1. **Problemet**

2. **Ideer**

3. **Ford-Fulkerson**

4. **Minimalt snitt**

5. **Matching**

Forelesning 12

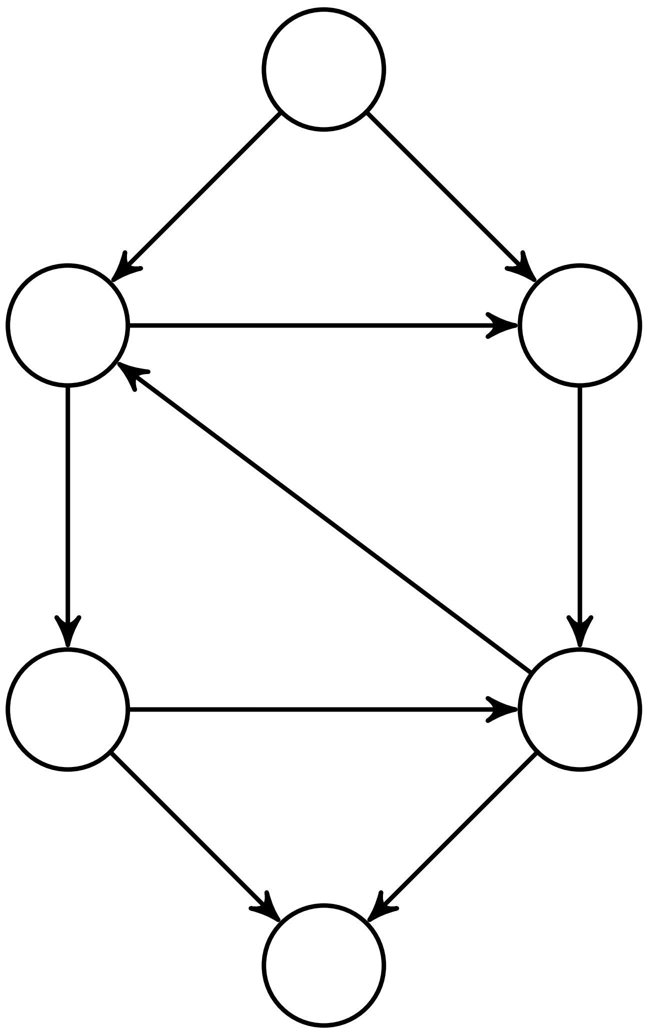
Maksimal flyt

1:5

Problemet

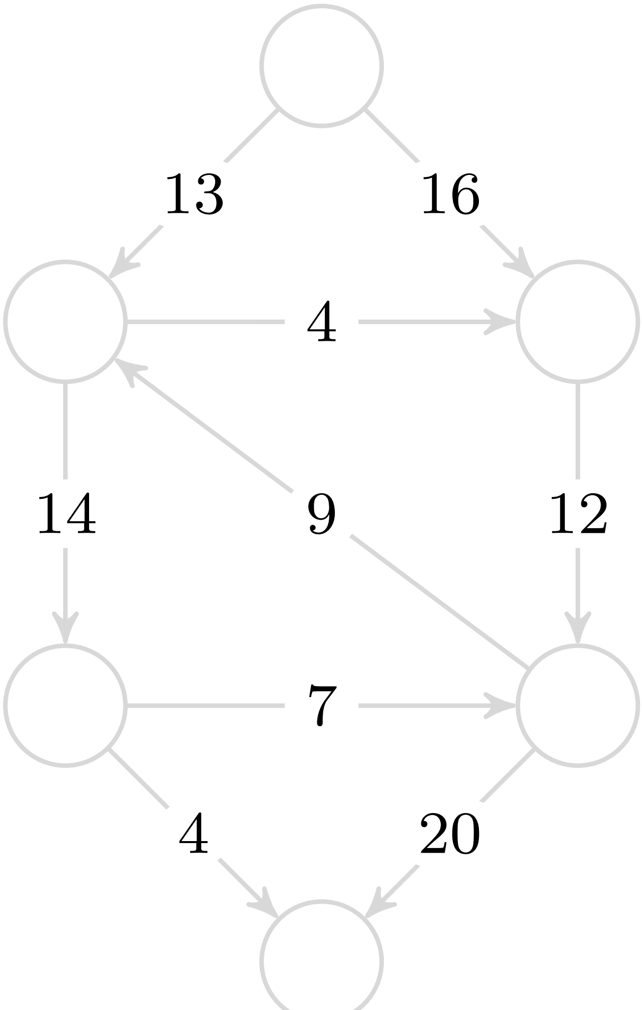
Flytnett: Rettet graf $G = (V, E)$

Ofte kalt flytnettverk.



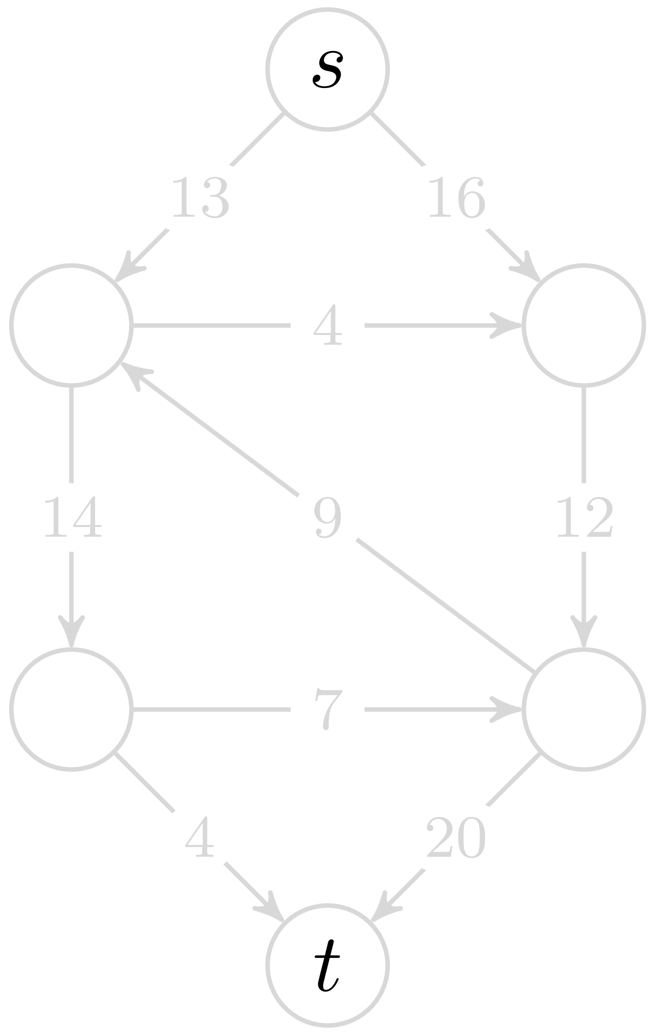
Flytnett: Rettet graf $G = (V, E)$

- Kapasiteter $c(u, v) \geqslant 0$



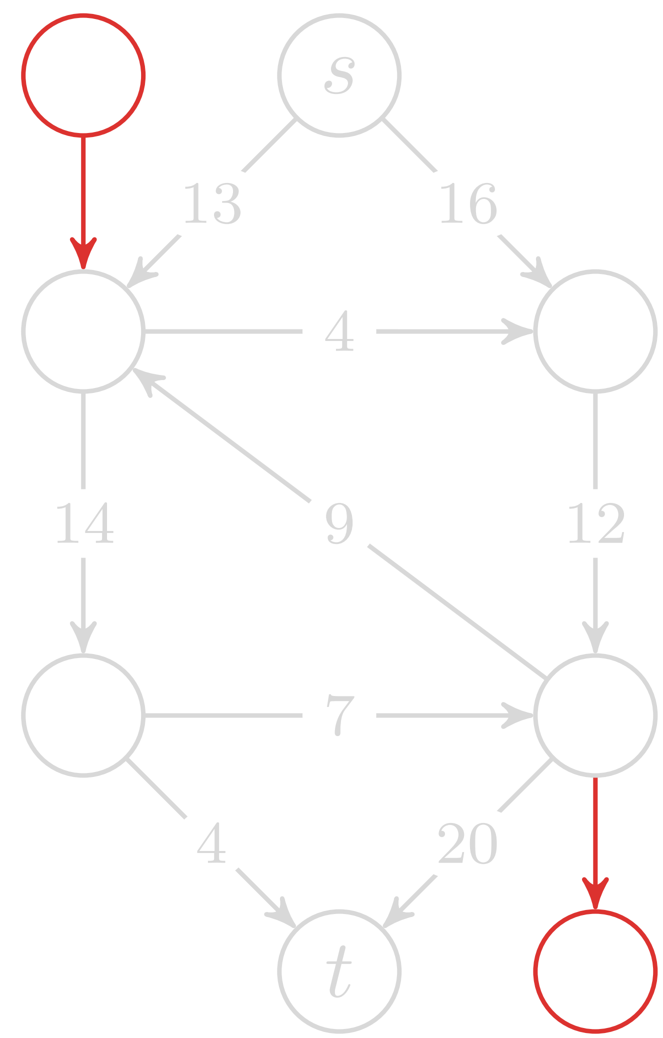
Flytnett: Rettet graf $G = (V, E)$

- Kapasiteter $c(u, v) \geqslant 0$
- Kilde og sluk $s, t \in V$



Flytnett: Rettet graf $G = (V, E)$

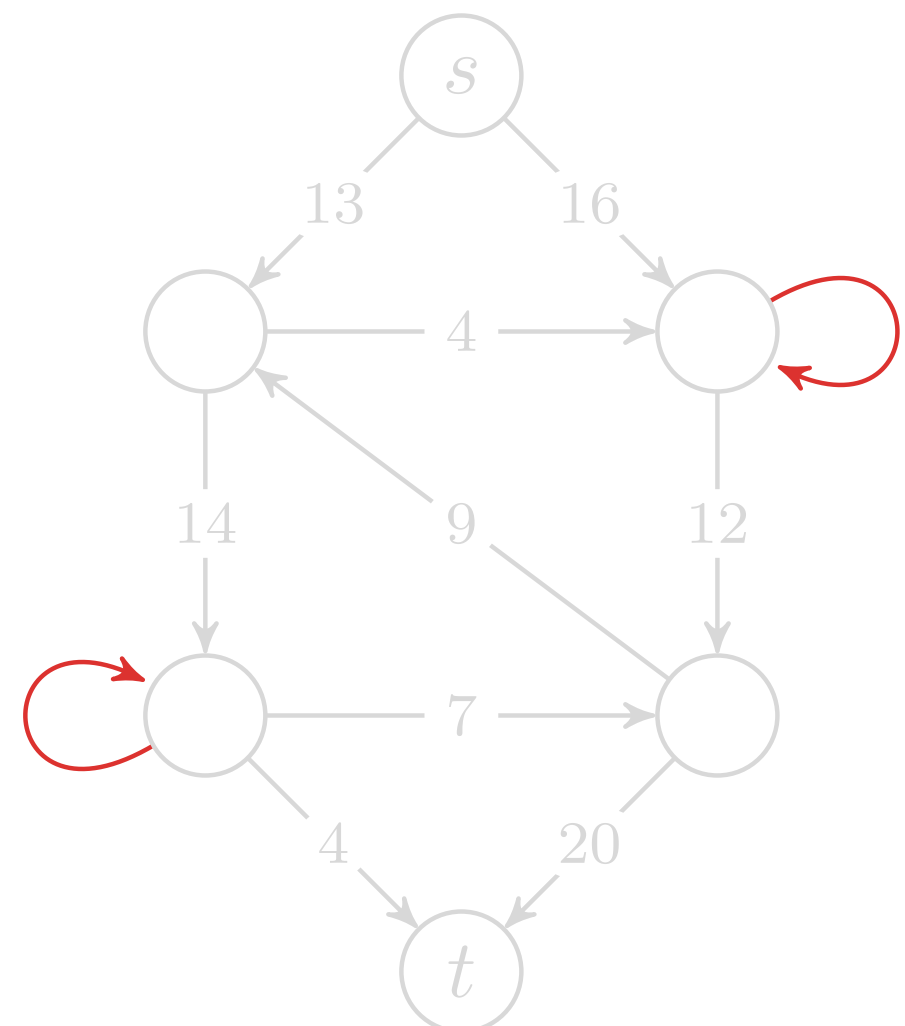
- Kapasiteter $c(u, v) \geqslant 0$
- Kilde og sluk $s, t \in V$
- $v \in V \implies s \leadsto v \leadsto t$



Forenklende (harmløs) antagelse: Alle noder er på en sti fra s til t

Flytnett: Rettet graf $G = (V, E)$

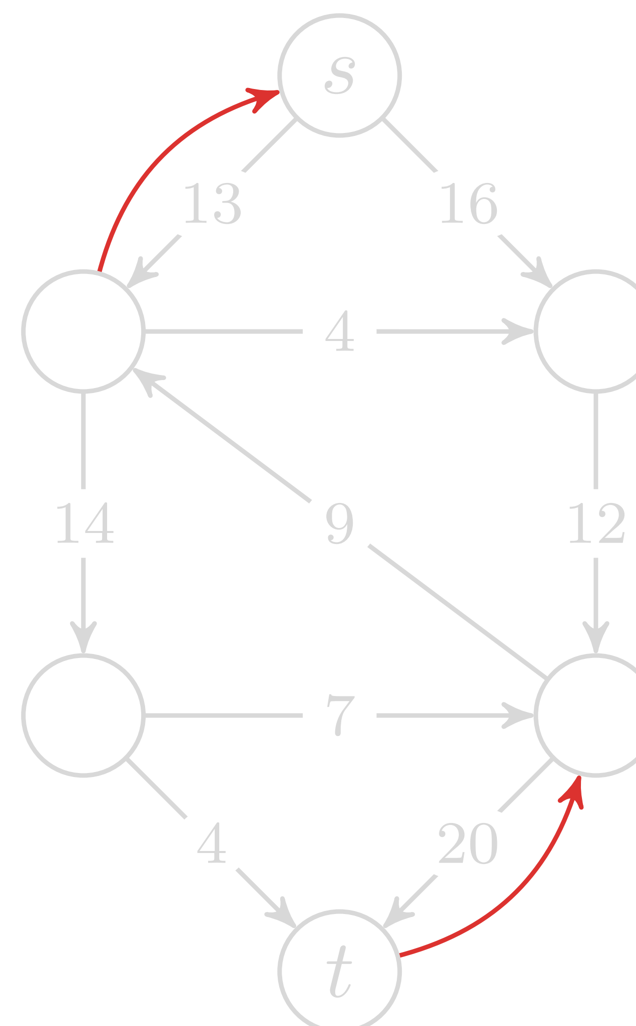
- Kapasiteter $c(u, v) \geqslant 0$
- Kilde og sluk $s, t \in V$
- $v \in V \implies s \rightsquigarrow v \rightsquigarrow t$
- Ingen løkker (*self-loops*)



Merk: Vi kan ha sykler! (Se $1 \to 3 \to 2 \to 1$, f.eks.)

Flytnett: Rettet graf $G = (V, E)$

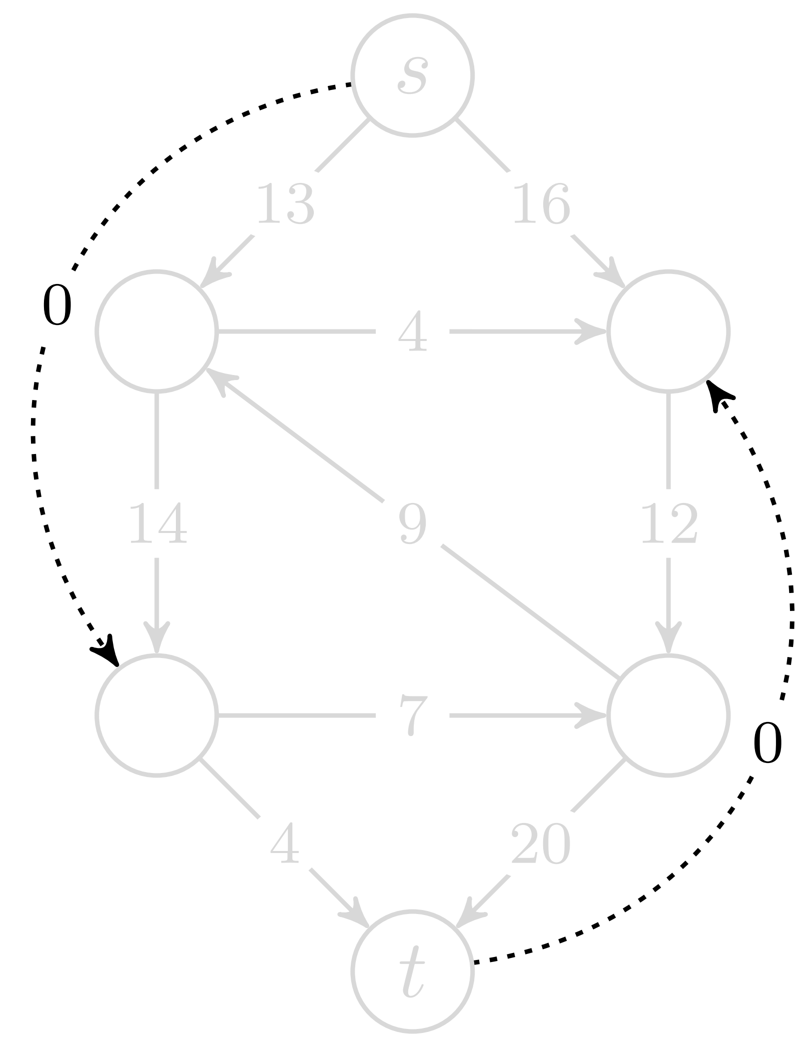
- Kapasiteter $c(u, v) \geqslant 0$
- Kilde og sluk $s, t \in V$
- $v \in V \implies s \leadsto v \leadsto t$
- Ingen løkker (*self-loops*)
- $(u, v) \in E \implies (v, u) \notin E$



Nok en forenkling: Tillater ikke antiparallelle kanter

Flytnett: Rettet graf $G = (V, E)$

- Kapasiteter $c(u, v) \geqslant 0$
- Kilde og sluk $s, t \in V$
- $v \in V \implies s \leadsto v \leadsto t$
- Ingen løkker (*self-loops*)
- $(u, v) \in E \implies (v, u) \notin E$
- $(u, v) \notin E \implies c(u, v) = 0$

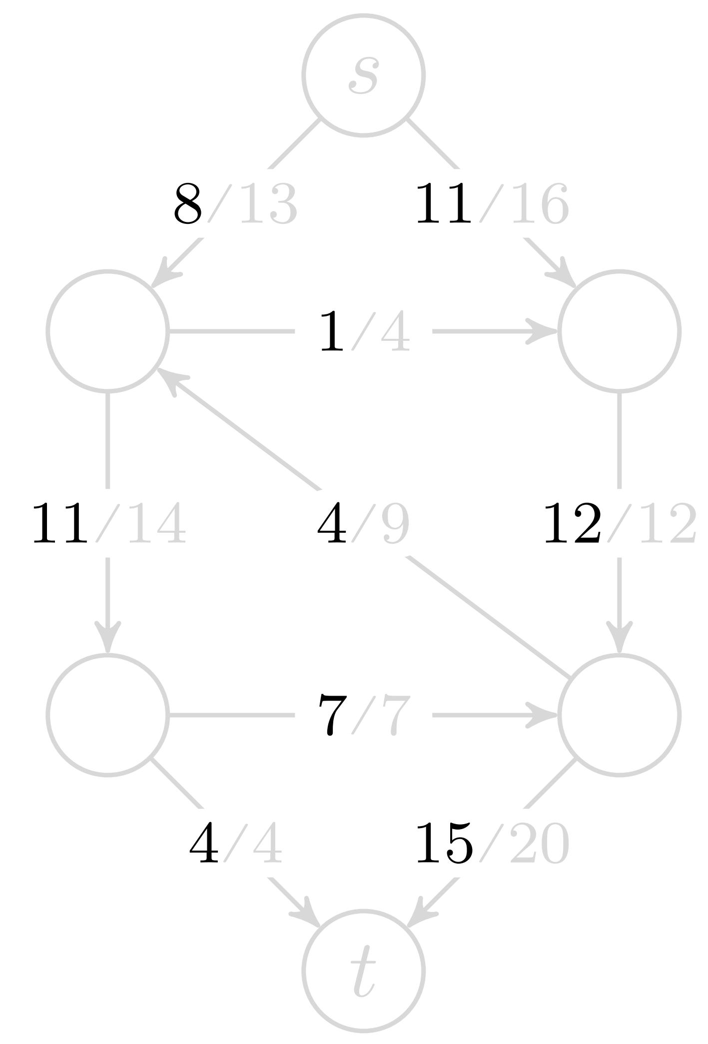


Ingen kapasitet uten kant

Flytnett: Rettet graf $G = (V, E)$

- Kapasiteter $c(u, v) \geqslant 0$
- Kilde og sluk $s, t \in V$
- $v \in V \implies s \leadsto v \leadsto t$
- Ingen løkker (*self-loops*)
- $(u, v) \in E \implies (v, u) \notin E$
- $(u, v) \notin E \implies c(u, v) = 0$

Flyt: En funksjon $f : V \times V \to \mathbb{R}$



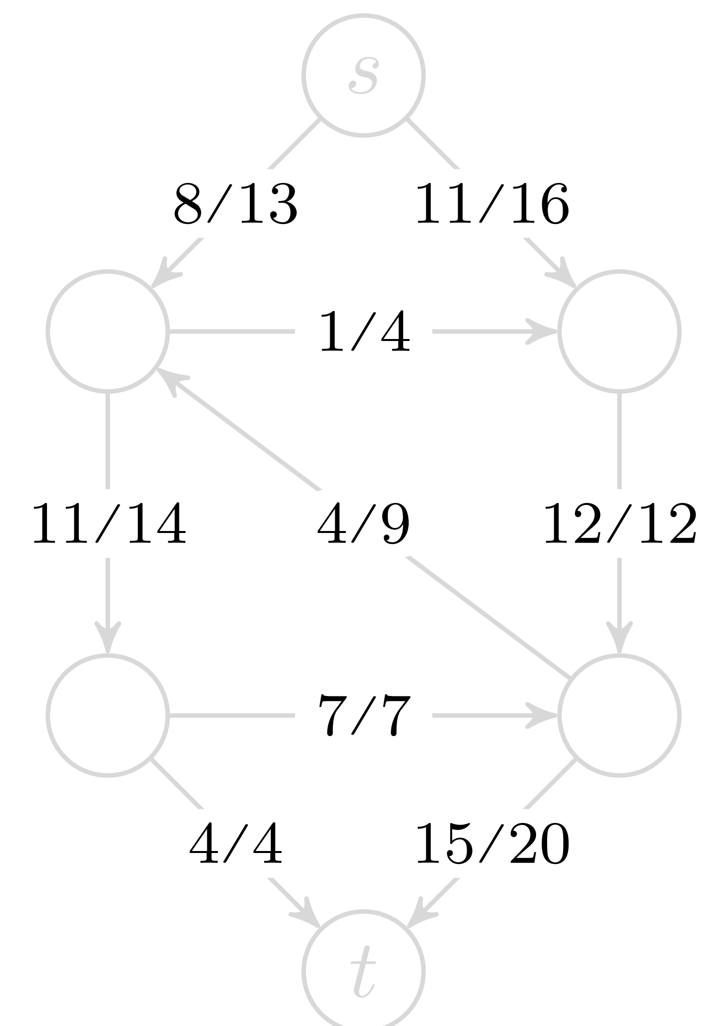
Vi har flyt fra enhver node til enhver annen

Flytnett: Rettet graf $G = (V, E)$

- Kapasiteter $c(u, v) \geqslant 0$
- Kilde og sluk $s, t \in V$
- $v \in V \implies s \leadsto v \leadsto t$
- Ingen løkker (*self-loops*)
- $(u, v) \in E \implies (v, u) \notin E$
- $(u, v) \notin E \implies c(u, v) = 0$

Flyt: En funksjon $f : V \times V \to \mathbb{R}$

- $0 \leqslant f(u, v) \leqslant c(u, v)$

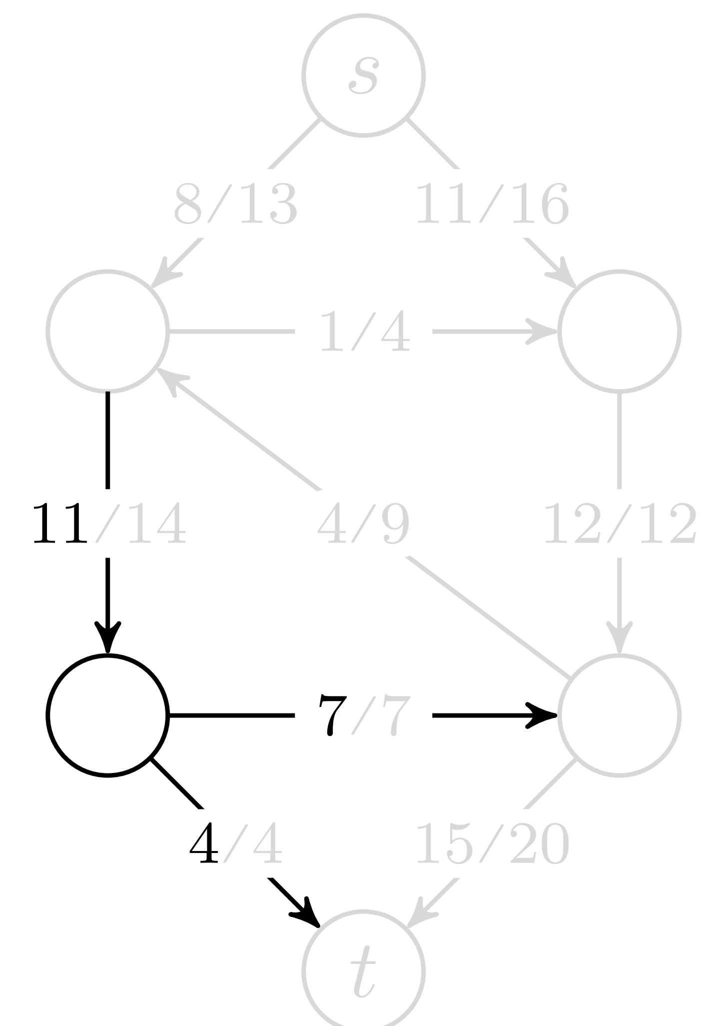


Flytnett: Rettet graf $G = (V, E)$

- Kapasiteter $c(u,v) \geqslant 0$
- Kilde og sluk $s, t \in V$
- $v \in V \implies s \leadsto v \leadsto t$
- Ingen løkker (*self-loops*)
- $(u,v) \in E \implies (v,u) \notin E$
- $(u,v) \notin E \implies c(u,v) = 0$

Flyt: En funksjon $f : V \times V \to \mathbb{R}$

- $0 \leqslant f(u,v) \leqslant c(u,v)$
- $u \neq s,t \implies \sum_v f(v,u) = \sum_v f(u,v)$



Flyt inn = flyt ut (tilsv. Kirchhoffs første lov)

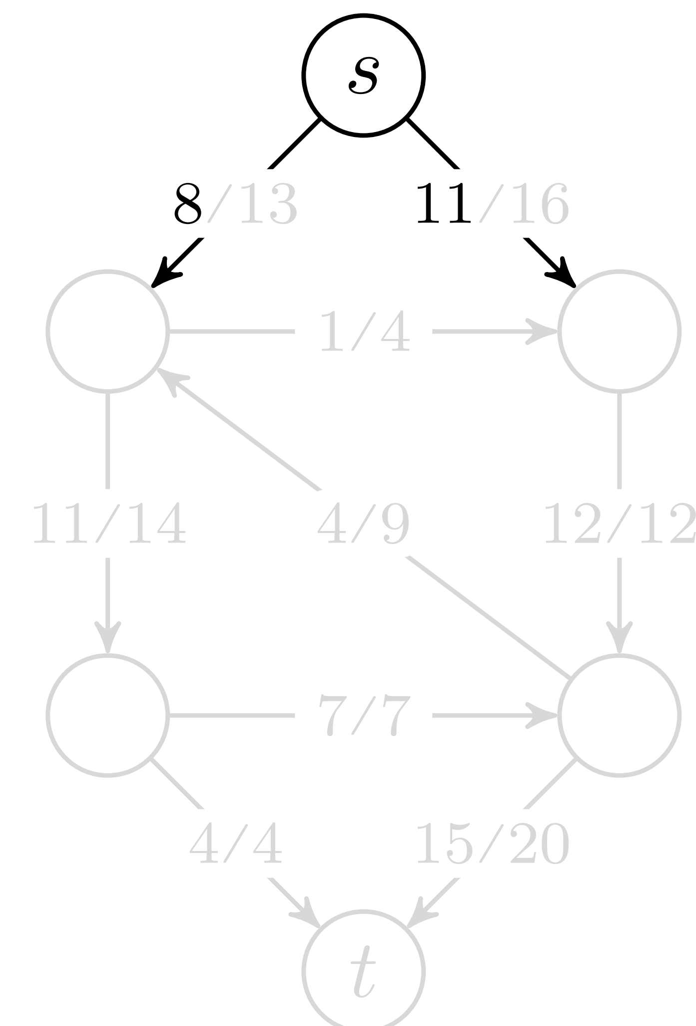
Flytnett: Rettet graf $G = (V, E)$

- Kapasiteter $c(u, v) \geqslant 0$
- Kilde og sluk $s, t \in V$
- $v \in V \implies s \leadsto v \leadsto t$
- Ingen løkker (*self-loops*)
- $(u, v) \in E \implies (v, u) \notin E$
- $(u, v) \notin E \implies c(u, v) = 0$

Flyt: En funksjon $f : V \times V \to \mathbb{R}$

- $0 \leqslant f(u, v) \leqslant c(u, v)$
- $u \neq s, t \implies \sum_v f(v, u) = \sum_v f(u, v)$

Flytverdi: $|f| = \sum_v f(s, v) - \sum_v f(v, s)$



Input: Et flytnett G.

Output: En flyt f for G med maks. $|f|$.

- **Cormen 1 & 2 har andre definisjoner**
- **Antiparallelle kanter:**
  Splitt den ene med en node
- **Flere kilder og sluk:**
  Legg til super-kilde og super-sluk

2:5

Ideer

Restnett

Ofte kalt residualnettverk.

- **Engelsk: Residual network**
- **Fremoverkant ved ledig kapasitet**
- **Bakoverkant ved flyt**

Forøkende sti

- **Engelsk: Augmenting path**
- **En sti fra kilde til sluk i restnettet**
- **Langs fremoverkanter: Flyten kan økes**
- **Langs bakoverkanter: Flyten kan omdirigeres**
- **Altså: En sti der den totale flyten kan økes**

Flytoppheving

- **Vi kan «sende» flyt baklengs langs kanter der det allerede går flyt**
- **Vi opphever da flyten, så den kan omdirigeres til et annet sted**
- **Det er dette bakoverkantene i restnettet representerer**

$$c_f(u, v) =$$

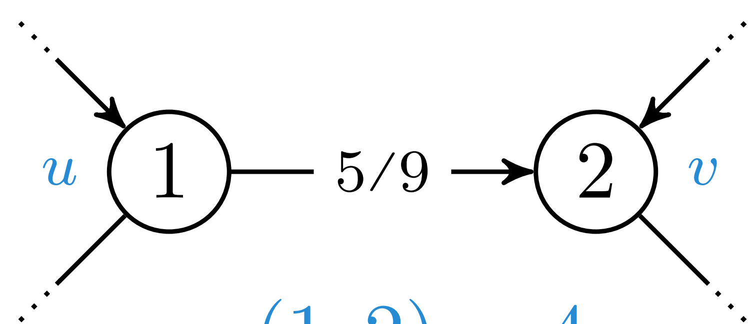
Restkapasitet: Hvor mye kan vi øke flyten fra u til v?

$$c_f(u,v) = \begin{cases} & \text{if } (u,v) \in E\,, \\ & \text{if } (v,u) \in E\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

$c_f(1,2) = 0$

Ingen kant: Har ingen flyt og kan ikke få det

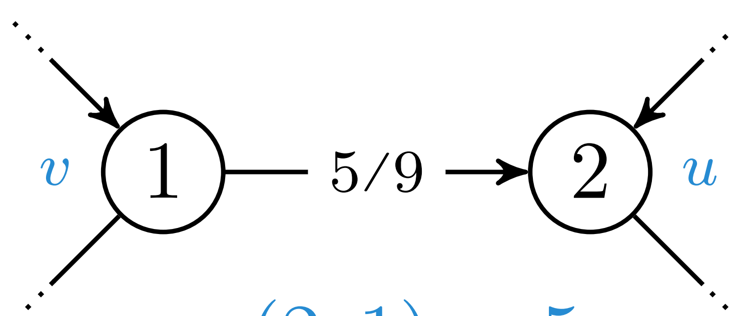
$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in \mathrm{E}\,, \\ & \text{if } (v,u) \in \mathrm{E}\,, \\ 0 & \text{otherwise}\,. \end{cases}$$



$c_f(1,2) = 4$

Fremover: Kan øke med det som gjenstår av $c(u,v)$

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in \mathrm{E}\,, \\ f(v,u) & \text{if } (v,u) \in \mathrm{E}\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

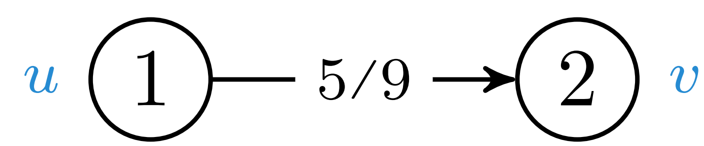


$c_f(2,1) = 5$

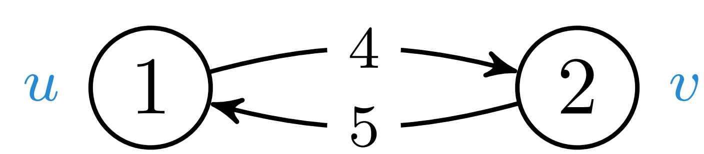
Bakover: Kan sende tilbake og omdirigere $f(v,u)$ enheter

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in \mathrm{E}\,, \\ f(v,u) & \text{if } (v,u) \in \mathrm{E}\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

G



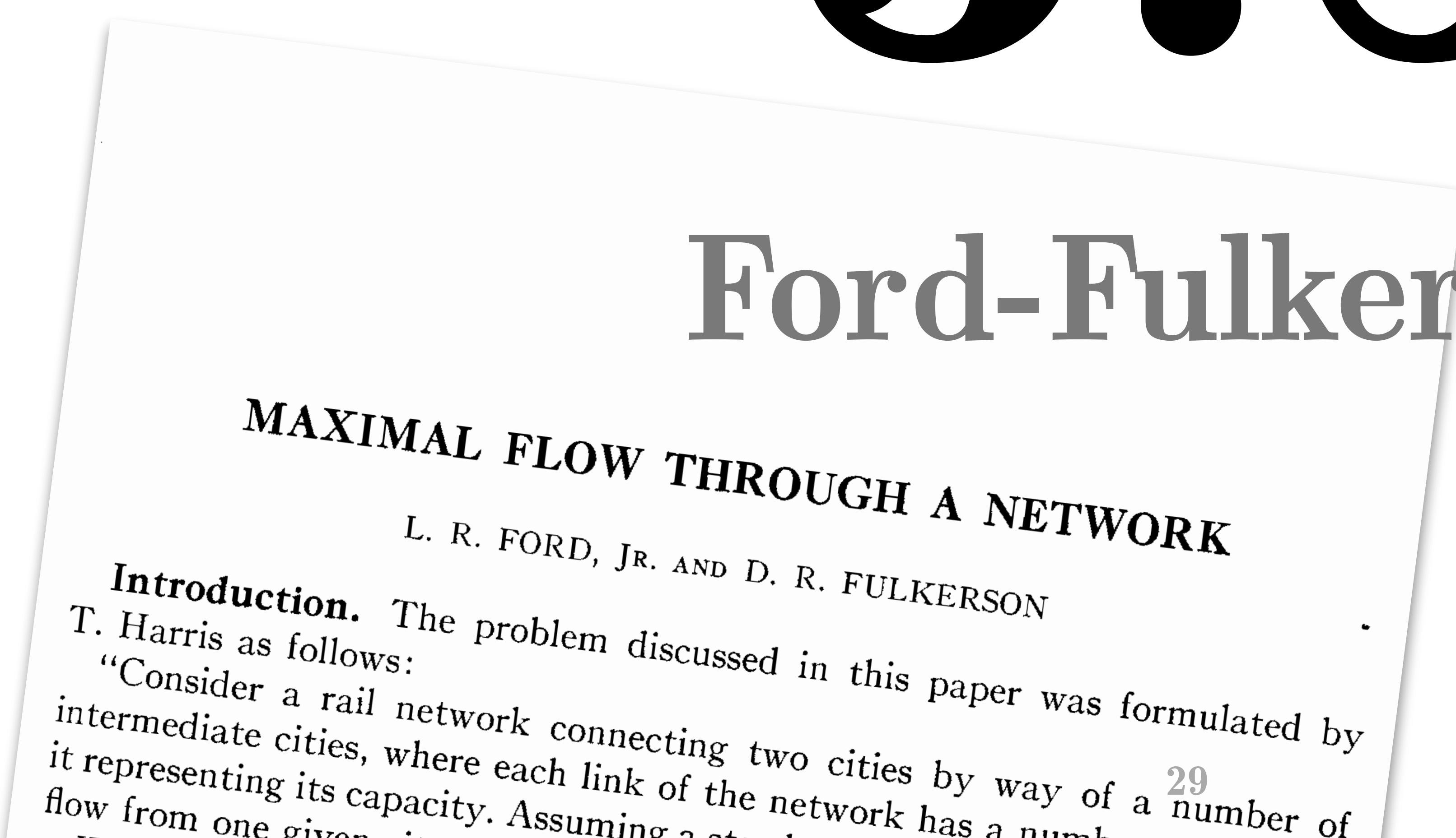
G_f



$$(u,v) \in \mathrm{E}_f \iff c_f(u,v) > 0$$

3:5

Ford-Fulkerson



MAXIMAL FLOW THROUGH A NETWORK

L. R. FORD, JR. AND D. R. FULKERSON

Introduction. The problem discussed in this paper was formulated by T. Harris as follows:

"Consider a rail network connecting two cities by way of a number of intermediate cities, where each link of the network has a numb... it representing its capacity. Assuming a st... flow from one given ...

