WalkingTime: Dynamic Graph Embedding Using Temporal-Topological Flows

D Bayani, Carnegie Mellon University (*)
dcbayani@alumni.cmu.edu

(*) Acknowledgements: Thanks to Dr. Reihaneh Rabbany for thoughts and suggestions

Embeddings

- Capture meaningful information via intermediate representation for downstream
  - Debates about what other properties are good
- have long history in language processing, information retrivial
  - tf-idf, LSI/ SVD, latent dirlechet processes, etc.

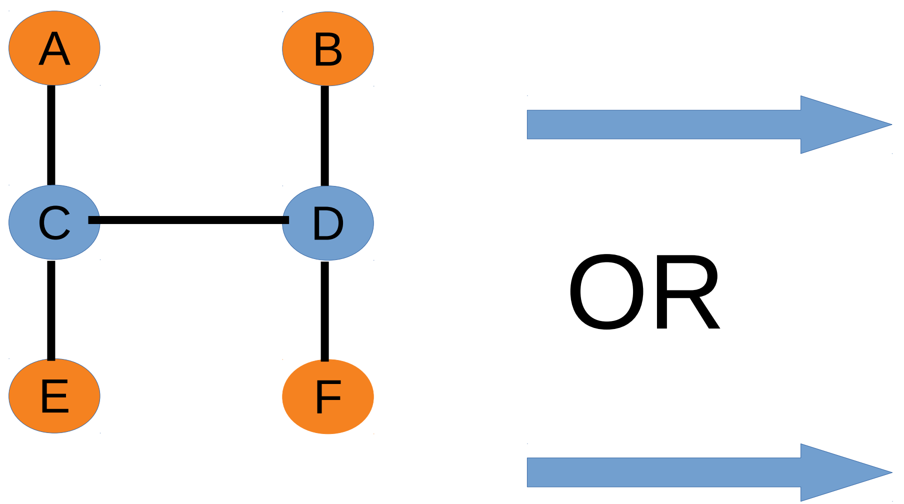
Graph Embeddings

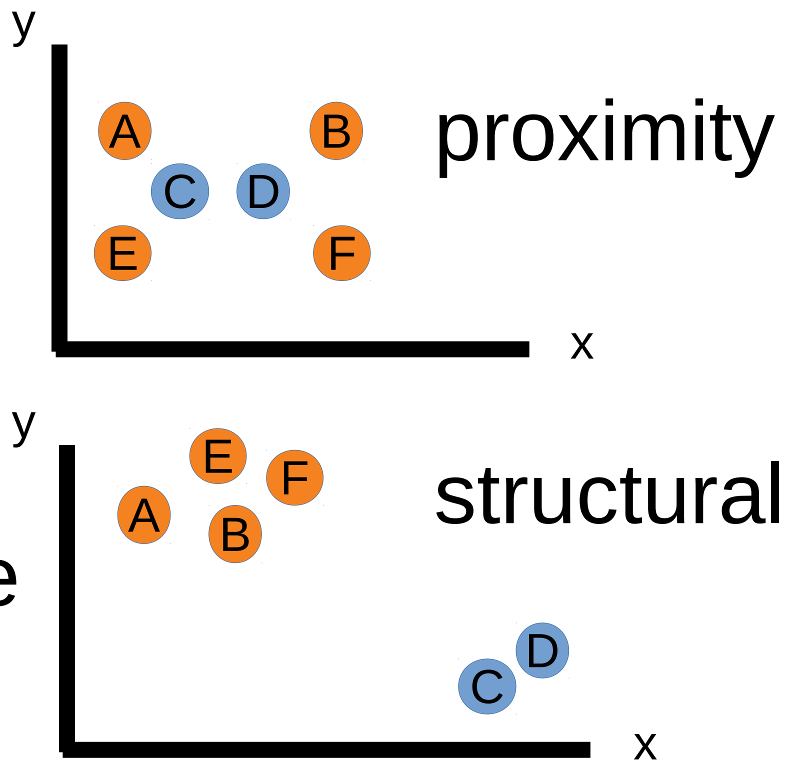
- Have to choose what structures to capture
- Variety of granularities:
  - Whole-graph embeddings
  - Sub-graph embeddings
  - Node embeddings

