Efficient multi-relational network representation using primes

9.1.2023

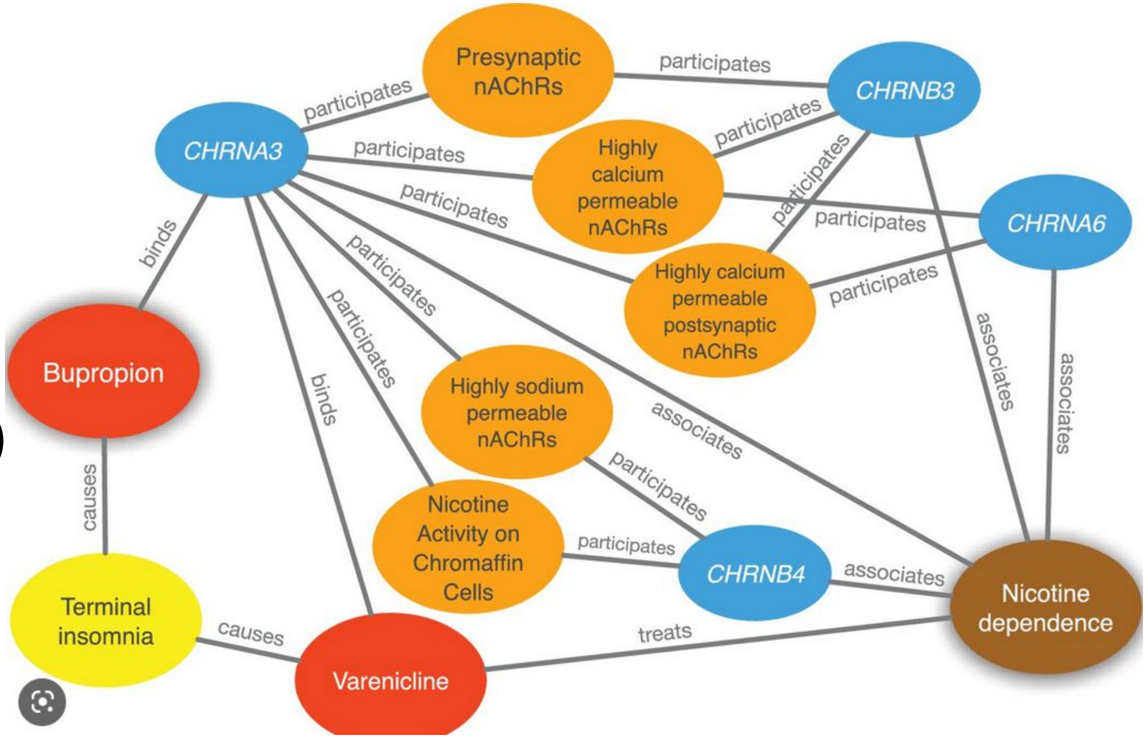
Konstantinos Bougiatiotis
CNA @ NCSR-D

Introduction

- **Multi-relational** networks (multi-layer/plex/dimensional …)

  Everything that can be expressed as a collection of triples

  (head - relation - tail)



- Many tasks rely on capturing information

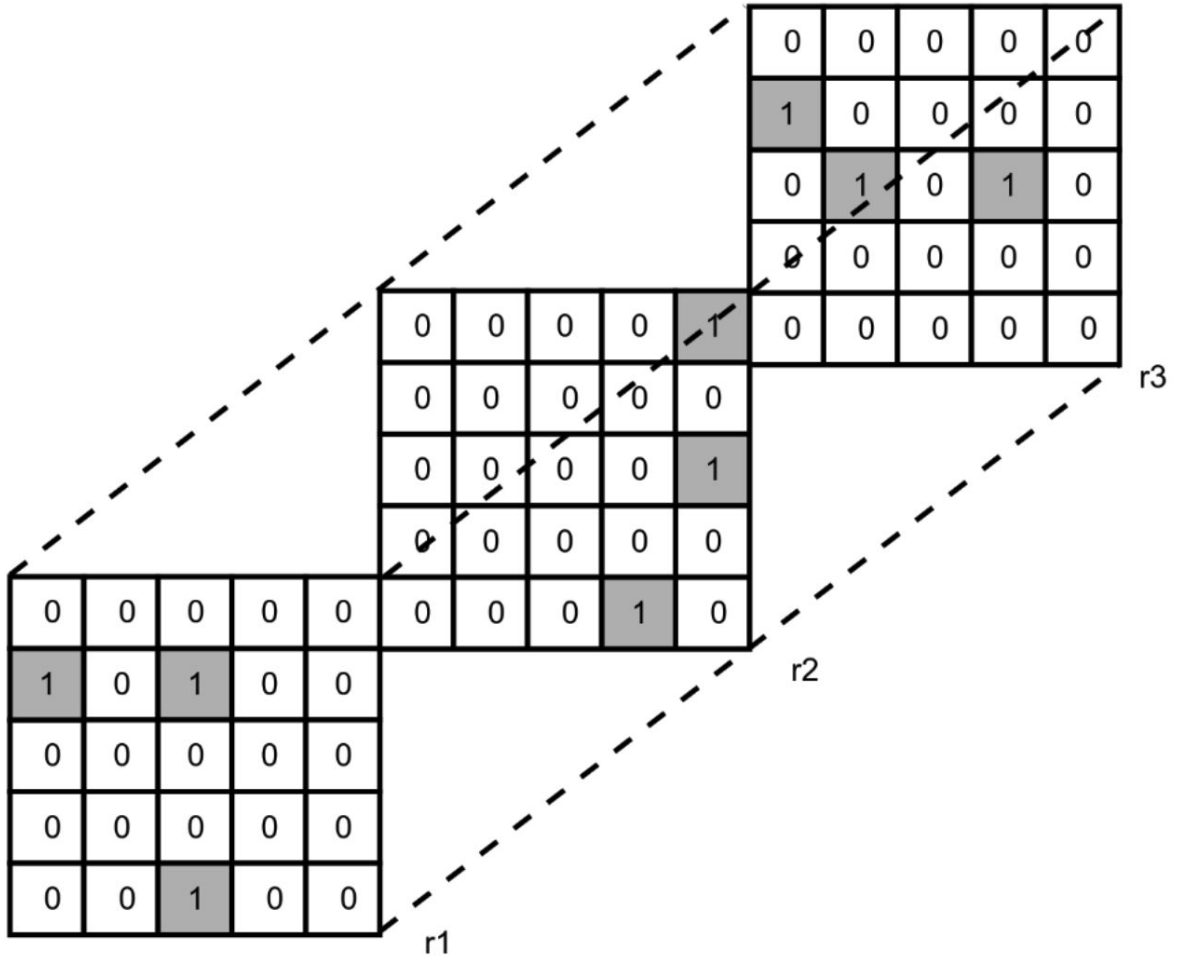
  by taking into account the **paths between** the **entities**

- We need **fast** and **scalable** ways to do so

  (current approaches rely on **path-sampling**)

