

Korteste veje

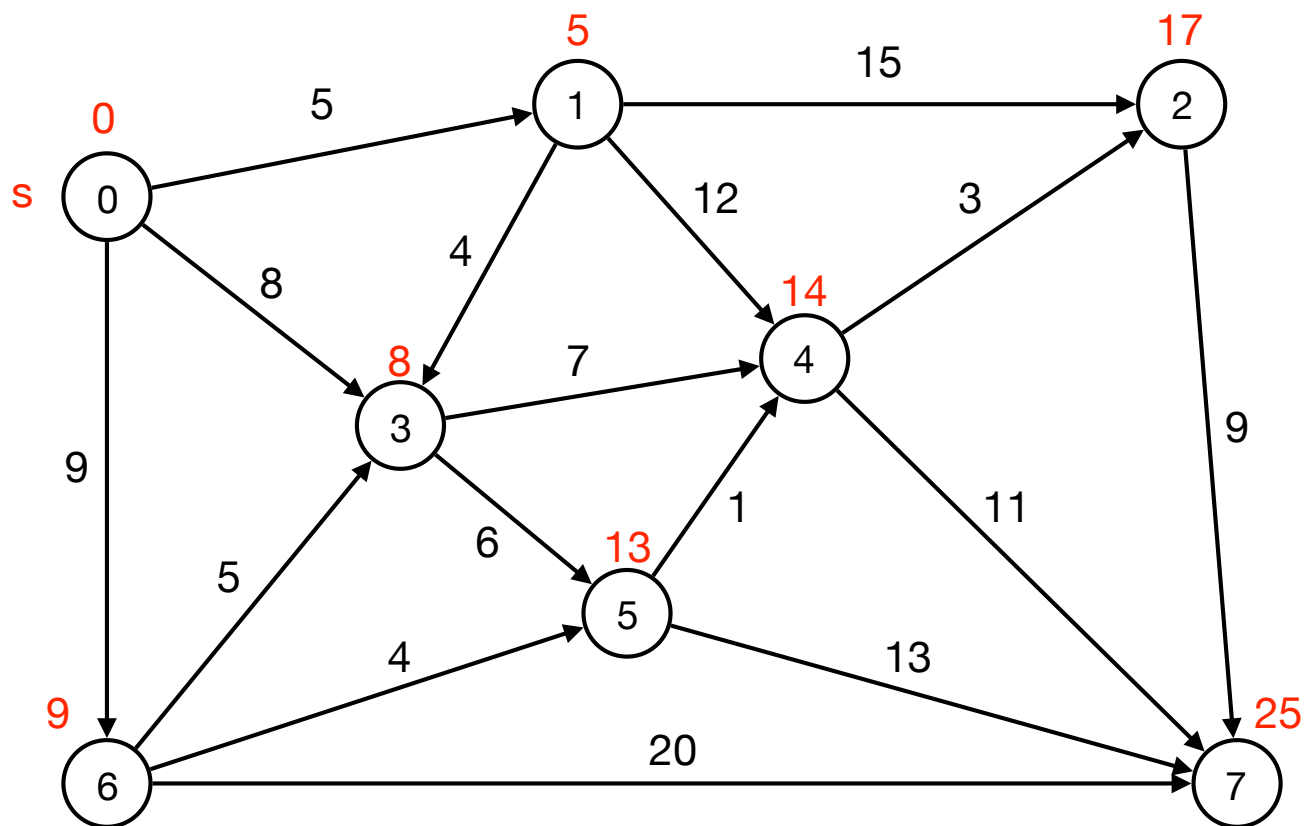
- Introduktion
- Egenskaber for korteste veje
- Dijkstras algoritme
- Korteste veje på DAGs

Korteste veje

- **Introduktion**
- Egenskaber for korteste veje
- Dijkstras algoritme
- Korteste veje på DAGs