Fra 1956

- **Finn forøkende stier så lenge det går**
- **Deretter er flyten maksimal**
- **Generell metode, ikke en algoritme**
- **Om vi bruker BFS: «Edmonds-Karp»**

- **Normal implementasjon:**
  - **Finn forøkende sti først**
  - **Finn så flaskehalsen i stien**
  - **Oppdater flyt langs stien med denne verdien**

FORD-FULKERSON-METHOD(G, s, t)

G	flytnett
s	kilde
t	sluk

Finn maksimal flyt fra s til t i G

FORD-FULKERSON-METHOD(G, s, t)
1 initialize flow f to 0

G	flytnett
s	kilde
t	sluk
f	flyt

Lovlig, men neppe optimal løsning. Forbedres gradvis

Ford-Fulkerson-Method(G, s, t)

initialize flow f to 0

while there is an augm. path p in G_f

G	flytnett
s	kilde
t	sluk
f	flyt
G_f	restnett
p	$s \rightsquigarrow t$ i G_f

Ikke nødvendigvis en sti i G; kanter kan gå baklengs i p der

FORD-FULKERSON-METHOD(G, s, t)
1 initialize flow f to 0
2 **while** there is an augm. path p in G_f
3 augment flow f along p

G	flytnett
s	kilde
t	sluk
f	flyt
G_f	restnett
p	$s \rightsquigarrow t$ i G_f

Flyten i hver kant kan økes (→) eller oppheves og omdirigeres (←)

Ford-Fulkerson-Method(G, s, t)
initialize flow f to 0
while there is an augm. path p in G_f
    augment flow f along p
return f

G	flytnett
s	kilde
t	sluk
f	flyt
G_f	restnett
p	$s \rightsquigarrow t$ i G_f

mlh@ntnu.no

Ford-Fulkerson(G, s, t)

G	flyttnett
s	kilde
t	sluk

En litt mer detaljert beskrivelse av flyt-oppdateringen

```
FORD-FULKERSON(G, s, t)
for each edge (u, v) ∈ G.E
```

G	flyttnett
s	kilde
t	sluk
u	node
v	node

FORD-FULKERSON(G, s, t)

for each edge $(u, v) \in$ G.E
    $(u, v).f = 0$

G	flyttnett
s	kilde
t	sluk
u	node
v	node
f	flyt

Ford-Fulkerson(G, s, t)

for each edge $(u, v) \in G.E$
    $(u, v).f = 0$
while there is a path p from s to t in G_f

G	flyttnett
s	kilde
t	sluk
u	node
v	node
f	flyt
G_f	restnett
p	$s \rightsquigarrow t$ i G_f

Alle kanter i G_f har restkapasitet. En sti $s \rightsquigarrow t$ er dermed forøkende

```
FORD-FULKERSON(G, s, t)
for each edge (u, v) ∈ G.E
        (u, v).f = 0
while there is a path p from s to t in G_f
        c_f(p) = min {c_f(u, v) : (u, v) is in p}
```

G	flyttnett
s	kilde
t	sluk
u	node
v	node
f	flyt
G_f	restnett
p	$s \rightsquigarrow t$ i G_f
c_f	restkap.

Dette er «flaskehalsen» langs p: Minste restkapasitet

Ford-Fulkerson(G, s, t)
for each edge $(u, v) \in$ G.E
    $(u, v).f = 0$
while there is a path p from s to t in G_f
    $c_f(p) = \min\{c_f(u, v) : (u, v) \text{ is in } p\}$
    for each edge (u, v) in p

G	flyttnett
s	kilde
t	sluk
u	node
v	node
f	flyt
G_f	restnett
p	$s \leadsto t$ i G_f
c_f	restkap.

Vi vil øke flyten langs p med denne flaskehals-restkapasiteten

```
FORD-FULKERSON(G, s, t)
for each edge (u, v) ∈ G.E
    (u, v).f = 0
while there is a path p from s to t in G_f
    c_f(p) = min {c_f(u, v) : (u, v) is in p}
    for each edge (u, v) in p
        if (u, v) ∈ E
```

G	flyttnett
s	kilde
t	sluk
u	node
v	node
f	flyt
G_f	restnett
p	$s \rightsquigarrow t$ i G_f
c_f	restkap.

Foroverkant med restkapasitet?

FORD-FULKERSON(G, s, t)

for each edge $(u, v) \in G.E$
    $(u, v).f = 0$
while there is a path p from s to t in G_f
    $c_f(p) = \min\{c_f(u, v) : (u, v) \text{ is in } p\}$
    for each edge (u, v) in p
        if $(u, v) \in E$
            $(u, v).f = (u, v).f + c_f(p)$

G	flyttnett
s	kilde
t	sluk
u	node
v	node
f	flyt
G_f	restnett
p	$s \leadsto t$ i G_f
c_f	restkap.

Da kan flyten økes langs kanten

FORD-FULKERSON(G, s, t)

for each edge $(u, v) \in G.E$
    $(u, v).f = 0$
while there is a path p from s to t in G_f
    $c_f(p) = \min\{c_f(u, v) : (u, v) \text{ is in } p\}$
    for each edge (u, v) in p
        if $(u, v) \in E$
            $(u, v).f = (u, v).f + c_f(p)$
        else $(v, u).f = (v, u).f - c_f(p)$

G	flyttnett
s	kilde
t	sluk
u	node
v	node
f	flyt
G_f	restnett
p	$s \rightsquigarrow t$ i G_f
c_f	restkap.

Baklengs: Opphev flyt; omdirigeres implisitt til neste kant i p

- **Alternativ: «Flett inn» BFS**
  - **Finn flaskehalser underveis!**
  - **Hold styr på hvor mye flyt vi får frem til hver node**
  - **Traverser bare noder vi ikke har nådd frem til ennå**
- **Denne «implementasjonen» står ikke i boka**

Dette er ment å gi en mer konkret og detaljert forståelse av hvordan Edmonds-Karp fungerer, og hvordan den kan implementeres, men dette er kun et supplement for økt forståelse. Du trenger ikke «pugge» denne detaljerte varianten, eller huske akkurat hvordan den fungerer, så lenge du har skjønt og husker det som står i pensum.

$$v.a$$

Mulig økning (*augmentation*)

Hvor mye mer flyt får vi til å sende fra s til v?

$$v.a$$

Mulig økning (*augmentation*)

Gitt av flaskehalsen $c_f(s \rightsquigarrow v)$ for en eller annen sti $s \rightsquigarrow v$

$$v.a$$

Mulig økning (*augmentation*)

Etter traversering er $t.a = c_f(p)$ for en forøkende sti p (eller 0)

Akkurat navnet «a» er bare et vilkårlig valg fra min side.

$$v.a$$

Mulig økning (*augmentation*)

Implementasjonsdetalj fra original-algoritmen; diskuteres ikke i boka

Edmonds-Karp(G, s, t)

G flytnett
s kilde
t sluk

Bruker BFS for å finne forøkende sti. Atskillig mer detaljert …

EDMONDS-KARP(G, s, t)

for each edge $(u, v) \in$ G.E

G flytnett
s kilde
t sluk
u node
v node

Edmonds-Karp(G, s, t)
1 **for** each edge $(u, v) \in$ G.E
2 $\quad (u, v).f = 0$

G	flytnett
s	kilde
t	sluk
u	node
v	node
f	flyt

$\textsc{Edmonds-Karp}(\mathrm{G}, s, t)$

```
1  for each edge (u, v) ∈ G.E
2      (u, v).f = 0
3  repeat  ▷ until t.a == 0
```

G	flytnett
s	kilde
t	sluk
u	node
v	node
f	flyt
a	økning

Gjenta til vi ikke får mer flyt frem til t

Edmonds-Karp(G, s, t)

```
for each edge (u, v) ∈ G.E
    (u, v).f = 0
repeat › until t.a == 0
    for each vertex u ∈ G.V
```

G	flytnett
s	kilde
t	sluk
u	node
v	node
f	flyt
a	økning

I hver iterasjon . . .

```
EDMONDS-KARP(G, s, t)
for each edge (u, v) ∈ G.E
        (u, v).f = 0
repeat  ▹ until t.a == 0
        for each vertex u ∈ G.V
                u.a = 0  ▹ reaching u in G_f
```

G	flytnett
s	kilde
t	sluk
u	node
v	node
f	flyt
a	økning

Hvor mye mer flyt klarer vi å få frem til u?

Edmonds-Karp(G, s, t)

for each edge $(u, v) \in G.E$
    $(u, v).f = 0$
repeat › *until* $t.a == 0$
    for each vertex $u \in G.V$
        $u.a = 0$ › *reaching* u *in* G_f
        $u.\pi = \text{NIL}$

G	flytnett
s	kilde
t	sluk
u	node
v	node
f	flyt
a	økning

Dette er forgjengeren i BFS-treet

```
EDMONDS-KARP(G, s, t)
for each edge (u, v) ∈ G.E
        (u, v).f = 0
repeat  › until t.a == 0
        for each vertex u ∈ G.V
                u.a = 0  › reaching u in G_f
                u.π = NIL
        s.a = ∞
```

G	flytnett
s	kilde
t	sluk
u	node
v	node
f	flyt
a	økning

Alltid uendelig mye mer flyt hos s. Vi vil sende den videre

EDMONDS-KARP(G, s, t)

for each edge $(u, v) \in$ G.E
    $(u, v).f = 0$
repeat › *until* $t.a == 0$
    for each vertex $u \in$ G.V
        $u.a = 0$ › *reaching* u *in* G_f
        $u.\pi =$ NIL
    $s.a = \infty$
    $Q = \emptyset$

G	flytnett
s	kilde
t	sluk
u	node
v	node
f	flyt
a	økning
Q	kø

FIFO-kø til bredde-først-søk i G_f

```
EDMONDS-KARP(G, s, t)
for each edge (u, v) ∈ G.E
    (u, v).f = 0
repeat  ▷ until t.a == 0
    for each vertex u ∈ G.V
        u.a = 0  ▷ reaching u in G_f
        u.π = NIL
    s.a = ∞
    Q = ∅
    ENQUEUE(Q, s)
```

G	flytnett
s	kilde
t	sluk
u	node
v	node
f	flyt
a	økning
Q	kø

Vi traverserer fra s, og leter etter t

```
EDMONDS-KARP(G, s, t)
for each edge (u, v) ∈ G.E
    (u, v).f = 0
repeat › until t.a == 0
    for each vertex u ∈ G.V
        u.a = 0 › reaching u in G_f
        u.π = NIL
    s.a = ∞
    Q = ∅
    ENQUEUE(Q, s)
    . . .
```

G	flytnett
s	kilde
t	sluk
u	node
v	node
f	flyt
a	økning
Q	kø

EDMONDS-KARP(G, s, t)
9 ...

G flytnett
s kilde
t sluk

$\textsc{Edmonds-Karp}(G, s, t)$

```
        ...
        while t.a == 0 and Q ≠ ∅
```

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø

Her begynner BFS. Avbrytes hvis vi finner t

Edmonds-Karp(G, s, t)

    …
    while $t.a == 0$ and $\mathrm{Q} \neq \emptyset$
        $u = \textsc{Dequeue}(\mathrm{Q})$

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node

Neste node vi skal besøke

```
EDMONDS-KARP(G, s, t)
        ...
        while t.a == 0 and Q ≠ ∅
            u = DEQUEUE(Q)
            for all edges (u, v), (v, u) ∈ G.E
```

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node
v	node

For alle potensielle naboer i G_f: Se etter positiv restkapasitet

Edmonds-Karp(G, s, t)

```
        ...
        while t.a == 0 and Q ≠ ∅
            u = Dequeue(Q)
            for all edges (u, v), (v, u) ∈ G.E
                if (u, v) ∈ G.E
```

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node
v	node

Fremoverkant?

Edmonds-Karp(G, s, t)

```
    ...
    while t.a == 0 and Q ≠ ∅
        u = Dequeue(Q)
        for all edges (u, v), (v, u) ∈ G.E
            if (u, v) ∈ G.E
                c_f(u, v) = c(u, v) − (u, v).f
```

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node
v	node
c	kapasitet
c_f	restkap.
f	flyt

Restkapasitet = mulig økning av flyt langs kanten

```
EDMONDS-KARP(G, s, t)
        ...
        while t.a == 0 and Q ≠ ∅
            u = DEQUEUE(Q)
            for all edges (u, v), (v, u) ∈ G.E
                if (u, v) ∈ G.E
                    c_f(u, v) = c(u, v) − (u, v).f
                else c_f(u, v) = (v, u).f
```

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node
v	node
c	kapasitet
c_f	restkap.
f	flyt

Bakoverkant: Restkapasitet = mulig oppheving/omdirigering

```
Edmonds-Karp(G, s, t)
        ...
        while t.a == 0 and Q ≠ ∅
            u = Dequeue(Q)
            for all edges (u, v), (v, u) ∈ G.E
                if (u, v) ∈ G.E
                    c_f(u, v) = c(u, v) − (u, v).f
                else c_f(u, v) = (v, u).f
                if c_f(u, v) > 0 and v.a == 0
```

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node
v	node
c	kapasitet
c_f	restkap.
f	flyt
a	økning

Restkapasitet $\iff$ kant i G_f. Ingen flytøkning $\iff$ ubesøkt (hvit)

EDMONDS-KARP(G, s, t)

```
    ...
    while t.a == 0 and Q ≠ ∅
        u = DEQUEUE(Q)
        for all edges (u, v), (v, u) ∈ G.E
            if (u, v) ∈ G.E
                c_f(u, v) = c(u, v) − (u, v).f
            else c_f(u, v) = (v, u).f
            if c_f(u, v) > 0 and v.a == 0
                v.a = min(u.a, c_f(u, v))
```

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node
v	node
c	kapasitet
c_f	restkap.
f	flyt
a	økning

Vi har med oss $u.a$ ekstra flyt; får maks $c_f(u, v)$ gjennom (u, v)

Edmonds-Karp(G, s, t)

…

while $t.a == 0$ and $Q \neq \emptyset$

  $u = \text{Dequeue}(Q)$

  for all edges $(u, v), (v, u) \in G.E$

    if $(u, v) \in G.E$

      $c_f(u, v) = c(u, v) - (u, v).f$

    else $c_f(u, v) = (v, u).f$

    if $c_f(u, v) > 0$ and $v.a == 0$

      $v.a = \min(u.a, c_f(u, v))$

      $v.\pi = u$

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node
v	node
c	kapasitet
c_f	restkap.
f	flyt
a	økning
π	forgjenger

Hvor kom vi fra da vi oppdaget v?

EDMONDS-KARP(G, s, t)

    …
    while $t.a == 0$ and $\mathrm{Q} \neq \emptyset$
        $u = $ DEQUEUE(Q)
        for all edges $(u, v), (v, u) \in \mathrm{G.E}$
            if $(u, v) \in \mathrm{G.E}$
                $c_f(u, v) = c(u, v) - (u, v).f$
            else $c_f(u, v) = (v, u).f$
            if $c_f(u, v) > 0$ and $v.a == 0$
                $v.a = \min(u.a, c_f(u, v))$
                $v.\pi = u$
                ENQUEUE(Q, v)

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node
v	node
c	kapasitet
c_f	restkap.
f	flyt
a	økning
π	forgjenger

Så vi husker å traversere v senere, og sende flyt videre fra den

```
EDMONDS-KARP(G, s, t)
    ...
    while t.a == 0 and Q ≠ ∅
        u = DEQUEUE(Q)
        for all edges (u, v), (v, u) ∈ G.E
            if (u, v) ∈ G.E
                c_f(u, v) = c(u, v) − (u, v).f
            else c_f(u, v) = (v, u).f
            if c_f(u, v) > 0 and v.a == 0
                v.a = min(u.a, c_f(u, v))
                v.π = u
                ENQUEUE(Q, v)
    ...
```

G	flytnett
s	kilde
t	sluk
a	økning
Q	kø
u	node
v	node
c	kapasitet
c_f	restkap.
f	flyt
a	økning
π	forgjenger

$\textsc{Edmonds-Karp}(\mathrm{G}, s, t)$

19 $\quad\dots$

G flytnett
s kilde
t sluk

Slutt på **while**: Har fått frem flyt til t eller gått tom for noder

$\textsc{Edmonds-Karp}(G, s, t)$

$\ldots$

$u,\ v = t.\pi,\ t$ ▷ *at this point,* $t.a = c_f(p)$

G	flytnett
s	kilde
t	sluk
u	node
v	node
π	forgjenger

Vi har fått frem $t.a$ ekstra flyt, kun begrenset av flaskehalsen $c_f(p)$

Edmonds-Karp(G, s, t)

```
19                    ...
20      u, v = t.π, t    ▷ at this point, t.a = c_f(p)
21      while u ≠ NIL
```

G	flytnett
s	kilde
t	sluk
u	node
v	node
π	forgjenger

Tilbakesporing: Følg π-feltene tilbake fra t til s

```
Edmonds-Karp(G, s, t)
                  …
      u, v = t.π, t    ▷ at this point, t.a = c_f(p)
      while u ≠ NIL
          if (u, v) ∈ G.E
```

G	flytnett
s	kilde
t	sluk
u	node
v	node
π	forgjenger

Fremoverkant?

Edmonds-Karp(G, s, t)

$\dots$

$u, v = t.\pi, t$ ▷ *at this point,* $t.a = c_f(p)$

while $u \neq$ NIL

  if $(u, v) \in$ G.E

    $(u, v).f = (u, v).f + t.a$

G	flytnett
s	kilde
t	sluk
u	node
v	node
π	forgjenger
f	flyt
a	økning

Øk flyten med $t.a$, dvs. $c_f(p)$

Edmonds-Karp(G, s, t)

…

  $u,\ v = t.\pi,\ t$ › *at this point,* $t.a = c_f(p)$

  while $u \neq$ NIL

    if $(u, v) \in$ G.E

      $(u, v).f = (u, v).f + t.a$

    else $(v, u).f = (v, u).f - t.a$

G	flytnett
s	kilde
t	sluk
u	node
v	node
π	forgjenger
f	flyt
a	økning

Bakoverkant: Opphev og omdirigér $t.a$, dvs. $c_f(p)$

EDMONDS-KARP(G, s, t)

$\quad\ldots$

$\quad u,\ v = t.\pi,\ t$ › *at this point,* $t.a = c_f(p)$

$\quad$ **while** $u \neq$ NIL

$\quad\quad$ **if** $(u, v) \in G.E$

$\quad\quad\quad (u, v).f = (u, v).f + t.a$

$\quad\quad$ **else** $(v, u).f = (v, u).f - t.a$

$\quad\quad u,\ v = u.\pi,\ u$

G	flytnett
s	kilde
t	sluk
u	node
v	node
π	forgjenger
f	flyt
a	økning

Gå ett skritt bakover langs p

```
EDMONDS-KARP(G, s, t)
                    ...
      u, v = t.π, t   › at this point, t.a = c_f(p)
      while u ≠ NIL
          if (u, v) ∈ G.E
              (u, v).f = (u, v).f + t.a
          else (v, u).f = (v, u).f − t.a
          u, v = u.π, u
  until t.a == 0
```

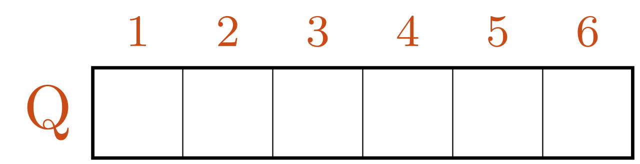
G	flytnett
s	kilde
t	sluk
u	node
v	node
π	forgjenger
f	flyt
a	økning

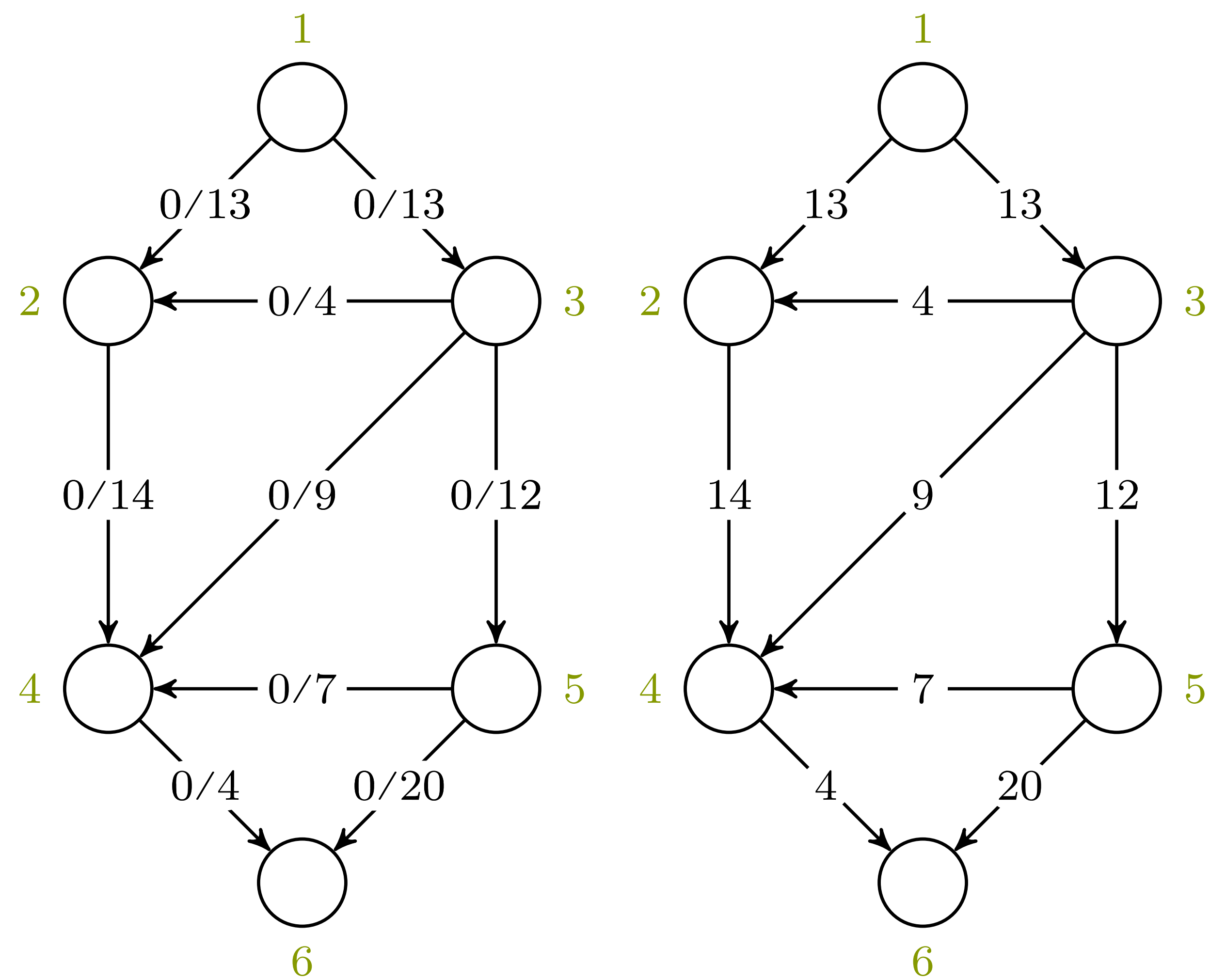
Ingen forøkende sti p funnet; flyten er maksimal

Merk: Selv i den fulle simuleringen så dropper jeg her noen detaljer etter hvert (og hopper forbi noen løkker, etc.).

```
Edmonds-Karp(G, s, t)
for each edge (u, v) ∈ G.E
    (u, v).f = 0
repeat
    for each vertex u ∈ G.V
        u.a = 0   › flow reaching u in G_f
        u.π = NIL
    s.a = ∞
    Q = ∅
    Enqueue(Q, s)
    while t.a == 0 and Q ≠ ∅
        u = Dequeue(Q)
        for all edges (u, v), (v, u) ∈ G.E
            if (u, v) ∈ G.E
                c_f(u, v) = c(u, v) − (u, v).f
            else c_f(u, v) = (v, u).f
            if c_f(u, v) > 0 and v.a == 0
                v.a = min(u.a, c_f(u, v))
                v.π = u
                Enqueue(Q, v)
    u, v = t.π, t   › at this point, t.a == c_f(p)
    while u ≠ NIL
        if (u, v) ∈ G.E
            (u, v).f = (u, v).f + t.a
        else (v, u).f = (v, u).f − t.a
        u, v = u.π, u
until t.a == 0
```

$u, v \;=\; -, -$





lyt

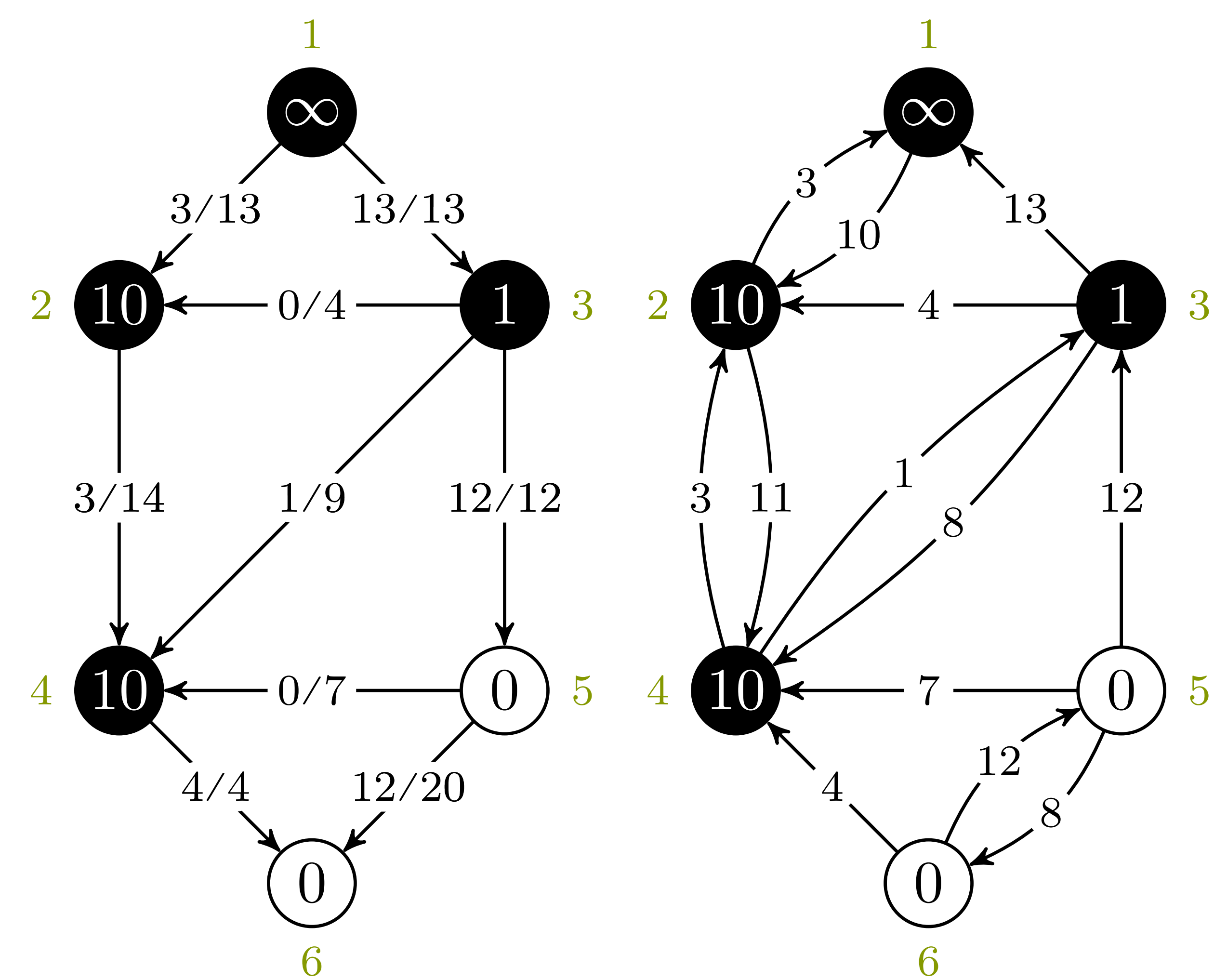
Rest

Node 1 er kilde, node 6 er sluk.

```
EDMONDS-KARP(G, s, t)
for each edge (u, v) ∈ G.E
    (u, v).f = 0
repeat
    for each vertex u ∈ G.V
        u.a = 0   › flow reaching u in G_f
        u.π = NIL
    s.a = ∞
    Q = ∅
    ENQUEUE(Q, s)
    while t.a == 0 and Q ≠ ∅
        u = DEQUEUE(Q)
        for all edges (u, v), (v, u) ∈ G.E
            if (u, v) ∈ G.E
                c_f(u, v) = c(u, v) − (u, v).f
            else c_f(u, v) = (v, u).f
            if c_f(u, v) > 0 and v.a == 0
                v.a = min(u.a, c_f(u, v))
                v.π = u
                ENQUEUE(Q, v)
    u, v = t.π, t   › at this point, t.a == c_f(p)
    while u ≠ NIL
        if (u, v) ∈ G.E
            (u, v).f = (u, v).f + t.a
        else (v, u).f = (v, u).f − t.a
        u, v = u.π, u
until t.a == 0
```

