Solving Evacuation Problems in Polynomial Space

Miriam Schlöter & Martin Skutella

Flows over Time

Definition

Flows over Time

Definition

Many real life problems crucially depend on *time*:

- logistic - public transport - evacuation problems

Flows over Time

Definition

Many real life problems crucially depend on *time*:

- logistic - public transport - evacuation problems

Flows over time are like…

…like classical (static) network flows + *time component*:

- flow needs time to travel through an arc a:
  arc a has a *transit time* τ_a (length)
- a bounded amount of flow can enter an arc a per time unit:
  arc a has a *capacity* u_a (width)

Flows over Time

Definition

Flows over Time

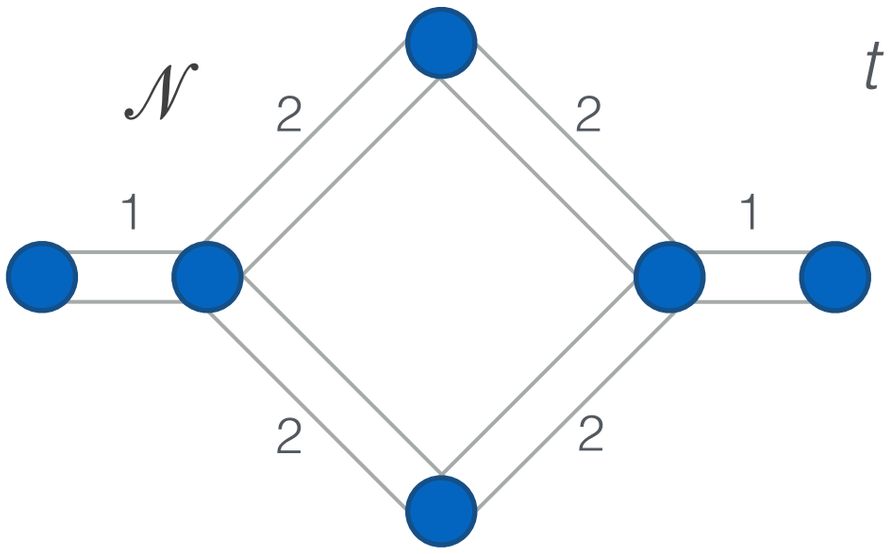
Definition

dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$

Flows over Time

Definition

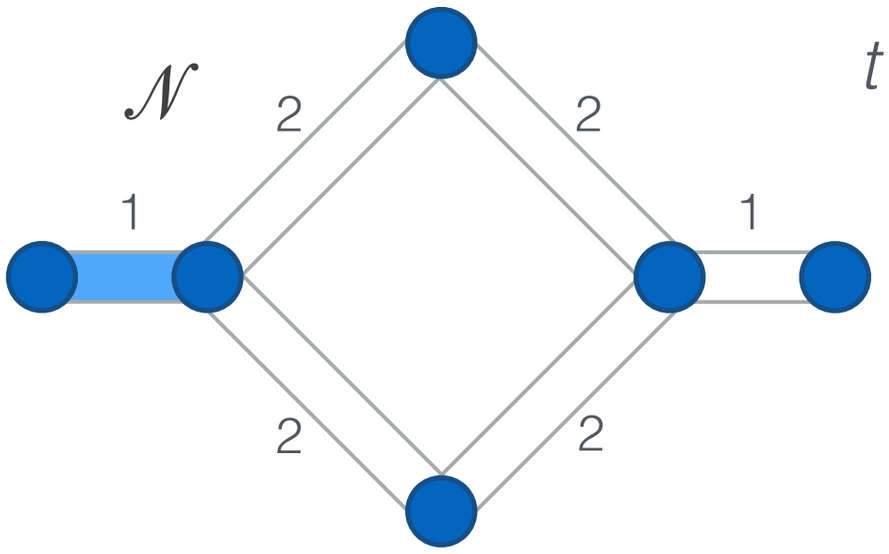
dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$



Flows over Time

Definition

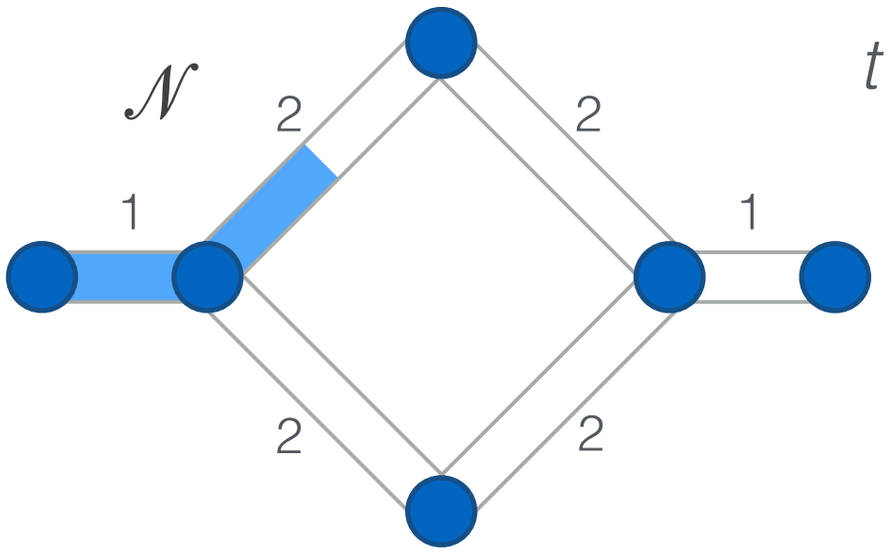
dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$