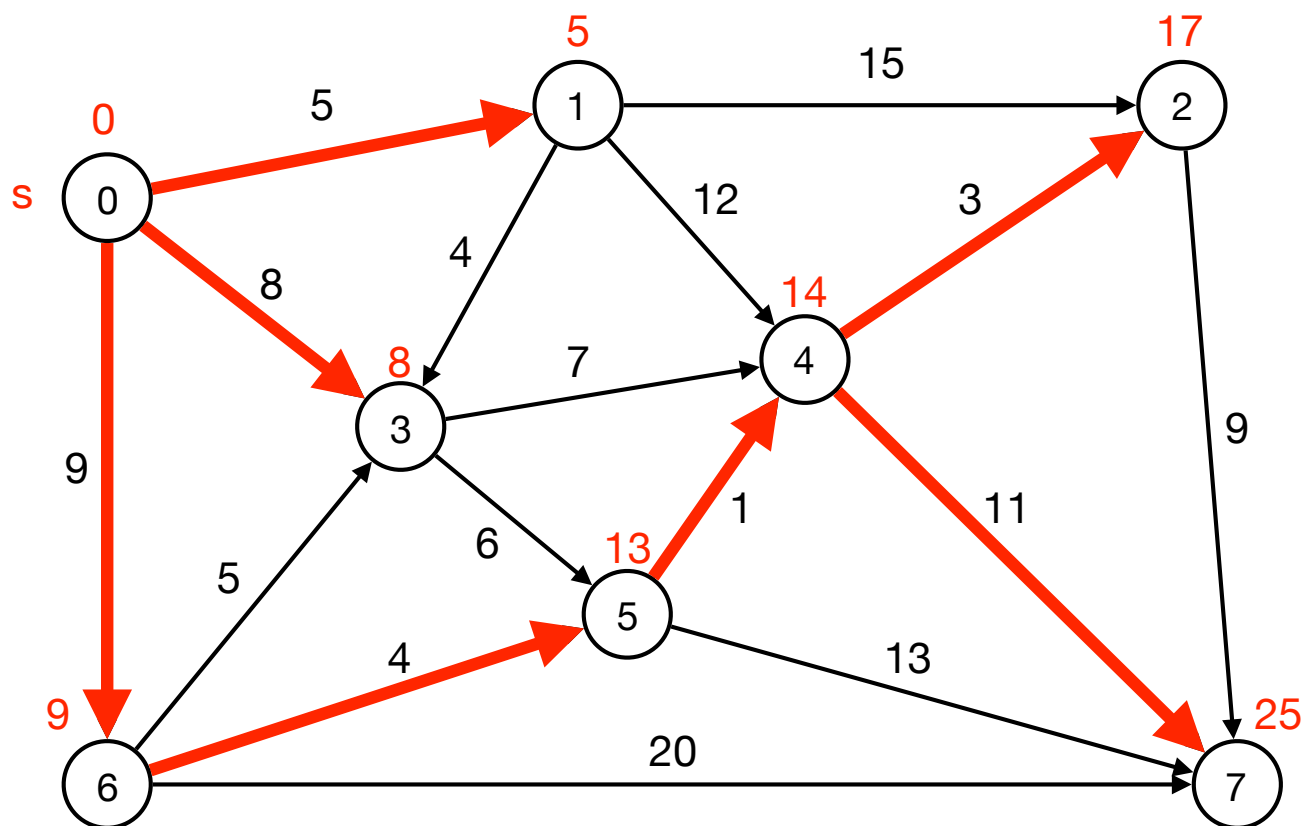
Introduktion

- **Korteste veje.** Givet en orienteret, vægtet graf G og en knude s , find korteste vej fra s til alle knuder i G .



Introduktion

- **Korteste veje.** Givet en orienteret, vægtet graf G og en knude s , find korteste vej fra s til alle knuder i G .
- **Korteste veje træ.** Repræsenterer korteste veje som et **træ** fra s .



Anvendelser

- Google maps, bilnavigation, rutning i netværk, skedulering, pipelining, ...

Korteste veje

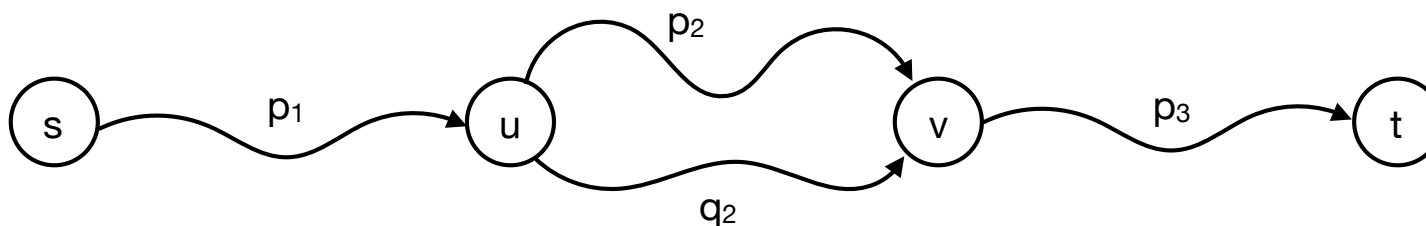
- Introduktion
- Egenskaber for korteste veje
- Dijkstras algoritme
- Korteste veje på DAGs

Korteste veje egenskaber

- **Simplificerende antagelse.**
 - Alle knuder kan nås fra s.
- \Rightarrow der findes altid (korteste) vej til en knude.

Korteste veje egenskaber

- **Lemma.** Enhver delvej af en korteste vej er en korteste vej.
- **Bevis.** Modstridsbevis.
 - Kig på en korteste vej p fra s til t bestående af p_1 , p_2 og p_3 .



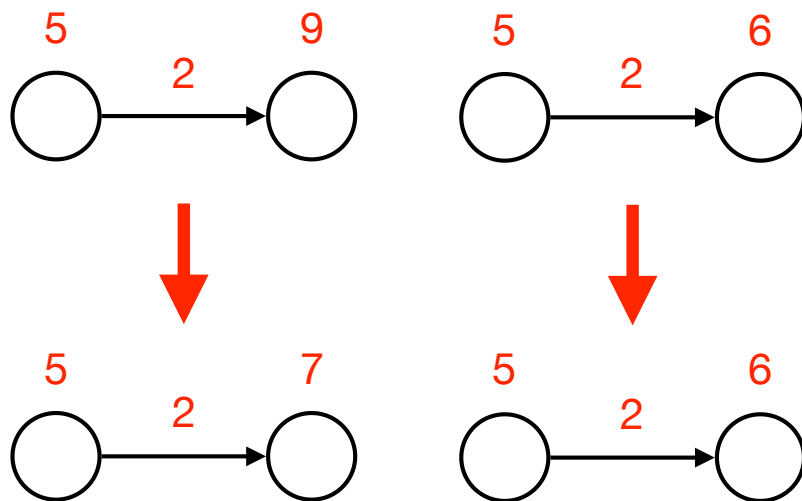
- Antag q_2 er kortere en delvej p_2 .
- \Rightarrow Da er p_1 , q_2 og p_3 en kortere vej fra s til t end p .

