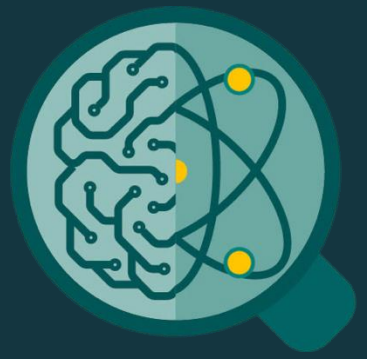
Graph Analytics: A Tale of Investors Product journey



Danial Senejohnny, Data Scientist & Analytics Translator

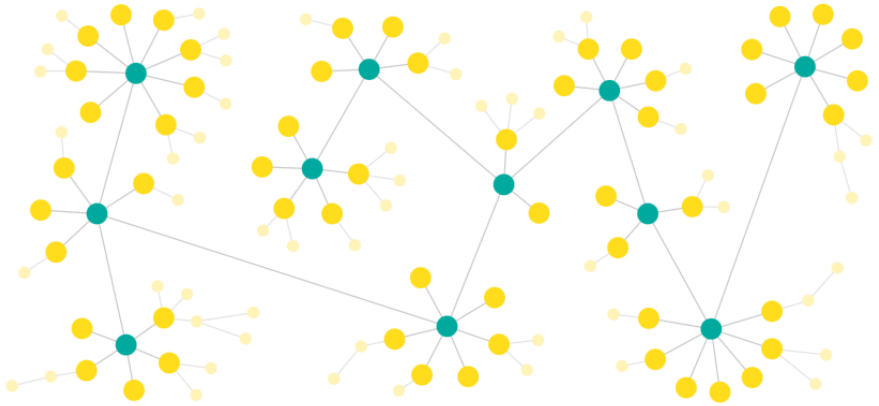
Knowledge Sharing Lecture Notes, Sep & Nov 2023

Agenda

- Review of Graph/Network Analytics: Part 1
- Discussion: Potential Use Cases
- Graph/Network Analytics: Part 2

Application of Graph Analytics

- Customer or product Journey
- Detecting Fraud and Money Laundering Patterns
- Personalized Recommendations
- Patient Referrals to specialist in healthcare
- Transportation logistics route network
- Cybersecurity threat detection



- Many domains where Networks is in the heart:
  - District heating
  - Electricity
  - Communication/Internet
  - ….

Self-directed Plus

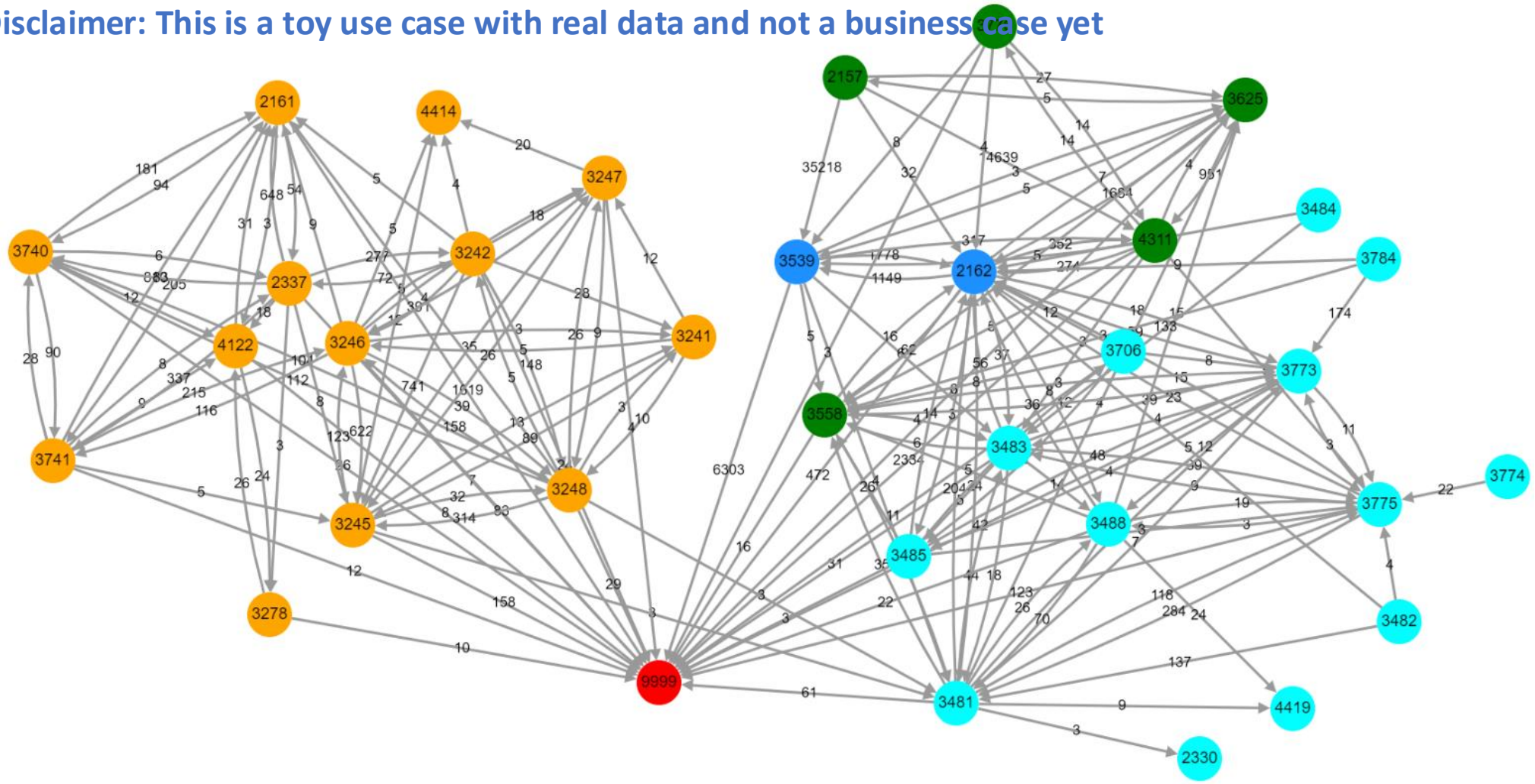
DPM

Advisory

Self-directed Rest

Investors Product Journey

- At contract level all the product id's that **wealth** and **C&A** can take sequentially
- Filtered for links larger and equal 3
- **Disclaimer: This is a toy use case with real data and not a business case yet**



Graph Analytics Frameworks

Descriptive

NetworkX
Network Analysis in Python

pgmpy

Predictive

KARATE CLUB

CD·LIB

Big Data Base

neo4j

Spark GraphX

Visualization

Cytoscape

plotly Dash Cytoscape

NetworkX
Network Analysis in Python

Spark GraphX

neo4j

Graph Analytics Methods

Descriptive

- **Path Finding & Search:**
Used by other methods

- **Centrality:**
Captures node importance

- **Community Detection**
Overlapping & non-overlapping

Predictive

Markov Chain Model:

- Transition probability
- Reaching time & state
- Random Walk
- PageRank

Graph Learning

- Node Classification
- Link prediction
- Label Propagation

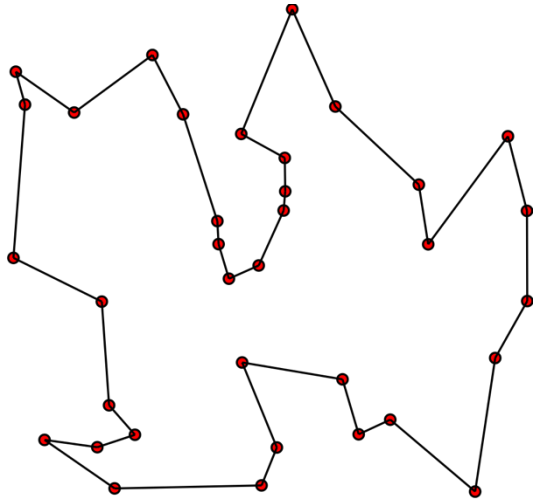
Embeddings:

- Node
- Edge
- Graph

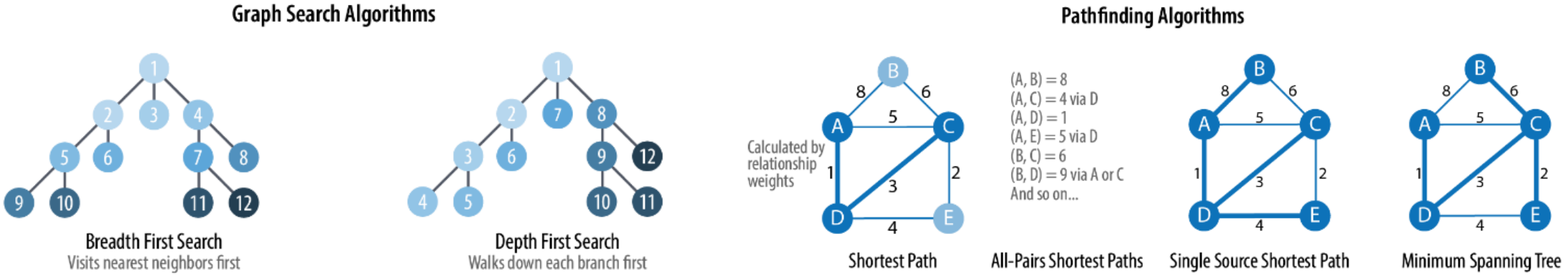
Community Detection

Path Finding & Graph Search

Travelling salesman problem:
"Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?“



