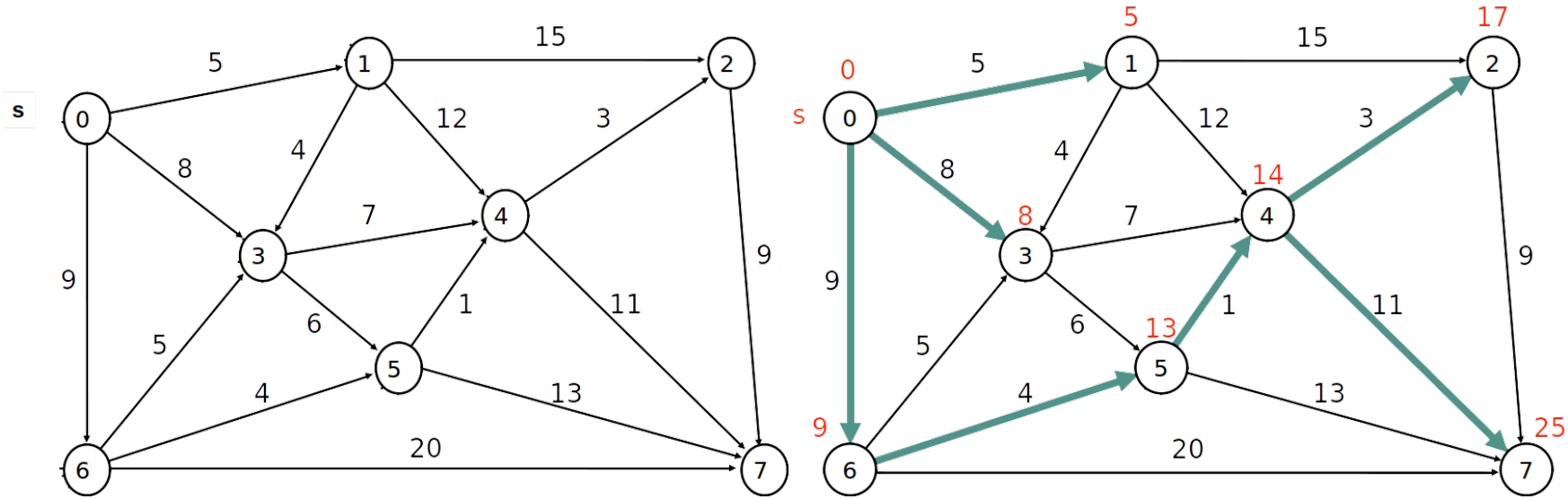
Korteste vej

- Korteste vej
- Egenskaber ved korteste veje
- Dijkstras algoritme
- Korteste vej på en acyklisk rettet graf

Korteste veje

Input: En vægtet, rettet graf G, og en knude s i G.



Algoritmisk problem: beregn korteste veje fra s til alle andre knuder i G.

Korteste veje-træ: Træ over korteste veje fra s.

Korteste veje

- Korteste vej
- Egenskaber ved korteste veje
- Dijkstras algoritme
- Korteste vej på en acyklisk rettet graf

Korteste veje – egenskaber

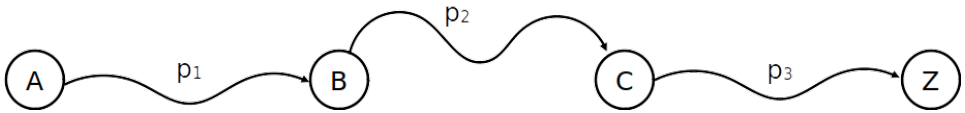
Antagelse: Alle knuder kan nås fra s.

Så der findes en vej (og dermed også en korteste).

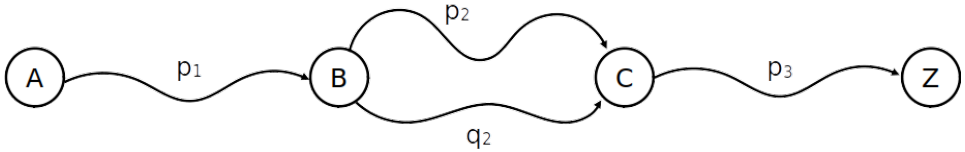
Korteste veje – egenskaber

Delvejsegenskaben: En delvej af en korteste vej er selv en korteste vej.