Korteste veje

- Introduktion
- Egenskaber for korteste veje
- Dijkstras algoritme
- Korteste veje på DAGs

Dijkstras algoritme

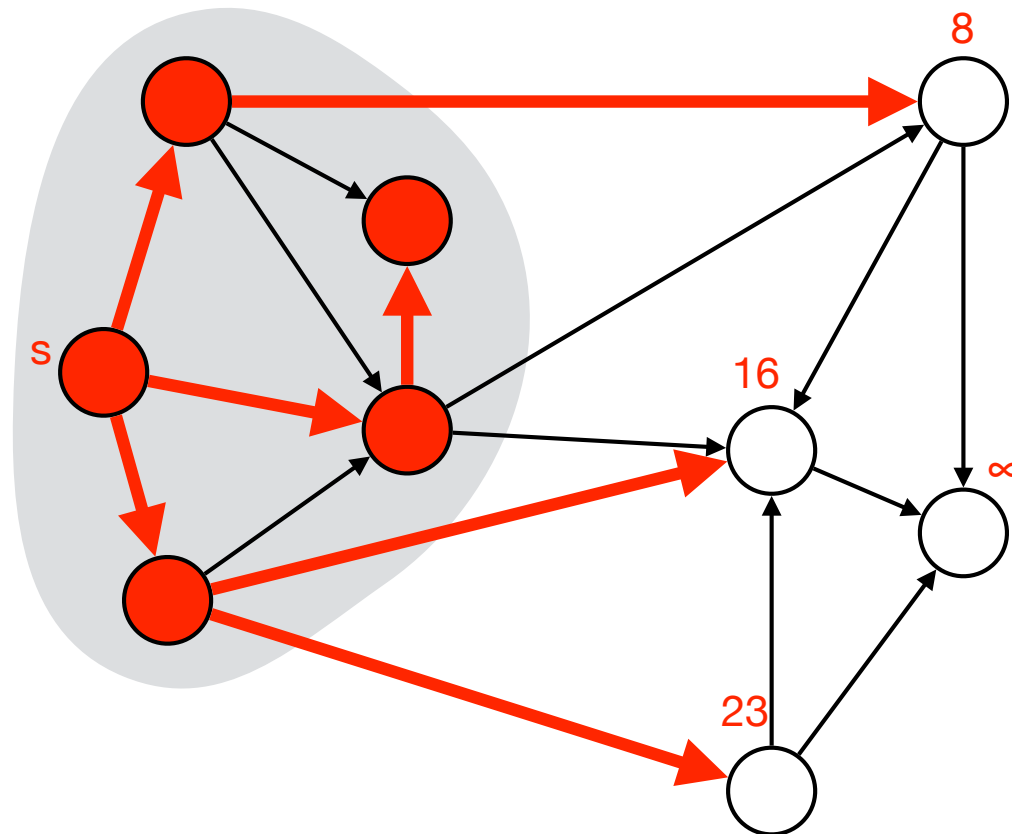
- **Mål.** Givet en orienteret og vægtet graf G med **ikke-negative vægte** og en knude s, beregn korteste vej fra s til alle andre knuder.
- **Dijkstras algoritme.**
 - Vedligeholder **afstandsestimat** v.d for hver knude v = længde af korteste **kendte** vej til v fra s.
 - Opdaterer afstandsestimater ved at **afspænde** (*relax*) kanter.