Motivation

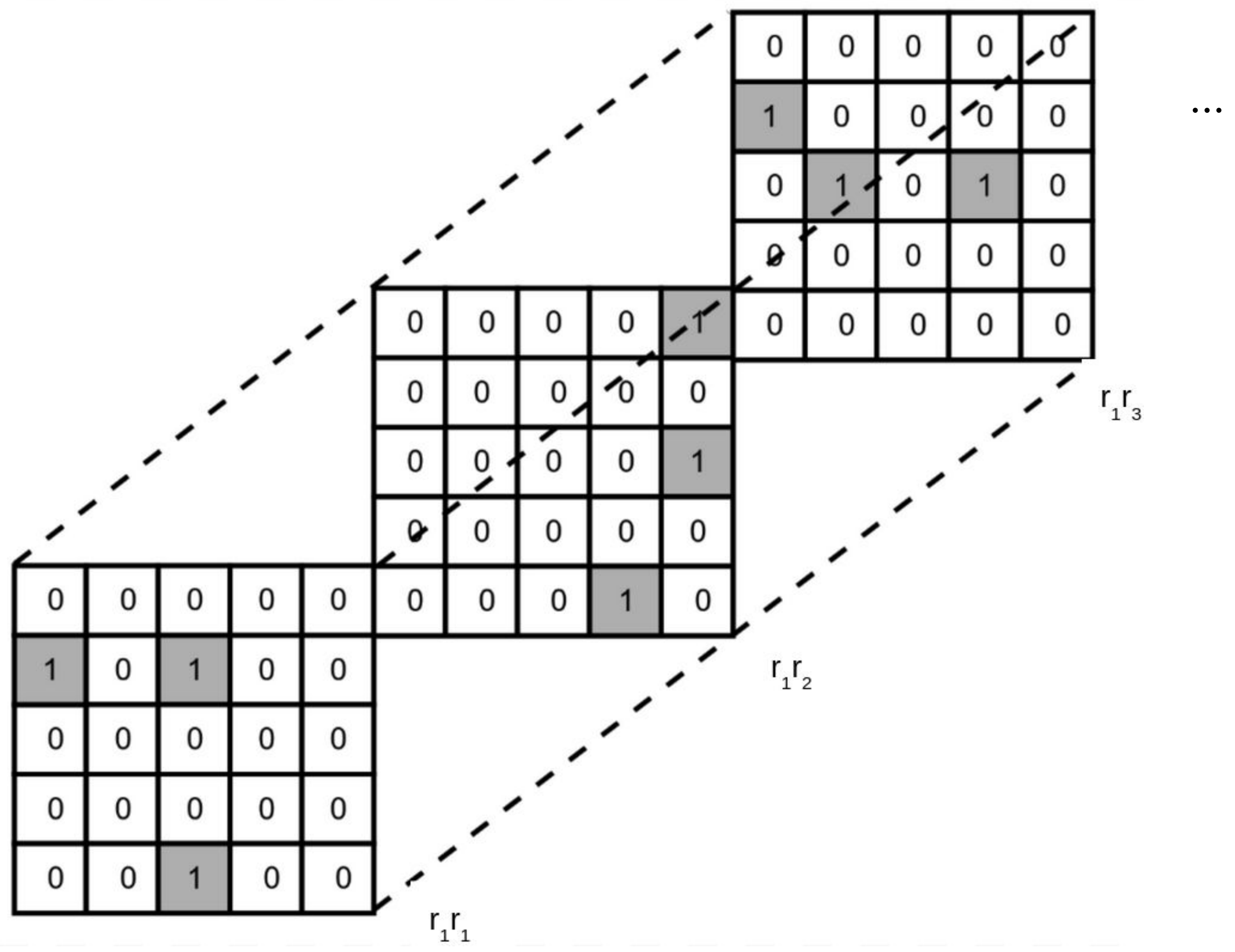
- Representing the network with a **tensor** of shape **NxNxR**



Motivation

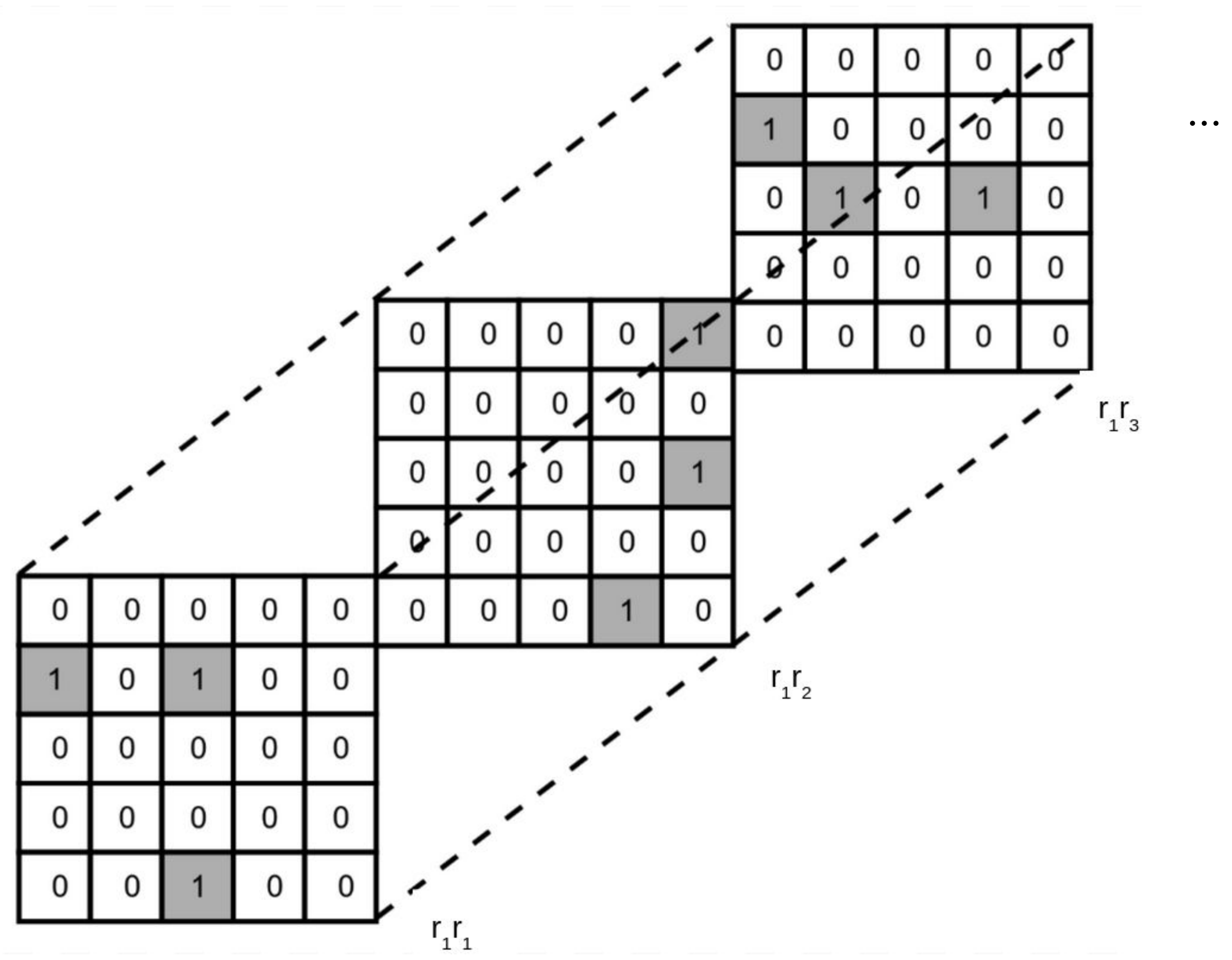
- Representing the network with a **tensor** of shape **NxNxR**

- Calculating the values of the 2-hop adj. tensor would take a tensor of **NxNxR^2**.



