

Dynamic Graph Clustering with Variable Time Resolution

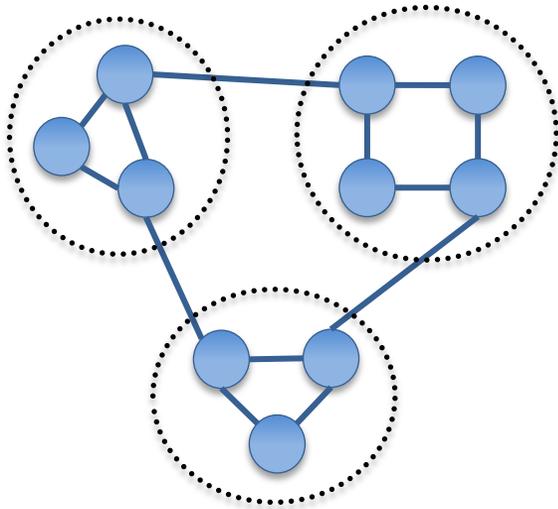


2/26/2019

Timothy La Fond
CASC Postdoctoral Researcher



Graph Clustering: Find Communities of Nodes



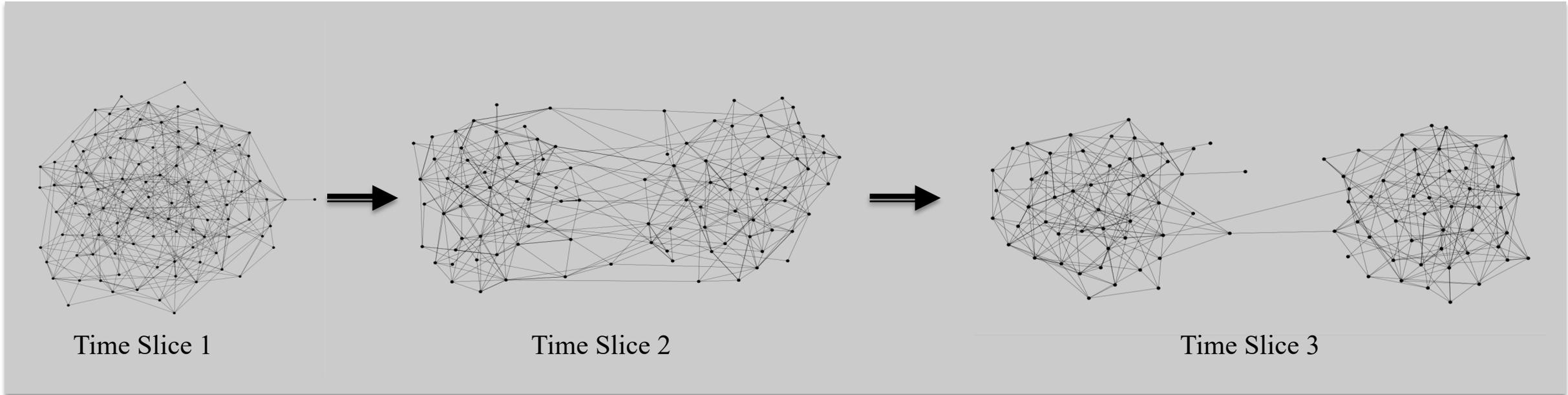
Community/Cluster: Group of associated nodes

Community structure critical to analysis of the network

Poulin, Th  berge. "Ensemble Clustering for Graphs." *International Workshop on Complex Networks and their Applications*. 2018

La Fond et al. "An Ensemble Framework for Detecting Community Changes in Dynamic Networks." *High Performance Extreme Computing Conference*. 2017

Clustering a Dynamic Network

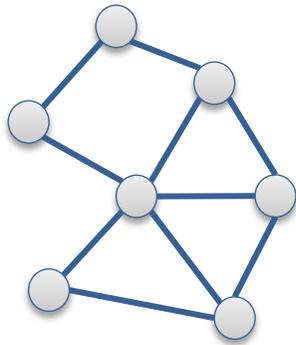


Treat Network as sequence of static graph instances using time slices

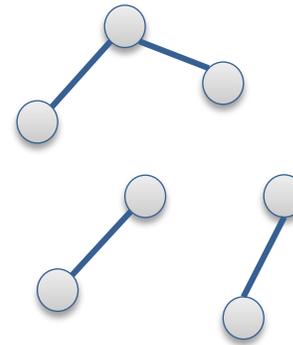
Cluster each instance independently

If data is timestamped stream of communications, slice width is an arbitrary parameter

