Computing reachability under waiting-time constraints in linear time

Filippo Brunelli, Laurent Viennot

Tempogral Workshop

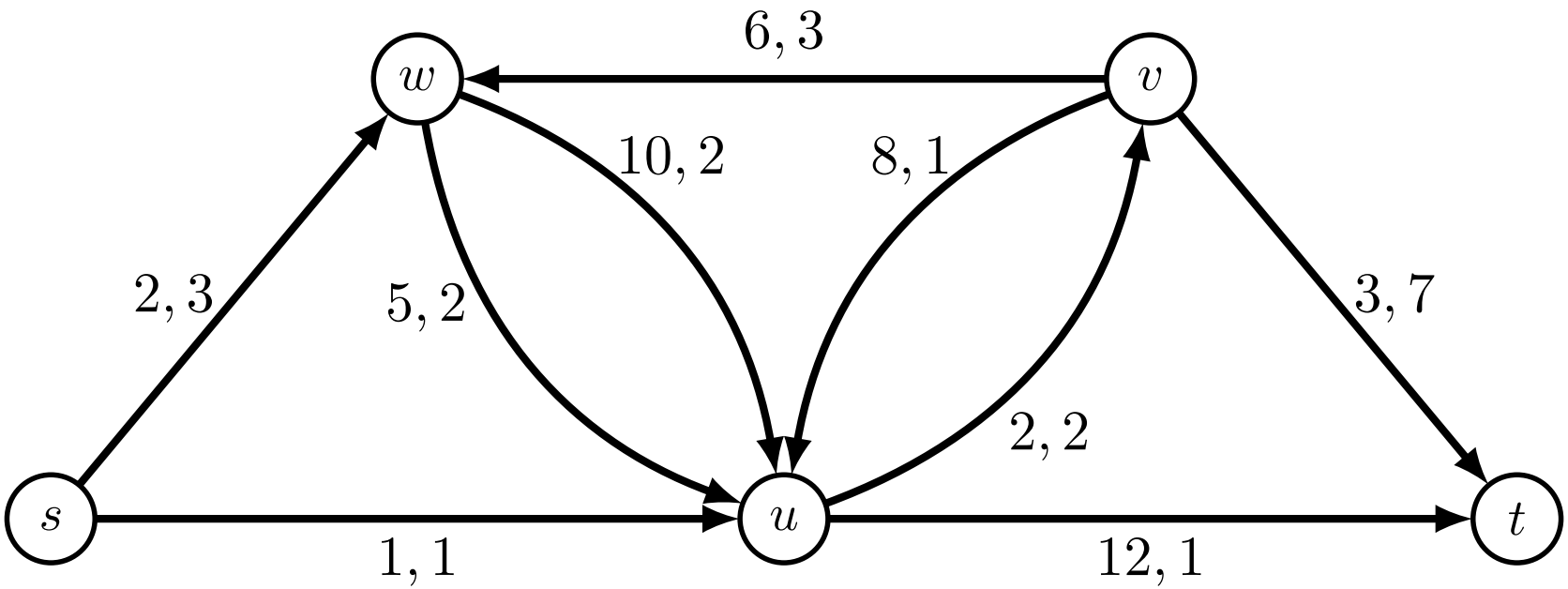
5 July 2023

Université Paris Cité

INSTITUT DE RECHERCHE EN INFORMATIQUE FONDAMENTALE

Inria

Temporal graph model



Temporal graph model

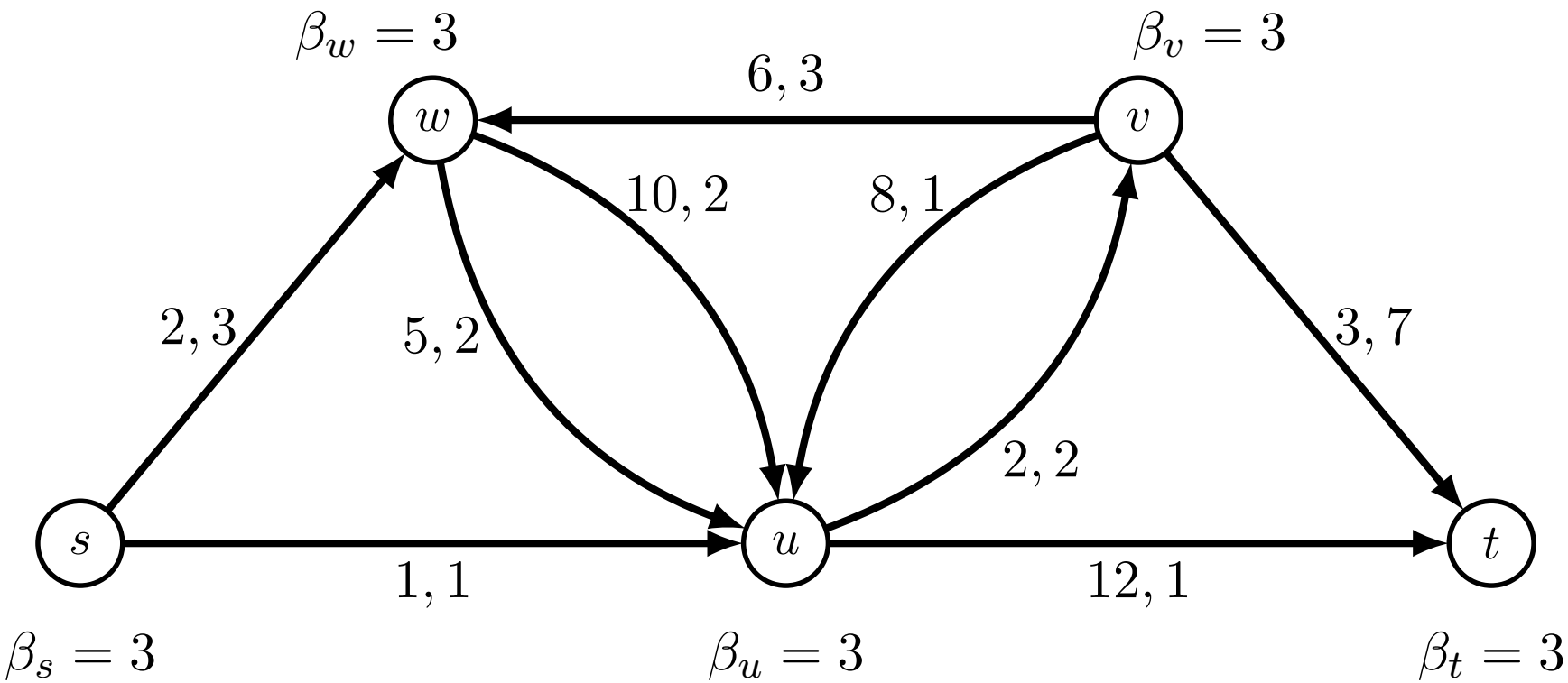
Temporal graph

A *temporal graph* is a list of quadruples $e = (u, v, \tau, \lambda)$, called *temporal edges*, where:

- $u \in V$ is the *tail* of e,
- $v \in V$ is the *head* of e,
- $\tau \in \mathbb{R}$ is the *departure time* of e,
- $\lambda \in \mathbb{R}_{>0}$ is the *travel time* of e.

We also define the *arrival time* of e as $\tau + \lambda$.

Maximum waiting time constraints



Temporal walks

Theorem (Casteigts, Himmel, Molter, Zschoche 2021)

In a temporal graph subject to waiting constraints, deciding whether there exists a temporal path between two nodes is NP-hard.

Temporal walks

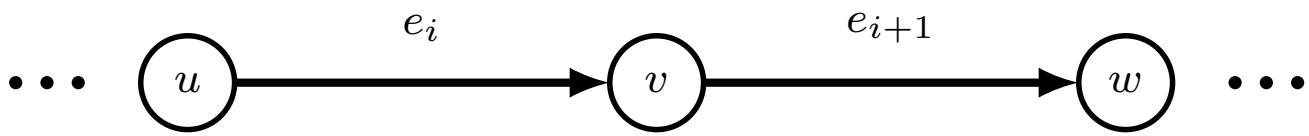
Theorem (Casteigts, Himmel, Molter, Zschoche 2021)

In a temporal graph subject to waiting constraints, deciding whether there exists a temporal path between two nodes is NP-hard.

This moves the focus on studying temporal walks.

Temporal graph model

- A *temporal walk* is a sequence of temporal edges, such that for any two consecutive:

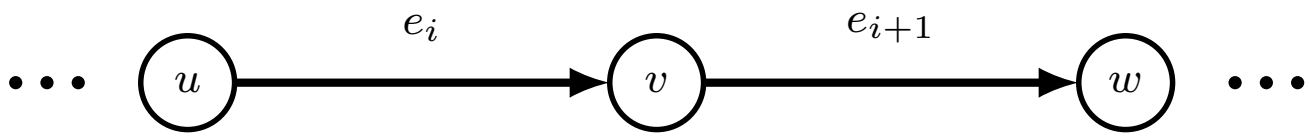


$$arr(e_i) \leq dep(e_{i+1})$$

- Temporal paths: A *temporal path* P is a walk in which all vertices are pairwise distinct.

Temporal graph model

- A *temporal walk* is a sequence of temporal edges, such that for any two consecutive:

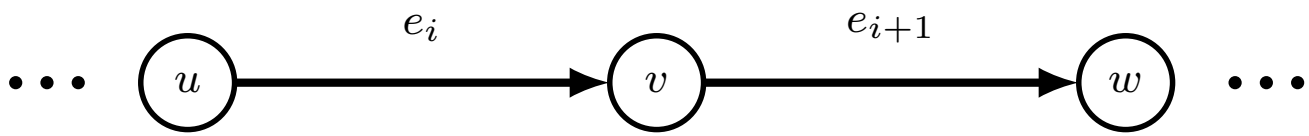


$$arr(e_i) \leq dep(e_{i+1}) \leq arr(e_i) + \beta_v$$

- Temporal paths: A *temporal path* P is a walk in which all vertices are pairwise distinct.

Temporal graph model

- A *temporal walk* is a sequence of temporal edges, such that for any two consecutive:

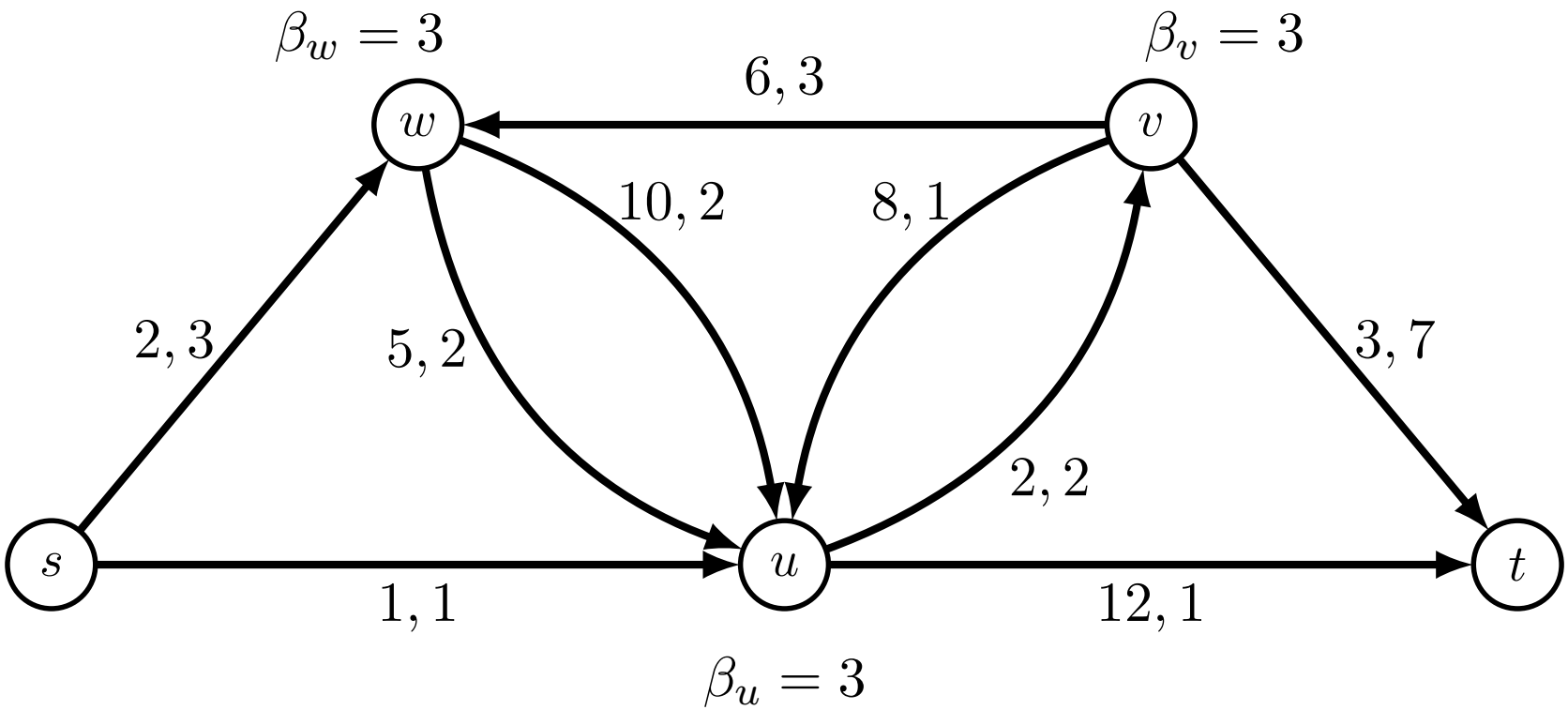


$$arr(e_i) \leq dep(e_{i+1}) \leq arr(e_i) + \beta_v$$

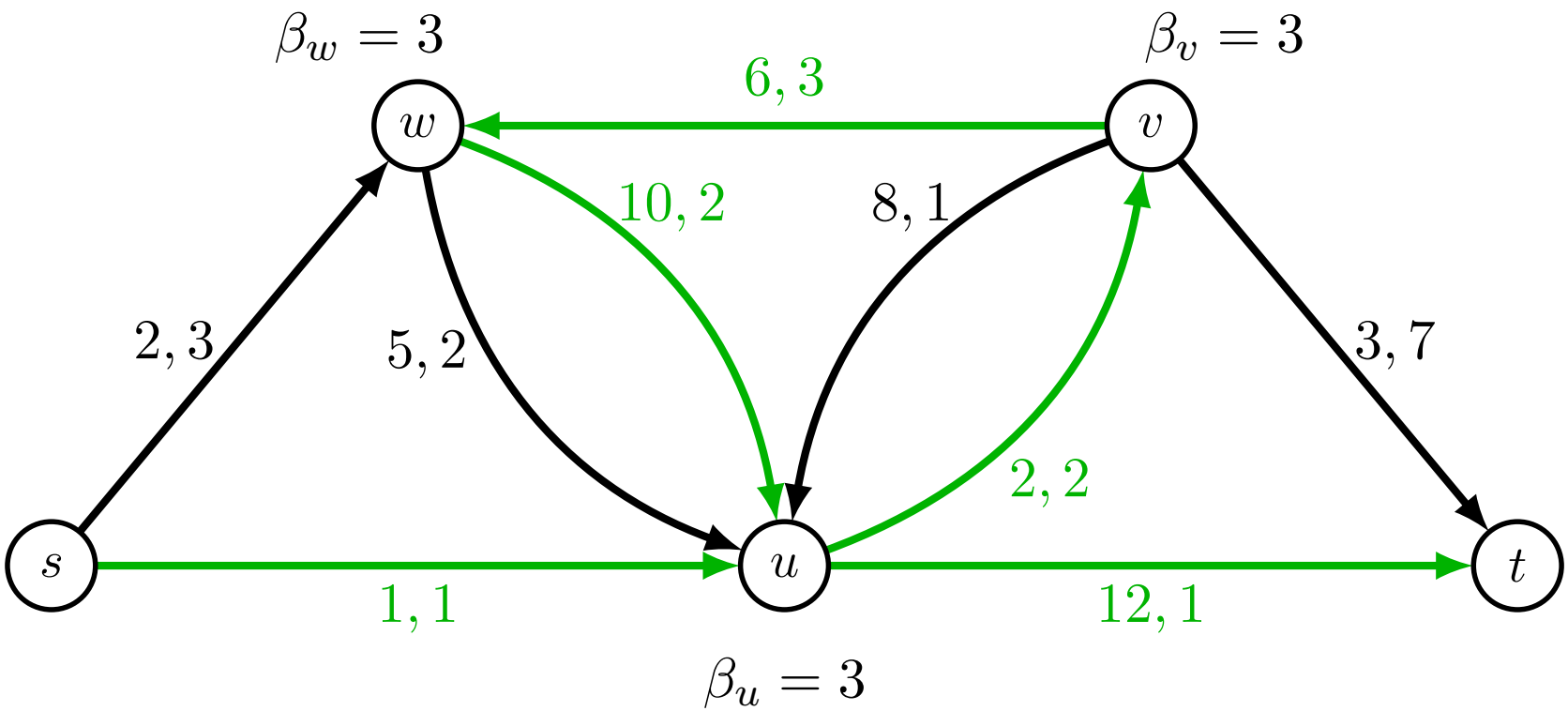
- Temporal paths: A *temporal path* P is a walk in which all vertices are pairwise distinct.

Notice that walks and paths are *strict* since the travel times are assumed strictly positive.

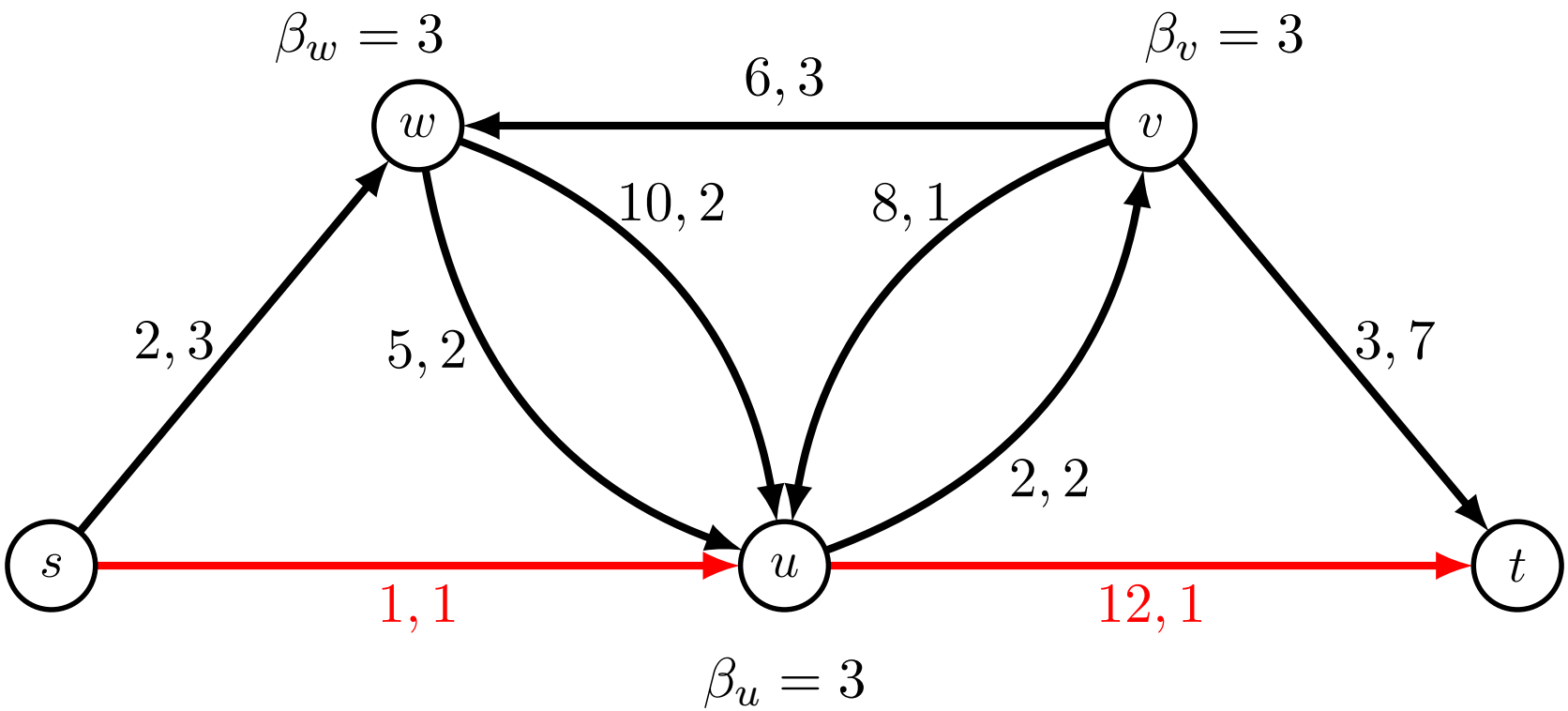
Example of temporal walks



A temporal walk



Not a temporal path



Definition of the reachability problem

A temporal edge e is s-reachable if there exists a temporal walk starting from s ending with e.