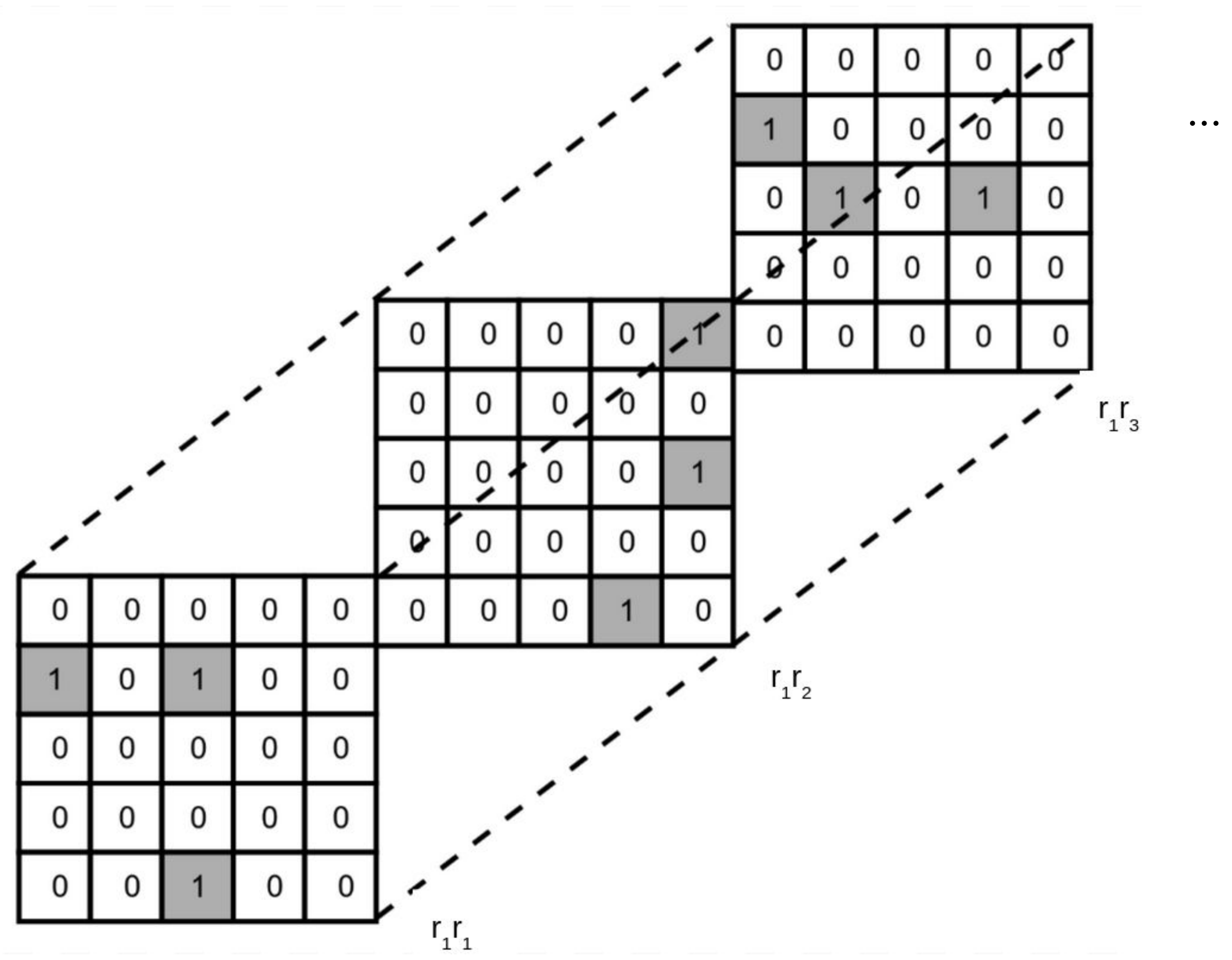
Motivation

- Representing the network with a **tensor** of shape **NxNxR**
- Calculating the values of the 2-hop adj. tensor would take a tensor of **NxNxR^2**.
- In general **k-hops** would need **NxNxR^k**



Motivation

- Representing the network with a **tensor** of shape **NxNxR**
- Calculating the values of the 2-hop adj. tensor would take a tensor of **NxNxR^2**.
- In general **k-hops** would need **NxNxR^k**
- **Not efficient** even for small-scale real-world datasets



Idea

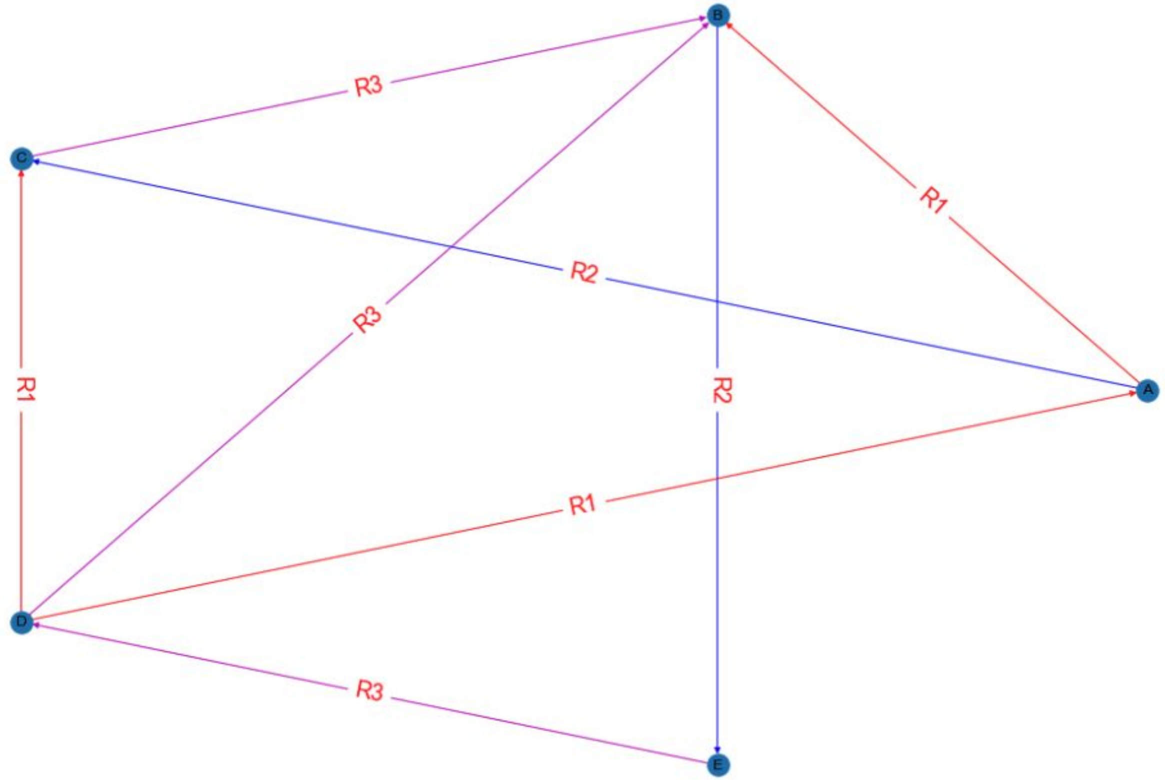
- **Collapse the NxNxR tensor into a**

  NxN adj. matrix in a **lossless** manner

Idea

- **Collapse the NxNxR tensor into a**

  NxN adj. matrix in a **lossless** manner



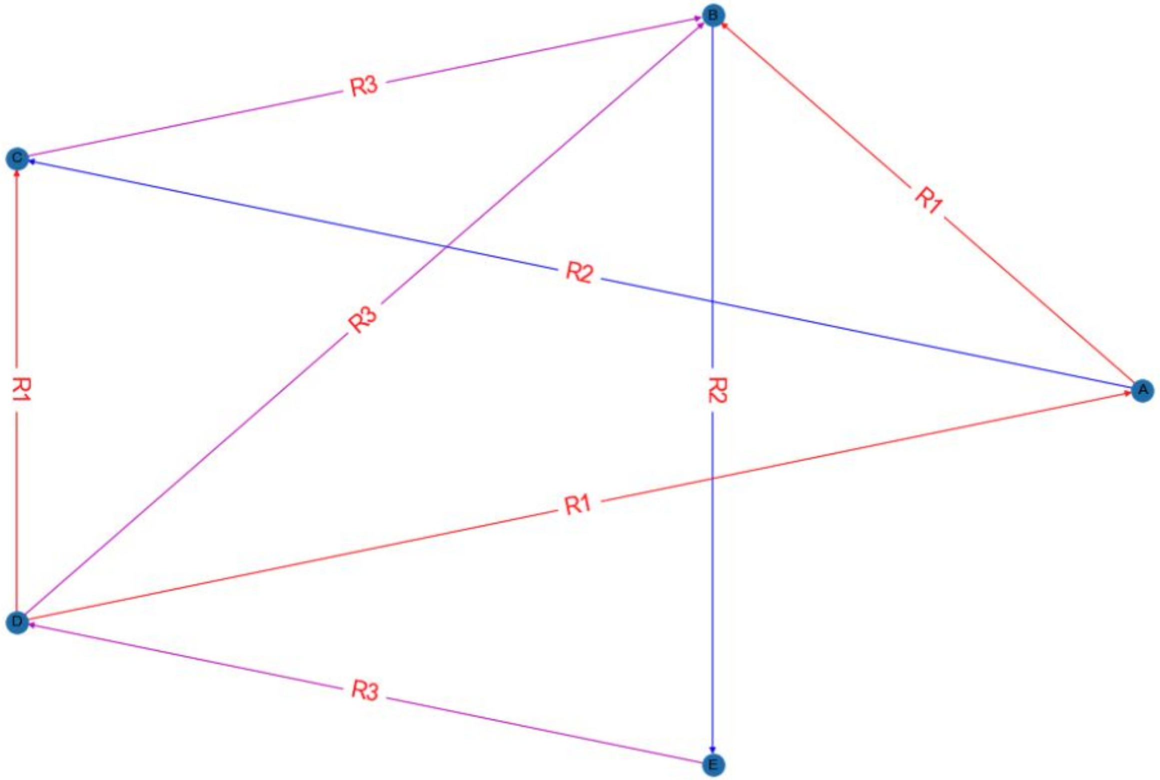
	A	B	C	D	E
A	0	r1	r2	0	0
B	0	0	0	0	r2
C	0	r3	0	0	0
D	r1	r3	r1	0	0
E	0	0	0	r3	0