Wikipedia



Mark Needham and Amy E. Hodler, Graph Algorithms

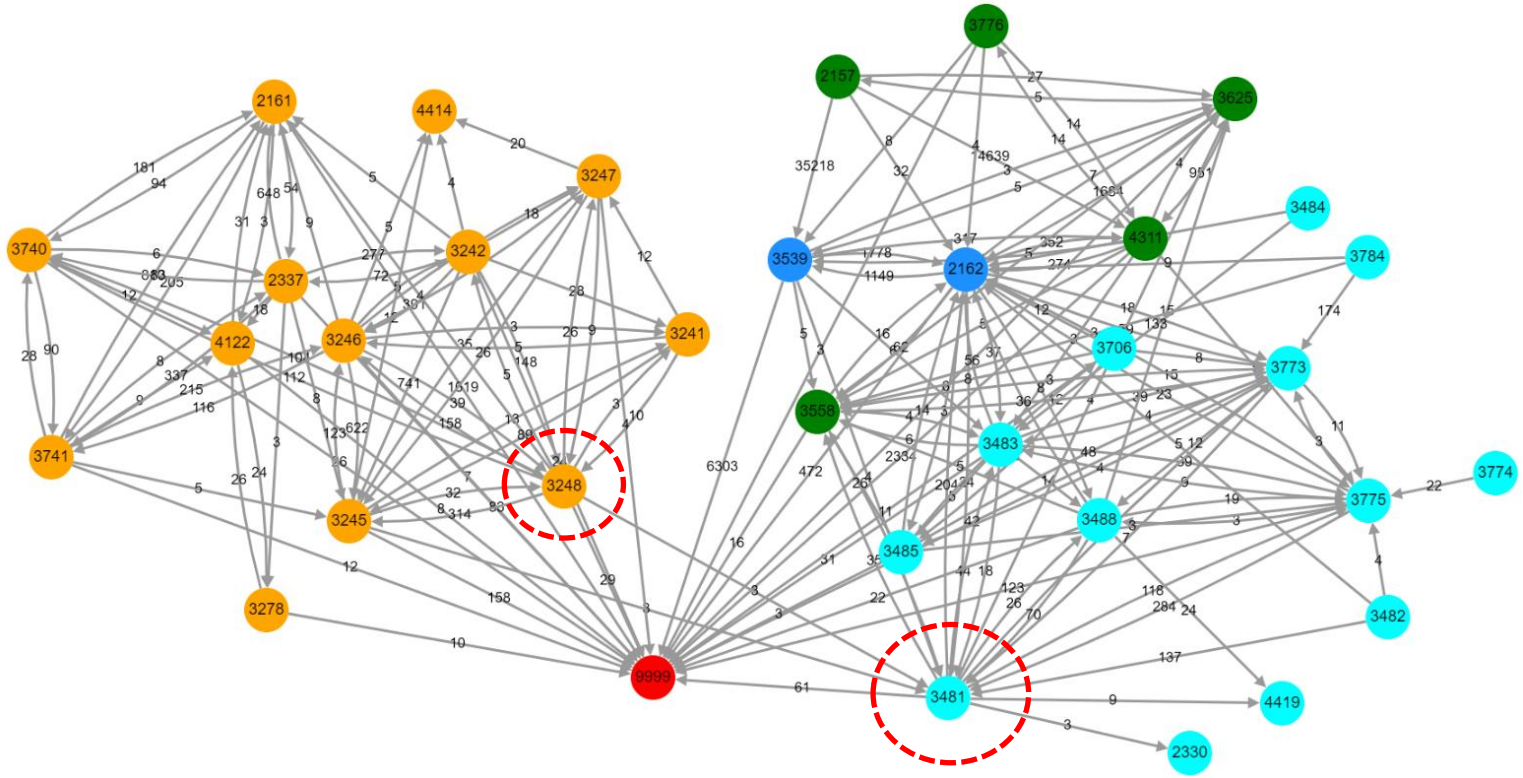
Self-directed Plus
DPM
Advisory
Self-directed Rest

Centrality: Degree

- A node is important given the fraction of nodes it is connected to.

Business Questions:

- Which products are popular among investors?



For example: most important product per concept are

DPM: 3246
Advisory: 3481

Self-directed Plus
DPM
Advisory
Self-directed Rest

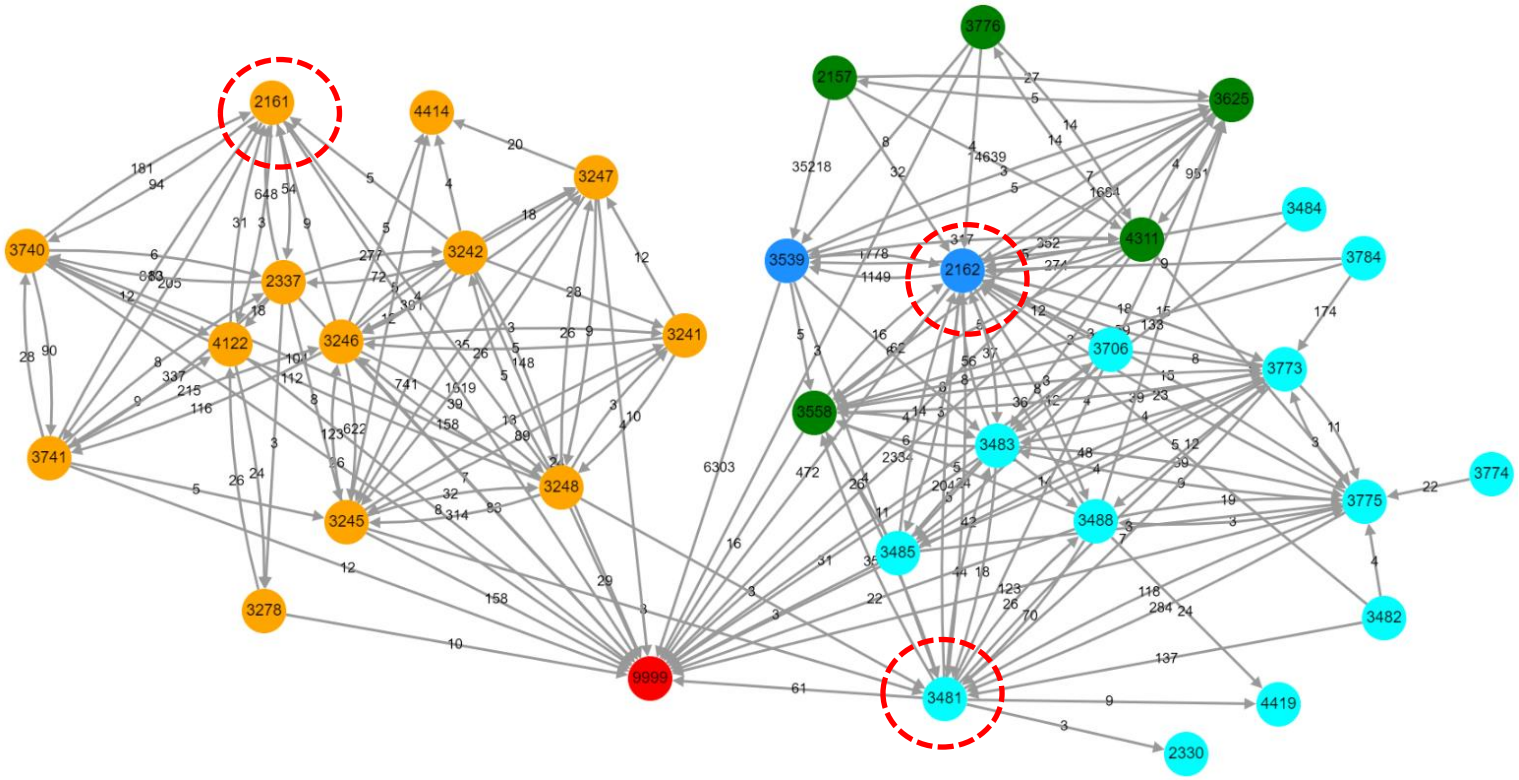
Centrality: Closeness

- A node is important if it has small **shortest path** length to many other nodes.
- Nodes that can most easily reach other graph nodes

$$C(u) = \frac{n-1}{\sum_{v=1}^{n-1} d(v,u)},$$

Business Questions:

- Which products facilitate the investors to move to other products?



Higher values of closeness indicate higher centrality.

All: 2162
DPM: 2161
Advisory: 3481

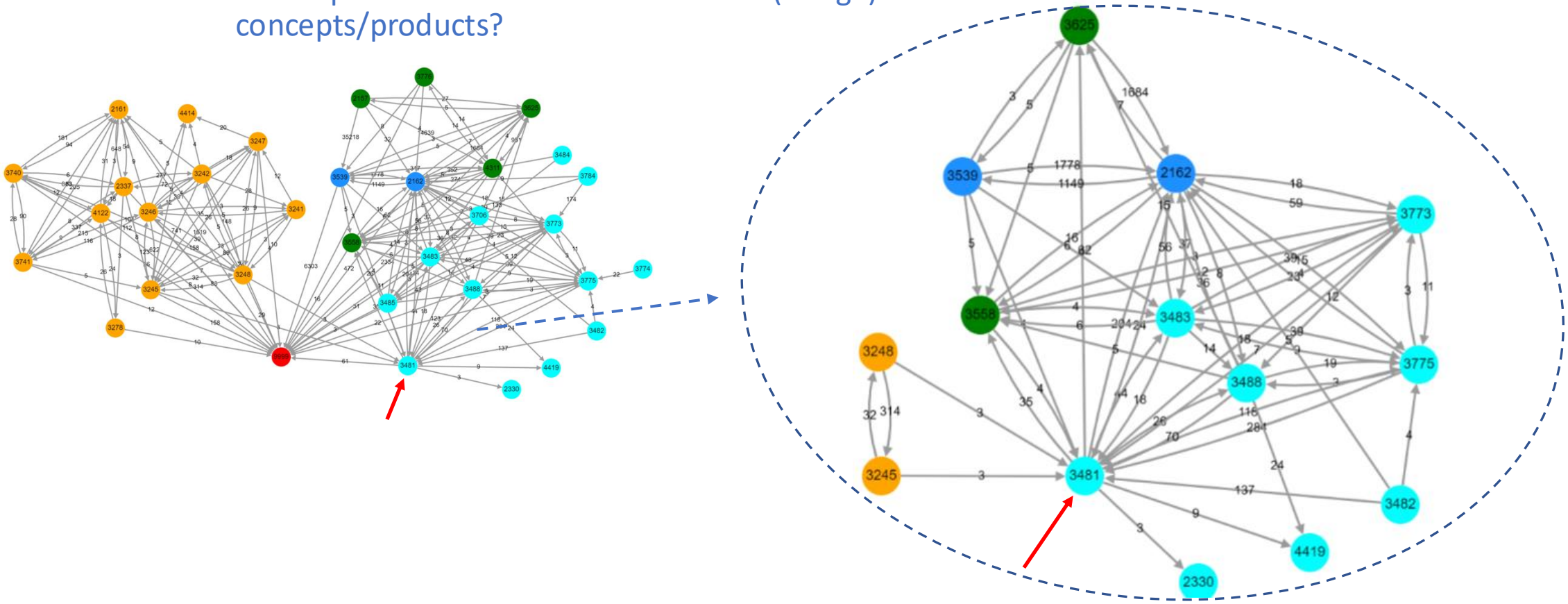
Self-directed Plus
DPM
Advisory
Self-directed Rest

Centrality: Betweenness

- A node is important if it lies on many **shortest paths** between pair of nodes.
- A node's betweenness is the sum of the fraction of all-pairs shortest paths that pass through it

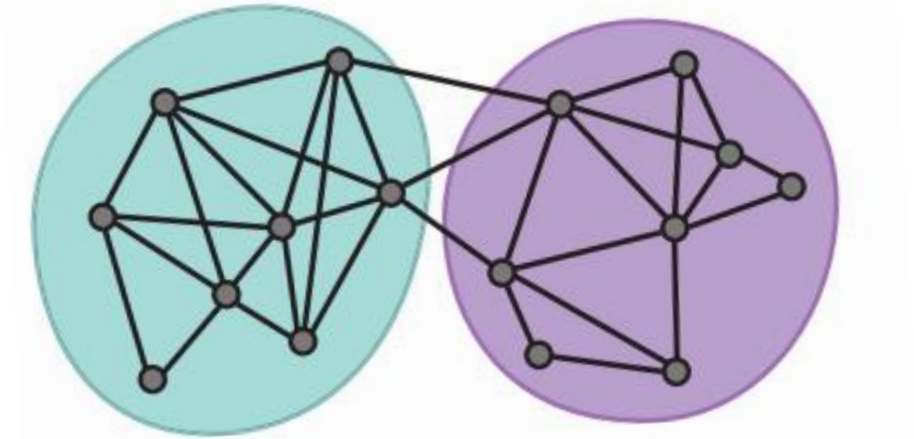
Business Questions:

- Which products has the most influence (bridge) for investors to move to different investment concepts/products?