Single-source all-reachable-edges

Given a temporal graph with waiting constraints $G = (V, E, \beta)$ and a source node $s \in V$, compute all the s-reachable temporal edges.

Definition of the reachability problem

A temporal edge e is s-reachable if there exists a temporal walk starting from s ending with e.

Single-source all-reachable-edges

Given a temporal graph with waiting constraints $G = (V, E, \beta)$ and a source node $s \in V$, compute all the s-reachable temporal edges.

This in particular solves the *earliest arrival time* problem.

Definition of the reachability problem

A temporal edge e is s-reachable if there exists a temporal walk starting from s ending with e.

Single-source all-reachable-edges

Given a temporal graph with waiting constraints $G = (V, E, \beta)$ and a source node $s \in V$, compute all the s-reachable temporal edges.

This in particular solves the *earliest arrival time* problem.
State of the art:

- $O(M \log M)$ [**Bentert**, **Himmel**, **Nichterlein**, **Niedermeier**, **2020**]
- $O(M)$ [**Villacis-Llobet**, **Bui-Xuan**, **Potop-Butucaru**, **2022**]

For simplicity we will speak about the case in which all edges have strictly positive travel time ($\lambda > 0$).

Main result

A simple algorithm that solves the single-source all-reachable-edge in linear time and space. It receives as input a doubly-sorted representation of the temporal graph.

Temporal graph representation

Definition

A *doubly-sorted representation* of a temporal graph (V, E, β) consists in two lists E^{arr} and E^{dep}, where E^{arr} contains all temporal edges in E sorted by non-decreasing arrival time, and E^{dep} by non-decreasing departure time.

Temporal graph representation

Definition

A *doubly-sorted representation* of a temporal graph (V, E, β) consists in two lists E^{arr} and E^{dep}, where E^{arr} contains all temporal edges in E sorted by non-decreasing arrival time, and E^{dep} by non-decreasing departure time.

Theorem

Doubly-sorted and time-expanded are equivalent representations of temporal graphs.

Algorithm

Single-source all-reachable-edges

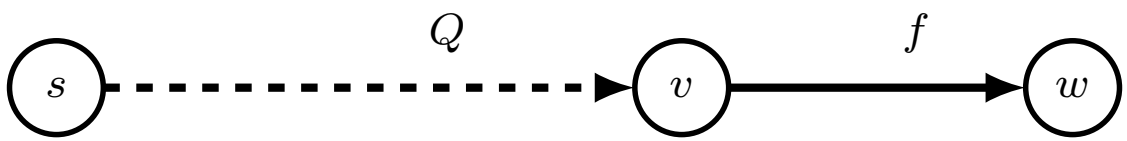
The problem consists in computing all the edges that are reachable (i.e. contained in a walk) from the source s.

Algorithm

Single-source all-reachable-edges

The problem consists in computing all the edges that are reachable (i.e. contained in a walk) from the source s.

Extending a walk: f extends Q if $Q.f$ is a walk.



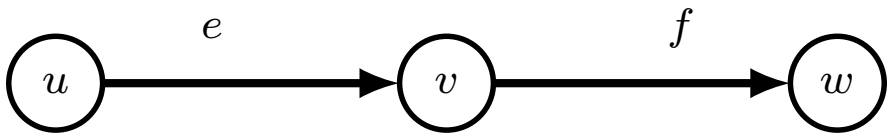
Goal: mark all edges that extend a walk from the source s.

Algorithm

Single-source all-reachable-edges

The problem consists in computing all the edges that are reachable (i.e. contained in a walk) from the source s.

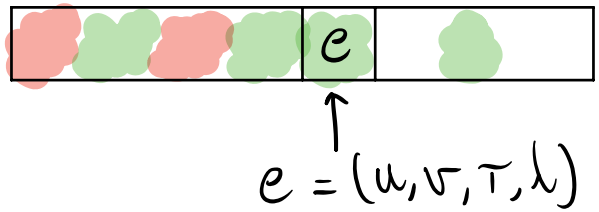
It matters just the last edge of the walk:



If e is s-reachable and f extends e, then f is s- reachable.

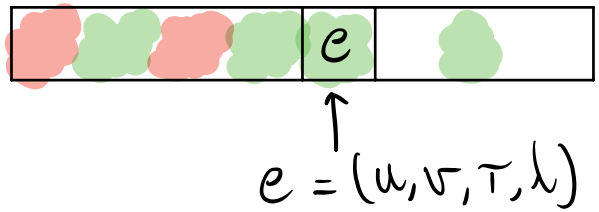
Algorithm overview

Linear scan of E^{arr}



Algorithm overview

Linear scan of E^{arr}

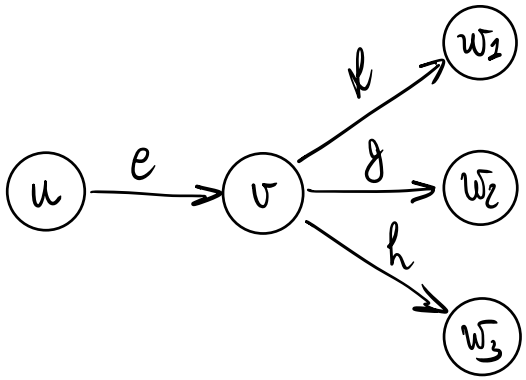


- We consider iteration k : $e = E^{arr}[k]$
- All edges that extend a walk in G_{k-1} have been marked

Algorithm overview

Linear scan of E^{arr}

$e = (u, v, \tau, \lambda)$

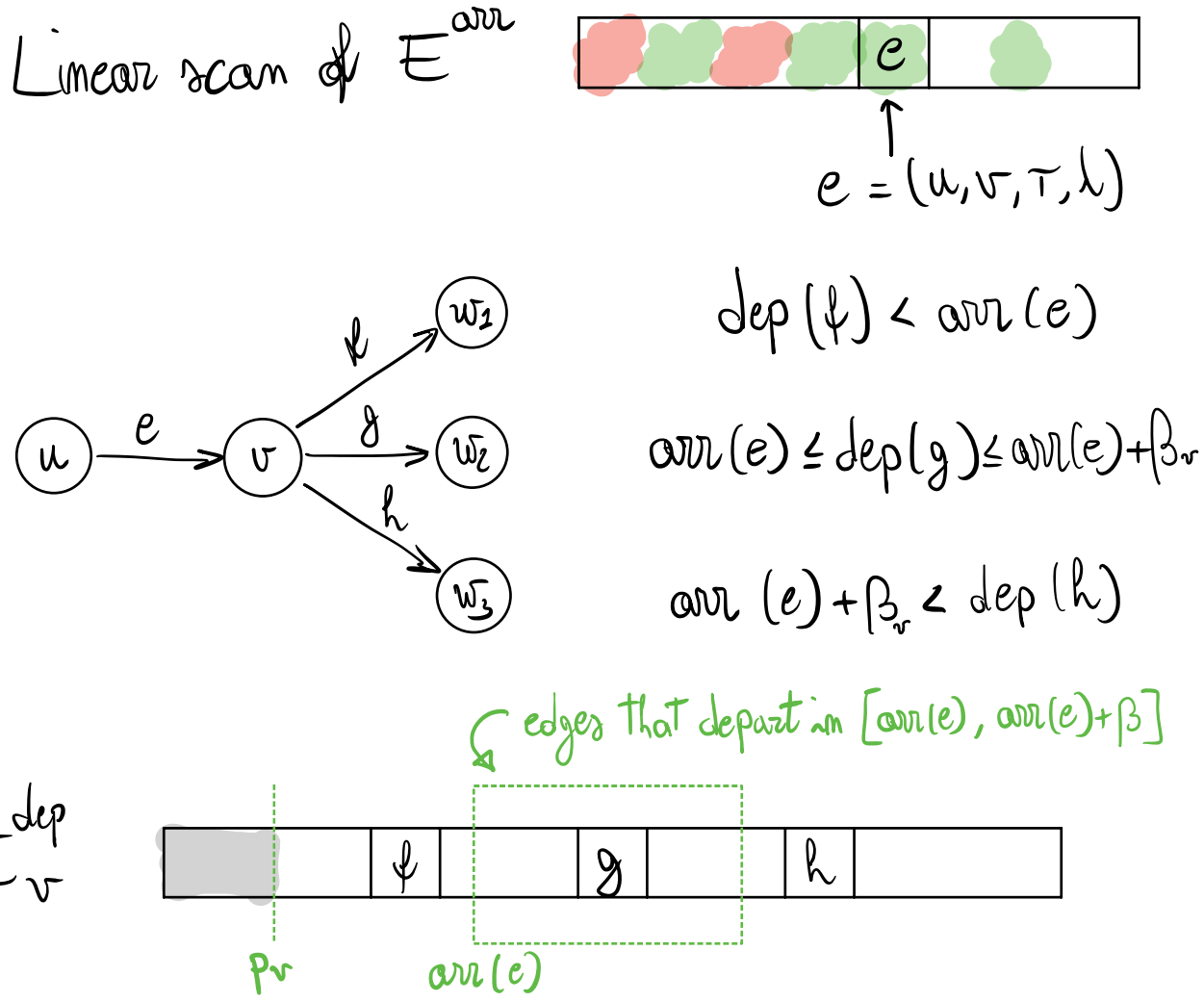


$dep(f) < arr(e)$

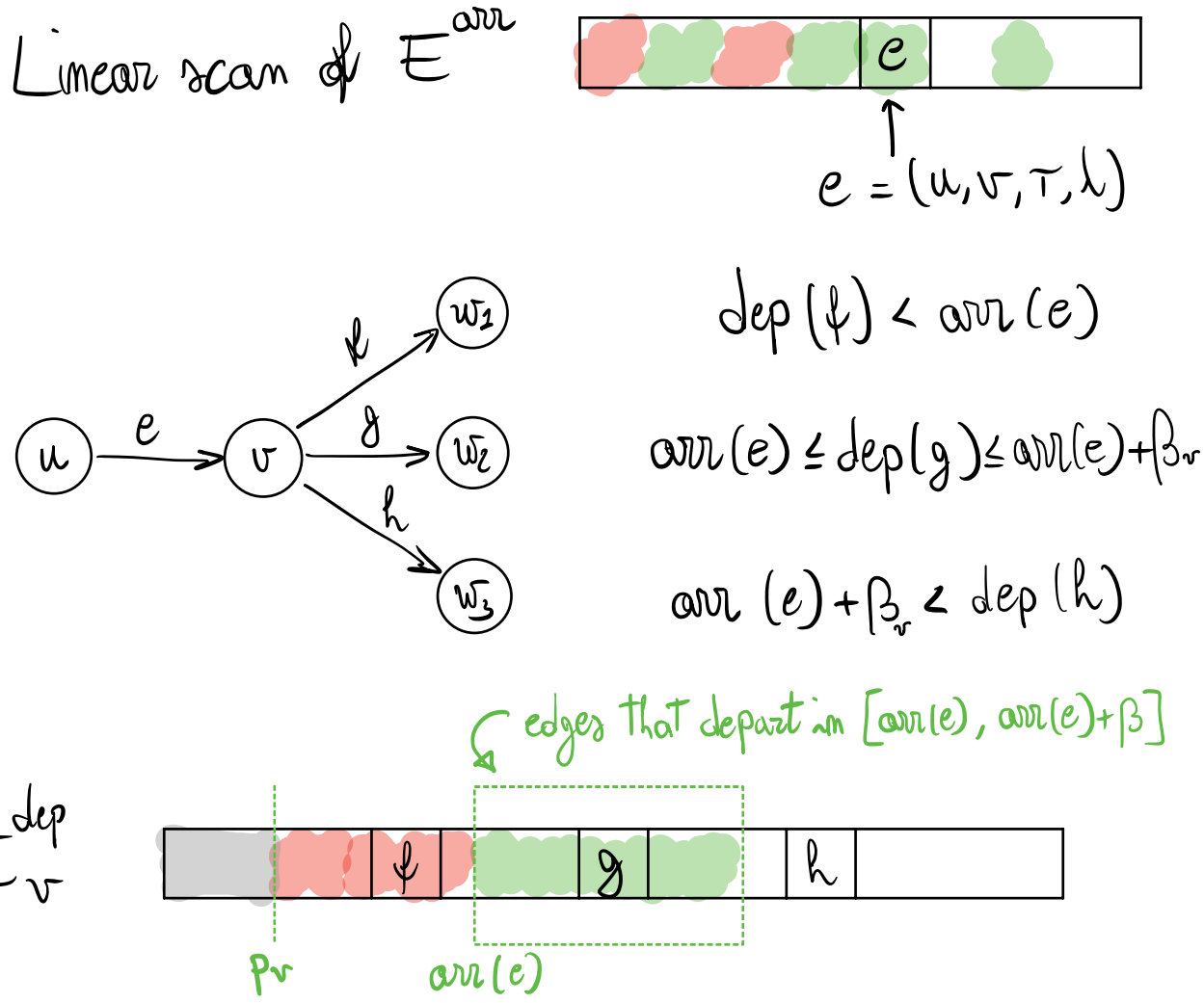
$arr(e) \leq dep(g) \leq arr(e) + \beta_v$

$arr(e) + \beta_v < dep(h)$

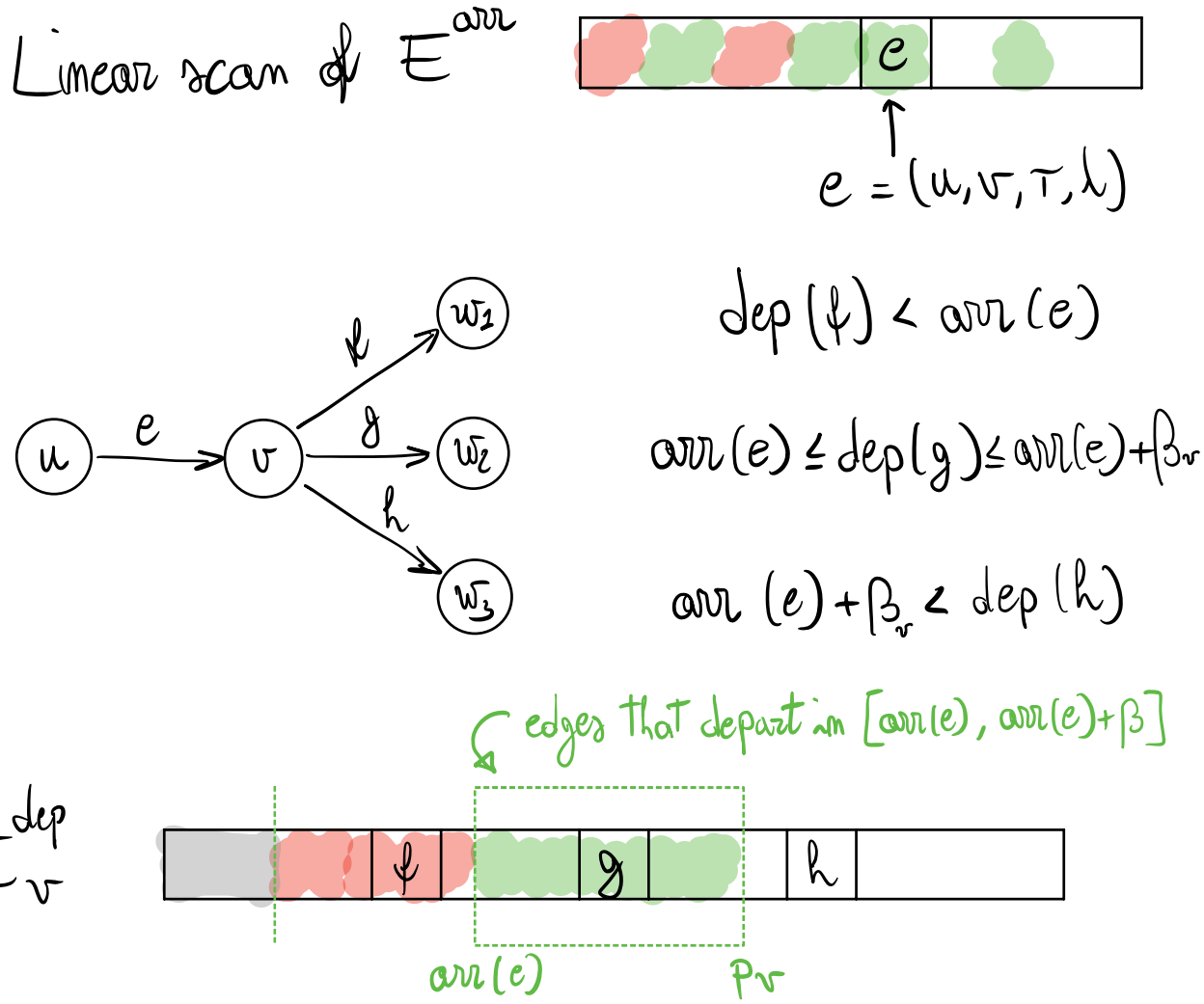
Algorithm overview



Algorithm overview



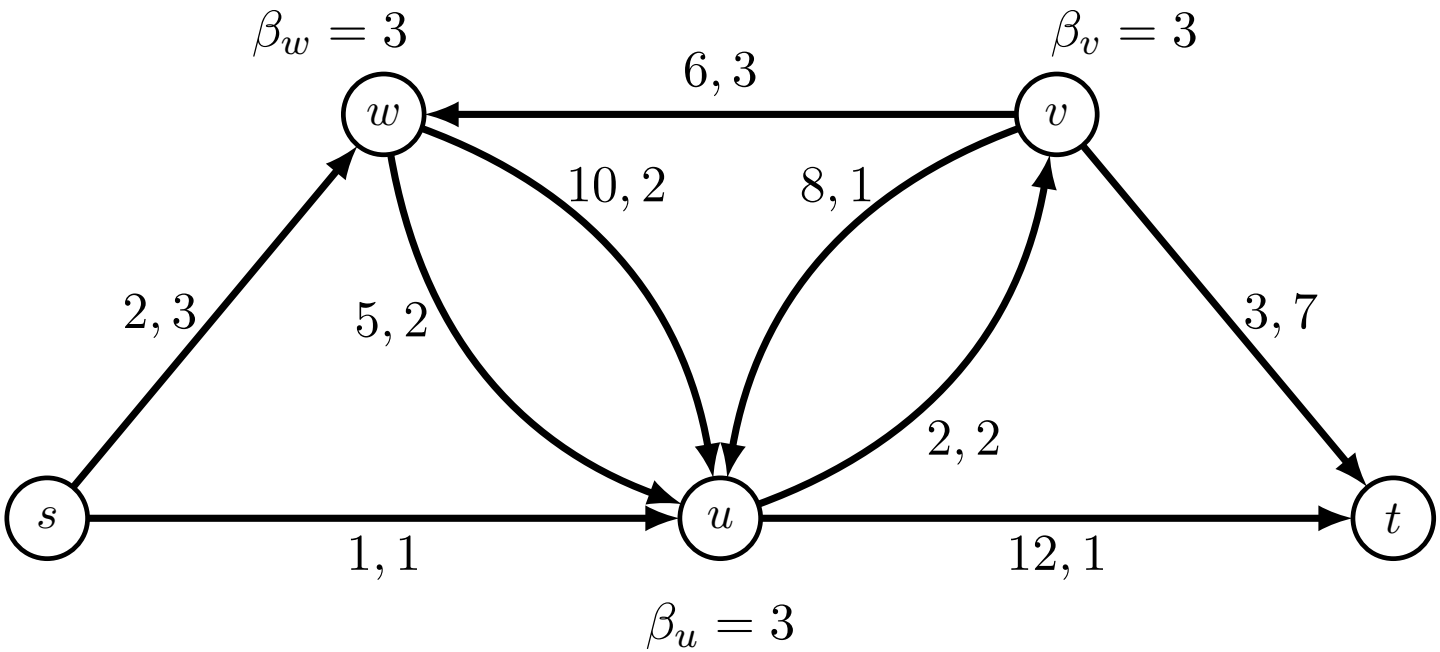
Algorithm overview



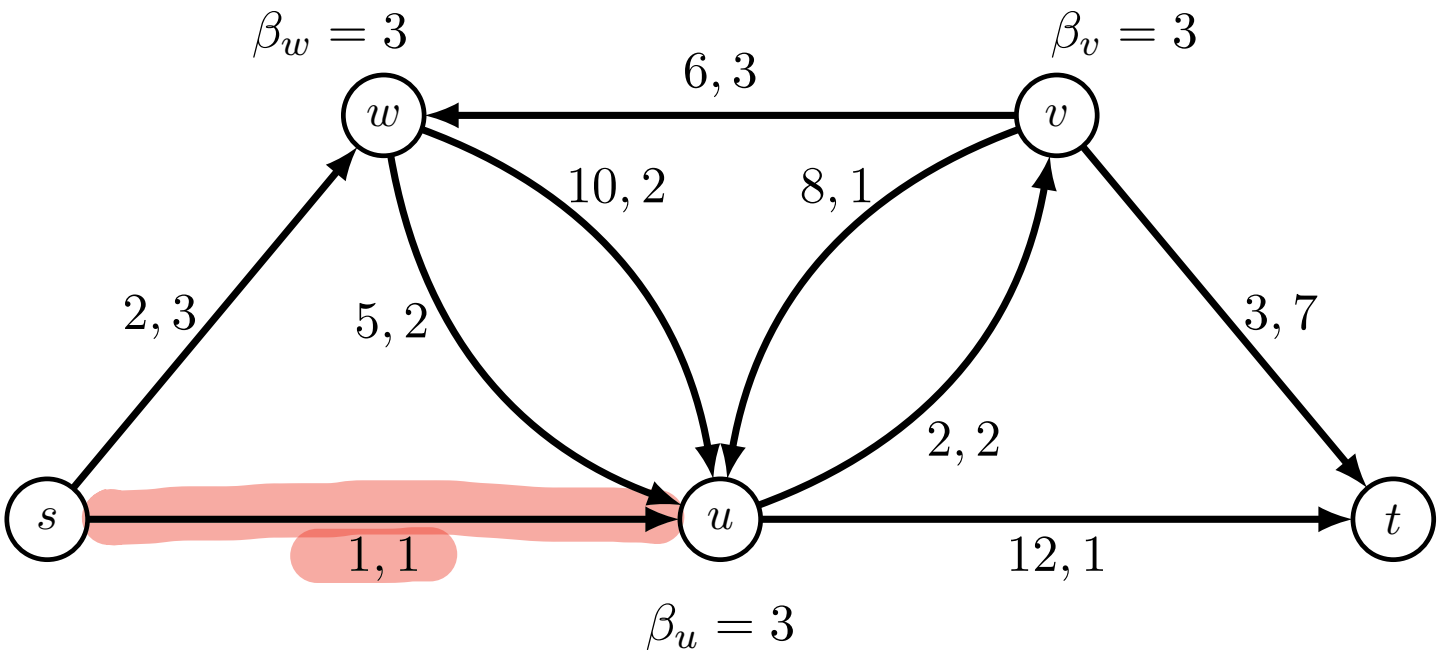
Input: A doubly-sorted representation (E^{arr}, E^{dep}).
Output: The sets $(A_v)_{v \in V}$ of s-reachable edges at each node v sorted by arrival time.