decomposed

Idea

- **Collapse the NxNxR tensor into a NxN adj. matrix** in a **lossless** manner

- Map all **relations** to a **distinct prime number**



R1 (red) : 3
R2 (blue) : 5
R3 (purple) : 7

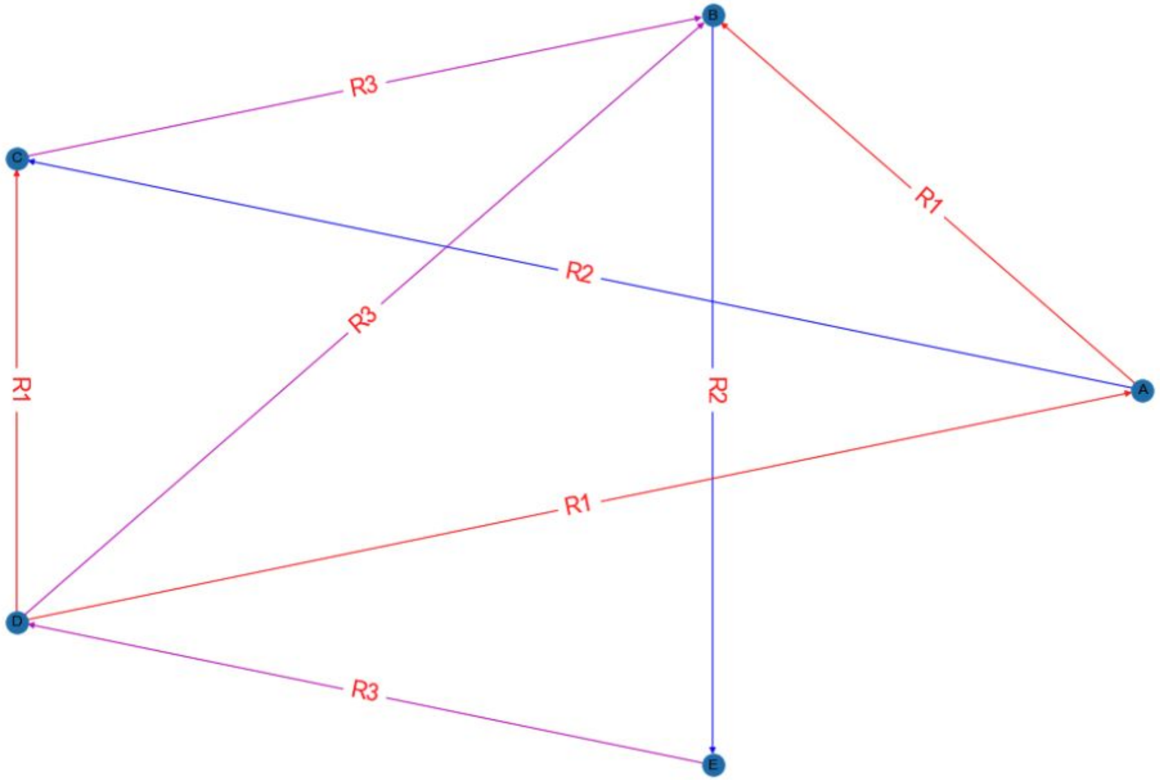
	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

decomposed

Idea

- **Collapse the NxNxR tensor into a** **NxN adj. matrix** in a **lossless** manner

- Map all **relations** to a **distinct prime number**



R1 (red) : 3
R2 (blue) : 5
R3 (purple) : 7

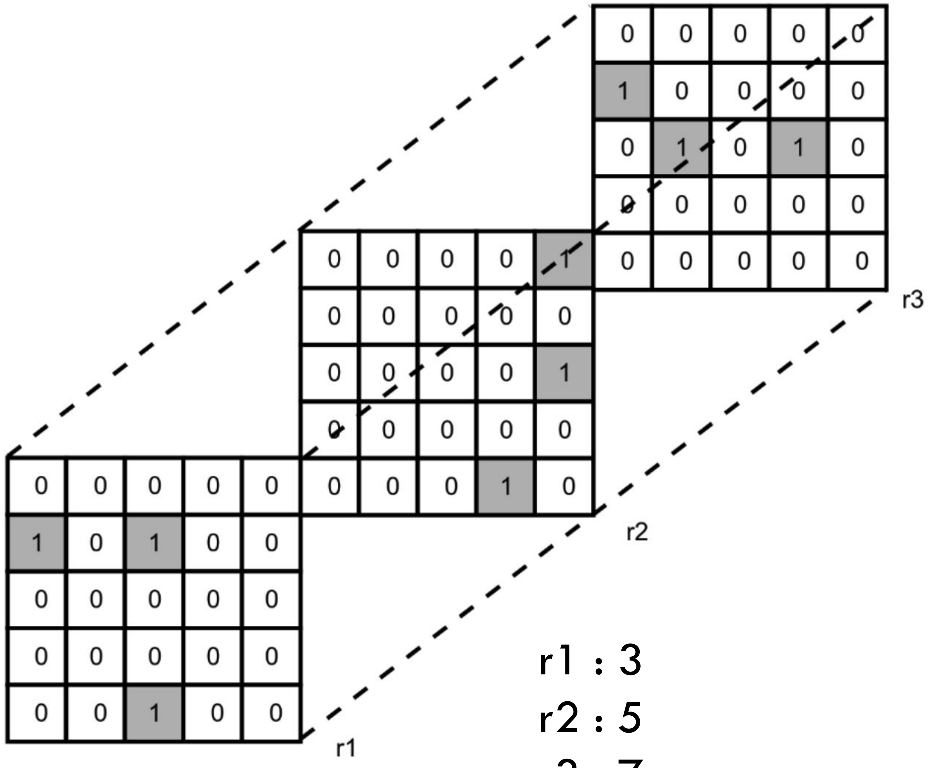
	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

Motivation

*A given number is uniquely decomposed into its **prime factors***

Features

- **Lossless compression** if we use the product of relations



r1 : 3
r2 : 5
r3 : 7

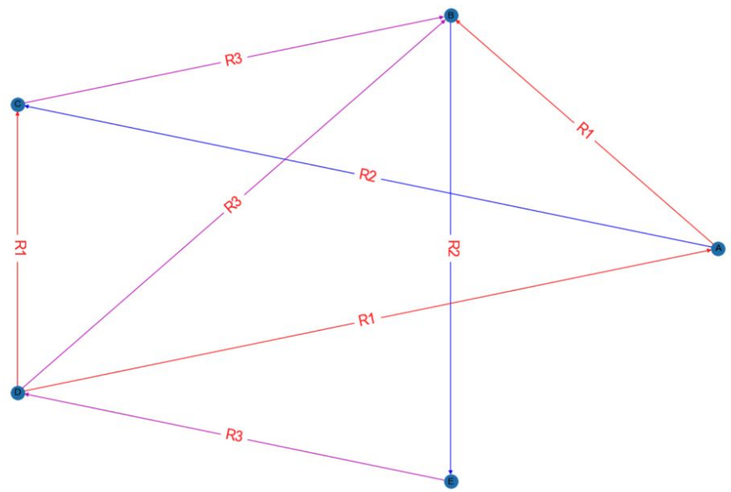
	A	B	C	D	E
A	0	0	0	0	5
B	21	0	3	0	0
C	0	7	0	7	5
D	0	0	0	0	0
E	0	0	3	5	0