Flows over Time

Definition

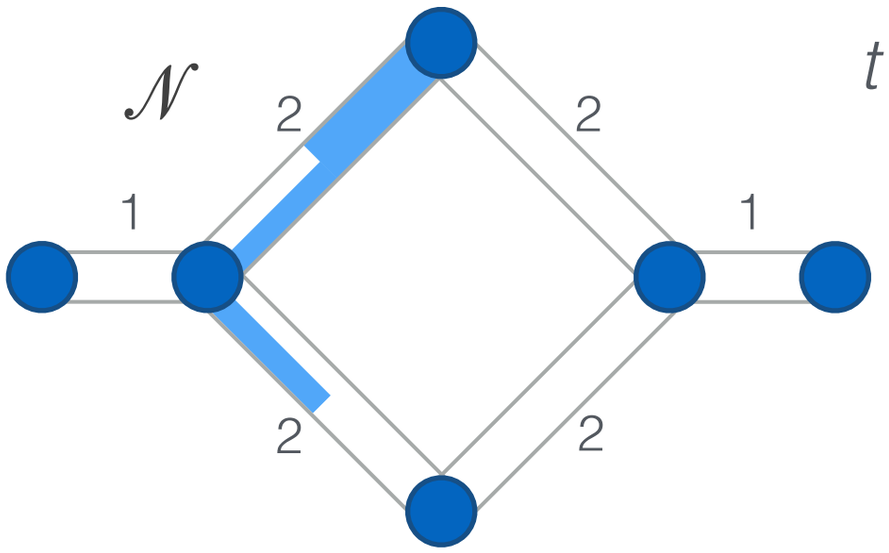
dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$



Flows over Time

Definition

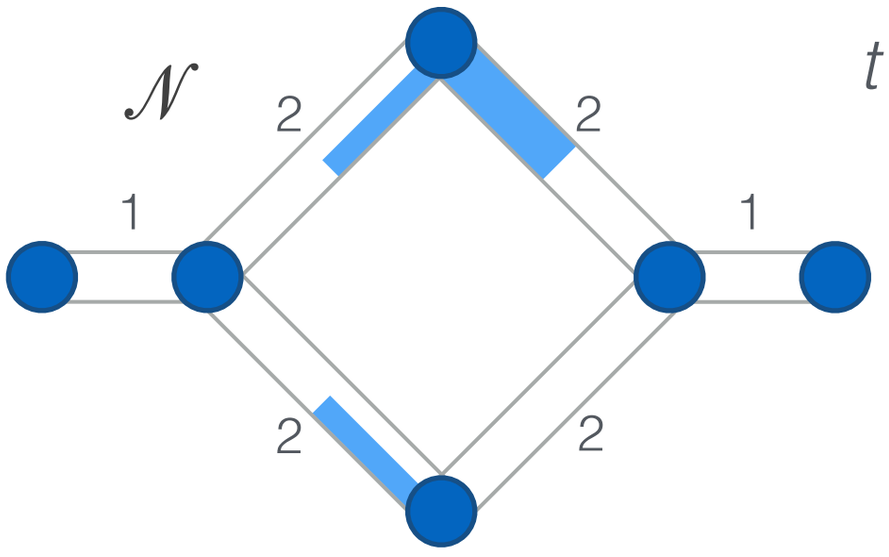
dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$



Flows over Time

Definition

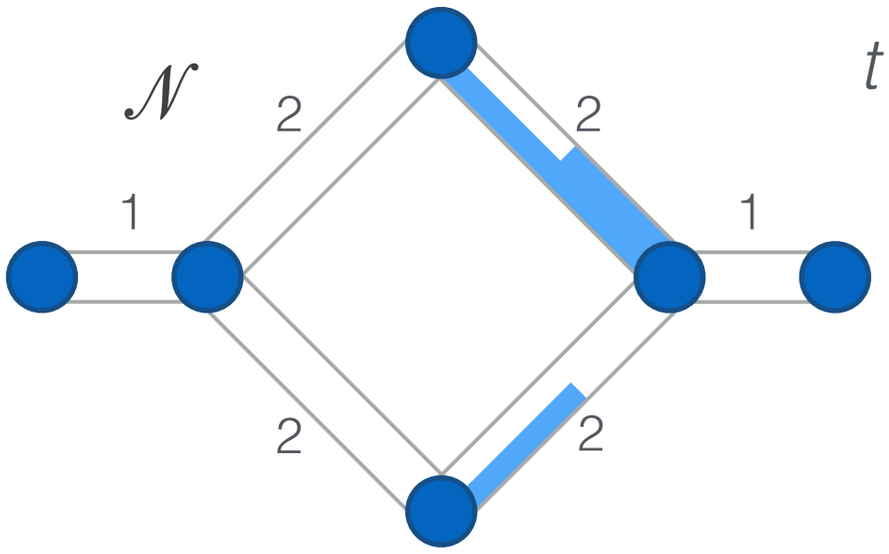
dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$