For each node v, generate the list E_v^{dep} by bucket sorting E^{dep}.
For *each node* v **do**
    Set $A_v := \emptyset$. `/* Set of` s`-reachable edges (as a sorted list). */`
    Set $p_v := 0$. `/* Index of the last processed edge in` E_v^{dep}`. */`
Set all the edges in E^{arr} as unmarked.
Set $P[e] := \perp$ for each edge $e \in E^{arr}$.`/* Parent of` e`, initially null. */`
For *each edge* $e = (u, v, \tau, \lambda)$ *in* E^{arr} **do**
    If $u = s$ *or* e *is marked* **then**
        `/*` e `is` s`-reachable. */`
        $A_v := A_v \cup \{e\}$
        Let $a = \tau + \lambda$ be the arrival time of e.
        `/* Process further edges from` v `until dep.time` $\geq a + \beta_v$`: */`
        Let $l > p_v$ be the first index of an edge $(v, w, \tau', \lambda') \in E_v^{dep}$ such that $\tau' \geq a$ (set $l := |E_v^{dep}| + 1$ if no such index exists).
        Let $r \geq l$ be the last index of an edge $(v, w, \tau', \lambda') \in E_v^{dep}$ such that $\tau' \leq a + \beta_v$ (set $r := l - 1$ if no such index exists).
        `/* Mark unmarked edges with dep.time in` $[a, a + \beta_v]$`: */`
        If $l \leq r$ **then** mark each edge $f \in E_v^{dep}[l : r]$ and set $P[f] := e$.
        Set $p_v := r$.
Return *the sets* $(A_v)_{v \in V}$.

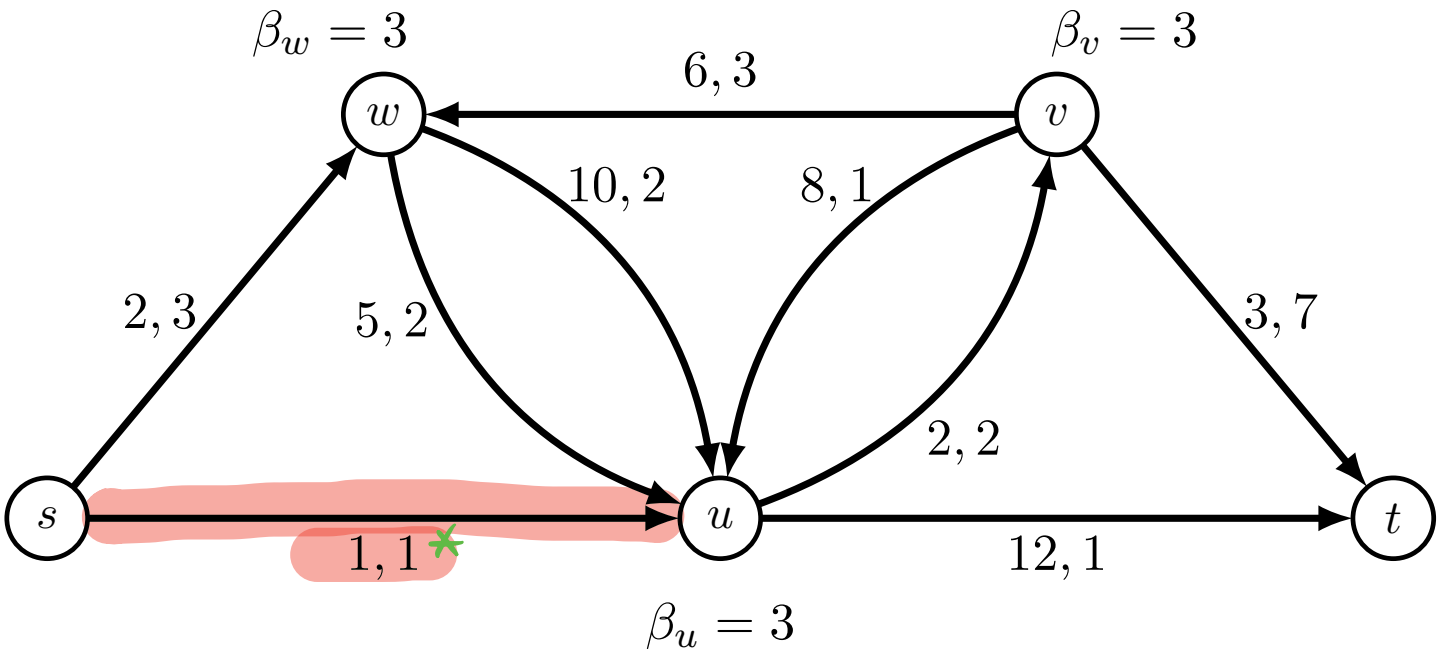
$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



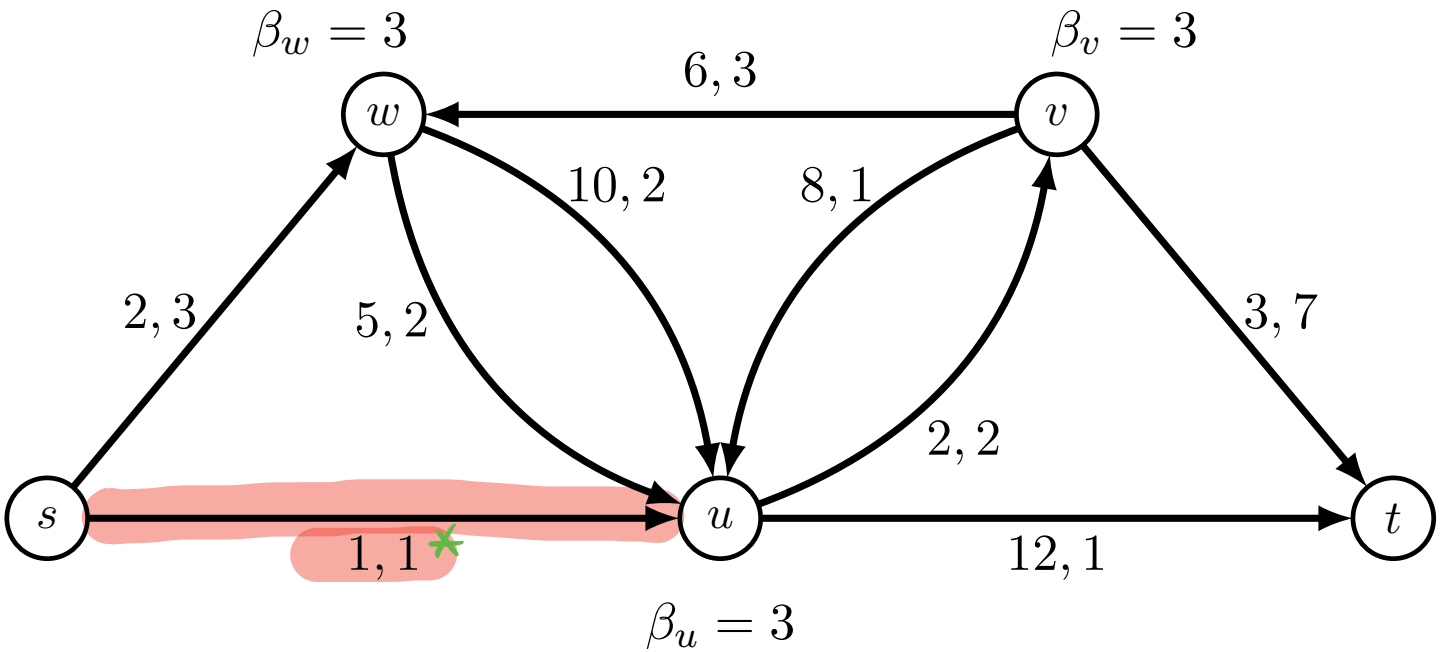
$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$



$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$

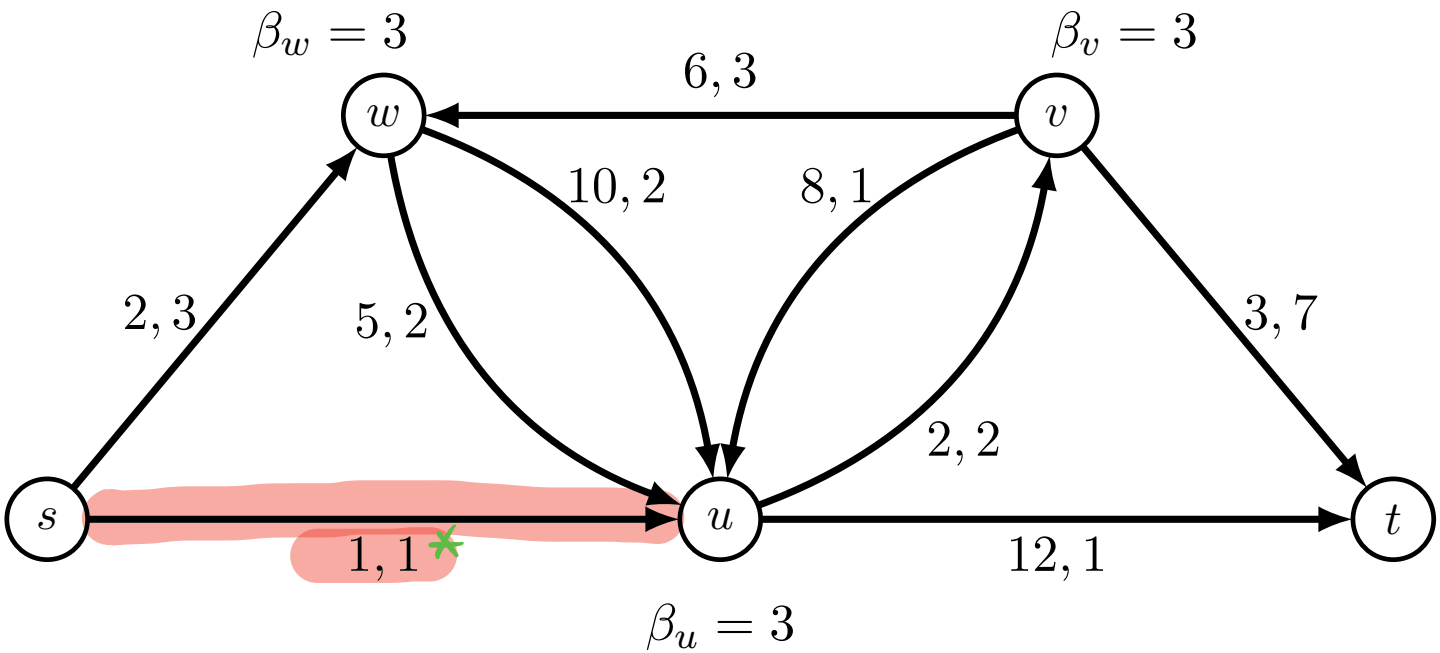


$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$



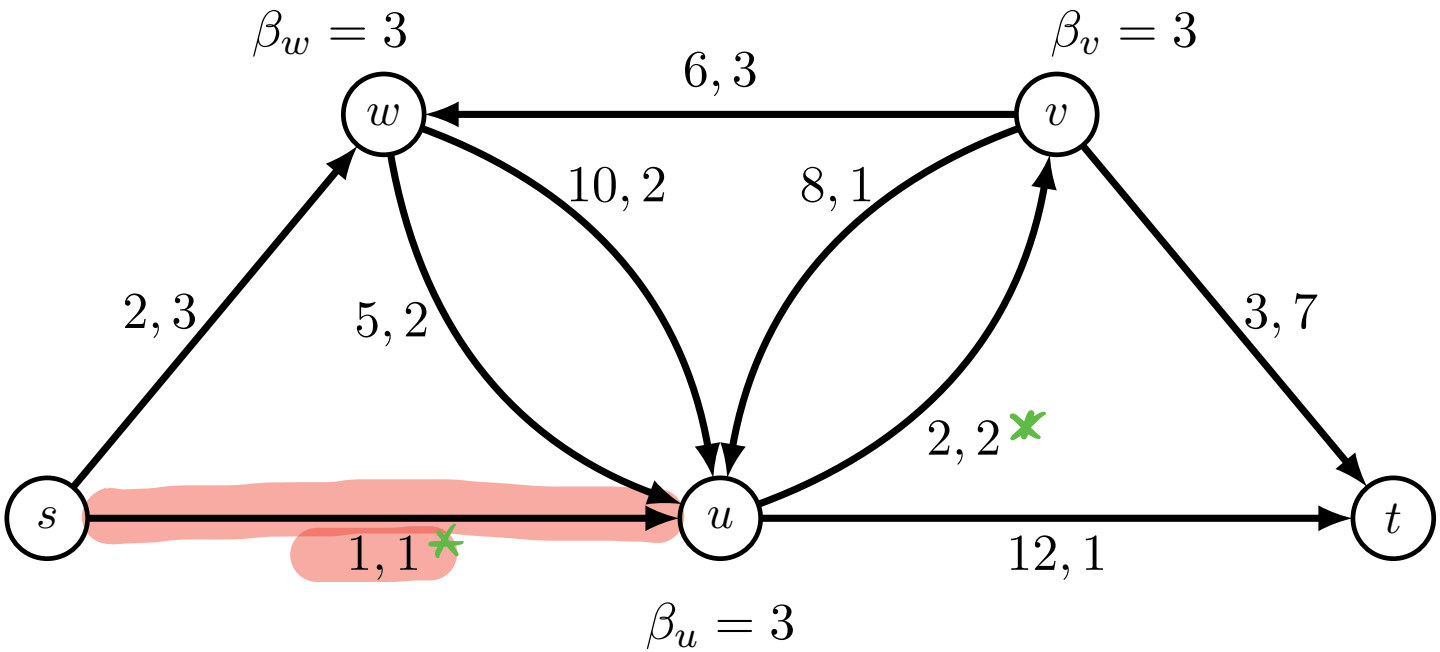
E_u^{dep} : $(u,v,2,2), (u,t,12,1)$, $p_u = 0$

$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$



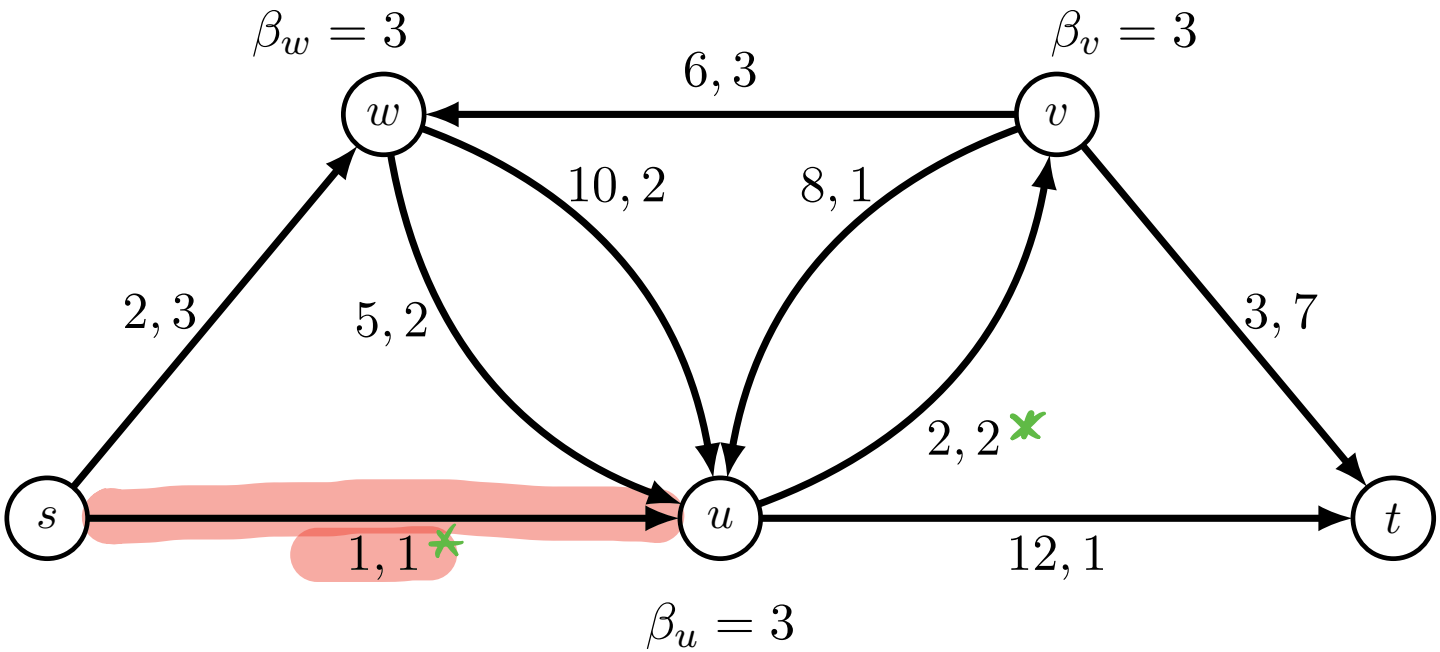
E_u^{dep} : $(u,v,2,2), (u,t,12,1)$, $p_u = 0$