Features

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

- **Lossless compression** if we use the product of relations

- The **Prime Adjacency Matrix** (PAM) is a very expressive representation
  - **Non-zero** values indicate:
    - Rows/columns depict the outgoing/incoming edges per relation, per node
    - **Frequency counts** of relations exhibited
    - …



Features

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

- **Lossless compression** if we use the product of relations

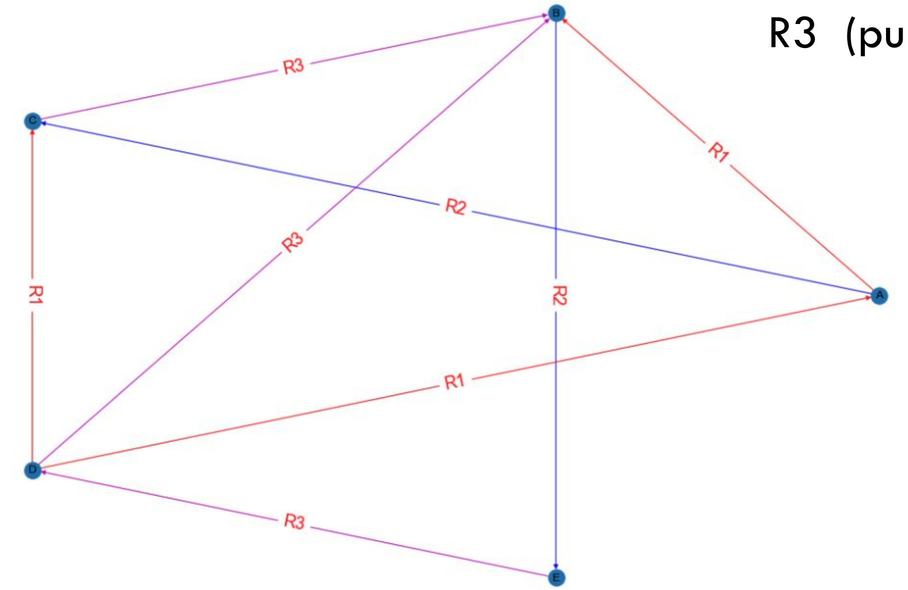
- The **Prime Adjacency Matrix** (PAM) is a very expressive representation
  - **Non-zero** values indicate:
    - Rows/columns depict the outgoing/incoming edges per relation, per node
    - **Frequency counts** of relations exhibited
    - …

- It allows us to use **tools from classical network analysis**
  - Focusing on **powers of the adjacency matrix**

Features

- **PAM k-hop adjacency captures information from all relational layers**

  (while keeping matrix shapes to NxN)



R1 (red) : 3
R2 (blue) : 5
R3 (purple) : 7

1-hop

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

2-hop

	A	B	C	D	E
A	0	35	0	0	15
B	0	0	0	35	0
C	0	0	0	0	35
D	0	30	15	0	35
E	21	49	21	0	0

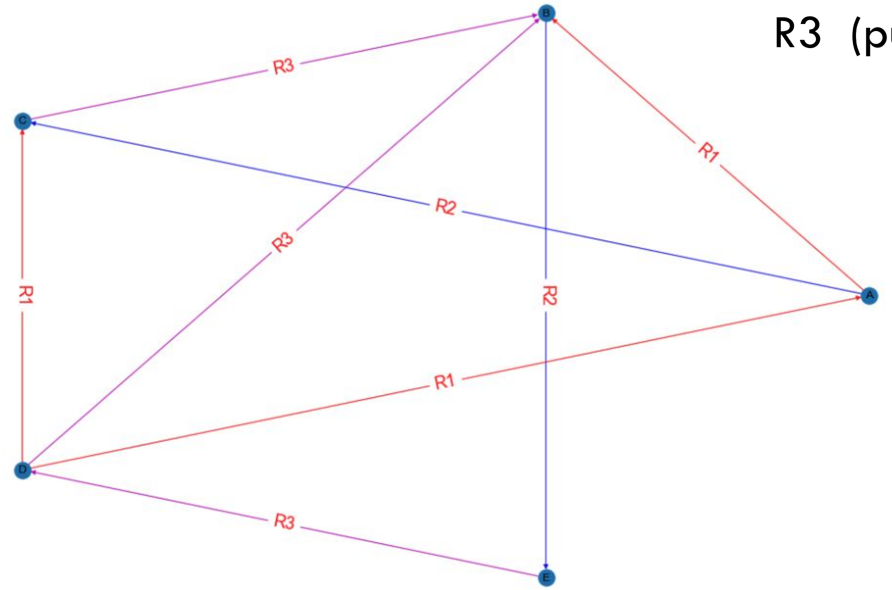
Features

- **PAM k-hop adjacency captures information from all relational layers**

  (while keeping matrix shapes to NxN)

  $PAM^2[A,B] = 35 = 5*7 = A\text{-}5\text{->}C\text{-}7\text{->}B$

R1 (red) : 3
R2 (blue) : 5
R3 (purple) : 7



1-hop

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

2-hop

	A	B	C	D	E
A	0	**35**	0	0	15
B	0	0	0	35	0
C	0	0	0	0	35
D	0	30	15	0	35
E	21	49	21	0	0

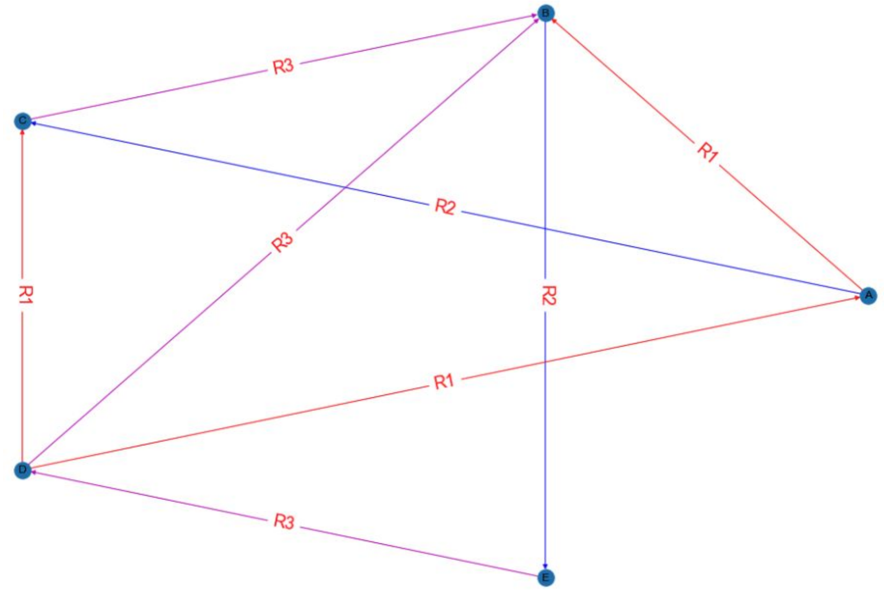
Features

- **PAM k-hop adjacency captures information from all relational layers**

  (while keeping matrix shapes to NxN)

  PAM^2[A,B] = 35 = 5*7 = A-5->C-7->B

  PAM^2[D,B] = 30 = 3*3 + 3*7 = D-3->A-3->B + D-3->C-7->B



1-hop

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

2-hop

	A	B	C	D	E
A	0	35	0	0	15
B	0	0	0	35	0
C	0	0	0	0	35
D	0	**30**	15	0	35
E	21	49	21	0	0

Features

- **PAM k-hop adjacency captures information from all relational layers**

  (while keeping matrix shapes to NxN)

  $PAM^2[A,B] = 35 = 5*7 = A\text{-}5\text{->}C\text{-}7\text{->}B$

  $PAM^2[D,B] = 30 = 3*3 + 3*7 = D\text{-}3\text{->}A\text{-}3\text{->}B + D\text{-}3\text{->}C\text{-}7\text{->}B$

1-hop

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

2-hop

	A	B	C	D	E
A	0	35	0	0	15
B	0	0	0	35	0
C	0	0	0	0	35
D	0	**30**	15	0	35
E	21	49	21	0	0

IMPORTANT

PAM^k values express the **aggregates of paths that connect two nodes**

Features

- **PAM k-hop adjacency captures information from all relational layers** (while keeping matrix shapes to NxN)

- **Structural characteristics** for nodes, pairs, paths, subgraphs, and the whole graph can be simply extracted **by looking up** these higher-order matrices.