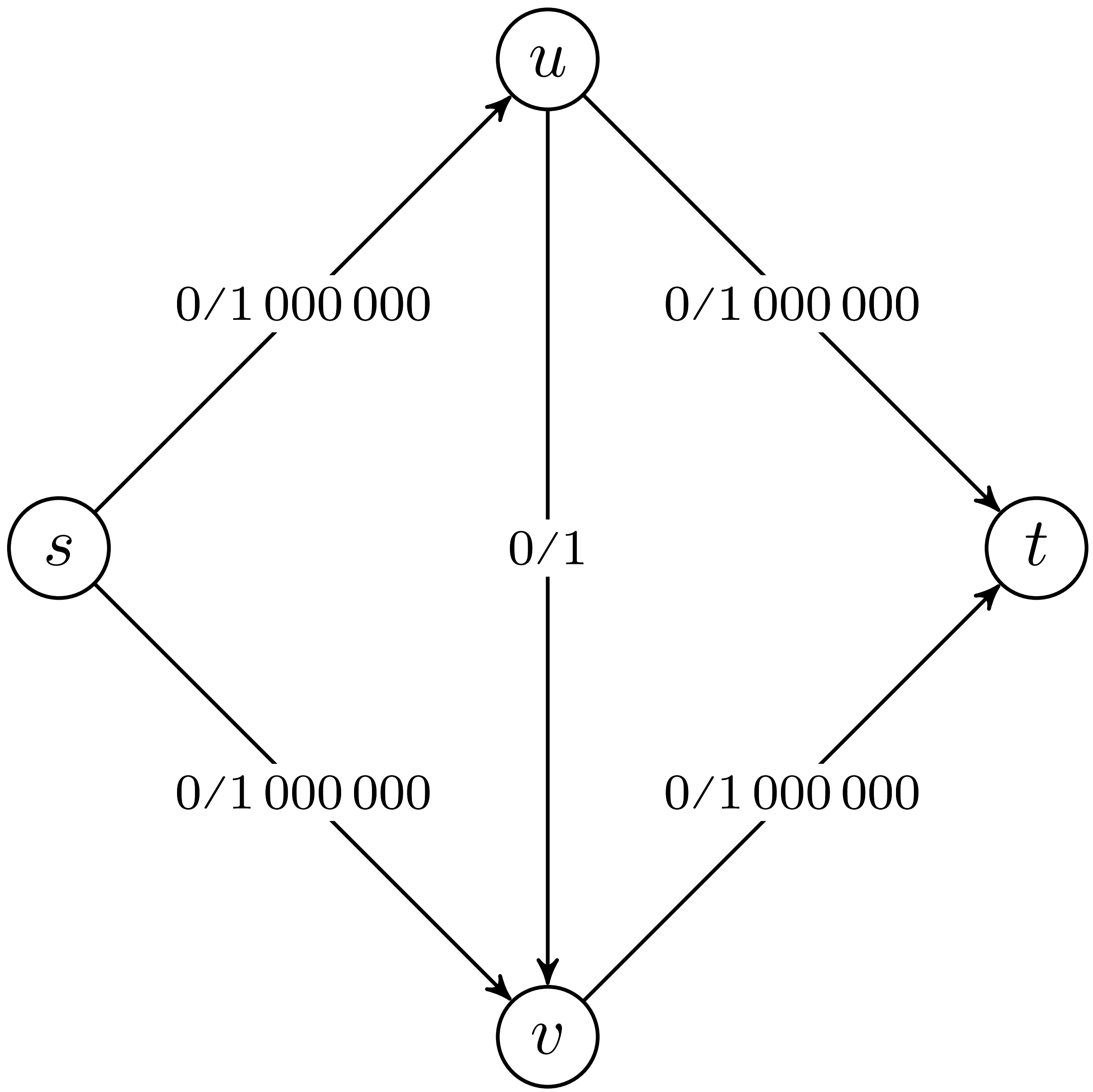
$u, v = -, -$

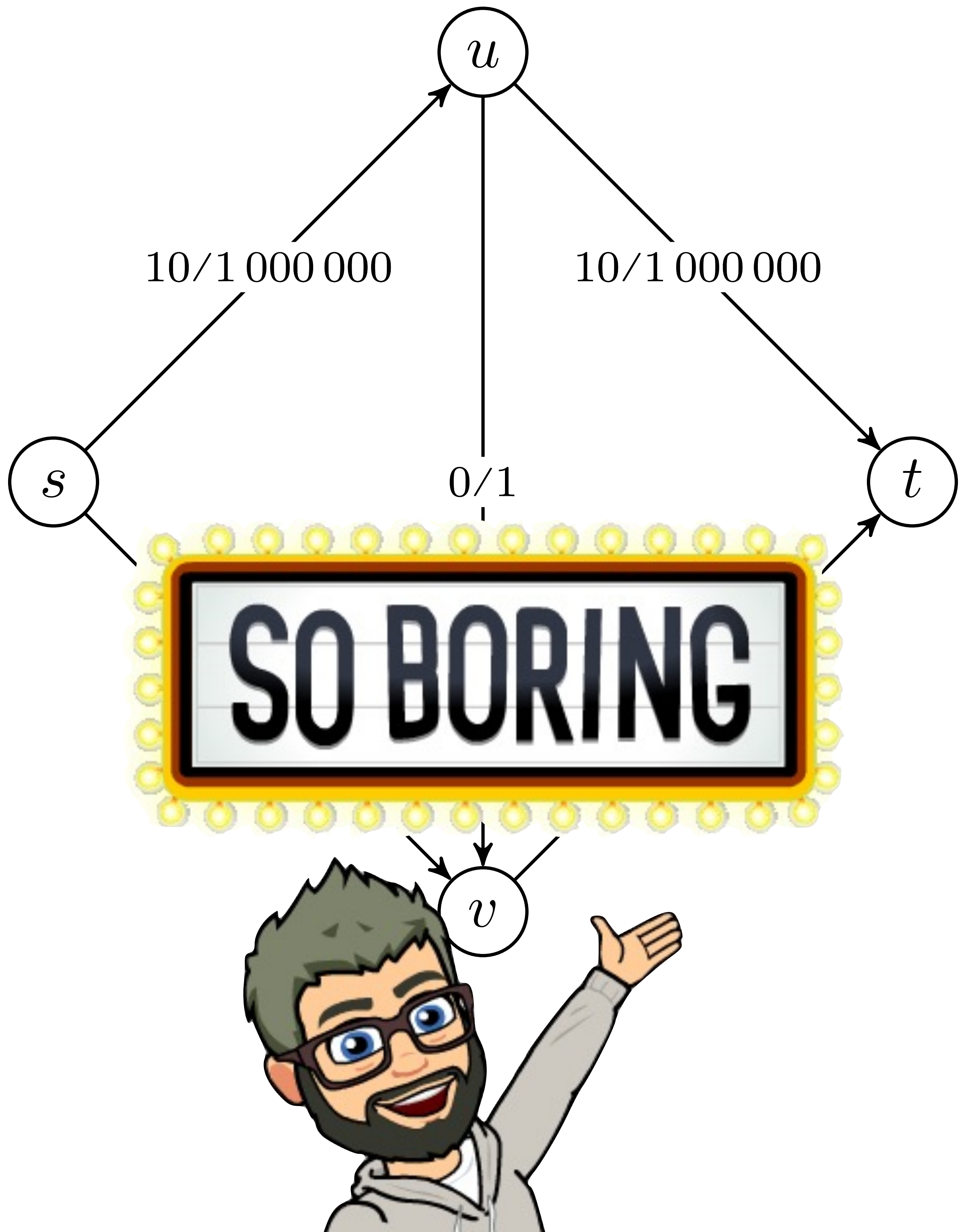
	1	2	3	4	5	6
Q	1	2	4	3	5	6



Flyt Rest

Kan gå ille uten BFS


u
s
t
v
0/1 000 000
0/1 000 000
0/1
0/1 000 000
0/1 000 000


u
10/1 000 000
10/1 000 000
s
0/1
t
SO BORING
v

Operasjon	**Antall**	**Kjøretid**
Finn forøkende sti		

Med uspesifisert traversering i restnett

Operasjon	Antall	Kjøretid
Finn forøkende sti		$O(E)$

Alle nås fra s, så $V + E = \Theta(E)$. Kan stanses tidlig; derfor O

Operasjon	**Antall**	**Kjøretid**
Finn forøkende sti	$O(\lvert f^*\rvert)$	$O(E)$

Heltallskapasiteter: Flyt økes med heltall $\geqslant 1$ (Teorem 26.10)

Operasjon	**Antall**	**Kjøretid**
Finn forøkende sti	$O(\lvert f^* \rvert)$	$O(E)$

Totalt: $O(E\lvert f^* \rvert)$

Rasjonale kapasiteter kan skaleres til heltall

mlh@ntnu.no

Operasjon	**Antall**	**Kjøretid**
Finn forøkende sti	$\mathrm{O}(\lvert f^*\rvert)$	$\mathrm{O}(\mathrm{E})$

Totalt: $\mathrm{O}(\mathrm{E}\lvert f^*\rvert)$

Kapasiteter kan være (eksponentielt) store!

Operasjon	**Antall**	**Kjøretid**
Finn forøkende sti	$O(\|f^*\|)$	$O(E)$

Totalt: $O(E|f^*|)$

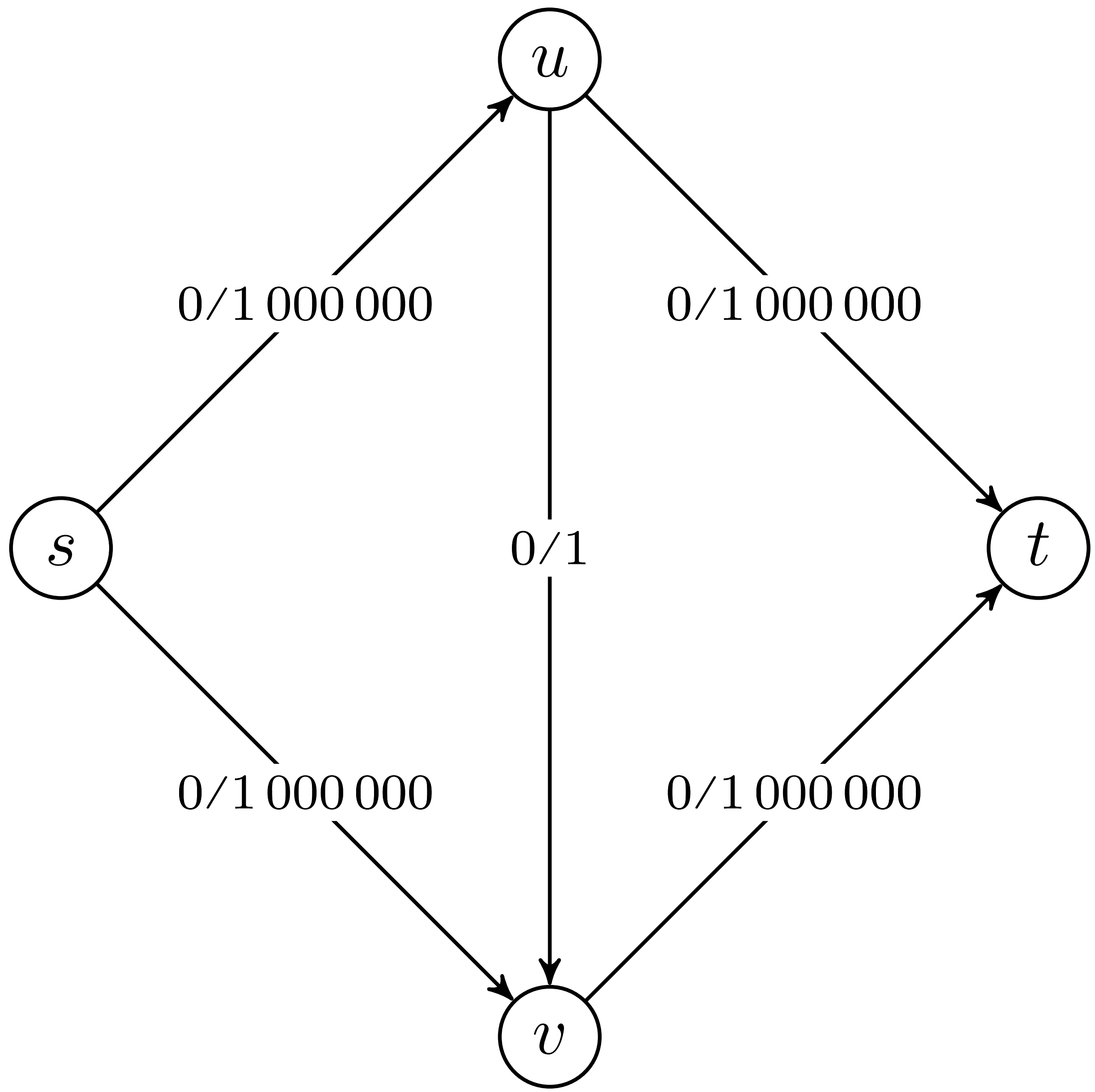
Irrasjonale kapasiteter: Ford-Fulkerson terminerer kanskje ikke!

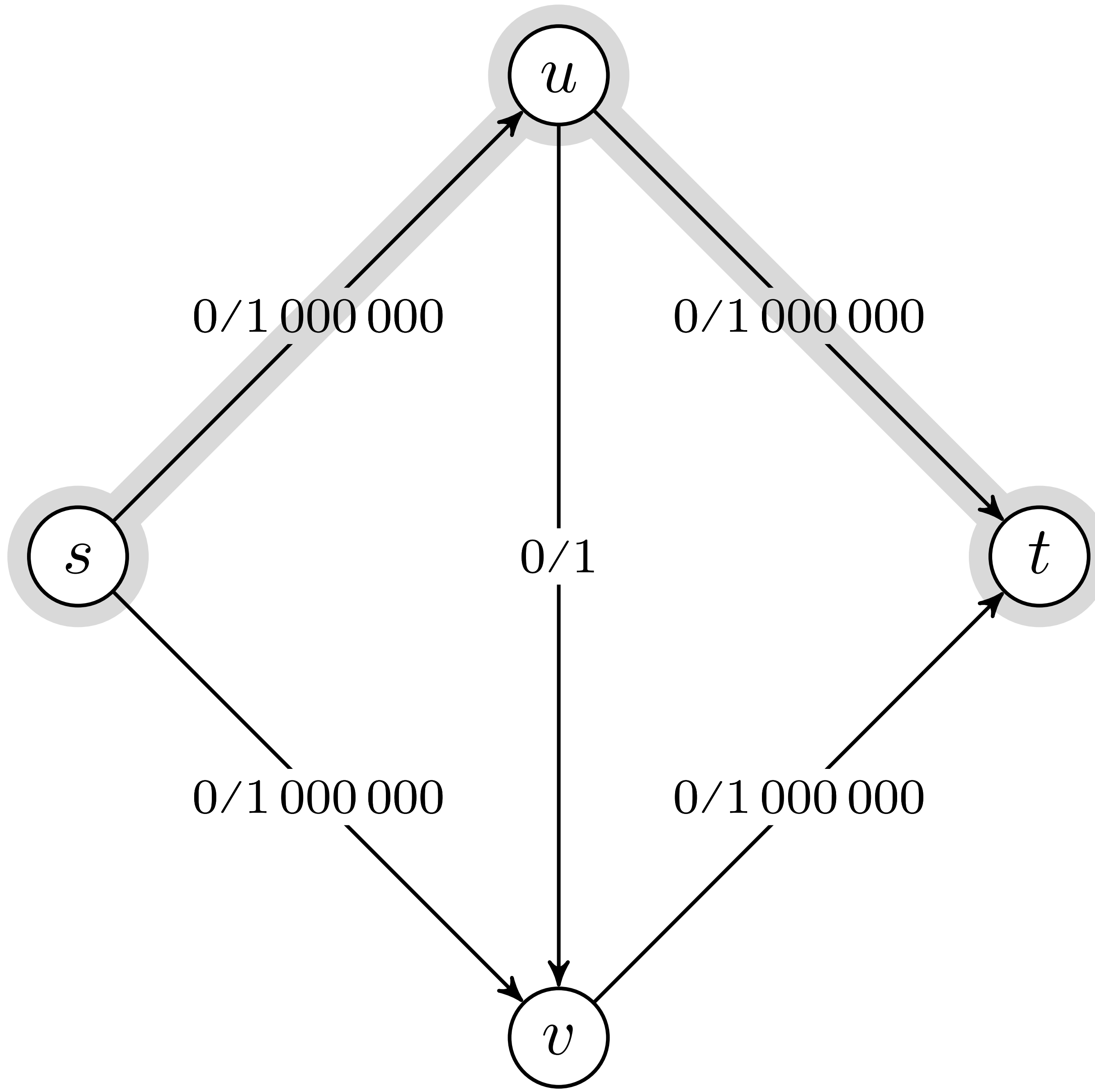
Pseudopolynomisk.

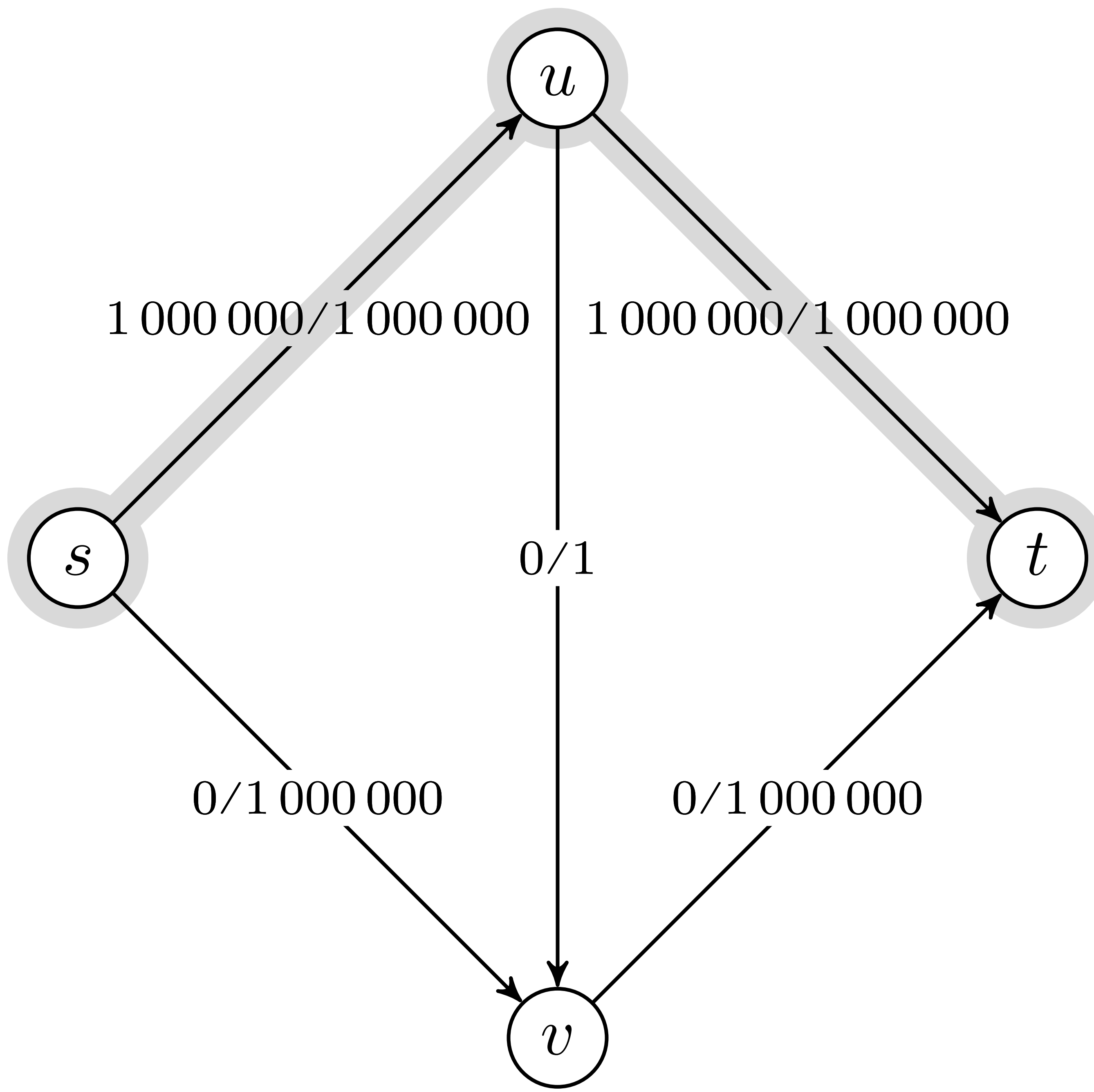
Eksponentielt!

Dette blir som for 0-1-knapsack: Kjøretiden er en polynomisk funksjon av bl.a. ett av tallene i input; dermed er den eksponentiell som en funksjon av problemstørrelsen.

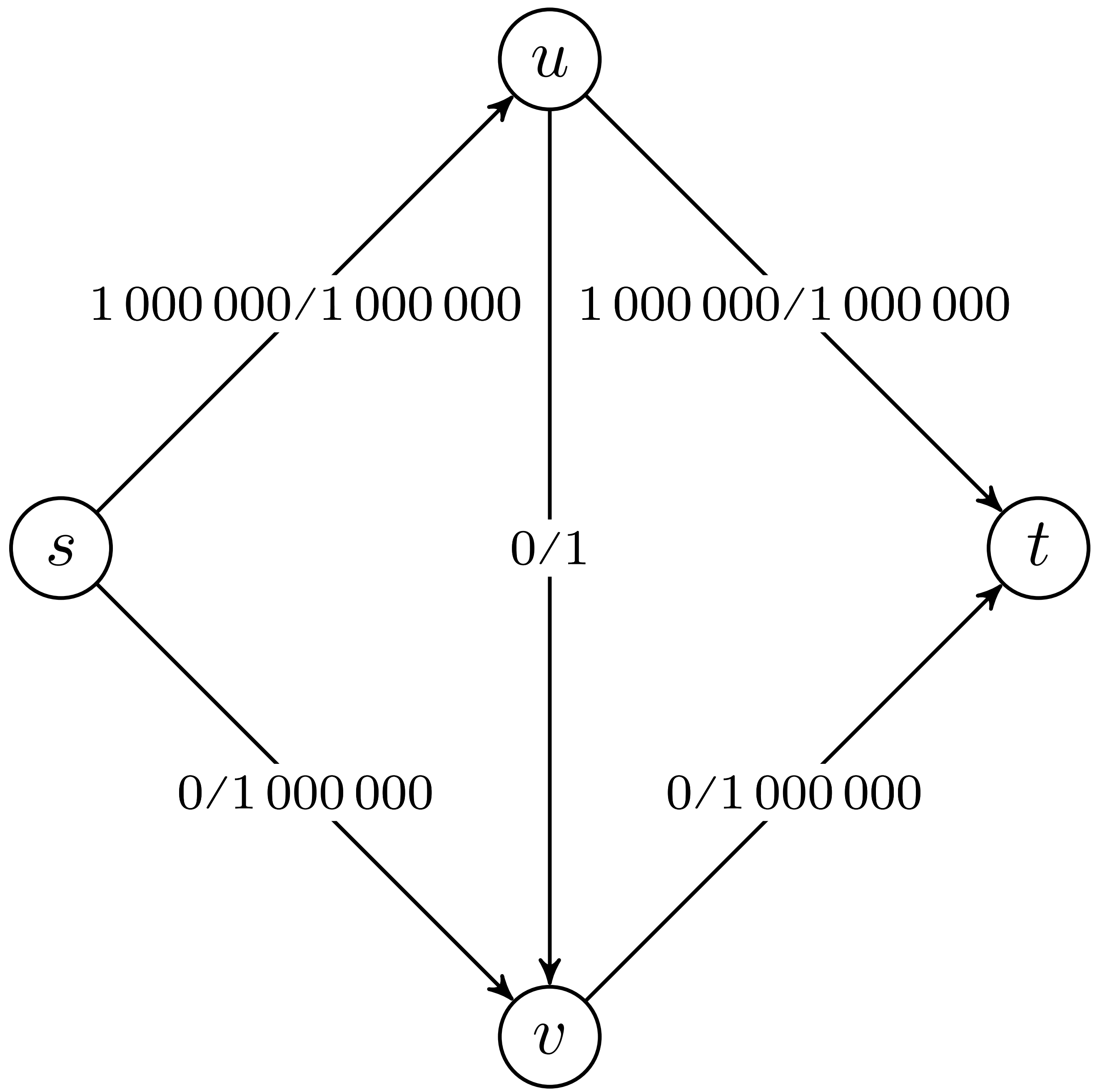
Bruk BFS!

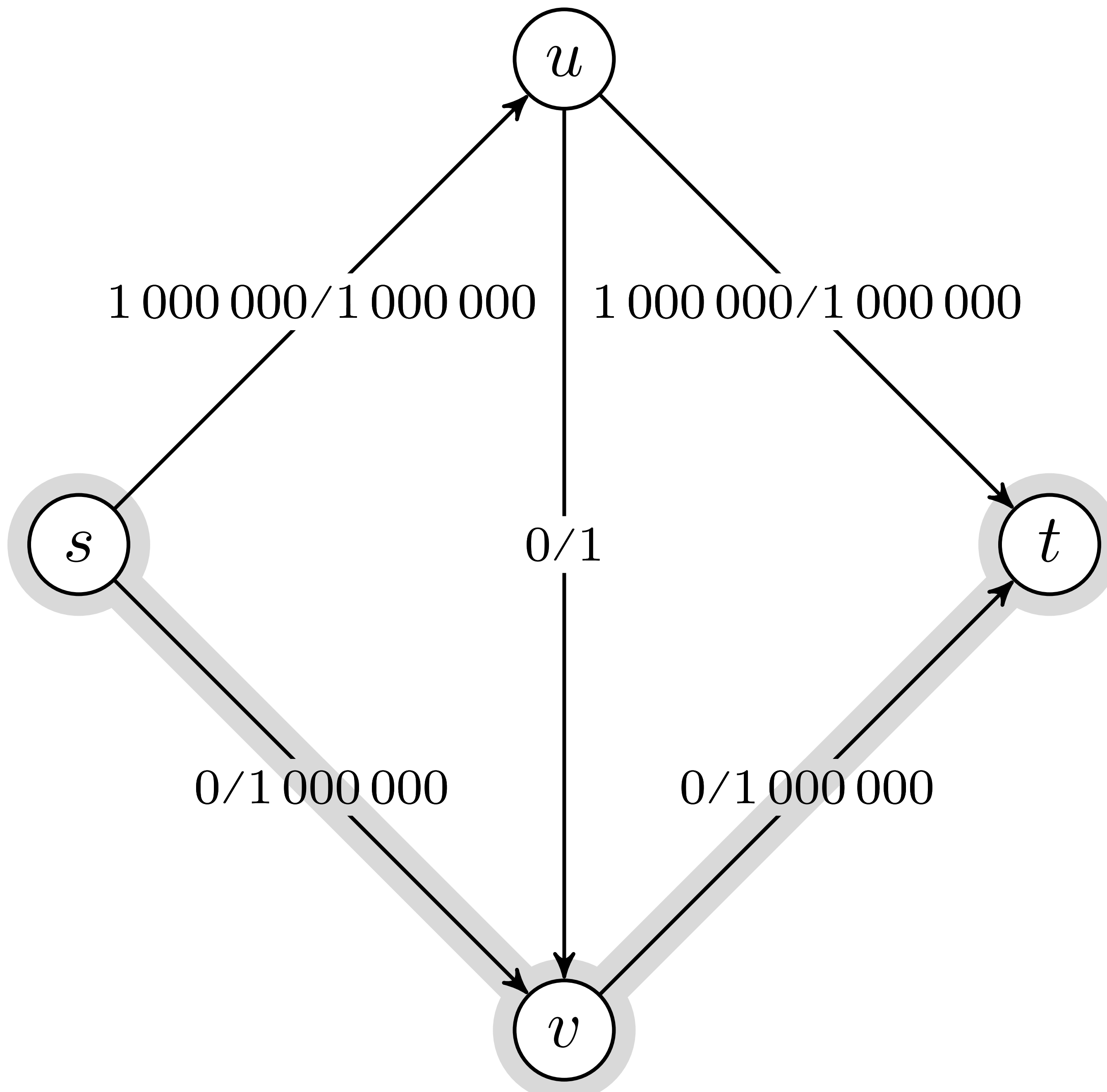

u
s
t
v
0/1 000 000
0/1 000 000
0/1
0/1 000 000
0/1 000 000

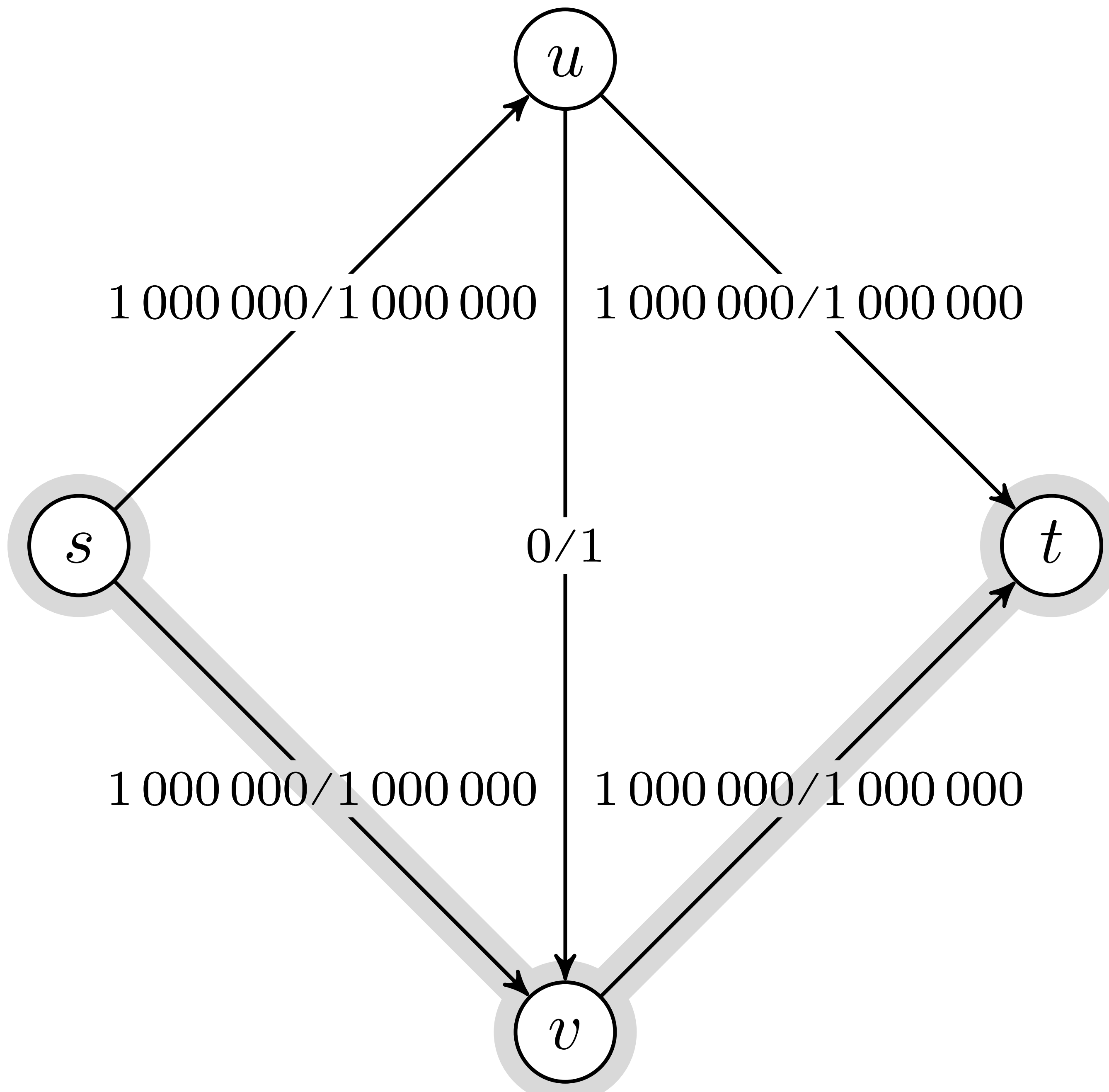

u
s
t
v
0/1 000 000
0/1 000 000
0/1
0/1 000 000
0/1 000 000

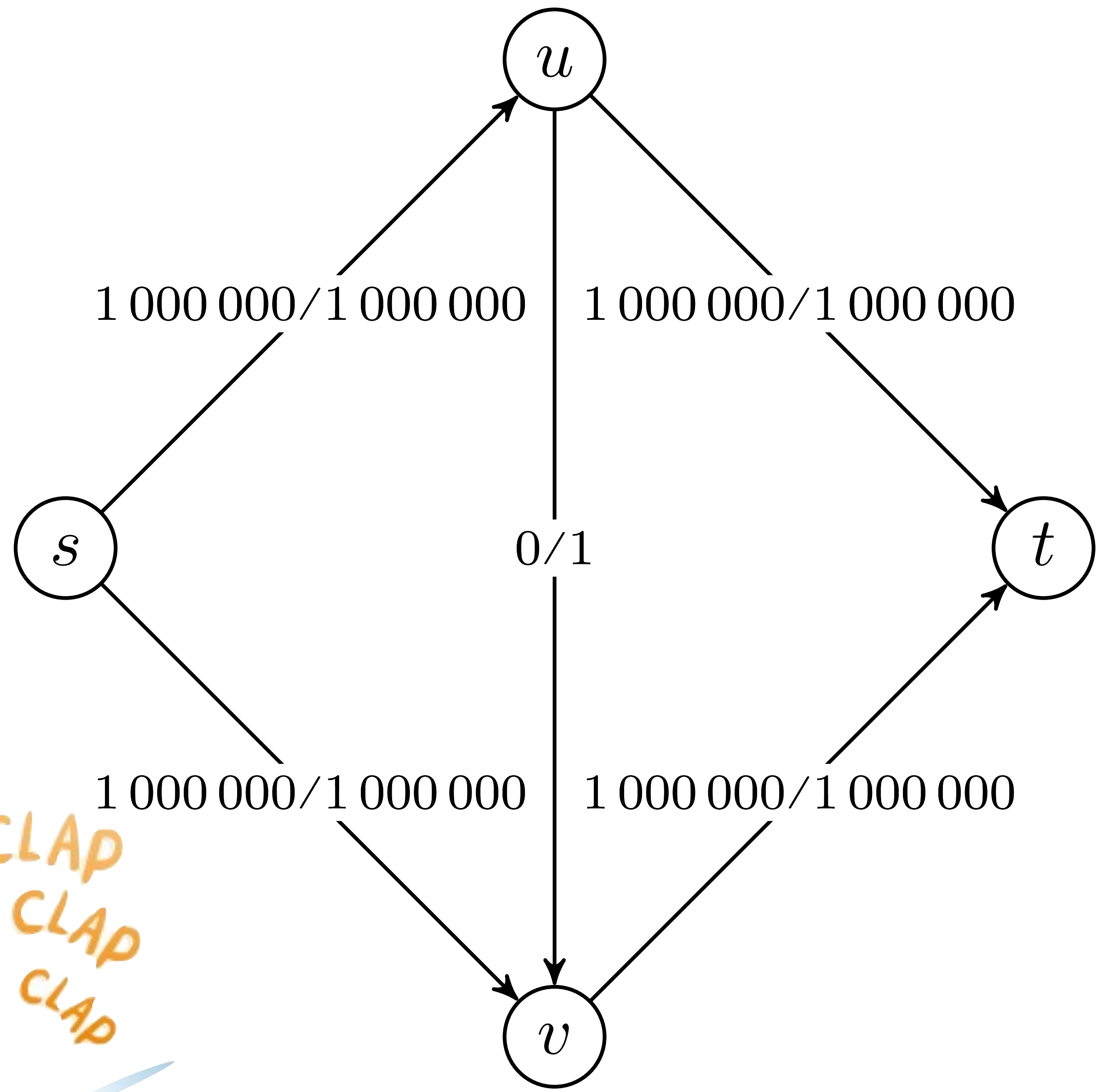


u
s
t
v
1 000 000/1 000 000
1 000 000/1 000 000
0/1
0/1 000 000
0/1 000 000


u
s
t
v
1 000 000/1 000 000
1 000 000/1 000 000
0/1
0/1 000 000
0/1 000 000


u
s
t
v
1 000 000/1 000 000
1 000 000/1 000 000
0/1
0/1 000 000
0/1 000 000


u
s
t
v
1 000 000/1 000 000
1 000 000/1 000 000
0/1
1 000 000/1 000 000
1 000 000/1 000 000


u
s
t
v
1 000 000/1 000 000
1 000 000/1 000 000
0/1
1 000 000/1 000 000
1 000 000/1 000 000


CLAP
CLAP
CLAP

Operasjon	Antall	Kjøretid
Finn forøkende sti		

Med bredde-først-søk i restnett

Operasjon	**Antall**	**Kjøretid**
Finn forøkende sti		O(E)

Alle nås fra s, så $\mathrm{V} + \mathrm{E} = \Theta(\mathrm{E})$. Kan stanses tidlig; derfor O

Operasjon	**Antall**	**Kjøretid**
Finn forøkende sti	O(VE)	O(E)

Det er her forskjellen mellom BFS og f.eks. DFS kommer inn!

