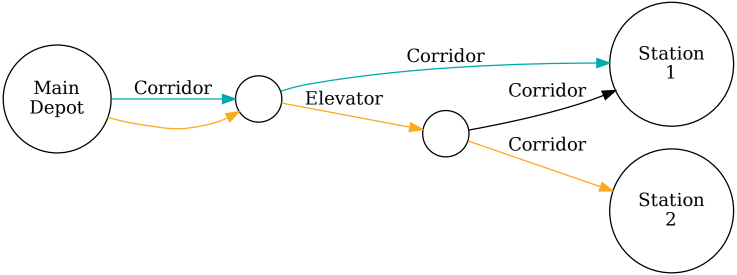
Generalized Temporally Repeated Flows for the Quickest Transshipment and Related Problems

Emma Ahrens

Supervised by Prof. Dr. Christina Büsing

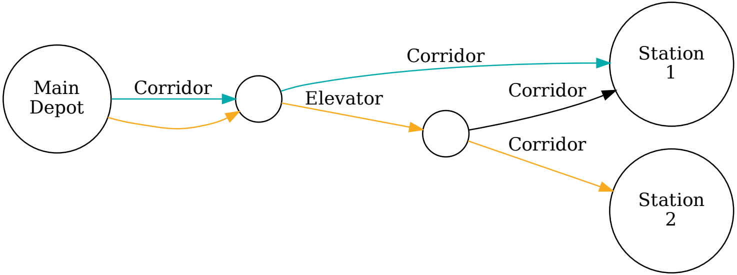
~~July 18, 2025~~

October 28, 2022



- Transportation of hospital beds during the day blocks elevators for time-critical processes

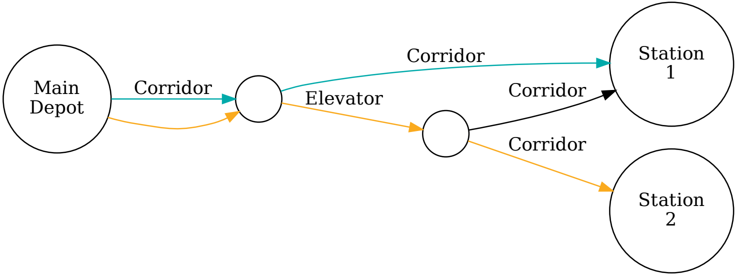
Motivation: Transportation of Beds in a Hospital



- Transportation of hospital beds during the day blocks elevators for time-critical processes

⇒ Distribution after the main working hours? What would be important?

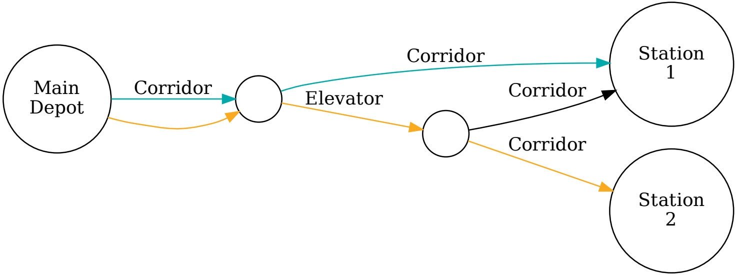
Motivation: Transportation of Beds in a Hospital



- Transportation of hospital beds during the day blocks elevators for time-critical processes

⇒ Distribution after the main working hours? What would be important?

- As quick as possible to relieve the employees

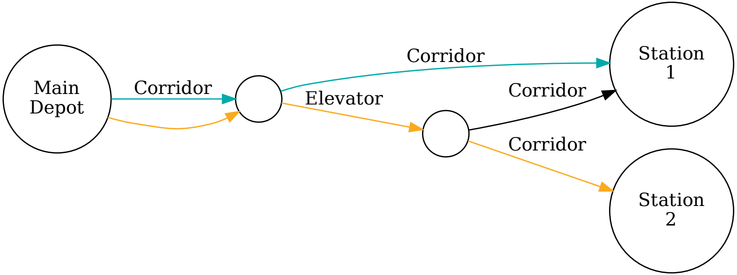


- Transportation of hospital beds during the day blocks elevators for time-critical processes

⇒ Distribution after the main working hours? What would be important?

- As quick as possible to relieve the employees
- As easy to remember as possible to reduce confusion

Motivation: Transportation of Beds in a Hospital



- Transportation of hospital beds during the day blocks elevators for time-critical processes

⇒ Distribution after the main working hours? What would be important?

- As quick as possible to relieve the employees
- As easy to remember as possible to reduce confusion
- As few employees involved as possible

Contents

Table of Contents

Definition

For

- a directed graph $G = (V, A)$,
- a *capacity* function $u : A \to \mathbb{N}_0$,
- a *transit time* function $\tau : A \to \mathbb{N}_0$, and
- a *cost* function $c : A \to \mathbb{N}_0$,

(G, u), (G, u, c), (G, u, τ) and (G, u, τ, c) are *networks*.

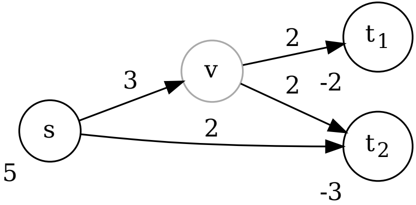


Figure: Network with labels u and b.

Definition

For

- a directed graph $G = (V, A)$,
- a *capacity* function $u : A \to \mathbb{N}_0$,
- a *transit time* function $\tau : A \to \mathbb{N}_0$, and
- a *cost* function $c : A \to \mathbb{N}_0$,

(G, u), (G, u, c), (G, u, τ) and (G, u, τ, c) are *networks*.

Definition

A function $b : V \to \mathbb{Z}$ with $\sum_{v \in V} b(v) = 0$ is a *balance* function.

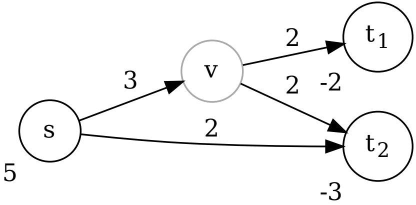


Figure: Network with labels u and b.

Definition

For

- a network (G,u), and
- a balance function b,

a *static* b*-flow* is a function $x : A \to \mathbb{R}_{\geq 0}$, which satisfies

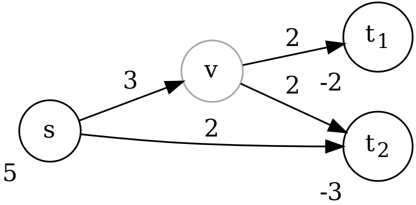


Figure: Network with labels u and b.

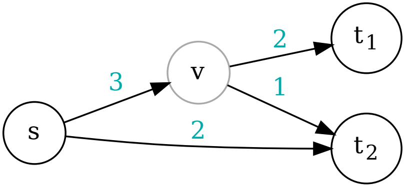


Figure: Flow represented by labels x on arcs.

Definition

For

- a network (G, u), and
- a balance function b,

a *static b-flow* is a function $x : A \to \mathbb{R}_{\geq 0}$, which satisfies

1. the *capacity constraint* $0 \leq x(a) \leq u(a)$ for all $a \in A$, and

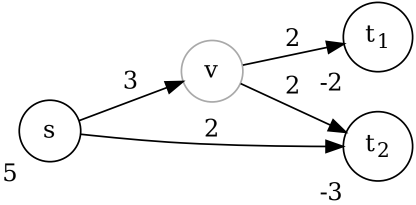


Figure: Network with labels u and b.

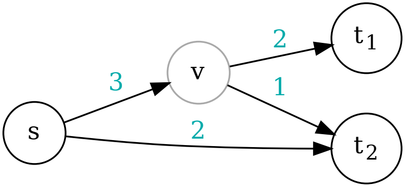


Figure: Flow represented by labels x on arcs.

Definition

For

- a network (G,u), and
- a balance function b,

a *static* b*-flow* is a function $x : A \to \mathbb{R}_{\geq 0}$, which satisfies

2. the *flow conservation*

$$\sum_{a \in \delta^-(v)} x(a) - \sum_{a \in \delta^+(v)} x(a) = -b(v)$$

for all $v \in V$.

