

# Grafteori og Nettverksvitenskap

## ELMED219: Momentliste N01–N10

ELMED219

Vår 2026

- 1 Grunnleggende grafteori
- 2 Sentralitetsmål
- 3 Community detection

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

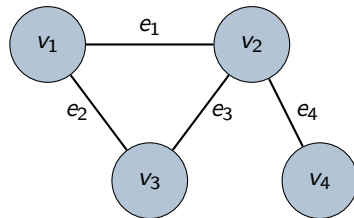
## Definisjon

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

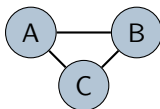


$$V = \{v_1, v_2, v_3, v_4\}$$
$$E = \{e_1, e_2, e_3, e_4\}$$

## N02: Rettet vs. urettet graf

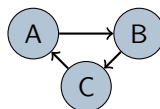
### ↔ Urettet graf:

- Kantene har ingen retning
- $A - B$  betyr gjensidig forbindelse
- Eksempel: Vennskap



### → Rettet graf (digraf):

- Kantene har retning
- $A \rightarrow B \neq B \rightarrow A$
- Eksempel: Twitter-følgere



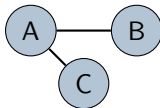
### Medisinsk kontekst

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

## N03: Vektet vs. uvektet graf

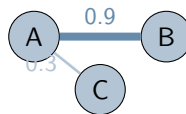
### Uvektet graf:

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



### Vektet graf:

- Kanter har numeriske vektor
- Representerer styrke/avstand

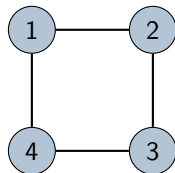


### I PSN

Vekten representerer **likhet** mellom pasienter.  
Høyere vekt = mer like pasienter.

## N04: Nabomatrise (adjacency matrix)

**Nabomatrise  $A$ :**



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

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

### Vektet graf

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

## N05: Degree Centrality (grad-sentralitet)

### Definisjon

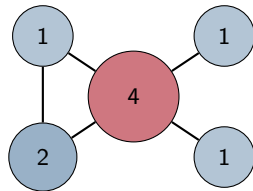
**Degree** = antall kanter til en node

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

(normalisert: delt på maks mulige naboer)

### Tolkning:

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



*Node A har degree 4 (hub)*

### Medisinsk relevans

I PSN: Pasienter med høy degree ligner på mange andre → “typiske” pasienter

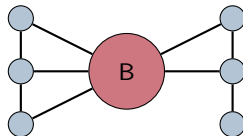
## N06: Betweenness Centrality (mellomliggenhet)

### Definisjon

Andel korteste stier som går gjennom en node

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

$\sigma_{st}$  = antall korteste stier fra  $s$  til  $t$



*Node B er bro mellom to clusters*

### Tolkning:

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



# N07: Eigenvector Centrality

## Idé

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

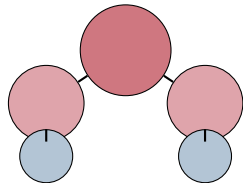
### Beregning:

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

Eller i matriseform:  $Ax = \lambda x$

### Forskjell fra degree:

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



*A har høyest eigenvector centrality*

## Kjent anvendelse

Google PageRank er en variant av eigenvector centrality!

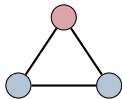
## N08: Clustering Coefficient (klyngekoeffisient)

### Definisjon

Hvor mye naboene til en node er koblet til hverandre

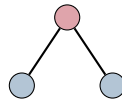
$$C_i = \frac{\text{antall kanter mellom naboer}}{\text{maks mulige kanter mellom naboer}}$$

**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

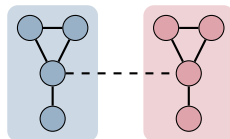
# N09: Community Detection og Louvain-algoritmen

## Community Detection

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

### Louvain-algoritmen:

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



### Modularitet $Q$ :

- Måler kvalitet på inndeling
- $Q > 0.3$ : 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)
```

```
# Sentralitet
```

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

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

```
# Community detection
```

```
from community import louvain
```

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

```
# Visualisering
```

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

```
# Fra nabomatrise
```

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

## Ressurser

Dokumentasjon: <https://networkx.org/documentation/>

# Oppsummering: N01–N10

## Grafteori-grunnlag:

- Graf = noder + kanter
- Rettet vs. urettet
- Vektet vs. uvektet
- 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)!