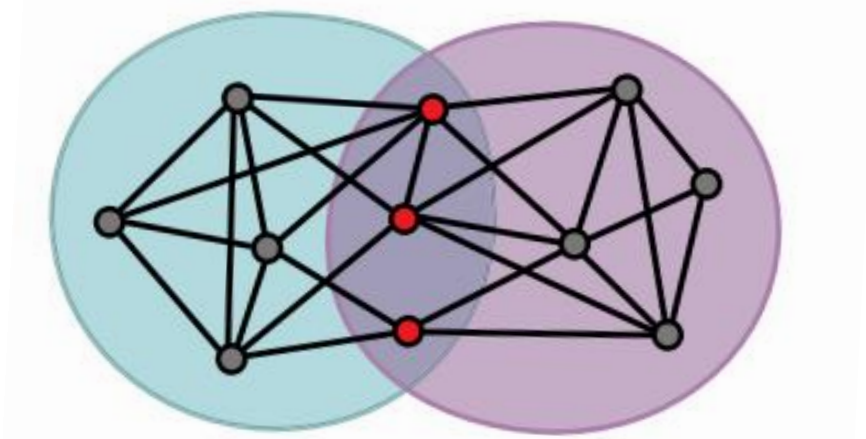


Community Detection

Non-overlapping Communities



Overlapping Communities



Overlapping Community Detection at Scale: A Nonnegative Matrix Factorization Approach

Modularity

- Identify Communities by **maximizing modularity**
- **Modularity:** measures the density of links inside communities compared to links between communities

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

- A_{ij}: edge weight between node *i, j*
- k_i & k_j*:* sum of the edge weights attached to node *i, j*
- m*:* sum of all graph edge weights
- c_i & c_j: community in which node *i, j* belong
- δ: indicator function, $\delta(c_i, c_j) = 1$ if $c_i = c_j$ else 0

$\boldsymbol{Q} \in [0.3, 0.7]$ means graph has significant community structure

Community Detection [Louvain Method]

Louvain Algorithm is a de facto method for community detection:

- Fast & scalable, addresses weighted graphs, hierarchical communities
- Greedy: making locally optimal choices at each step -> eventually global optimal solution

The Algorithm passes/iteration:

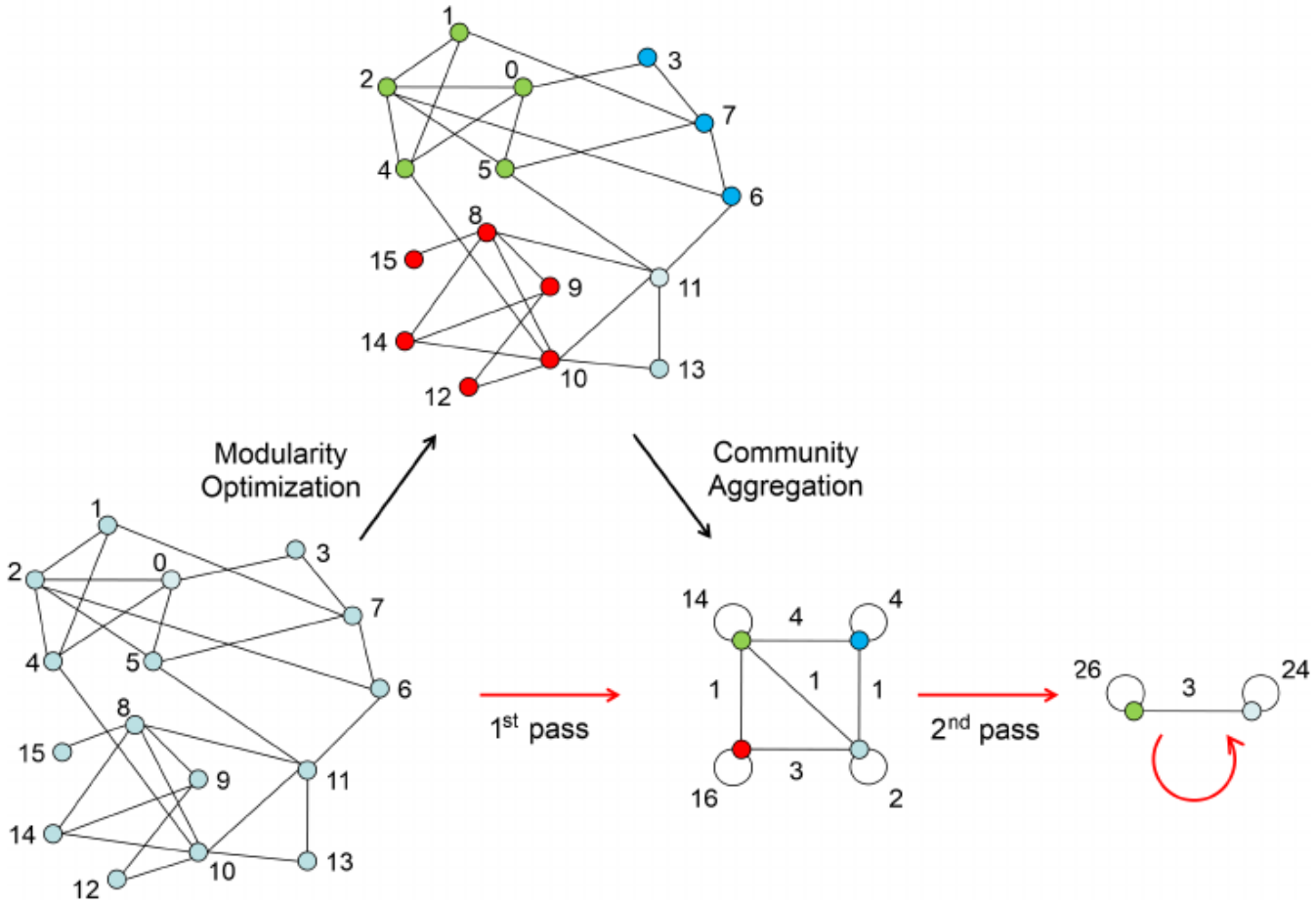
Phase 1:

- Put each node in a community
- Compute $\boldsymbol{\Delta Q}$ when putting node *i* into the community of some neighbor *j*
- Move node *i* to a community of node *j* with maximum $\boldsymbol{\Delta Q}$ gain
- Repeat above steps and stop when no individual move yield modularity gain

Phase 2:

- Communities obtained in **Phase 1** are absorbed into super nodes and make a super graph
- Go to **Phase 1**

Louvain Algorithm [Example]

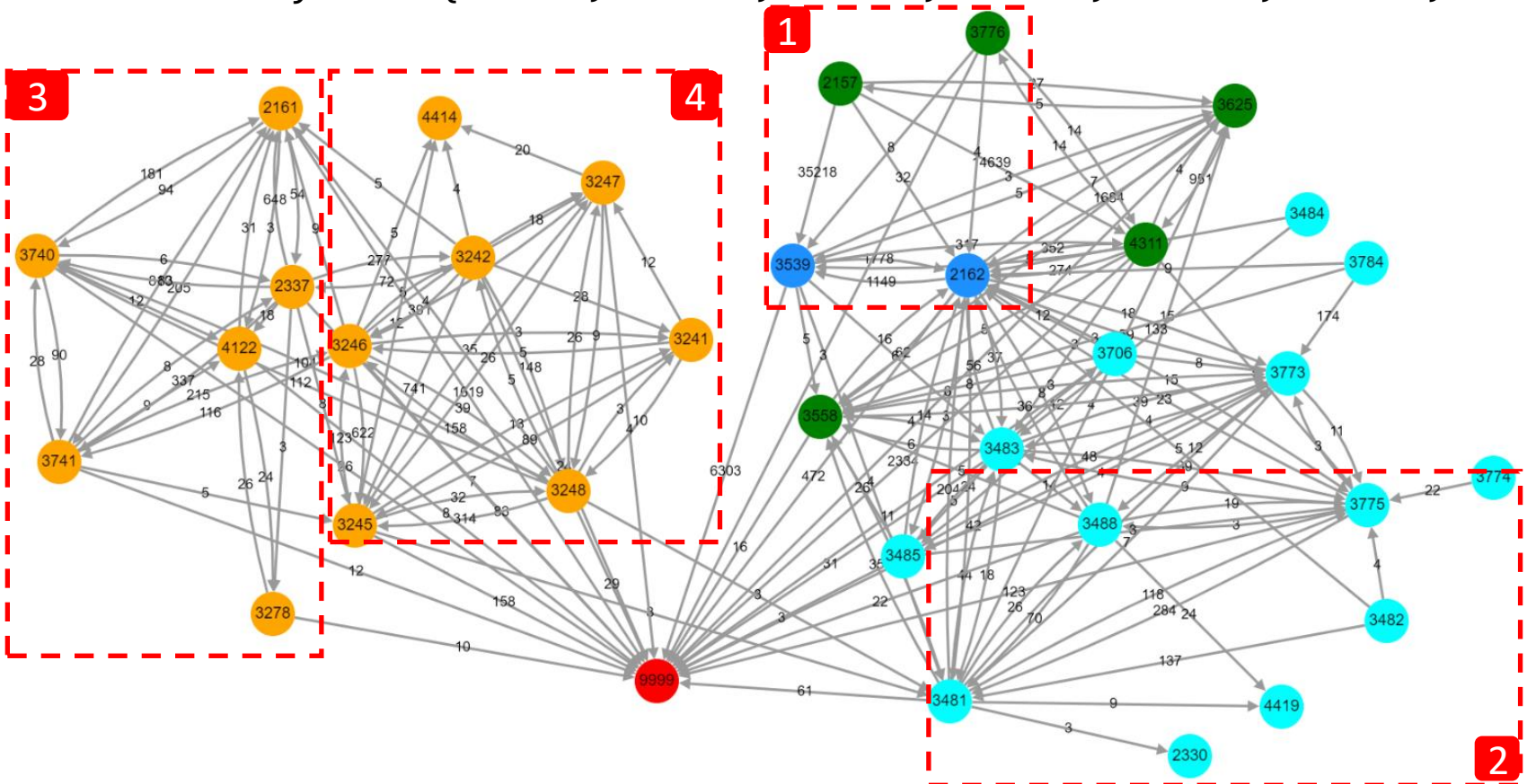


V. Blondel, et al, Fast unfolding of communities in large networks, 2008

Self-directed Plus
DPM
Advisory
Self-directed Rest

Louvain Algorithm Example

```
Community 1 -> {'2157', '3539', '3776', '4311’}
Community 2 -> {'2330', '3481', '3482', '3488', '3774', '3775', '4419’}
Community 3 -> {'2161', '2337', '3278', '3740', '3741', '4122’},
Community 4 -> {'3241', '3242', '3245', '3246', '3247', '3248', '4414'}
```



Modularity Score: 0.57

Community Detection [Leiden Method]

From Louvain to Leiden: guaranteeing well-connected communities

V. A. Traag, L. Waltman & N. J. van Eck

Value proposition

“The Louvain algorithm may yield arbitrarily badly connected communities. In the worst case, communities may even be disconnected, especially when running the algorithm iteratively”