Slices Too Narrow: Internally Disconnected Communities



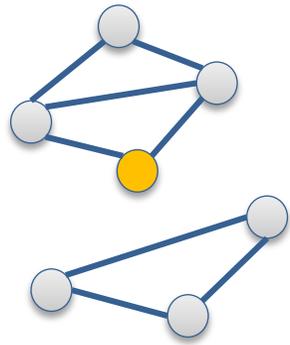
Ideal



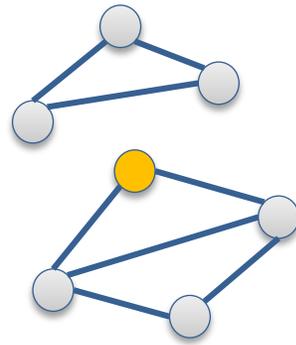
Too Sparse

Minimum amount of edge observations required to represent cluster structure

Slices Too Wide: Fail to Detect Behavior

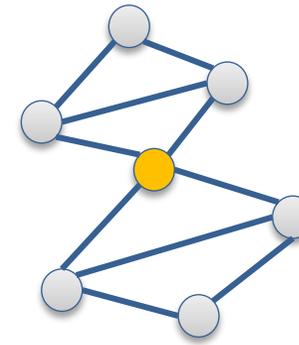


Time Slice 1



Time Slice 2

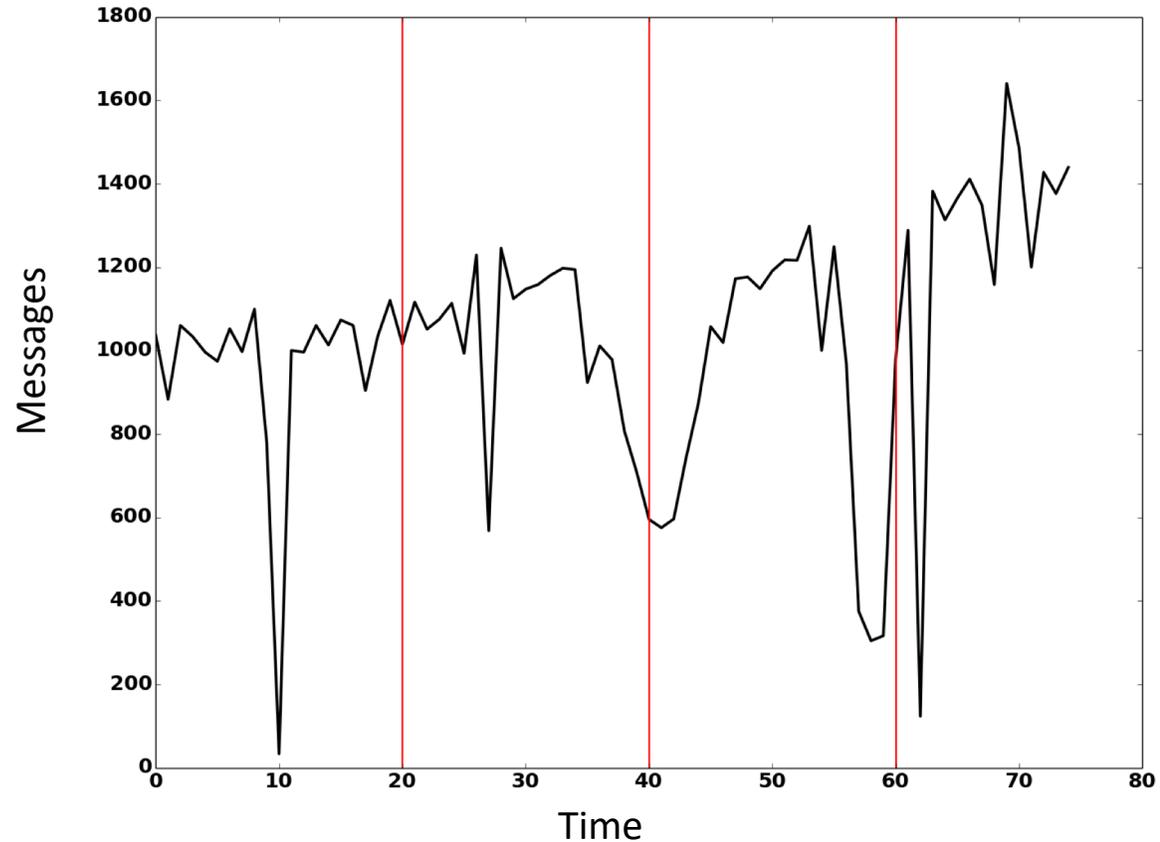
VS



Single Time Slice

Wide time slices blur behavior, fail to detect changes promptly or at all

Choice of Time Slicing is Critical

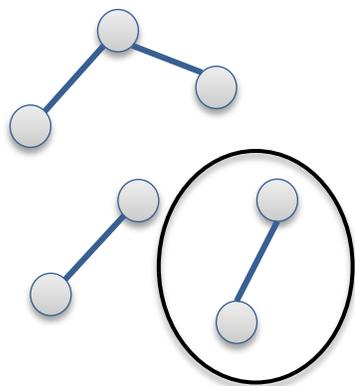


