

Grafteori og Nettverksvitenskap

ELMED219: Momentliste N01–N10

ELMED219

Vår 2026

1 Grunnleggende grafteori

- N01: Hva er en graf? (Noder og kanter)
- N02: Rettet vs. ikke-rettet graf
- N03: Vektet vs. ikke-vektet graf
- N04: Nabomatrise (adjacency matrix)

2 Sentralitetsmål

- N05: Degree Centrality (grad-sentralitet)
- N06: Betweenness Centrality (mellomliggenhet)
- N07: Eigenvector Centrality (egenvektor-sentralitet)
- N08: Clustering Coefficient (klyngekoeffisient)

3 Community detection og verktøy

- N09: Community Detection og Louvain-algoritmen
- N10: NetworkX-biblioteket for Python

N01: Hva er en graf? (Noder og kanter)

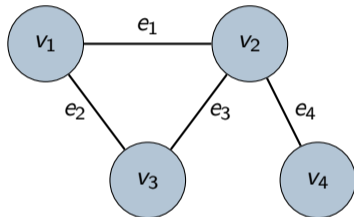
Definisjon

En **graf** $G = (V, E)$ består av:

- V : mengde av **noder** (vertices)
- E : mengde av **kanter** (edges)

Medisinsk eksempel:

- Noder = Pasienter
- Kanter = Likhet mellom pasienter

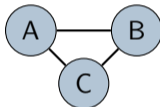


$$V = \{v_1, v_2, v_3, v_4\}$$
$$E = \{e_1, e_2, e_3, e_4\}$$

N02: Rettet vs. ikke-rettet graf

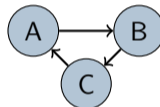
↔ Ikke-rettet graf:

- Kantene har ingen retning
- $A - B$ betyr gjensidig forbindelse
- Eksempel: Pasient-likhetsnettverk (PSN)



→ Rettet graf (digraf):

- Kantene har retning
- $A \rightarrow B \neq B \rightarrow A$
- Eksempel: Gen-regulatoriske nettverk



Rettet asyklisk graf (DAG)

En rettet graf **uten sykler** (ingen sti $A \rightarrow \dots \rightarrow A$). Brukes i kausal inferens, Bayesianske nettverk, og arbeidsflytsystemer.

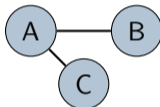
Medisinsk kontekst

PSN (pasient-likhetsnettverk) er vanligvis **ikke-rettet**: likhet mellom pasient A og B er symmetrisk.

N03: Vektet vs. ikke-vektet graf

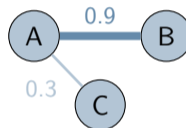
Ikke-vektet graf:

- Alle kanter er "like"
- Binær: forbindelse eller ikke



Vektet graf:

- Kanter har numeriske vektor
- Representerer styrke/avstand/likhet



I PSN (pasient-likhetsnettverk)

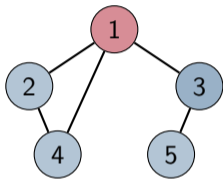
Vekten representerer **likhet (similaritet)** mellom pasienter, beregnet fra kliniske variabler.

Høyere vekt (w) = mer like pasienter (lignende symptomer, biomarkører, forløp, BMI, samme kjønn, ...).

Lavere vekt = mindre like pasienter.

Ofte brukes en **terskelverdi** for å filtrere bort svake koblinger (f.eks. $w < 0.5$).

N04: Nabomatrise (adjacency matrix)



deg: $1 \rightarrow 3$, $2 \rightarrow 2$, $3 \rightarrow 2$, $4 \rightarrow 2$, $5 \rightarrow 1$

Nabomatrise A:

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- $A_{ij} = 1$ hvis kant mellom node i og j
- $A_{ij} = 0$ ellers
- Symmetrisk for ikke-rettede grafer

Vektet graf

I vektet graf: $A_{ij} = w_{ij}$ (vekten av kanten)

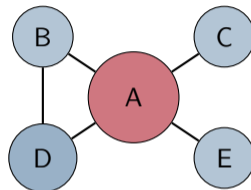
N05: Degree Centrality (grad-sentralitet)

Definisjon

Degree = antall kanter til en node

$$C_D(v) = \frac{\deg(v)}{n - 1}$$

(normalisert: delt på maks mulige naboer)



Tolkning:

- Høy degree = mange forbindelser
- “Hub” i nettverket
- Enkel å beregne

Node	deg	C_D (norm.)
A	4	1.00
B	2	0.50
C	1	0.25
D	2	0.50
E	1	0.25

Medisinsk relevans

I PSN: Pasienter med høy grad-sentralitet ligner på mange andre → “typiske” pasienter.