Flows over Time

Definition

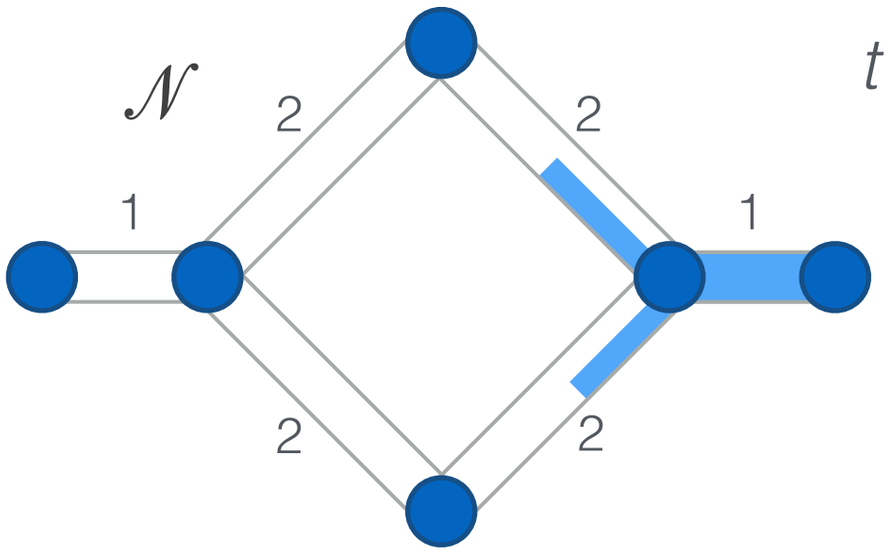
dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$



Flows over Time

Definition

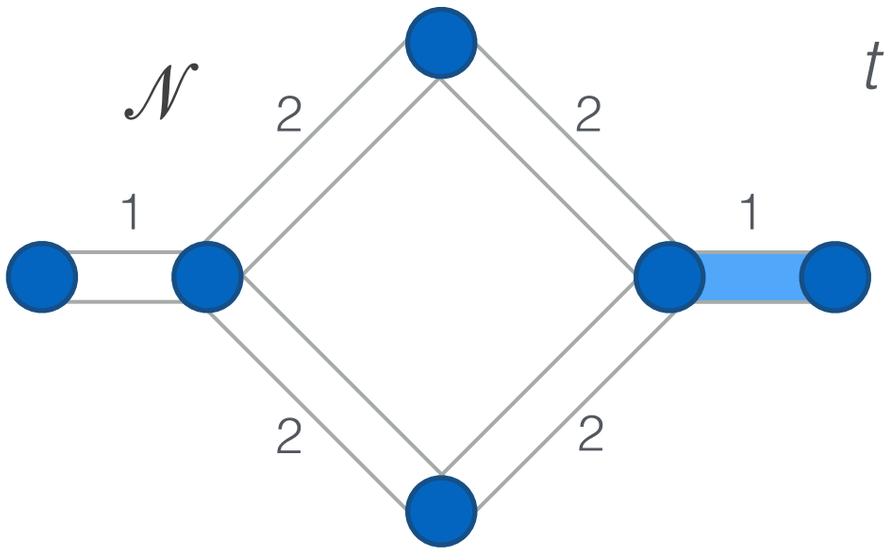
dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$



Flows over Time

Definition

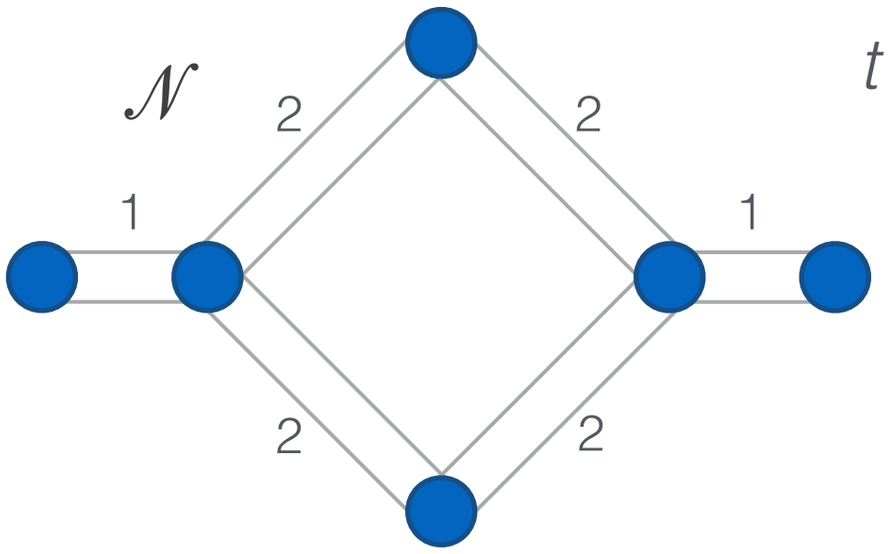
dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$