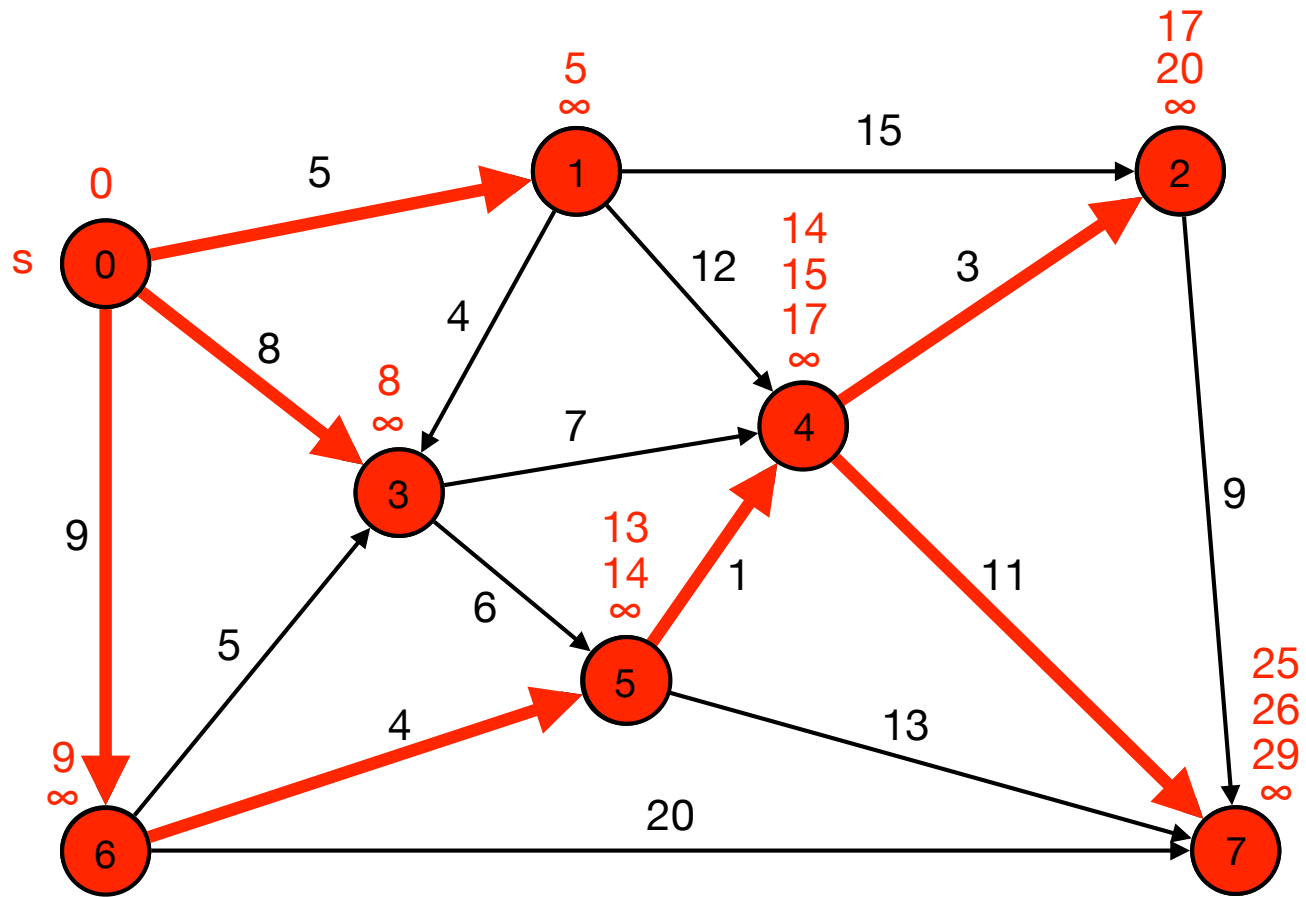


```
RELAX(u,v)
  if (v.d > u.d + w(u,v))
    v.d = u.d + w(u,v)
```

Dijkstras algoritme

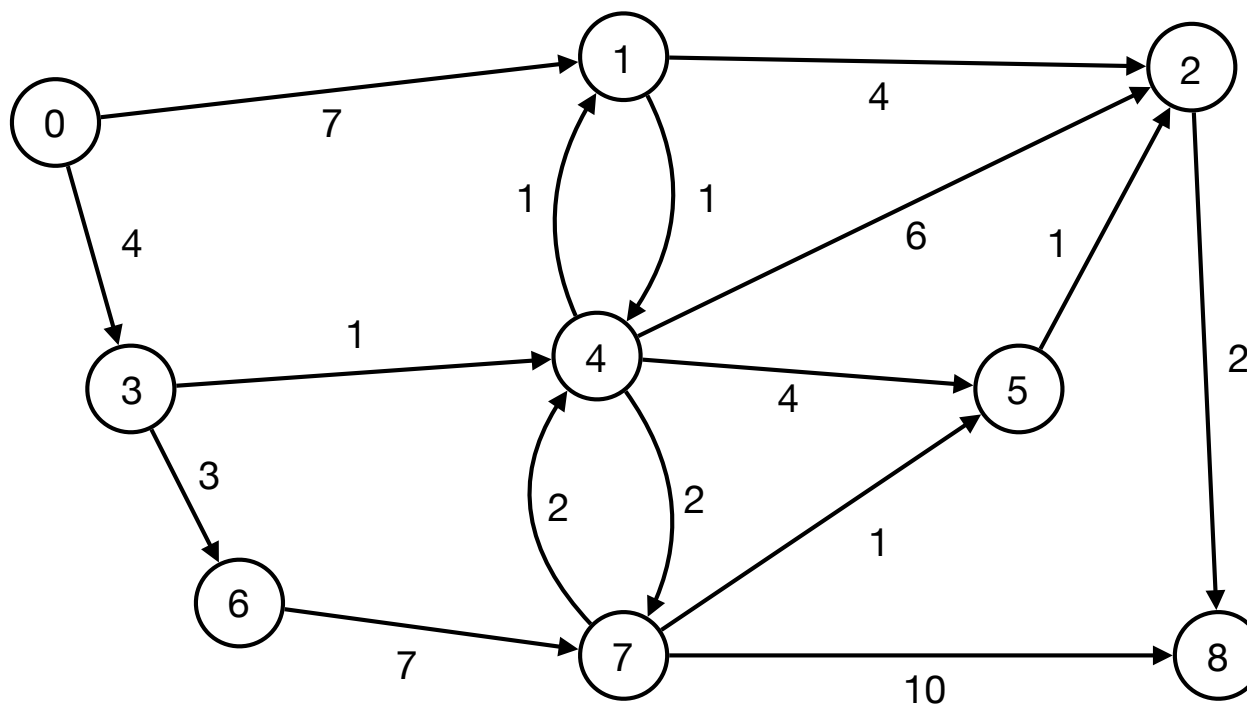
- Sæt $s.d = 0$ og $v.d = \infty$ for alle knuder $v \in V \setminus \{s\}$.
- Opbyg et træ T fra s .
- I hvert skridt, tilføj knude v med **mindste** afstandsestimat til T .
- Afspænd alle kanter som v peger på.