$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$



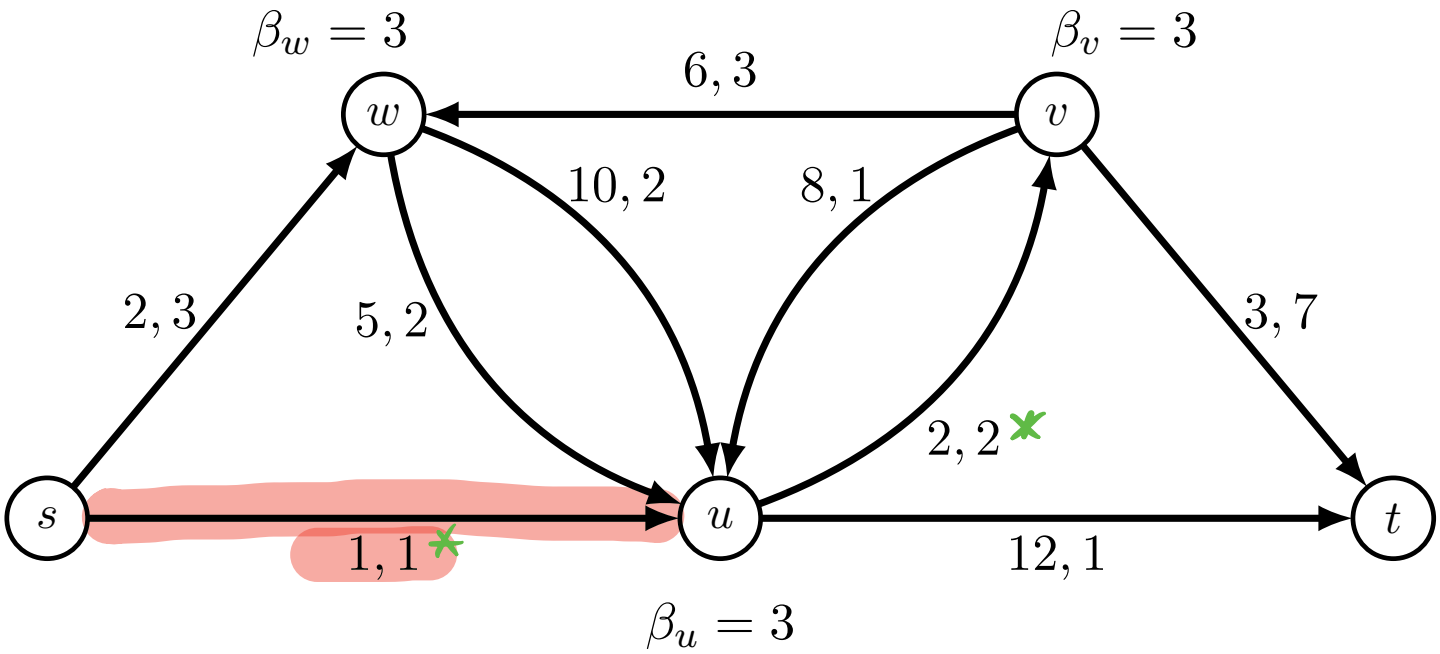
E_u^{dep} : $(u,v,2,2), (u,t,12,1)$, $p_u = 0$

$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$



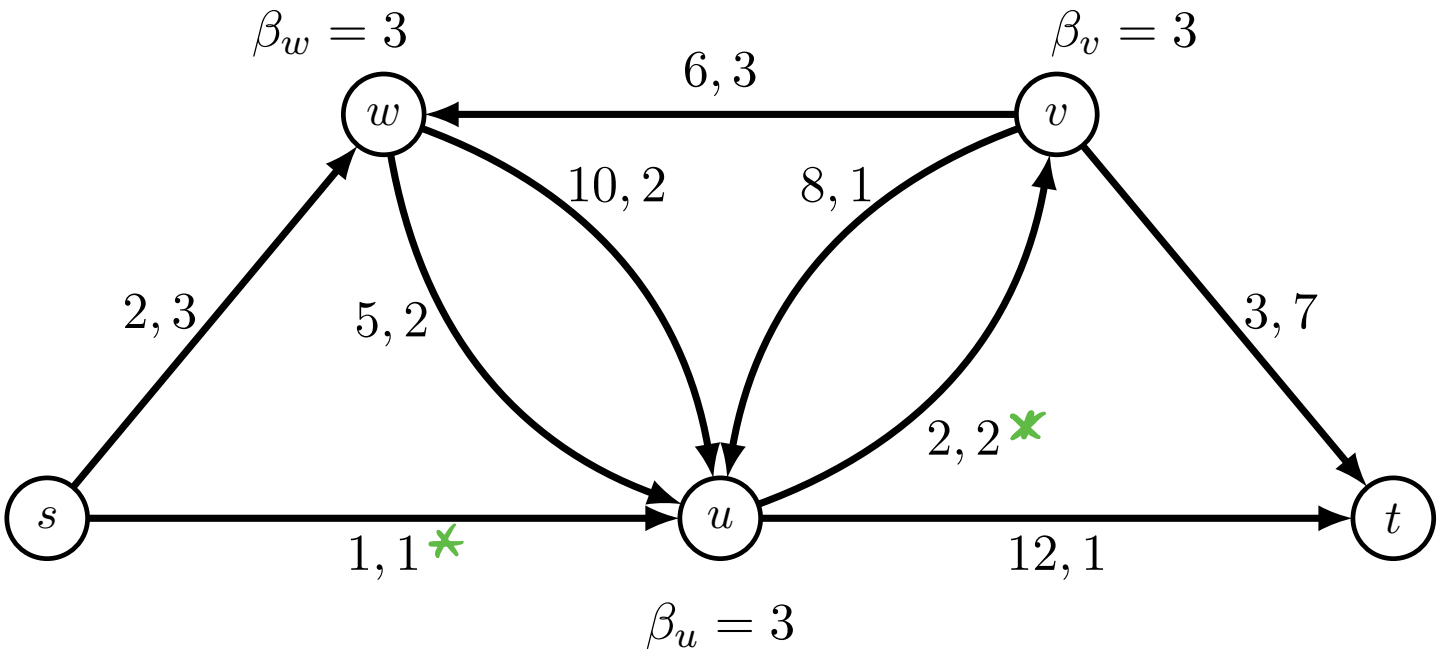
E_u^{dep} : $(u,v,2,2), (u,t,12,1)$, $p_u = 0$

$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$

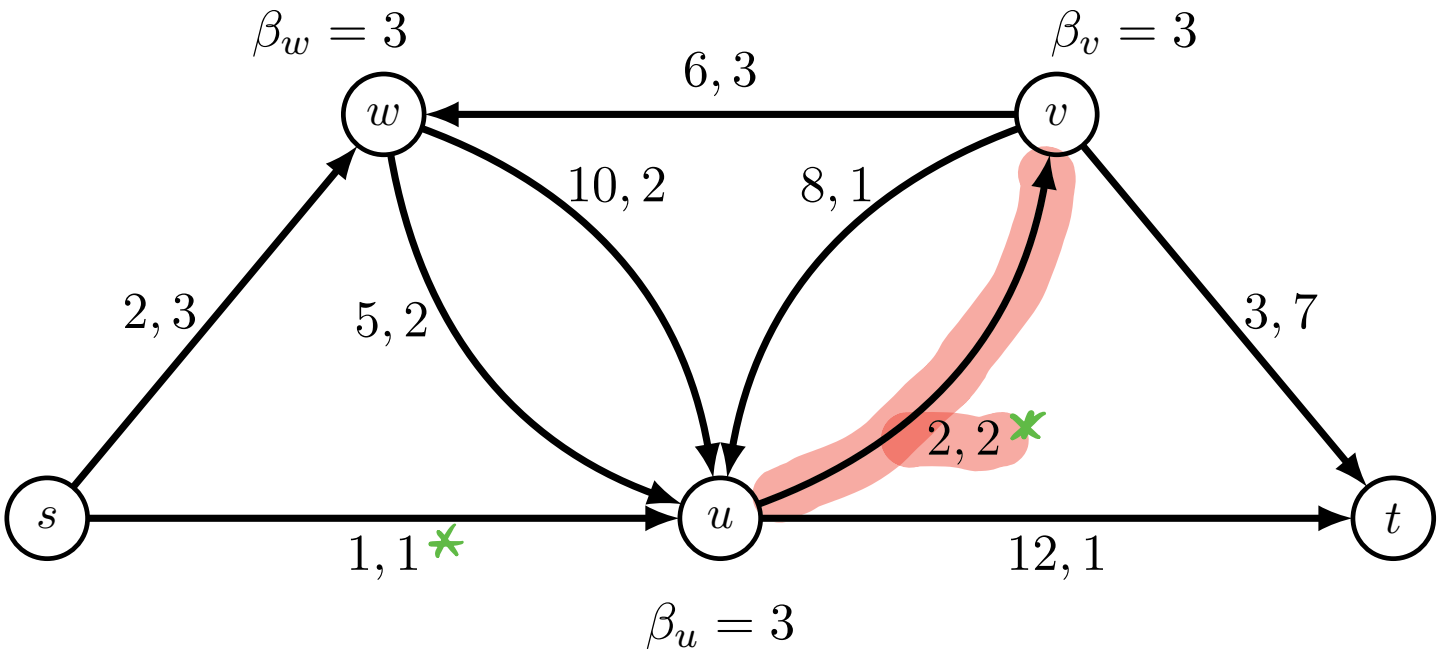


E_u^{dep} : $(u,v,2,2), (u,t,12,1)$, $P_u = 1$

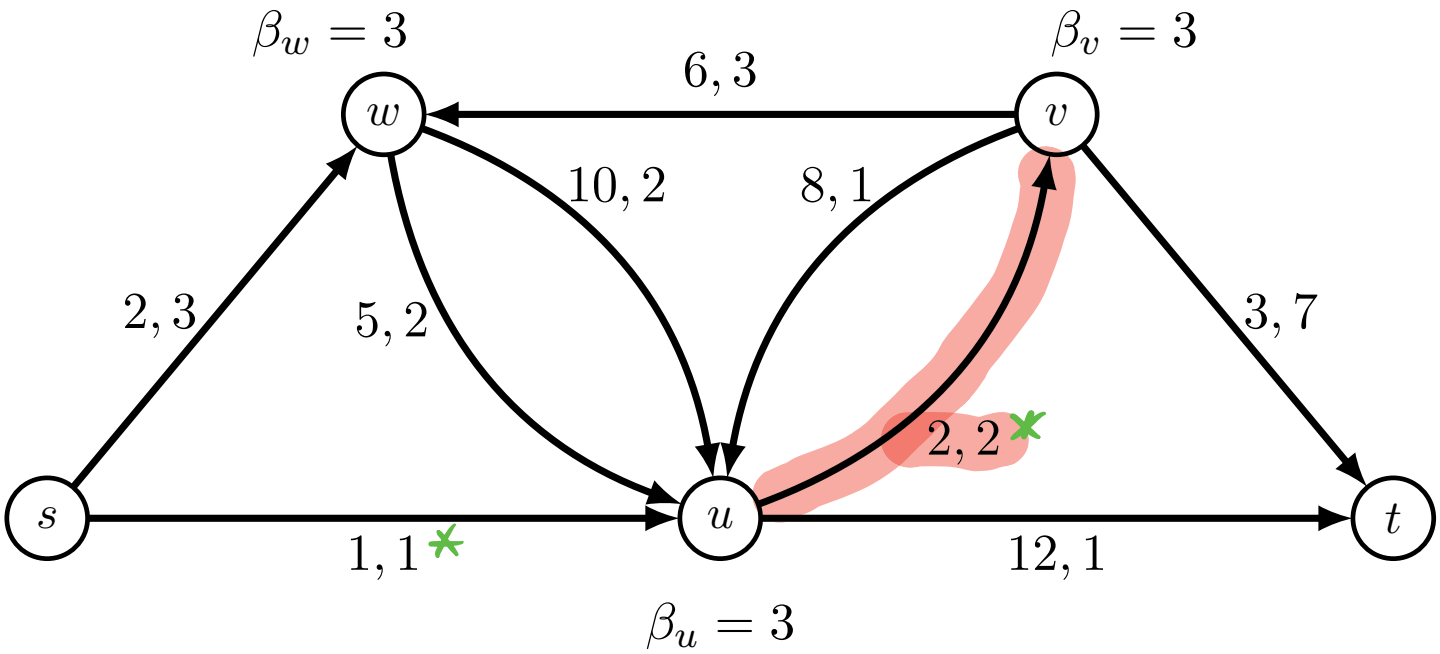
$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

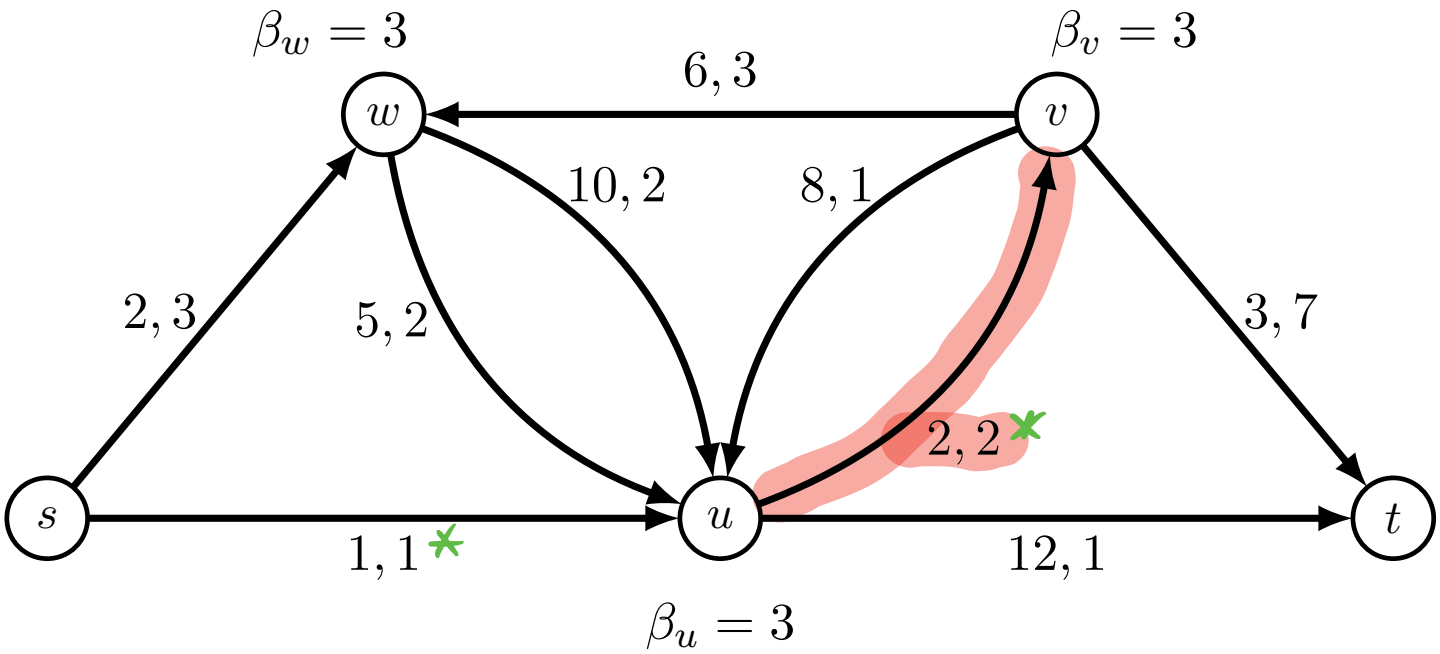


$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



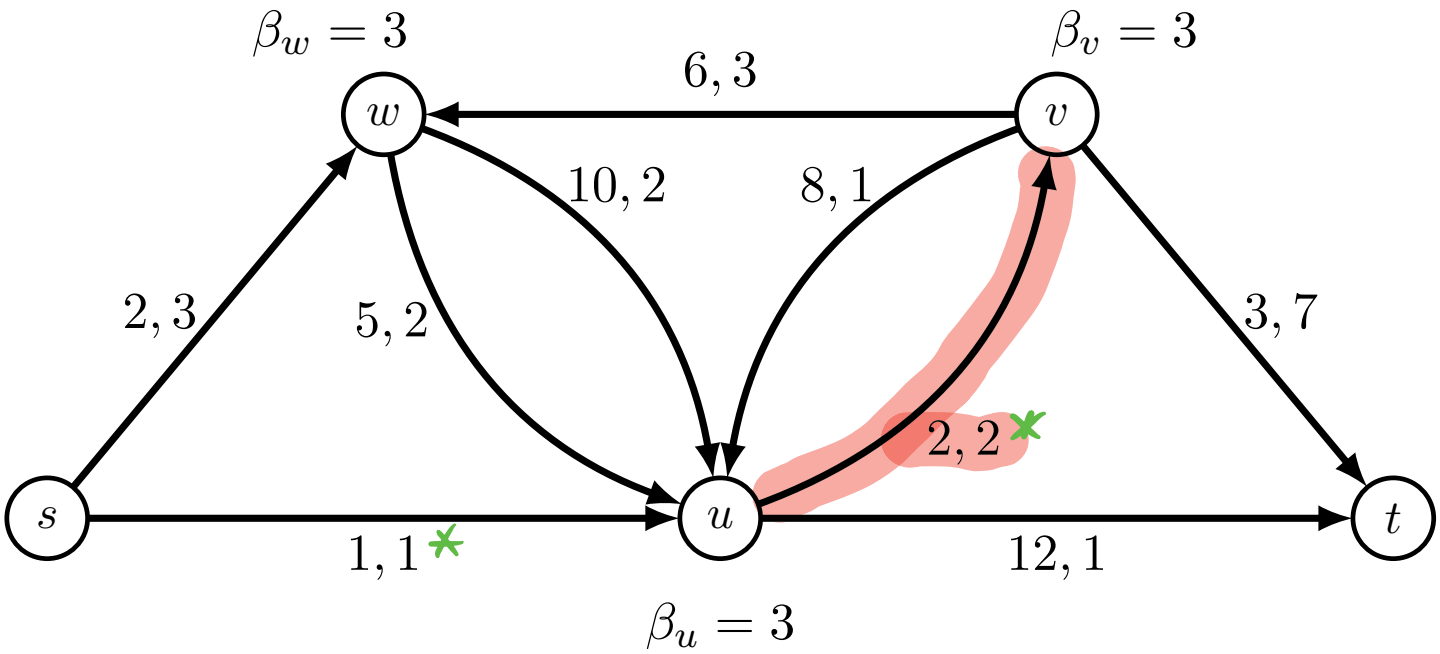
E_v^{dep} : $(v,t,3,7),(v,w,6,3),(v,u,8,1)$, $P_v = 0$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



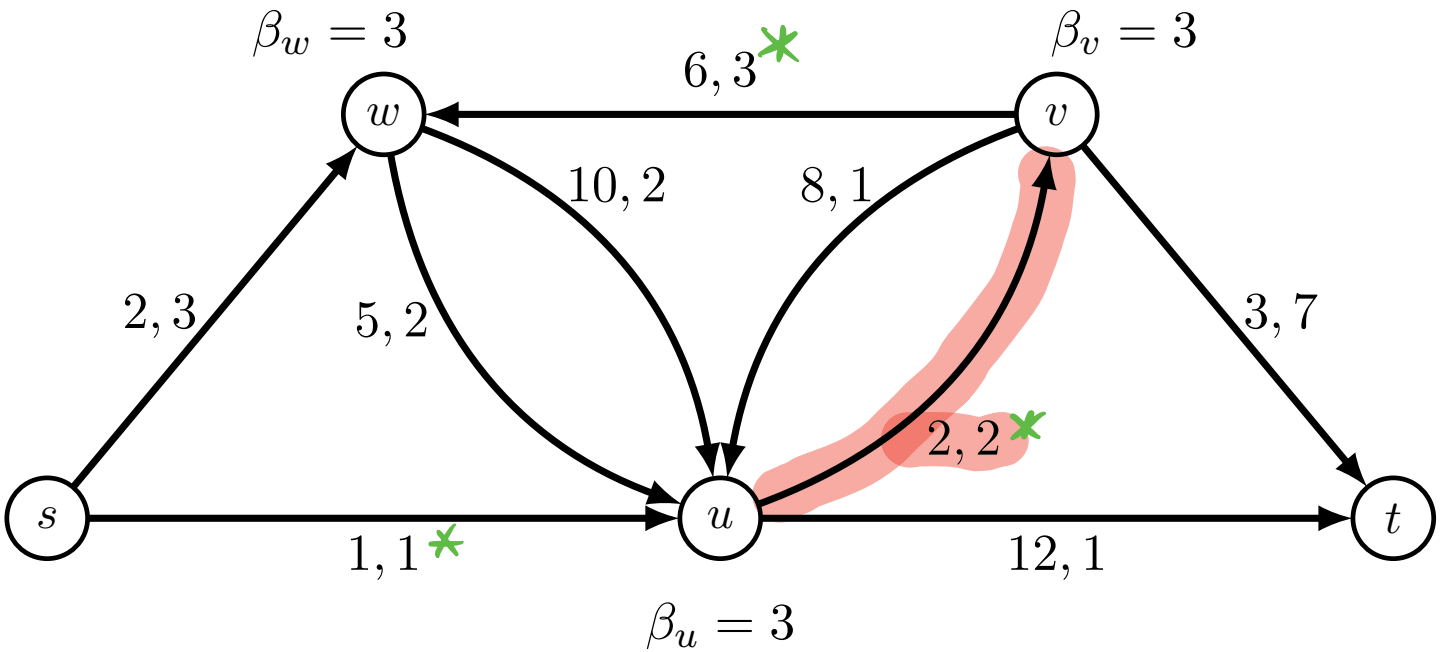
E_v^{dep} : $(v,t,3,7),(v,w,6,3),(v,u,8,1)$, $P_v = 0$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



