioning

Proofs

Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solution to $(\mathcal{S}, 1)$ with lifetime $T\langle\mathcal{S}, 1\rangle \geq f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle\mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle\mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$$
  $$= f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$$

Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

Area Partitioning

Proofs

Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solution to $(\mathcal{S}, 1)$ with lifetime $T\langle\mathcal{S}, 1\rangle \geq f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle\mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle\mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle = f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$$

Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4\Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning

Proofs

Lemma 2

Let $\epsilon \in (0,1]$. *Algorithm 2* yields a feasible solution to $(\mathcal{S},1)$ with lifetime $T\langle\mathcal{S},1\rangle \geq f\cdot(1-\epsilon)\cdot T_{\text{opt}}\langle\mathcal{S},1\rangle$.

Lifetime

- $$T\langle\mathcal{S},1\rangle = \frac{1-\epsilon}{k}\sum_{i\in\mathbb{Z}_k} T\langle\mathcal{S},1\rangle^i \geq \frac{1-\epsilon}{k}\sum_{i\in\mathbb{Z}_k} f\cdot T_{\text{opt}}\langle\mathcal{S},1\rangle = f\cdot(1-\epsilon)\cdot T_{\text{opt}}\langle\mathcal{S},1\rangle$$

Feasibililty

- $$\frac{1-\epsilon}{k}\sum_{i\in\mathbb{Z}_k}\sum_{j:S_n\in C_j^i} t_j^i \leq \frac{1-\epsilon}{k}\Big((k-2)\cdot 1 + 2\cdot 4\Big)\cdot b_n \leq b_n \quad \forall S_n\in\mathcal{S}.$$

$$T\langle\mathcal{S},1\rangle^i \mathrel{\hat{=}} \sum_{j=1}^{|C^i|} t_j^i$$

$$(\underline{C},\underline{t}) = \Big(\bigcup_{i\in\mathbb{Z}_k}\underline{C}^i, \frac{(1-\epsilon)}{k}\cdot\bigcup_{i\in\mathbb{Z}_k}\underline{t}^i\Big)$$

Area Partitioning

Proofs

Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solution to $(\mathcal{S}, 1)$ with lifetime $T\langle\mathcal{S}, 1\rangle \geq f \cdot (1-\epsilon) \cdot T_{\mathrm{opt}}\langle\mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle\mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i\in\mathbb{Z}_k} T\langle\mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i\in\mathbb{Z}_k} f \cdot T_{\mathrm{opt}}\langle\mathcal{S}, 1\rangle = f \cdot (1-\epsilon) \cdot T_{\mathrm{opt}}\langle\mathcal{S}, 1\rangle$$

$$T\langle\mathcal{S}, 1\rangle^i \geq f \cdot T_{\mathrm{opt}}\langle\mathcal{S}, 1\rangle$$

Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i\in\mathbb{Z}_k} \sum_{j:S_n\in C_j^i} t_j^i \leq \frac{1-\epsilon}{k}\Big((k-2)\cdot 1 + 2\cdot 4\Big)\cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i\in\mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i\in\mathbb{Z}_k} \underline{t}^i\Big)$$

Area Partitioning

Proofs

Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solution to $(\mathcal{S}, 1)$ with lifetime $T\langle\mathcal{S}, 1\rangle \geq f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle\mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle\mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle = f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$$

Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4\Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

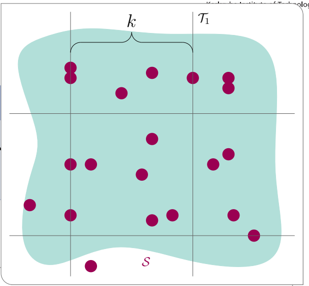
$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning

Proofs

Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solu
me $T\langle \mathcal{S}, 1\rangle \geq f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$.