1-hop

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

2-hop

	A	B	C	D	E
A	0	35	0	0	15
B	0	0	0	35	0
C	0	0	0	0	35
D	0	30	15	0	35
E	21	49	21	0	0

IMPORTANT

PAM^k values express the **aggregates of paths that connect two nodes**

Features

- **PAM k-hop adjacency captures information from all relational layers** (while keeping matrix shapes to NxN)
- **Structural characteristics** for nodes, pairs, paths, subgraphs, and the whole graph can be simply extracted **by looking up** these higher-order matrices.
- Works the same for **heterogeneous** or **undirected** networks

IMPORTANT

PAM^k values express the **aggregates of paths that connect two nodes**

1-hop

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

2-hop

	A	B	C	D	E
A	0	35	0	0	15
B	0	0	0	35	0
C	0	0	0	0	35
D	0	30	15	0	35
E	21	49	21	0	0

Applications

- Usability

Dataset	N	R	# Edges	P^5 (seconds)
CoDEx-S	2,034	42	32,888	2.71
WN18RR	40,493	11	86,835	5.03
FB15k-237	14,541	237	272,115	59.29
YAGO3-10	123,182	37	1,079,040	69.69
HetioNet	45,158	24	2,250,197	418.35 ($\approx$ 7 mins)
ogbl-wikikg2	2,500,604	535	17,137,181	4174.28 ($\approx$ 70 mins)

(11^5 = 161K , 237^5 = 747B)

Applications

1. **Node representations**

A B C D E

A [0, 3, 5, 0, 0]* = 15

A B C D E

A [0, 0, 0, 3, 0]* = 3

A = [15, 3] B = [5, 147]

- Node classification

- Clustering/ Visualization (based on PAM coordinates)

1-hop

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

Applications

1. **Node representations**

2. **Path representations**

k-hop adjacency matrix **expresses the chain of relations** that connect a pair of nodes

- Node representation (**capturing k-hop chains** around each node)

A [15, 3] -> A [15, 3, 35*15 ,21 …]

- Link prediction (k-hop feature vector representing each pair)

1-hop 2-h 3-h …

A-B 3 [35, 525, …]

1-hop

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

2-hop

	A	B	C	D	E
A	0	35	0	0	15
B	0	0	0	35	0
C	0	0	0	0	35
D	0	30	15	0	35
E	21	49	21	0	0

Applications

1. **Node representations**
2. **Path representations**
3. **Sub-/Graph representations**

G : [3,3,3,5,5,7,7,7]* = 231525

- **Unique Number** expressing **a specific number of edges per relation**
- k-hop number as well (e.g. 2-hop = 218882287968750, 3-hop etc..)
- Fast graph "**fingerprinting**"/classification

- k-hop PAM can be used to **express counts of k-motifs**
- (# 35 in 2-hop PAM -> # 3*5=R1R2 motifs etc.)

1-hop

	A	B	C	D	E
A	0	3	5	0	0
B	0	0	0	0	5
C	0	7	0	0	0
D	3	7	3	0	0
E	0	0	0	7	0

2-hop

	A	B	C	D	E
A	0	35	0	0	15
B	0	0	0	35	0
C	0	0	0	0	35
D	0	30	15	0	35
E	21	49	21	0	0

Use-case: Triple classification

Given a KG with **N** nodes/entities and **R** types of relations, find the most **probable relation** that **should connect two existing nodes** - i.e. complete the triplet (head, ?, tail)

Use-case: Triple classification

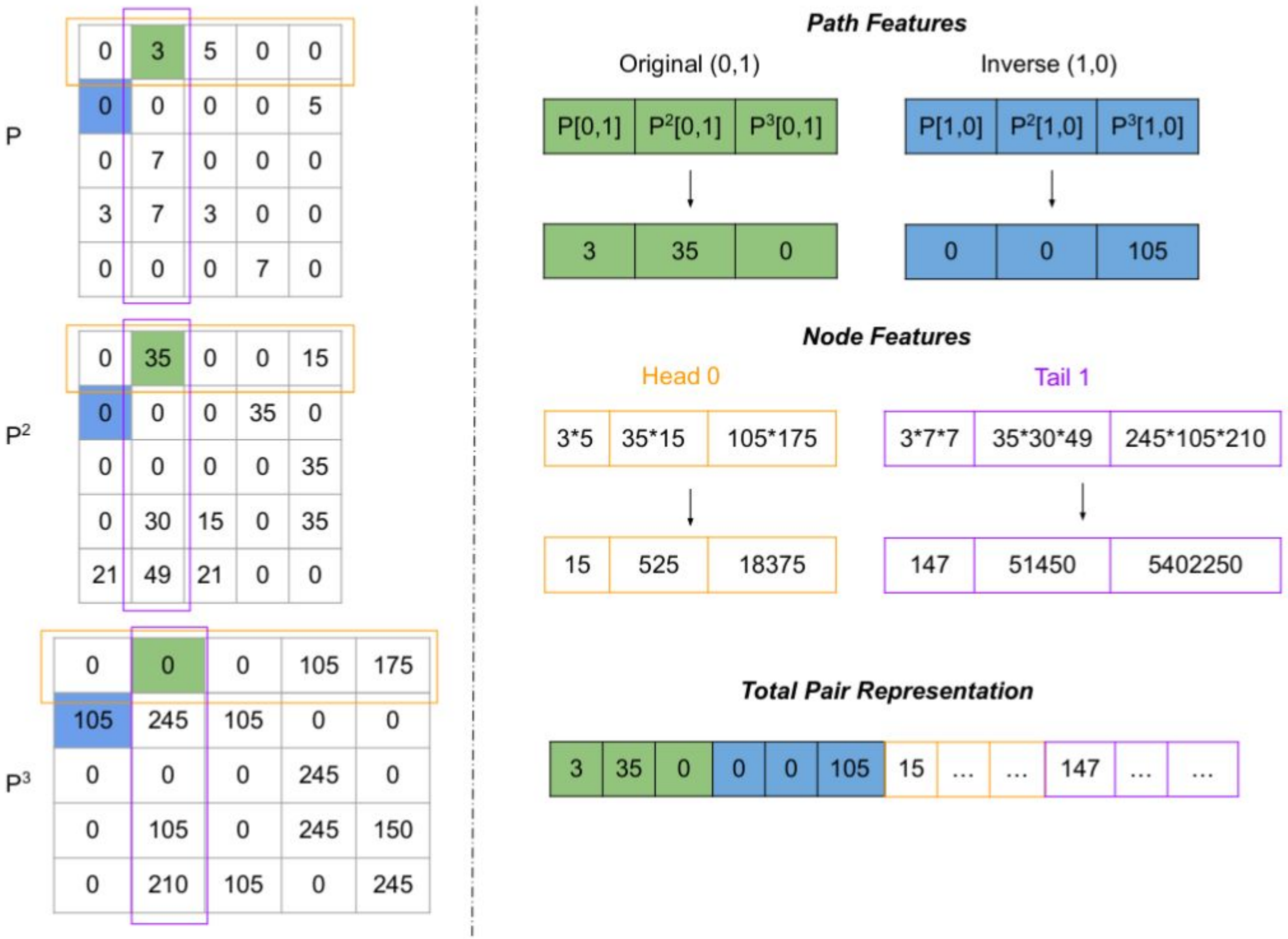
Given a KG with **N** nodes/entities and **R** types of relations, find the most **probable relation** that **should connect two existing nodes** - i.e. complete the triplet (head, ?, tail)