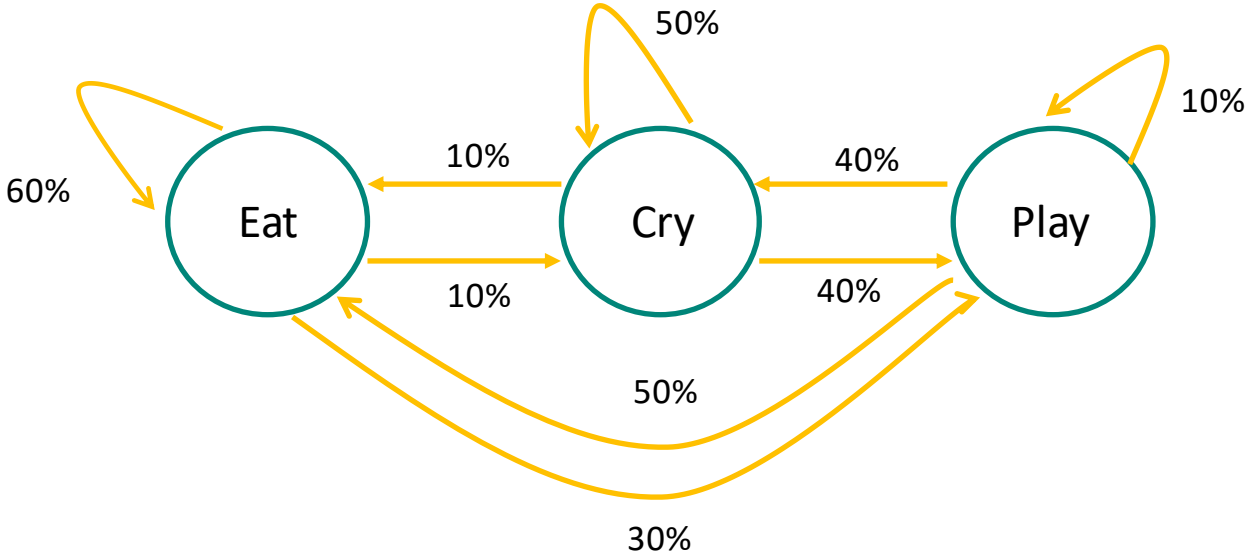
Markov Chain

Markov property

$$\mathbb{P}(X_{t+1} = s \,|\, X_t = s_t, X_{t-1} = s_{t-1}, \ldots, X_0 = s_0) \;=\; \mathbb{P}(X_{t+1} = s \,|\, X_t = s_t),$$

for all $t = 1, 2, 3, \ldots$ and for all states $s_0, s_1, \ldots, s_t, s$.

Example



Markov Chain [Transition Probability]

Markov Transition Probability Matrix

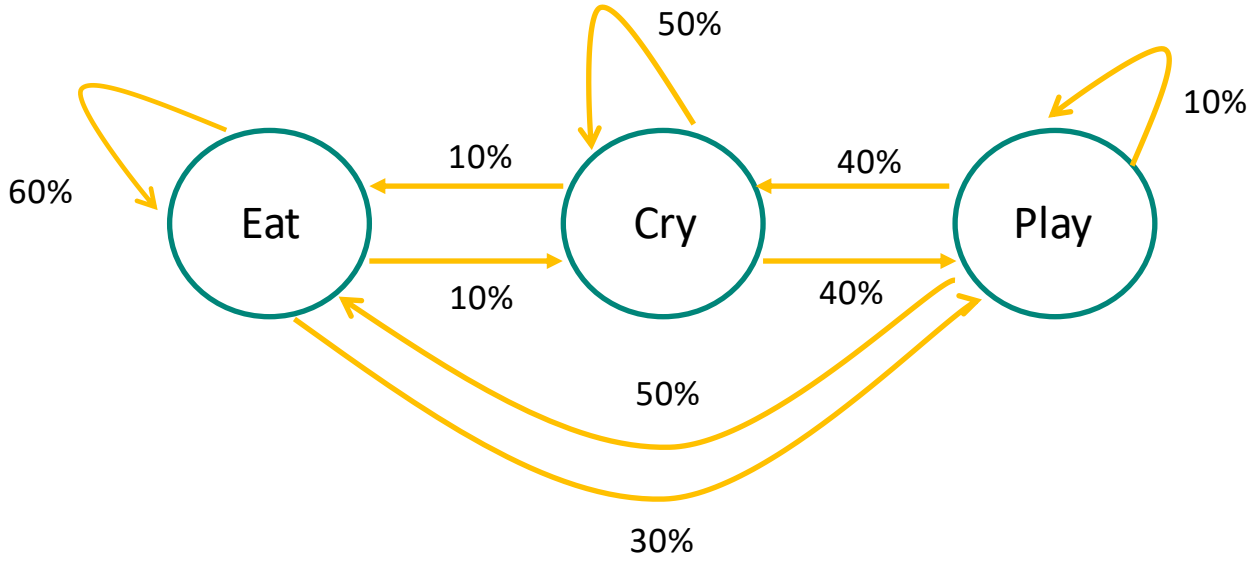
	Eat	Cry	Play
Eat	0.6	0.1	0.3
Cry	0.1	05	0.4
Play	0.5	0.4	0.1

Rows add to 1

Markov States:

$$X_t = [Eat \quad Cry \quad Play]$$

$$P = \begin{bmatrix} 0.6 & 0.1 & 0.3 \\ 0.1 & 05 & 0.4 \\ 0.5 & 0.4 & 0.1 \end{bmatrix}$$



Markov Chain [Transition Probability]

Initial probability distribution

1-step transition

2-step transition

…

t-step transition

$$
\begin{aligned}
X_0 &\sim \boldsymbol{\pi}^T \\
X_1 &\sim \boldsymbol{\pi}^T P \\
X_2 &\sim \boldsymbol{\pi}^T P^2 \\
&\vdots \\
X_t &\sim \boldsymbol{\pi}^T P^t.
\end{aligned}
$$

$$
\boldsymbol{\pi} = \begin{pmatrix} \pi_1 \\ \pi_2 \\ \vdots \\ \pi_N \end{pmatrix} = \begin{pmatrix} \mathbb{P}(X_0 = 1) \\ \mathbb{P}(X_0 = 2) \\ \vdots \\ \mathbb{P}(X_0 = N) \end{pmatrix}
$$

Discrete State Transition

$$P^T = M$$

$$X^T = p$$

$$
\begin{cases}
X_{t+1} = X_t\, P \\
p_{t+1} = M\, p_t
\end{cases}
$$

P: rows add to 1

X_t: horizontal vector $\quad X_t = [X_1 \quad \dots \quad X_n]$

M: columns add to 1

p_t: vertical vector $\quad p_t = \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix}$

Self-directed Plus
DPM
Advisory
Self-directed Rest

Markov Chain Model

State transitions in the next t-step, given transition probabilities

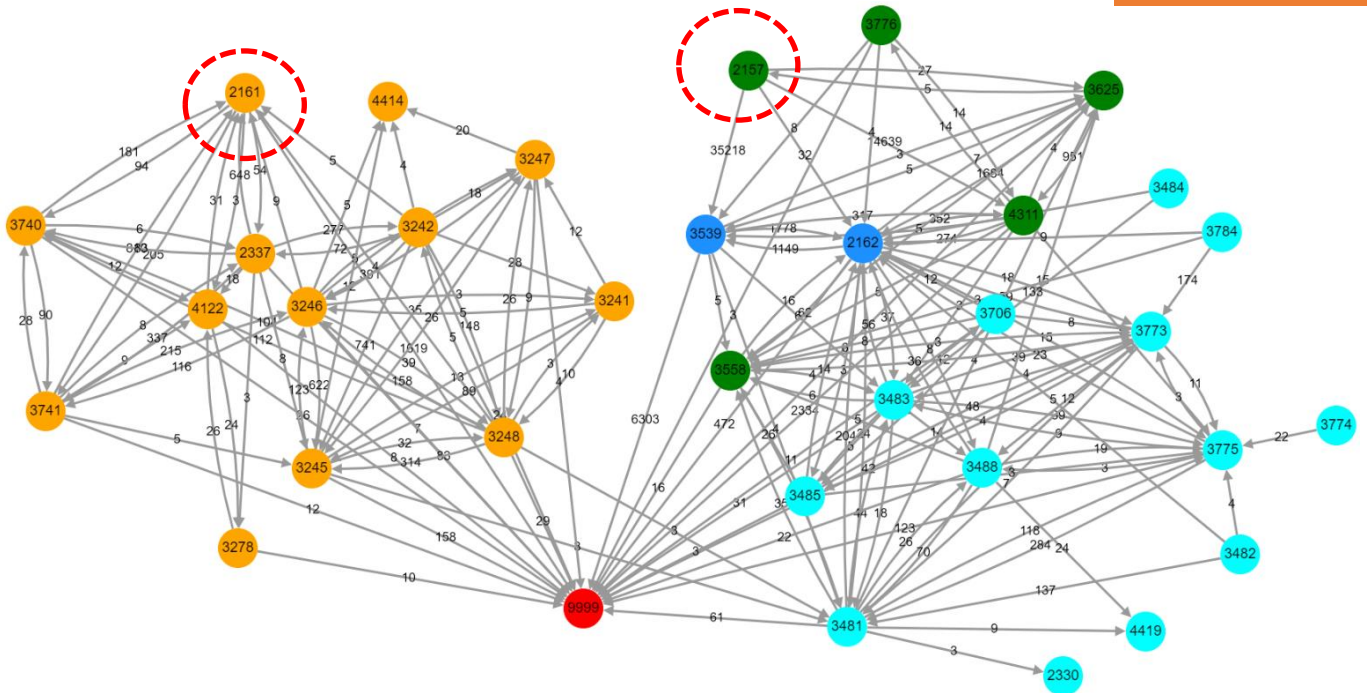
$$X_{t+1} = X_t\, P$$

Business Questions:

Given that a client using a product, what is the most probable next product(s)/move(s)?