N06: Betweenness Centrality (mellomliggenhet)

Definisjon

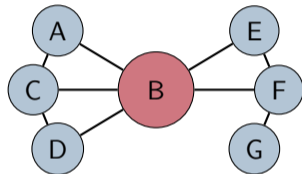
Andel korteste stier som går gjennom en node

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

σ_{st} = antall korteste stier fra s til t

Tolkning:

- Høy betweenness = “bro” mellom grupper
- Kontrollerer informasjonsflyt
- Kritisk for nettverkets struktur



Node	C_B	Node	C_B
A	0.0	E	0.0
B	15.0	F	2.0
C	1.0	G	0.0
D	0.0		

Node B er bro mellom to clusters

N07: Eigenvector Centrality (eigenvektor-sentralitet)

Idé

En node er viktig hvis den er koblet til **andre viktige noder**

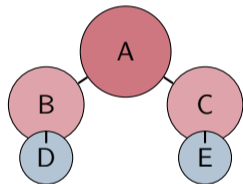
Beregning:

$$x_i = \frac{1}{\lambda} \sum_{j \in N(i)} x_j$$

Matriseform: $Ax = \lambda x$ (eigenvektor-ligning)

Forskjell fra degree:

- Degree: teller bare antall naboer
- Eigenvector: vekter naboer etter deres viktighet



A har høyest eigenvector centrality

Kjent anvendelse

Google PageRank er en variant av eigenvector centrality!

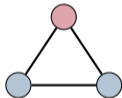
N08: Clustering Coefficient (klyngekoeffisient)

Definisjon

Hvor mye naboene til en node er koblet til hverandre

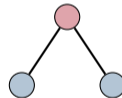
$$C_i = \frac{\text{antall kanter mellom naboer}}{\text{maks mulige kanter mellom naboer}}$$

Høy clustering ($C = 1$):



“Naboene kjenner hverandre”

Lav clustering ($C = 0$):



“Naboene kjenner ikke hverandre”

Medisinsk relevans

Høy clustering i PSN kan indikere tette pasientsubgrupper (fenotyper).

Community Detection

Identifisere **grupper** (communities) av tett koblede noder

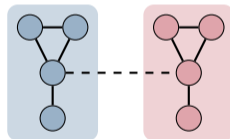
Louvain-algoritmen:

- 1 Start: hver node = egen gruppe
- 2 Flytt noder til nabogrupper som øker **modularitet**
- 3 Gjenta til ingen forbedring
- 4 Aggreger og repeter hierarkisk

Modularitet Q :

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

i, j =noder, A_{ij} =nabomatrise, m =kanter, k_i =grad av node i ,
 c_i =gruppe til node i , $\delta=1$ hvis $c_i = c_j$. $Q > 0.3$: god struktur.



To identifiserte communities

N10: NetworkX-biblioteket for Python

NetworkX

Python-bibliotek for å opprette, manipulere og analysere grafer

Vanlige operasjoner:

```
import networkx as nx
```

```
# Opprett graf
```

```
G = nx.Graph()
```

```
G.add_edge('A', 'B', weight=0.8)
```

```
G.add_edge('B', 'C', weight=0.5)
```

```
# Sentralitetsmål
```

```
nx.degree_centrality(G)
```

```
nx.betweenness_centrality(G)
```

```
nx.eigenvector_centrality(G)
```

```
nx.clustering(G)
```

```
# Community detection
```

```
from community import louvain
```

```
partition = louvain.best_partition(G)
```

```
# Visualisering
```

```
nx.draw(G, with_labels=True)
```

```
# Fra nabomatrise (numpy)
```

```
G = nx.from_numpy_array(A)
```

```
# Grafegenskaper
```

```
G.number_of_nodes()
```

```
G.number_of_edges()
```

```
nx.density(G)
```

Oppsummering: N01–N10

Grafteori-grunnlag:

- Graf $G = (V, E)$: noder (V) + kanter (E)
- Rettet vs. ikke-rettet
- Vektet vs. ikke-vektet
- Nabomatrise-representasjon

Sentralitetsmål:

- Degree: antall forbindelser
- Betweenness: bro-rolle
- Eigenvector: viktige naboer

Strukturanalyse:

- Clustering coefficient: lokal tetthet
- Community detection: finne grupper
- Louvain-algoritme: effektiv metode

Verktøy:

- [NetworkX](#) (Python)
- Integrasjon med [NumPy/Pandas](#)

Neste steg

Bruk disse konseptene til å bygge **pasient-likhetsnettverk** (PSN)!