Focus of this talk

Graph Embeddings

- Emphasis structural similarity or proximity

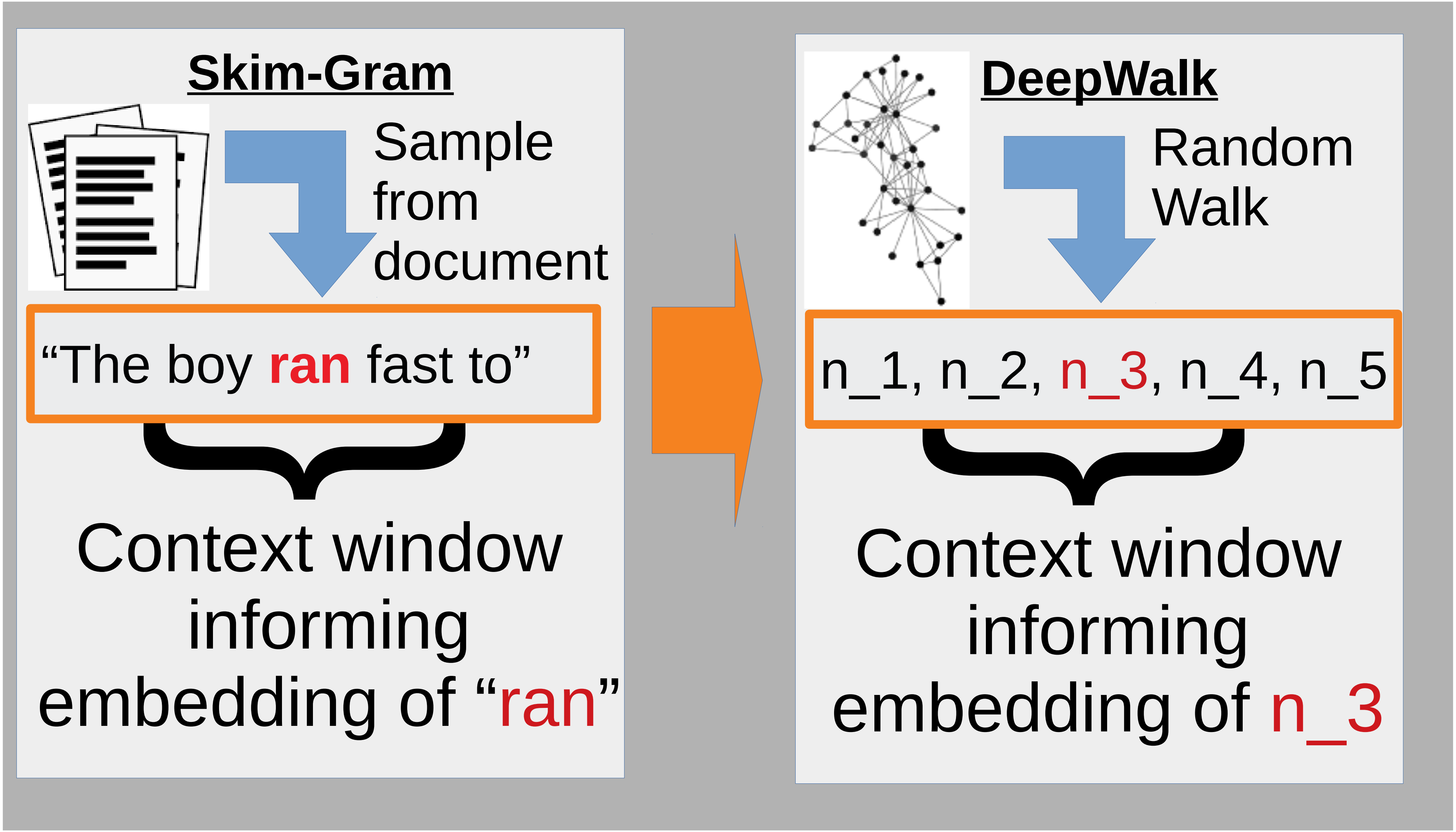




- Can have different distance decays
    - Katz score, A^k

Graph Embedding Techniques

- LLE([11]), Laplacian Eigenmaps ([1])
  - Proximity based, basically matrix factorization
- Autoencoder and convutional neural net approaches
- DeepWalk([10]), node2vec([6])
  - Based on language models, Skip-Gram model ([9]), and rand. walk


Skim-Gram
Sample from document
"The boy ran fast to"
Context window informing embedding of "ran"
DeepWalk
Random Walk
n_1, n_2, n_3, n_4, n_5
Context window informing embedding of n_3

Graph Embedding Techniques

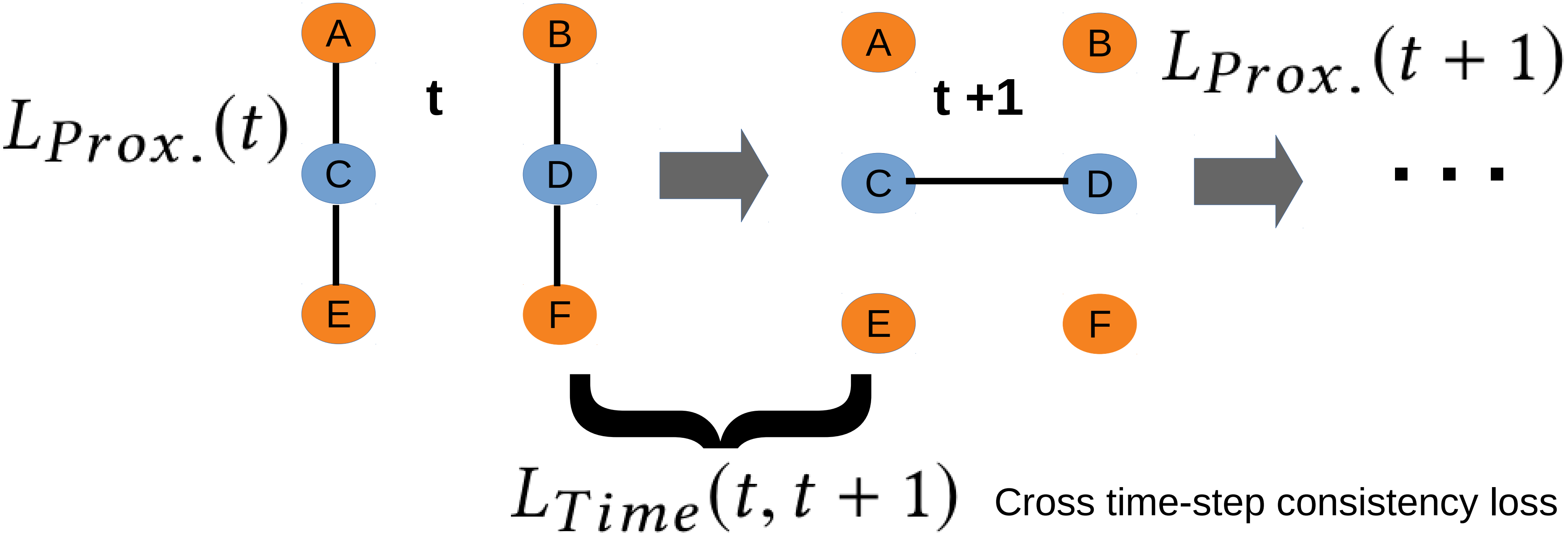
- Many techniques build on node2vec. Ex:
  - Harp ([2])
    - Heirarchy of refinement graphs, repeating embedding to avoid bad local minima
  - Metapath2vec ([4])
    - Bias random walk of node2vec based on edge and node attributes
    - Users provide meta-templates to guide attribute-walks

Temporal Graph Embedding

- Increased attention within last four years
- Motivations from:
  - Disease tracking
  - IoT, autonomous network systems
  - Casuality studies