Example:

- Product user 2157 next moves are 3539 [60%] & 4311 [25%]
- Product user 2161 next moves are churn [80%] & 3741 [10%]



Probability Transition Matrix is 40 x40

```
[[0.   0.71 0.29 ... 0.   0.   0.  ]
 [0.   0.   0.04 ... 0.   0.   0.  ]
 [0.   0.51 0.   ... 0.   0.   0.  ]
 ...
 [0.   1.   0.   ... 0.   0.   0.  ]
 [0.   0.   0.   ... 0.   0.   0.  ]
 [0.   1.   0.   ... 0.   0.   0.  ]]
```

Random Walk

Starting at a random node (product) in the graph what are the potential next 50/100 nodes (products) to visit

`'2330' -> '3481' -> '3482’ -> … -> … -> '3774’`

Using the random walk one can map the graph as a sentence

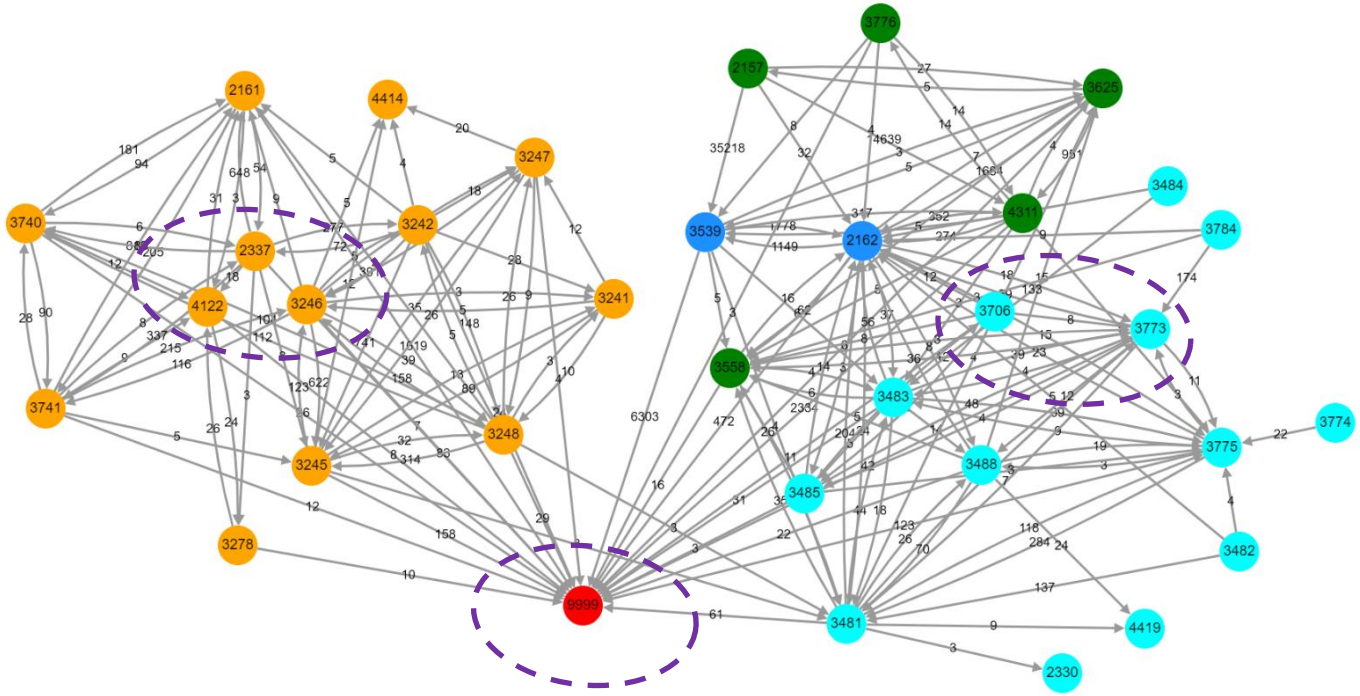
Word := node
Length of sentence := #Nodes

Markov Hitting Probability & Time

- Considering a desired set of nodes, what is the probability/time to reach that desired set from any node

Business Questions:

- Given that a client is using a product,
  - what is the probability that the client acquire a desired product/churn?
  - How many steps does it take the client to move to a desired product/churn?



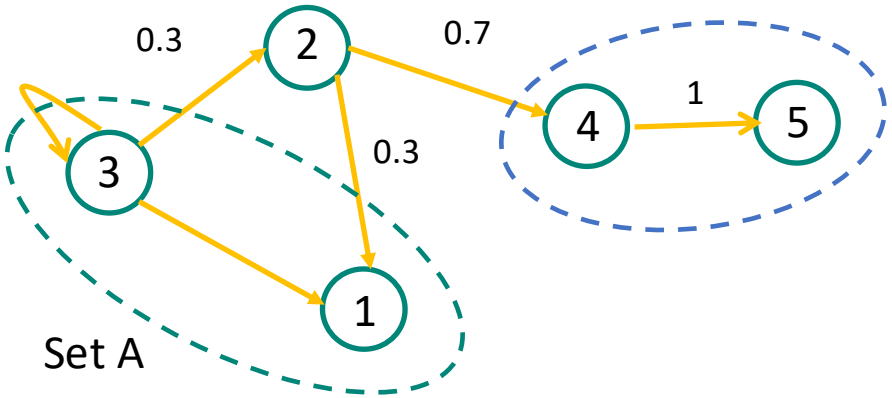
Markov Hitting Probability

Hitting/Reaching probability:

- Probability of ever reaching the set A, starting from any initial state.

$$h_{iA} = \begin{cases} 1 & \text{for} \quad i \in A, \\ \sum_{j \in S} p_{ij} h_{jA} & \text{for} \quad i \notin A. \end{cases}$$

$$\boldsymbol{h_A} = \begin{pmatrix} h_{1A} \\ h_{2A} \\ h_{3A} \\ h_{4A} \\ h_{5A} \end{pmatrix} = \begin{pmatrix} 1 \\ 0.3 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$



1

- Closed reaching set [B]: hitting probability = absorption probability.

Closed form hitting probabilities [special case]:

Let set A be, the set with absorbing nodes and B the rest of the nodes. Then

$$\begin{bmatrix} P_{\mathrm{AA}} & P_{\mathrm{AB}} \\ P_{\mathrm{BA}} & P_{BB} \end{bmatrix} = \begin{bmatrix} I & 0 \\ P_{\mathrm{BA}} & P_{BB} \end{bmatrix} \longrightarrow H = (I - P_{BB})^{-1} P_{BA}$$

Markov Hitting Time

- **Hitting/Reaching/Absorption time:**
- **Time:** means how many step [maybe translate each step to time unit for your use case]

$$m_{iA} = \begin{cases} 0 & \text{for} \quad i \in A, \\ 1 + \sum_{j \notin A} p_{ij} m_{jA} & \text{for} \quad i \notin A. \end{cases}$$

Closed form hitting probabilities [special case]:

Let set A be, the set with absorbing nodes and B the rest of the nodes. Then

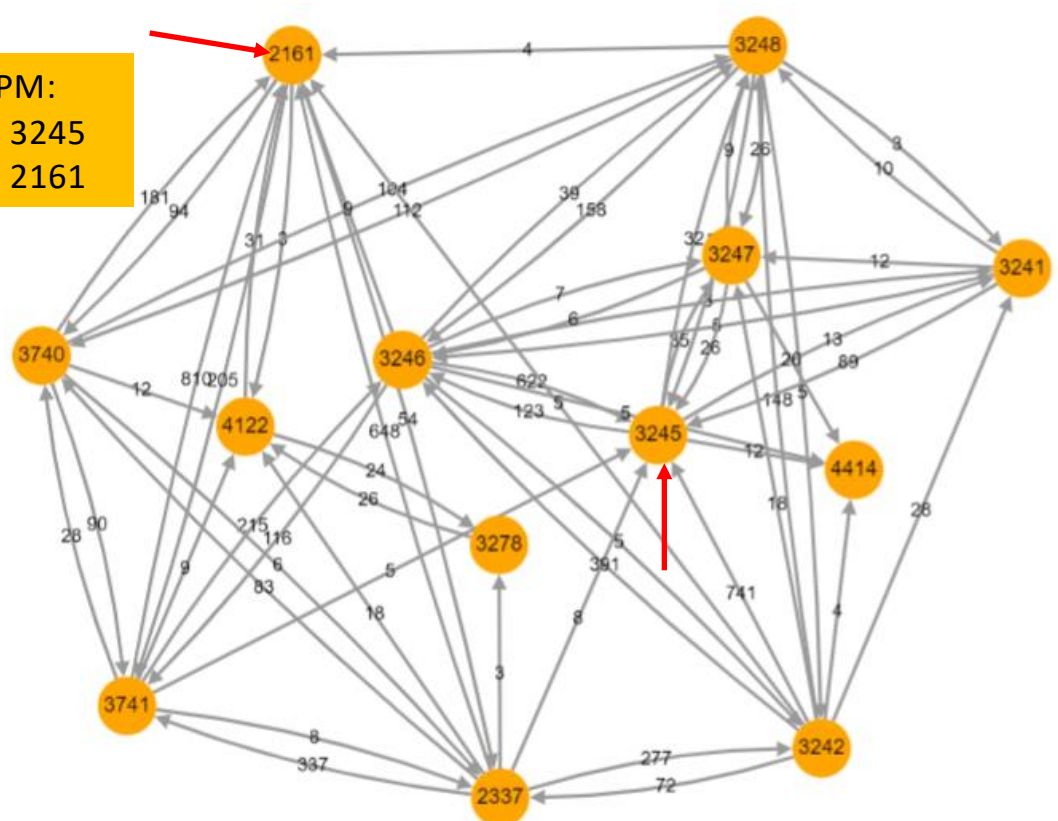
$$\begin{bmatrix} P_{\mathrm{AA}} & P_{\mathrm{AB}} \\ P_{\mathrm{BA}} & P_{BB} \end{bmatrix} = \begin{bmatrix} I & 0 \\ P_{\mathrm{BA}} & P_{BB} \end{bmatrix} \longrightarrow \quad T = (I - P_{BB})^{-1}\mathbf{1}$$

Self-directed Plus
DPM
Advisory
Self-directed Rest

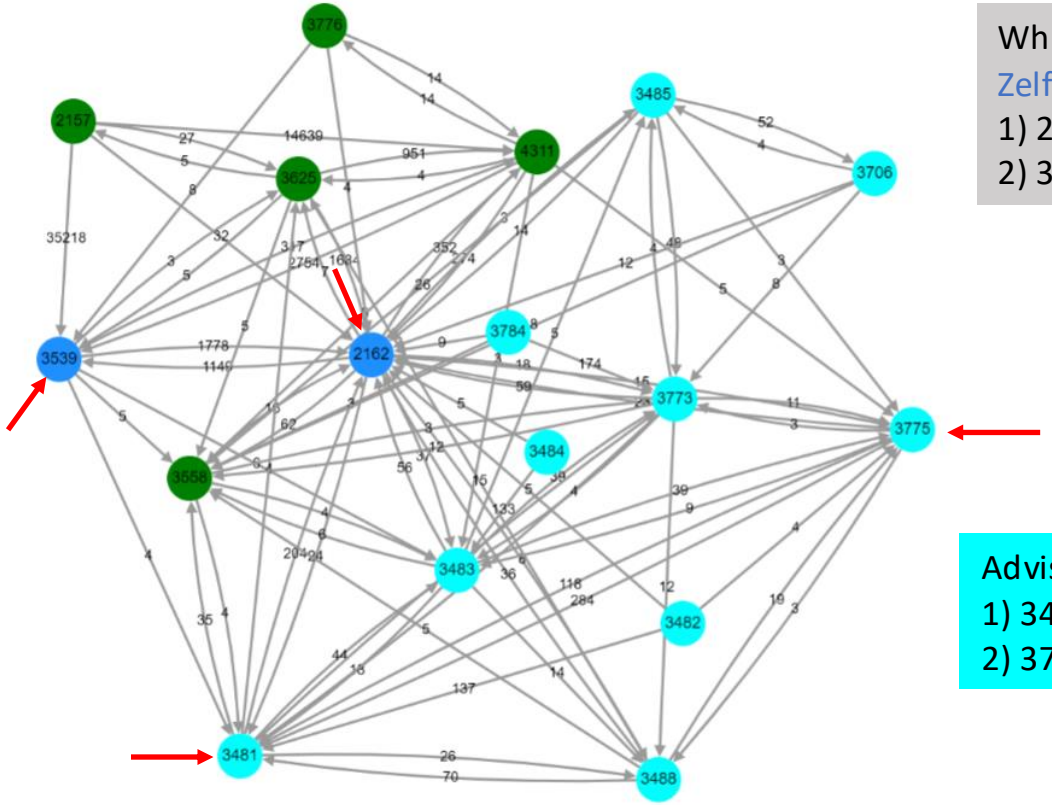
PageRank

- Nodes ranking based on the structure of the incoming links.
- A node that is linked to by many node with high rank receives a high rank itself.
- PageRank considers the (directional) influence of nodes neighbours and their neighbours.
- **Business Questions:**
  - Which alternative investment product can we recommend to the investors?