Operasjon	**Antall**	**Kjøretid**
Finn forøkende sti	$O(VE)$	$O(E)$

Totalt: $O(VE^2)$

Operasjon	**Antall**	**Kjøretid**
Finn forøkende sti	$O(VE)$	$O(E)$

Totalt: $O(VE^2)$

Men … hvorfor $O(VE)$ iterasjoner?

mlh@ntnu.no

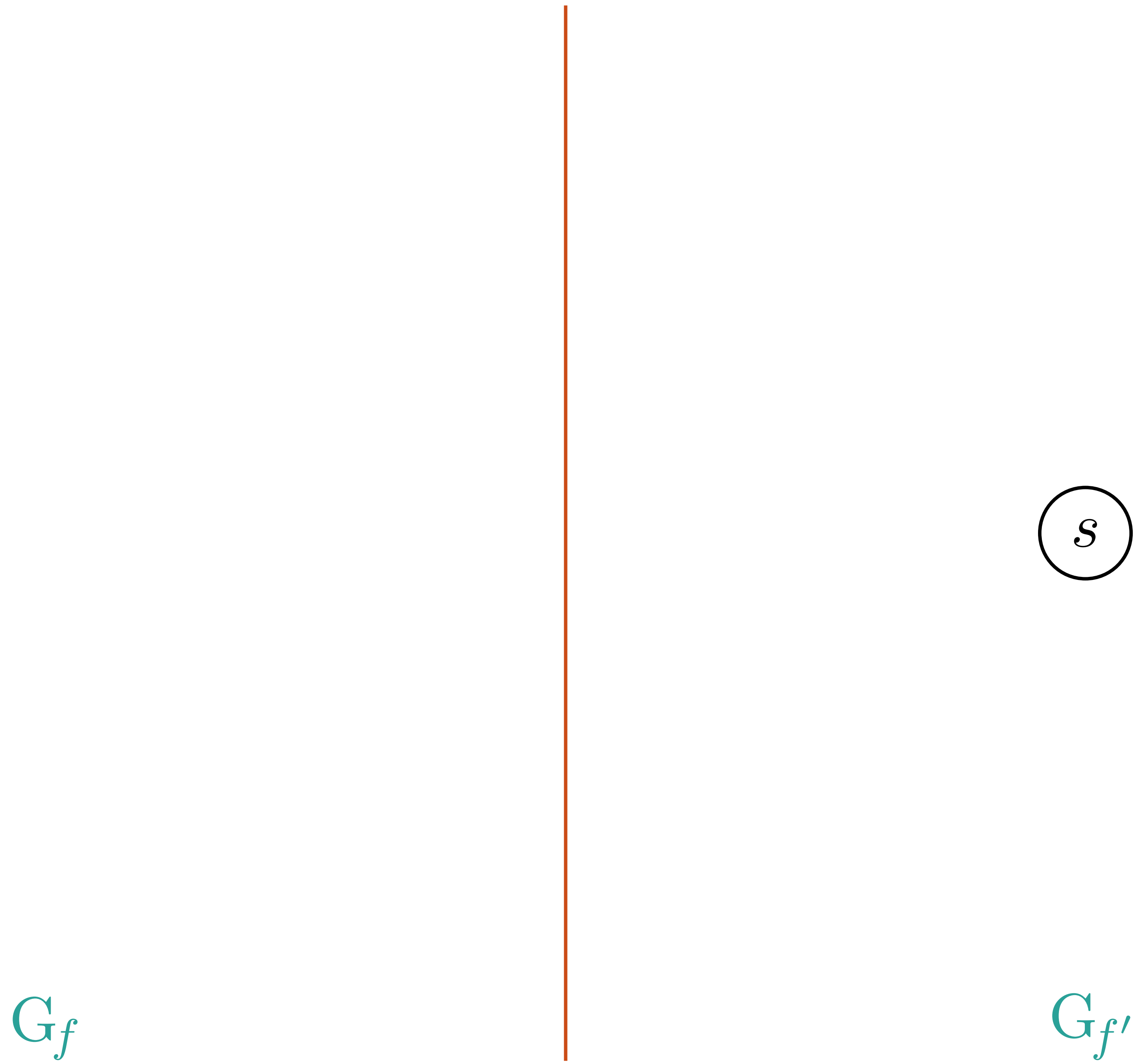
Avstander synker ikke

I restnettet

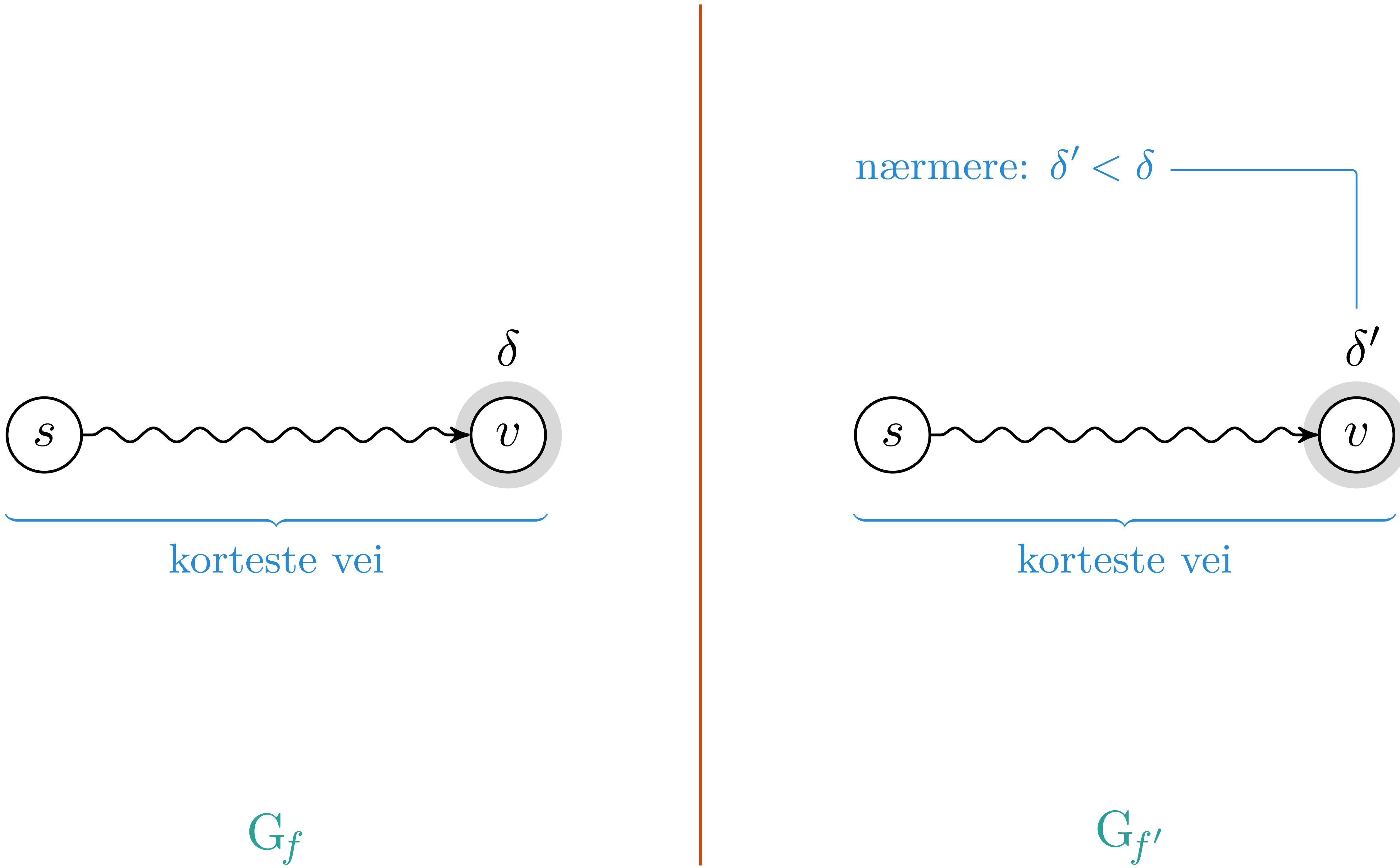
G_f

$G_{f'}$

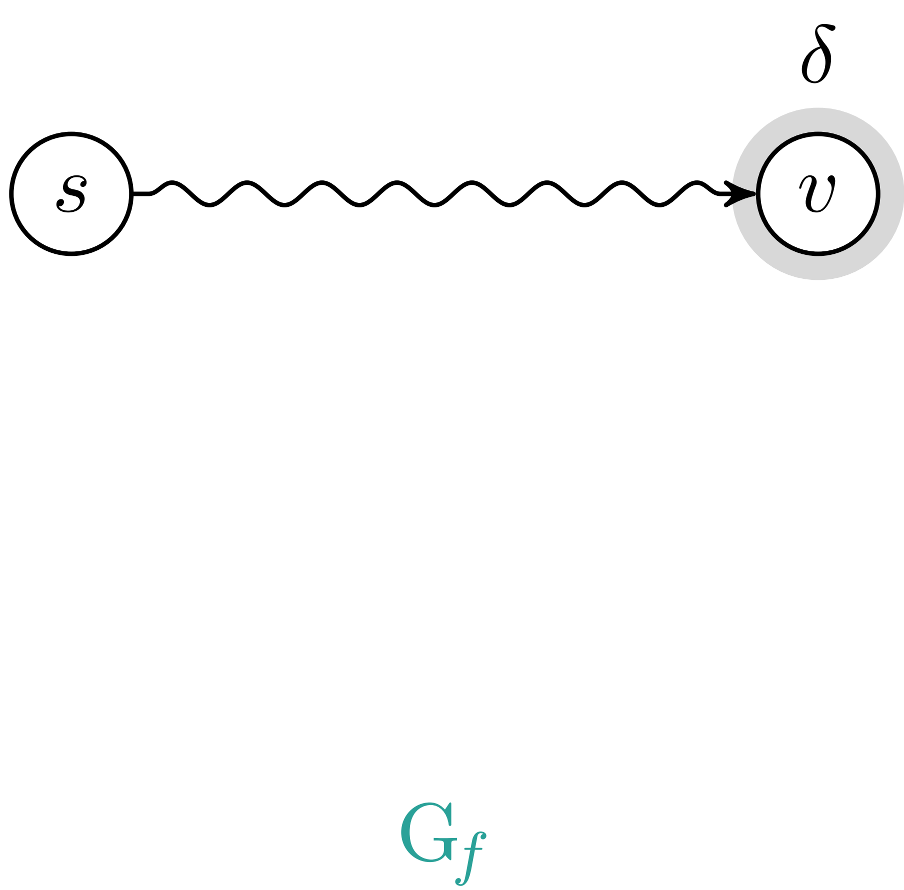
Restnett før og etter en flytøkning

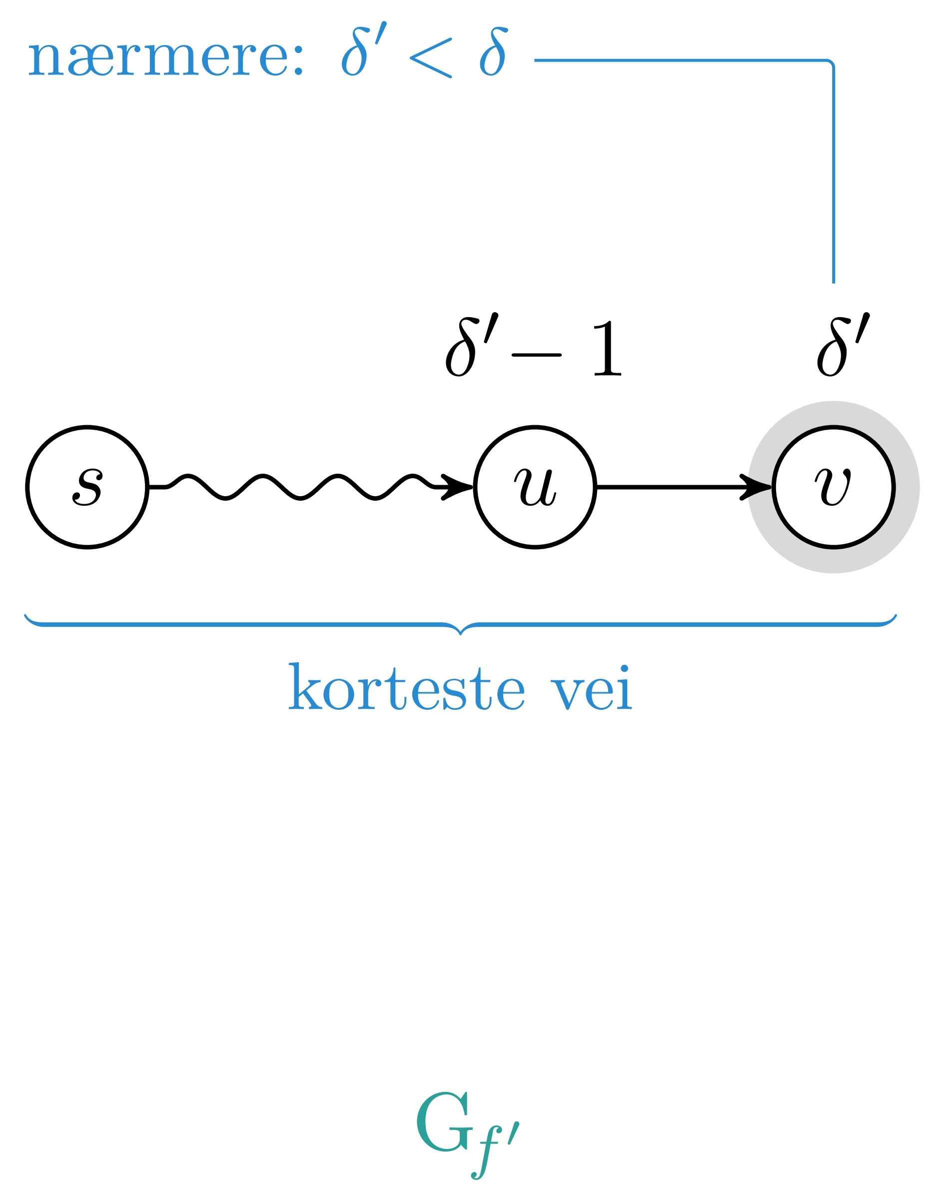


Kan noen noder ha kommet nærmere s?

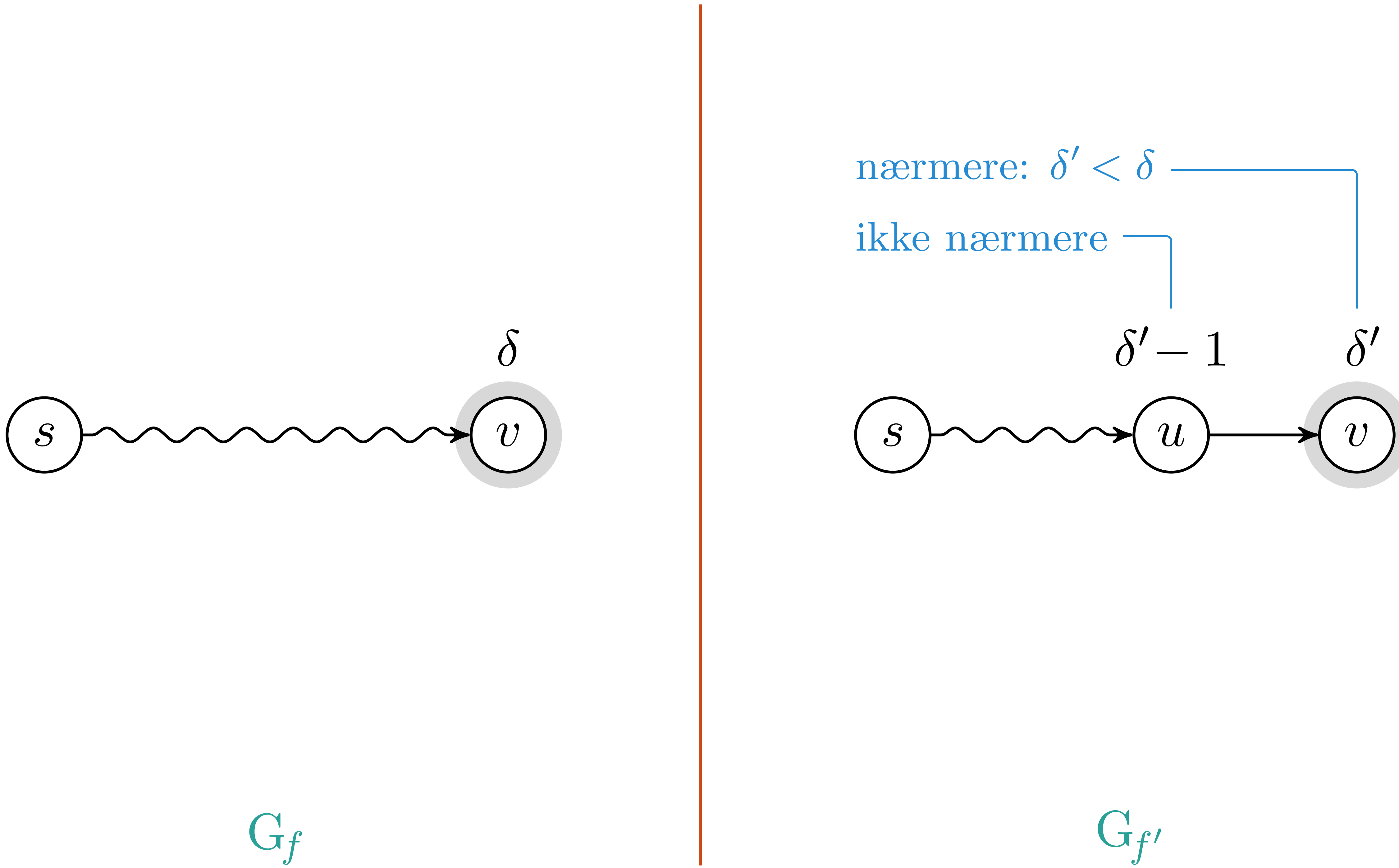


La v være den som kom nærmere og havnet nærmest

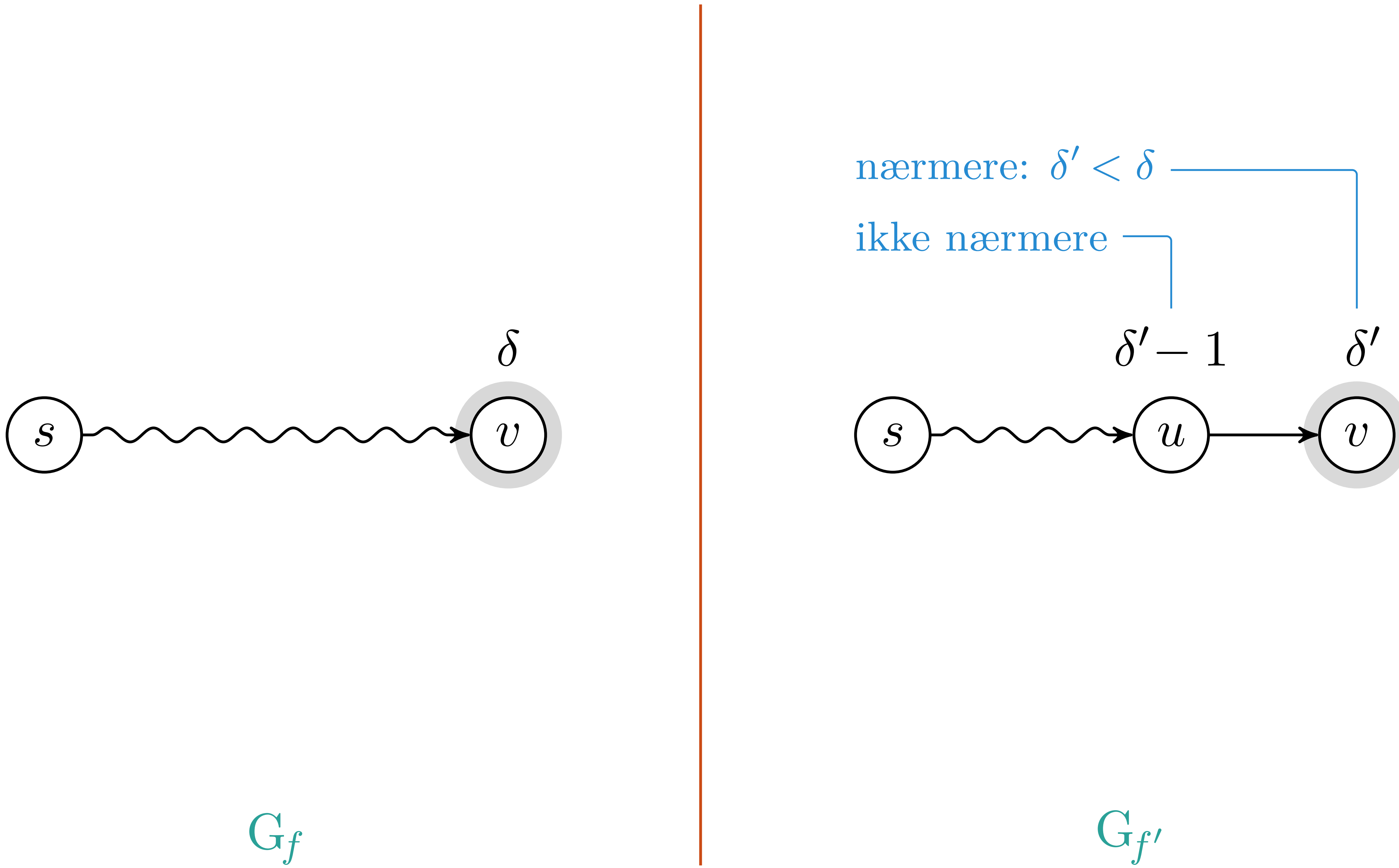




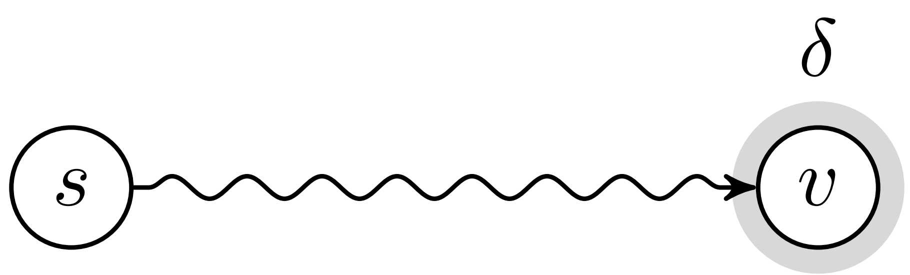
La u være forrige node langs korteste vei etterpå