Dijkstras algoritme

- Sæt $s.d = 0$ og $v.d = \infty$ for alle knuder $v \in V \setminus \{s\}$.
- Opbyg et træ T fra s .
- I hvert skridt, tilføj knude v med **mindste** afstandsestimat til T .
- Afspænd alle kanter som v peger på.
- **Opgave.** Håndkør Dijkstras algoritme fra knude 0 på følgende graf.



Dijkstras algoritme

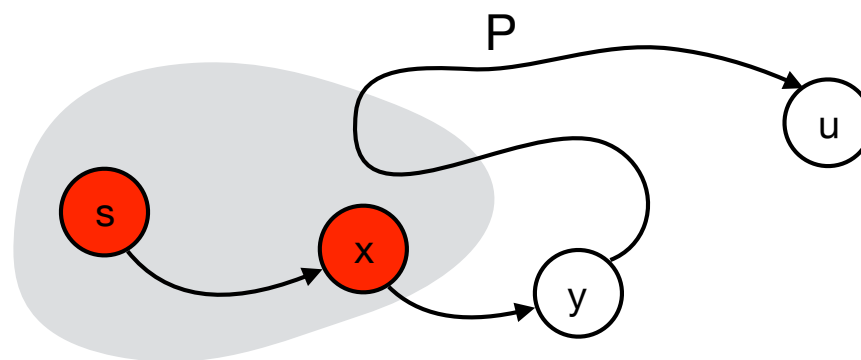
- **Notation.** $\delta(s,v)$ er længden af den korteste vej fra s til v .
- **Lemma.** Når afstandsestimater opdateres med afspænding er $v.d \geq \delta(s,v)$ for alle knuder v .
- **Bevis.** Induktion over antallet af afspændinger.

Dijkstras algoritme

- **Lemma.** Dijkstras algoritme beregner $v.d = \delta(s,v)$ for alle knuder v .
- **Bevis.** Modstridsbevis.
 - Lad u være den **første** knude valgt af algoritmen så $u.d$ er forkert. Lad P være den korteste vej fra s til u og lad (x,y) være første kant der forlader S . Kig på tidspunkt hvor algoritmen tilføjer u til S :

- Vi har:

1. $u.d \leq y.d$ (alg. valgte u før y).
2. $u.d > \delta(s,u)$ (u var forkert).



Vi har: $u.d > \delta(s,u) = w(P)$ (2 + def.)
 $\geq w(P \text{ til } x) + w(x,y)$ (P skåret af)
 $= \delta(s,x) + w(x,y)$ (delvej af korteste vej er korteste vej)

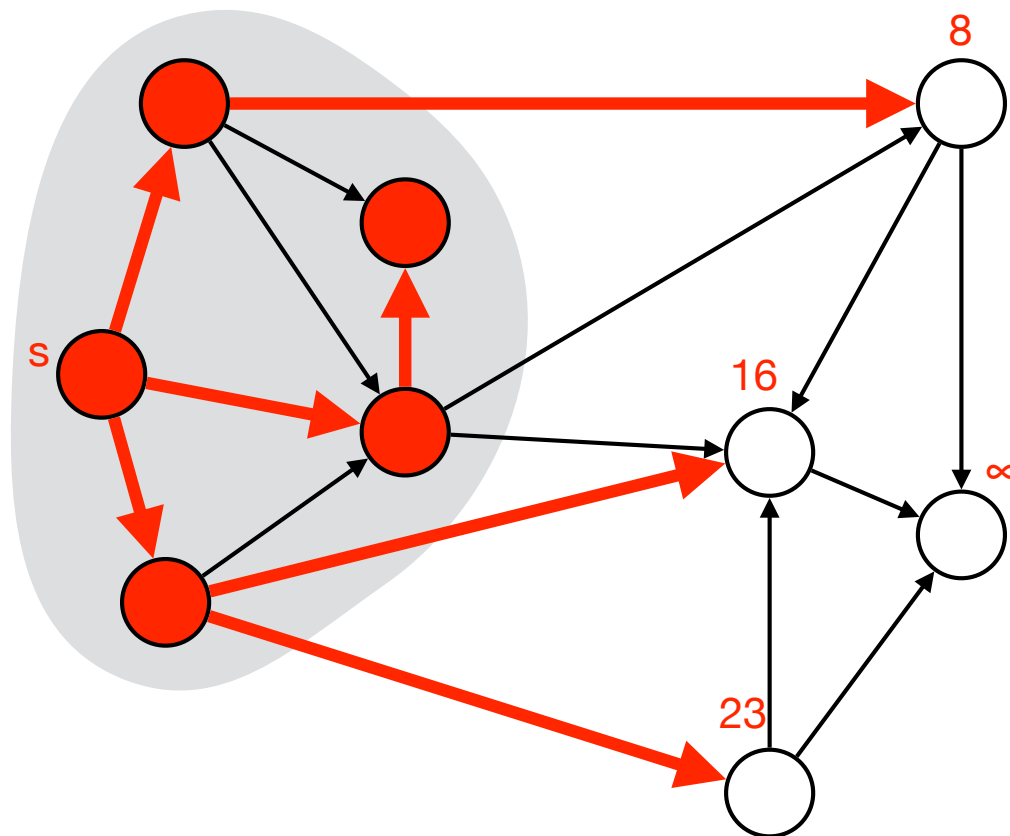
- $x \in S$ så (x,y) afspændt så:

$$y.d \leq x.d + w(x,y)$$
$$= \delta(s,x) + w(x,y) \quad (u \text{ er } \textbf{første} \text{ knude med forkert estimat så } x.d = \delta(s,x))$$

- $\Rightarrow u.d > \delta(s,x) + w(x,y) \geq y.d$ i modstrid med 1.

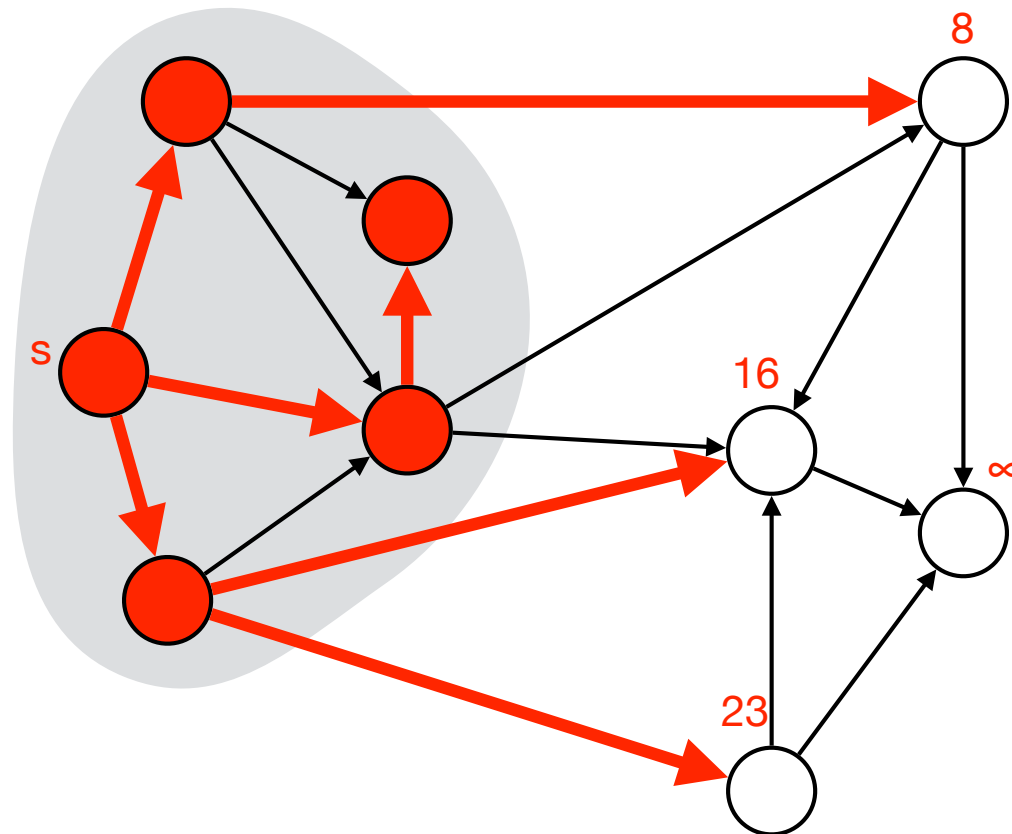
Dijkstras algoritme

- **Implementation.** Hvordan implementerer vi Dijkstras algoritme?
- **Udfordring.** Find knude med mindste afstandsestimat.