DPM:
1) 3245
2) 2161



Whole Graph:
Zelf Beleggen Plus
1) 2162
2) 3539



Advisory:
1) 3481
2) 3775

PageRank [Theory]

- p_t: Probability distribution that a surfer visits a page, with below random walk

$$p_{t+1} = M\, p_t$$

- PageRank is the stationary probability distribution of a random walk across pages

$$p_{t+1} := p_t = M\, p_t$$

- Stationary solution is the eigenvector corresponding to eigenvalue 1 of M

$$1 \times p^* = M\, p^*$$

PageRank [Numerical Solution]

- Power iteration finds the stationary probability distribution

Initial condition $p_0 = \begin{bmatrix} 1/n & \cdots & 1/n \end{bmatrix}^\top$

Iterator $p_{t+1} = G\, p_t$

Stop condition $|p_{t+1} - p_t| < \epsilon$

- where G is google matrix and n is number of nodes

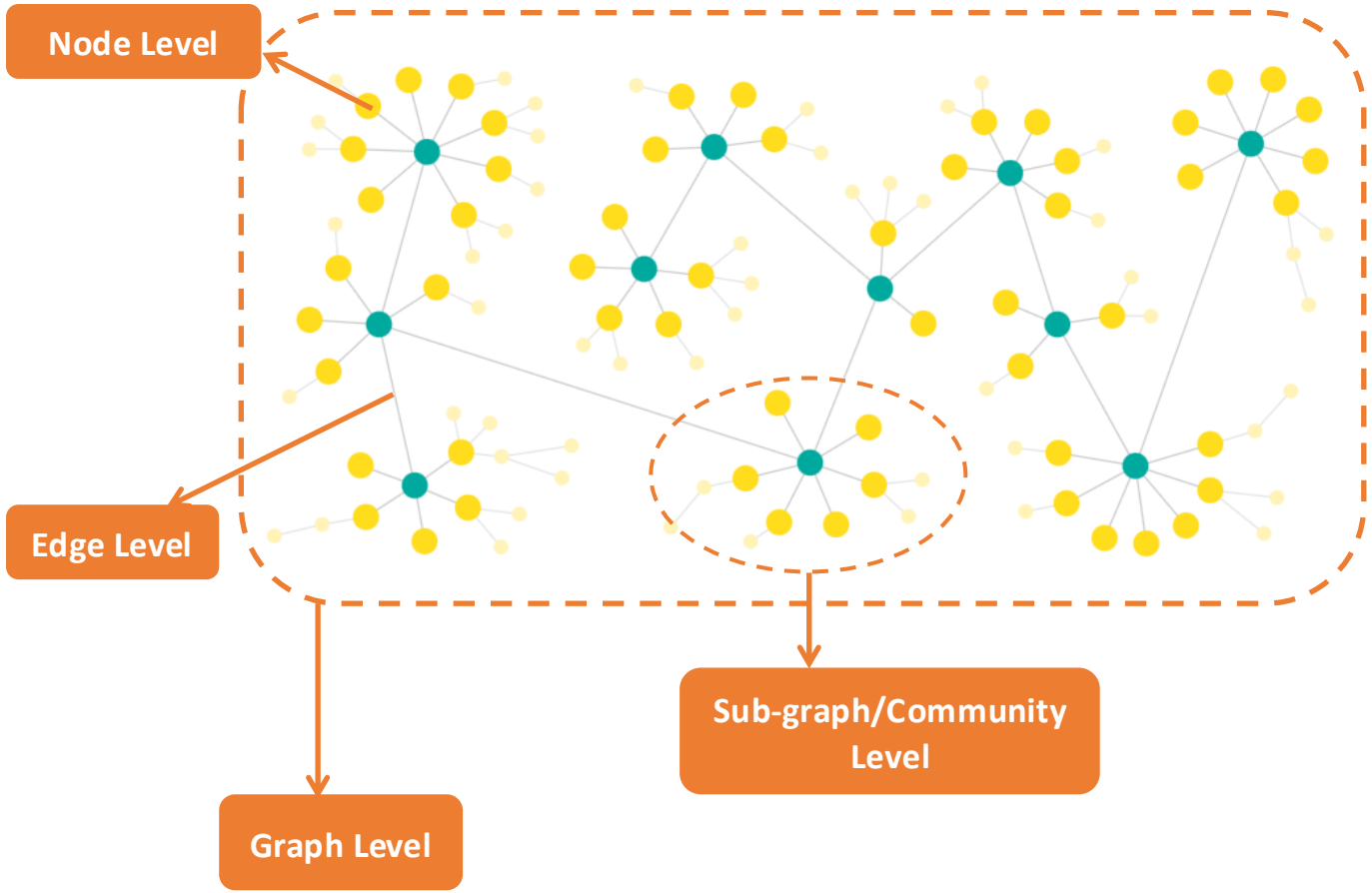
$$G = \beta\, M + (1-\beta) \left[\frac{1}{n}\right]_{n \times n}$$

$$\beta \in [0.8, 1]$$

Sporadic jumps to help random walk not to get stuck in dead end and traps

- Personalized PageRank allows customized jumps

Graph Learning Tasks



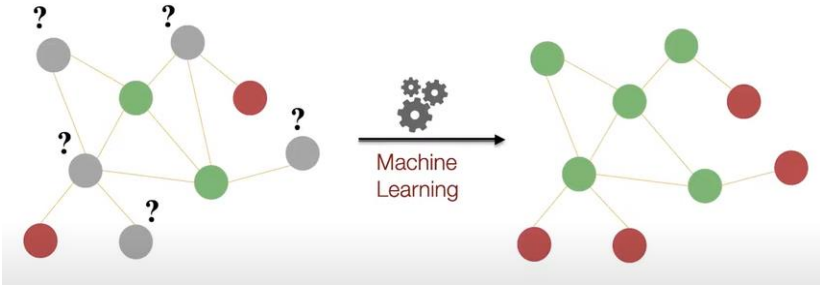
Node Level: [Node Classification]
Assign node type, using node features & graph structure
Application: fraud detection, opinion networks, churn

Edge Level: [Link Prediction]
Identify missing links,
Application: recommendation, fraud detection Knowledge graph completion

Sub-graph/Community Level: [Community Detection]
Assign group of node & edges to a community
Application: clustering

Graph Level: [Graph Clustering/labelling]
Categorize different graphs
Application: molecule property prediction

Node Classification/Clustering



Task: Predict unknown label given the nodes with known label

- **Business Questions:**
  - Which account/client in the transaction network is fraudulent?
  - Opinion/belief networks who votes to which presidential candidate

Classical Learning Approach:

1. Extract hand-crafted node-level structured features from the unstructured graph

Importance

Structure

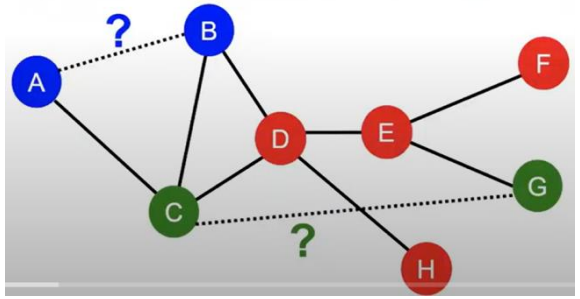
neighbors' connectivity

Count of different graphlets

Node	degree	Betweenness (c)	Closeness (c)	Eigenvector (c)	Clustering Coeff.	Graphlet Deg Vector	Label
3740	50	x	y	z	0.6	[2, 1, 3, 0, 0, 4]	DPM

2. fit a classical ML (RF, LR, …) classifier or clustering model

Edge Prediction



Task: Predict new edges given existing edges

- **Business Questions:**
  - Which new user/item to recommend to a new item/user?
  - Which profile do we suggest to someone just coming to use LinkedIn

Classical Learning Approach:

1. Extract edge-level structured features fro u

#Com. neighbors / #All neighbors

#Paths between node pair

Edge	Distances	Common Neighbor	Jaccard's Coeff.	Katz Index	exist
(2161, 3740)	1	7	11	3	1
(2161, 3485)	8	0	0	1	0

Local Neighborhood

Global Neighborhood

2. Training approaches

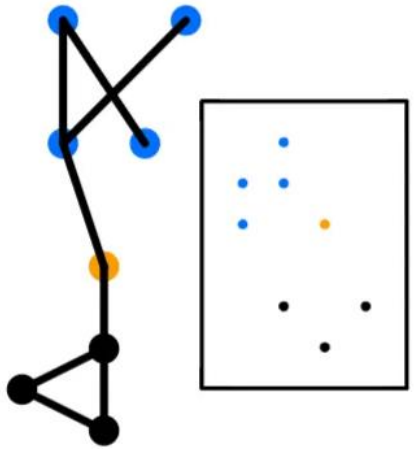
- **Link missing randomly:** train on existing edges and predict random missing links
- **Link over time:** train on existing edges and predict future coming edge [dynamic graphs like FB, LinkedIn, Twitter]

Embeddings in Graph Analytics

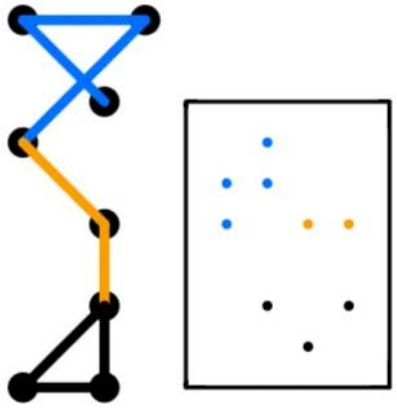
Embedding: capture key structured features while reducing dimensionality

- low dimensional vector representation of unstructured/semi-structured data [text, graph, time series, …].