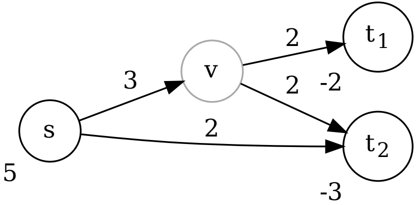


Figure: Network with labels u and b.

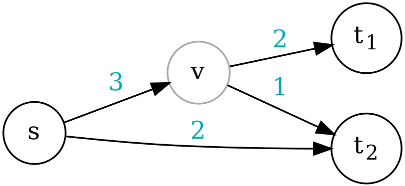


Figure: Flow represented by labels x on arcs.

Definition

For

- a network (G, u, τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a balance function b,

a b-*flow over time* is a family of functions $f_a : \mathbb{Z} \to \mathbb{R}_{\geq 0}$, $a \in A$, which satisfy

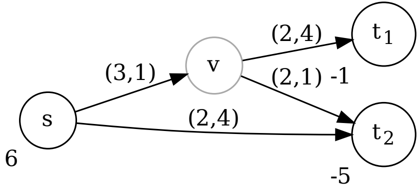


Figure: Network with labels representing (u, τ) and b.

Definition

For

- a network (G, u, τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a balance function b,

a b-*flow over time* is a family of functions $f_a : \mathbb{Z} \to \mathbb{R}_{\geq 0}$, $a \in A$, which satisfy

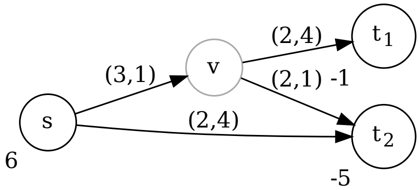


Figure: Network with labels representing (u, τ) and b.

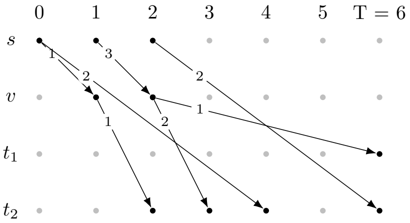


Figure: Labels represent load on arcs.

Definition

For

- a network (G,u,τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a balance function b,

a *b-flow over time* is a family of functions $f_a : \mathbb{Z} \to \mathbb{R}_{\geq 0}$, $a \in A$, which satisfy

1. the *capacity constraint* $0 \leq f_a(\theta) \leq u(a)$ for all $a \in A$ and $\theta \in \{0,\ldots,T\}$,

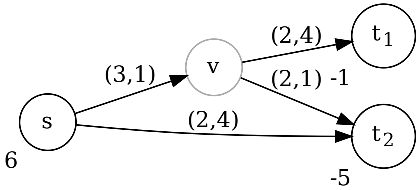


Figure: Network with labels representing (u,τ) and b.

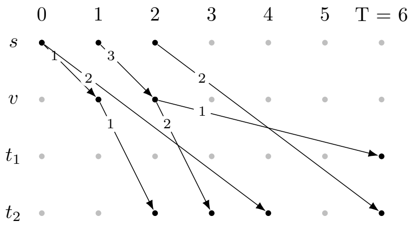


Figure: Labels represent load on arcs.

Definition

For

- a network (G,u,τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a balance function b,

a b-*flow over time* is a family of functions $f_a : \mathbb{Z} \to \mathbb{R}_{\geq 0}$, $a \in A$, which satisfy

2. the *flow completion* $f_a(\theta) = 0$ for all $a \in A$ and $\theta > T - \tau_a$,

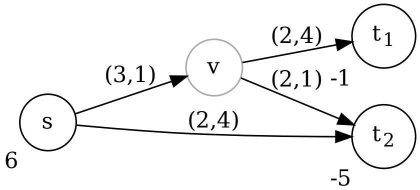


Figure: Network with labels representing (u,τ) and b.

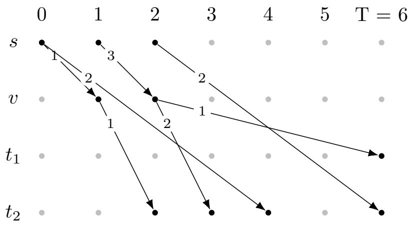


Figure: Labels represent load on arcs.

Definition

For

- a network (G, u, τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a balance function b,

a *b-flow over time* is a family of functions $f_a : \mathbb{Z} \to \mathbb{R}_{\geq 0}$, $a \in A$, which satisfy

3. the *weak flow conservation* for all $v \in V \setminus \{s\}$ and $\theta \in \{0, \ldots, T\}$

$$\sum_{a \in \delta^-(v)} \sum_{\xi=0}^{\theta-\tau_a} f_a(\xi) - \sum_{a \in \delta^+(v)} \sum_{\xi=0}^{\theta} f_a(\xi) \geq 0,$$

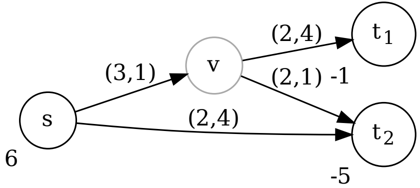


Figure: Network with labels representing (u, τ) and b.

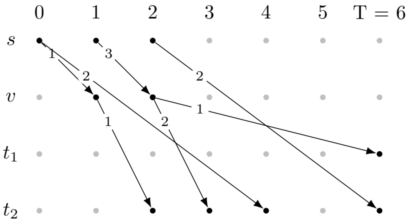


Figure: Labels represent load on arcs.

Definition

For

- a network (G, u, τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a balance function b,

a b-*flow over time* is a family of functions $f_a : \mathbb{Z} \to \mathbb{R}_{\geq 0}$, $a \in A$, which satisfy

4. the *strict flow conservation* for all $v \in V$

$$\sum_{a \in \delta^-(v)} \sum_{\xi=0}^{T-\tau_a} f_a(\xi) - \sum_{a \in \delta^+(v)} \sum_{\xi=0}^{T} f_a(\xi) = -b(v),$$

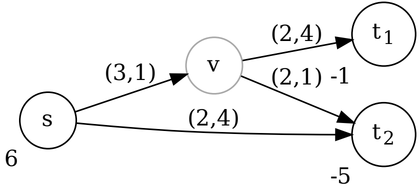


Figure: Network with labels representing (u, τ) and b.

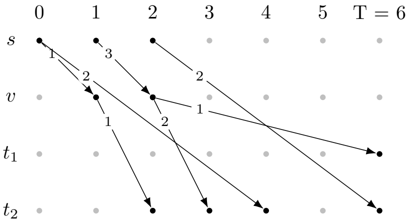


Figure: Labels represent load on arcs.

Definition

For

- a network (G,u,τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a balance function b,

a b-*flow over time* is a family of functions $f_a : \mathbb{Z} \to \mathbb{R}_{\geq 0}$, $a \in A$, which satisfy

5. and $f_a(\theta) = 0$ for $\theta \notin \{0,\ldots,T\}$.

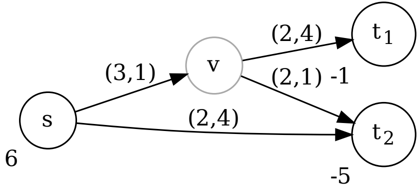


Figure: Network with labels representing (u,τ) and b.

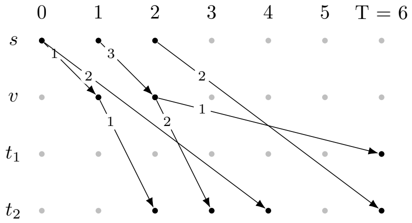


Figure: Labels represent load on arcs.

Definition

For

- a network (G, u, τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a static flow x over (G, u) with path decomposition $y : P \to \mathbb{R}_{\geq 0}$ and $t := \max_{p \in P} \tau(p)$,

the associated *uniform flow* f is

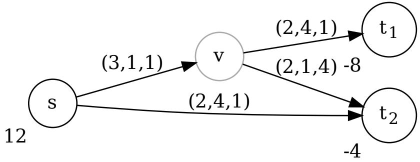


Figure: Network with labels representing (u, τ, c) and b.