Temporal Graph Embedding

- Most based on stringing together global snapshots



Temporal Graph Embedding

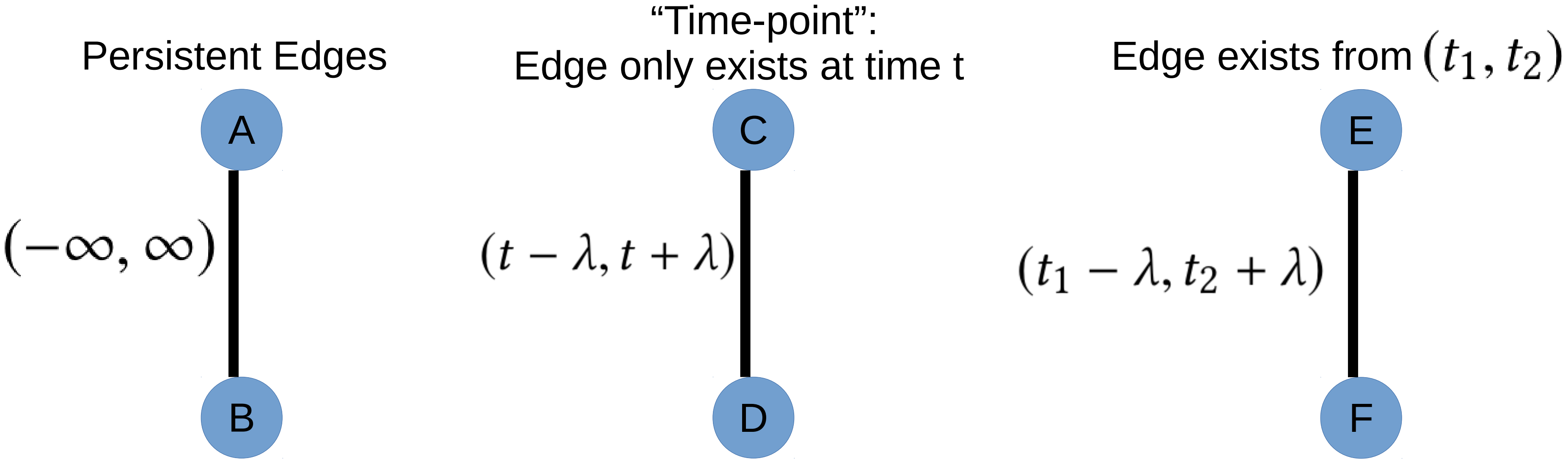
- Most based on stringing together global snapshots

Our Approach: WalkingTime

- Handles time differently:
  - Local
  - Continous
  - Allows forward and backward traversal
- Builds off of node2vec and collection of time-respecting path methods ([7])
  - Technically, handles a multi-graph
  - Maintains set of active times for nodes, only walks to those with overlap

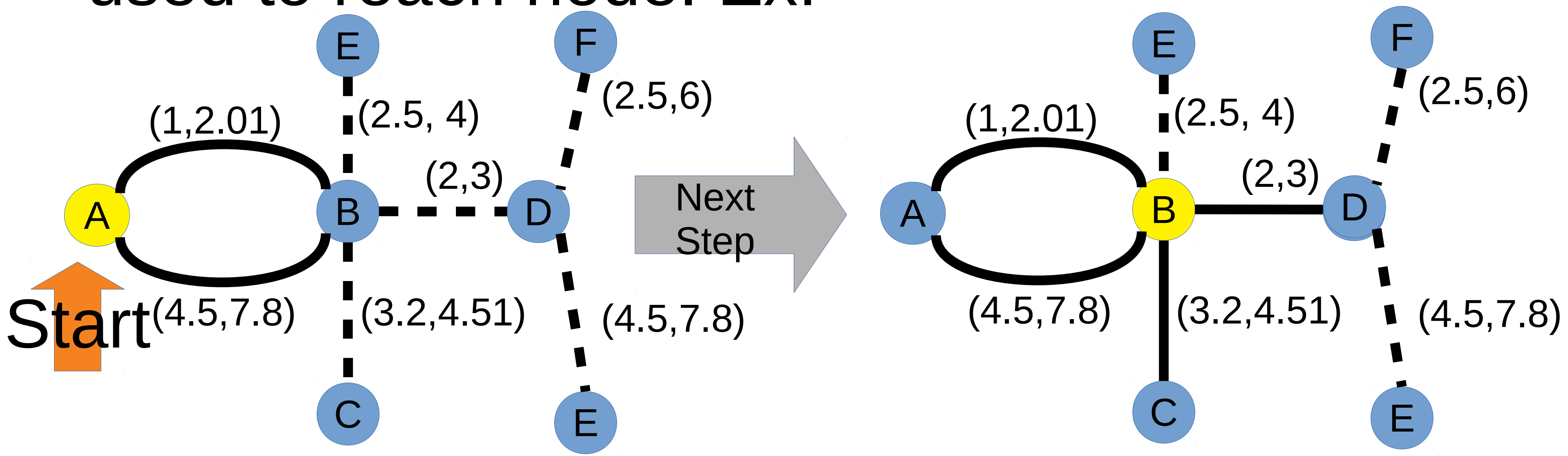
WalkingTime: Pre-processing

- Adds one new parameter compared to node2vec: λ
- Put time intervals on edges



WalkingTime: Random Walk Active Edges

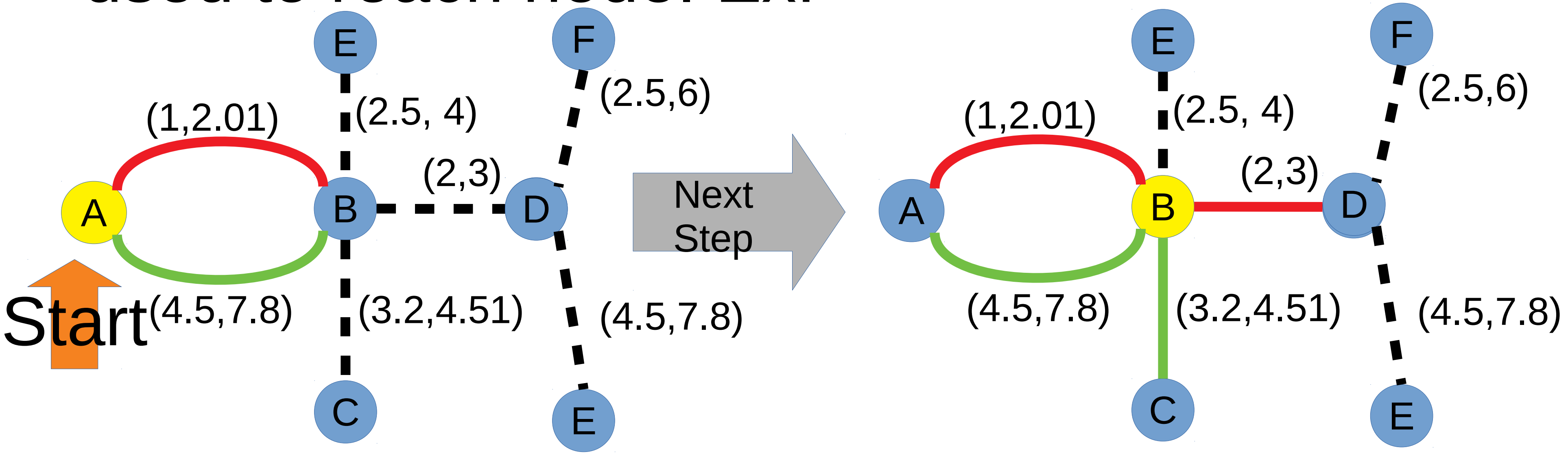
- active edges for each node depends on edges used to reach node. Ex:



Key: **solid line**= "active" edge (can be traversed), **dashed line**="inactive" edge
yellow nodes = node currently on, **blue nodes** = other nodes in the graph

WalkingTime: Random Walk Active Edges

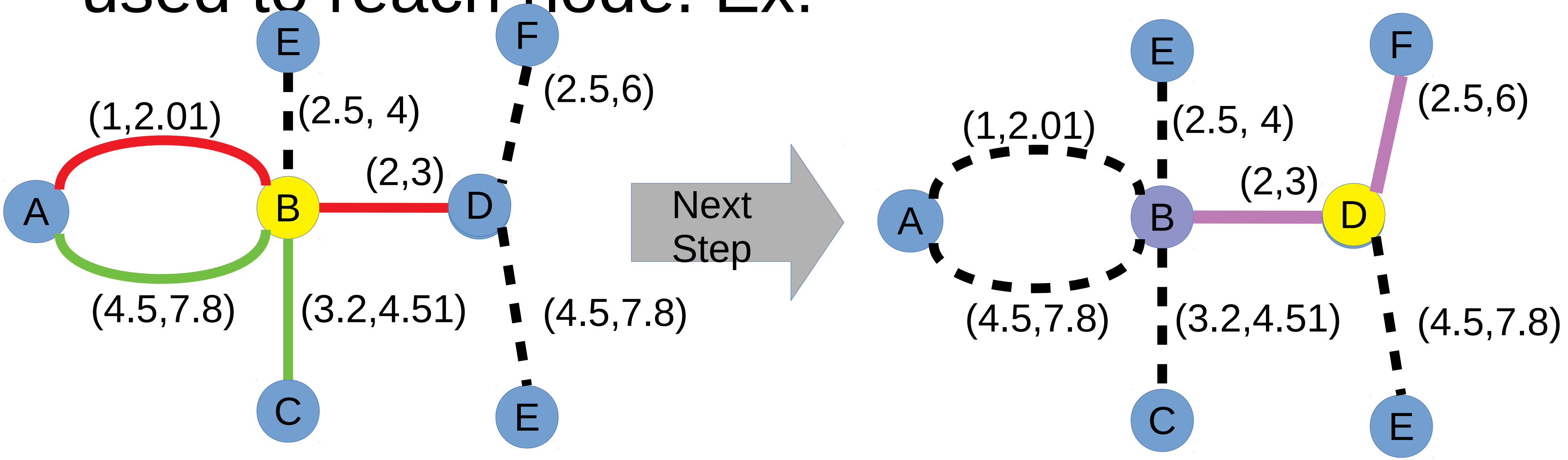
- active edges for each node depends on edges used to reach node. Ex:



Key: **solid line**= “active” edge (can be traversed), **dashed line**=”inactive” edge
yellow nodes = node currently on, **blue nodes** = other nodes in the graph

WalkingTime: Random Walk Active Edges

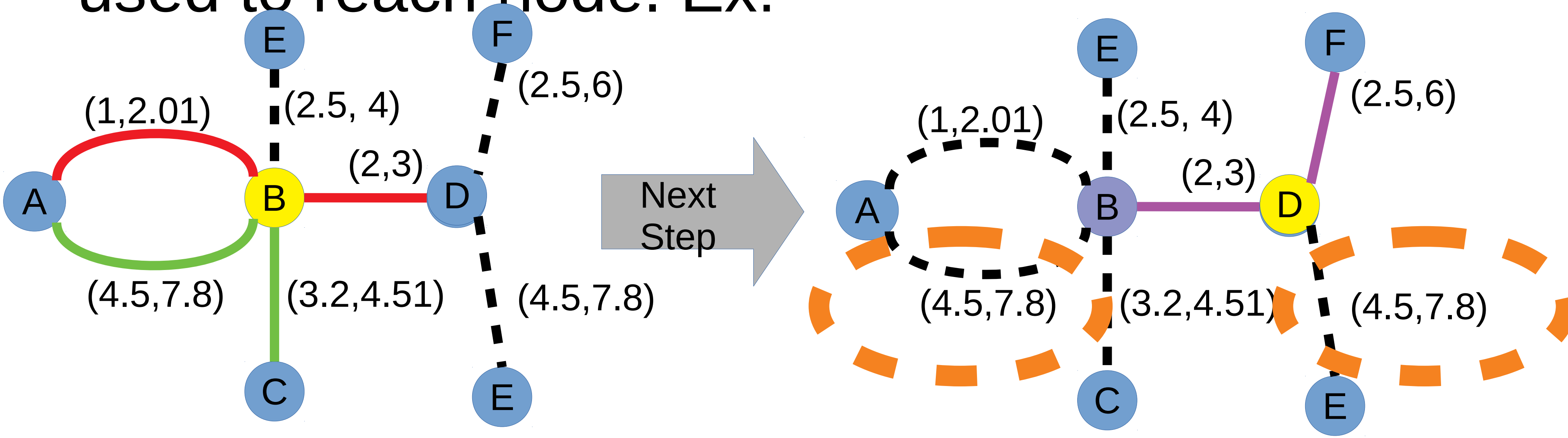
- active edges for each node depends on edges used to reach node. Ex:



Key: **solid line**= “active” edge (can be traversed), **dashed line**=”inactive” edge
yellow nodes = node currently on, **blue nodes** = other nodes in the graph

WalkingTime: Random Walk Active Edges

- active edges for each node depends on edges used to reach node. Ex:



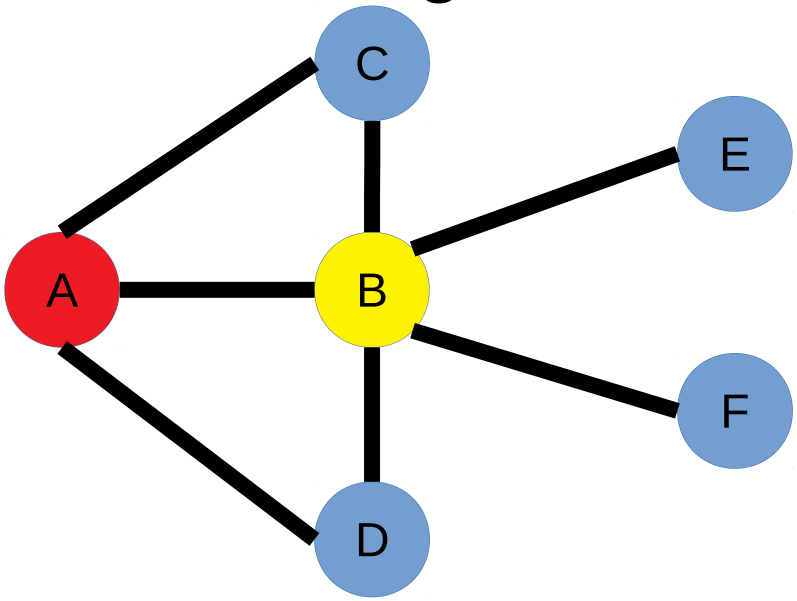
Key: **solid line**= “active” edge (can be traversed), **dashed line**=”inactive” edge
yellow nodes = node currently on, **blue nodes** = other nodes in the graph

WalkingTime: Random Walk Biased Sampling à la node2vec

- node2vec: p and q parameters influences sampling in same neighborhood

Key:
Yellow node=
Current node

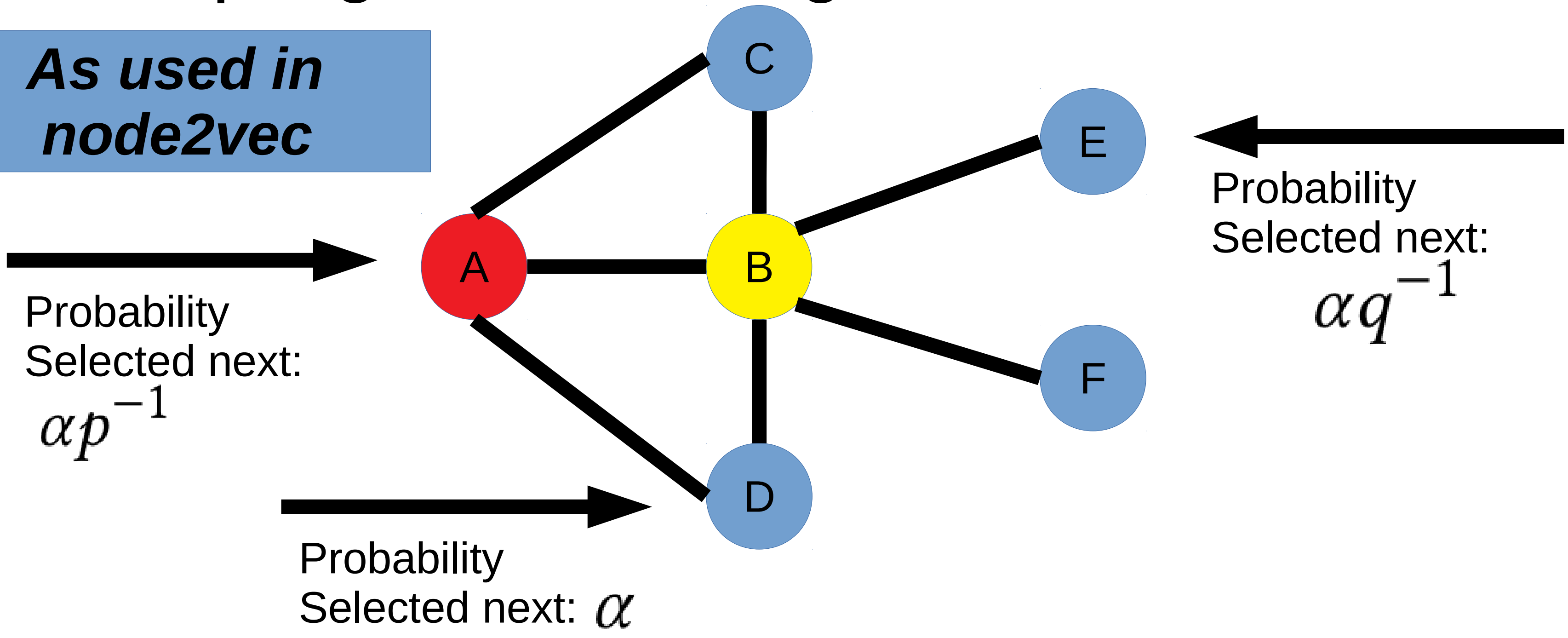
Red node=
Node came from
Last time step



WalkingTime: Random Walk Biased Sampling à la node2vec

- node2vec: p and q parameters influences sampling in same neighborhood

As used in node2vec

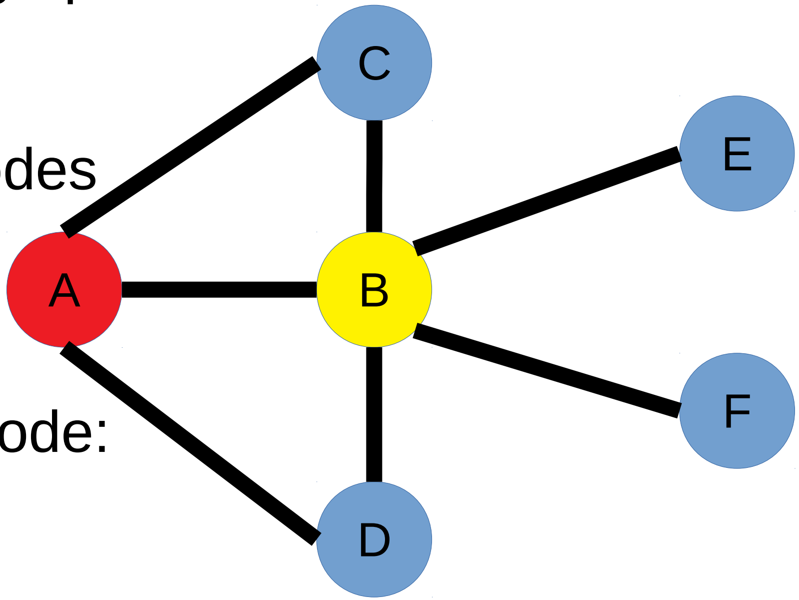


WalkingTime: Random Walk Biased Sampling in Our Method

Added efficiency: reinterprete parameters as rejection sampling probs.