v var den nærmeste som kom nærmere, så u kom ikke nærmere

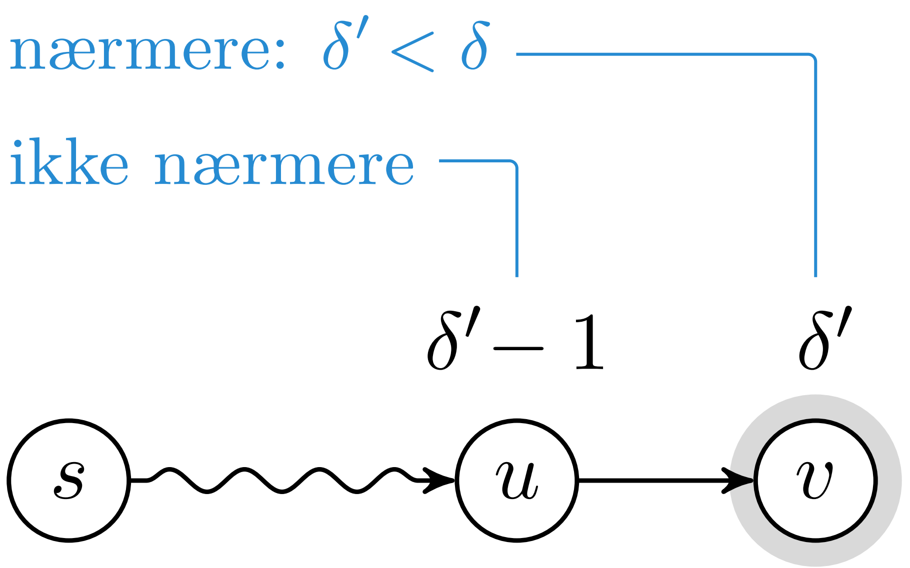


Om vi hadde (u, v) fra før, måtte u også ha kommet nærmere



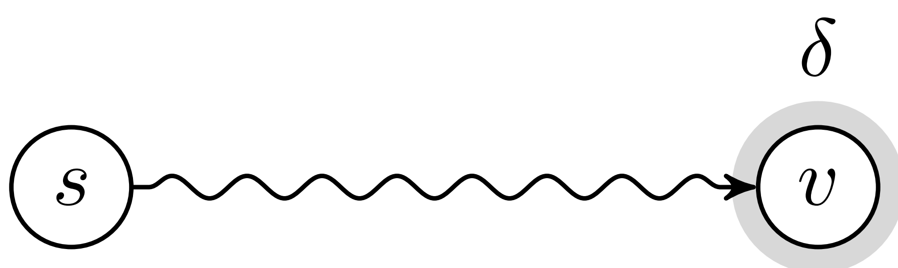
(u, v) finnes ikke

G_f



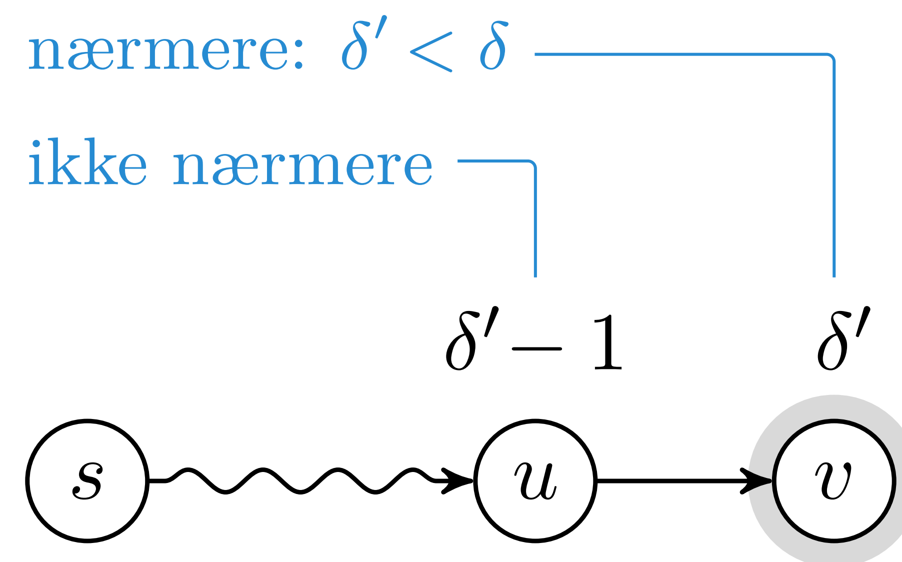
$\mathrm{G}_{f'}$

Om vi hadde (u, v) fra før, måtte u også ha kommet nærmere



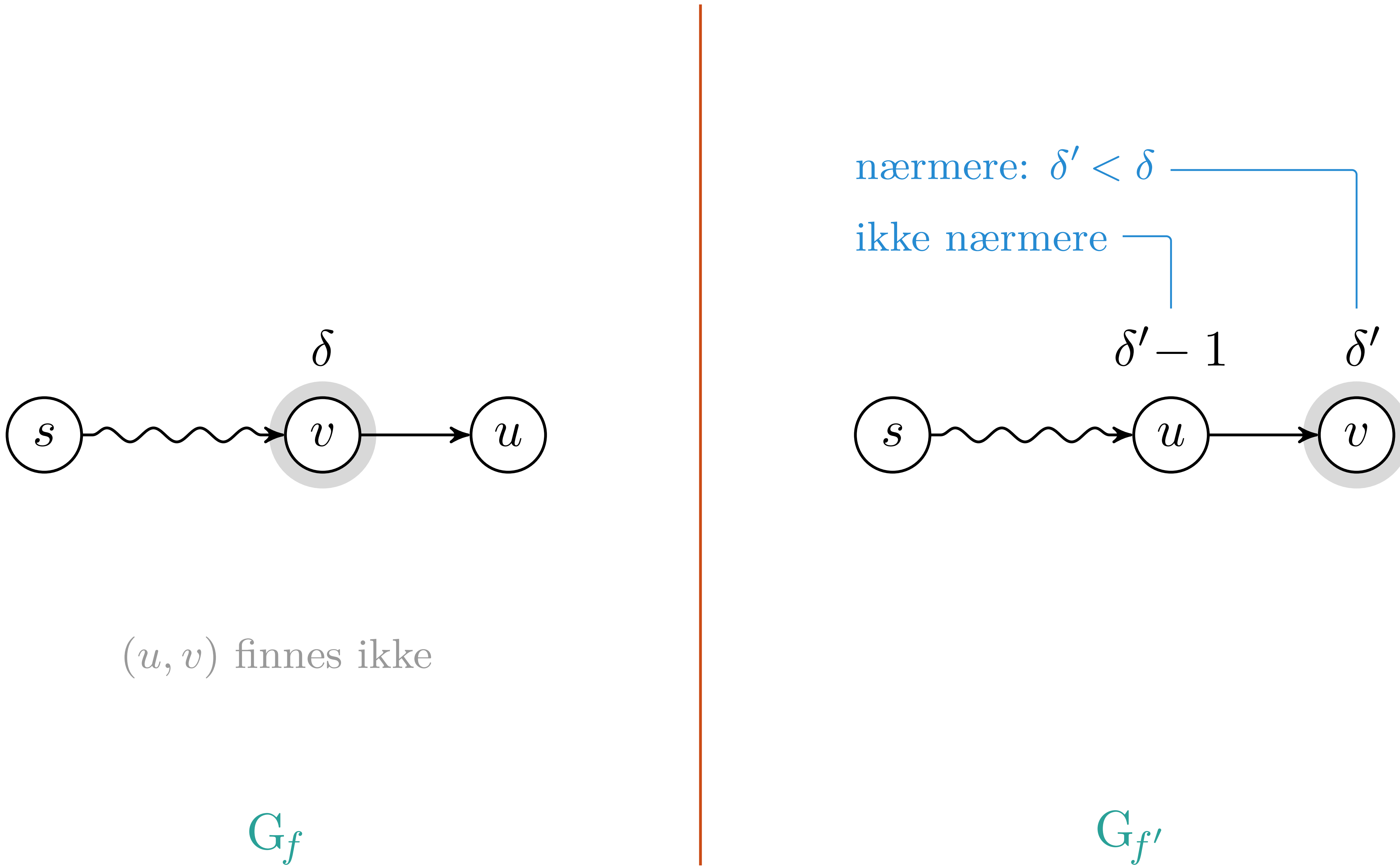
(u, v) finnes ikke

G_f

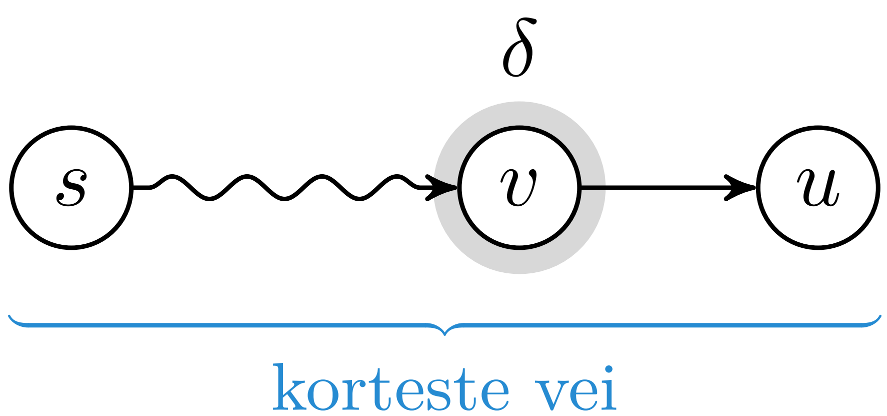


$G_{f'}$

Siden (u, v) dukket opp, må vi ha økt flyt fra v til u

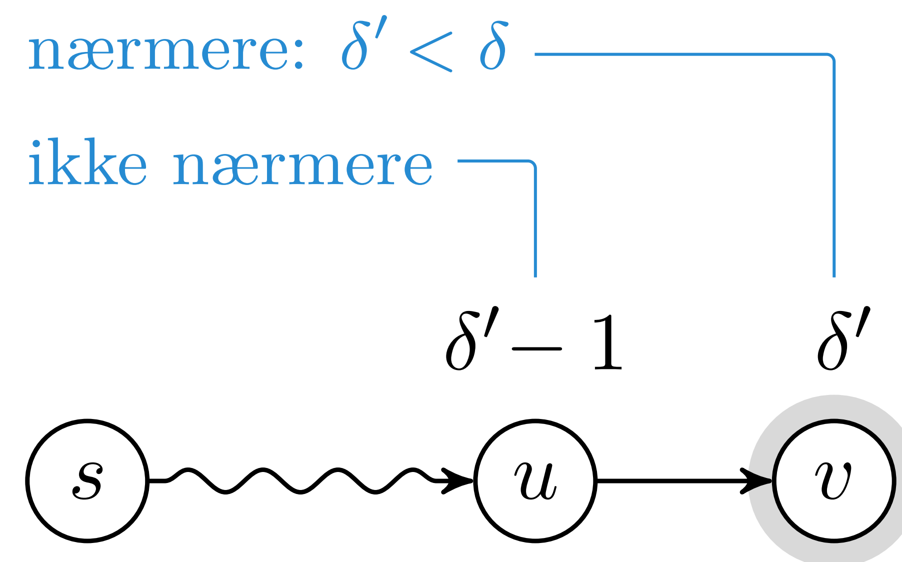


Siden (u, v) dukket opp, må vi ha økt flyt fra v til u



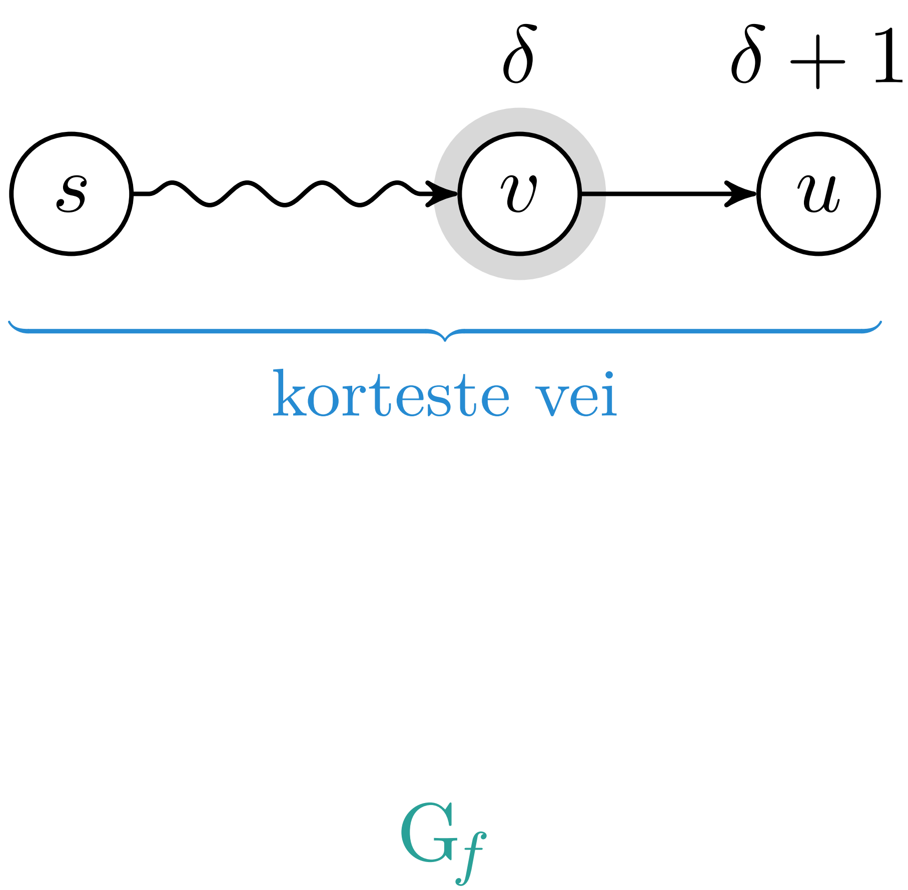
(u, v) finnes ikke

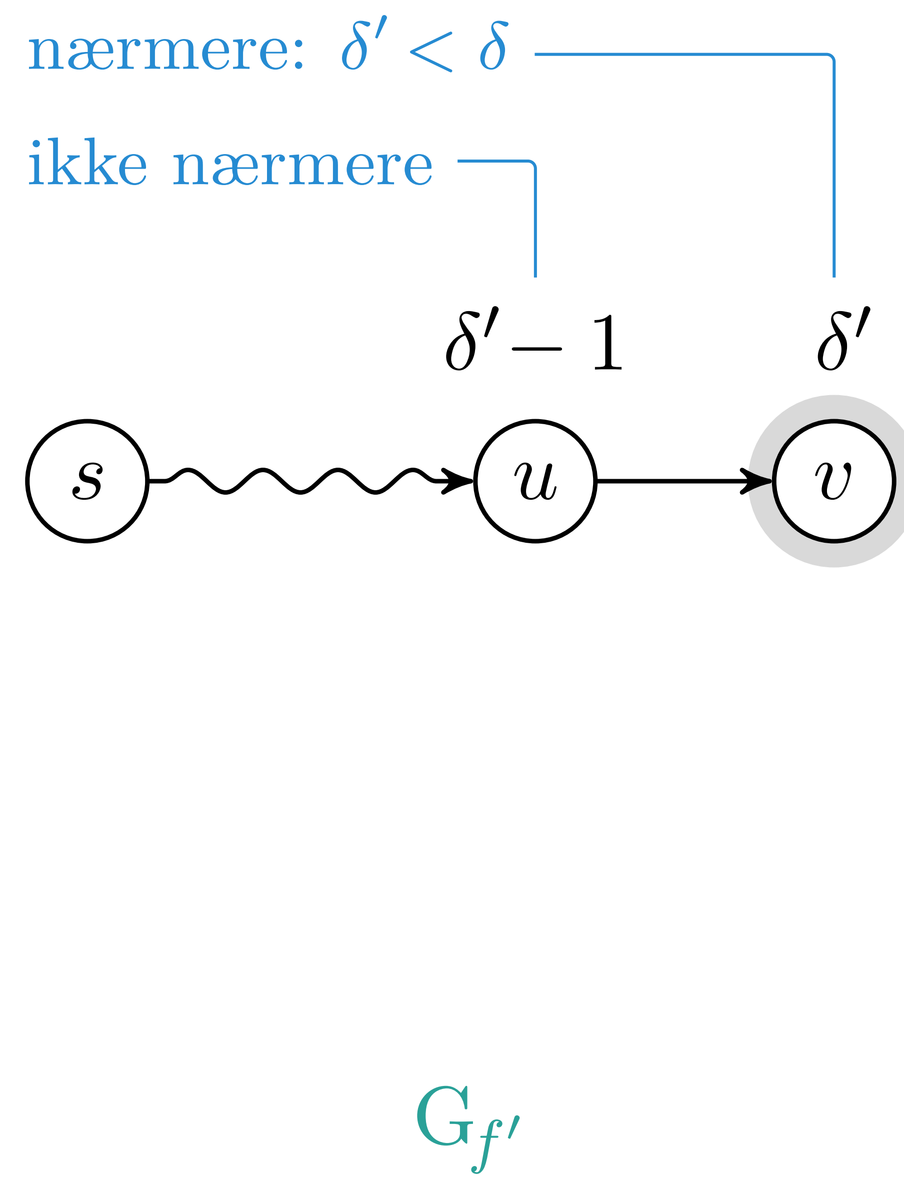
G_f



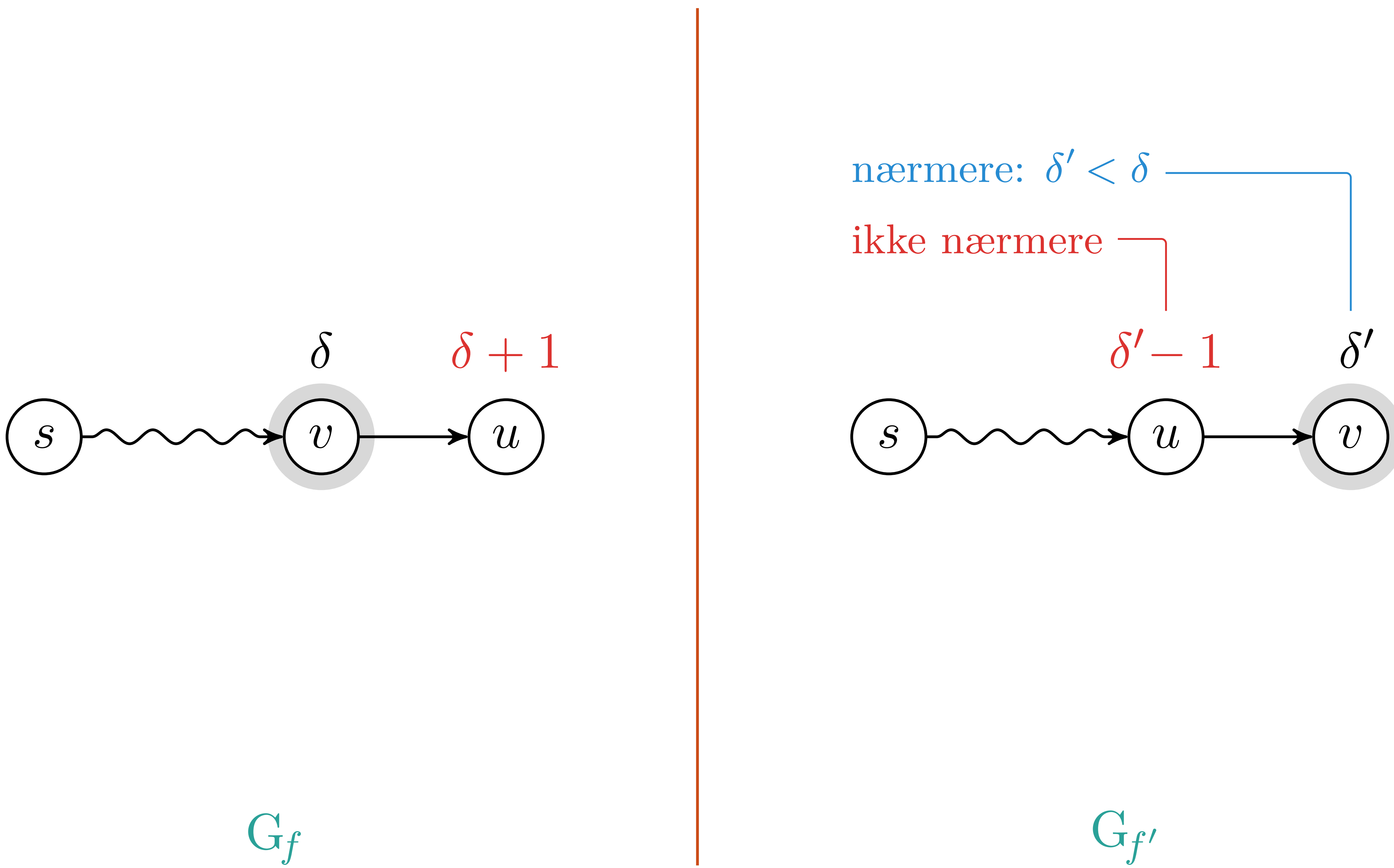
$G_{f'}$

Vi velger korteste forøkende stier, her med v før u

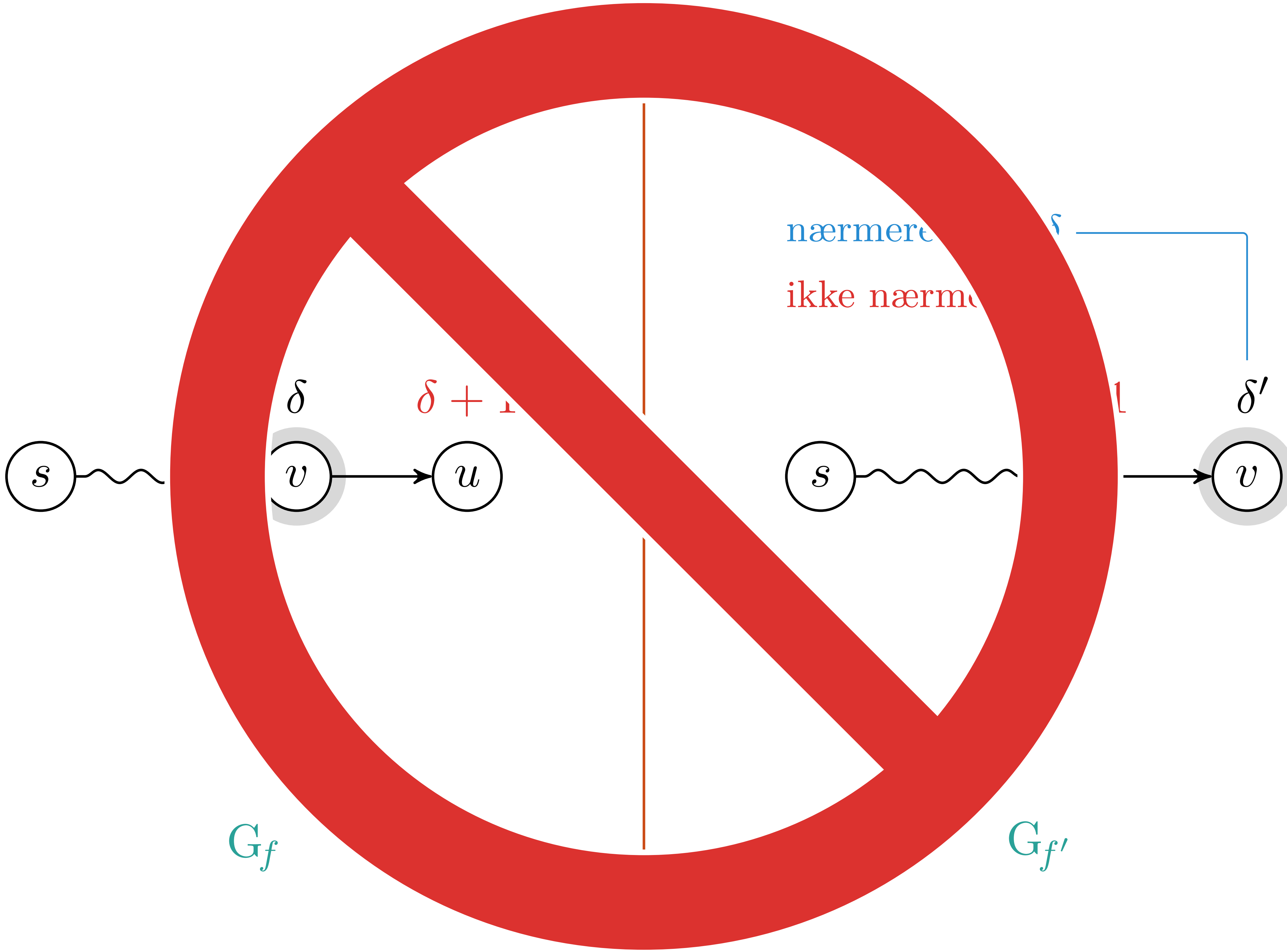




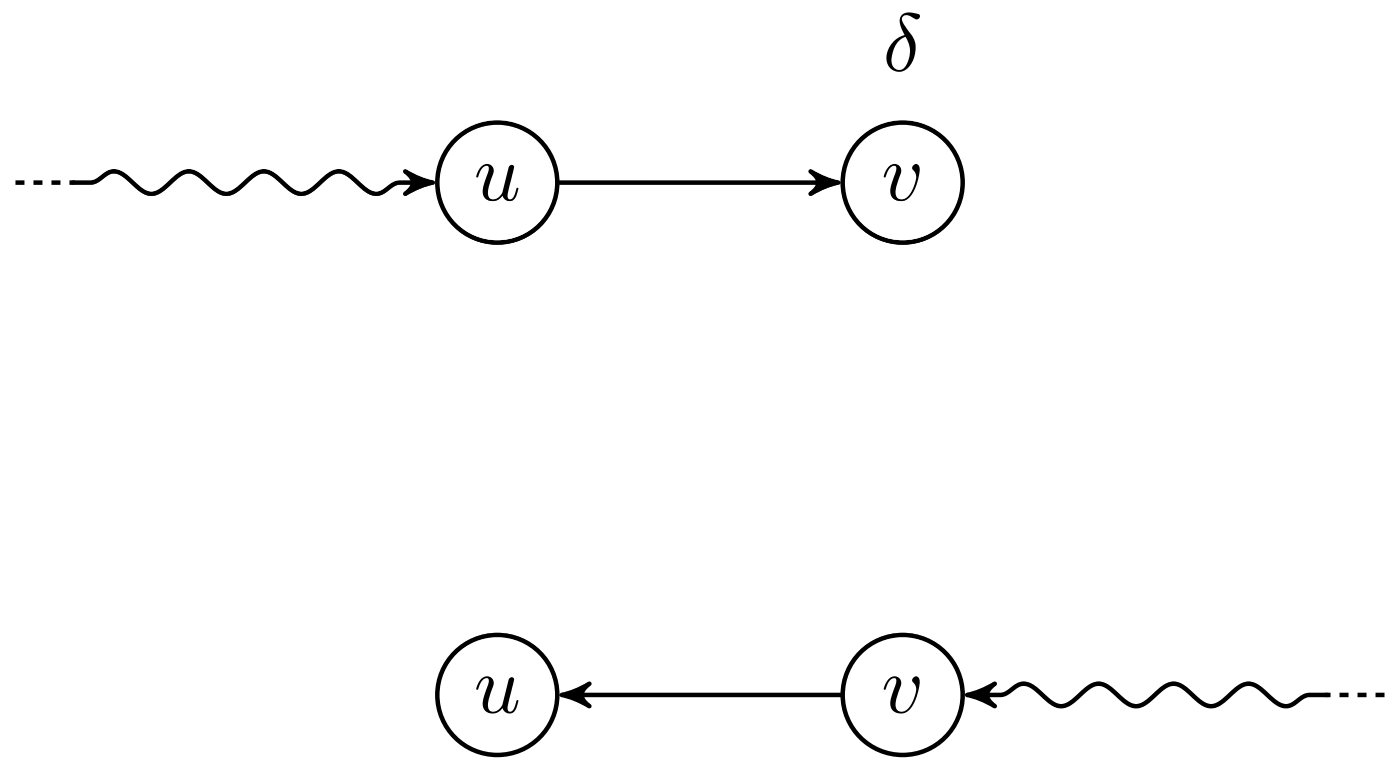
v må ha beveget seg forbi u i feil retning!



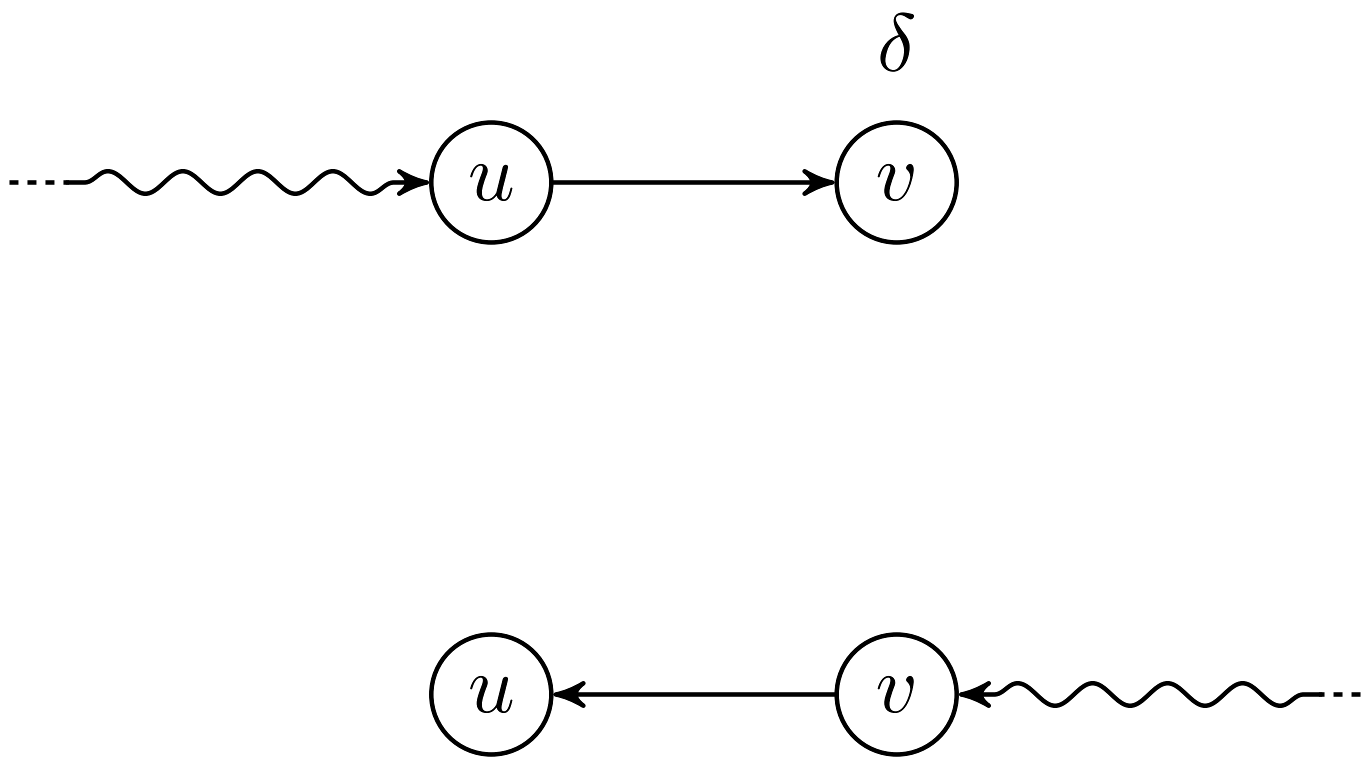
$$\delta < \delta + 1 \leqslant \delta' - 1 < \delta' \implies \delta < \delta'\text{; vi antok } \delta' < \delta!$$


nærmere
ikke nærm
δ
$\delta + 1$
δ'
s
v
u
s
v
G_f
$G_{f'}$

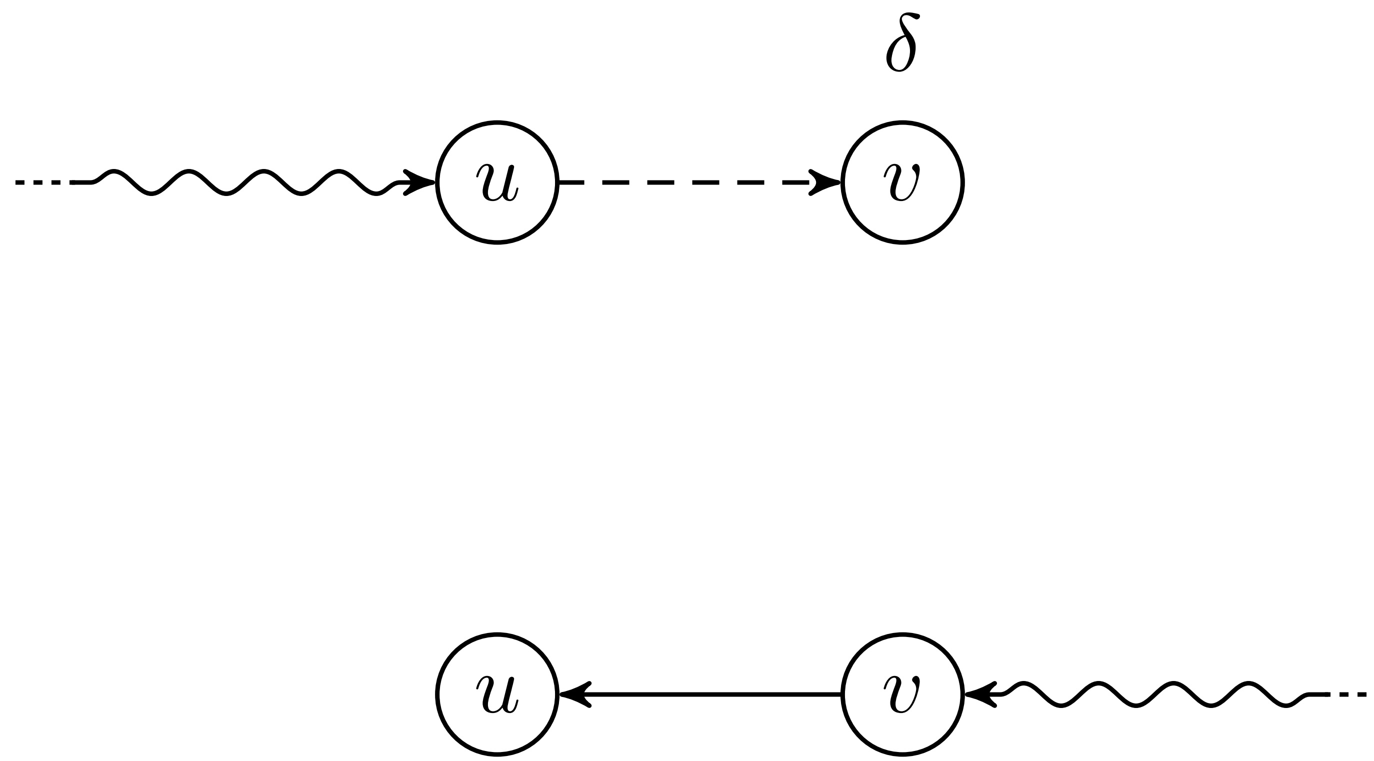
Begrenset gjenbruk av flaskehalser



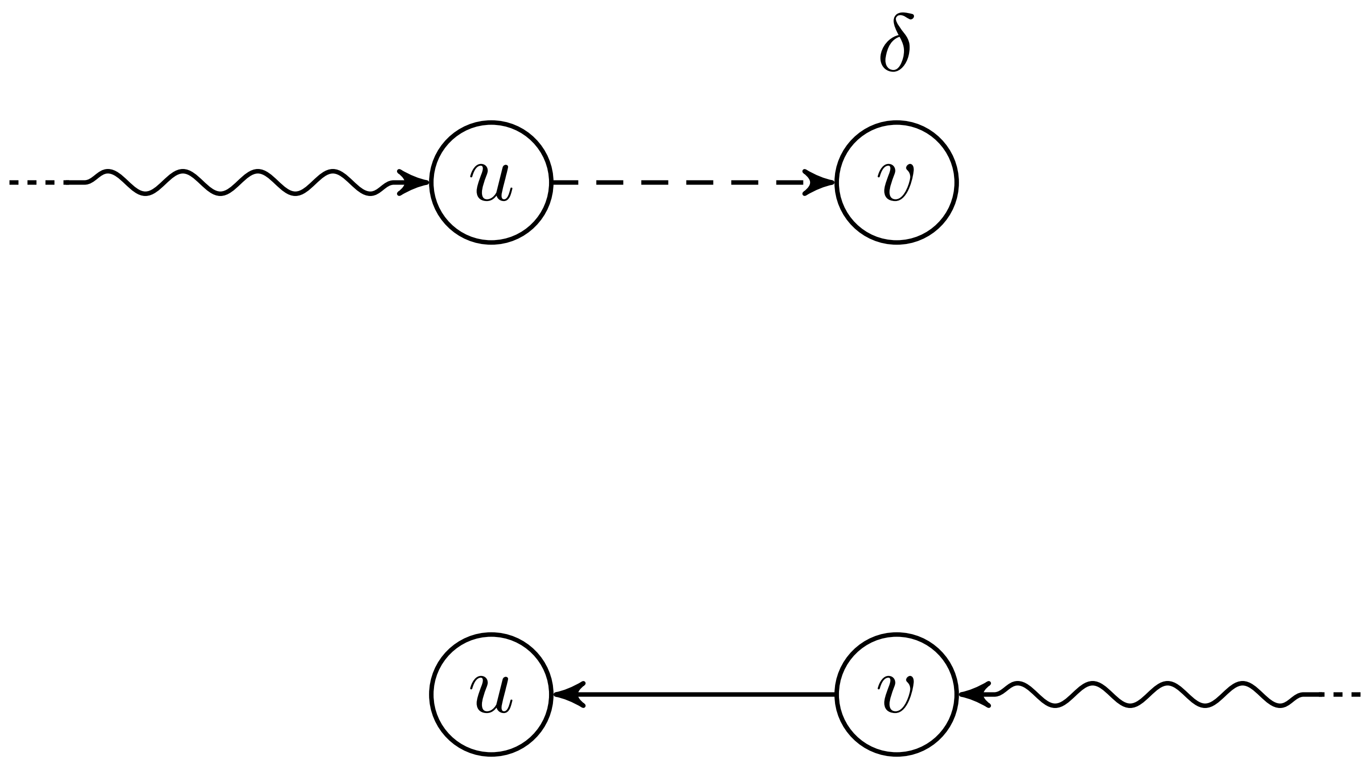
La (u, v) være flaskehals i en flytøkning



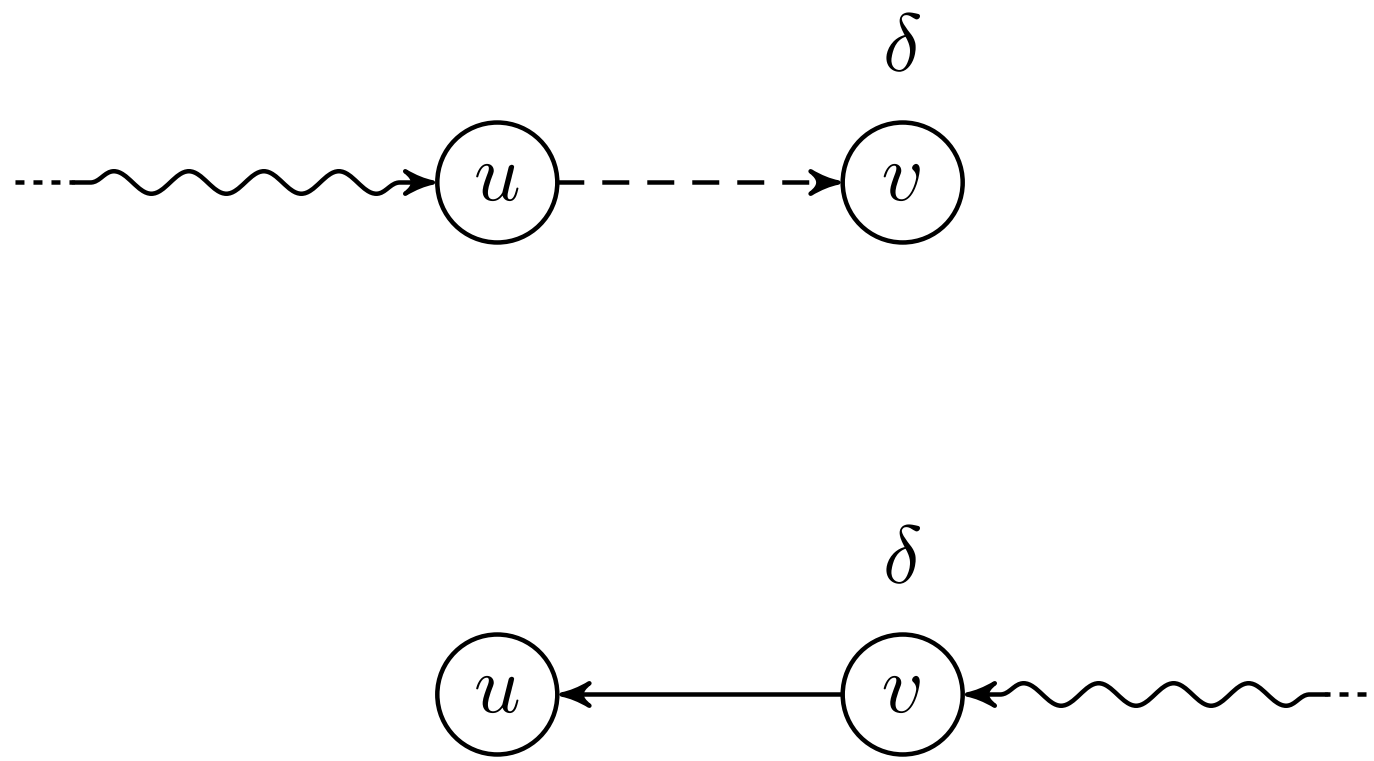
Etter økningen forsvinner (u, v) fra G_f



Etter økningen forsvinner (u, v) fra G_f

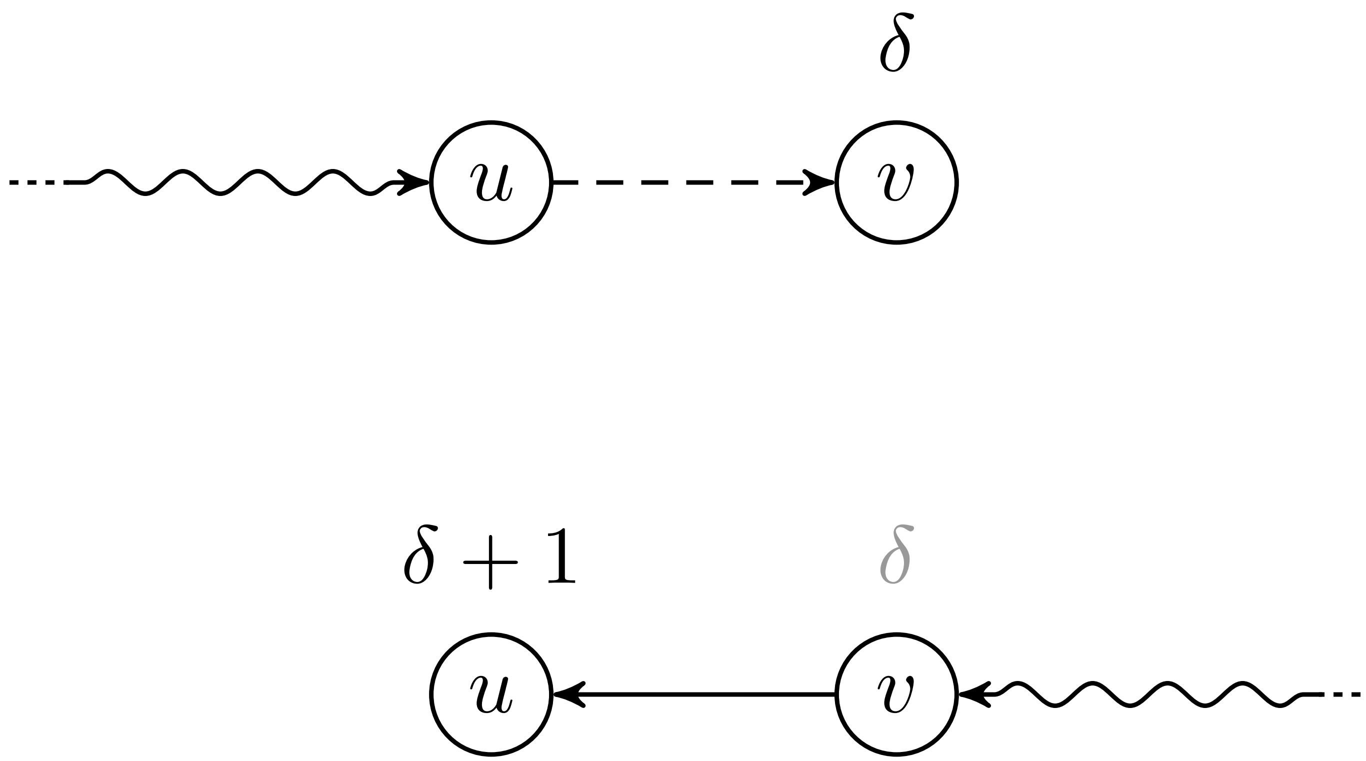


For å gjeninnføre (u, v) må vi øke flyt fra v til u

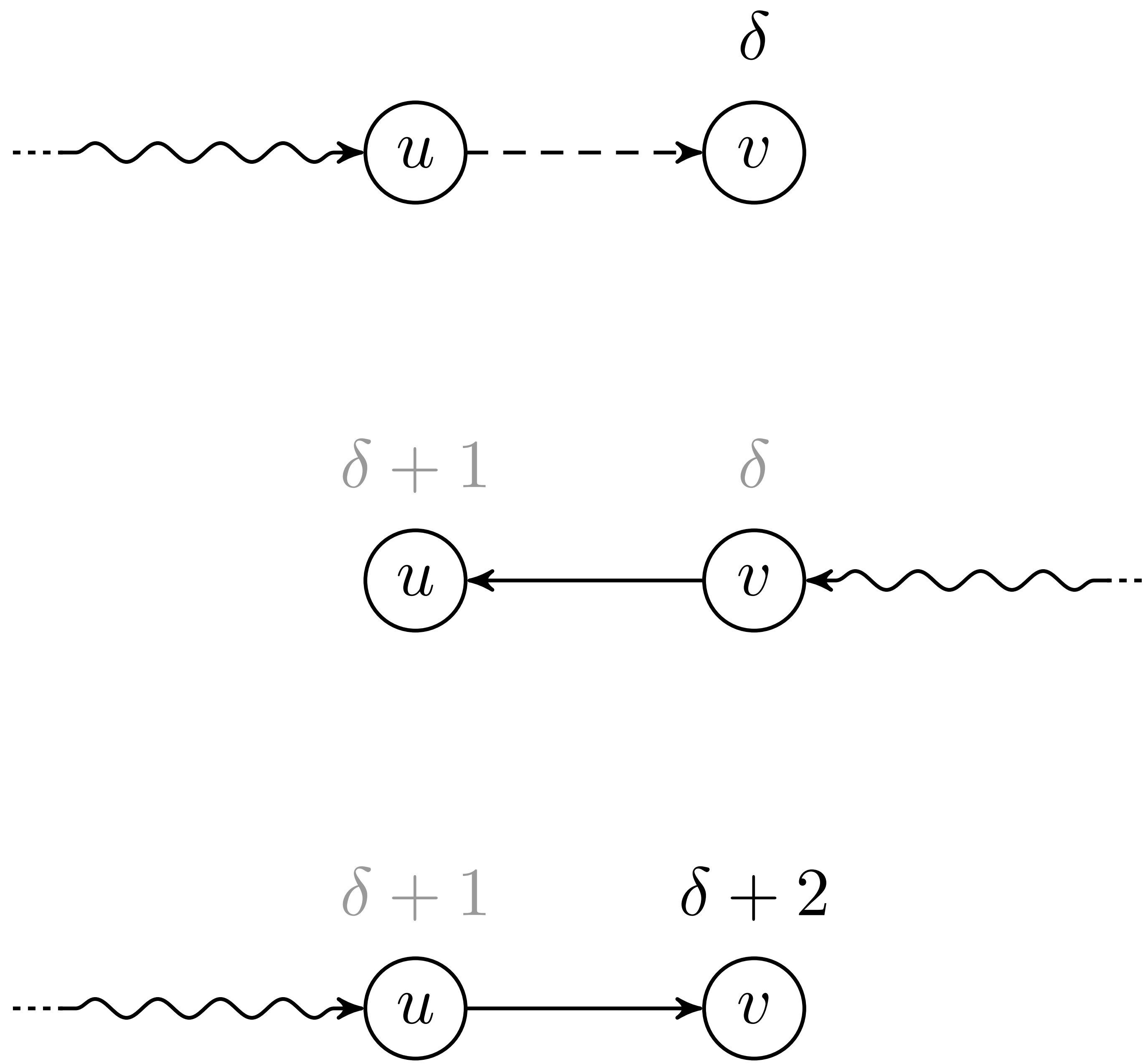


Har kan altså avstandene være enda større; dette er bare minimum.