Definition

For

- a network (G,u,τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a static flow x over (G,u) with path decomposition $y : P \to \mathbb{R}_{\geq 0}$ and $t := \max_{p \in P} \tau(p)$,

the associated *uniform flow* f is

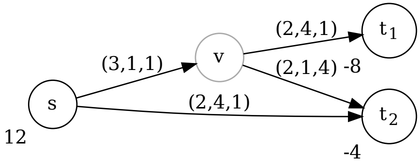


Figure: Network with labels representing (u,τ,c) and b.

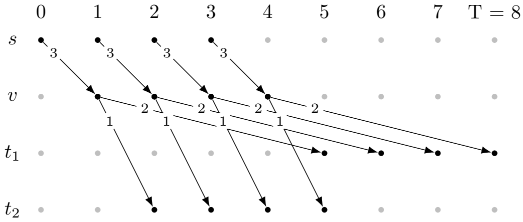


Figure: Labels represent load on arcs.

Uniform Flows

Definition

For

- a network (G, u, τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a static flow x over (G, u) with path decomposition $y : P \to \mathbb{R}_{\geq 0}$ and $t := \max_{p \in P} \tau(p)$,

the associated *uniform flow* f is

$$f_a(\theta) := \sum_{p \in P_a(\theta)} y(p) \qquad \forall a \in A,\ \theta \in \{0, \ldots, T\},$$

where
$P_a(\theta) := \{p \in P \mid a \in p,\ 0 \leq \theta - \tau(p_{[s,v]}) \leq T - t\}$
for $a = (v, w)$ and $p = (s, \ldots, t_i)$,
and $f_a(\theta) := 0$ for $\theta \notin \{0, \ldots, T\}$.

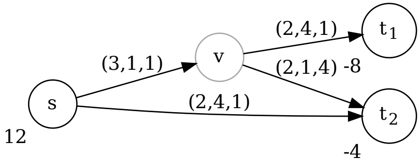


Figure: Network with labels representing (u, τ, c) and b.

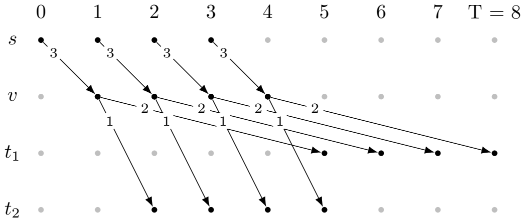


Figure: Labels represent load on arcs.

Definition

For

- a network (G,u,τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a static flow x over (G,u) with path decomposition $y : P \to \mathbb{R}_{\geq 0}$ and $t := \max_{p \in P} \tau(p)$,

the assoc. *temporally repeated flow f* is

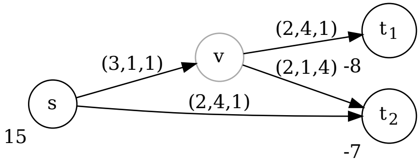


Figure: Network with labels representing (u,τ,c) and b.

Definition

For

- a network (G, u, τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a static flow x over (G, u) with path decomposition $y : P \to \mathbb{R}_{\geq 0}$ and $t := \max_{p \in P} \tau(p)$,

the assoc. *temporally repeated flow f* is

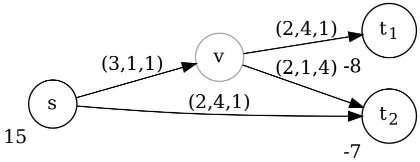


Figure: Network with labels representing (u, τ, c) and b.

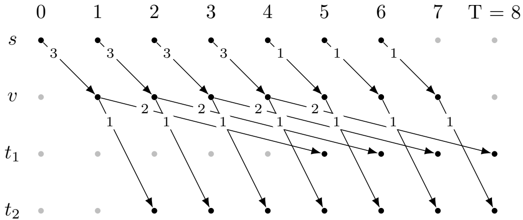


Figure: Labels represent load on arcs.

Definition

For

- a network (G, u, τ),
- a time horizon $T \in \mathbb{N}_0$, and
- a static flow x over (G, u) with path decomposition $y : P \to \mathbb{R}_{\geq 0}$ and $t := \max_{p \in P} \tau(p)$,

the assoc. *temporally repeated flow* f is

$$f_a(\theta) := \sum_{p \in P_a(\theta)} y(p) \qquad \forall a \in A,\ \theta \in \{0, \ldots, T\},$$

where $P_a(\theta) := \{p \in P \mid a \in p,\ \tau(p_{[s,v]}) \leq \theta,$ $\tau(p_{[w,t]}) \leq T - \theta\}$
for $a = (v, w)$ and $p = (s, \ldots, t_i)$,
and $f_a(\theta) := 0$ for $\theta \notin \{0, \ldots, T\}$.

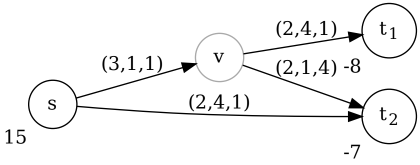


Figure: Network with labels representing (u, τ, c) and b.

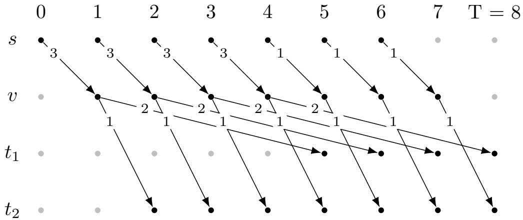
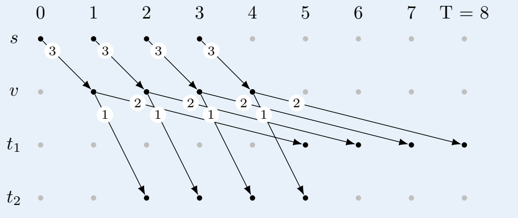
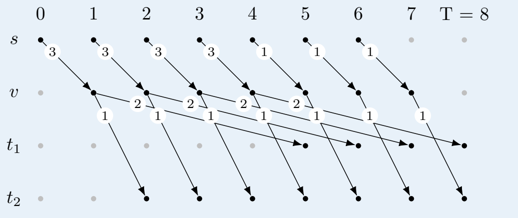


Figure: Labels represent load on arcs.

Uniform Flow

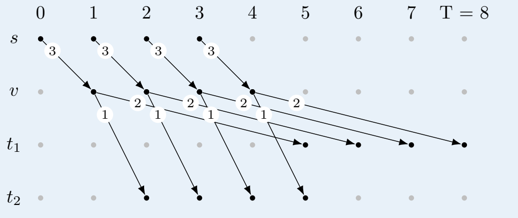


Temporally Repeated Flow

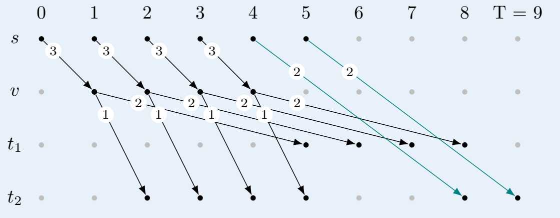


k-Uniform and k-Temporally Repeated Flows

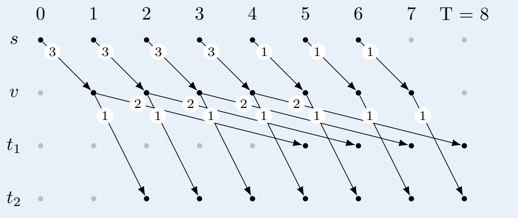
Uniform Flow



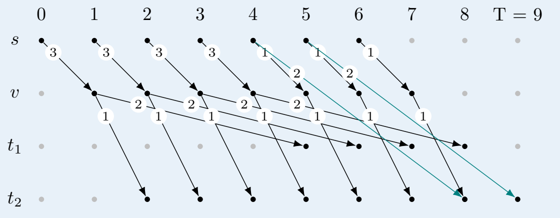
k-Uniform Flow for $k = 2$



Temporally Repeated Flow



k-Temporally Repeated Flow for $k = 2$



- Max Flow Problem

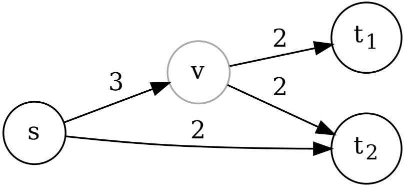


Figure: Network with labels representing u.