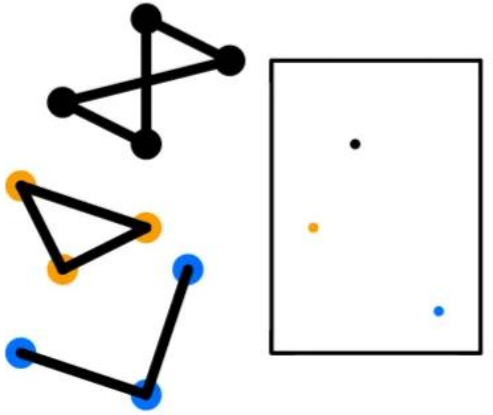
Node Embedding



Edge Embedding



Graph Embedding



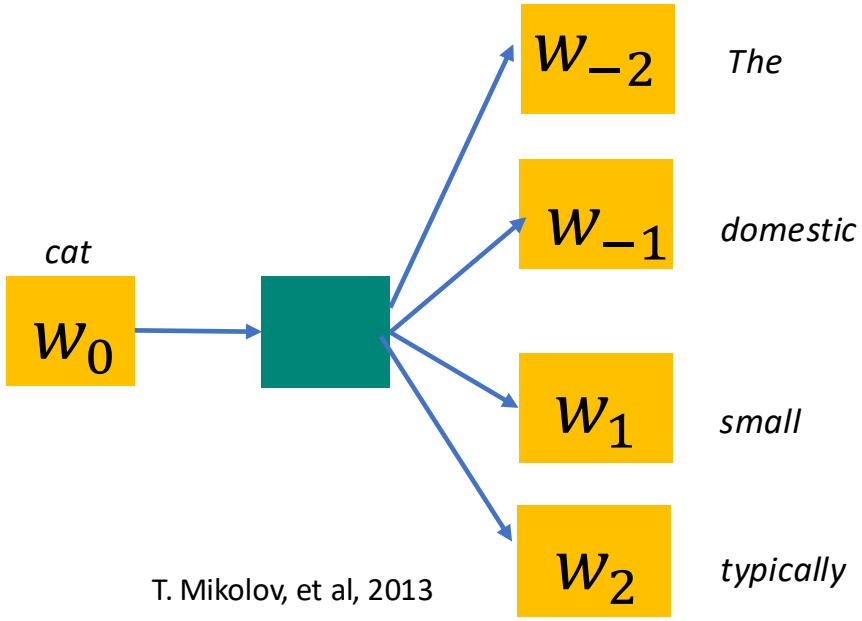
Images are provided by Vatsal, Toward Data Science, 2022

Word Embedding [Skip-gram]

- **Word2Vec: SkipGram**

For example:

The domestic **cat** is a small, typically furry carnivorous mammal

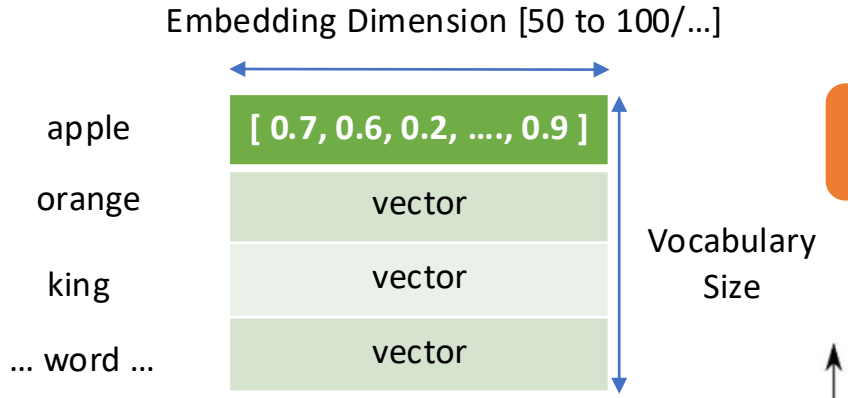


T. Mikolov, et al, 2013

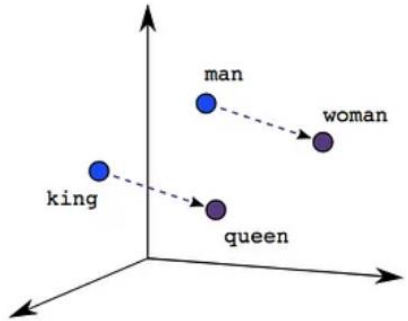
Classification with Neural Network

Word	Context	label
cat	small	1
cat	Furry	1
cat	car	0

Positive samples: target word & neighbours
Negative samples: randomly sampled words from lexicon



Embedding is NN weights as byproduct of a classification



Node Embedding

Deep (Random) Walk [2014]:

Training data:

start from each node run **short fixed-length** random walks, i.e., markov chain random walk

1) 3246 -> 3256 -> 4414-> … -> … -> 3775

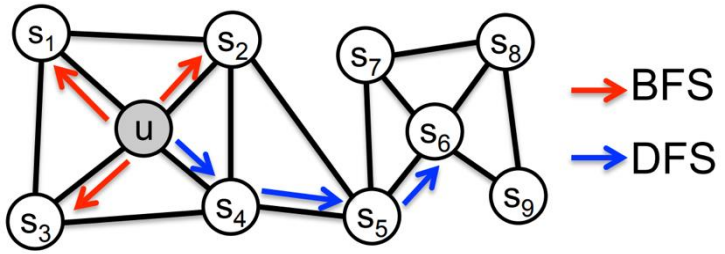
2) … -> … -> ….

Train a skip-gram mode, where nodes close in a graph are close in embedding space.

Limitation: constrained random walk

Node2Vec [2016]:

What other random walk strategy?



Grover & Leskovec, 2016

Walk strategies that can trade off between local & global views of the network.

Breath First Search: local/micro network view

Depth First Search: Global/macro network view

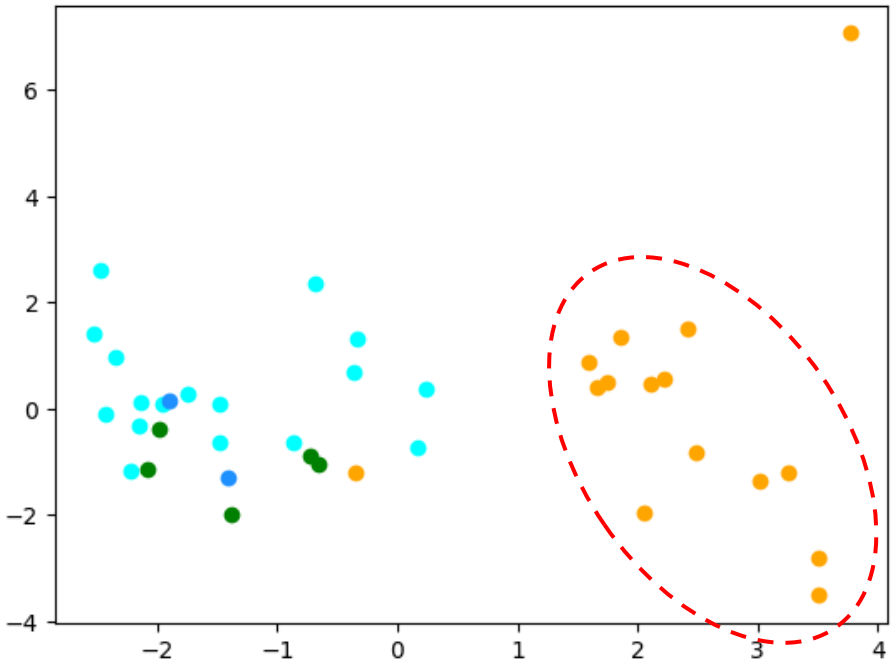
KARATE CLUB

Many other node embeddings: RandNE, Diff2Vec[18], NodeSketch[19], GLEE [20] …

Embedding Application

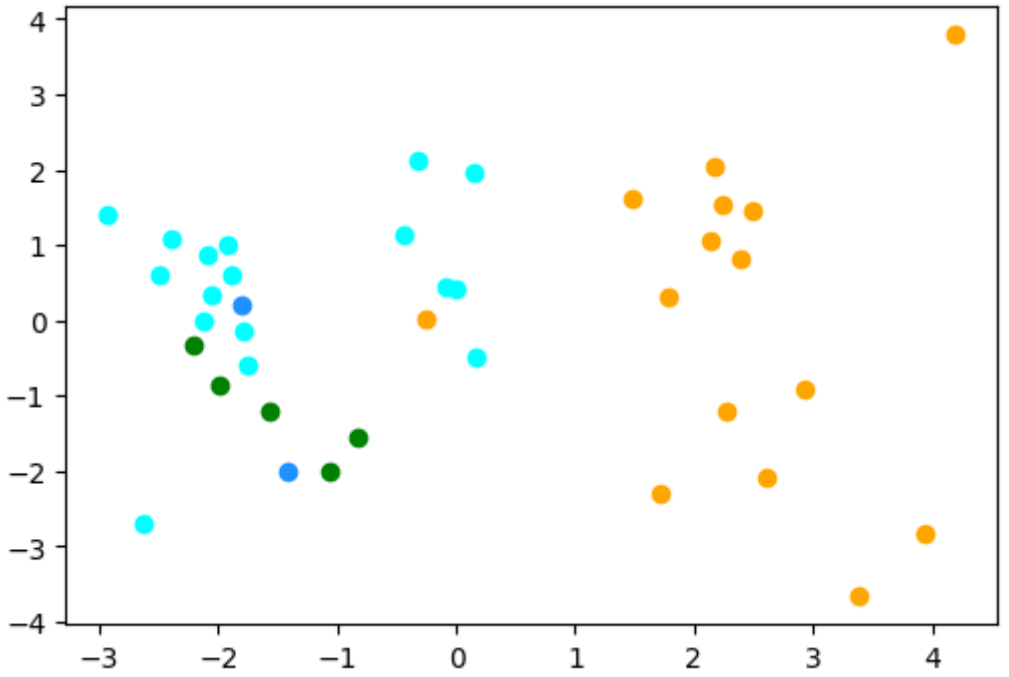
Make embedding in 5 dimension, then use PCA to reduce dimension space to 2 for visualisation