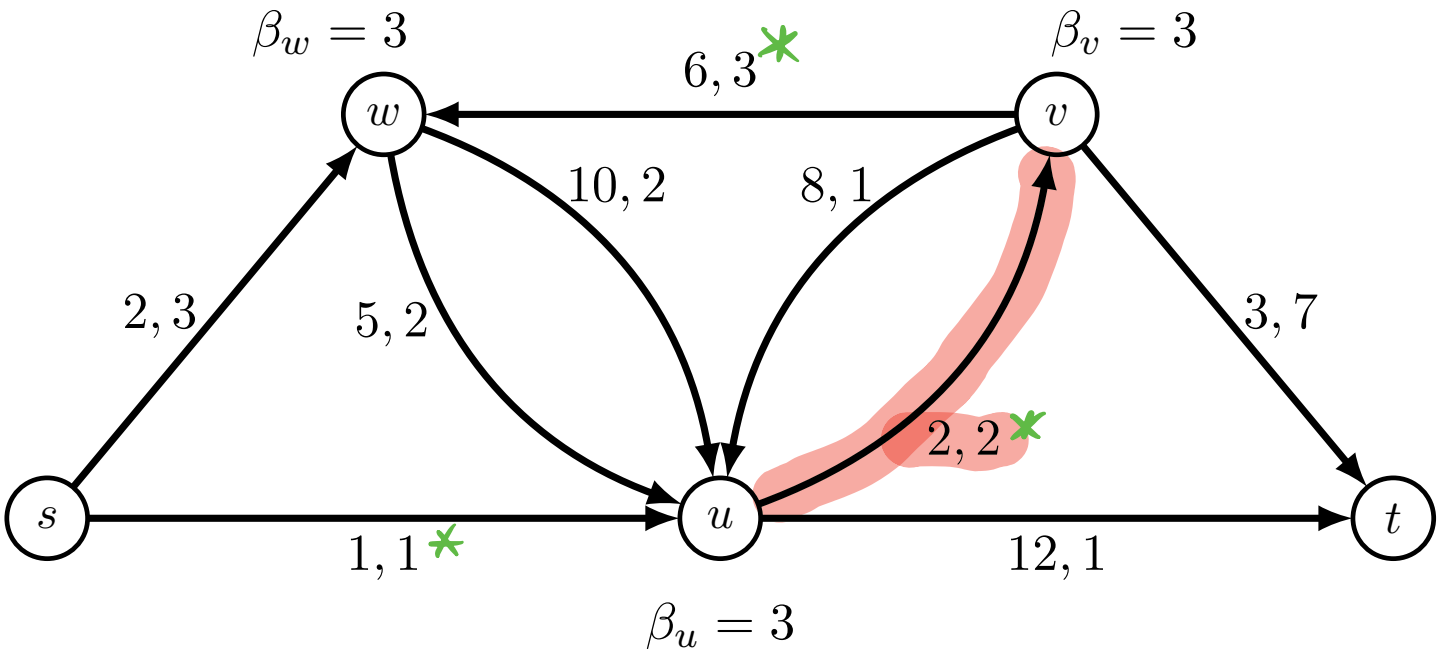
E_v^{dep} : $(v,t,3,7),(v,w,6,3),(v,u,8,1)$, $P_v = 0$

$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$



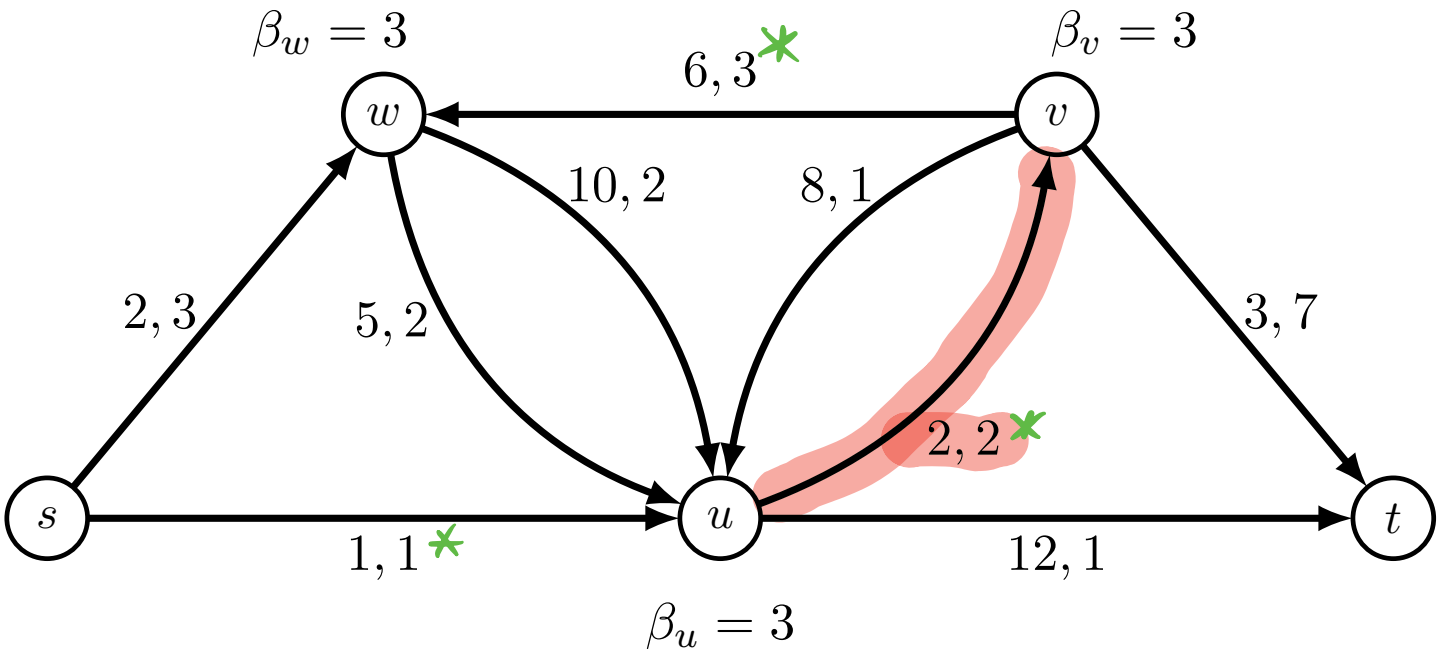
E_v^{dep} : $(v,t,3,7), (v,w,6,3), (v,u,8,1)$, $P_v = 0$

$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$



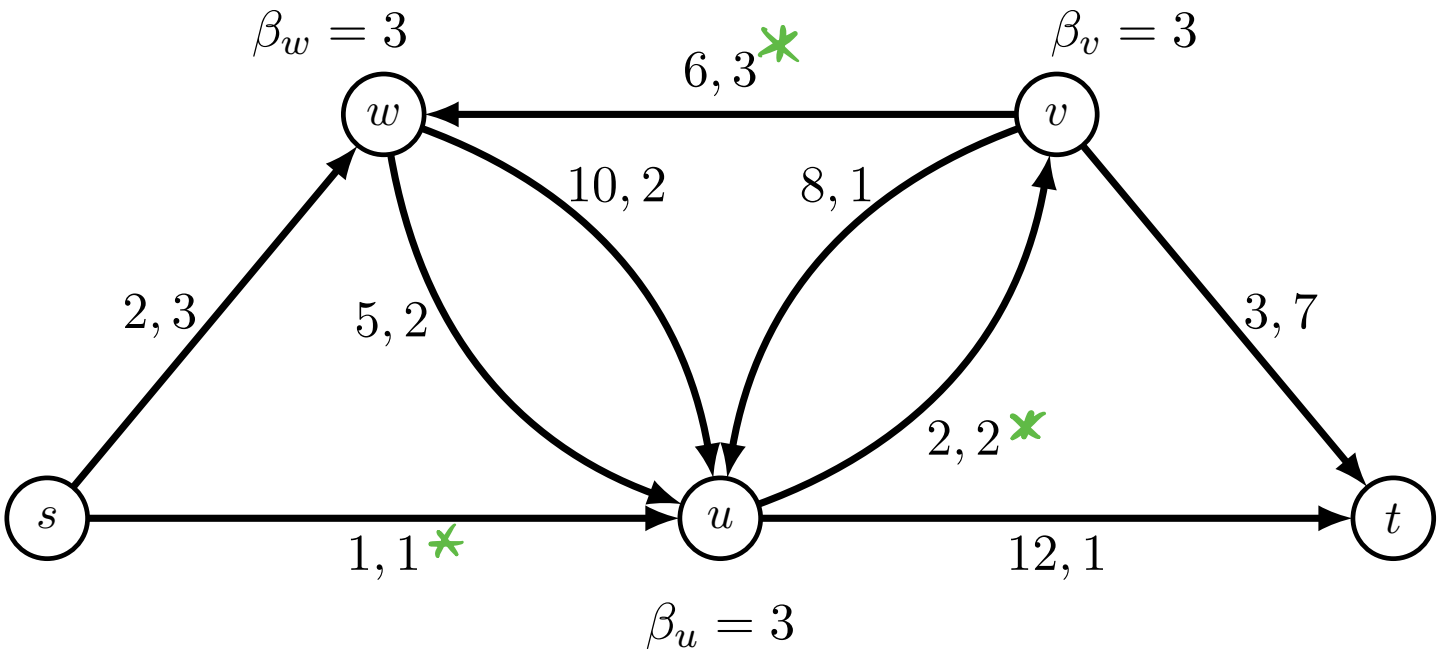
E_v^{dep} : $(v,t,3,7), (v,w,6,3), (v,u,8,1)$, $P_v = 0$

$$E^{arr} = \{(s,u,1,1), (u,v,2,2), (s,w,2,3), (w,u,5,2), (v,u,8,1), (v,w,6,3), (w,u,10,2), (v,t,3,7), (u,t,12,1)\}$$

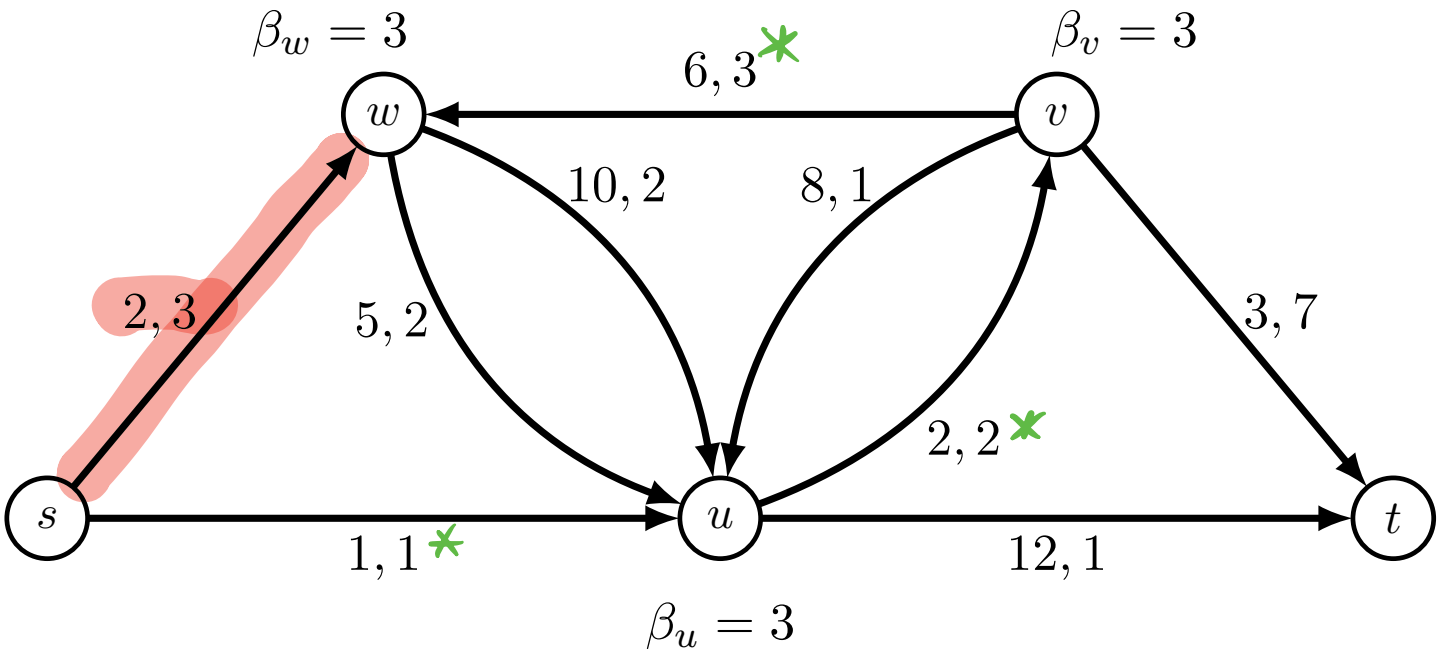


E_v^{dep} : $(v,t,3,7), (v,w,6,3), (v,u,8,1)$, $P_v = 2$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

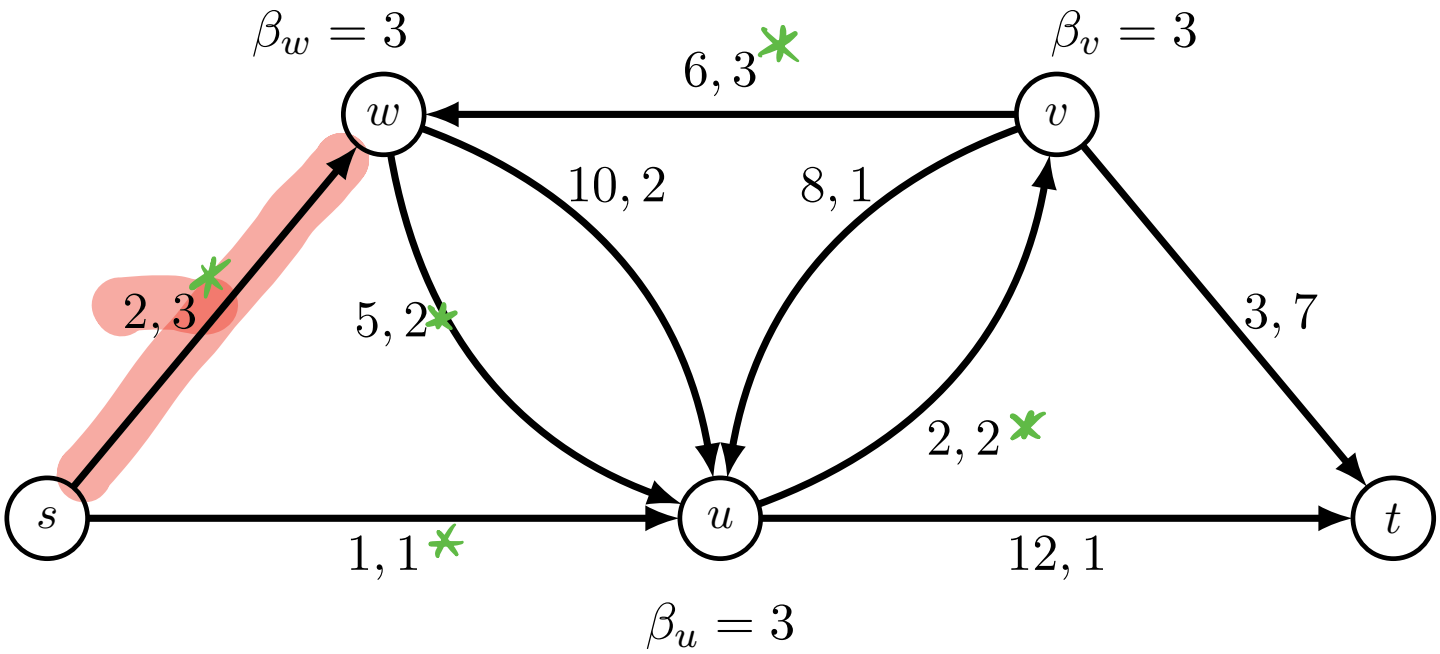


$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



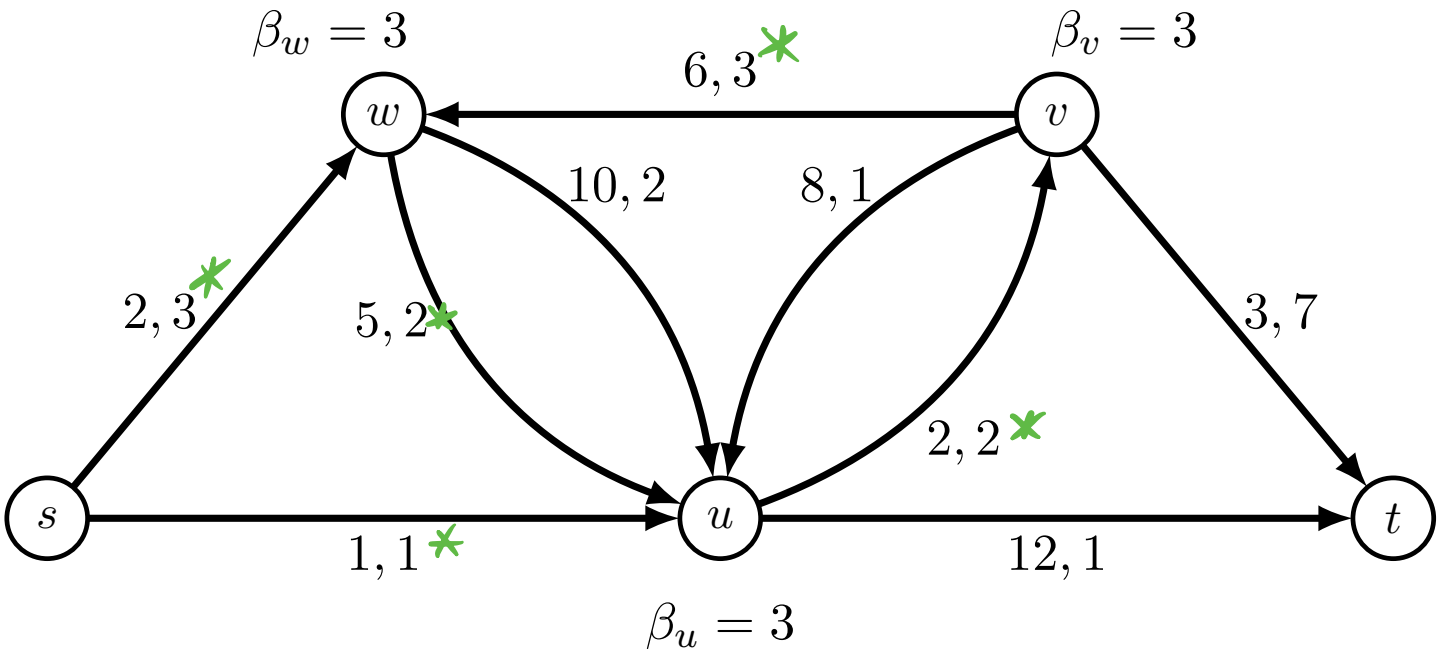
E^{dep}_w: $(w,u,5,2), (w,u,10,2)$, $P_w = 0$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

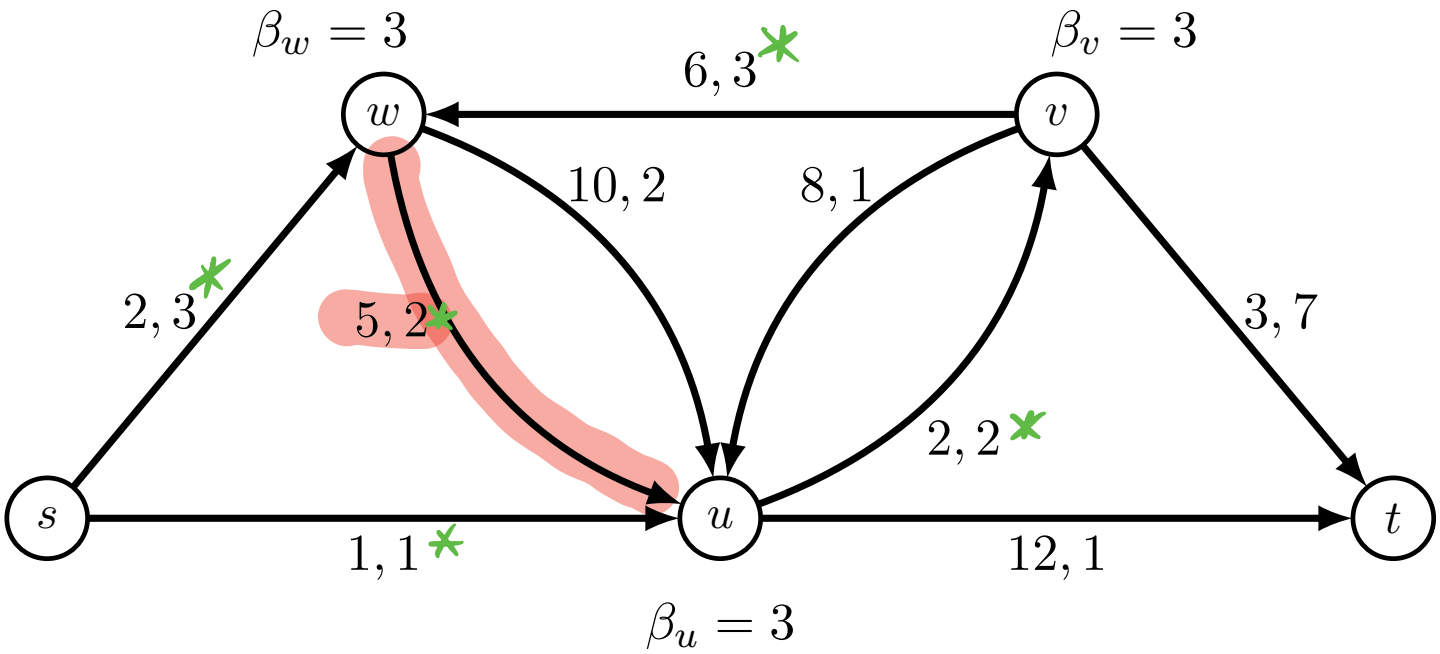


E^{dep}_w : $(w,u,5,2)$, $(w,u,10,2)$, $P_w = 1$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