Dijkstras algoritme

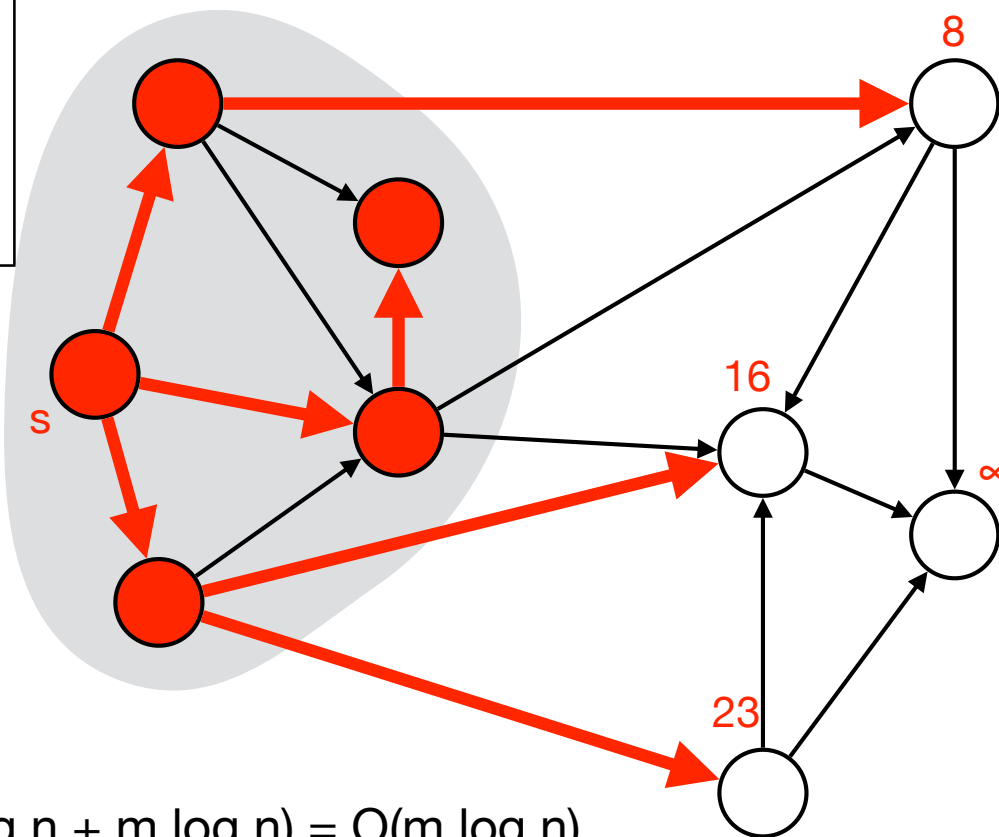
- **Implementation.** Vedligehold knuder med afstandsestimat i **prioritetskø**.
 - **Nøgle** af knude $v = v.d$.
 - I hvert skridt:
 - Find knude u med mindste afstandstandestimat = EXTRACT-MIN
 - Afspænd kanter som u peger på med DECREASE-KEY.



Dijkstras algoritme

```
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
```



- Tid.
 - n EXTRACT-MIN
 - n INSERT
 - < m DECREASE-KEY
- Samlet tid med min-hob. $O(n \log n + n \log n + m \log n) = O(m \log n)$

Dijkstras algoritme

- **Theorem.** Dijkstra algoritme implementeret med en min-hob beregner korteste veje i $O(m \log n)$ tid.
- **Grådighed.** Dijkstras algoritme er eksempel på en **grådig** algoritme.

Dijkstras algoritme

- **Prioritetskøer og Dijkstra.** Kompleksitet af Dijkstras algoritme afhænger af prioritetskø:
 - n INSERT
 - n EXTRACT-MIN
 - $< m$ DECREASE-KEY

Prioritetskø	INSERT	EXTRACT-MIN	DECREASE-KEY	Total
tabel	$O(1)$	$O(n)$	$O(1)$	$O(n^2)$
binær hob	$O(\log n)$	$O(\log n)$	$O(\log n)$	$O(m \log n)$
Fibonacci hob	$O(1)^\dagger$	$O(\log n)^\dagger$	$O(1)^\dagger$	$O(m + n \log n)$

† = **amortiseret** køretid

Edsger W. Dijkstra



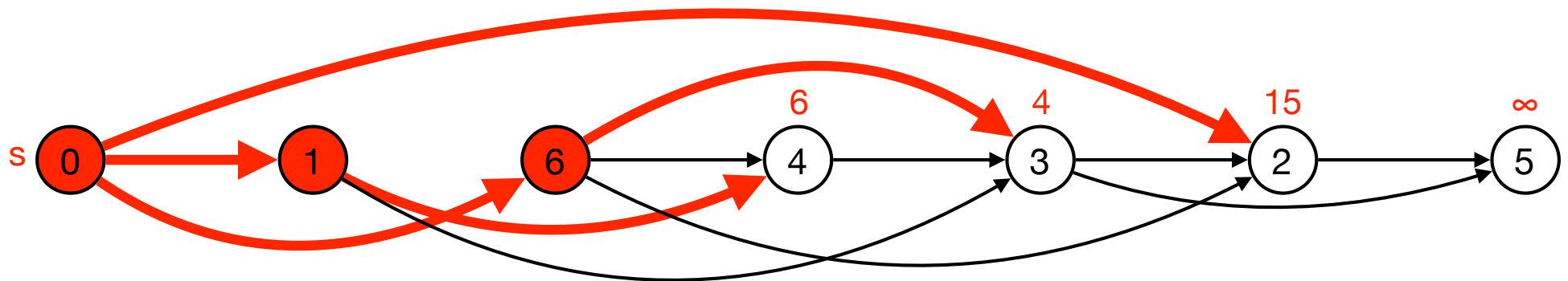
- Edsger Wybe Dijkstra (1930-2002)
- [Dijkstra algoritme](#). "A note on two problems in connexion with graphs". Numerische Mathematik 1, 1959.
- [Andre bidrag](#). Grundlæggende resultater i programmering, distribueret beregning, algoritmer, verifikation.
- [Citater](#). *"Object-oriented programming is an exceptionally bad idea which could only have originated in California."*
- *"The use of COBOL cripples the mind; its teaching should, therefore, be regarded as a criminal offence."*
- *"APL is a mistake, carried through to perfection. It is the language of the future for the programming techniques of the past: it creates a new generation of coding bums."*