- Max Flow Problem
- Min Cost Flow Problem

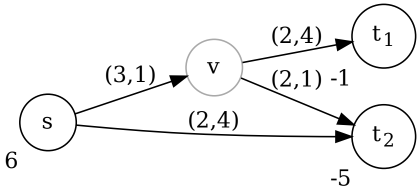


Figure: Network with labels representing (u, c) and b.

- Max Flow Problem
- Min Cost Flow Problem
- Max Flow over Time Problem

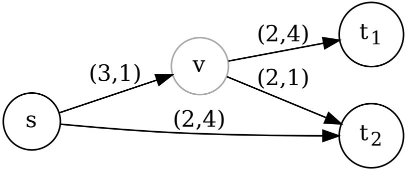


Figure: Network with labels representing (u, τ).

- Max Flow Problem
- Min Cost Flow Problem
- Max Flow over Time Problem
- Min Cost Flow over Time Problem

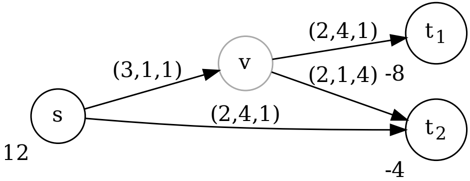


Figure: Network with labels representing (u, τ, c) and b.

- Max Flow Problem
- Min Cost Flow Problem
- Max Flow over Time Problem
- Min Cost Flow over Time Problem
- Earliest Arrival Flow over Time Problem

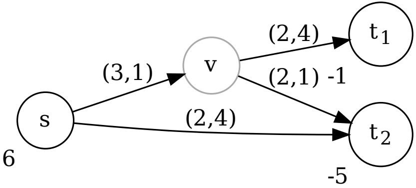


Figure: Network with labels representing (u, τ) and b.

- Max Flow Problem
- Min Cost Flow Problem
- Max Flow over Time Problem
- Min Cost Flow over Time Problem
- Earliest Arrival Flow over Time Problem
- Quickest Transshipment Problem

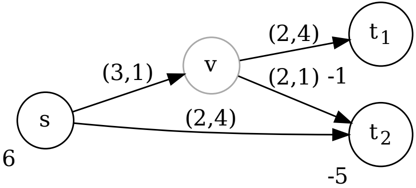


Figure: Network with labels representing (u, τ) and b.

- Max Flow Problem
- Min Cost Flow Problem
- Max Flow over Time Problem
- Min Cost Flow over Time Problem
- Earliest Arrival Flow over Time Problem
- Quickest Transshipment Problem
- Min Cost Uniform Flow Problem

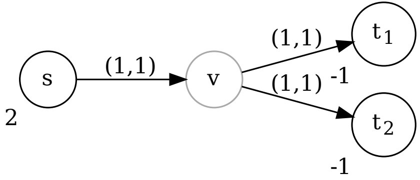


Figure: Network with labels representing (u, τ) and b.

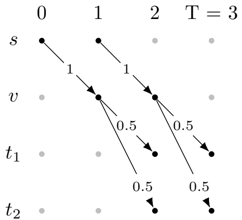


Figure: Labels represent loads on arcs.

- Max Flow Problem
- Min Cost Flow Problem
- Max Flow over Time Problem
- Min Cost Flow over Time Problem
- Earliest Arrival Flow over Time Problem
- Quickest Transshipment Problem
- Min Cost Uniform Flow Problem
- Quickest Transshipment Problem for Uniform Flows

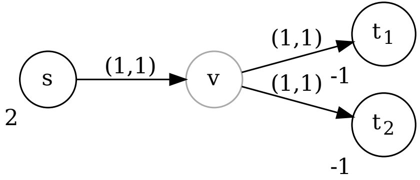


Figure: Network with labels representing (u,τ) and b.

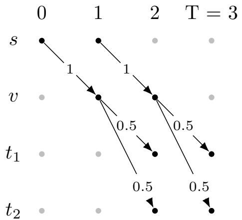


Figure: Labels represent loads on arcs.

Table of Contents

Definition

G is a tree, then (G, u, τ, c) is a *tree network*.

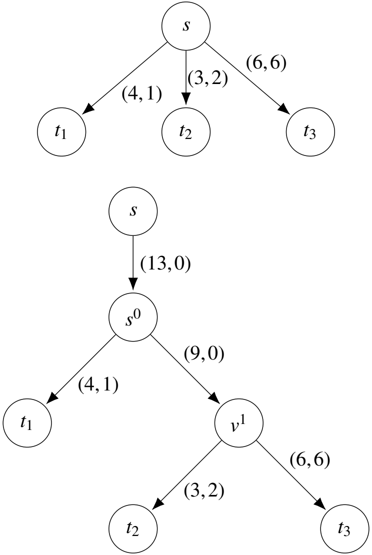


Figure: Networks with labels representing (u, τ).

Definition

G is a tree, then (G, u, τ, c) is a *tree network*.

Definition

Flow f on (G, u, τ, c) and f' on (G', u', τ', c').
Then $f \equiv f'$ if

- the number of sinks is equal,
- for each sink at each point in time the same number of units arrives with the same aggregated costs.

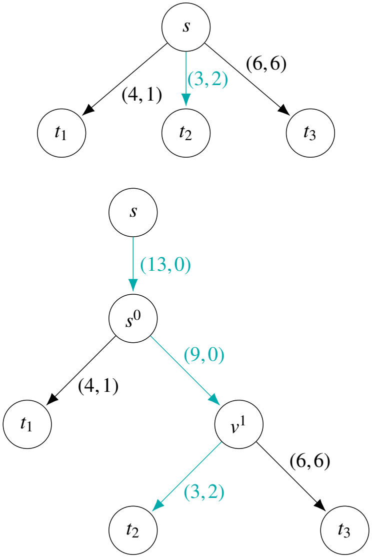


Figure: Networks with labels representing (u, τ).

Definition

Two tree networks (G, u, τ, c), (G', u', τ', c') are *equivalent*,

$$(G, u, \tau, c) \equiv (G', u', \tau', c'),$$

if for each flow f on (G, u, τ, c) there exists an equivalent flow f' on the other network (G', u', τ', c') and vice versa.

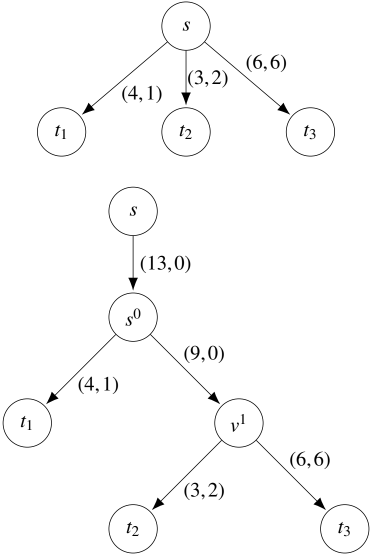


Figure: Networks with labels representing (u, τ).

Definition

Two tree networks (G,u,τ,c), (G',u',τ',c') are *equivalent*,

$$(G,u,\tau,c) \equiv (G',u',\tau',c'),$$

if for each flow f on (G,u,τ,c) there exists an equivalent flow f' on the other network (G',u',τ',c') and vice versa.

$\Rightarrow$ For all mentioned network problems: Given two equivalent networks, all optimal solutions have the *same objective value*. The optimal solutions have equivalent counterparts.

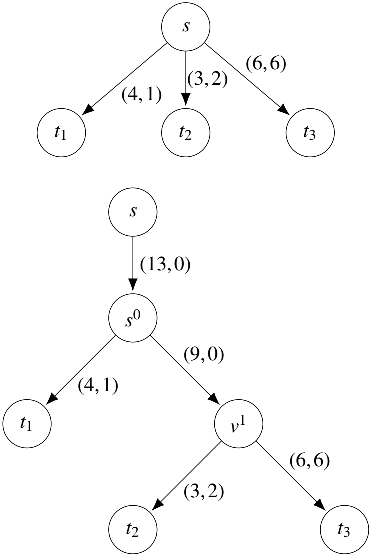


Figure: Networks with labels representing (u,τ).

Merge Nodes

Definition

Tree network (G,u,τ,c) and node $v \in V$. We define operation $\rho_1((G,u,\tau,c),v)$:
It maps to (G,u,τ,c) if v is

- a leaf,
- has at least two children, or
- is the root.

Otherwise, it maps to (G',u',τ',c') which resembles the network (G,u,τ,c) with the changes in the picture.

