Föreläsning 6
Datastrukturer (DAT037)

Fredrik Lindblad[1]

15 november 2017

[1] Slides skapade av Nils Anders Danielsson har använts som utgångspunkt. Se
http://www.cse.chalmers.se/edu/year/2015/course/DAT037

Innehåll

- Grafer – terminologi
- Grafer – datastrukturer
- Kortaste vägen – bredden först-sökning

Grafer

Grafer

Grafer kan representera:

- Nätverk.
- Beroenden.
- ...

Grafer

Givet en graf kan man ställa olika frågor:

- Nätverk.
  - Hur tar man sig från A till B? Snabbast? Billigast?
  - Vilken rutt har störst bandbredd?
- Beroenden.
  - Vad måste göras först?
- ...

Terminologi

Terminologi

Varning

Kan variera från författare till författare.

Terminologi

- Graf: $G = (V, E)$.
- V: Ändlig mängd av noder (vertex).
- E: Kanter/bågar (edge).
- Riktad graf: $E \subseteq V \times V$ (ordnade par av noder)
- Oriktad graf: $E \subseteq \{ U \subseteq V \mid 1 \leq |U| \leq 2 \}$ (oordnade par av noder)
- Viktad graf: $E \subseteq V \times V \times W$ eller $E \subseteq \{ U \subseteq V \mid 1 \leq |U| \leq 2 \} \times W$ (där W kan vara $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$, ...).
- I en *multigraf* kan det finnas flera kanter från u till v.

Terminologi

- Direkta efterföljare till u: $\{\, v \mid (u, v) \in E \,\}$.
- Direkta föregångare till v: $\{\, u \mid (u, v) \in E \,\}$.
- Ingrad: Antalet direkta föregångare.
- Utgrad: Antalet direkta efterföljare.

Begreppen definieras på motsvarande sätt för oriktade grafer/multigrafer/viktade grafer.

Terminologi

- Väg (path): $v_1 \to v_2 \to \ldots \to v_n$.
- Längd: $n - 1$.
- Vägar kan ha längd 0.
- Enkel väg: Alla noder distinkta (utom möjligtvis v_1 och v_n).
- Loop: Kant från nod till sig själv.

Terminologi

För riktade grafer:

- Cykel: Väg av längd ≥ 1 från v till v.
- Enkel cykel: Cykel som är enkel väg.
- (Riktad) acyklisk graf/DAG: Graf utan cykler.

För oriktade grafer:

- (Enkel) cykel:
  Enkel väg av längd ≥ 3 från v till v.

Terminologi

För oriktade grafer:

- Sammanhängande:
  Finns väg från varje nod till varje annan nod.

För riktade grafer:

- Starkt sammanhängande:
  Finns väg från varje nod till varje annan nod.
- Svagt sammanhängande:
  Finns väg från varje nod till varje annan nod,
  om man även får följa kanter baklänges.

Terminologi

Komplett graf:

- Oriktad.
- Inga loopar, inte multigraf.
- I övrigt så många kanter som möjligt.
- Antal bågar:

Terminologi

Komplett graf:

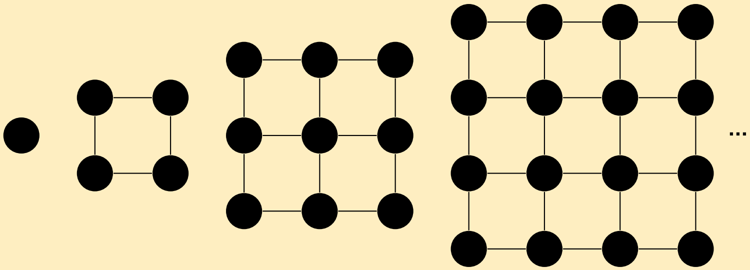
- Oriktad.
- Inga loopar, inte multigraf.
- I övrigt så många kanter som möjligt.
- Antal bågar:
  $|E| = |V|(|V| - 1)/2$

För oriktade, loop-fria grafer i allmänhet gäller
$|E| \leq |V|(|V| - 1)/2$

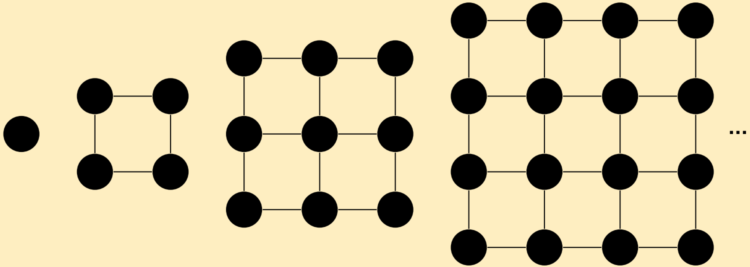
Terminologi

- Tät graf: Många kanter ($|E| \sim |V|^2$).
- Gles graf: Få kanter ($|E| \lesssim |V|$).