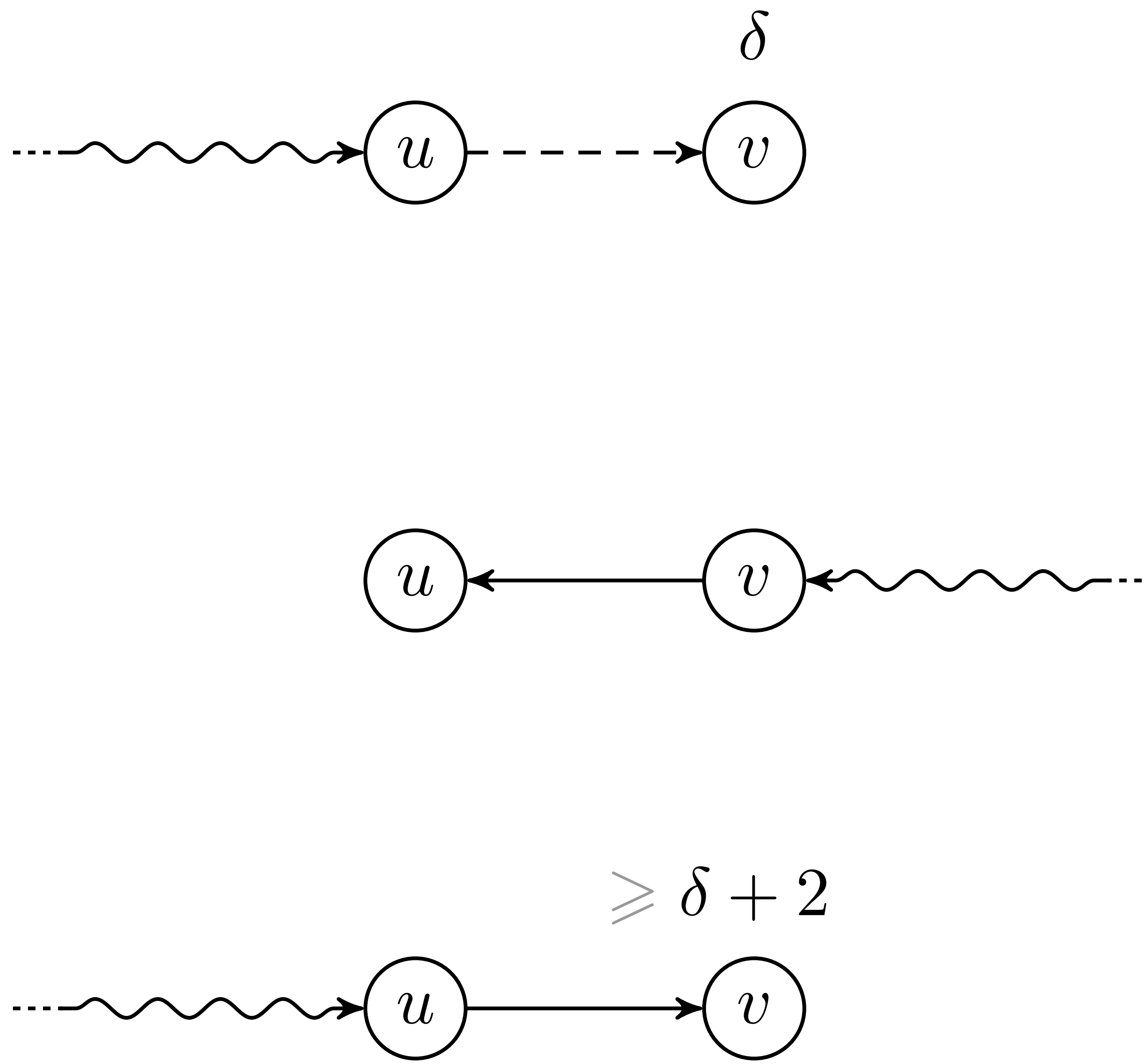
Husk: Avstander synker ikke. (De kan naturligvis øke)



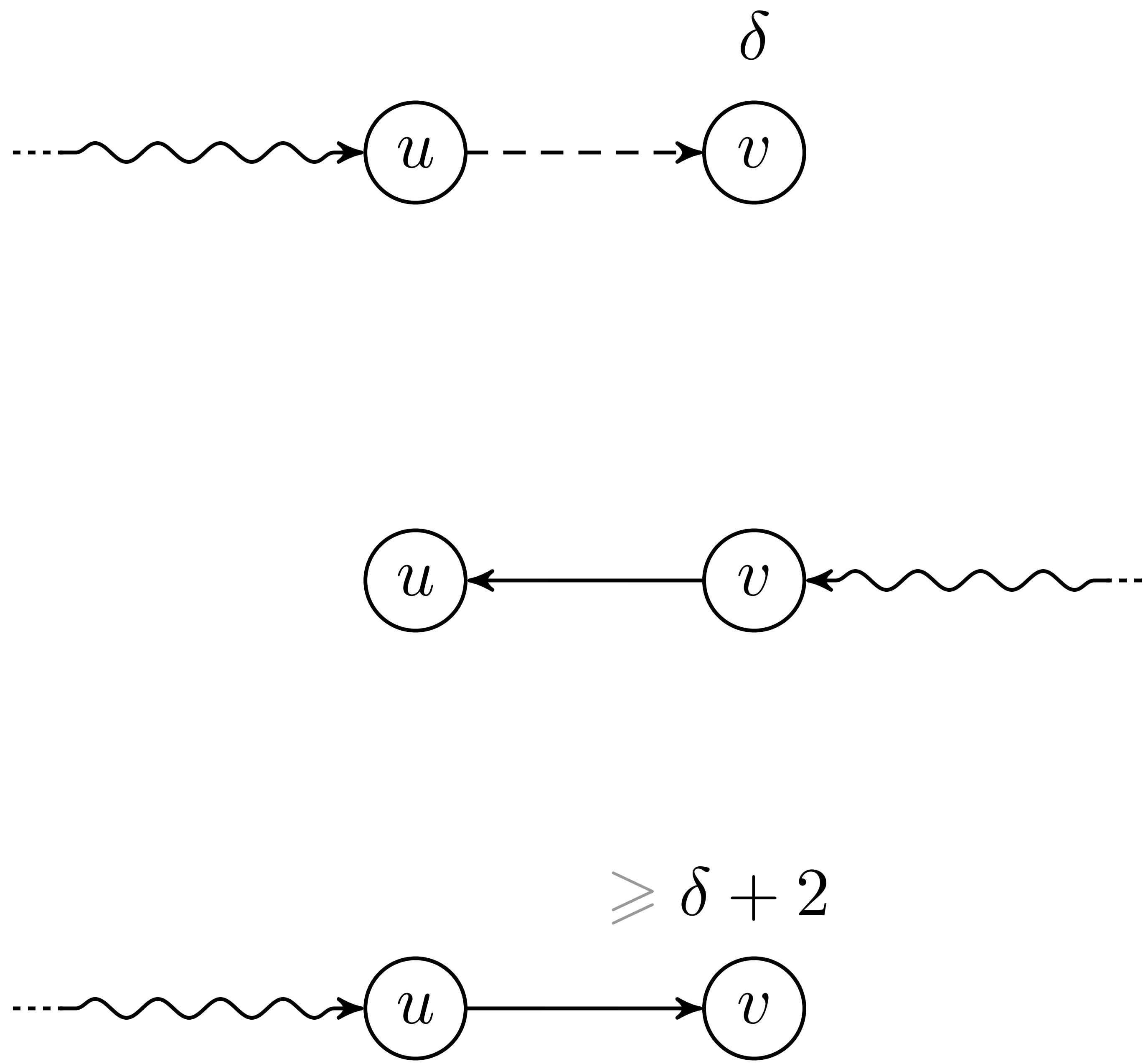
Vi øker flyt langs korteste vei



Så (u, v) er flaskehals maks annenhver gang . . .



…og $\delta(s, v)$ øker da med minst 2



O(E) nodepar er flaskehals O(V) ganger: Kjøretid O(VE)

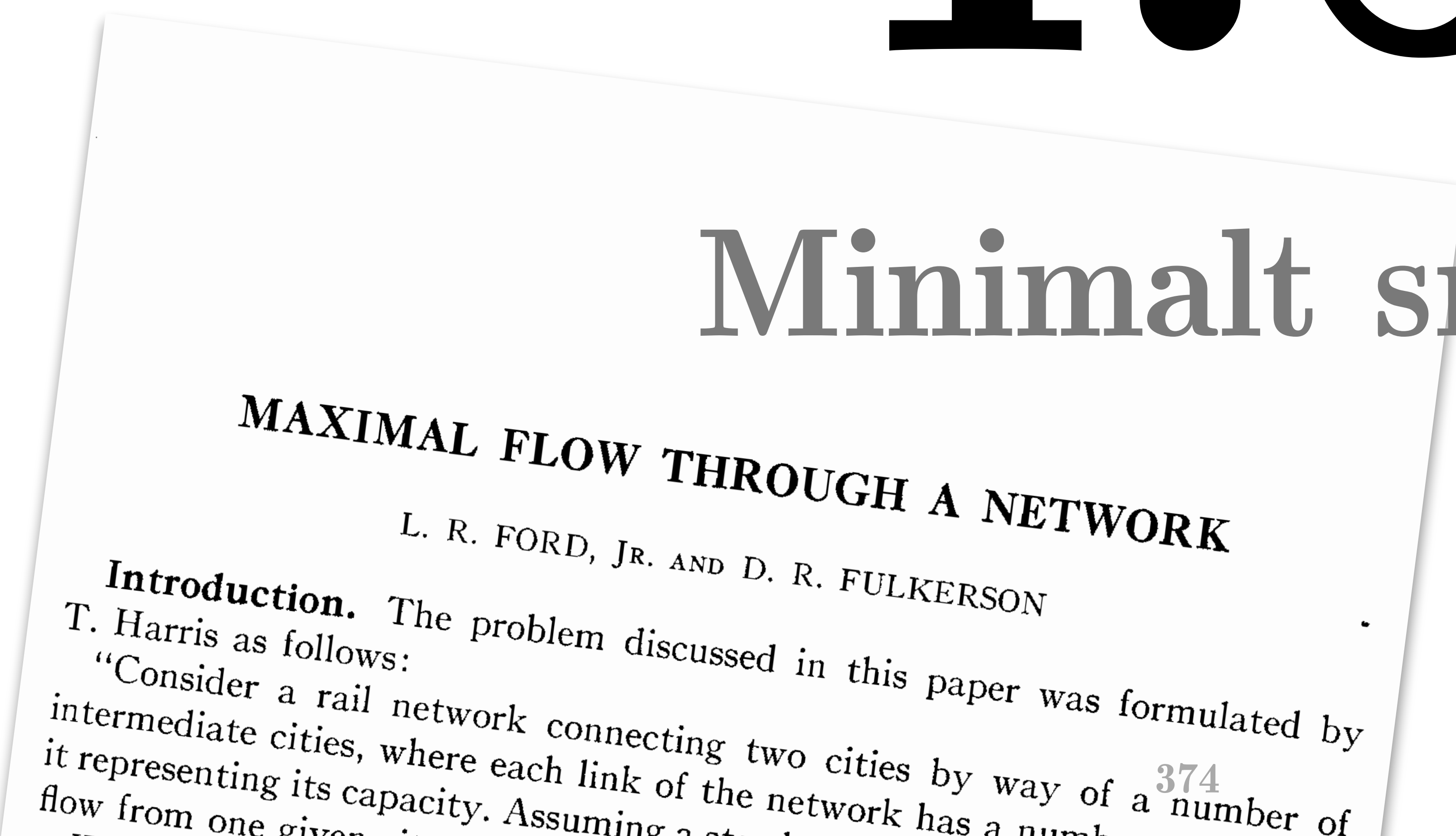
Korrekthet, da?

Vi ser på det såkalt duale problemet, der vi forsøker å finne flaskehalsen i nettet – det minimale snittet. Det vil hjelpe oss med å vise korrekthet.

Min. snitt og maks. flyt er såkalt duale problemer: Det ene er minimering, det andre er maksimering, og de har samme optimum.

4:5

Minimalt snitt



MAXIMAL FLOW THROUGH A NETWORK

L. R. FORD, JR. AND D. R. FULKERSON

Introduction. The problem discussed in this paper was formulated by T. Harris as follows:

"Consider a rail network connecting two cities by way of a number of intermediate cities, where each link of the network has a number
it representing its capacity. Assuming a st
flow from one given

Om du er nysgjerrig på dualitet generelt, så finner du mer om det i delkapittel 29.4.

MAXIMAL FLOW THROUGH A NETWORK

L. R. FORD, JR. AND D. R. FULKERSON

Introduction. The problem discussed in this paper was formulated by T. Harris as follows:

"Consider a rail network connecting two cities by way of a number of intermediate cities, where each link of the network has a number assigned to it representing its capacity. Assuming a steady state condition, find a maximal flow from one given city to the other."

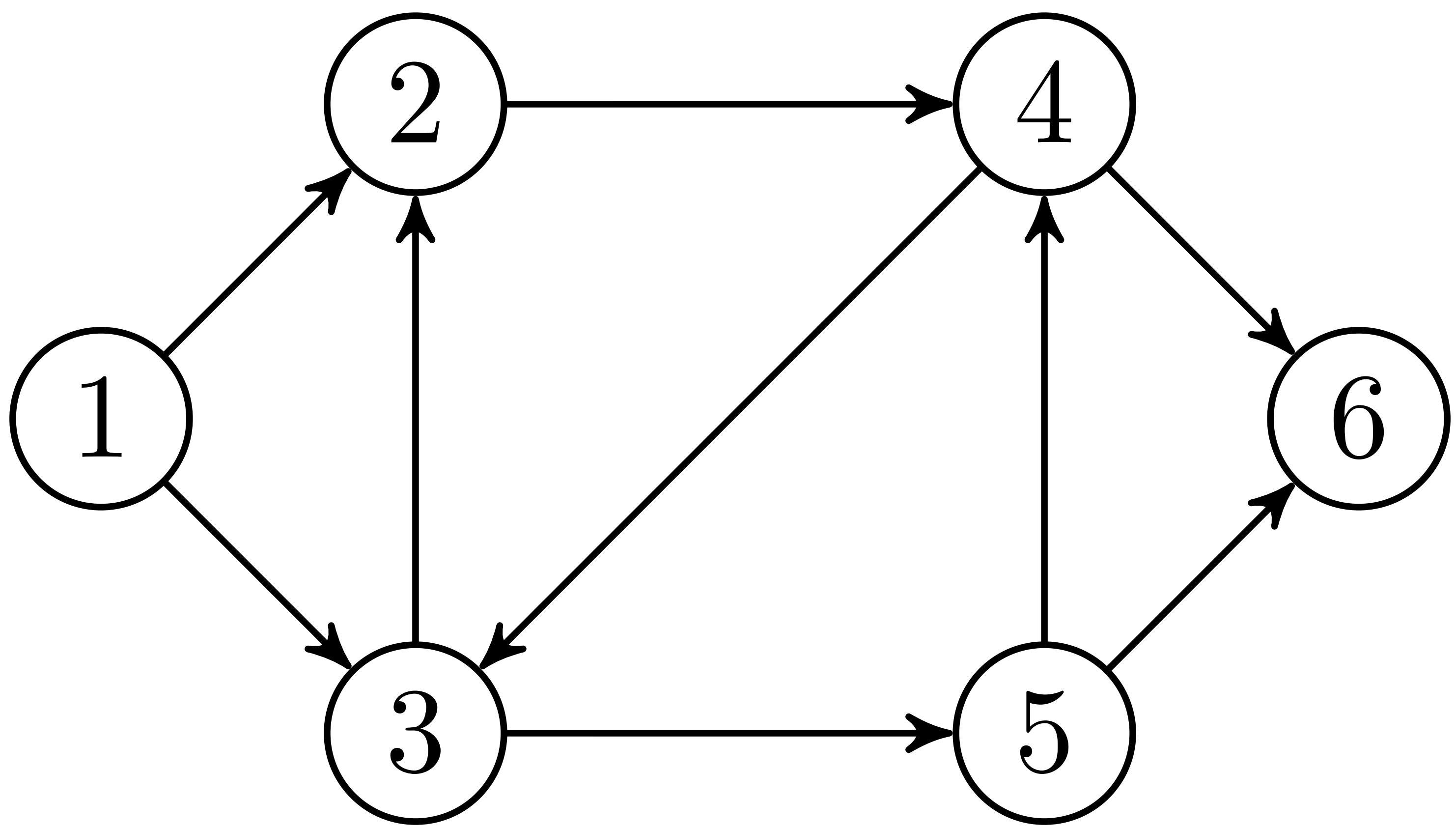
While this can be set up as a linear programming problem with as many equations as there are cities in the network, and hence can be solved by the simplex method **(1)**, it turns out that in the cases of most practical interest, where the network is planar in a certain restricted sense, a much simpler and more efficient hand computing procedure can be described.

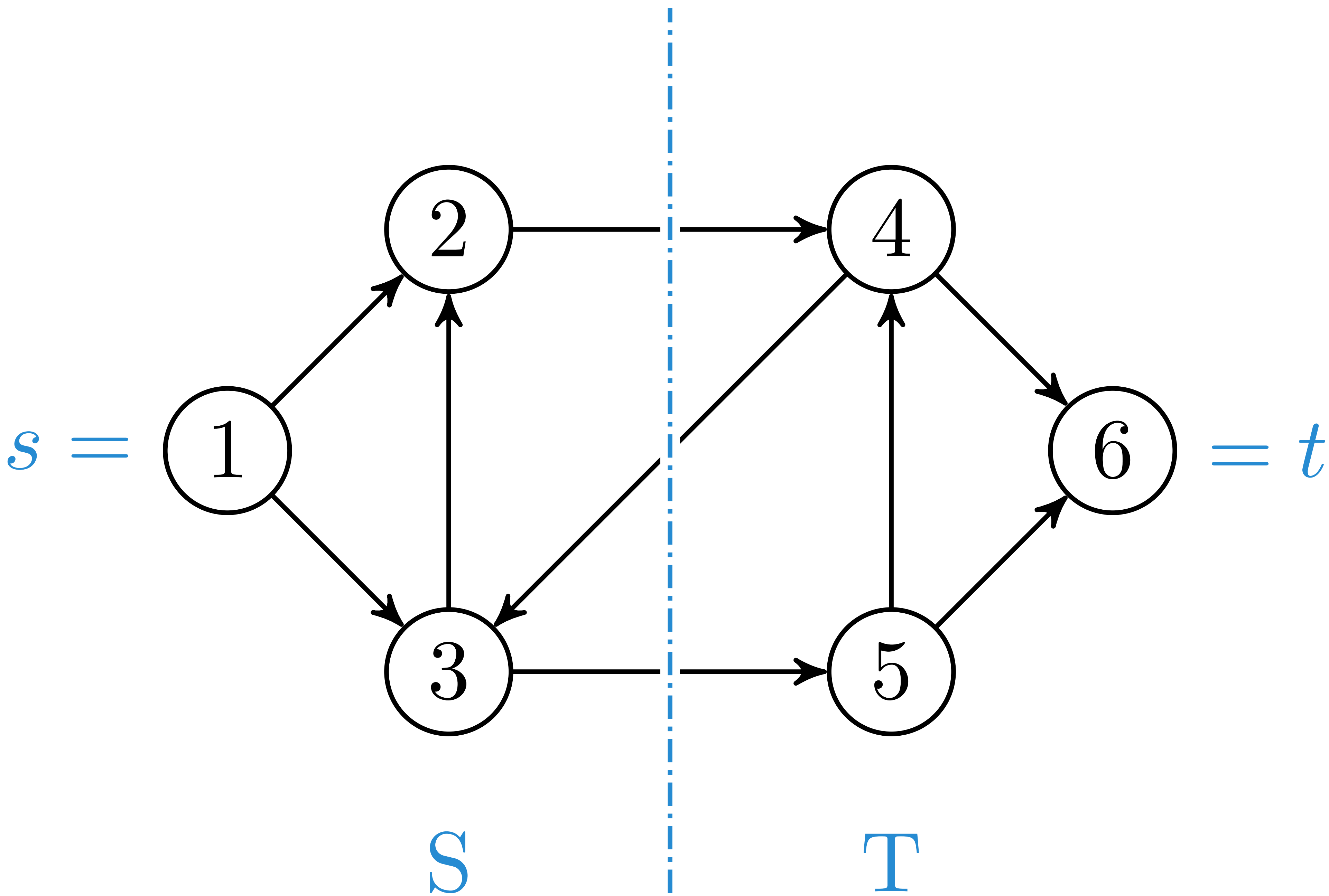
In §1 we prove the minimal cut theorem, which establishes that an obvious upper bound for flows over an arbitrary network can always be achieved. The proof is non-constructive. However, by specializing the network (§2), we obtain as a consequence of the minimal cut theorem an effective computational scheme. Finally, we observe in §3 the duality between the capacity problem and that of finding the shortest path, via a network, between two given points.

1. The minimal cut theorem. A *graph* G is a finite, 1-dimensional complex, composed of *vertices* $a, b, c, \ldots, e$, and *arcs* $\alpha(ab), \beta(ac), \ldots, \delta(ce)$. An arc $\alpha(ab)$ *joins* its end vertices a, b; it passes through no other vertices of G and intersects other arcs only in vertices. A *chain* is a set of distinct arcs of G which can be arranged as $\alpha(ab), \beta(bc), \gamma(cd), \ldots, \delta(gh)$, where the vertices $a, b, c, \ldots, h$ are distinct, i.e., a chain does not intersect itself; a chain *joins* its end vertices a and h.

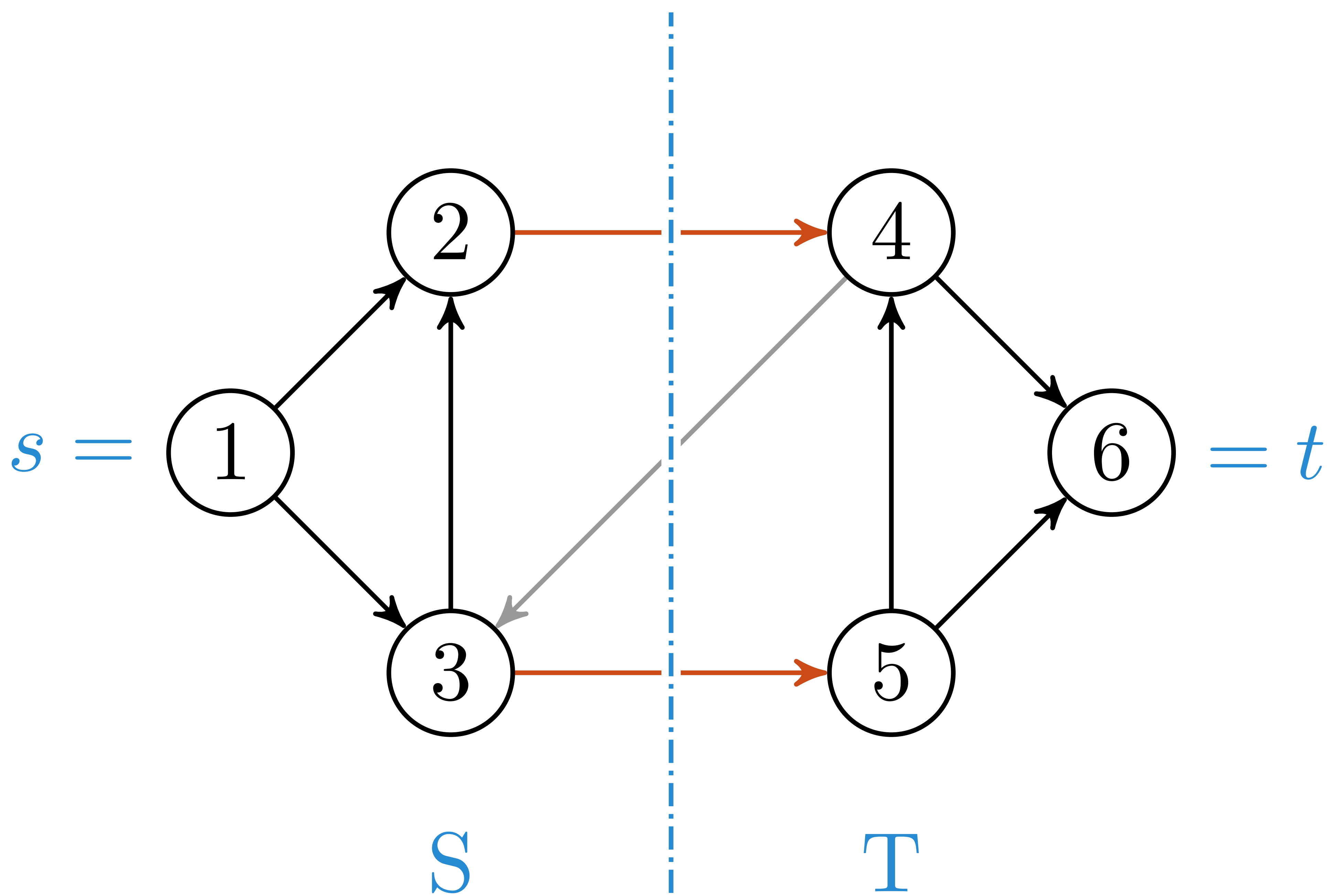
We distinguish two vertices of G: a, the *source*, and b, the *sink*.[1] A *chain flow* from a to b is a couple $(C; k)$ composed of a chain C joining a and b, and a non-negative number k representing the flow along C from source to sink.

Each arc in G has associated with it a positive number called its *capacity*. We call the graph G, together with the capacities of its individual arcs, a


1
2
3
4
5
6



$s =$
1
2
3
4
5
6
$= t$
S
T



Kapasitet gitt av kanter S → T

Snitt i flytnett: Partisjon (S, T) av V

- $s \in S$ og $t \in T$
- Netto-flyt:

$$f(S,T) = \sum_{u \in S} \sum_{v \in T} f(u,v) - \sum_{u \in S} \sum_{v \in T} f(v,u)$$

- Kapasitet:

$$c(S,T) = \sum_{u \in S} \sum_{v \in T} c(u,v)$$

Lemma 26.5: $f(S,T) = |f|$

- Korollar 26.5: $|f| \leqslant c(S,T)$

Beviset (s.722) krever en del utregning, men er ganske rett frem

Input: Et flytnett $G = (V, E)$ med kilde s og sluk t.

Output: Et snitt (S, T) med minst mulig kapasitet, dvs., der $c(S, T)$ er minimal.

For anvendelser vil c her ofte være en form for *kostnad*

Maks. flyt = min. snitt

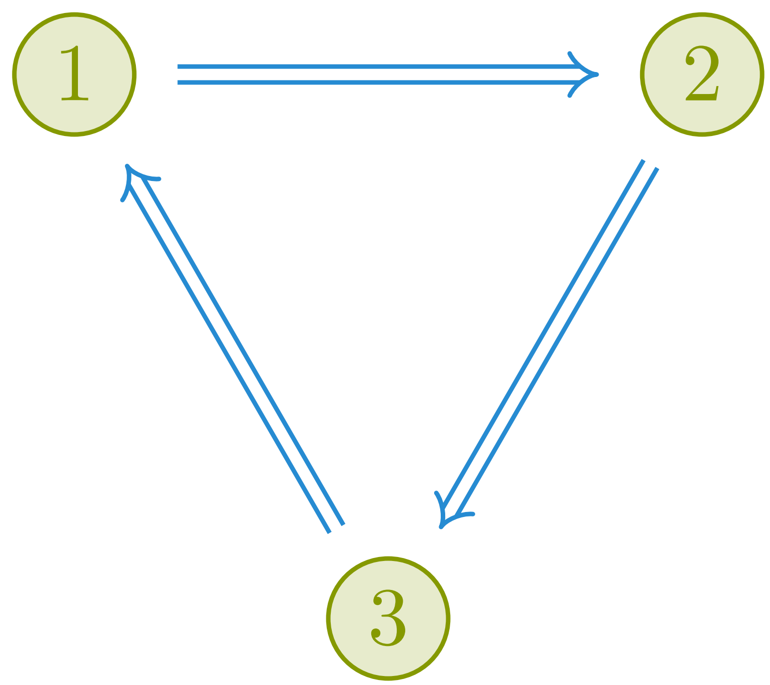
f er maks-flyt for G

G$_f$ har ingen forøkende sti

$$① \iff ②$$

f er maks-flyt for G

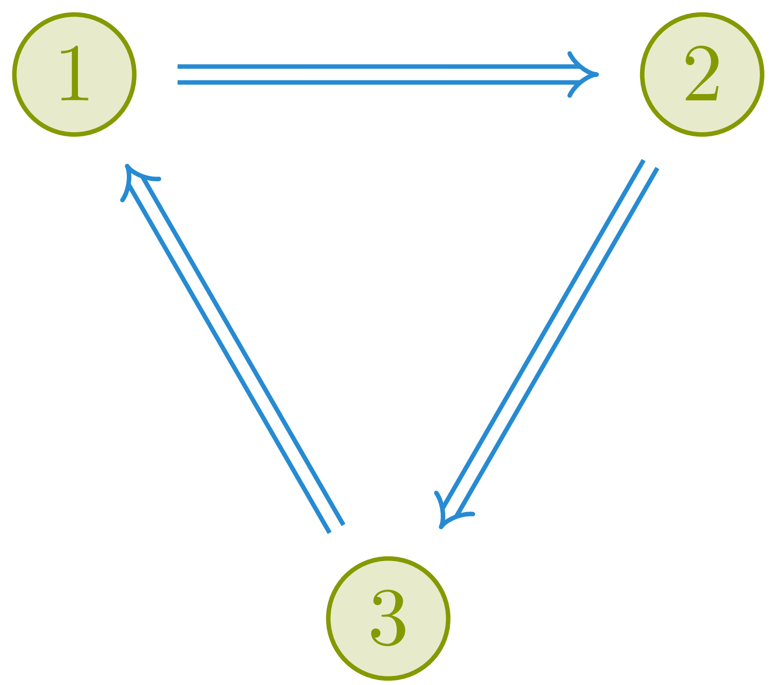
G_f har ingen forøkende sti



$|f| = c(\mathrm{S}, \mathrm{T})$ for et snitt (S, T)

f er maks-flyt for G

G_f har ingen forøkende sti



$|f| = c(\mathrm{S}, \mathrm{T})$ for et snitt (S, T)

Eksempel på såkalt *dualitet*

f er maks-flyt for G

G_f har ingen forøkende sti

1 $\Longrightarrow$ 2

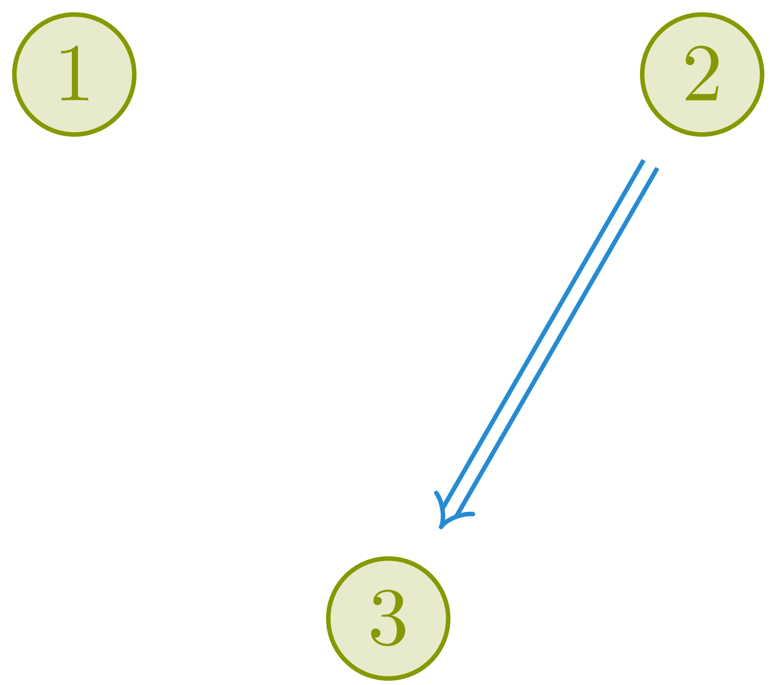
3

$|f| = c(\mathrm{S}, \mathrm{T})$ for et snitt (S, T)