Flows over Time

Definition

dynamic network $\mathcal{N} = (G=(V,A), u, \tau, S^+, S^-)$: digraph $G = (V,A)$ with capacities u, transit times τ, *sources* $S^+ \subset V$ and *sinks* $S^- \subset V$



Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

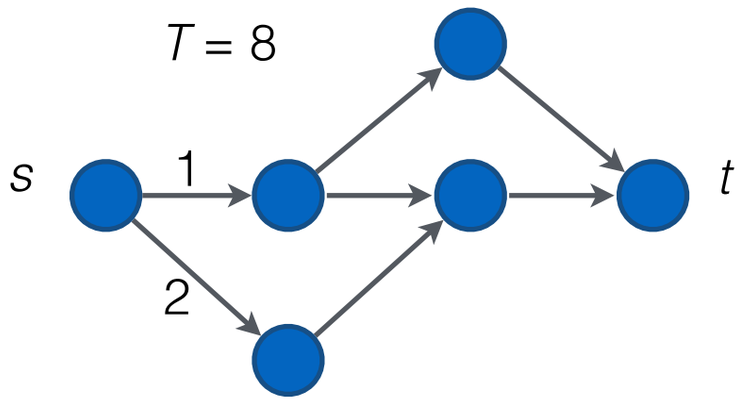
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



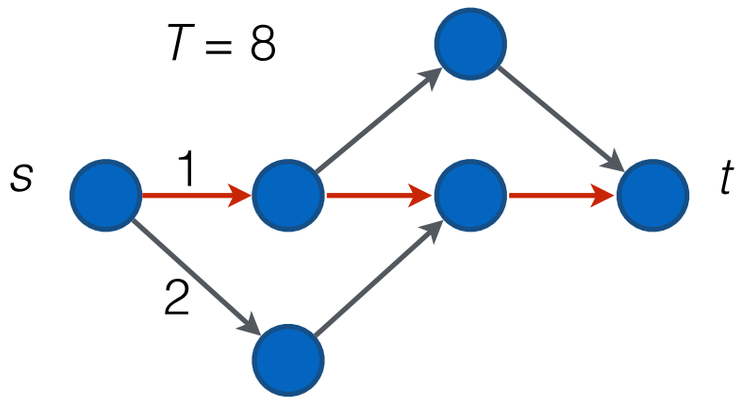
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



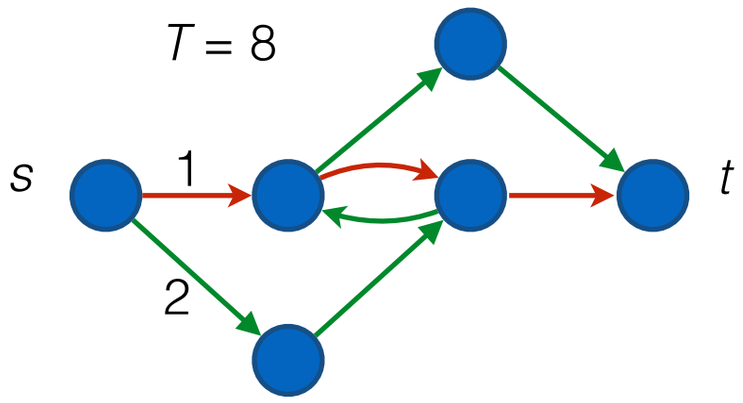
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



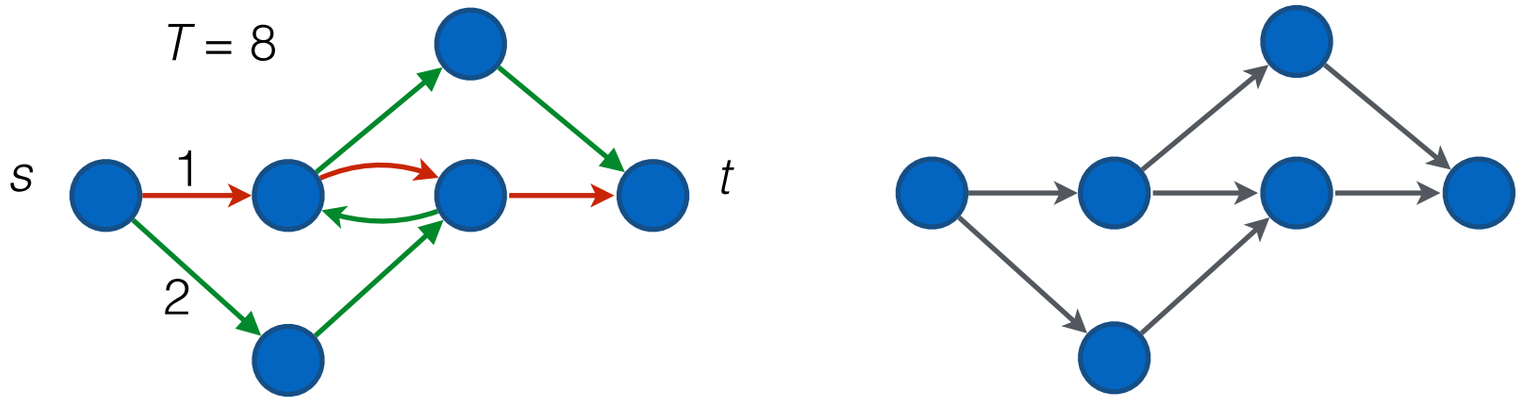
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