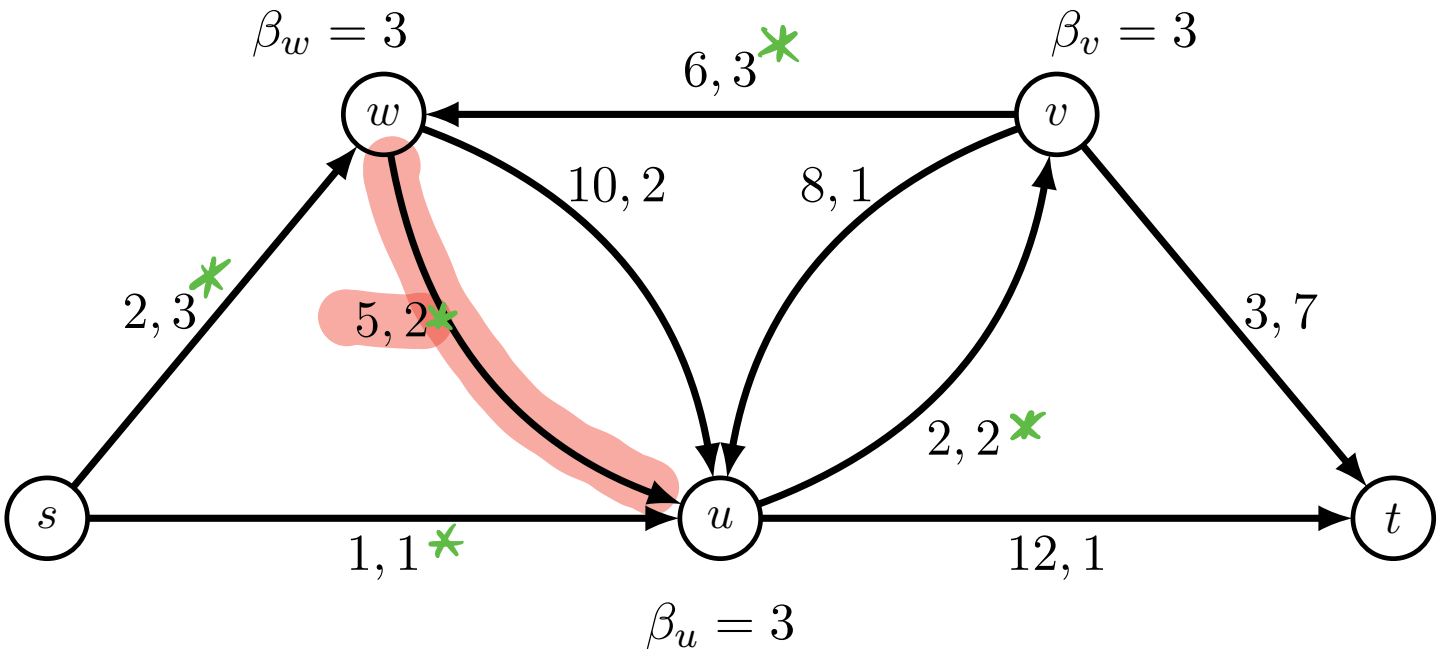


$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



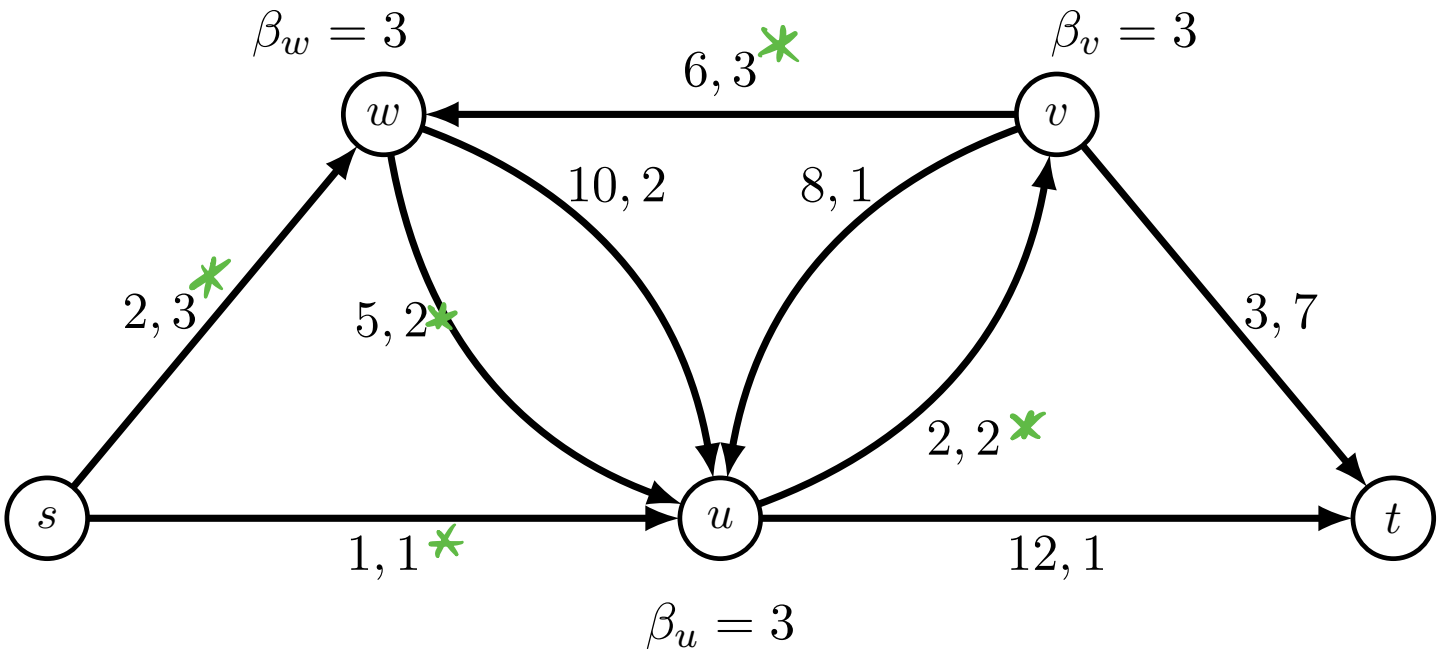
E_u^{dep} : $(u,v,2,2),(u,t,12,1)$, $P_u = 1$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

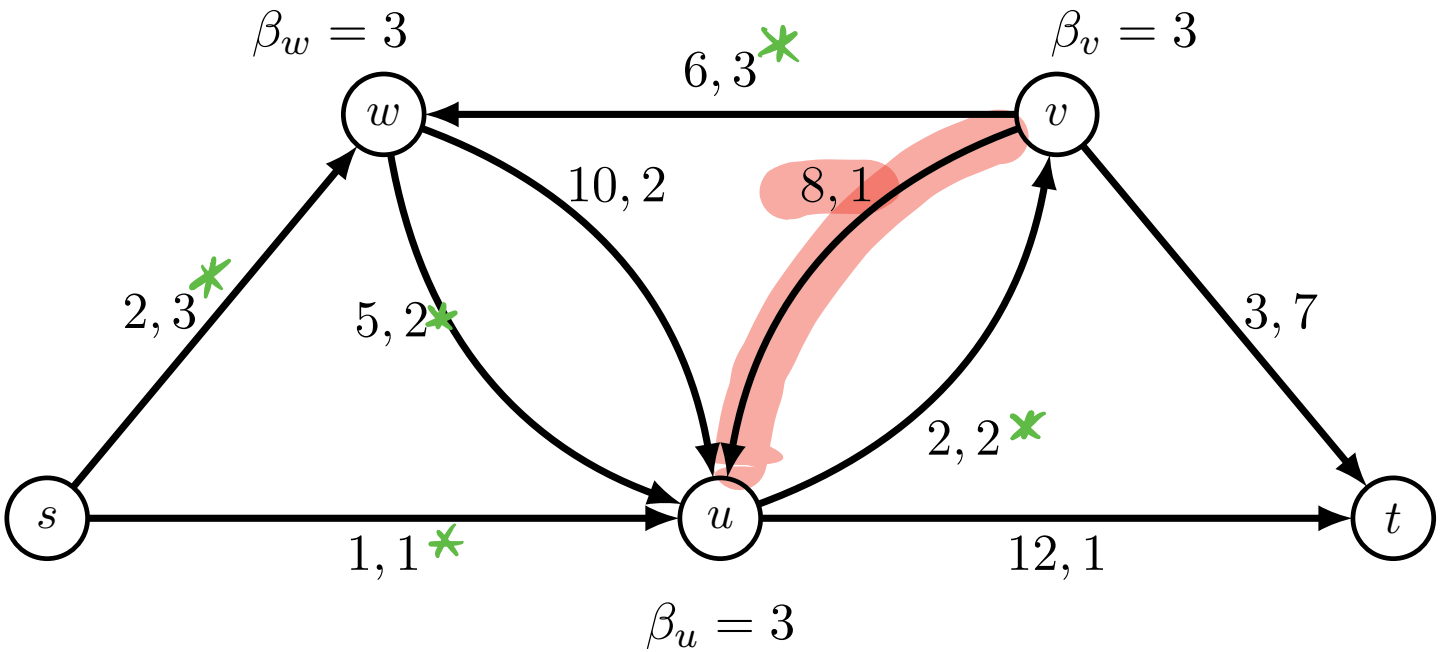


E_u^{dep} : $(u,v,2,2), (u,t,12,1)$, $P_u = 1$

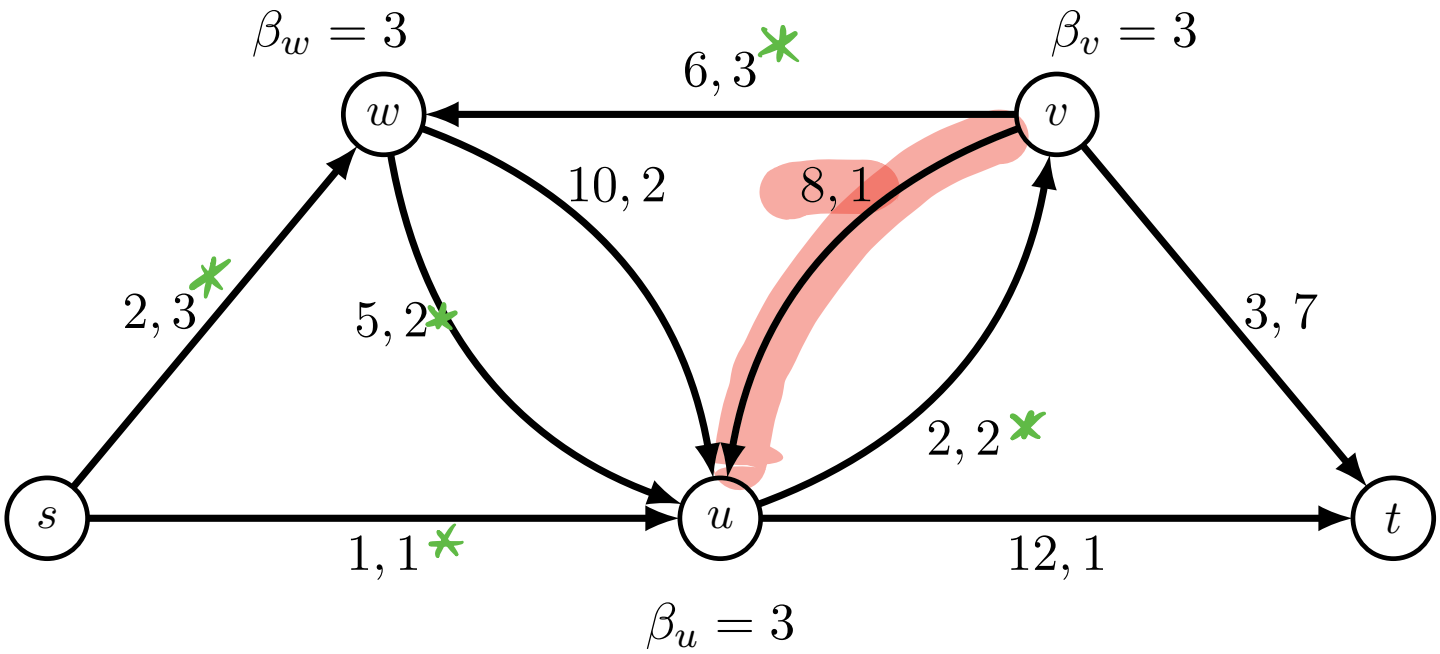
$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