Dvs: Hvis den korteste vej fra A til Z går igennem B og C, så er der ikke en hurtigere genvej til at forbinde B og C.



Hvorfor: Antag, der var en decideret kortere vej q fra B til C.



Så ville man jo kunne tage den oprindelige rute fra A til B, videre med q, og så den oprindelige rute frem til Z.

Det ville give en kortere vej.

Korteste veje

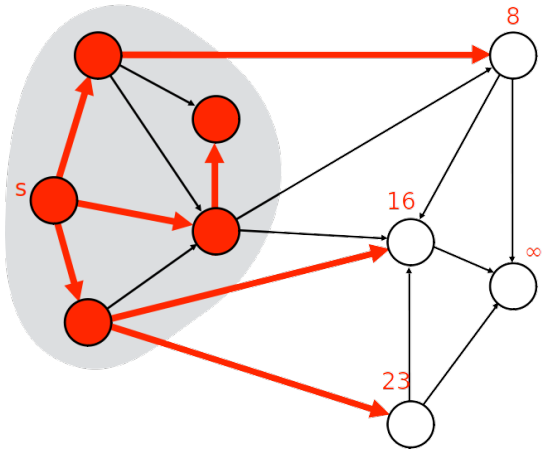
- Korteste vej
- Egenskaber ved korteste veje
- Dijkstras algoritme
- Korteste vej på en acyklisk rettet graf

Korteste veje – Dijkstras algoritme

Input: Rettet graf med ikke-negative kantvægte, og startknude s.

Dijkstras algoritme:

- Vedligeholder for hver knude v:
  - længde af korteste kendte vej v.d
- Opdaterer v.d ved at afspænde kanter.



Start: Starter i knude s (del af input).

- Afstanden til s er 0. $s.\mathrm{d} = 0$.
- Afstanden til alle andre knuder er ukendt. $v.\mathrm{d} = \infty$. $(v \neq s)$

Ide: Bygger et korteste-veje-træ fra s.

Skridt:

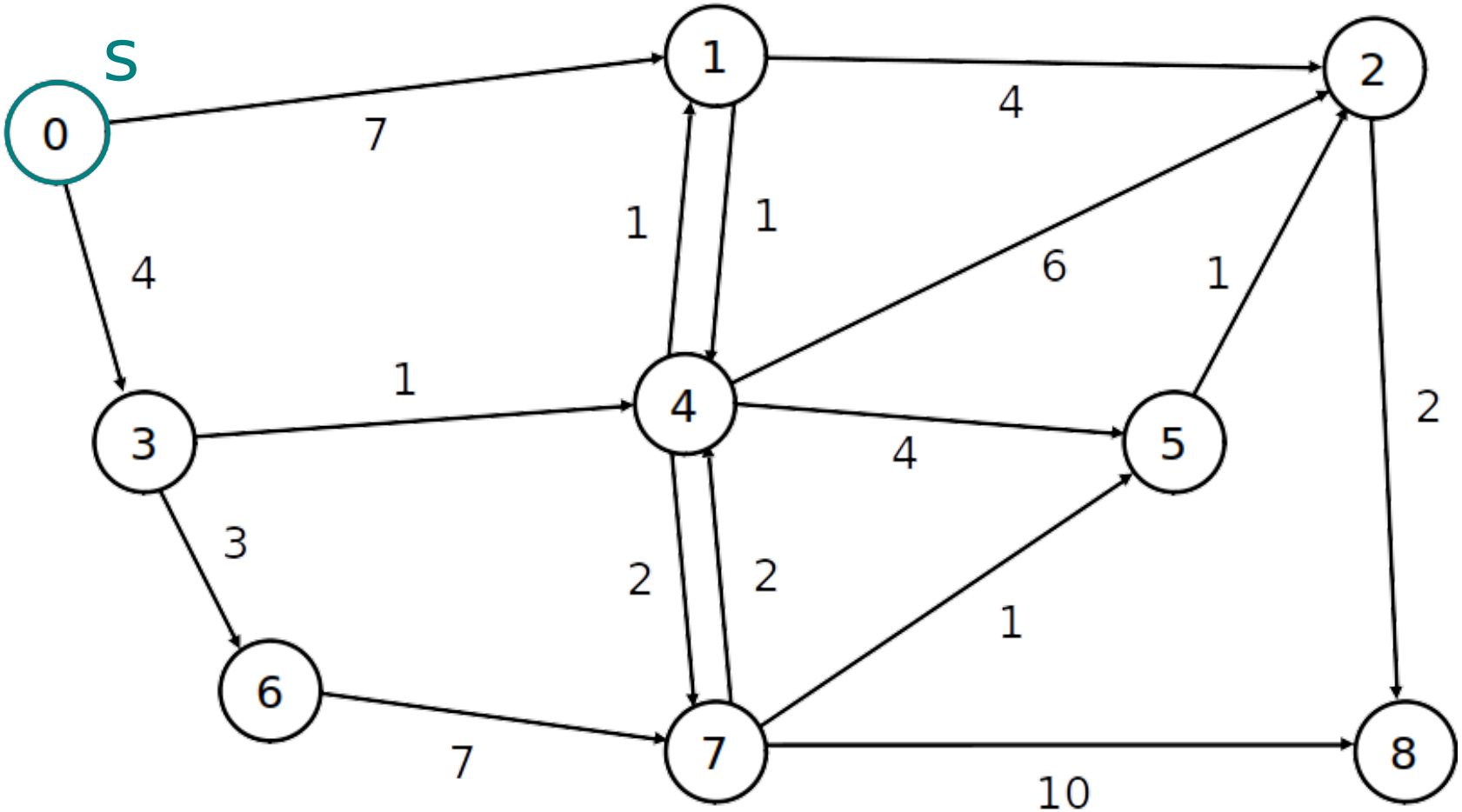
- Tilføj nærmeste knude. Knude v med mindste v.d.
- For hver kant vx ud af v: (Afspænd vx)
  - Opdater afstandsesitmat til x. (Måske går korteste vej til x gennem v)