Too narrow: sparse, internally disconnected

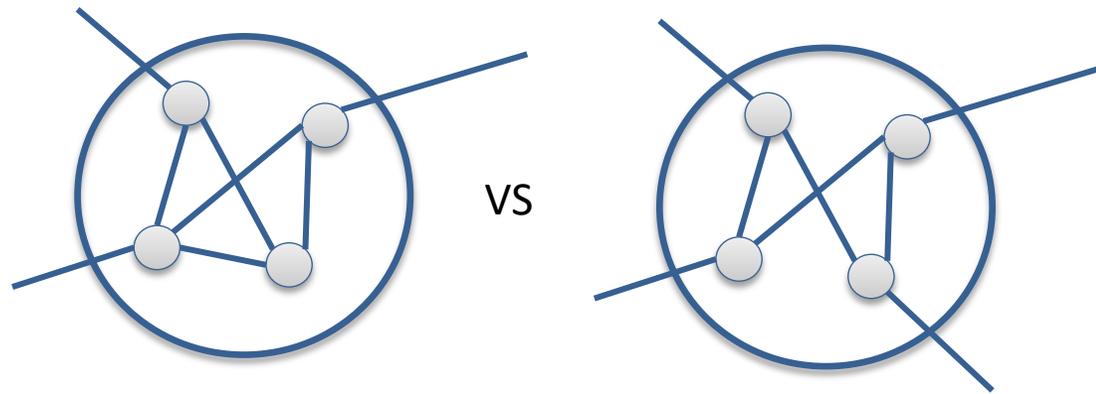
Too wide: granularity is poor, merge events

Don't want to split in the middle of events; ideally place at the boundary of events

Option 1: Prefer Minimum Slice Width without Internally Disconnected Communities



Trivial Cluster



Trivial Cluster: not significantly better than 50/50

Optimize the granularity by finding narrowest slices that still represent structure

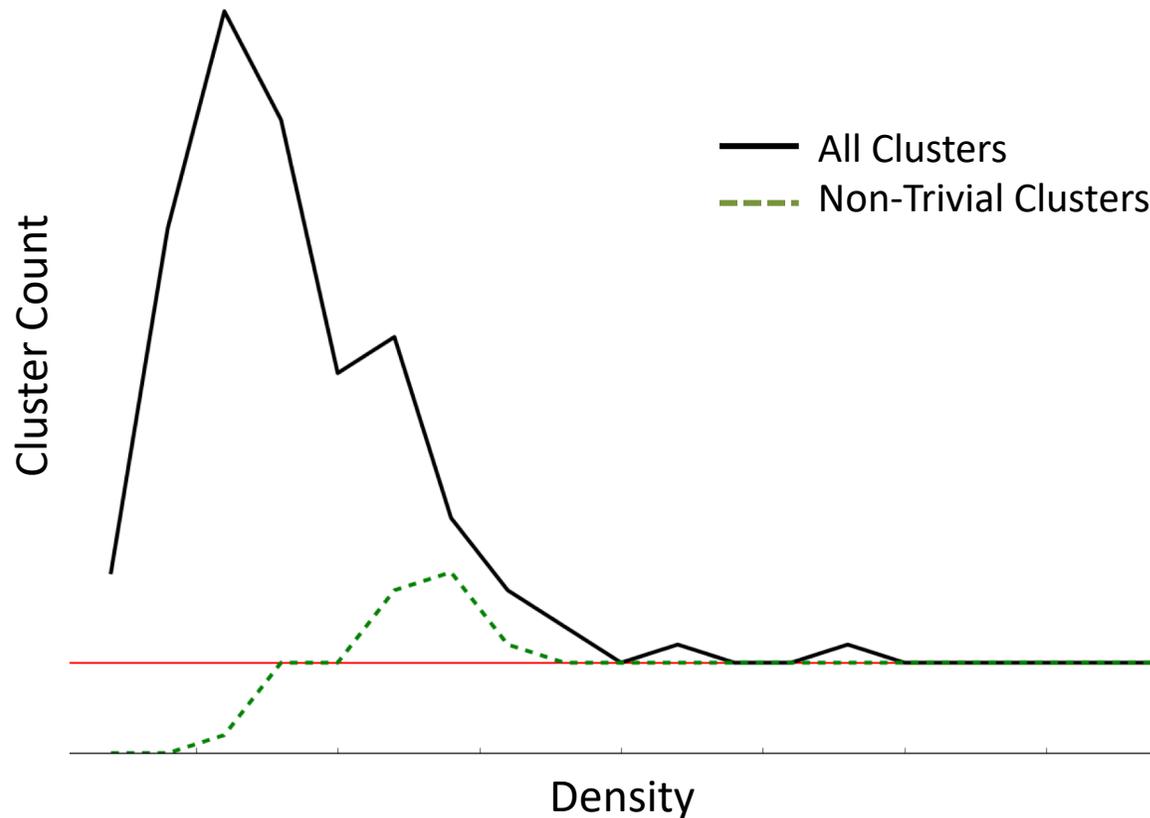
Use metric to determine when internal disconnections are likely occurring