Lemma

For tree network (G,u,τ,c) and any node $v \in V$, it is $\rho_1((G,u,\tau,c),v) \equiv (G,u,\tau,c)$.

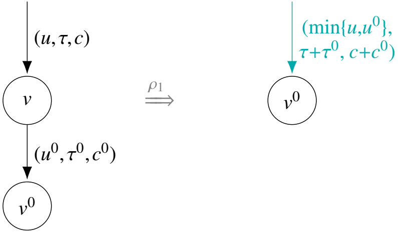


Figure: The operation ρ_1.

Definition

Tree network (G, u, τ, c) and node $v \in V$. We define operation $\rho_2((G, u, \tau, c), v)$:
It maps to (G, u, τ, c) if v has

- at most two children.

Otherwise, it maps to (G', u', τ', c') which resembles the network (G, u, τ, c) with the changes in the picture.

Lemma

For tree network (G, u, τ, c) and any node $v \in V$, it is $\rho_2((G, u, \tau, c), v) \equiv (G, u, \tau, c)$.

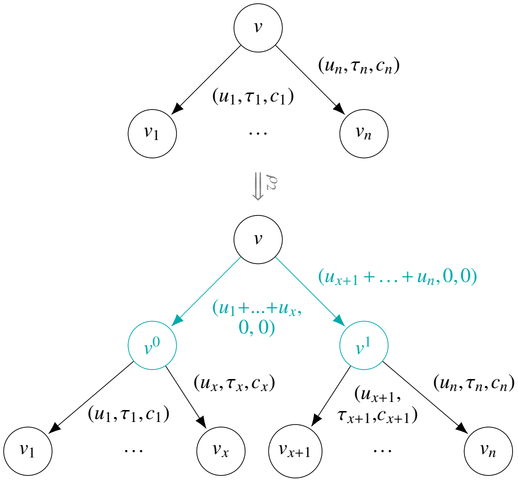


Figure: The operation ρ_2.

Definition

Tree network (G, u, τ, c) and node $v \in V$. We define operation $\rho_3((G, u, \tau, c), v)$:
It maps to (G, u, τ, c) if v has

- at most one child.

Otherwise, it maps to (G', u', τ', c') which resembles the network (G, u, τ, c) with the changes in the picture.

Lemma

For tree network (G, u, τ, c) and any node $v \in V$, it is $\rho_3((G, u, \tau, c), v) \equiv (G, u, \tau, c)$.

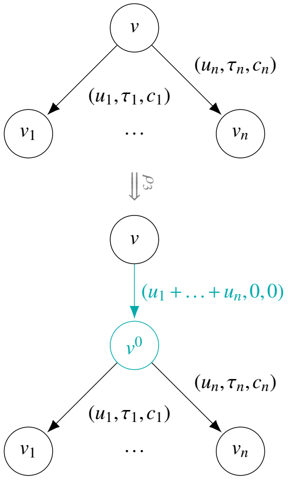


Figure: The operation ρ_3.

Definition

An *almost-binary tree* is a tree G with

- the root node $r \in V$ has at most one child $v \in V$, and
- the subtree of v is a complete binary tree (if v exists).

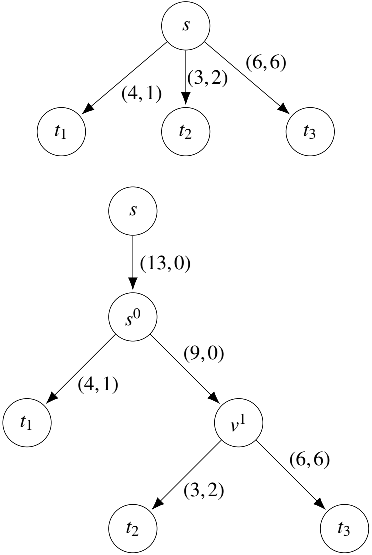


Figure: Networks with labels representing (u,τ).

Definition

An *almost-binary tree* is a tree G with

- the root node $r \in V$ has at most one child $v \in V$, and
- the subtree of v is a complete binary tree (if v exists).

Theorem

For each tree network (G, u, τ, c), there exists an equivalent tree network where the underlying graph is an almost-binary tree.

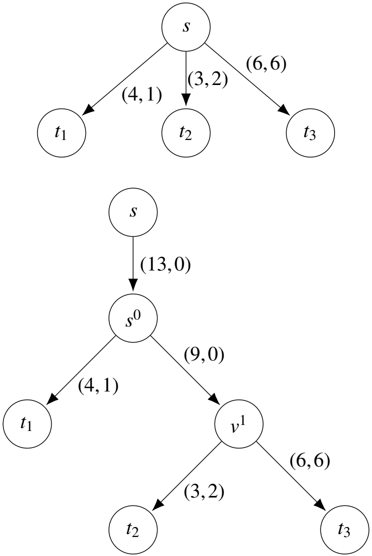


Figure: Networks with labels representing (u, τ).

Theorem

For each tree network (G,u,τ,c), there exists an equivalent tree network where the underlying graph is an almost-binary tree.

Theorem

Let (G,u,τ,c) be tree network, where G has degree $k \in \mathbb{N}_0$ and $|G| = n$. Then, an equivalent binary tree network can be computed in $\mathcal{O}(n \cdot k)$ steps.

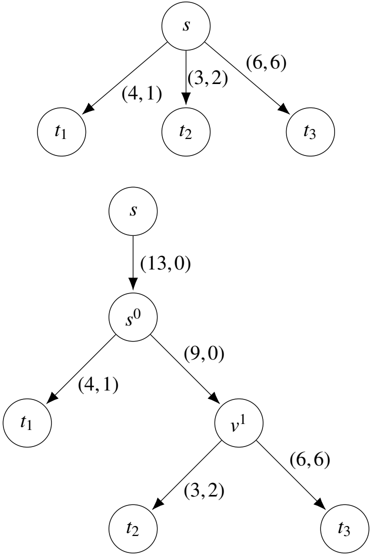
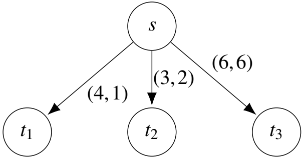
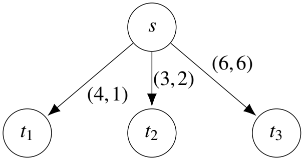
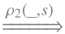


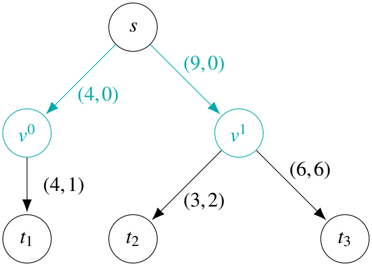
Figure: Networks with labels representing (u,τ).