Korteste veje – Dijkstras algoritme – opgave

Start: $s.\mathrm{d} = 0$, og for $v \neq s$ er $v.\mathrm{d} = \infty$

Skridt: Tilføj nærmeste knude. Knude v med mindste v.d.

For hver kant vx ud af v, afspænd vx: Opdater afstandsesitmat til x.



Korteste veje – Dijkstras algoritme – korrekthed

Dijkstras algoritme beregner korteste veje.

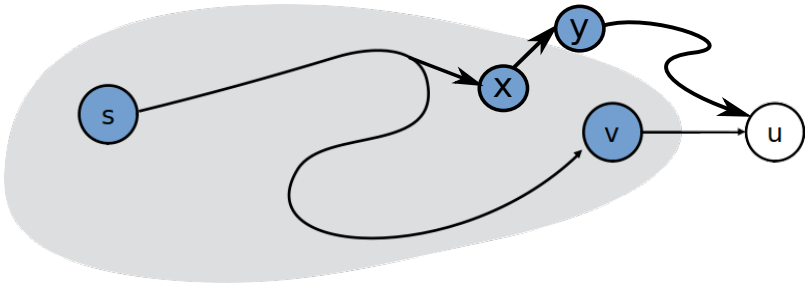
Faktisk: Til enhver tid er træet T bygget ud fra s et korteste-veje-træ.

Bevis: Induktion.

Induktionsstart: T er knuden s.

Skridt: Antag på nuværende tidspunkt i algoritmen, T er et korteste-veje-træ.

Betragt nærmeste knude u.



Den ægte korteste vej $s \longrightarrow u$ forlader T med en kant $x \rightarrow y$.

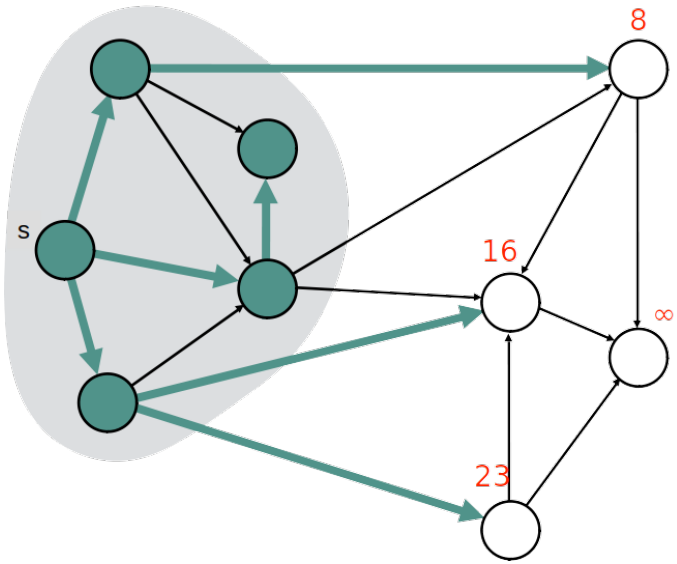
Men hvis vejen $y \longrightarrow u$ har længde > 0, så y er tættere på s end u.