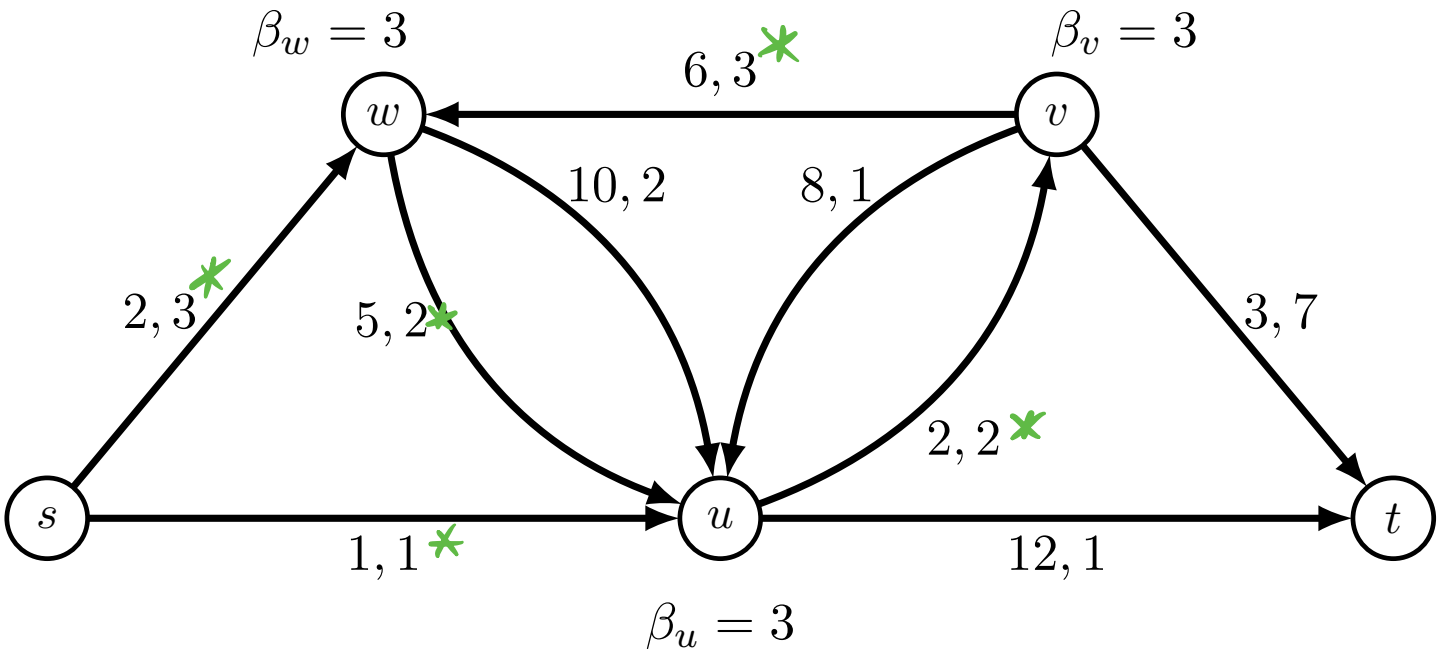


$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

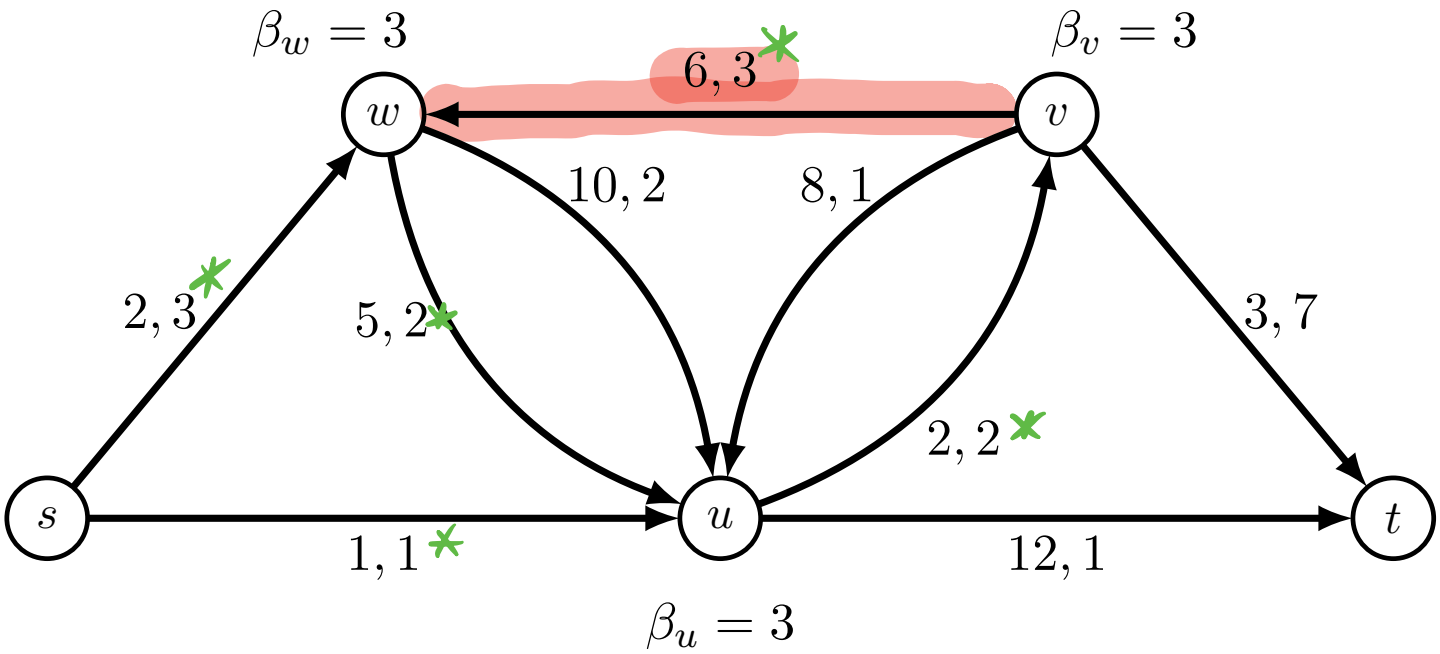


NOT REACHABLE

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

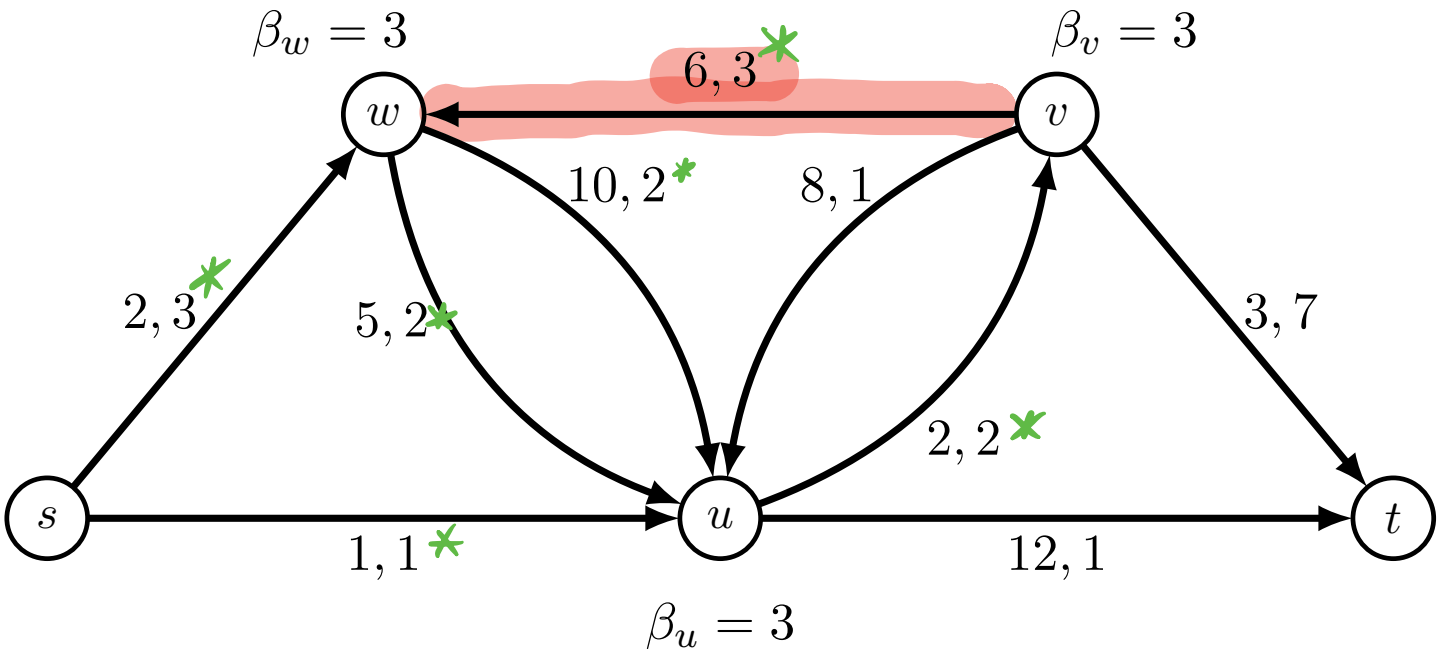


$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



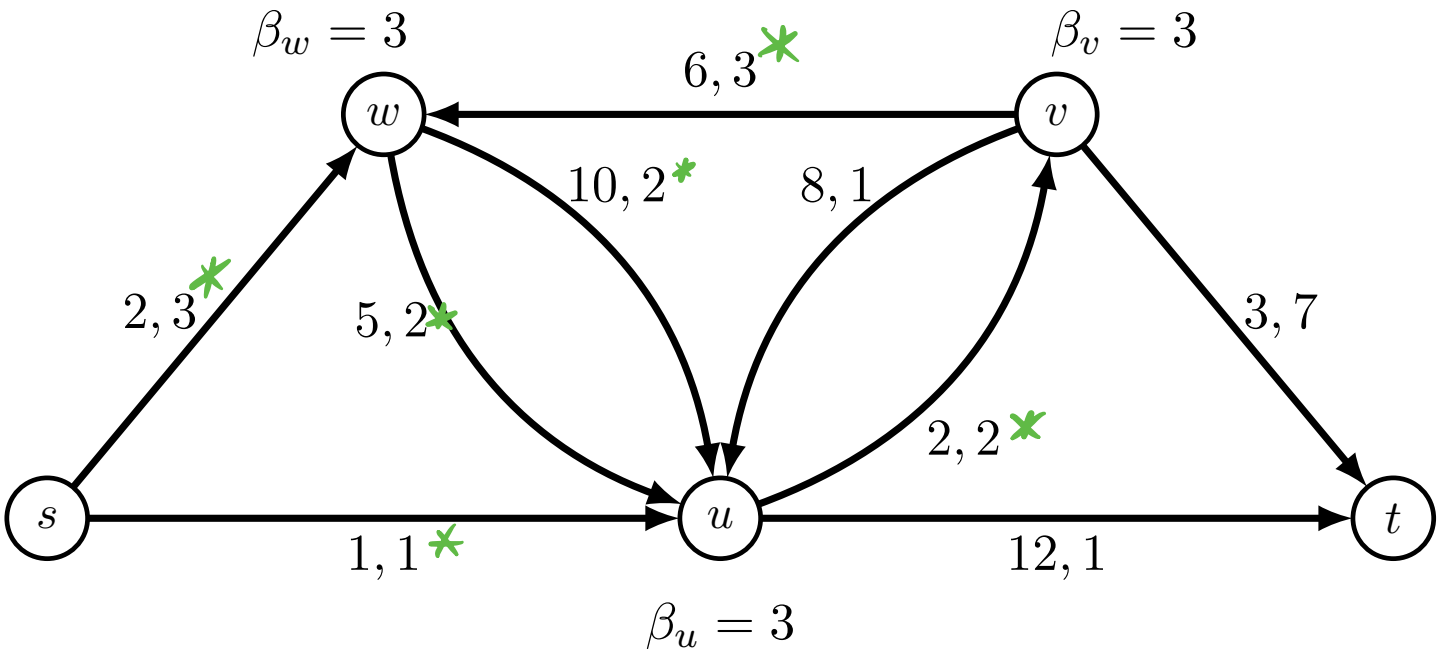
E_w^{dep} : $(w,u,5,2),(w,u,10,2)$, $P_w = 1$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

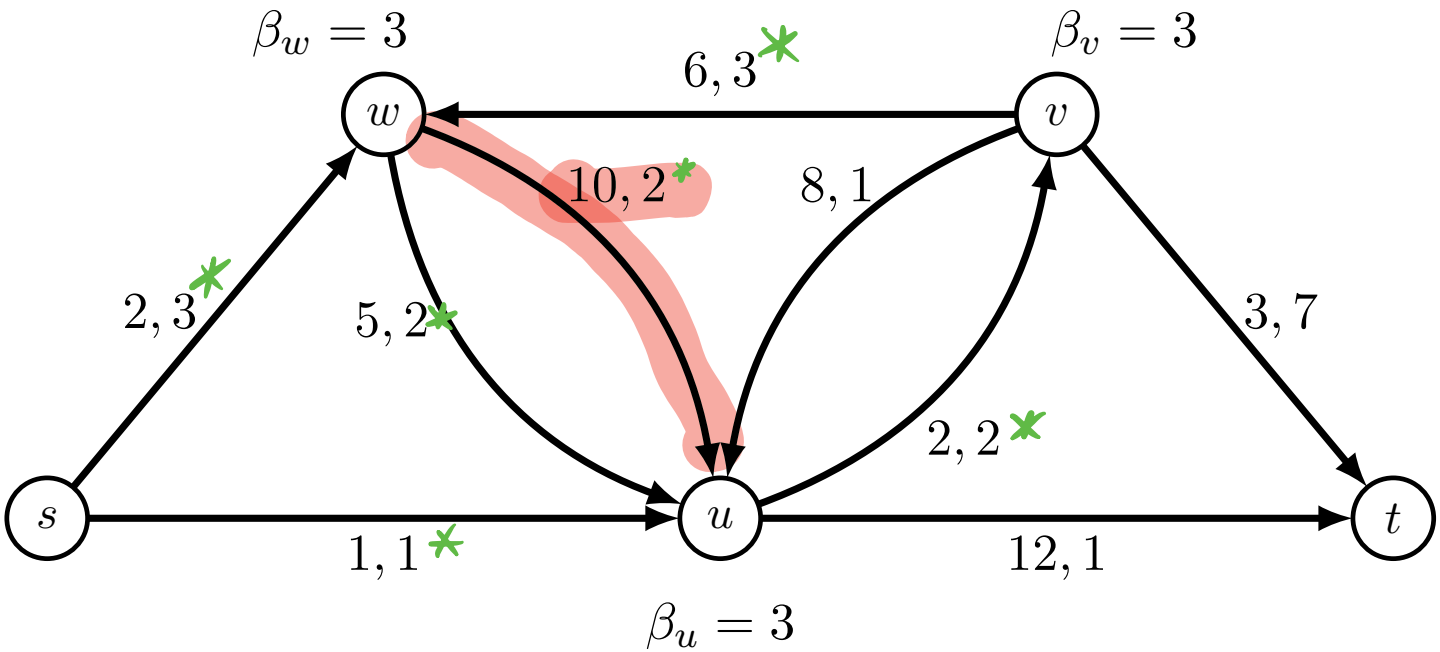


E_w^{dep} : $(w,u,5,2),(w,u,10,2)$, $P_w = 2$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

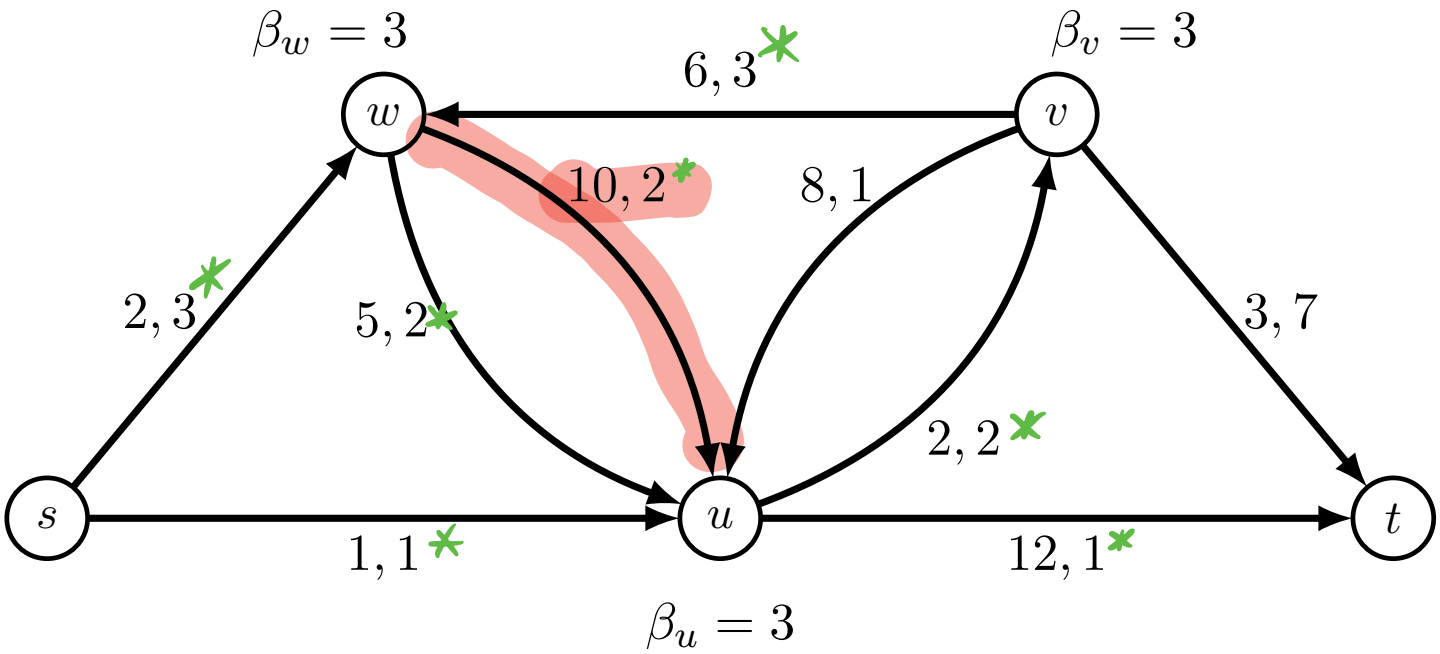


$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



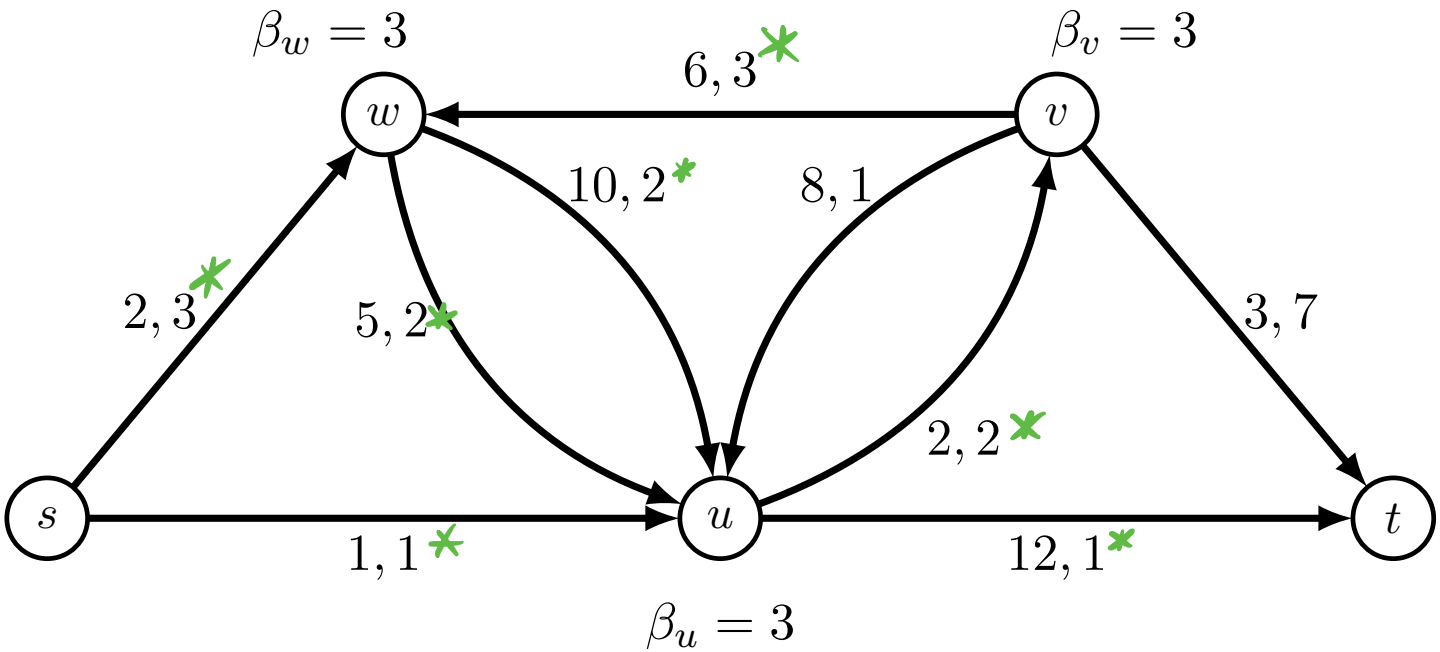
E_u^{dep} : $(u,v,2,2), (u,t,12,1)$, $P_u = 1$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

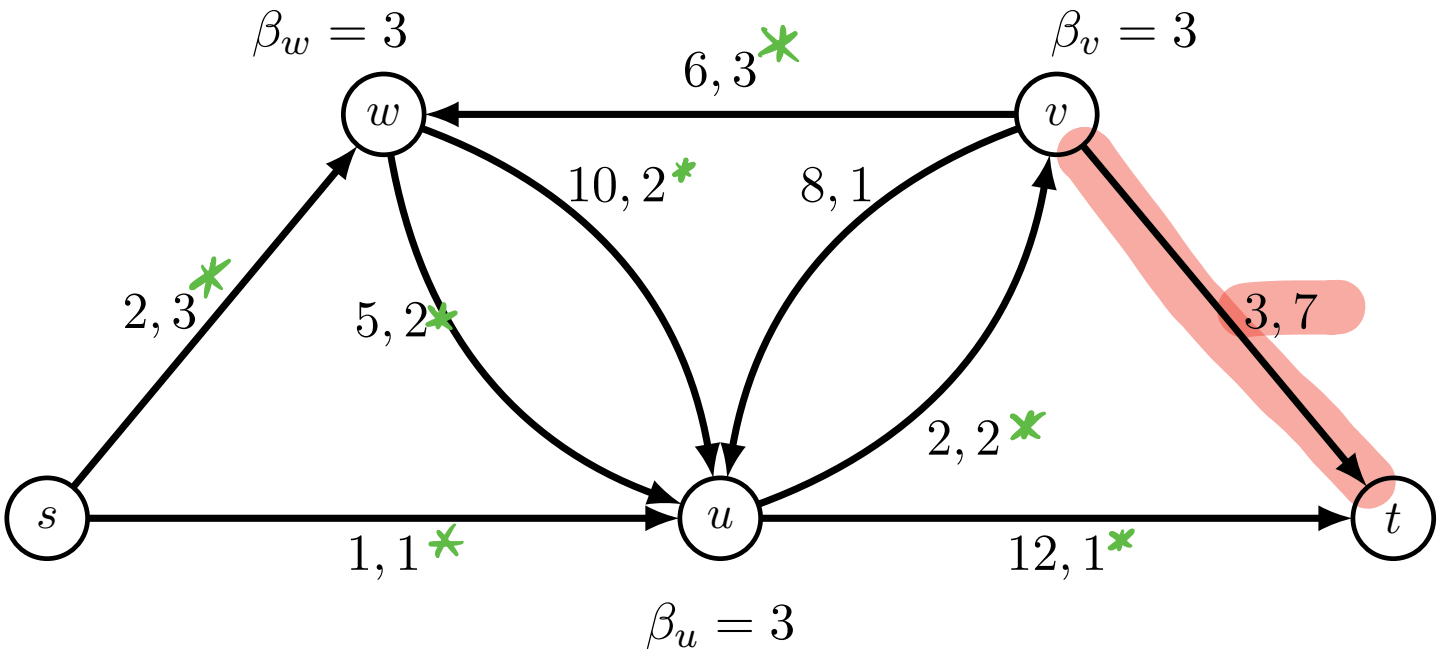


E_u^{dep} : $(u,v,2,2),(u,t,12,1)$, $P_u = 2$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

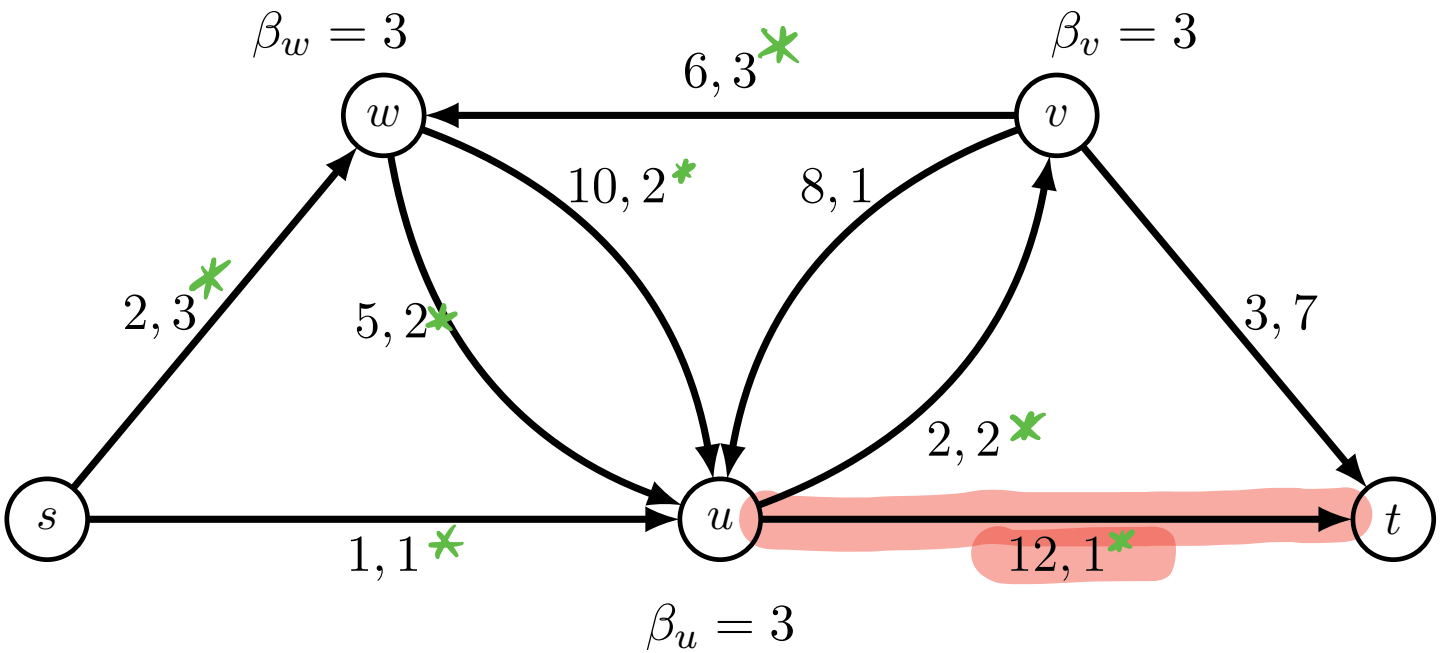


$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



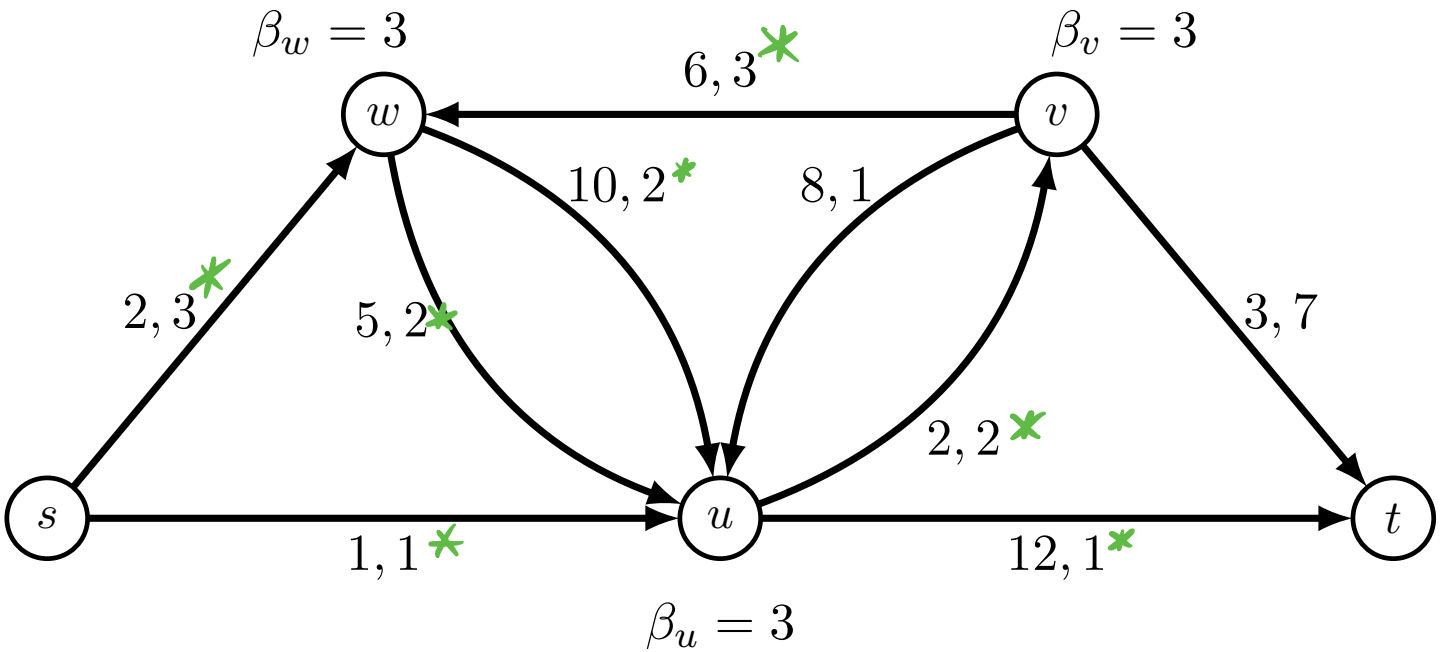
$E_t^{dep} = \emptyset$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