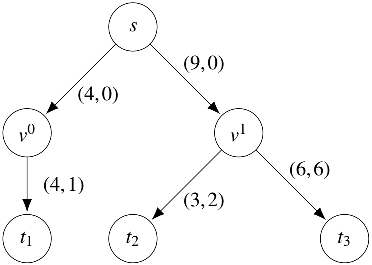
Equivalence Classes of Trees



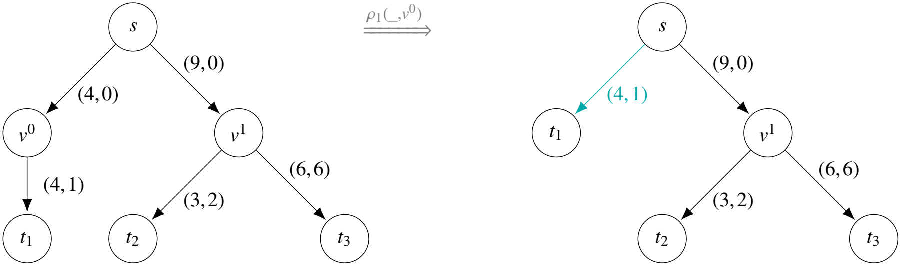

s
$(4,1)$
$(3,2)$
$(6,6)$
t_1
t_2
t_3


$\rho_2(\_,s)$

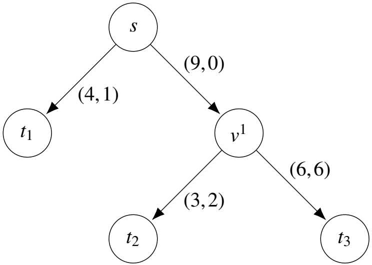

s
$(4,0)$
$(9,0)$
v^0
v^1
$(4,1)$
$(3,2)$
$(6,6)$
t_1
t_2
t_3


s
$(9,0)$
$(4,0)$
v^0
v^1
$(4,1)$
$(6,6)$
$(3,2)$
t_1
t_2
t_3

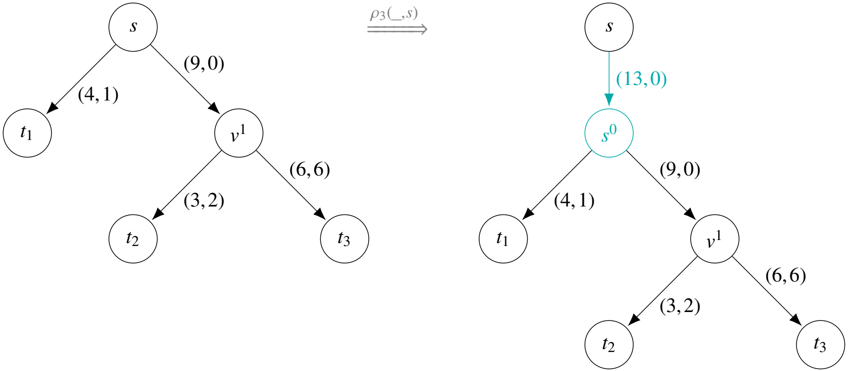
Equivalence Classes of Trees



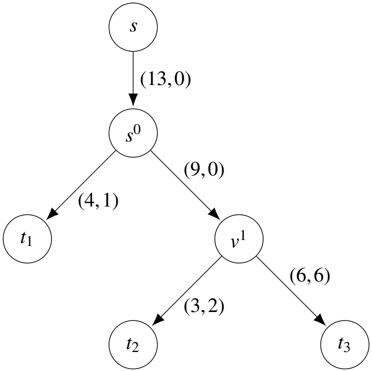
Equivalence Classes of Trees



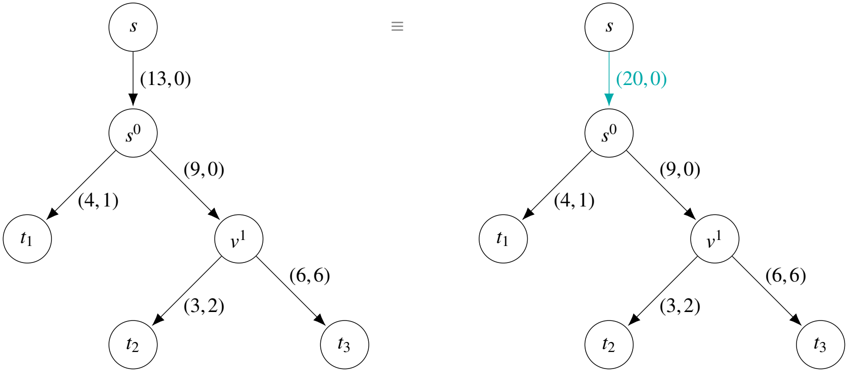
Equivalence Classes of Trees



Equivalence Classes of Trees



Equivalence Classes of Trees



Definition

Given a tree network (G,u,τ), and a balance function b, find an *integer* k*-uniform flow* with arbitrary $k \leq h$ which

- satisfies the balances and
- has minimal overall time horizon $T \in N$.

Definition

Given a tree network (G, u, τ), and a balance function b, find an *integer* k*-uniform flow* with arbitrary $k \leq h$ which

- satisfies the balances and
- has minimal overall time horizon.

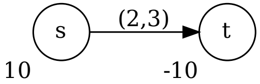


Figure: Network with labels representing (u, τ) and b.

Definition

Given a tree network (G,u,τ), and a balance function b, find an *integer* k*-uniform flow* with arbitrary $k \leq h$ which

- satisfies the balances and
- has minimal overall time horizon.

Lemma

For an almost-binary tree network (G,u,τ,c) *with one leaf and balance function* b*. Then, the minimal time horizon for a* 1*-uniform flow satisfying* b *is*

$$T := \left\lceil \frac{b}{u} \right\rceil + \tau - 1.$$

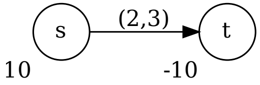


Figure: Network with labels representing (u,τ) and b.

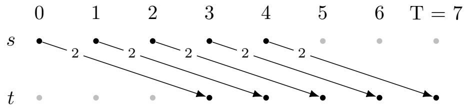


Figure: Labels represent loads on arcs.

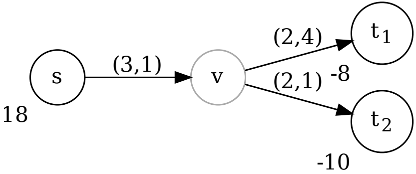


Figure: Network with labels representing (u,τ) and b.

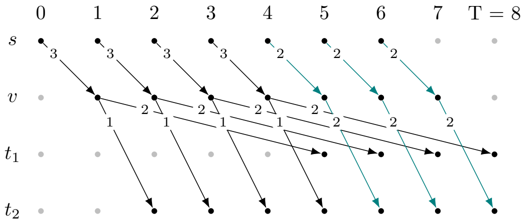


Figure: Labels represent loads on arcs.

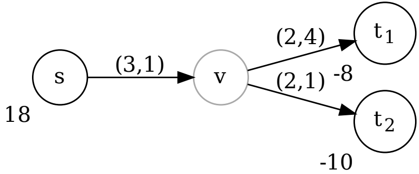


Figure: Network with labels representing (u, τ) and b.

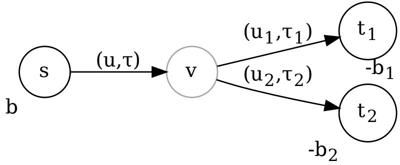


Figure: Network with labels representing (u, τ) and b.

Lemma

We set $\mathbf{u}_i := \min\{u, u_i\}$, $\boldsymbol{\tau}_i := \tau + \tau_i$.
The quickest 1- *or* 2-*uniform flow has time horizon*

$$
\begin{aligned}
T := \min\big(&\{ f(\mathbf{u}_1,\boldsymbol{\tau}_1,b_1,\mathbf{u}_2,\boldsymbol{\tau}_2,b_2) \} \\
&\cup \{ g(\mathbf{u}_1,\boldsymbol{\tau}_1,b_1,\mathbf{u}_2,\boldsymbol{\tau}_2,b_2,x_1,x_2) \mid \\
&\quad 1 \le x_1 \le \mathbf{u}_1,\ 1 \le x_2 \le \mathbf{u}_2,\ x_1 + x_2 \le u \} \\
&\cup \{ h(\boldsymbol{\tau}_1,b_1,\boldsymbol{\tau}_2,b_2,x_1,x_2,y_1,y_2,d) \mid \\
&\quad 1 \le x_1 \le \mathbf{u}_1,\ 1 \le y_1 \le \mathbf{u}_1,\ 1 \le x_2 \le \mathbf{u}_2, \\
&\quad 1 \le y_2 \le \mathbf{u}_2,\ x_1 + x_2 \le u,\ y_1 + y_2 \le u, \\
&\quad d < \min\left\{\left\lceil \tfrac{b_1}{x_1} \right\rceil, \left\lceil \tfrac{b_2}{x_2} \right\rceil\right\},\ \left\lceil \tfrac{b_1 - d\cdot x_1}{y_1} \right\rceil = \left\lceil \tfrac{b_2 - d\cdot x_2}{y_2} \right\rceil \} \big).
\end{aligned}
$$

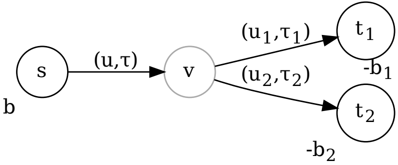


Figure: Network with labels representing (u, τ) and b.