Ved selvmotsigelse: En slik sti ville kunne øke f

f er maks-flyt for G

G_f har ingen forøkende sti

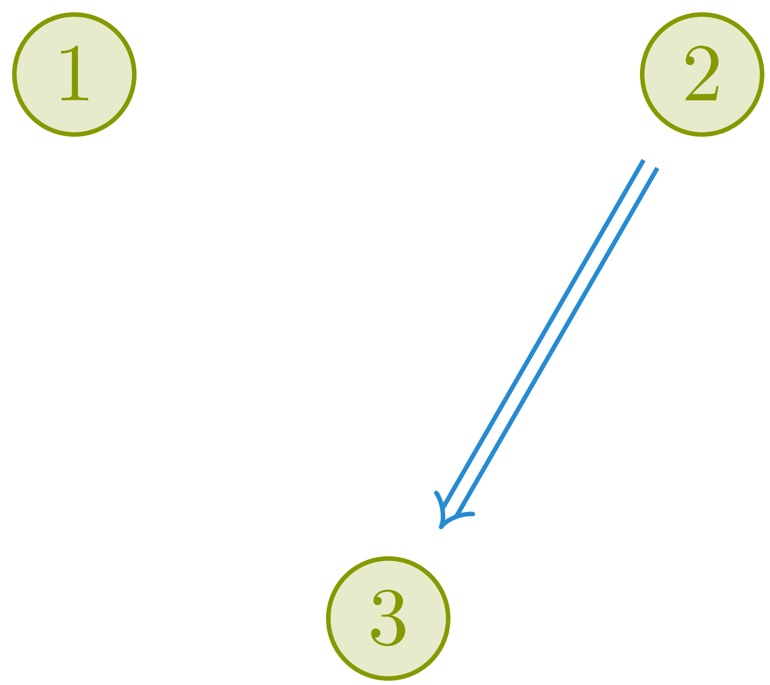


$|f| = c(S, T)$ for et snitt (S, T)

La S være noder som kan nås i G_f, og la $T = V - S$

f er maks-flyt for G

G_f har ingen forøkende sti

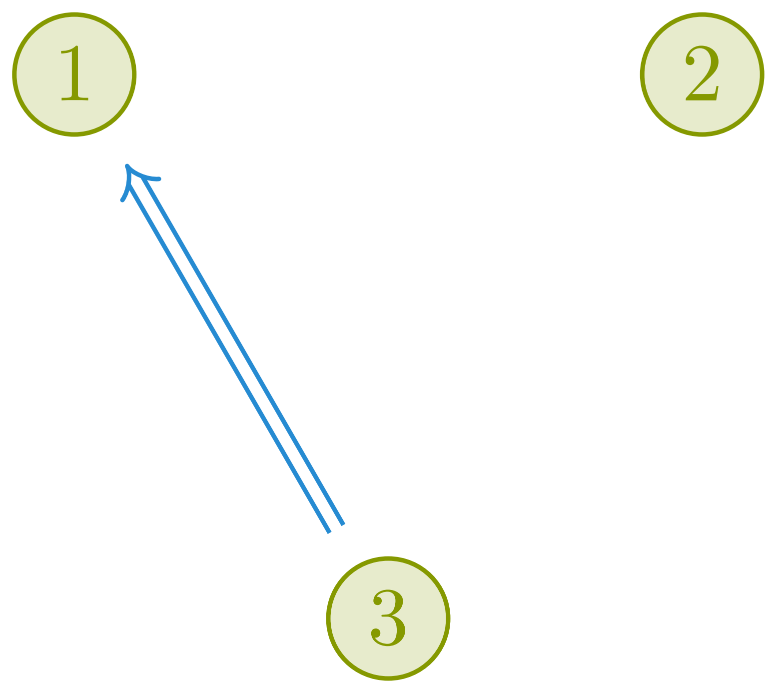


$|f| = c(S, T)$ for et snitt (S, T)

Snittet blokkerer: Kanter er fulle $(S \rightarrow T)$ eller tomme $(S \leftarrow T)$

f er maks-flyt for G

G_f har ingen forøkende sti

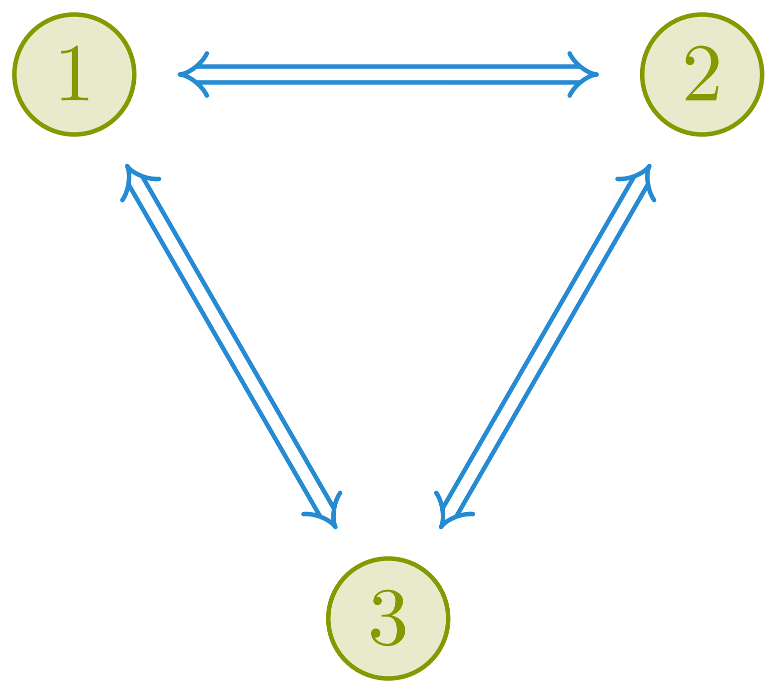


$|f| = c(S, T)$ for et snitt (S, T)

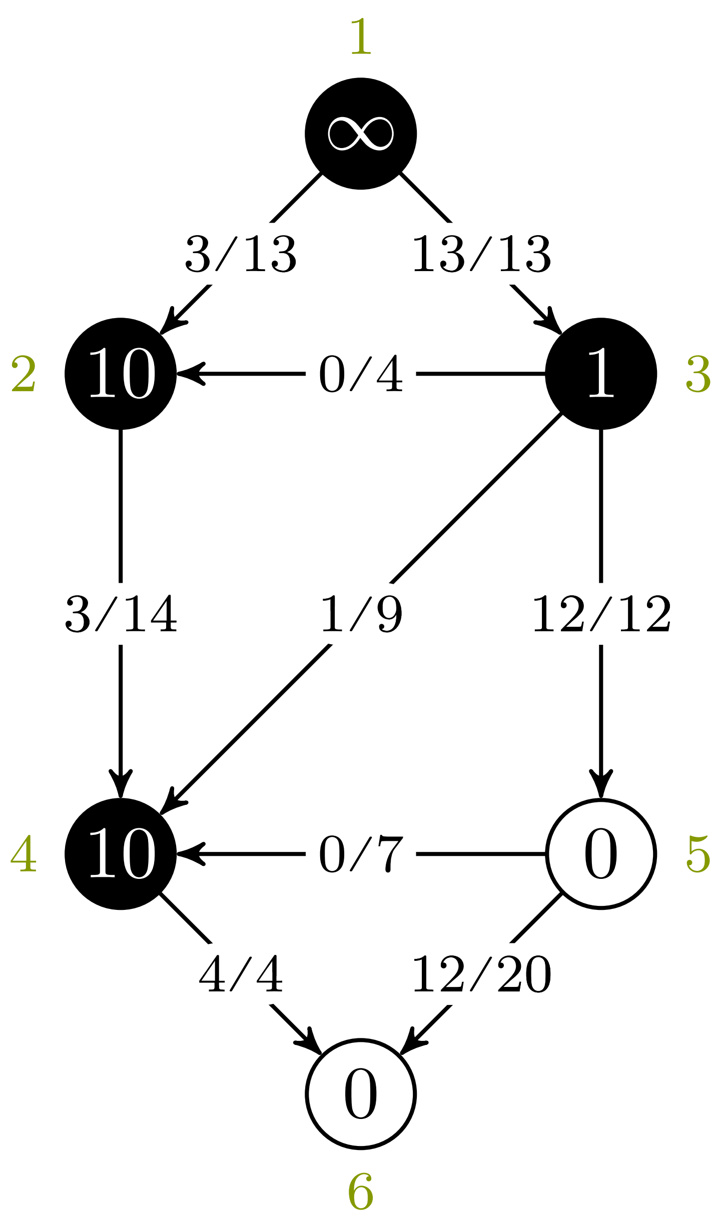
For enhver flyt har vi $|f| \leq c(S, T)$; siden $|f| = c(S, T)$ er f maksimal

f er maks-flyt for G

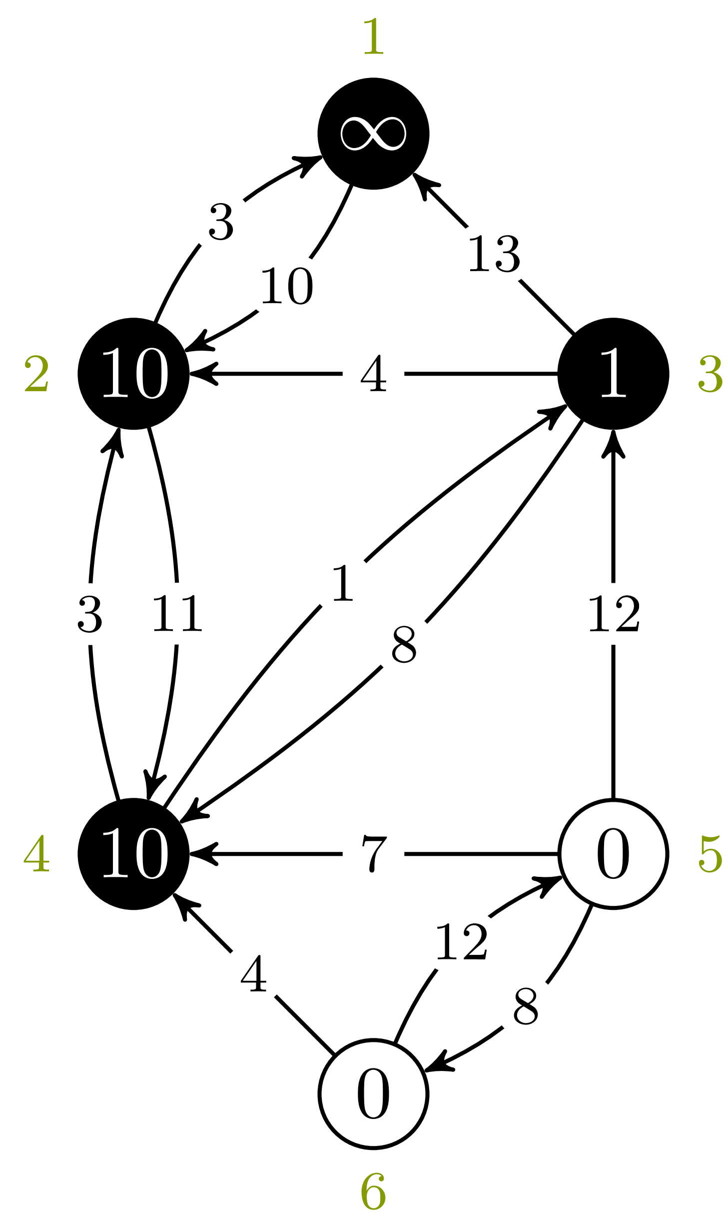
G_f har ingen forøkende sti



$|f| = c(S, T)$ for et snitt (S, T)

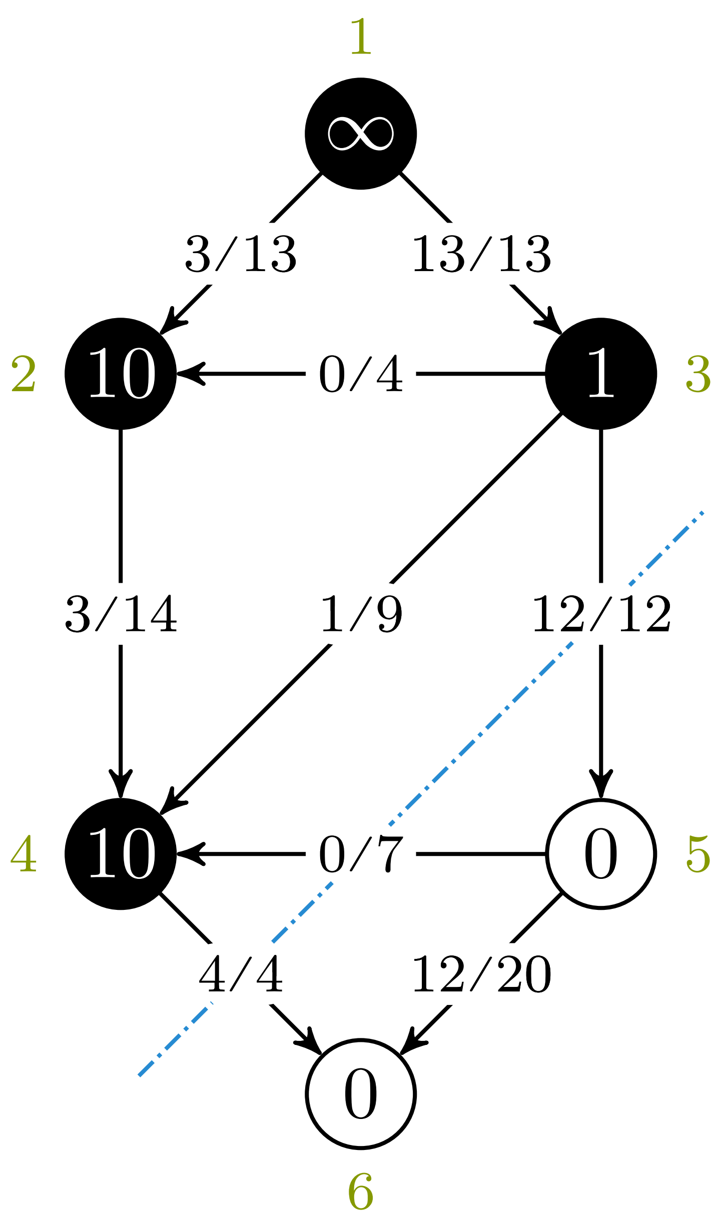


Flyt

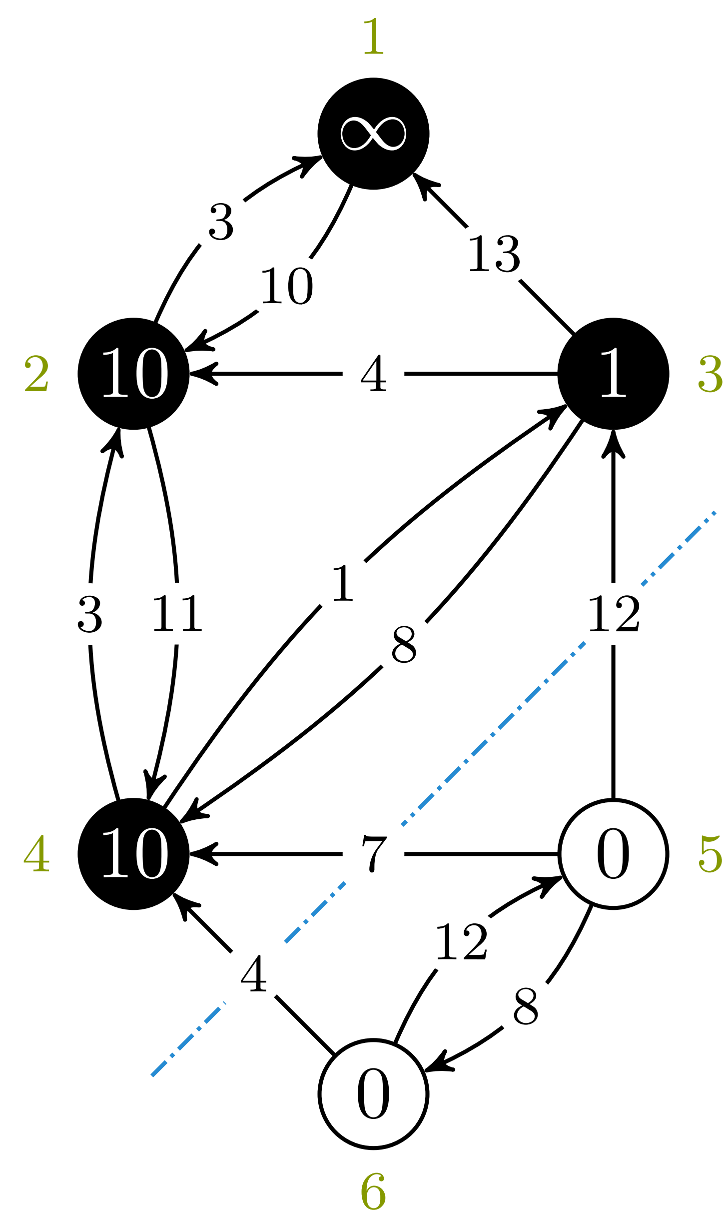


Rest

Etter kjøringen vår av Edmonds-Karp

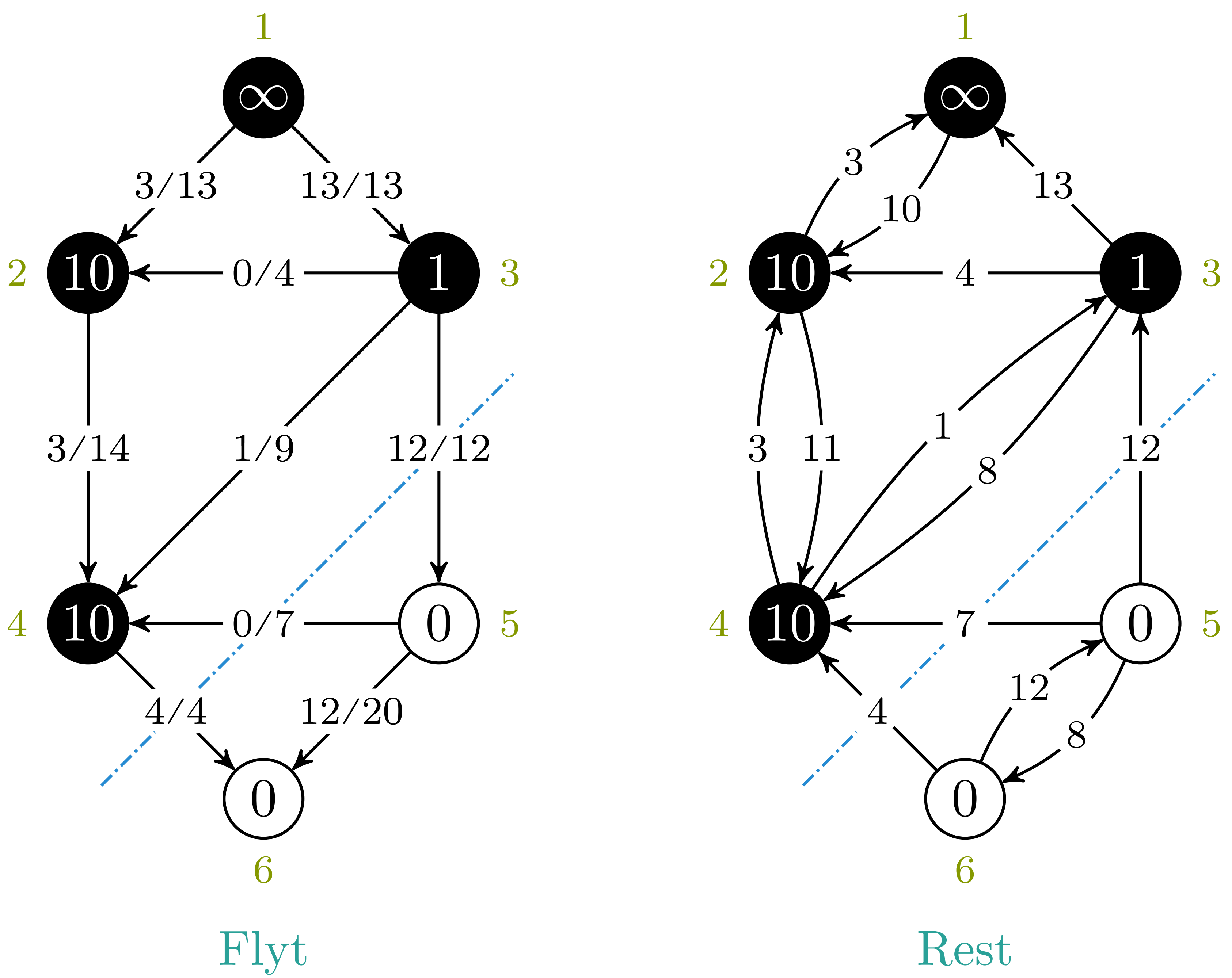


Flyt



Rest

Min. snitt: Mellom svart og hvit. $c(\mathrm{S},\mathrm{T}) = f(\mathrm{S},\mathrm{T}) = 16$



S → T: Fullt. S ← T: Tomt. Ingen sti S ⇝ T i restnetverk!

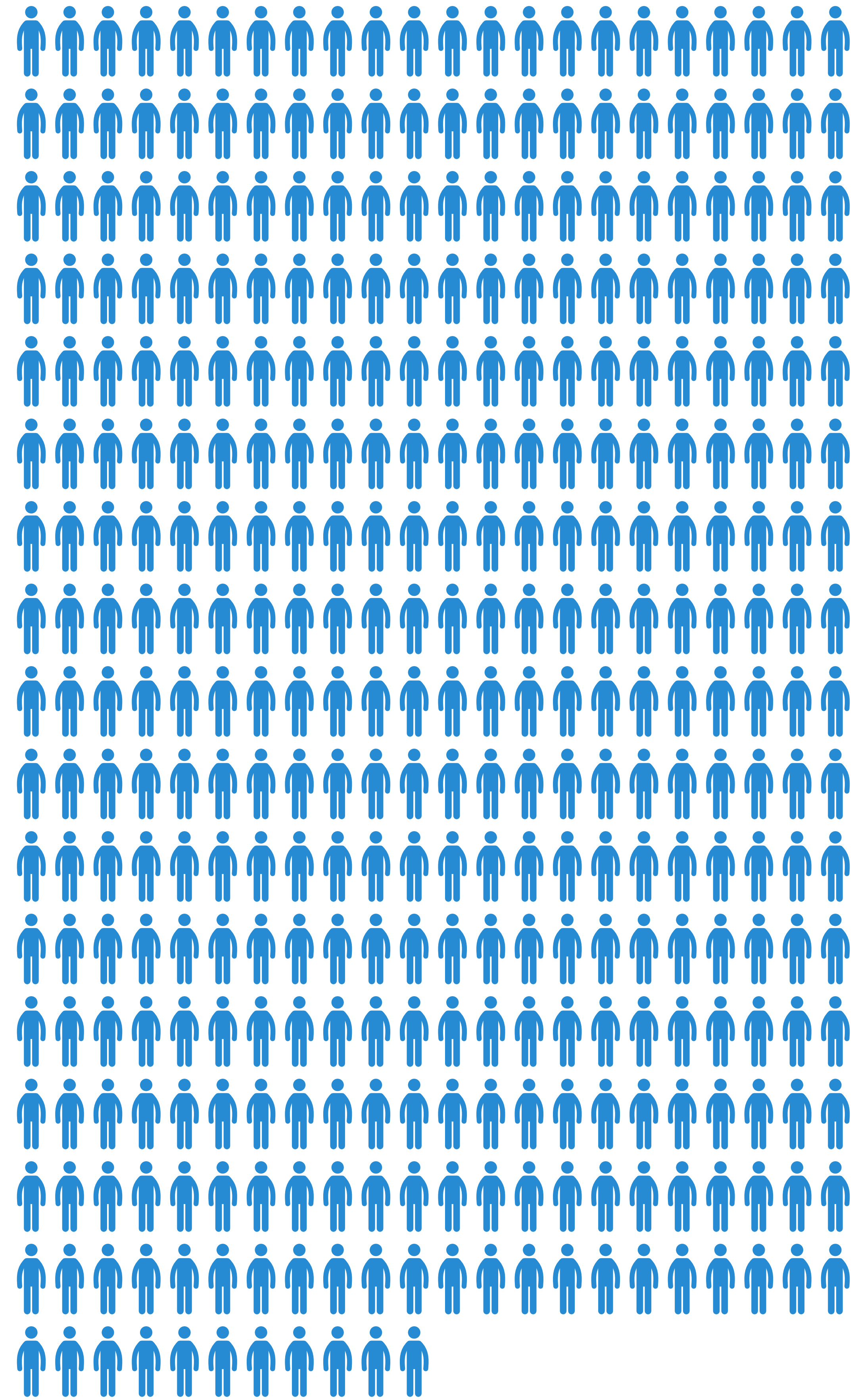
Så, endelig:
Hvordan kommer vi fra ...

1 000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000 000 000

×

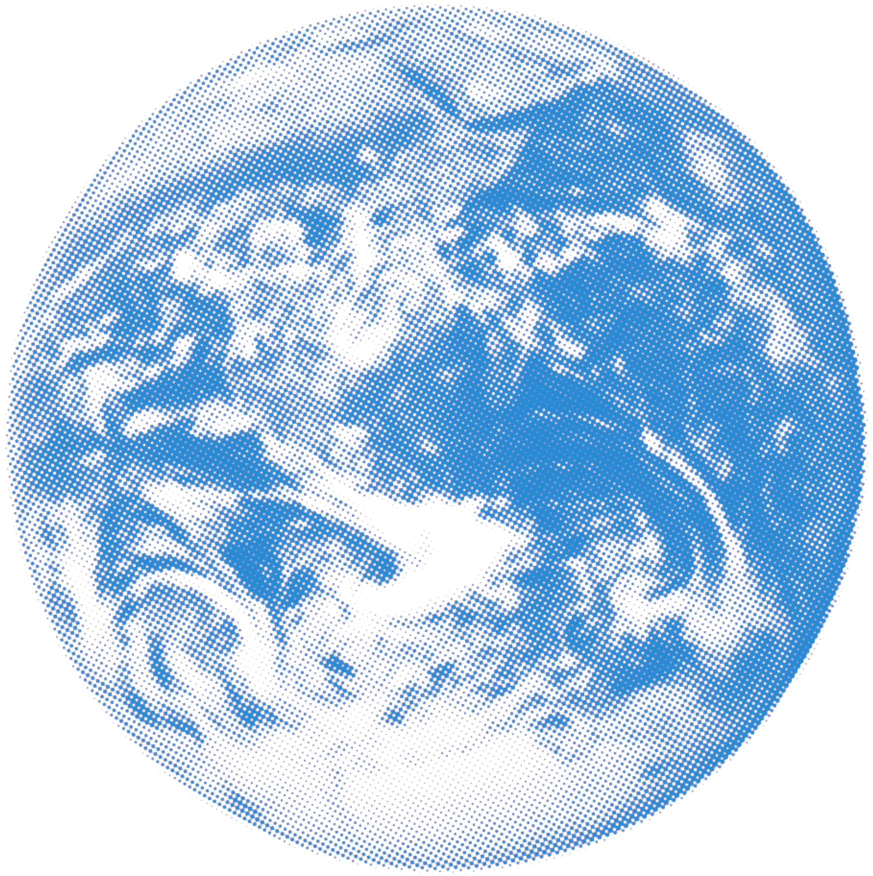
13.8 Ga

=



… til …

$$\frac{1}{2}\text{h} =$$



mlh@ntnu.no

$$\frac{1}{2}\,h\ ?$$

Vi bruker riktignok en litt langsommere algoritme enn det jeg brukte i utregningen, men likevel…

Dette er på mange vis et *eksempel* på bruk av flyt – eller *reduksjon til flyt*. Det går an å tenke seg at reduksjonen foregår i to trinn:

1. Reduser til flyt med mange kilder og sluk – hver donor er en kilde og hver mottaker er et sluk.
2. Reduser dette videre til å bruke én kilde og ett sluk, på den vanlige måten (med superkilde og supersluk).

5:5

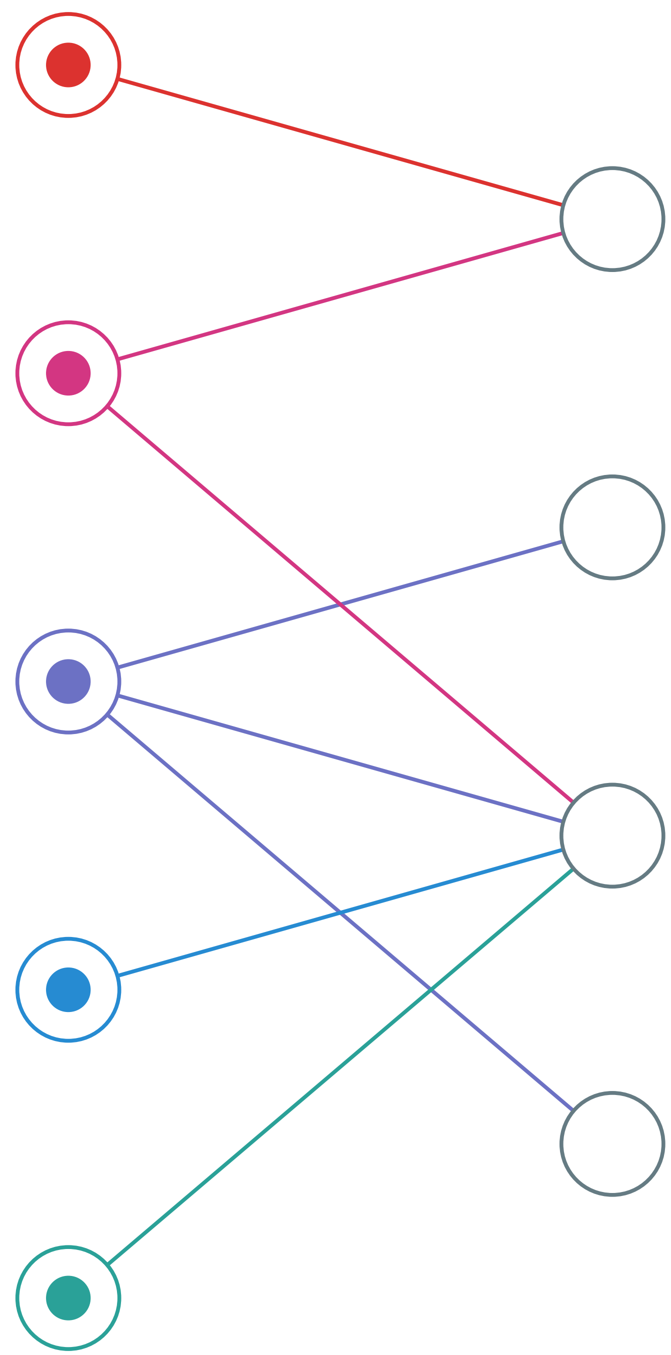
Matching

Matching: Delmengde $M \subseteq E$ for en urettet graf $G = (V, E)$

- Ingen av kantene i M deler noder
- Bipartitt matching: M matcher partisjonene

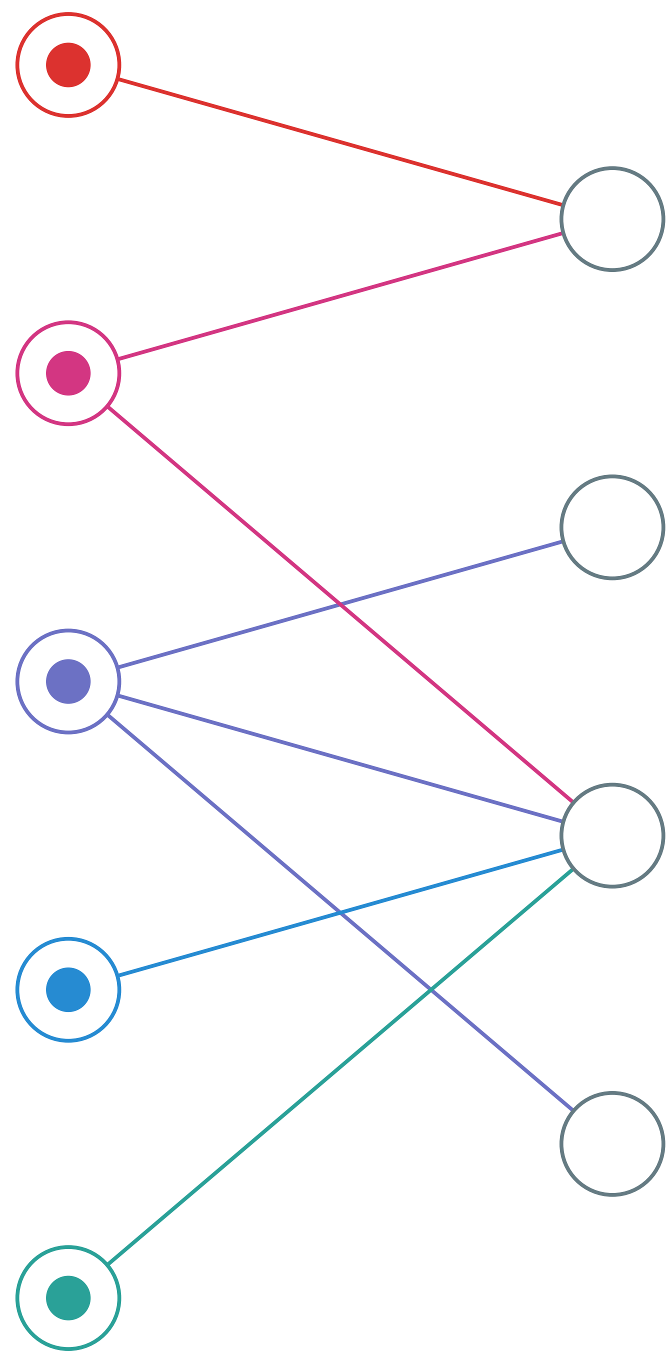
Input: En bipartitt urettet graf $G = (V, E)$.

Output: En matching $M \subseteq E$ med flest mulig kanter, dvs., der $|M|$ er maksimal.