Node2Vec



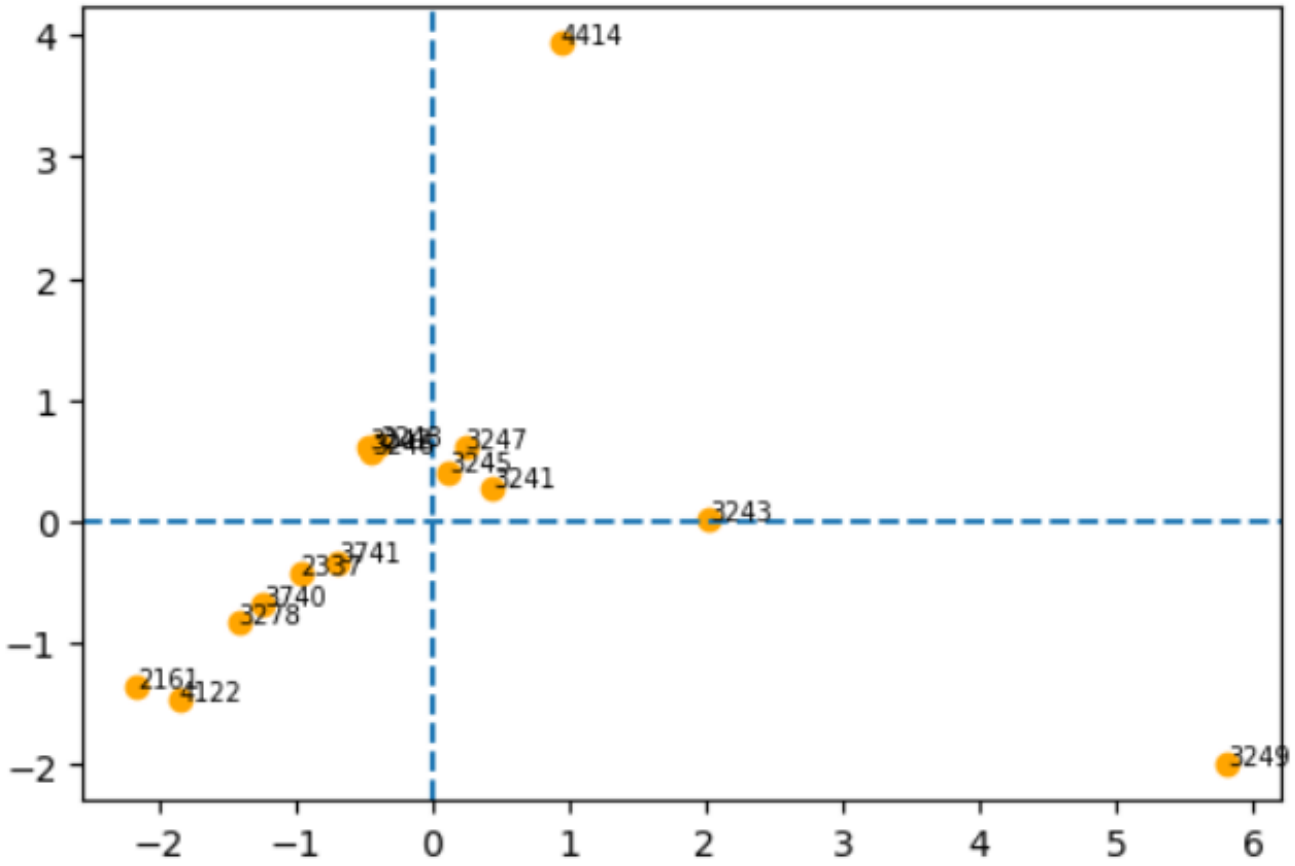
PCA Explained Variance [0.60 0.30]

DeepWalk



PCA Explained Variance [0.47 0.27]

DPM Products




```
{'2161': {'3740': 0.9948839, '3278': 0.99394006, '4122': 0.9933424
'3741': {'2337': 0.9916968, '3740': 0.947313, '2161': 0.9426436},
'3242': {'3248': 0.99478257, '3246': 0.9877767, '3245': 0.9668853},
'3245': {'3247': 0.9765182, '3242': 0.9668853, '3248': 0.9665001},
'2337': {'3741': 0.9916968, '3740': 0.9736819, '2161': 0.97137356},
'3246': {'3248': 0.9981524, '3242': 0.9877767, '3245': 0.95718044},
'3248': {'3246': 0.9981524, '3242': 0.99478257, '3245': 0.9665001},
'3740': {'3278': 0.9969384, '4122': 0.99612695, '2161': 0.9948839},
'3241': {'3247': 0.98654306, '3245': 0.95880204, '3242': 0.9029458}
'3247': {'3241': 0.98654306, '3245': 0.9765182, '3242': 0.94371545}
'4122': {'3278': 0.99913293, '3740': 0.99612695, '2161': 0.9933424
'3278': {'4122': 0.99913293, '3740': 0.9969384, '2161': 0.99394006}
'4414': {'3243': 0.016514644, '3245': 0.0029153244, '3248':-0.01833
'3249': {'3243': 0.8737285, '3241': 0.44131127, '3245': 0.3699685},
'3243': {'3249': 0.8737285, '3241': 0.7728805, '3247': 0.74570847}}
```

Edge Embedding

- Since our random walks are naturally based on the connectivity structure between nodes in the underlying network, we extend them to pairs of nodes using a bootstrapping approach over the feature representations of the individual nodes.

Operator	Symbol	Definition
Average	$\boxplus$	$[f(u) \boxplus f(v)]_i = \frac{f_i(u)+f_i(v)}{2}$
Hadamard	$\boxdot$	$[f(u) \boxdot f(v)]_i = f_i(u) * f_i(v)$
Weighted-L1	$\|\cdot\|_{\bar{1}}$	$\|f(u) \cdot f(v)\|_{\bar{1}i} = \|f_i(u) - f_i(v)\|$
Weighted-L2	$\|\cdot\|_{\bar{2}}$	$\|f(u) \cdot f(v)\|_{\bar{2}i} = \|f_i(u) - f_i(v)\|^2$

Table 1: Choice of binary operators $\circ$ for learning edge features. The definitions correspond to the ith component of $g(u, v)$.

vian nature of the random walk. Hence, our effective complexity is $O(\frac{l}{k(l-k)})$ per sample. For example, in Figure 1 we sample a random walk $\{u, s_4, s_5, s_6, s_8, s_9\}$ of length $l = 6$, which results in $N_S(u) = \{s_4, s_5, s_6\}$, $N_S(s_4) = \{s_5, s_6, s_8\}$ and $N_S(s_5) =$

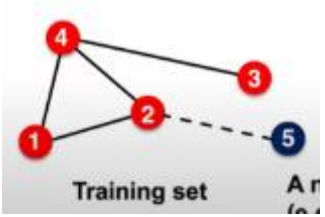
between nodes in the underlying network, we extend them to pairs of nodes using a bootstrapping approach over the feature representations of the individual nodes.

Given two nodes u and v, we define a binary operator $\circ$ over the corresponding feature vectors $f(u)$ and $f(v)$ in order to generate a representation $g(u, v)$ such that $g : V \times V \rightarrow \mathbb{R}^{d'}$ where d' is the representation size for the pair (u, v). We want our operators to be generally defined for any pair of nodes, even if an edge does not exist between the pair since doing so makes the representations useful for link prediction where our test set contains both true and false edges (*i.e.*, do not exist). We consider several choices for the operator $\circ$ such that $d' = d$ which are summarized in Table 1.

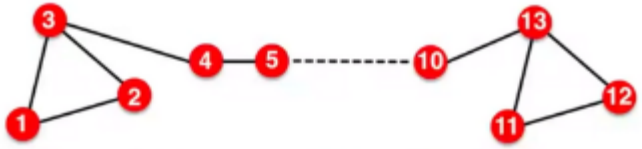
Node2Vec Paper, Grover & Leskovec, 2016

Limitations of Shallow Embeddings

- Can not drive embedding for a new node. Have to recompute embedding for all nodes



- Can not capture Structural similarity



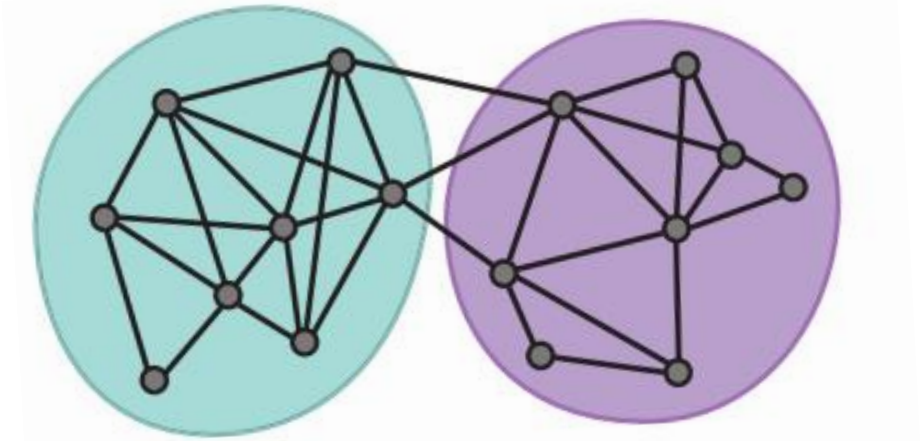
- Embeddings are unique to a graph

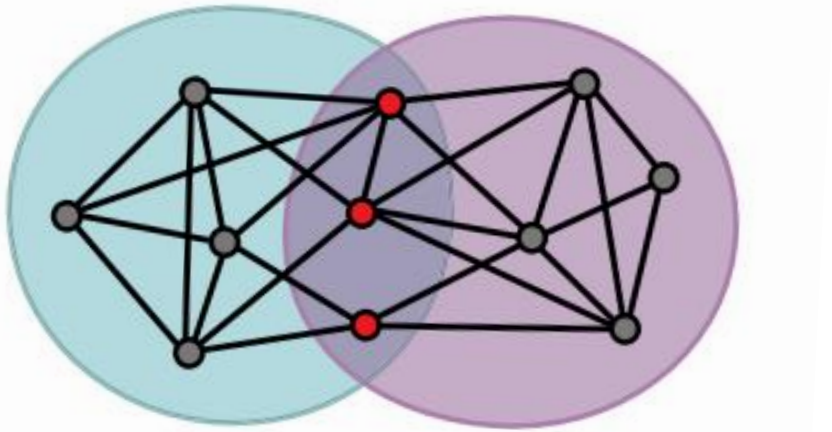
- Can not utilize node, edge, graph features

Solution is to employ Deep Representational Learning or Graph Neural Network

Community Detection

Overlapping Community Detection at Scale: A Nonnegative Matrix Factorization Approach





Non-Overlapping Community Detection

- Benedek Rozemberczki, Ryan Davies, Rik Sarkar, and Charles Sutton: **GEMSEC: Graph Embedding with Self Clustering** Paper, Code
- Pei-Zhen Li, Ling Huang, Chang-Dong Wang, and Jian-Huang Lai: **EdMot: An Edge Enhancement Approach for Motif-aware Community Detection** Paper, Video, Code
- Arnau Prat-Perez, David Dominguez-Sal, Joseph-Luis Larriba-Pey: **High Quality, Scalable and Parallel Community Detectionfor Large Real Graphs** Paper
- Usha Nandini Raghavan, Reka Albert, Soundar Kumara: **Near Linear Time Algorithm to Detect Community Structures in Large-Scale Networks** Paper, Code

Overlapping Community Detection

- Alessandro Epasto, Silvio Lattanzi, Renato Paes Leme: **Ego-splitting Framework: from Non-Overlapping to Overlapping Clusters** Paper, Slides, Video, Code
- Fanghua Ye, Chuan Chen, Zibin Zheng: **Deep Autoencoder-like Nonnegative Matrix Factorization for Community Detection** Paper, Code
- Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, Shiqiang Yang: **Community Preserving Network Embedding** Paper, Code
- Bing-Jie Sun, Huawei Shen, Jinhua Gao, Wentao Ouyang, Xueqi Cheng: **A Non-negative Symmetric Encoder-Decoder Approach for Community Detection** Paper
- Jaewon Yang and Jure Leskovec: **Overlapping Community Detection at Scale: A Nonnegative Matrix Factorization Approach** Paper, Slides, Video, Video
- Da Kuang, Chris Ding, Haesun Park: **Symmetric Nonnegative Matrix Factorization for Graph Clustering** Paper

Applications that could benefit from Graph Analytics Methods?

Thank you for your attention!

Any Questions?

Appendix

Graphlets

- Considering graphlets with 2 to 5 node we get a vector of 73 coordinates as node signature
- Captures
- Given node position and number of neighbors there are multiple possibilities