Men så ville vi jo have tilføjet y og ikke u. (Nærmeste knude.) Så $y = v$.

Ergo:

Når vi tilføjer den afspændte kant $\rightarrow u$ til T, har vi igen et korteste-veje-træ.

Korteste veje – Dijkstras algoritme – implementation



Implementation: Hvordan implementerer vi Dijkstra?

Udfordring: Hvordan finder vi hele tiden den nærmeste knude?

Løsning: Prioritetskø.

- Nøgle af v: $v.\mathrm{d}$
- Afspænd kanter: decrease-key $\times m$
- Nærmeste knude: extract-min. $\times n$

Tid? $m\times$decrease-key, $n\times$extract-min. Total: $O(m \log n)$.

Korteste veje – Dijkstras algoritme – pseudokode

```
DIJKSTRA(G, s)
    for alle knuder v∈V
    v.d = ∞
        v.π = null
        INSERT(P,v)
    DECREASE-KEY(P,s,0)
    while (P ≠ ∅)
        u = EXTRACT-MIN(P)
        for alle v som u peger på
            RELAX(u,v)
```

```
RELAX(u,v)
    if (v.d > u.d + w(u,v))
        v.d = u.d + w(u,v)
        DECREASE-KEY(P,v,v.d)
        v.π = u
```

Korteste veje – Dijkstras algoritme – køretid

Dijkstras algoritme kan implementeres med prioritetskø.

Priority queue	INSERT	EXTRACT-MIN	DECREASE-KEY	Total
array	O(1)	O(n)	O(1)	$O(n^2)$
binary heap	O(log n)	O(log n)	O(log n)	O(m log n)
Fibonacci heap	O(1)†	O(log n)†	O(1)†	O(m + n log n)

Urettede grafer: Dijkstras algoritme virker også på urettede grafer, svarer til at erstatte hver kant $u \leftrightarrow v$ med to kanter $u \rightarrow v$ og $u \leftarrow v$

Korteste veje

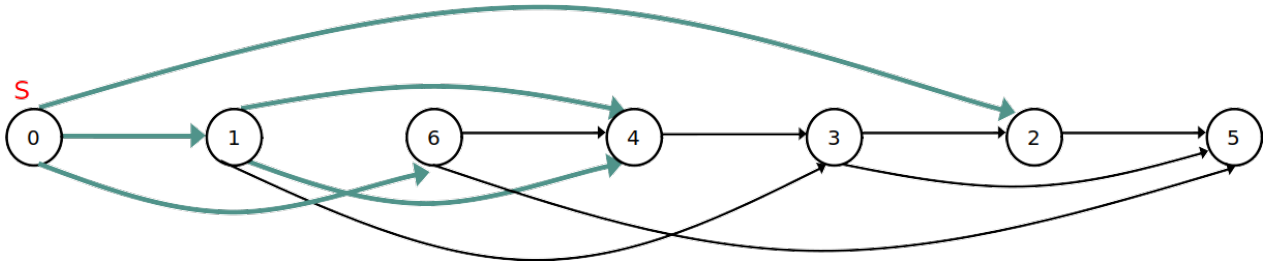
- Korteste vej
- Egenskaber ved korteste veje
- Dijkstras algoritme
- Korteste vej på en acyklisk rettet graf (DAG)

Korteste veje – acykliske rettede grafer

Observation: En acyklisk rettet graf er en ganske særlig type graf.

Spørgsmål: Bedre algoritmiske resultater for acykliske rettede grafer?

Svar: Ja.



Algoritme:

- Find topologisk sortering. $O(m)$
- Betragt knuderne ifølge den topologiske sorteringsorden. $O(n)$
- For knude v, afspænd alle kanter ud af v. $O(m)$

Tid: $O(m)$

Bonus, kun for acykliske grafer: virker også med negative kantvægte.

Korteste veje

- Korteste vej
- Egenskaber ved korteste veje
- Dijkstras algoritme
- Korteste vej på en acyklisk rettet graf