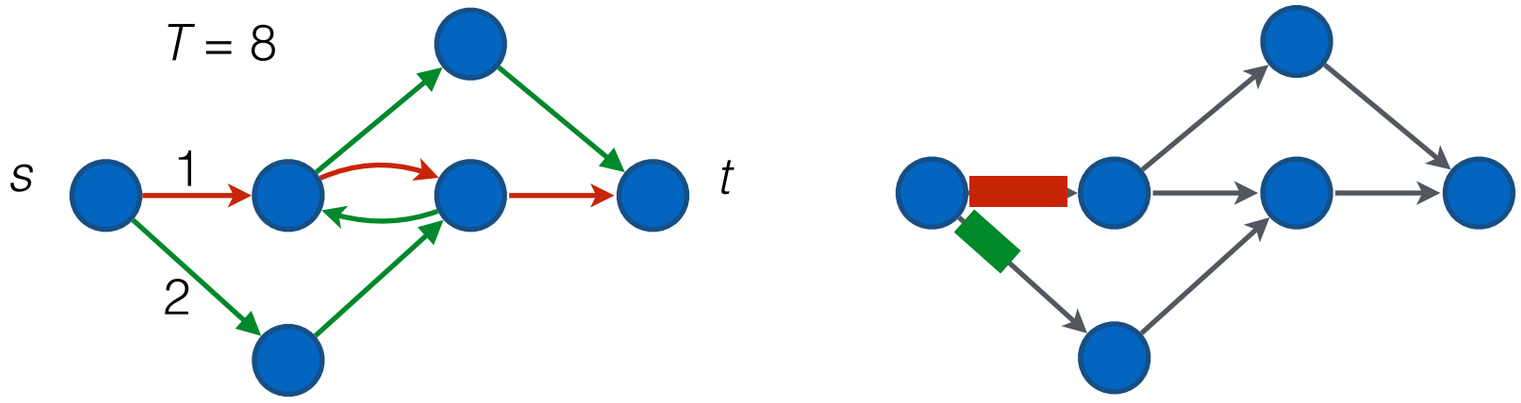
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



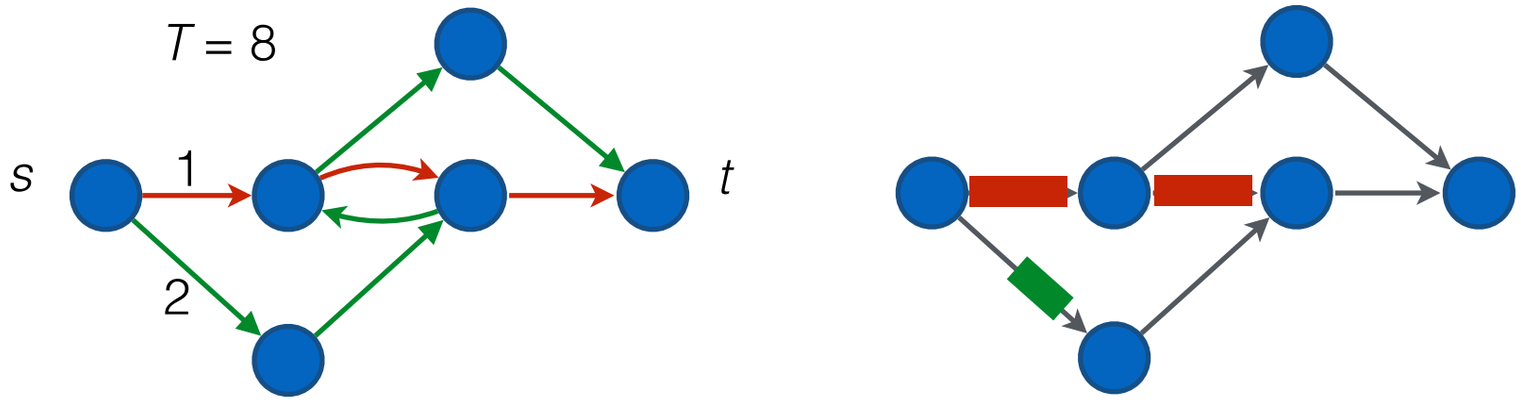
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