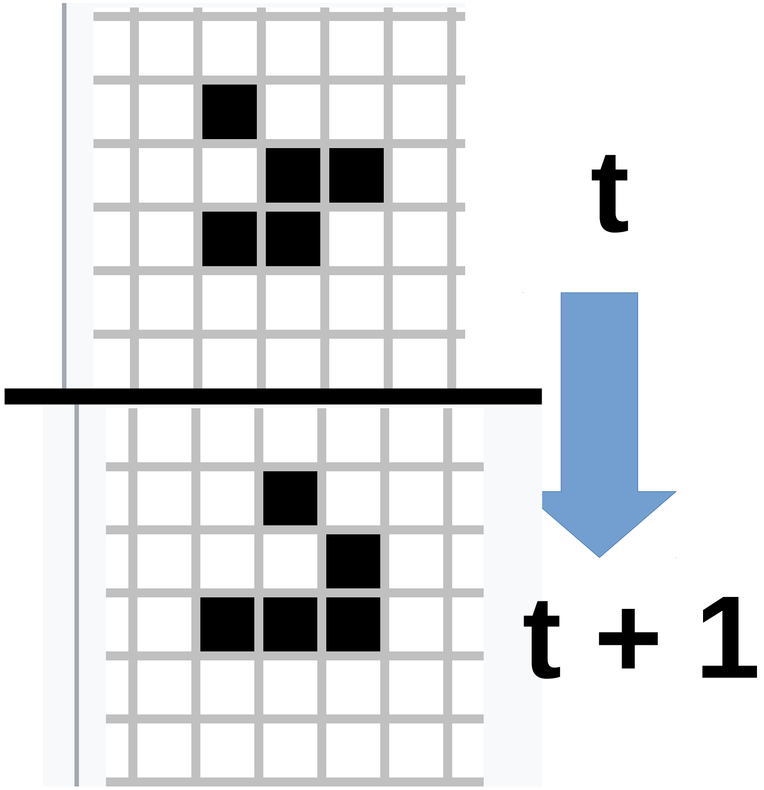
1) Uniform rand. sample a “neighbor in static graph”
   - i.e., edge connects nodes, regardless of if active
2) Find out if there is an active edge linking nodes
   - If not, got to (1)
3) Choose new node with prob. specified by parameters

If sample all nodes and not yet chosen new node: use cached results and alias sampling to do node2vec method



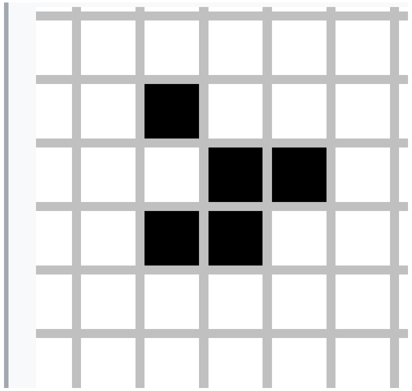
Experiments

- Datasets:
  - Synthetic: Conway's Game of Life ([5])
    - Famous cellular automata
    - One node per grid-cell
    - If two cells share a vertex ***and*** are active within one time-step: form edge