Lifetime

- $$T\langle \mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle \mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$$
$$= f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$$

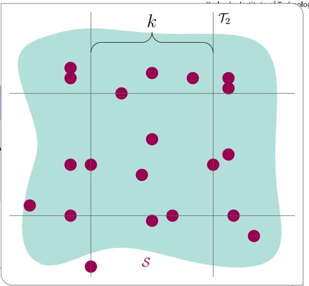
Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning

Proofs



Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solu
me $T\langle\mathcal{S}, 1\rangle \geq f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle\mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle\mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle = f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle\mathcal{S}, 1\rangle$$

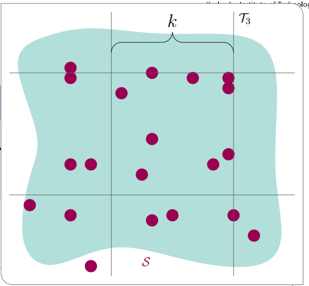
Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4\Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning

Proofs



Lemma 2

Let $\epsilon \in (0,1]$. *Algorithm 2* yields a feasible solu
me $T\langle \mathcal{S}, 1\rangle \geq f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle \mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle \mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$$
  $$= f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$$

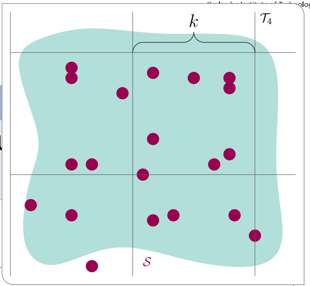
Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning

Proofs



Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solu
me $T\langle \mathcal{S}, 1\rangle \geq f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle \mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle \mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle = f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$$

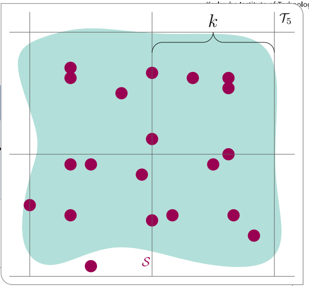
Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j: S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning

Proofs



Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solu
me $T\langle \mathcal{S}, 1\rangle \geq f \cdot (1 - \epsilon) \cdot T_{\mathrm{opt}}\langle \mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle \mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle \mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\mathrm{opt}}\langle \mathcal{S}, 1\rangle$$
  $$= f \cdot (1 - \epsilon) \cdot T_{\mathrm{opt}}\langle \mathcal{S}, 1\rangle$$

Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

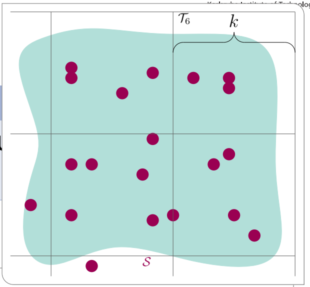
$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning

Proofs

Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solu
me $T\langle \mathcal{S}, 1\rangle \geq f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$.



Lifetime

- $$T\langle \mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle \mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle = f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$$

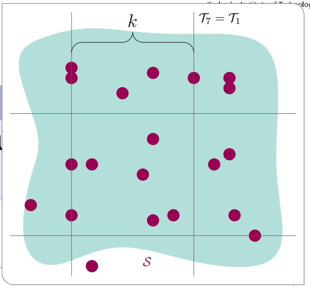
Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning

Proofs



Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solu
me $T\langle \mathcal{S}, 1\rangle \geq f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle \mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle \mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$$
  $$= f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$$

Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

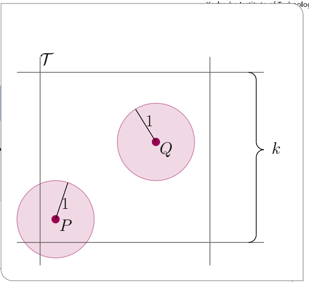
$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning

Proofs

Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solu
me $T\langle \mathcal{S}, 1\rangle \geq f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$.



Lifetime

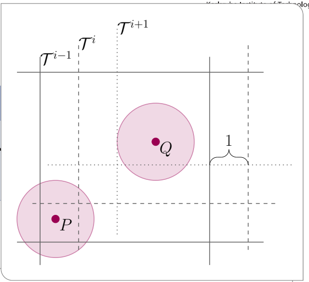
- $T\langle \mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle \mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$
  $= f \cdot (1 - \epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$

Feasibililty

- $\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$

$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Area Partitioning
Proofs



Lemma 2

Let $\epsilon \in (0, 1]$. *Algorithm 2* yields a feasible solu
me $T\langle \mathcal{S}, 1\rangle \geq f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$.