Korteste veje

- Introduktion
- Egenskaber for korteste veje
- Dijkstras algoritme
- Korteste veje på DAGs

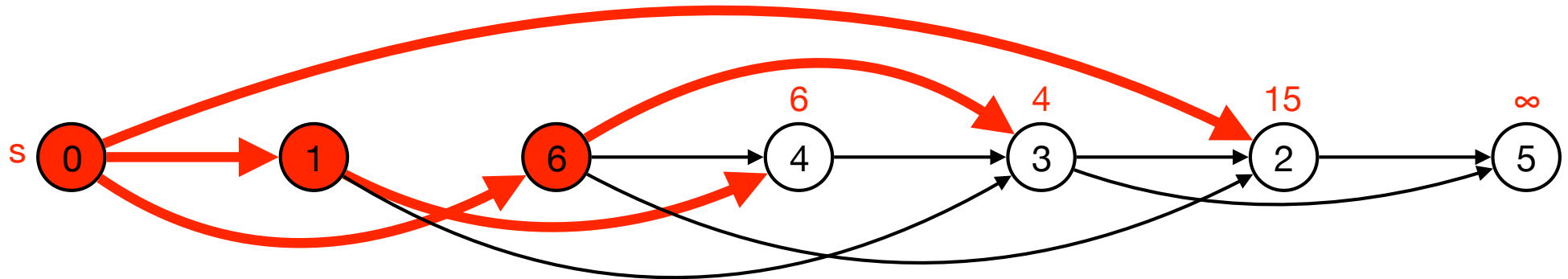
Korteste veje på DAGs

- **Udfordring.** Er det nemmere at beregne korteste veje på DAGs?
- **DAG korteste veje algoritme.**
 - Behandl knuder i topologisk orden.
 - For hver knude v , afspænd alle kanter fra v .
- Virker også for **negative** kanter.



Korteste veje på DAGs

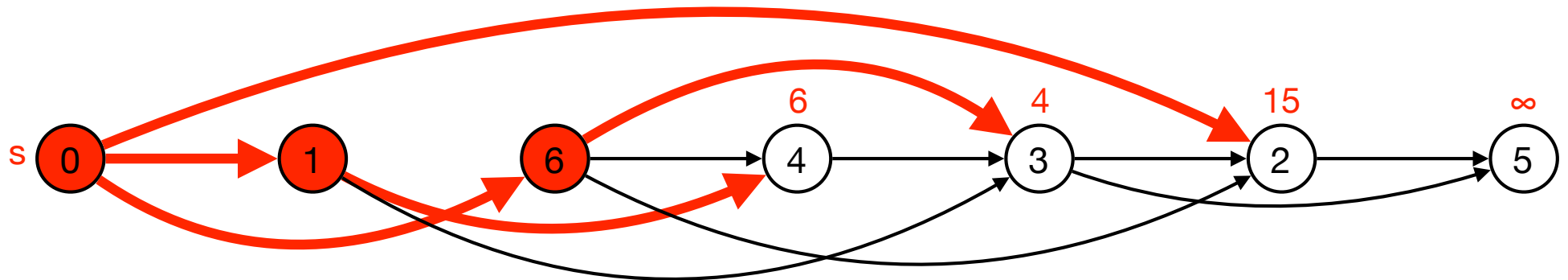
- **Lemma.** Algoritme beregner $v.d = \delta(s,v)$ for alle knuder v .



- **Bevis.** Induktion over topologisk sortering af knuder v_0, v_1, \dots, v_{n-1}
- **Basis.** $v_0.d = 0 = \delta(v_0, v_0)$.
- **Induktionsskridt.** Kig på v_i for $i > 0$. Antag $v_j.d = \delta(s, v_j)$ for $j < i$
 - $v_i.d = \min(v_j.d + w(v_j, v_i))$ hvor (v_j, v_i) er kant i G så $j < i$.
 - $= \min(\delta(s, v_j) + w(v_j, v_i))$ (pga. induktionshypotese)
 - $= \delta(s, v_i)$ (ingen veje til v_i via senere knuder i topologisk orden)

Korteste veje på DAGs

- **Implementation.**
 - Sorter knuder i topologisk rækkefølge.
 - Afspænd kanter ud af hver knude.
- **Samlet tid.** $O(m + n)$.



Korteste veje varianter

- Knuder.

- Enkelt kilde (*single-source*): fra s til alle andre knuder.
- Enkelt kilde, enkelt dræn (*single-source single-target*): fra s til t.
- Alle par (*all pairs*): mellem alle par af knuder.

- Begrænsninger på kantvægte.

- Ikke-negative vægte.
- Vilkårlige vægte.
- Euklidiske vægte.

- Kredse.

- Ingen kredse.
- Ingen negative kredse.

Korteste veje

- Introduktion
- Egenskaber for korteste veje
- Dijkstras algoritme
- Korteste veje på DAGs