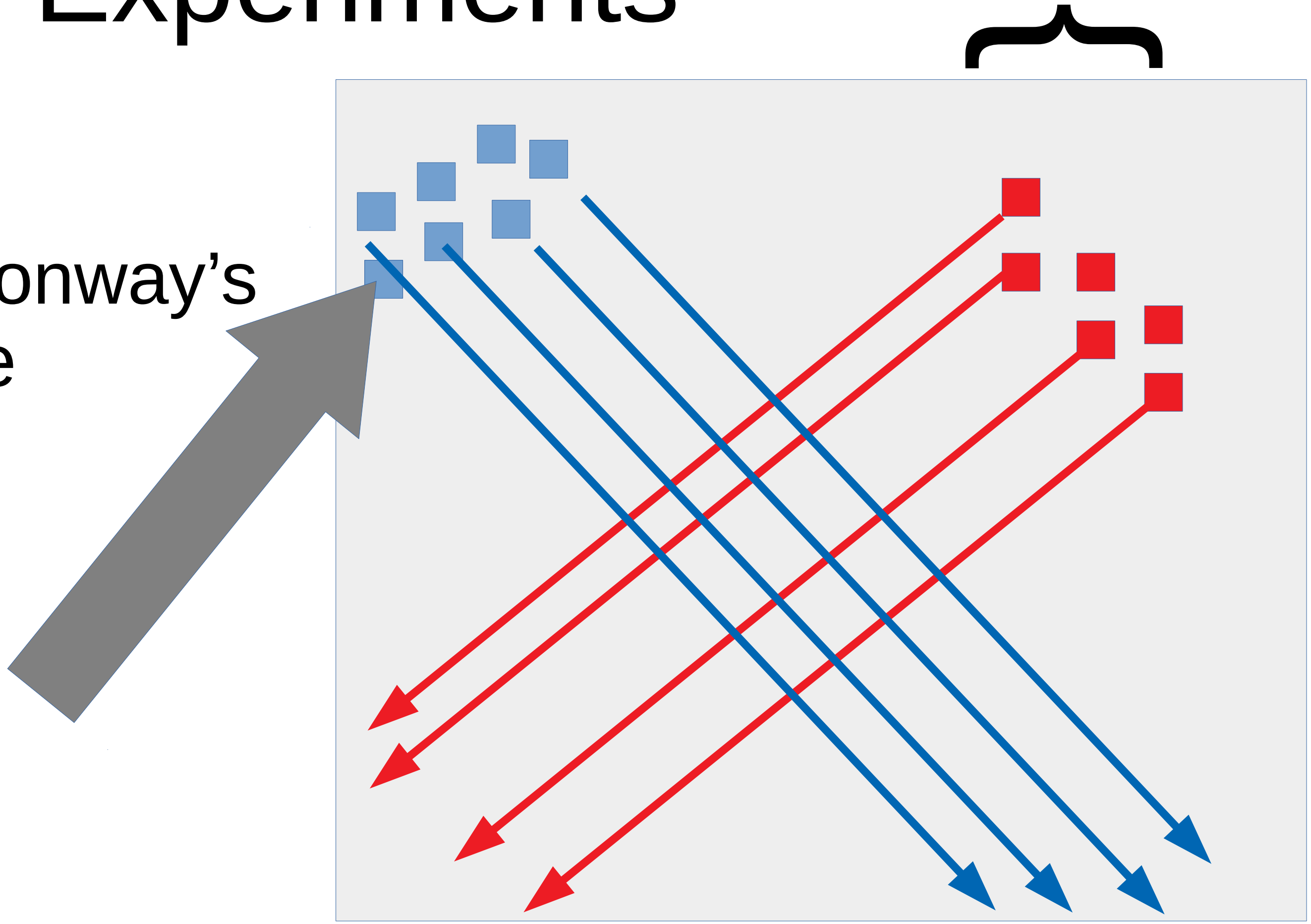


Experiments

- Datasets:
  - Synthetic: Conway's Game of Life

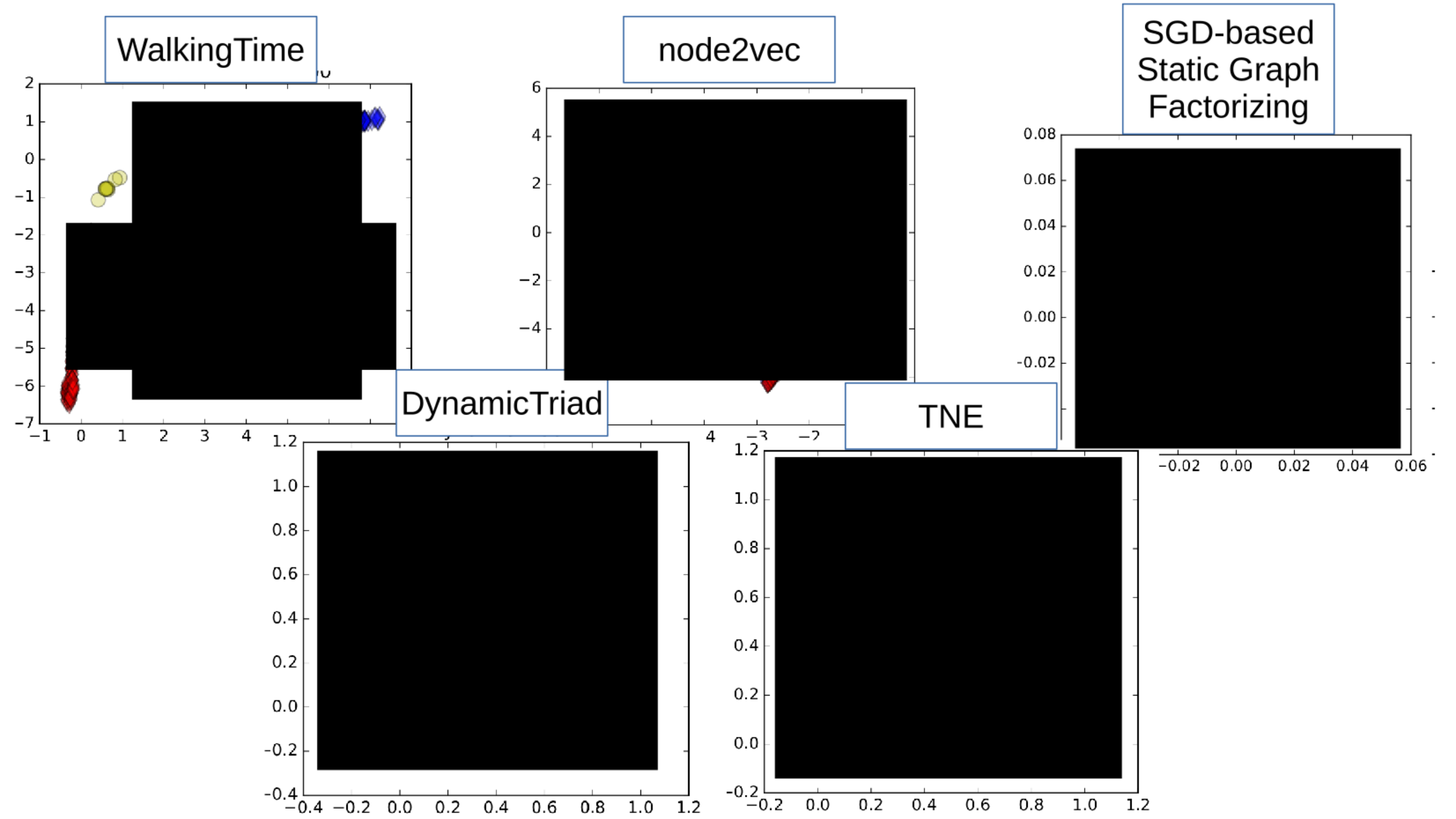


"Glider"