Lemma

We set $\mathbf{u}_i := \min\{u, u_i\}$, $\boldsymbol{\tau}_i := \tau + \tau_i$.
The quickest 1- *or* 2-*uniform flow has time horizon*

$$T := \min\Big(\{\, f(\mathbf{u}_1,\boldsymbol{\tau}_1,b_1,\mathbf{u}_2,\boldsymbol{\tau}_2,b_2) \,\} \\ \cup \{\, g(\mathbf{u}_1,\boldsymbol{\tau}_1,b_1,\mathbf{u}_2,\boldsymbol{\tau}_2,b_2,x_1,x_2) \mid \\ 1 \le x_1 \le \mathbf{u}_1,\ 1 \le x_2 \le \mathbf{u}_2,\ x_1 + x_2 \le u \,\} \\ \cup \{\, h(\boldsymbol{\tau}_1,b_1,\boldsymbol{\tau}_2,b_2,x_1,x_2,y_1,y_2,d) \mid \\ 1 \le x_1 \le \mathbf{u}_1,\ 1 \le y_1 \le \mathbf{u}_1,\ 1 \le x_2 \le \mathbf{u}_2, \\ 1 \le y_2 \le \mathbf{u}_2,\ x_1 + x_2 \le u,\ y_1 + y_2 \le u, \\ d < \min\Big\{\Big\lceil \frac{b_1}{x_1} \Big\rceil, \Big\lceil \frac{b_2}{x_2} \Big\rceil\Big\},\ \Big\lceil \frac{b_1 - d\cdot x_1}{y_1} \Big\rceil = \Big\lceil \frac{b_2 - d\cdot x_2}{y_2} \Big\rceil \,\} \Big).$$

It is $f : \mathbb{N}_0^6 \to \mathbb{N}_0, (\mathbf{u}_1,\boldsymbol{\tau}_1,b_1,\mathbf{u}_2,\boldsymbol{\tau}_2,b_2) \mapsto$

$$\begin{cases} \Big\lceil \frac{b_1}{\mathbf{u}_1} \Big\rceil + \Big\lceil \frac{b_2}{\mathbf{u}_2} \Big\rceil + \max\Big\{\boldsymbol{\tau}_2, \boldsymbol{\tau}_1 - \Big\lceil \frac{b_2}{\mathbf{u}_2} \Big\rceil\Big\} - 1, & \boldsymbol{\tau}_1 \ge \boldsymbol{\tau}_2, \\ f(\mathbf{u}_2,\boldsymbol{\tau}_2,b_2,\mathbf{u}_1,\boldsymbol{\tau}_1,b_1), & \textit{otherwise.} \end{cases}$$

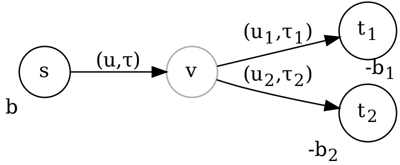


Figure: Network with labels representing (u, τ) and b.

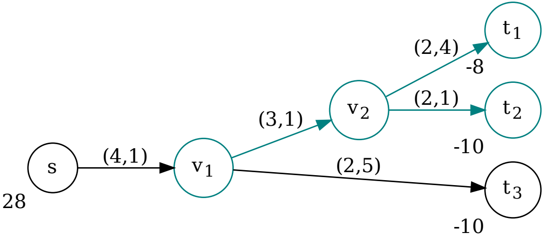
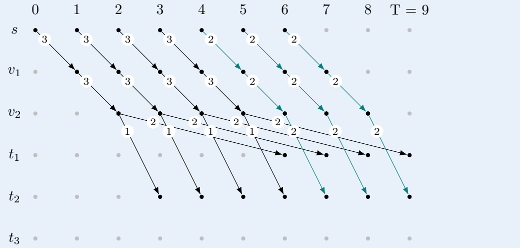
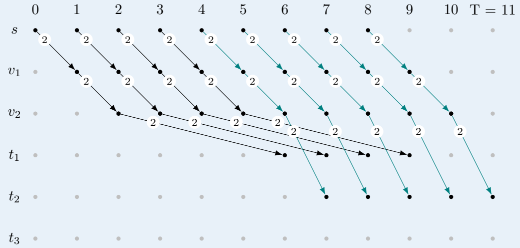


Figure: Network with labels representing (u,τ) and b.

Optimal Solution for Subgraph



Non-optimal Solution



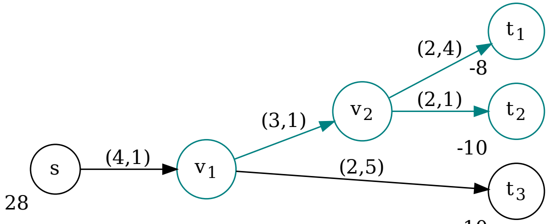
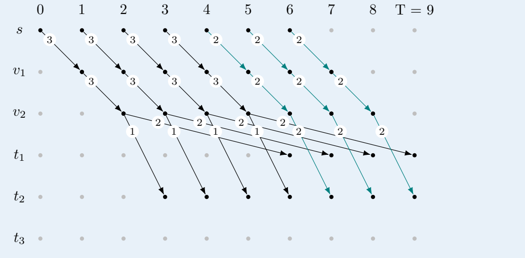
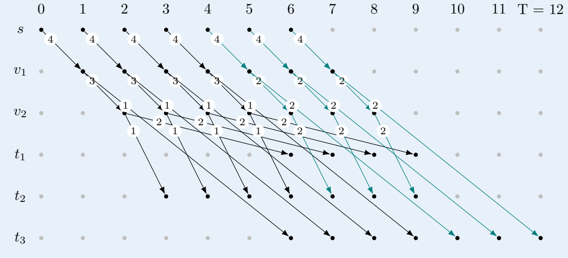


Figure: Network with labels representing (u, τ) and b.

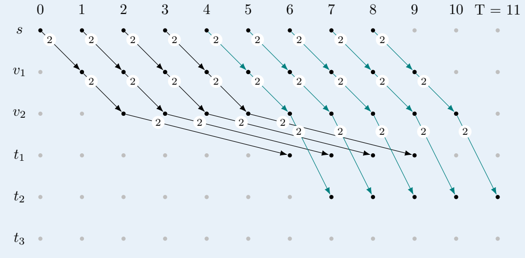
Optimal Solution for Subgraph



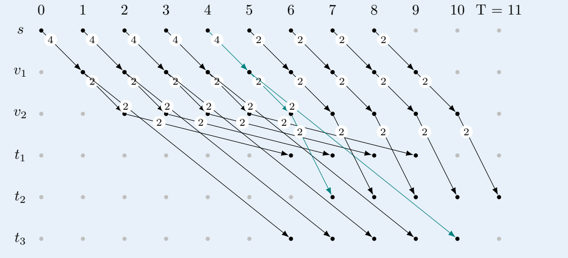
Extension of Optimal Solution



Non-optimal Solution



Extension of Non-optimal Solution



1-Uniform Flow with Minimal Time Horizon for Tree Network

$$\min \quad d \quad \text{s.t.} \quad \sum_{i \in I_a} -b_i \leq d \cdot u(a), \quad a \in A, I_a := \{i \in \{1, \ldots, h\} \mid a \in p_i\}, \quad d \in \mathbb{N}_0.$$

1-Uniform Flow with Minimal Time Horizon for Tree Network

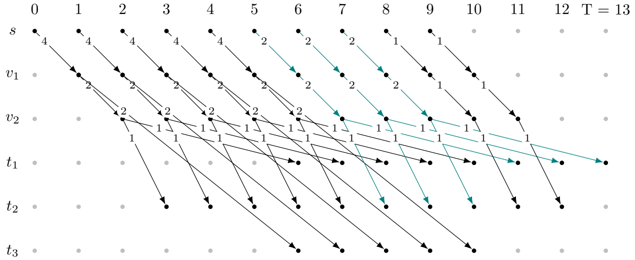
$$\min \quad d \quad \text{s.t.} \quad \sum_{i \in I_a} -b_i \leq d \cdot u(a), \quad a \in A, I_a := \{i \in \{1,\ldots,h\} \mid a \in p_i\}, \quad d \in \mathbb{N}_0.$$

$\Rightarrow$ Linear algorithm transforms an optimal solution of the *linear relaxation* into an optimal *load-consistent flow*.