Lifetime

- $$T\langle \mathcal{S}, 1\rangle = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} T\langle \mathcal{S}, 1\rangle^i \geq \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} f \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle = f \cdot (1-\epsilon) \cdot T_{\text{opt}}\langle \mathcal{S}, 1\rangle$$

Feasibililty

- $$\frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \sum_{j : S_n \in C_j^i} t_j^i \leq \frac{1-\epsilon}{k} \Big((k-2) \cdot 1 + 2 \cdot 4 \Big) \cdot b_n \leq b_n \quad \forall S_n \in \mathcal{S}.$$

$$(\underline{C}, \underline{t}) = \Big(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \Big)$$

Complete Approximation Algorithm

Procedure

Complete Algorithm:

input: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameters $\delta \in [0, 1], \epsilon \in (0, 1]$
output: schedule $(\underline{C}, \underline{t})$

- Define grid of width $\delta/2$.
- Move every node in $\mathcal{S}$ to the closest point on the grid $\rightarrow \tilde{\mathcal{S}}$.
- Define k partitions $\mathcal{T}^i$,
  $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ for each square of $\mathcal{T}^i$, (using algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.

Complete Approximation Algorithm

Procedure

Complete Algorithm:

input: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameters $\delta \in [0, 1], \epsilon \in (0, 1]$
output: schedule $(\underline{C}, \underline{t})$

- Define grid of width $\delta/2$.
- Move every node in $\mathcal{S}$ to the closest point on the grid $\rightarrow \tilde{\mathcal{S}}$.
- Define k partitions $\mathcal{T}^i$,
  $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\mathcal{S}, 1 \quad)$ for each square of $\mathcal{T}^i$, (using algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = (\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i)$.

Algorithm 2

Complete Approximation Algorithm

Procedure

Complete Algorithm:

input: instance $(\mathcal{S}, 1)$, algorithm $\mathcal{A}$, parameters $\delta \in [0,1], \epsilon \in (0,1]$
output: schedule $(\underline{C}, \underline{t})$

- Define grid of width $\delta/2$.
- Move every node in $\mathcal{S}$ to the closest point on the grid $\rightarrow \tilde{\mathcal{S}}$.
- Define k partitions $\mathcal{T}^i$,
  $k = \lceil \frac{10}{\epsilon} \rceil, i \in \mathbb{Z}_k = \{1, ..., k\}$.
- For each partition $\mathcal{T}^i$,
  - solve $(\tilde{\mathcal{S}}, 1 + \frac{\delta}{2})$ for each square of $\mathcal{T}^i$, (using algorithm $\mathcal{A}$)
  - combine solutions $\rightarrow (\underline{C}^i, \underline{t}^i)$.
- Return $(\underline{C}, \underline{t}) = \left(\bigcup_{i \in \mathbb{Z}_k} \underline{C}^i, \frac{(1-\epsilon)}{k} \cdot \bigcup_{i \in \mathbb{Z}_k} \underline{t}^i \right)$.

Algorithm 1

Complete Approximation Algorithm

Theorem

Theorem 1

Let $\delta \in [0,1]$ and $k = \lceil 10/\epsilon \rceil$ with $\epsilon \in (0,1]$. *Complete Algorithm* yields a feasible solution $(\underline{C}, \underline{t})$ to $(\mathcal{S}, 1+\delta)$ with lifetime

$$T\langle \mathcal{S}, 1+\delta \rangle \geq (1-\epsilon) \cdot f \cdot T_{\mathrm{opt}}\langle \mathcal{S}, 1 \rangle.$$

Its runtime complexity is bounded by

$$O\Big(|\mathcal{S}| + \epsilon|\mathcal{S}| \cdot g_{\mathcal{A}}(O(1/\delta^2\epsilon^2))\Big) = O(|\mathcal{S}|)$$

with $g_{\mathcal{A}}(|\mathcal{S}|)$ the runtime of algorithm $\mathcal{A}$.