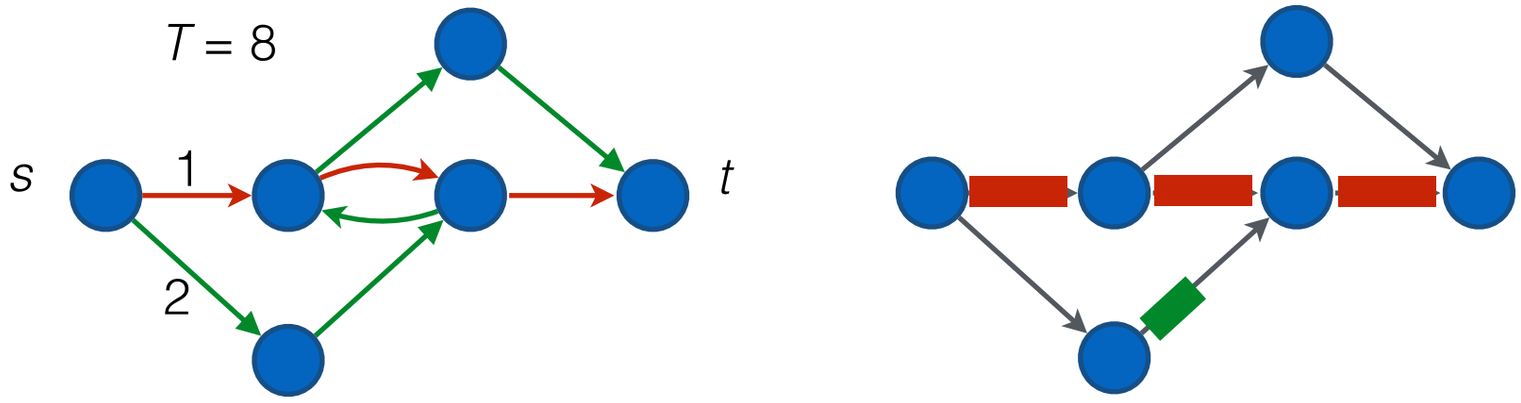
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



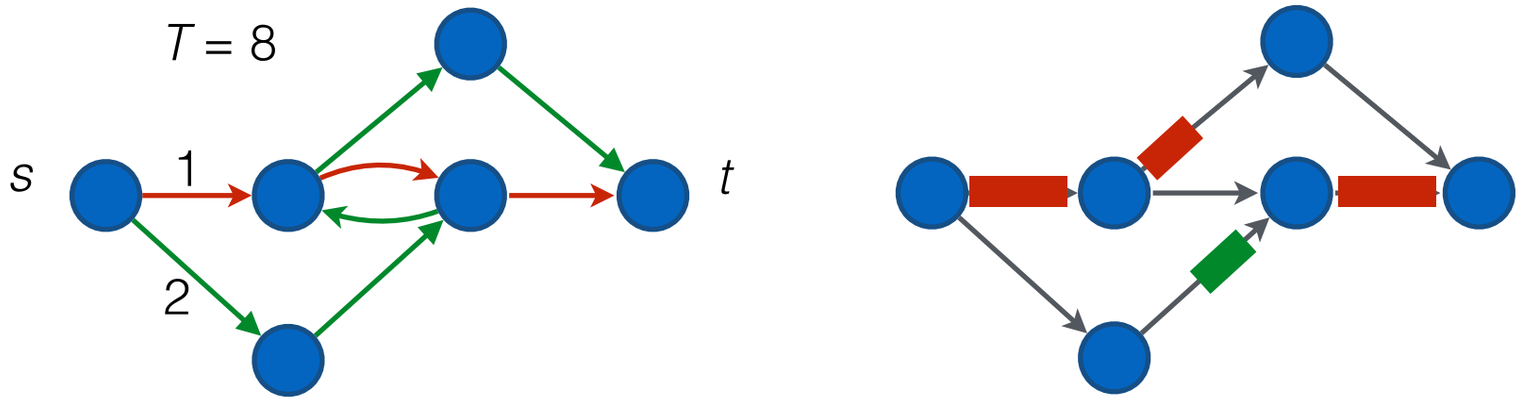
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



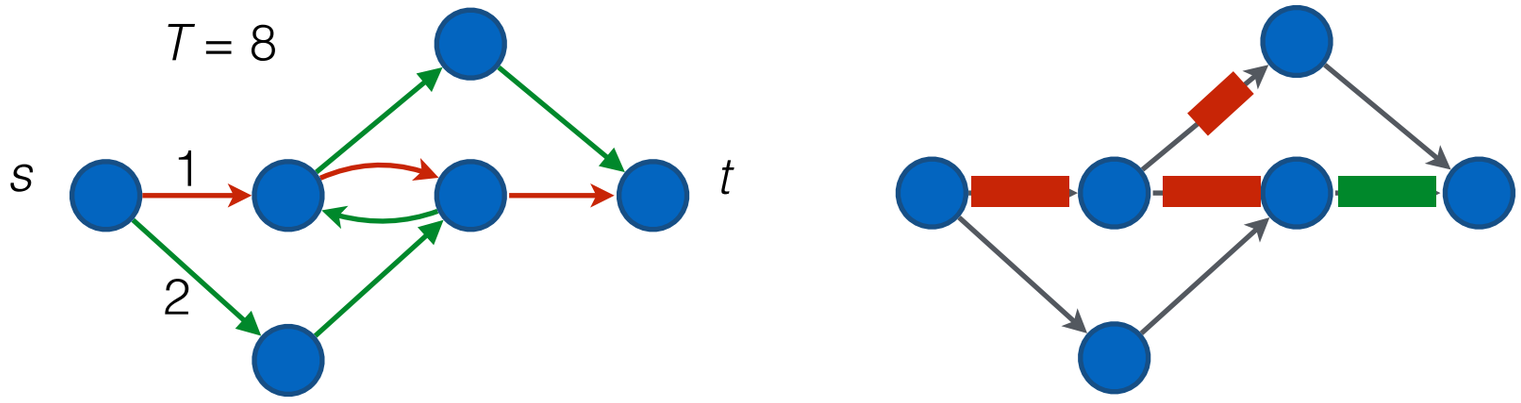
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