1-Uniform Flow with Minimal Time Horizon for Tree Network

$$\min \quad d \quad \text{s.t.} \quad \sum_{i \in I_a} -b_i \leq d \cdot u(a), \quad a \in A, I_a := \{i \in \{1,\dots,h\} \mid a \in p_i\}, \quad d \in \mathbb{N}_0.$$

⇒ Linear algorithm transforms an optimal solution of the *linear relaxation* into an optimal *load-consistent flow*.



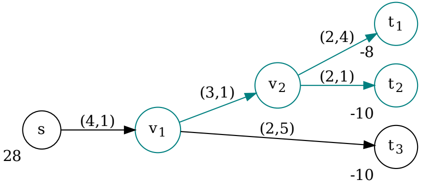


Table of Contents

Definition

Given a network (G,u,τ,c), source and sink $s,t \in V$, and a time horizon $T \in \mathbb{N}_0$, find an *integer temporally repeated flow* with

- maximal flow and
- minimized maximal costs over all points in time.

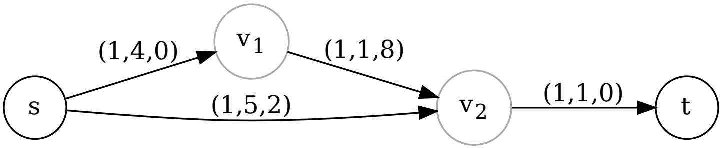


Figure: Network with labels representing (u,τ,c).

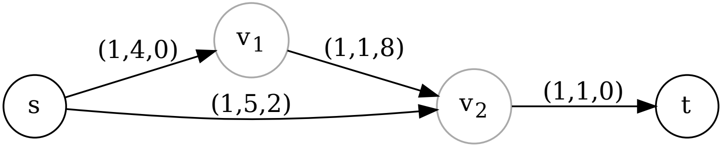
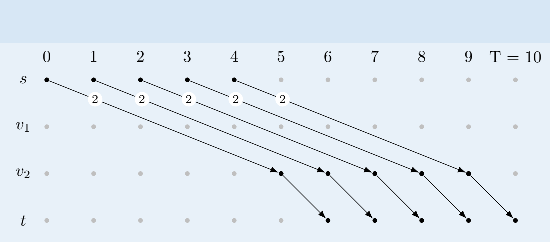
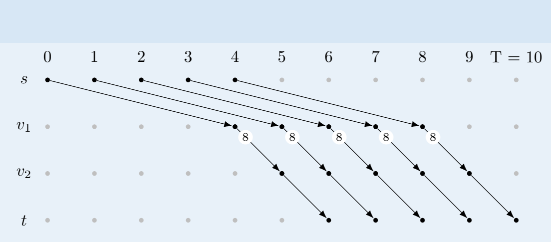


Figure: Network with labels representing (u, τ, c).





- Both flows have *total value* 5 and *time horizon* 10.
- The left flow has *total costs* 10, whereas the right flow has *total costs* 40.

Cost Minimization at each Point in Time

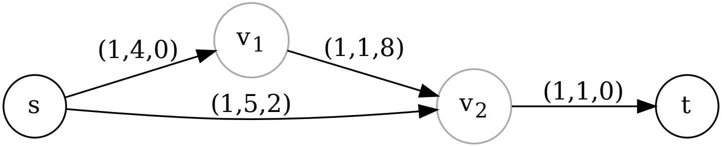
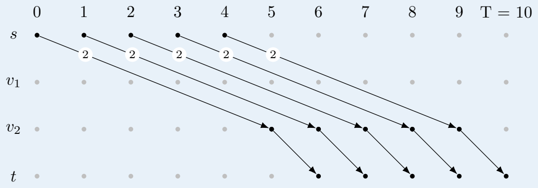
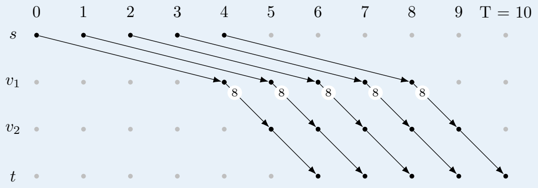


Figure: Network with labels representing (u, τ, c).

Maximal costs are 10 at time 4.



Maximal costs are 8 at time 4.



- Both flows have *total value* 5 and *time horizon* 10.
- The left flow has *total costs* 10, whereas the right flow has *total costs* 40.

Listing: Ford-Fulkerson Algorithm for Maximal (s,t)-Flows over Time

```
Input: Network (G,u,τ),
   source and sink s,t ∈ V, and
   time horizon T
Output: Temporally repeated flow f
   with time horizon T
   with maximal flow

Calculate static (s,t)-flow x
   that maximizes
   T · value(x) − ∑_{a∈A} τ(a) · x(a)
Calculate path decomposition
Construct temporally repeated flow f
   with time horizon T
Return f
```

Listing: Maximal (s,t)-Flows with Minimal Costs over all Points in Time

```
Input: Network (G,u,τ,c),
   source and sink s,t ∈ V, and
   time horizon T
Output: Temporally repeated flow f
   with time horizon T
   with maximal flow and minimized
   maximal costs at each time point

Set d : A → ℕ_0, a ↦ M · τ(a) + c(a) · τ(a)
Calculate static (s,t)-flow x
   that maximizes
   M · T · value(x) − ∑_{a∈A} d(a) · x(a)
Calculate path decomposition
Construct temporally repeated flow f
   with time horizon T
Return f
```

$$\begin{aligned} T \cdot \text{value}(x) - \sum_{a \in A} \tau(a) \cdot x(a) \quad &\leftrightarrow \quad M \cdot T \cdot \text{value}(x) - \sum_{a \in A} d(a) \cdot x(a) \\ &\Leftrightarrow \quad M \cdot T \cdot \text{value}(x) - \sum_{a \in A} \big(M \cdot \tau(a) + c(a) \cdot \tau(a)\big) \cdot x(a) \\ &\Leftrightarrow \quad M \cdot \underbrace{\Big(T \cdot \text{value}(x) - \sum_{a \in A} \tau(a) \cdot x(a)\Big)}_{\text{Maximize first.}} \underbrace{- \sum_{a \in A} c(a) \cdot \tau(a) \cdot x(a)}_{\text{Minimize second.}} \end{aligned}$$

Achievements

- Several types of flows over time

Future Work

Achievements

- Several types of flows over time
- Observation of tree networks and notion of equivalence

Future Work

Achievements

- Several types of flows over time
- Observation of tree networks and notion of equivalence
- Two linear algorithms that calculate non-optimal solutions of the QTP on Trees

Future Work

Achievements

- Several types of flows over time
- Observation of tree networks and notion of equivalence
- Two linear algorithms that calculate non-optimal solutions of the QTP on Trees
- Polynomial algorithm that maximizes value and minimizes costs

Future Work

Achievements

- Several types of flows over time
- Observation of tree networks and notion of equivalence
- Two linear algorithms that calculate non-optimal solutions of the QTP on Trees
- Polynomial algorithm that maximizes value and minimizes costs

Future Work

- Computation of equivalent almost-binary tree with minimal capacities

Achievements

- Several types of flows over time
- Observation of tree networks and notion of equivalence
- Two linear algorithms that calculate non-optimal solutions of the QTP on Trees
- Polynomial algorithm that maximizes value and minimizes costs

Future Work

- Computation of equivalent almost-binary tree with minimal capacities
- Analyze QTP on differently structured graphs

Achievements

- Several types of flows over time
- Observation of tree networks and notion of equivalence
- Two linear algorithms that calculate non-optimal solutions of the QTP on Trees
- Polynomial algorithm that maximizes value and minimizes costs

Future Work

- Computation of equivalent almost-binary tree with minimal capacities
- Analyze QTP on differently structured graphs
- Explore whether the solutions of subtrees for almost-binary tree networks can be restricted to a very small (of constant size?) set of relevant solutions

L. R. Ford Jr. and D. R. Fulkerson.
Constructing maximal dynamic flows from static flows.
Oper. Res., 6(3):419–433, 1958.

Horst W. Hamacher.
Temporally repeated flow algorithms for dynamic min cost flows.
In *Proceedings of the 28th IEEE Conference on Decision and Control,*, pages 1142–1146 vol.2, 1989.

Bruce Hoppe and Éva Tardos.
The quickest transshipment problem.
Math. Oper. Res., 25(1):36–62, 2000.

David P. Williamson.
Network Flow Algorithms.
Cambridge University Press, 2019.