Complete Approximation Algorithm

Proofs - 1

Theorem 1 - (part a)

Let $\delta \in [0,1]$ and $k = \lceil 10/\epsilon \rceil$ with $\epsilon \in (0,1]$. *Complete Algorithm* yields a feasible solution $(\underline{C}, \underline{t})$ to $(\mathcal{S}, 1+\delta)$ with lifetime

$$T\langle \mathcal{S}, 1+\delta \rangle \geq (1-\epsilon) \cdot f \cdot T_{\text{opt}}\langle \mathcal{S}, 1 \rangle.$$

Feasibility

- follows directly from Lemma 1 & 2

Approximation Guarantee

- node discretization $\rightarrow T\langle \mathcal{S}, 1+\delta \rangle \geq f \cdot T_{\text{opt}}\langle \mathcal{S}, 1 \rangle$ for all squares
- combining solutions to all tiles $\rightarrow$ additional factor $(1-\epsilon)$

Complete Approximation Algorithm

Proofs - 1

Theorem 1 - (part a)

Let $\delta \in [0, 1]$ and $k = \lceil 10/\epsilon \rceil$ with $\epsilon \in (0, 1]$. *Complete Algorithm* yields a feasible solution $(\underline{C}, \underline{t})$ to $(\mathcal{S}, 1 + \delta)$ with lifetime

$$T\langle \mathcal{S}, 1 + \delta \rangle \geq (1 - \epsilon) \cdot f \cdot T_{\text{opt}}\langle \mathcal{S}, 1 \rangle.$$

Feasibility

- follows directly from Lemma 1 & 2

Approximation Guarantee

- node discretization $\rightarrow T\langle \mathcal{S}, 1 + \delta \rangle \geq f \cdot T_{\text{opt}}\langle \mathcal{S}, 1 \rangle$ for all squares
- combining solutions to all tiles $\rightarrow$ additional factor $(1 - \epsilon)$

Complete Approximation Algorithm

Proofs - 2

Theorem 1 - (part b)

The runtime complexity of *Complete Algorithm* is bounded by

$$\mathcal{O}\Big(|\mathcal{S}| + {}^{1}/_{\epsilon}|\mathcal{S}| \cdot g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2}))\Big) = \mathcal{O}(|\mathcal{S}|)$$

with $g_{\mathcal{A}}(|\mathcal{S}|)$ runtime of algorithm $\mathcal{A}$ with respect to number of nodes.

Runtime

- $|\mathcal{S}|$: discretizing nodes
- $\mathcal{O}({}^{1}/_{\epsilon}|\mathcal{S}|)$: squares to be computed
- $g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2}))$: runtime of algorithm $\mathcal{A}$ for each square

Complete Approximation Algorithm

Proofs - 2

Theorem 1 - (part b)

The runtime complexity of *Complete Algorithm* is bounded by

$$\mathcal{O}\Big(|\mathcal{S}| + {}^{1}/_{\epsilon}|\mathcal{S}| \cdot g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2}))\Big) = \mathcal{O}(|\mathcal{S}|)$$

with $g_{\mathcal{A}}(|\mathcal{S}|)$ runtime of algorithm $\mathcal{A}$ with respect to number of nodes.

Runtime

- $|\mathcal{S}|$: discretizing nodes
- $\mathcal{O}({}^{1}/_{\epsilon}|\mathcal{S}|)$: squares to be computed
- $g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2}))$: runtime of algorithm $\mathcal{A}$ for each square

Complete Approximation Algorithm

Proofs - 2

Theorem 1 - (part b)

The runtime complexity of *Complete Algorithm* is bounded by

$$\mathcal{O}\Big(|\mathcal{S}| + {}^1\!/\!_\epsilon|\mathcal{S}| \cdot g_\mathcal{A}(\mathcal{O}({}^1\!/\!_{\delta^2\epsilon^2}))\Big) = \mathcal{O}(|\mathcal{S}|)$$

with $g_\mathcal{A}(|\mathcal{S}|)$ runtime of algorithm $\mathcal{A}$ with respect to number of nodes.