Gäller $|E| = \Theta(|V|^2)$ för följande klass av grafer?



Gäller $|E| = \Theta(|V|^2)$ för följande klass av grafer?



Svar: Nej, $|E| = \Theta(|V|)$.

Data-
strukturer

Grannmatriser

- Kvadratisk matris med $|V|^2$ element.
- Elementen kan t ex vara `true`/`false`.
- Tar stor plats om grafen är gles.
- Gå igenom en nods direkta efterföljare: $\Theta(|V|)$.
- Gå igenom alla noders direkta efterföljare: $\Theta(|V|^2)$.
- Avgöra om det finns en kant från u till v: $\Theta(1)$.
- Antagande ovan: Känner till noders index.
- Kan använda avbildning (map): nod $\mapsto$ index.

Grannlistor

En variant:

- Array av storlek $|V|$...
- ...innehållandes listor med direkta efterföljare.
- Ibland också listor med direkta föregångare.
- Gå igenom en nods direkta efterföljare: $O(V)$ (och $O(|E|)$).
- Gå igenom alla noders direkta efterföljare: $\Theta(|V| + |E|)$.
- Avgöra om det finns en kant från u till v: $O(V)$ (och $O(|E|)$).

Hur stor plats tar en array med grannlistor? (Anta att etiketter/vikter tar liten plats.)

- $\Theta(|V|)$.
- $\Theta(|E|)$.
- $\Theta(|V| + |E|)$.
- $\Theta(|V|^2)$.

Hur stor plats tar en array med grannlistor? (Anta att etiketter/vikter tar liten plats.)

- $\Theta(|V|)$.
- $\Theta(|E|)$.
- $\Theta(|V| + |E|)$.
- $\Theta(|V|^2)$.

Svar: $\Theta(|V| + |E|)$.

Kortaste vägen

Kortaste vägen

Kostnad av väg $v_1, ..., v_n$:

$$\sum_{i=1}^{n-1} c_{i,i+1}$$

där $c_{j,k}$ är vikten på kanten från nod j till nod k.
I oviktad graf: $n - 1$.

Kortaste vägen

Kortaste vägen-problem:

- Givet två noder u och v,
  hitta en kortaste väg från u till v.
- Givet nod u, för varje nod v,
  hitta en kortaste väg från u till v.
- Hitta kortaste vägen
  från varje nod till varje annan.

Visar sig naturligt att lösa den mittersta varianten ovan.

I algoritmen på nästa slide:

- `d` lagrar kortaste kända vägen till alla noder.
- `p` lagrar föregående nod för kortaste vägen.
- `q` är en kö som bestämmer besöksordningen av noder.

Oviktade grafer: bredden först-sökning

```
d = new array of size |V|, initialised to ∞
p = new array of size |V|, initialised to null
q = new empty queue

q.enqueue(s)
d[s] = 0

while q is non-empty do
  v = q.dequeue()
  for each direct successor v' of v do
    if d[v'] = ∞ then
      d[v'] = d[v] + 1
      p[v'] = v
      q.enqueue(v')

return (d, p)
```

Oviktade grafer: bredden först-sökning

```
d = new array of size |V|, initialised to ∞     O(|V|)
p = new array of size |V|, initialised to null  O(|V|)
q = new empty queue

q.enqueue(s)
d[s] = 0

while q is non-empty do                       O(|V|) ggr
  v = q.dequeue()
  for each direct successor v' of v do        O(|E|) ggr
    if d[v'] = ∞ then
      d[v'] = d[v] + 1
      p[v'] = v
      q.enqueue(v')

return (d, p)                            Totalt O(|V| + |E|)
```