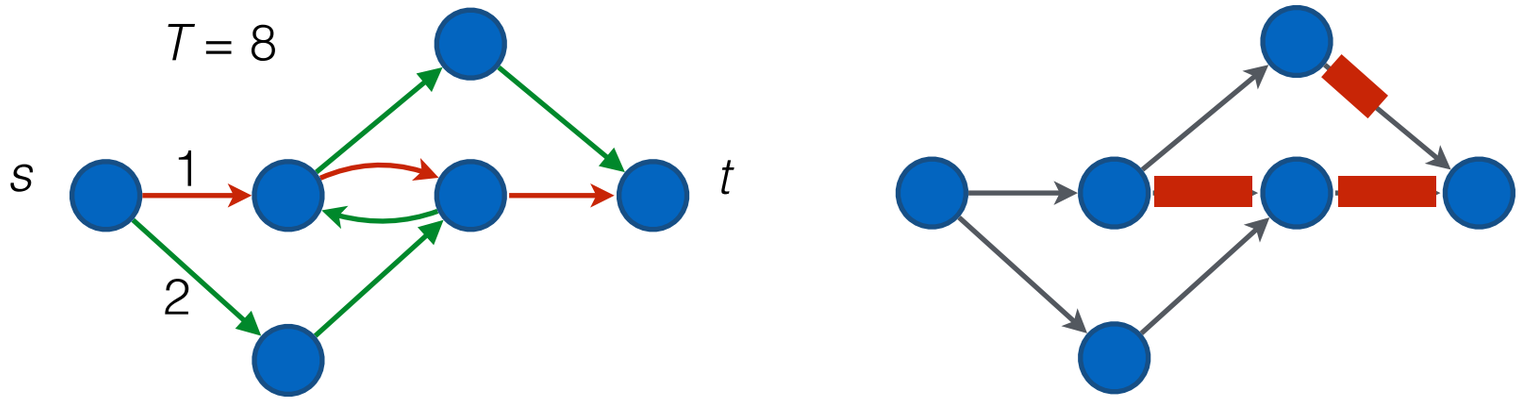
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



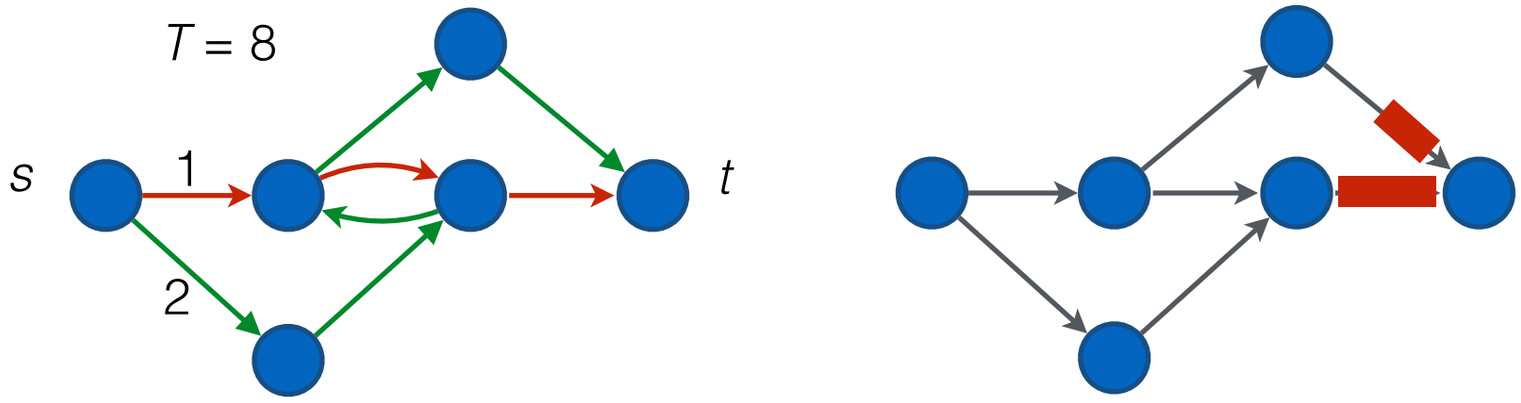
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