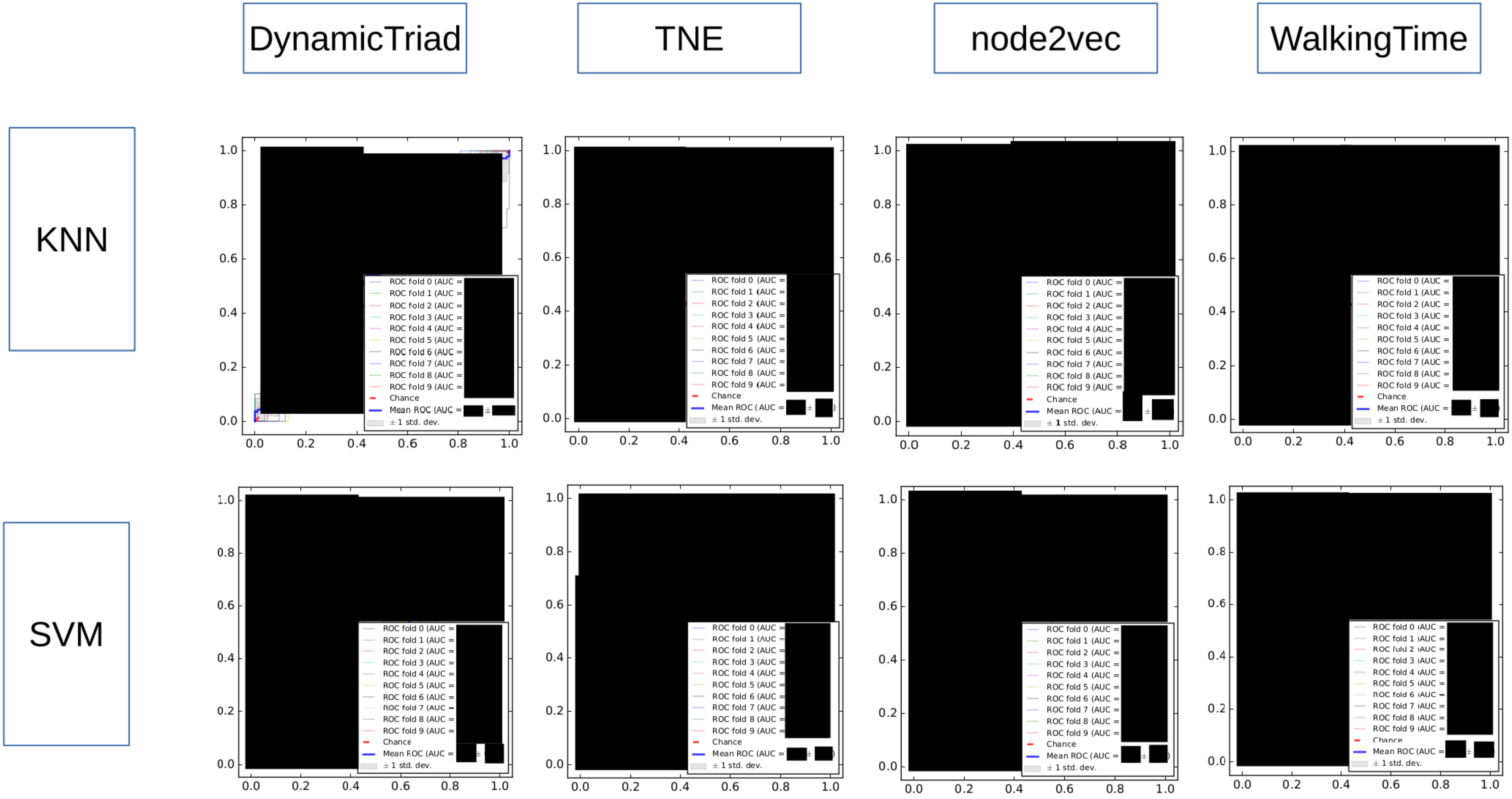


Experiments

- Baseline Algos.:
  - Static graph factorization
  - node2vec
  - TNE ([14])
  - DynamicTriad ([13])
- Evaluation Methods:
  - Node classification
  - 2D Visualization



WalkingTime
2
1
0
−1
−2
−3
−4
−5
−6
−7
−1
0
1
2
3
4
node2vec
6
4
2
0
−2
−4
4
−3
−2
SGD-based Static Graph Factorizing
0.08
0.06
0.04
0.02
0.00
-0.02
−0.02
0.00
0.02
0.04
0.06
DynamicTriad
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
TNE
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2


DynamicTriad
TNE
node2vec
WalkingTime
KNN
SVM
ROC fold 0 (AUC = [redacted]
ROC fold 1 (AUC = [redacted]
ROC fold 2 (AUC = [redacted]
ROC fold 3 (AUC = [redacted]
ROC fold 4 (AUC = [redacted]
ROC fold 5 (AUC = [redacted]
ROC fold 6 (AUC = [redacted]
ROC fold 7 (AUC = [redacted]
ROC fold 8 (AUC = [redacted]
ROC fold 9 (AUC = [redacted]
Chance
Mean ROC (AUC = [redacted] ± [redacted]
± 1 std. dev.

Further Experiments

- Also trying on DBLP ([8,12]) and Higgs-Twitter ([8,3])

Further Experiments

- On-going works:
  - Latent Graph Reconstruction
  - Link Prediction
- Finding datasets that clear and numerous cause-effect relations in lab sciences

References

[1] Mikhail Belkin and Partha Niyogi. 2002. Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. In *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani (Eds.). MIT Press, 585–591. http://papers.nips.cc/paper/1961-laplacian-eigenmaps-and-spectral-techniques-for-embedding-and-clustering.pdf

[2] Haochen Chen, Bryan Perozzi, Yifan Hu, and Steven Skiena. 2017. HARP: hierarchical representation learning for networks. *arXiv preprint arXiv:1706.07845* (2017).

[3] Manlio De Domenico, Antonio Lima, Paul Mougel, and Mirco Musolesi. 2013. The anatomy of a scientific rumor. *Scientific reports* 3 (2013), 2980.

[4] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 135–144.

[5] M. Gardner. 1970. Mathematical Games. *Scientific American* 223 (Oct. 1970), 120–123. https://doi.org/10.1038/scientificamerican1070-120

[6] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[7] Petter Holme and Jari Saramäki. 2012. Temporal networks. *Physics reports* 519, 3 (2012), 97–125.

[8] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data. (June 2014).

[9] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[10] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, New York, NY, USA, 701–710. https://doi.org/10.1145/2623330.2623732

[11] Sam T. Roweis and Lawrence K. Saul. 2000. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science* 290, 5500 (2000), 2323–2326. https://doi.org/10.1126/science.290.5500.2323 arXiv:http://science.sciencemag.org/content/290/5500/2323.full.pdf

[12] Jaewon Yang and Jure Leskovec. 2012. Defining and Evaluating Network Communities based on Ground-truth. (2012). arXiv:cs.SI/1205.6233

[13] L. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang. 2018. Dynamic Network Embedding by Modelling Triadic Closure Process. In *AAAI*.

[14] L. Zhu, D. Guo, J. Yin, G. V. Steeg, and A. Galstyan. 2016. Scalable Temporal Latent Space Inference for Link Prediction in Dynamic Social Networks. *IEEE Transactions on Knowledge and Data Engineering* 28, 10 (Oct 2016), 2765–2777. https://doi.org/10.1109/TKDE.2016.2591009