1. Simple **kNN-based** approach to showcase the **expressivity** of the PAM matrices.
    a. For all given (h,r,t) triples, **embed (h,t)** and label them with **r.**
    b. At query time, embed (h_q , t_q) in the **same space**
    c. Find the most-possible relation using **the labels of its nearest neighbors**

Use-case: Triple classification

Embedding pair (A,B) = (0,1) with label 3

1. Represent the path (direct and inverse)
2. Represent the head and tail
3. Concatenate 1,2



Use-case: Triple classification

- Datasets

Dataset	N	R	# training	# validation	# test #	$\mathbb{E}[d]$	$\mathrm{Var}[d]$
NELL995	63,917	198	137,465	5,000	5,000	4.3	750.6
WN18RR	40,493	11	86,835	3,034	3,134	4.2	64.3
DDB14	9,203	14	36,561	4,000	4,000	7.9	978.8

- HyperParams
  - # of k-hops
  - # of neighbors
  - euclidean distance or one-hot encoding + manhattan

Use-case: Triple classification

- Results

	NELL995			WN18RR			DDB14		
	MRR	H@1	H@3	MRR	H@1	H@3	MRR	H@1	H@3
TransE	0.784	0.669	0.870	0.841	**0.781**	0.889	**0.966**	**0.948**	0.980
CompleX	0.840	0.777	0.880	0.703	0.625	0.765	0.953	0.931	0.968
DistMult	0.847	**0.787**	0.891	0.634	0.524	0.720	0.927	0.886	0.961
RotatE	0.799	0.735	0.823	0.729	0.691	0.756	0.953	0.934	0.964
DRUM	**0.854**	0.778	**0.912**	0.715	0.640	0.740	0.958	0.930	**0.987**
knn-PP	0.730	0.627	0.821	**0.857**	0.741	**0.972**	0.925	0.888	0.962

- Comparable performance (**better in WN18RR**)

Wang H, Ren H, Leskovec J (2021) Relational message passing for knowledge graph completion. SIGKDD-21

Use-case: Triple classification

- Results

	NELL995			WN18RR			DDB14		
	MRR	H@1	H@3	MRR	H@1	H@3	MRR	H@1	H@3
TransE	0.784	0.669	0.870	0.841	**0.781**	0.889	**0.966**	**0.948**	0.980
CompleX	0.840	0.777	0.880	0.703	0.625	0.765	0.953	0.931	0.968
DistMult	0.847	**0.787**	0.891	0.634	0.524	0.720	0.927	0.886	0.961
RotatE	0.799	0.735	0.823	0.729	0.691	0.756	0.953	0.934	0.964
DRUM	**0.854**	0.778	**0.912**	0.715	0.640	0.740	0.958	0.930	**0.987**
knn-PP	0.730	0.627	0.821	**0.857**	0.741	**0.972**	0.925	0.888	0.962

- Comparable performance (**better in WN18RR**)
- **Training takes a matter of minutes** vs hours on GPUs

Use-case: Triple classification

- Results

	NELL995			WN18RR			DDB14		
	MRR	H@1	H@3	MRR	H@1	H@3	MRR	H@1	H@3
TransE	0.784	0.669	0.870	0.841	**0.781**	0.889	**0.966**	**0.948**	0.980
CompleX	0.840	0.777	0.880	0.703	0.625	0.765	0.953	0.931	0.968
DistMult	0.847	**0.787**	0.891	0.634	0.524	0.720	0.927	0.886	0.961
RotatE	0.799	0.735	0.823	0.729	0.691	0.756	0.953	0.934	0.964
DRUM	**0.854**	0.778	**0.912**	0.715	0.640	0.740	0.958	0.930	**0.987**
knn-PP	0.730	0.627	0.821	**0.857**	0.741	**0.972**	0.925	0.888	0.962

- Comparable performance (**better in WN18RR**)
- **Training takes a matter of minutes** vs hours on GPUs
- **No-trainable parameters**

Model	TransE	ComplEx	DisMult	RotatE	**kNN-PP**
# trainable param.	3.7M	7.4M	3.7M	7.4M	0

Use-case: Graph Classification

- Binary classification of molecular graphs

Dataset	# Graphs	Avg. # Nodes	Avg. # Edges
AIDS	2,000	15.69	16.20
BZR_MD	306	21.30	225.06
COX2_MD	303	26.28	335.12
DHFR_MD	393	23.87	283.01
ER_MD	446	21.33	234.85
MUTAG	188	17.93	19.79
Mutagenicity	4337	30.32	30.77
PTC_FM	349	14.11	14.48
PTC_FR	351	14.56	15.00
PTC_MM	336	13.97	14.32
PTC_MR	344	14.29	14.69

- Benchmark this through the scope of graph kernels

Use-case: Graph Classification

- Use **PAMs to embed each graph** in a space

$$P = \begin{bmatrix} 0 & 3 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 \\ 0 & 7 & 0 & 0 & 0 \\ 3 & 7 & 3 & 0 & 0 \\ 0 & 0 & 0 & 7 & 0 \end{bmatrix} \Rightarrow \prod\{3,3,3,5,5,7,7,7\} = 231525$$

$$P^2 = \begin{bmatrix} 0 & 35 & 0 & 0 & 15 \\ 0 & 0 & 0 & 35 & 0 \\ 0 & 0 & 0 & 0 & 35 \\ 0 & 30 & 15 & 0 & 35 \\ 21 & 49 & 21 & 0 & 0 \end{bmatrix} \Rightarrow \prod\{...\} = 218882287968750$$

$$P^3 = \begin{bmatrix} 0 & 0 & 0 & 105 & 175 \\ 105 & 245 & 105 & 0 & 0 \\ 0 & 0 & 0 & 245 & 0 \\ 0 & 105 & 0 & 245 & 150 \\ 0 & 210 & 105 & 0 & 245 \end{bmatrix} \Rightarrow \prod\{...\} = V$$

G = [231525, 218882287968750, v]

- **Calculate distance** between graphs (using a default RBF kernel) using their representations

Use-case: Graph Classification

- PAMs **do not handle node-labels**, so we also have a variant that combines the distance of the PAM methodology with a **vertext-histogram** kernel distance (PP-VH)

- Measure performance (**time, accuracy**) against one of the best-performing kernels, the **Weisfeiler-Lehman Optimal Assignment** kernel (WL-OA)

- All models are equipped with an **SVM** on-top

- 5-fold outer cross-validation scheme
  - with an inner 3-fold cross-validation scheme to select penalty C

Kriege, N.M., Johansson, F.D. & Morris, C. A survey on graph kernels. Appl Netw Sci 5, 6 (2020)

Use-case: Graph Classification

- Results

	PP		PP-VH	
Dataset	$\Delta Acc\%$	$\Delta Time\%$	$\Delta Acc\%$	$\Delta Time\%$
AIDS	-1.45	-99.65	+0.40	-99.44
BZR_MD	+5.66	-94.78	+12.22	-88.42
COX2_MD	-20.20	-95.79	+0.15	-93.26
DHFR_MD	-16.99	-93.78	-48.90	-18.50
ER_MD	-0.38	-92.10	+7.45	-89.90
MUTAG	-8.69	-89.88	+2.21	-82.93
Mutagenicity	-28.53	-99.75	-20.23	-95.72
PTC_FM	-2.99	-95.37	-18.06	-67.87
PTC_FR	-10.25	-94.16	+10.97	+234.68
PTC_MM	-18.35	-93.99	-7.73	-27.64
PTC_MR	-9.83	-95.45	-14.39	-72.10

- Accuracy: WL-OA better than PP, better in 5/11 datasets versus PP-VH

- Time: PP more than 90% faster in all cases, PP-VH is more than 50% faster in most cases

Advantages

1. **Fast** and **scalable** (needs to be computed only once)

Advantages

1. **Fast** and **scalable** (needs to be computed only once)

2. **Very rich** in terms of capturing **(higher-order) structural properties**

Advantages

1. **Fast** and **scalable (**needs to be computed only once)