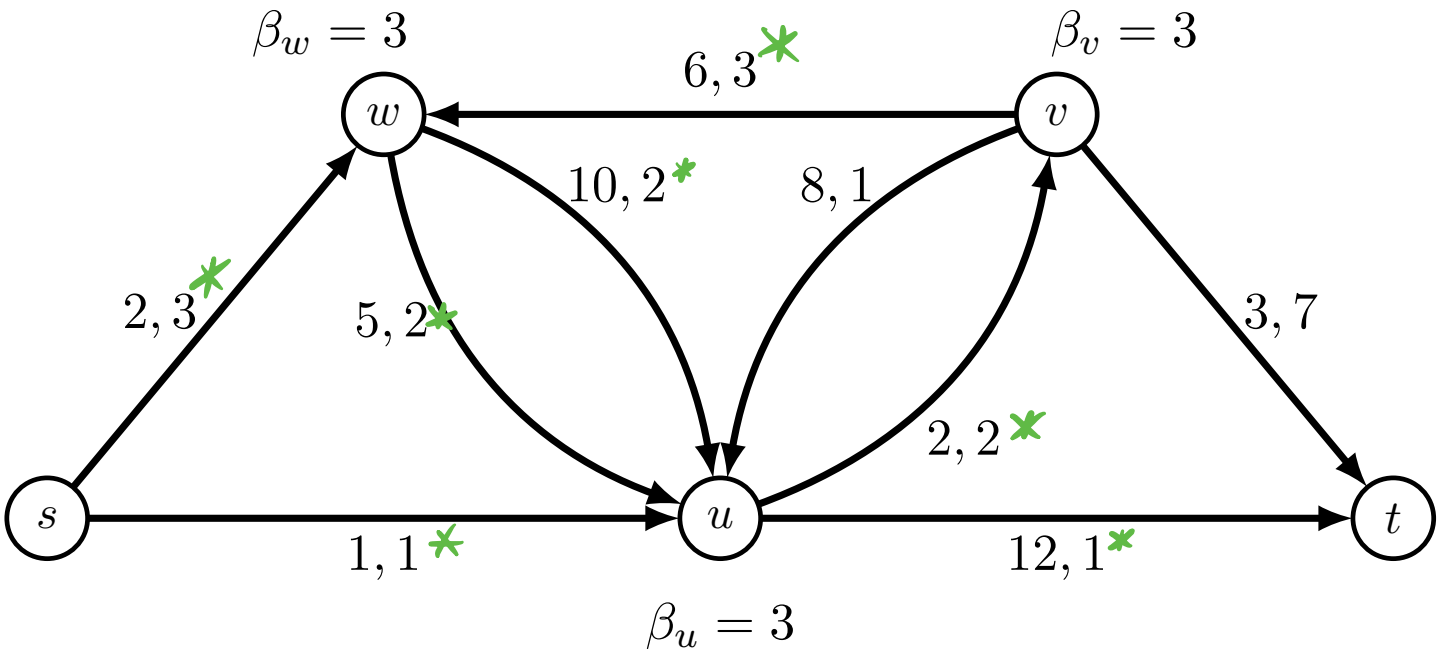


$$\mathcal{E}_t^{dep} = \emptyset$$

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

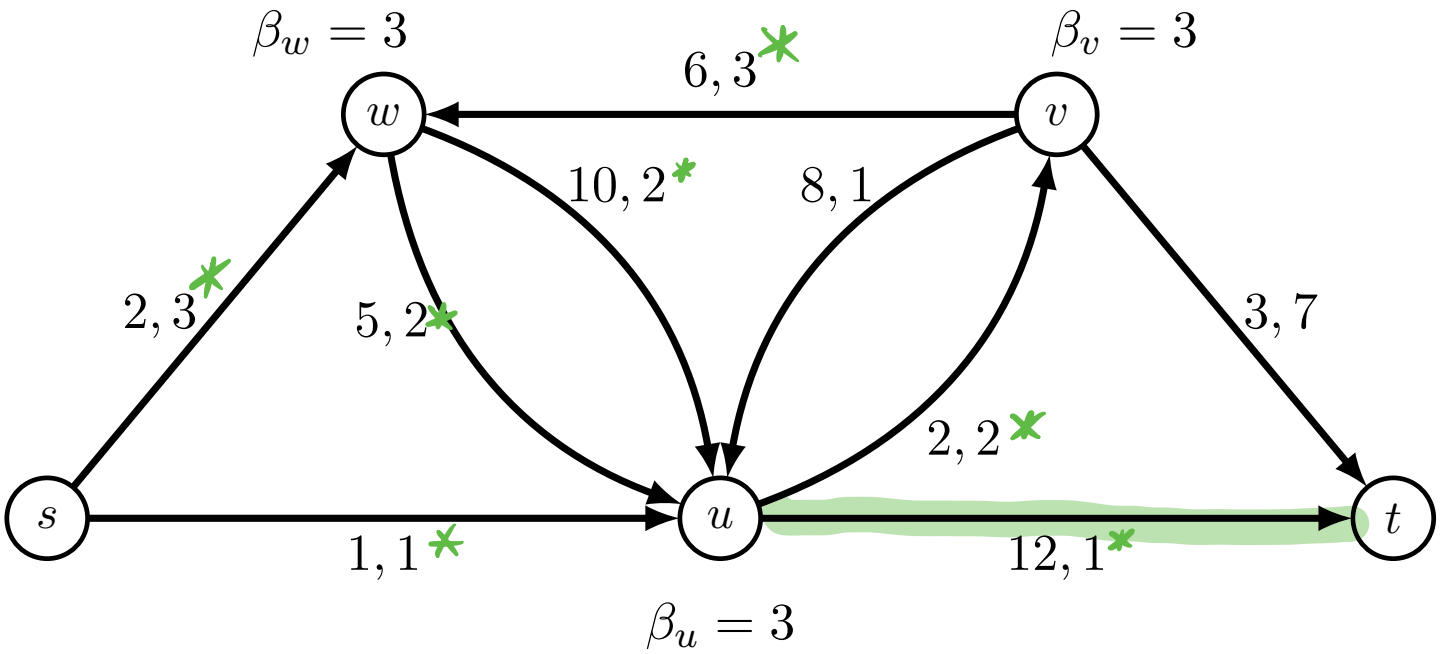


$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$



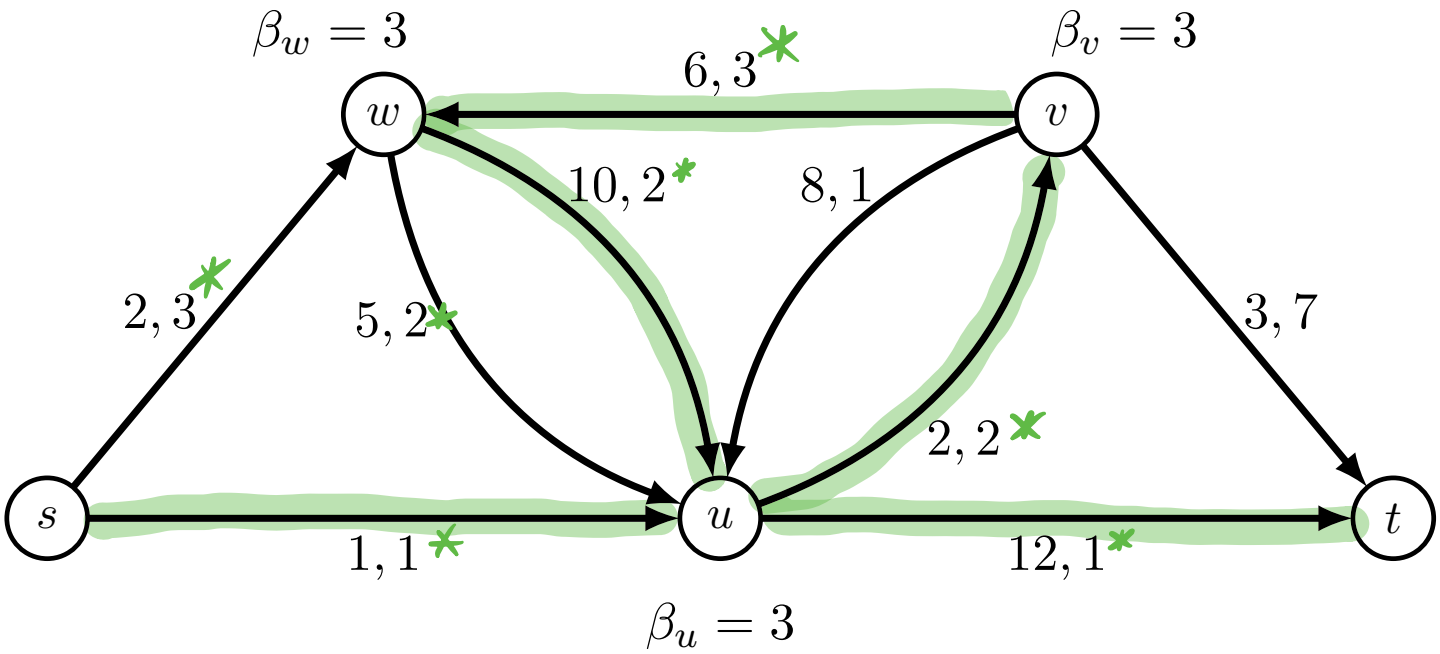
Extract the temporal walk following parent pointers

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

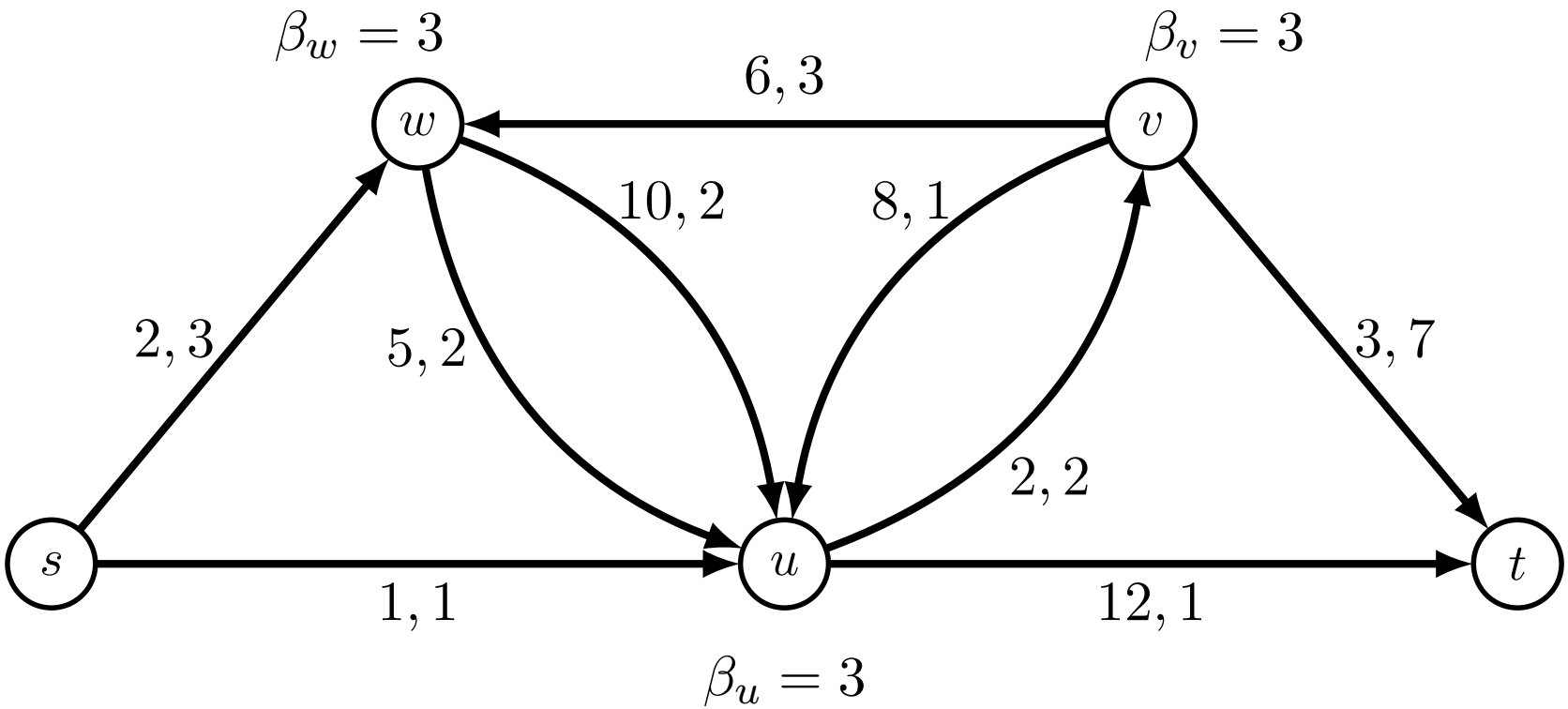


Extract the temporal walk following parent pointers

$$E^{arr} = \{(s,u,1,1),(u,v,2,2),(s,w,2,3),(w,u,5,2),(v,u,8,1),(v,w,6,3),$$
$$(w,u,10,2),(v,t,3,7),(u,t,12,1)\}$$

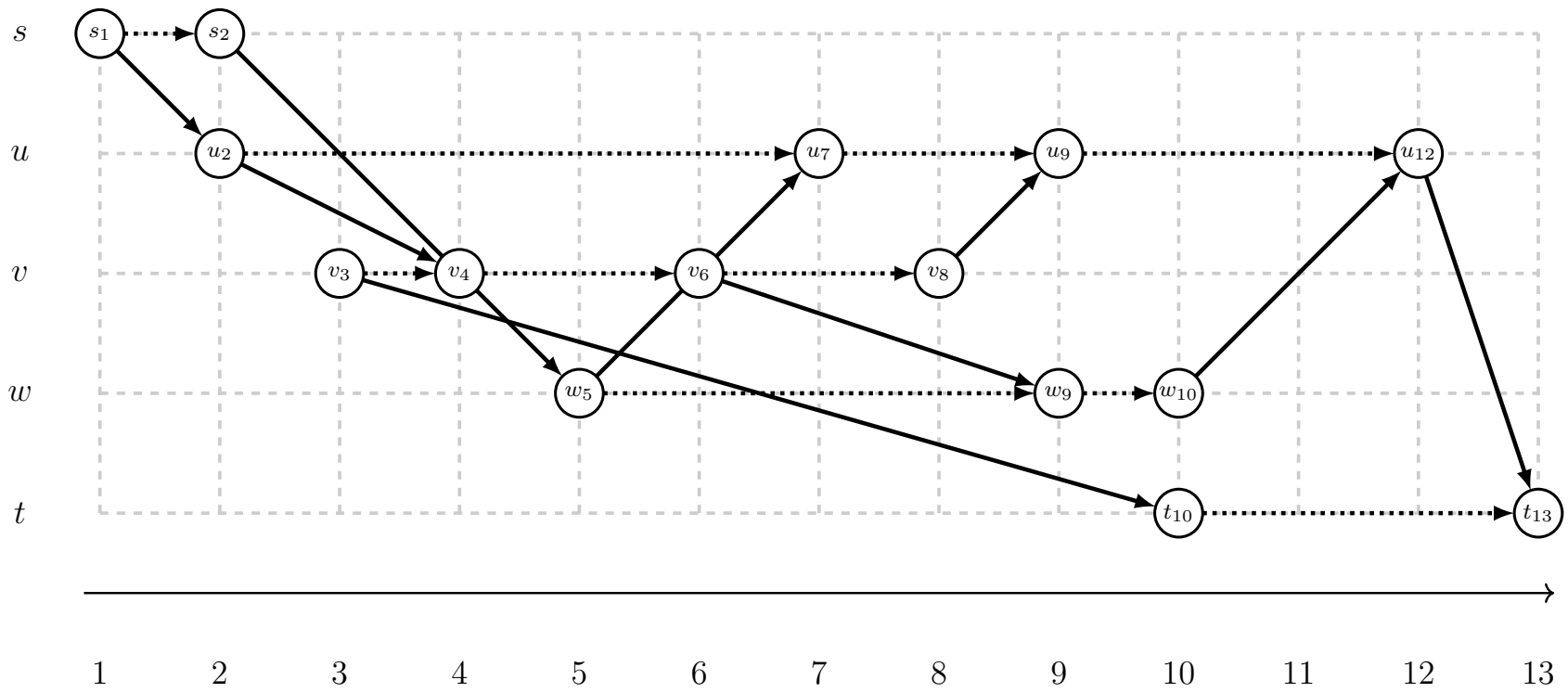


Extract the temporal walk following parent pointers



$$E^{arr} = \{(s, u, 1, 1), (u, v, 2, 2), (s, w, 2, 3), (w, u, 5, 2), (v, u, 8, 1), ...\}$$
$$E^{dep} = \{(s, u, 1, 1), (s, w, 2, 3)(u, v, 2, 2), (v, t, 3, 7), (w, u, 5, 2), ...\}$$

Time-expanded representation



Doubly-sorted representation

For simplicity we will speak about the case in which all edges have strictly positive travel time ($\lambda > 0$).

Definition

A *doubly-sorted representation* of a temporal graph (V, E, β) consits in two lists E^{arr} and E^{dep} where E^{arr} contains all temporal edges in E sorted by non-decreasing arrival time, and E^{dep} by non-decreasing departure time.

Doubly-sorted representation

For simplicity we will speak about the case in which all edges have strictly positive travel time ($\lambda > 0$).

Definition

A *doubly-sorted representation* of a temporal graph (V, E, β) consits in two lists E^{arr} and E^{dep} where E^{arr} contains all temporal edges in E sorted by non-decreasing arrival time, and E^{dep} by non-decreasing departure time.

Weaker assumptions:

- E^{arr} is node-arrival sorted and walk-respecting
- E^{dep} is node-departure sorted

They can be computed in linear time from a space-time representation.

Perspectives

The simple core idea of the algorithm we presented can be exploited to design more involved and efficient algorithms.

Perspectives

The simple core idea of the algorithm we presented can be exploited to design more involved and efficient algorithms.
We used it to compute in temporal graphs subject to waiting constraints, in linear time and space:

- shortest duration temporal walks,
- fewest hops temporal walks,
- minimum waiting-time temporal walks,
- many other criteria.

Thank you for your attention