Runtime

- $|\mathcal{S}|$: discretizing nodes
- $\mathcal{O}({}^1\!/\!_\epsilon|\mathcal{S}|)$: squares to be computed
- $g_\mathcal{A}(\mathcal{O}({}^1\!/\!_{\delta^2\epsilon^2}))$: runtime of algorithm $\mathcal{A}$ for each square

Complete Approximation Algorithm

Proofs - 2

Theorem 1 - (part b)

The runtime complexity of *Complete Algorithm* is bounded by

$$\mathcal{O}\Big(|\mathcal{S}| + {}^{1}/_{\epsilon}|\mathcal{S}| \cdot g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2}))\Big) = \mathcal{O}(|\mathcal{S}|)$$

with $g_{\mathcal{A}}(|\mathcal{S}|)$ runtime of algorithm $\mathcal{A}$ with respect to number of nodes.

Runtime

- $|\mathcal{S}|$: discretizing nodes
- $\mathcal{O}({}^{1}/_{\epsilon}|\mathcal{S}|)$: squares to be computed
- $g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2}))$: runtime of algorithm $\mathcal{A}$ for each square

Complete Approximation Algorithm

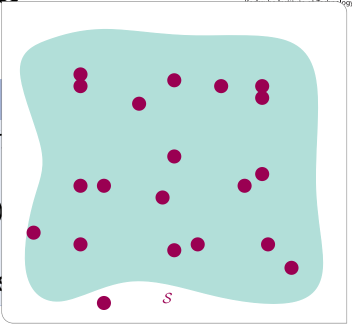
Proofs - 2

Theorem 1 - (part b)

The runtime complexity of *Complete Algorit*

$$\mathcal{O}\Big(|\mathcal{S}| + {}^{1}/_{\epsilon}|\mathcal{S}| \cdot g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2})$$

with $g_{\mathcal{A}}(|\mathcal{S}|)$ runtime of algorithm $\mathcal{A}$ with res



Runtime

- $|\mathcal{S}|$: discretizing nodes
- $\mathcal{O}({}^{1}/_{\epsilon}|\mathcal{S}|)$: squares to be computed
- $g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2}))$: runtime of algorithm $\mathcal{A}$ for each square

Complete Approximation Algorithm

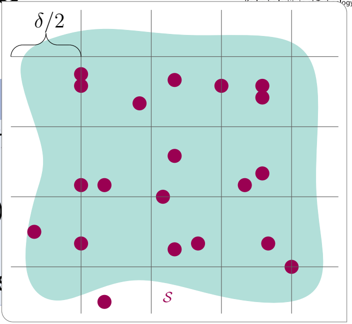
Proofs - 2

Theorem 1 - (part b)

The runtime complexity of *Complete Algorit*

$$\mathcal{O}\left(|\mathcal{S}| + {}^1/_\epsilon|\mathcal{S}| \cdot g_{\mathcal{A}}(\mathcal{O}({}^1/_{\delta^2\epsilon^2})\right.$$

with $g_{\mathcal{A}}(|\mathcal{S}|)$ runtime of algorithm $\mathcal{A}$ with res



Runtime

- $|\mathcal{S}|$: discretizing nodes
- $\mathcal{O}({}^1/_\epsilon|\mathcal{S}|)$: squares to be computed
- $g_{\mathcal{A}}(\mathcal{O}({}^1/_{\delta^2\epsilon^2}))$: runtime of algorithm $\mathcal{A}$ for each square

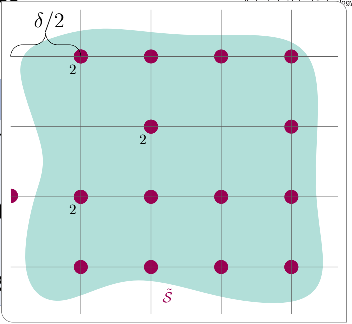
Complete Approximation Algorithm
Proofs - 2

Theorem 1 - (part b)

The runtime complexity of *Complete Algorit*

$$\mathcal{O}\Big(|\mathcal{S}| + {}^{1}/_{\epsilon}|\mathcal{S}| \cdot g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2})$$

with $g_{\mathcal{A}}(|\mathcal{S}|)$ runtime of algorithm $\mathcal{A}$ with res



Runtime

- $|\mathcal{S}|$: discretizing nodes
- $\mathcal{O}({}^{1}/_{\epsilon}|\mathcal{S}|)$: squares to be computed
- $g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2}))$: runtime of algorithm $\mathcal{A}$ for each square