2. **Very rich** in terms of capturing **(higher-order) structural properties**

3. **Freedom of modelling** in downstream tasks
    a. **Node** / **Edges** / (Sub)-**Graphs** representations

Advantages

1. **Fast** and **scalable** (needs to be computed only once)

2. **Very rich** in terms of capturing **higher-order structural properties**

3. **Freedom of modelling** in downstream tasks

4. Fast **structure-based method** to complement others or use as **baseline**
   a. E.g. support node-feature based methodologies

Advantages

1. **Fast** and **scalable** (needs to be computed only once)

2. **Very rich** in terms of capturing **higher-order structural properties**

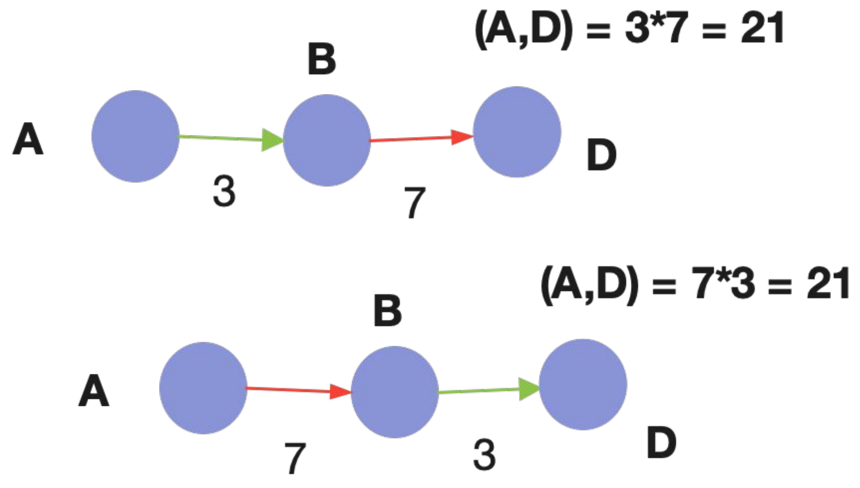
3. **Freedom of modelling** in downstream tasks

4. Fast **structured-based method** to complement other ones or use as **baseline**

5. Opens up the use of other **tools from classical network analysis in complex networks** as well
   a. E.g. spectral analysis, topological graph theory …

Open Challenges

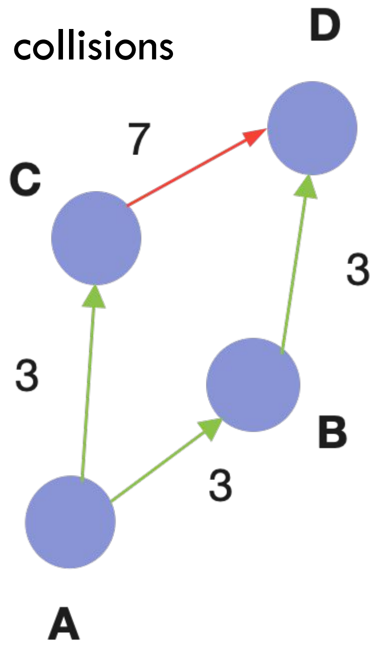
- **Order of relations** is not maintaned in PAM^k
  - It may be possible, if we keep track of the chain up to PAM^{k-1}



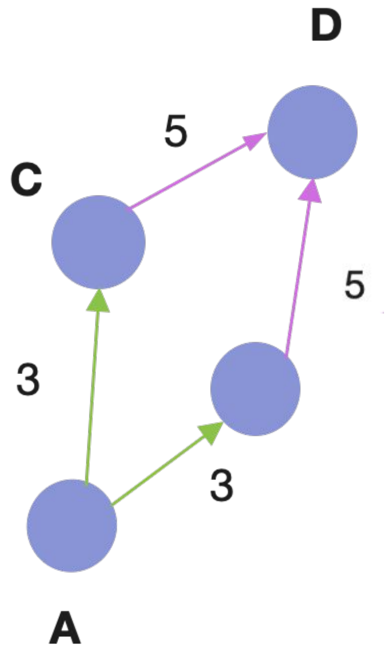
Open Challenges

- **Order of relations** is not maintaned in PAM^k

- Exact **decomposition of a sum of products** is not feasible
  - **Collisions** due to overlaps of sums
  - Empirically: greater **space** between the primes leads to fewer collisions
    - E.g. [3,31,311] is better than [3,5,7]
  - Selection of specific set of numbers such as their sum of products are unique(?)



(A,D) = 3*7 + 3*3 = 30



(A,D) = 3*5 + 3*5 = 30

Open Challenges

- **Order of relations** is not maintaned in PAM^k
- Exact **decomposition of a sum of products** is not feasible
- **Overflowing** problems
  - Internally using log-transform for the downstream tasks

$$P \quad \begin{bmatrix} 0 & 3 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 \\ 0 & 7 & 0 & 0 & 0 \\ 3 & 7 & 3 & 0 & 0 \\ 0 & 0 & 0 & 7 & 0 \end{bmatrix} \Rightarrow \prod\{3,3,3,5,5,7,7,7\} = 231525$$

$$P^2 \quad \begin{bmatrix} 0 & 35 & 0 & 0 & 15 \\ 0 & 0 & 0 & 35 & 0 \\ 0 & 0 & 0 & 0 & 35 \\ 0 & 30 & 15 & 0 & 35 \\ 21 & 49 & 21 & 0 & 0 \end{bmatrix} \Rightarrow \prod\{...\} = 218882287968750$$

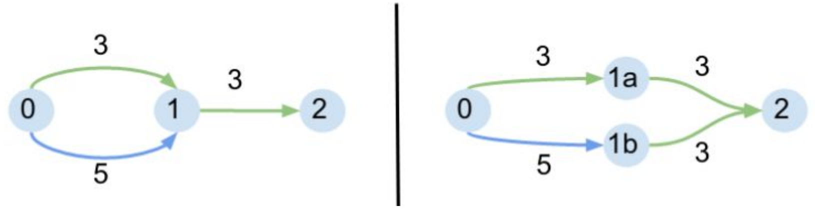
$$P^3 \quad \begin{bmatrix} 0 & 0 & 0 & 105 & 175 \\ 105 & 245 & 105 & 0 & 0 \\ 0 & 0 & 0 & 245 & 0 \\ 0 & 105 & 0 & 245 & 150 \\ 0 & 210 & 105 & 0 & 245 \end{bmatrix} \Rightarrow \prod\{...\} = v$$

$$G = [231525, 218882287968750, v]$$

Thank you!

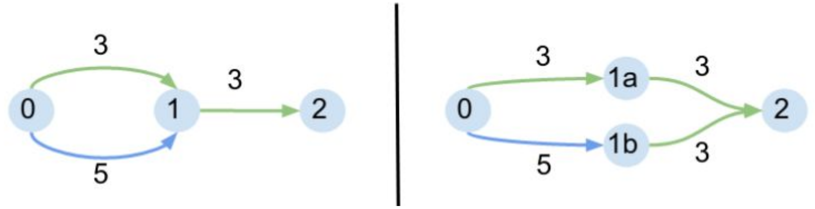
Questions/Ideas?

PAM vs PAM$_+$



- We have 2 paths connecting 0->2
- 0-3->1-3->2 = 3*3 = 9 and 0-5->1-3->2 = 5*3 = 15
- We would expect $P^2[0,2] = 9 + 15 = 24$
- If we use $P[0,1] = 3*5 = 15$, then $P^2[0,2] = 15*3 = 45 \neq 24$
- But if we use $P[0,1] = 3 + = 8$, then $P^2[0,2] = 8*3 = (3+5)*3 = 3*3 + 5*3 = 24$

PAM vs PAM_+



- Summation expresses aggregation of paths
- Multiplication expresses chaining of paths
- If we want **lossless representation** of the tensor, we use **the product of primes** in the 1-hop
- When we **calculate $\mathbf{P}^k$** , we use **the sum of primes** in the 1-hop
- If there are no such cases in the 1-hop these two representations are equal