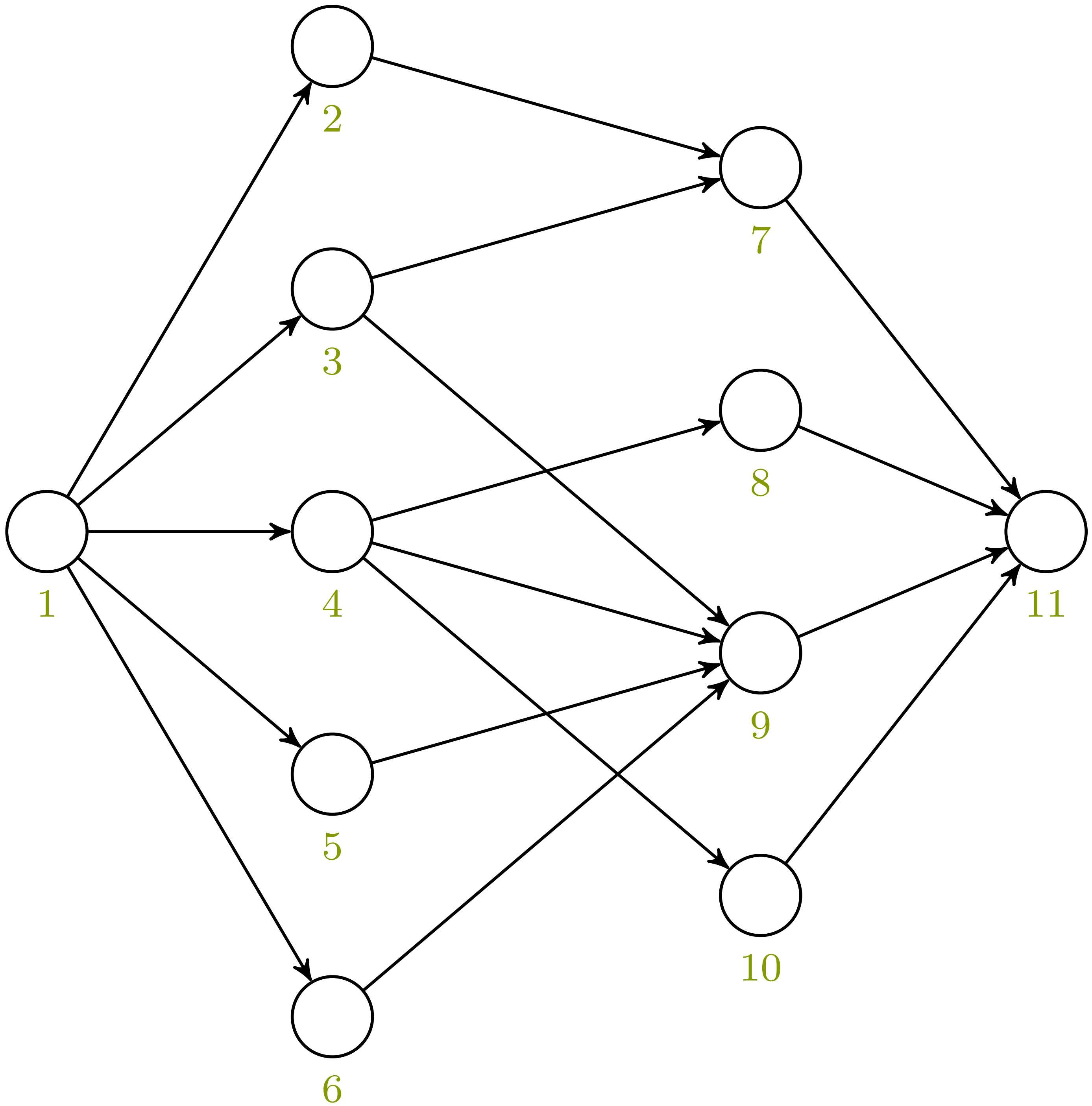


Hvordan får vi matchet flest mulig?

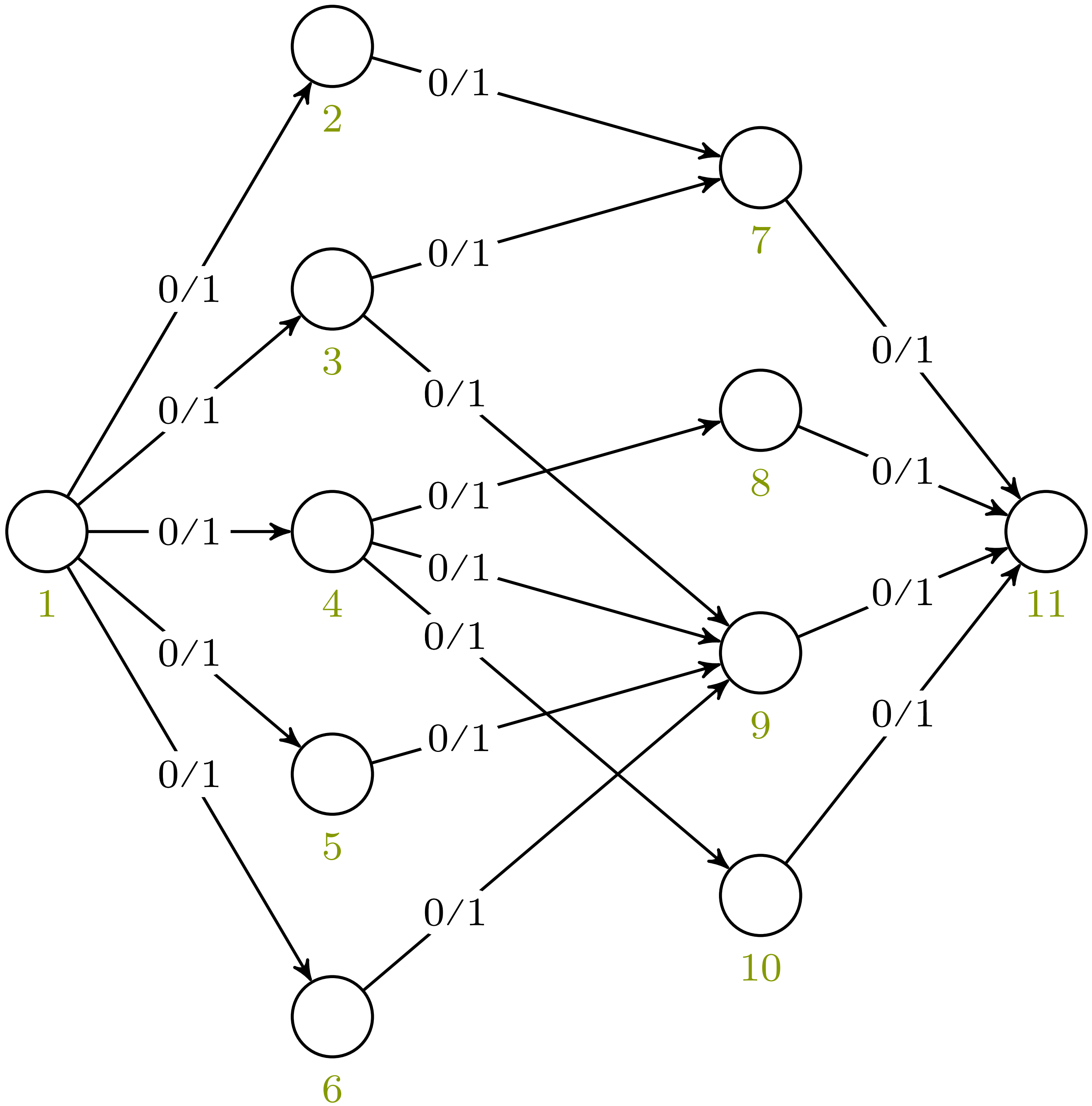
Heltallsteoremet (26.10):
For heltallskapasiteter gir Ford-Fulkerson heltallsflyt



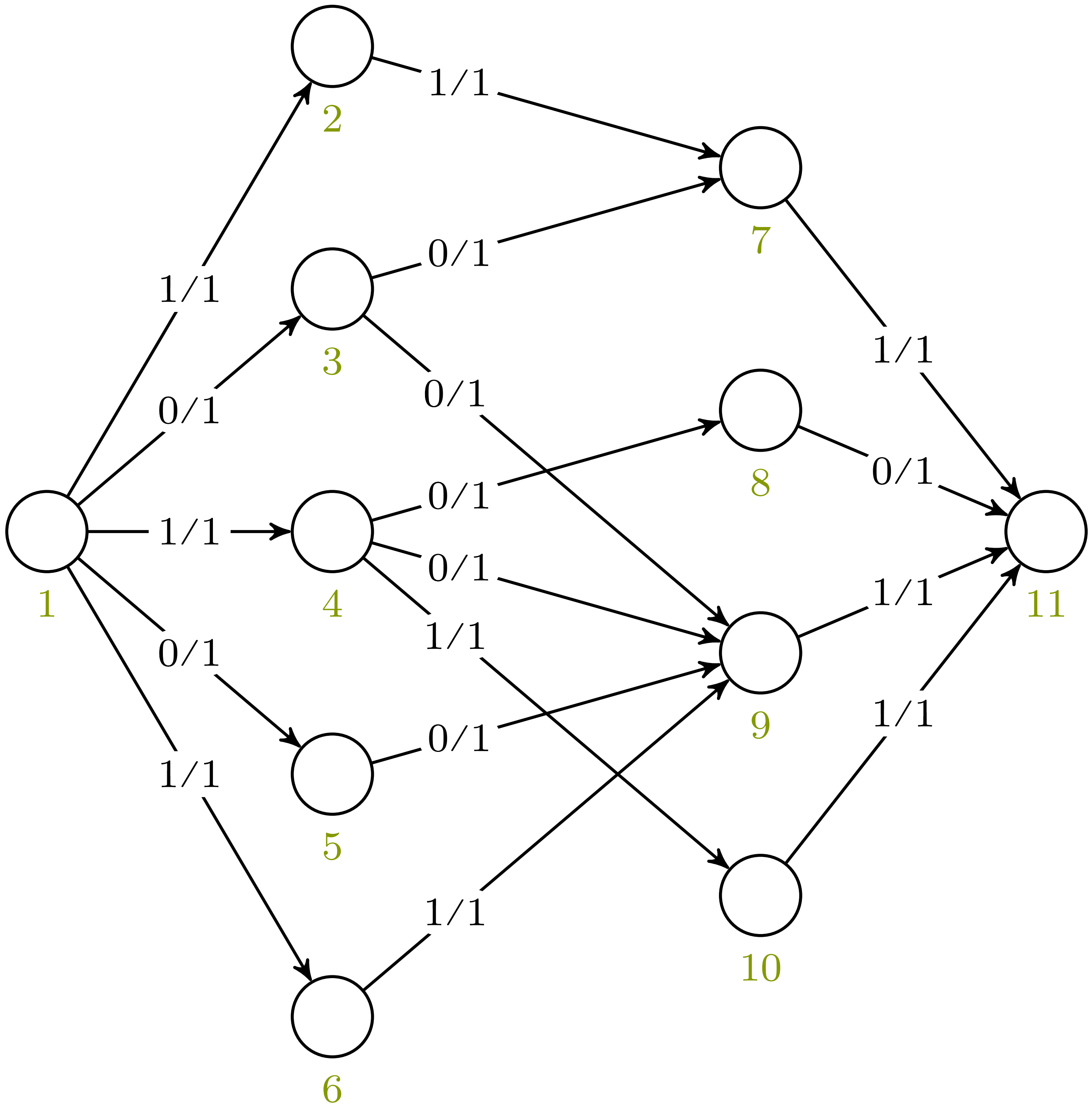
Hvordan får vi matchet flest mulig?



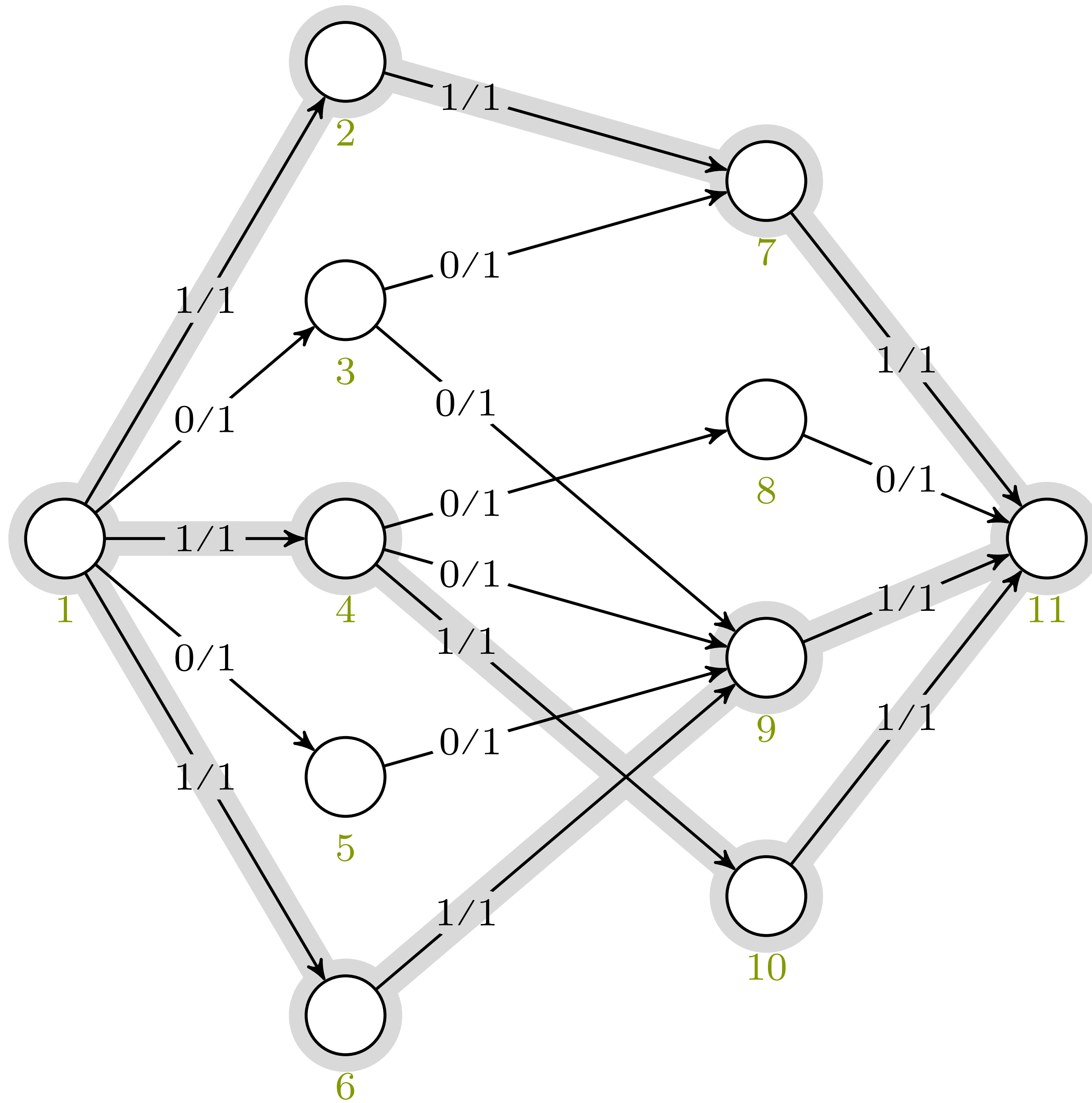
Legg til kilde og sluk



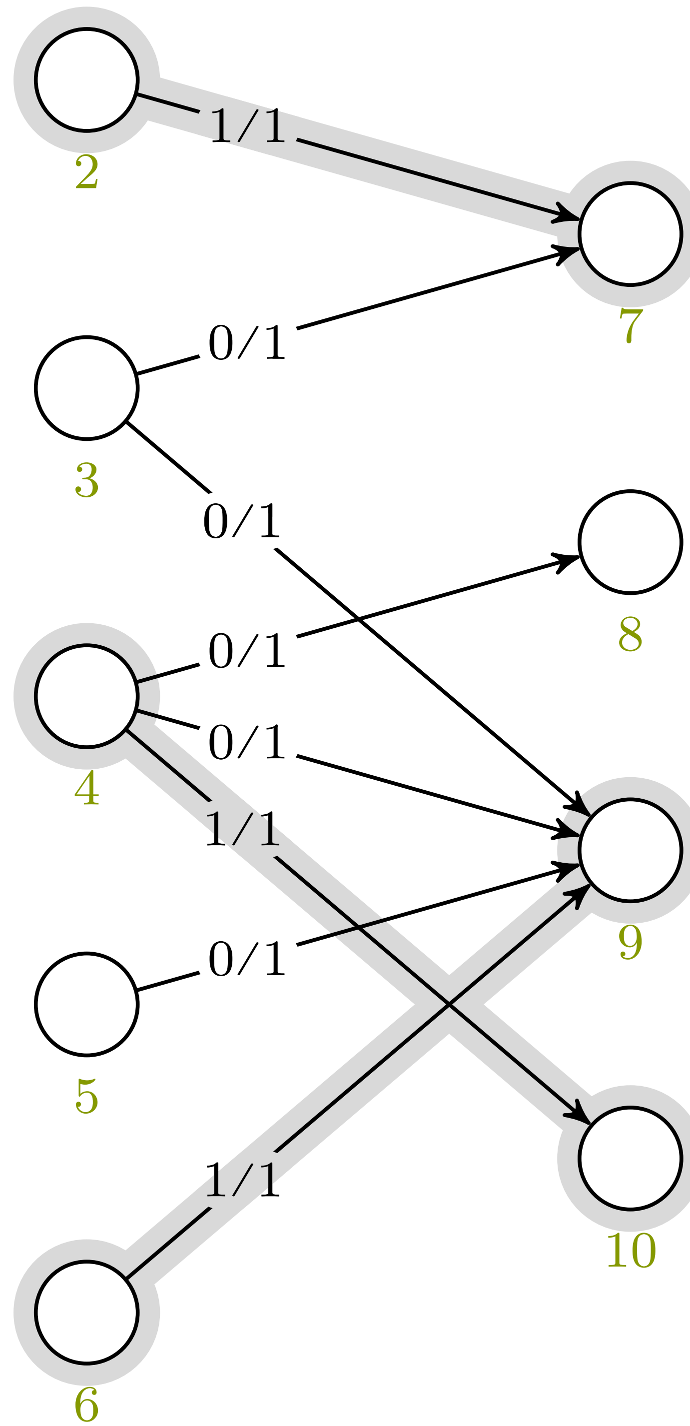
Hver kant og hver node kan inngå i maks ett par



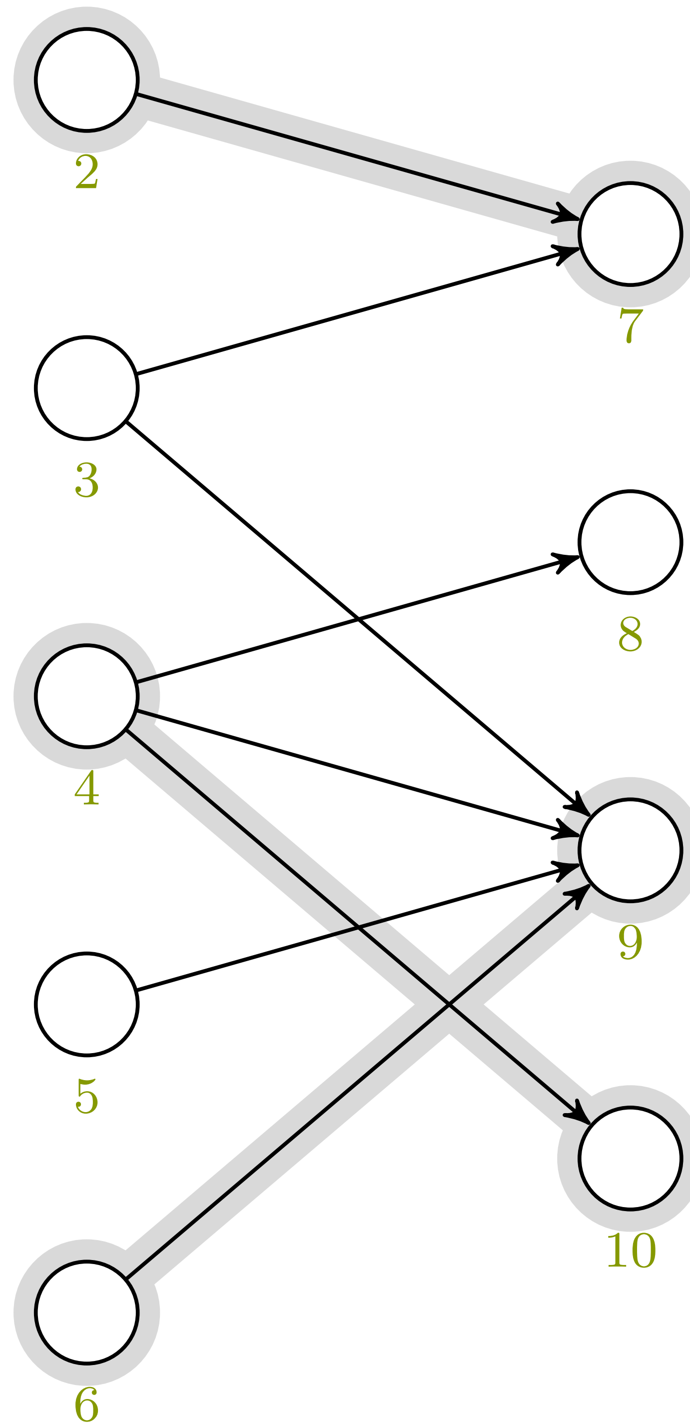
Kanter med flyt inngår i matchingen



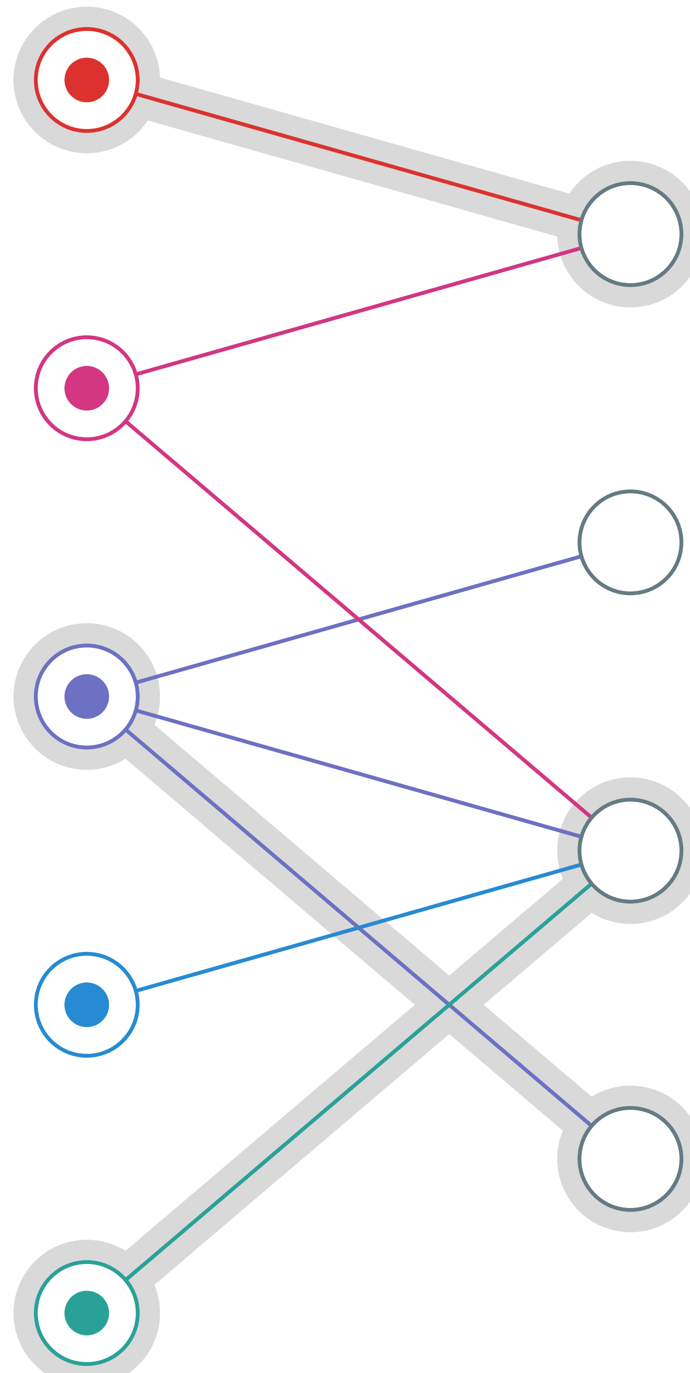
Kanter med flyt inngår i matchingen



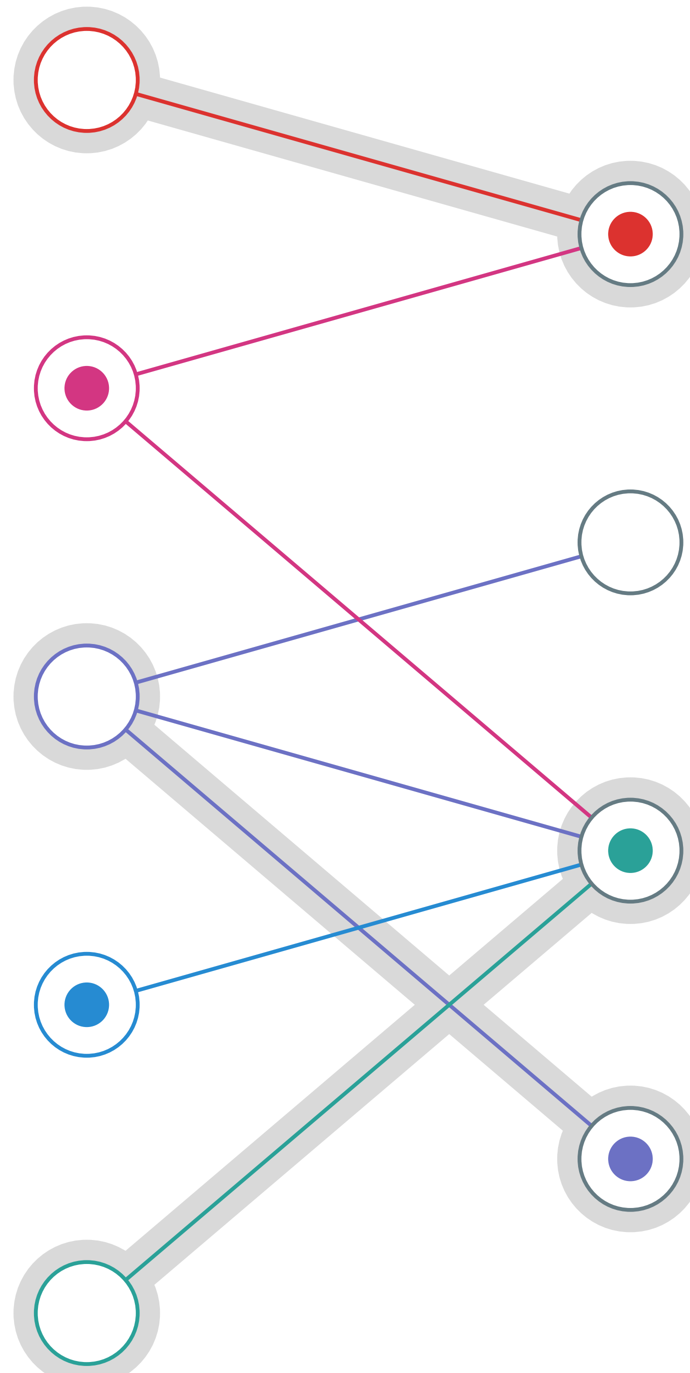
Flytnettverket har gjort jobben sin



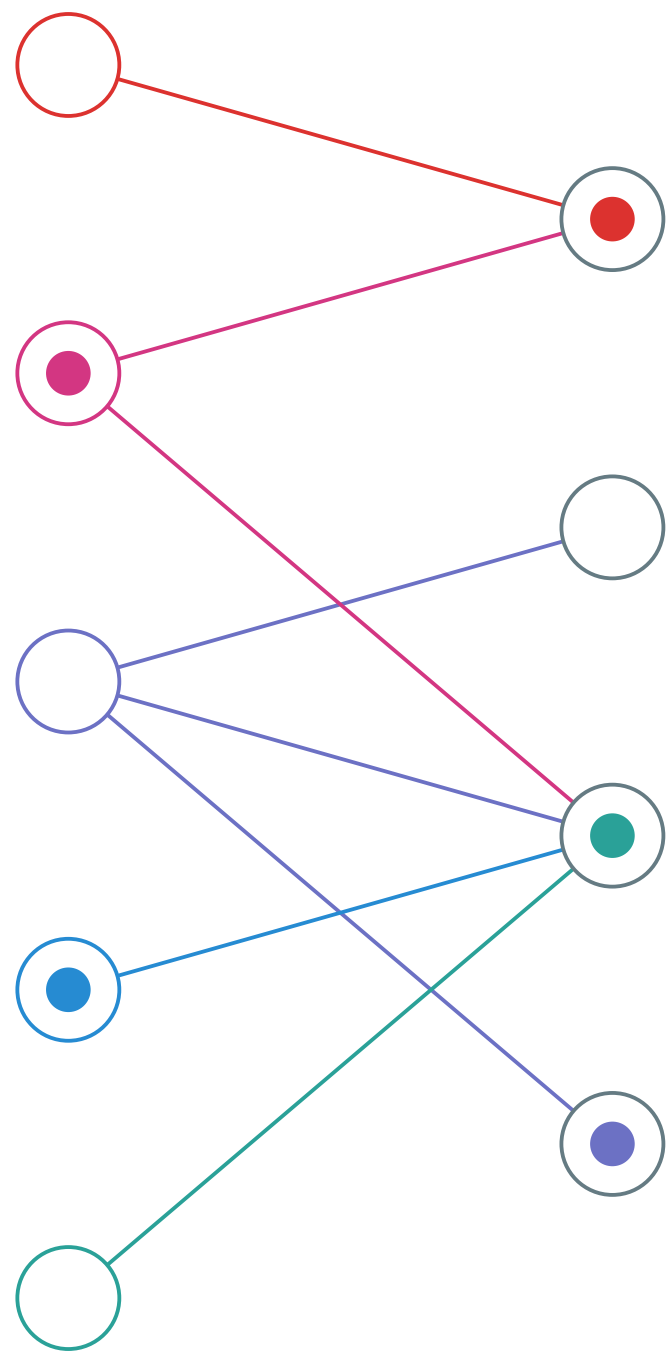
Flytnettverket har gjort jobben sin



Og her har vi endelig løsningen!



Og her har vi endelig løsningen!



Det finnes mer effektive løsninger enn Edmonds-Karp for bipartitt matching. F.esk. Hopcroft-Karp, som har en liten vri: I stedet for å finne én forøkende sti, så finner den så mange som mulig som ikke krysser hverandre. Kjøretiden går fra O(VE2) til O(E·sqrt(V)). Det finnes mange andre algoritmer som virker på andre måter (og som har bedre kjøretider) for både flyt generelt, og matching mer spesifikt.

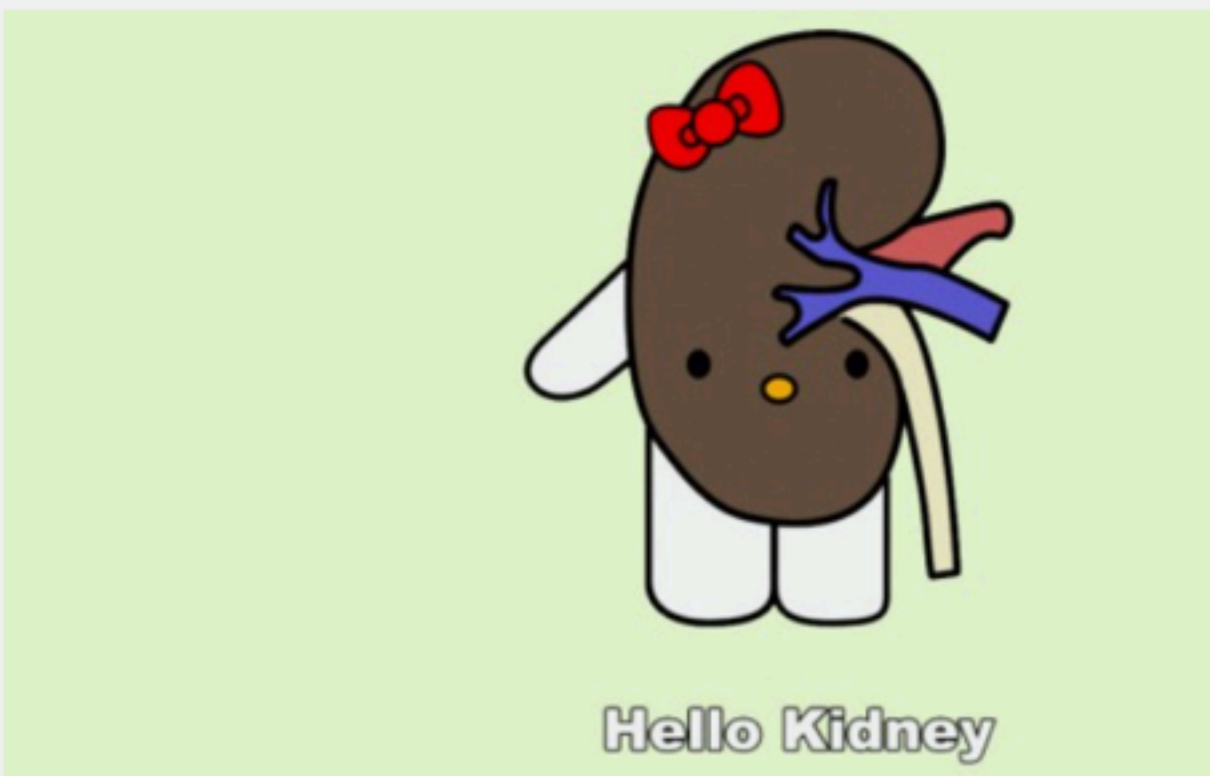
Og her har vi endelig løsningen!

SCIENTIFIC METHOD —

The math of organ donation: Kidneys are an NP-hard problem

Matching donors and recipients is a bit like the traveling salesman problem.

JOHN TIMMER - 1/6/2015, 3:18 PM



We're bilaterally symmetric organisms—we've got matching bits on our left and right side. But many critical organs are present in only a single copy (hello heart) or we need both to function optimally (see: lungs). The kidneys are rare exceptions, as your body gets by just fine with only a single one. That has enabled people to become living kidney donors, with both the donor and recipient continuing life with one kidney.

Dette gjelder naturligvis en litt annen variant enn det vi har sett på. Akkurat hva dette innebærer betyr kommer vi tilbake til i neste forelesning.

Merk: Det gjelder et litt annet problem, ikke matching

1. Problemet
2. Ideer
3. Ford-Fulkerson
4. Minimalt snitt
5. Matching

For den nysgjerrige:
http://www.idi.ntnu.no/~mlh/algkon/flow.pdf