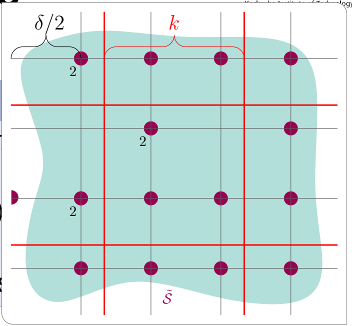
Complete Approximation Algorithm
Proofs - 2

Theorem 1 - (part b)

The runtime complexity of *Complete Algorit*

$$\mathcal{O}\Big(|\mathcal{S}| + {}^{1}/_{\epsilon}|\mathcal{S}| \cdot g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2})$$

with $g_{\mathcal{A}}(|\mathcal{S}|)$ runtime of algorithm $\mathcal{A}$ with res



Runtime

- $|\mathcal{S}|$: discretizing nodes
- $\mathcal{O}({}^{1}/_{\epsilon}|\mathcal{S}|)$: squares to be computed
- $g_{\mathcal{A}}(\mathcal{O}({}^{1}/_{\delta^2\epsilon^2}))$: runtime of algorithm $\mathcal{A}$ for each square

Complete Approximation Algorithm

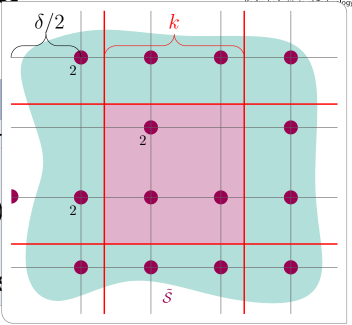
Proofs - 2

Theorem 1 - (part b)

The runtime complexity of *Complete Algorit*

$$\mathcal{O}\Big(|\mathcal{S}| + {}^1/_\epsilon|\mathcal{S}| \cdot g_{\mathcal{A}}(\mathcal{O}({}^1/_{\delta^2\epsilon^2})$$

with $g_{\mathcal{A}}(|\mathcal{S}|)$ runtime of algorithm $\mathcal{A}$ with res



Runtime

- $|\mathcal{S}|$: discretizing nodes
- $\mathcal{O}({}^1/_\epsilon|\mathcal{S}|)$: squares to be computed
- $g_{\mathcal{A}}(\mathcal{O}({}^1/_{\delta^2\epsilon^2}))$: runtime of algorithm $\mathcal{A}$ for each square

Conclusion

Contribution

- pseudo-linear time dual approximation scheme
  - $(1-\epsilon)$ approximation, if sensing ranges are allowed to grow by δ
  - runtime dependent on δ, ϵ, number of nodes
- proof of NP-completeness
  - respecting the geometric structure of the problem

Future Work

- enhance model (non-uniform sensing ranges, obstacles, . . .)
  $\rightarrow$ extension to low-dimensional metrics
- implementation using an exact solver as algorithm $\mathcal{A}$
- distributed algorithm

Thanks go to David Steurer

Conclusion

Contribution

- pseudo-linear time dual approximation scheme
  - $(1-\epsilon)$ approximation, if sensing ranges are allowed to grow by δ
  - runtime dependent on δ, ϵ, number of nodes
- proof of NP-completeness
  - respecting the geometric structure of the problem

Future Work

- enhance model (non-uniform sensing ranges, obstacles, ...)
  $\rightarrow$ extension to low-dimensional metrics
- implementation using an exact solver as algorithm $\mathcal{A}$
- distributed algorithm

Thanks go to David Steurer

Conclusion

Contribution

- pseudo-linear time dual approximation scheme
  - $(1 - \epsilon)$ approximation, if sensing ranges are allowed to grow by δ
  - runtime dependent on δ, ϵ, number of nodes
- proof of NP-completeness
  - respecting the geometric structure of the problem

Future Work

- enhance model (non-uniform sensing ranges, obstacles, . . .)
  $\rightarrow$ extension to low-dimensional metrics
- implementation using an exact solver as algorithm $\mathcal{A}$
- distributed algorithm

Thanks go to David Steurer

Thank you for your attention!

time for questions