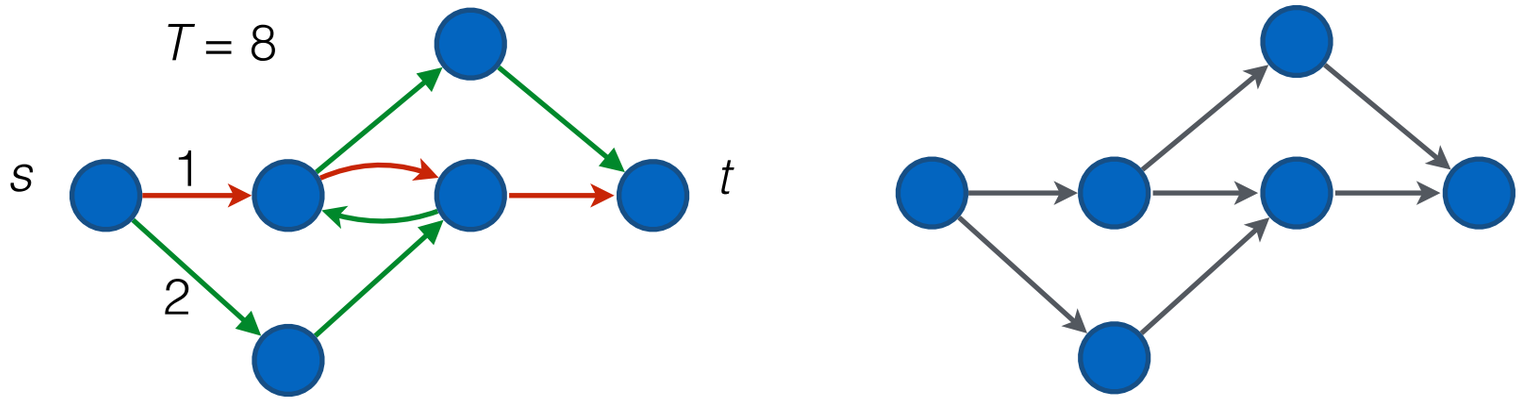
Earliest Arrival Flows

single source / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, s, t)$ and time horizon T

Aim: Flow over time f in $\mathcal{N}$ s.t. at each point in time $\theta \leq T$ as much flow as possible has reached the sink

Solution [Minieka '71, Wilkinson '71]: Send flow along paths P with $\tau(P) \leq T$ occurring in successive shortest path algorithm from s to t



Earliest Arrival Flows

multiple sources / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, S^+, t)$ with supplies v on the sources

Aim: Flow over time f in $\mathcal{N}$ that respects the supplies such that at each point in time as much flow as possible has reached the sink

Earliest Arrival Flows

multiple sources / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, S^+, t)$ with supplies v on the sources

Aim: Flow over time f in $\mathcal{N}$ that respects the supplies such that at each point in time as much flow as possible has reached the sink

→ Earliest Arrival Flows can be used to model *evacuation scenarios*

Earliest Arrival Flows

multiple sources / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, S^+, t)$ with supplies v on the sources

Aim: Flow over time f in $\mathcal{N}$ that respects the supplies such that at each point in time as much flow as possible has reached the sink

→ Earliest Arrival Flows can be used to model *evacuation scenarios*

Baumann & Skutella, (2006), No explicit time-expansion:

1. Construct *earliest arrival pattern p*
2. Compute the earliest arrival flow f using p

Earliest Arrival Flows

multiple sources / single sink

Given: $\mathcal{N} = (G = (V,A), u, \tau, S^+, t)$ with supplies v on the sources

Aim: Flow over time f in $\mathcal{N}$ that respects the supplies such that at each point in time as much flow as possible has reached the sink

→ Earliest Arrival Flows can be used to model *evacuation scenarios*

Baumann & Skutella, (2006), No explicit time-expansion:

1. Construct *earliest arrival pattern p*
2. Compute the earliest arrival flow f using p

attaches exponentially many super sources to the network

Earliest Arrival Flows

multiple sources / single sink - our results, SODA 2017

Solving Evacuation Problems in Polynomial Space

Earliest Arrival Flows

multiple sources / single sink - our results, SODA 2017

New strongly polynomial time algorithm for the
Quickest Transshipment Problem

Solving Evacuation Problems in Polynomial Space

Earliest Arrival Flows

multiple sources / single sink - our results, SODA 2017

New strongly polynomial time algorithm for the
Quickest Transshipment Problem

- A quickest transshipment problem can be solved as a *convex combination of lex-max flows over time.*

Solving Evacuation Problems in Polynomial Space

Earliest Arrival Flows

multiple sources / single sink - our results, SODA 2017

New strongly polynomial time algorithm for the
Quickest Transshipment Problem

- A quickest transshipment problem can be solved as a *convex combination of lex-max flows over time.*
- The convex combination can be computed via *one submodular function minimization*

Earliest Arrival Flows

multiple sources / single sink - our results, SODA 2017

New strongly polynomial time algorithm for the
Quickest Transshipment Problem

- A quickest transshipment problem can be solved as a *convex combination of lex-max flows over time.*
- The convex combination can be computed via *one submodular function minimization*

A polynomial space algorithm for a
generalization of lex-max flows over time

Earliest Arrival Flows

multiple sources / single sink - our results, SODA 2017

New strongly polynomial time algorithm for the
Quickest Transshipment Problem

+

A polynomial space algorithm for a
generalization of lex-max flows over time

Solving Evacuation Problems in Polynomial Space

Earliest Arrival Flows

multiple sources / single sink - our results, SODA 2017

New strongly polynomial time algorithm for the **Quickest Transshipment Problem**

+

A polynomial space algorithm for a **generalization of lex-max flows over time**

↓ ↓

Earliest arrival flow problems can be solved as **convex combinations of generalization of lex-max flows over time** which can be computed using only **polynomial space**

Thank You!

Solving Evacuation Problems in Polynomial Space