Example: internal vs. external edge ratio should be significantly greater than θ (e.g. 50/50)

Garcia et al. "Applications of community detection techniques to brain graphs: Algorithmic considerations and implications for neural function."
Proceedings of the IEEE 106.5 (2018): 846-867

Synthetic Example: Trivial Clusters Detected

Trivial Clusters as a function of Density



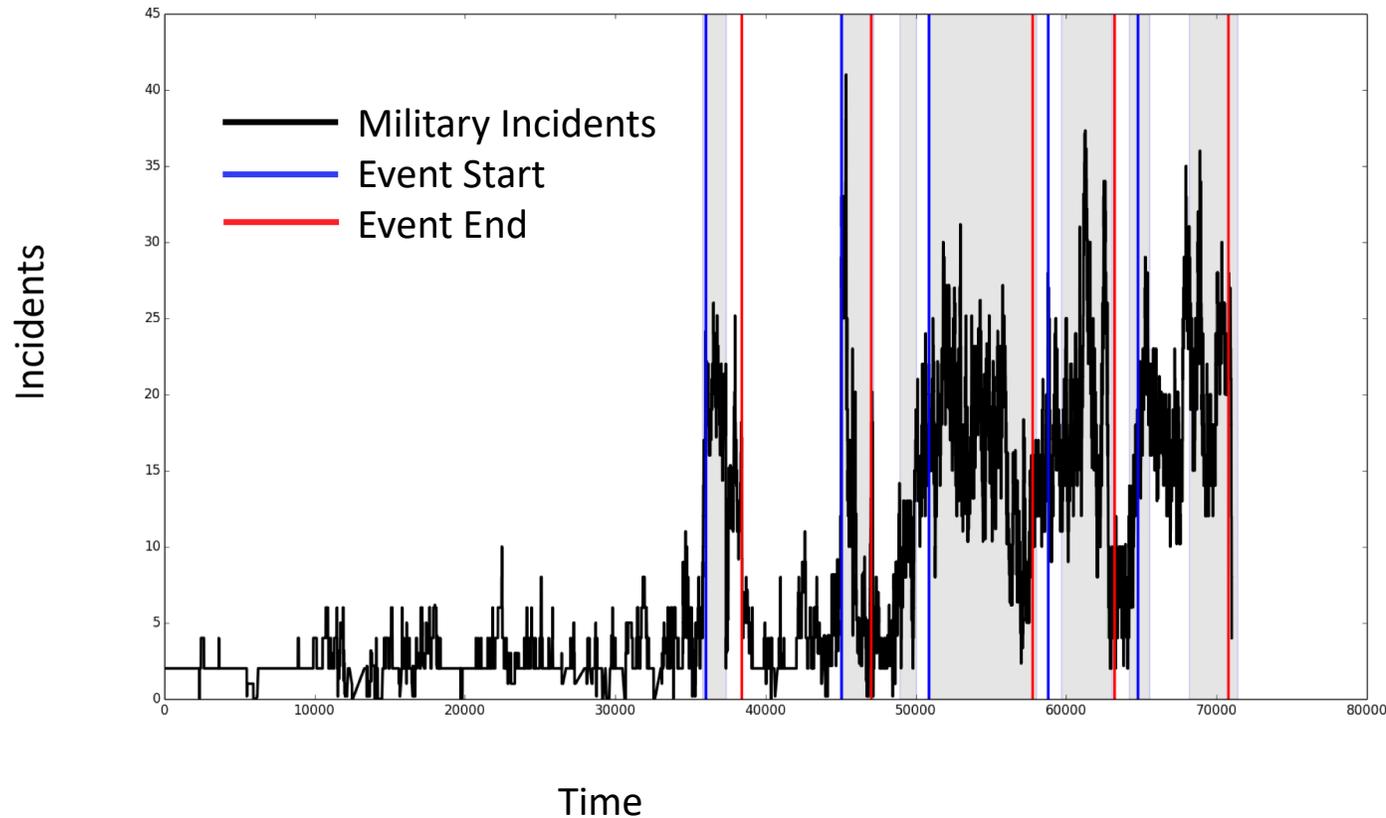
Generate data from stochastic block model (n=100, C=5) at different densities

“Trivial” clusters fail metric

Trivial clusters found increases as density decreases

If proportion of trivial to non-trivial is too high a wider slice is needed

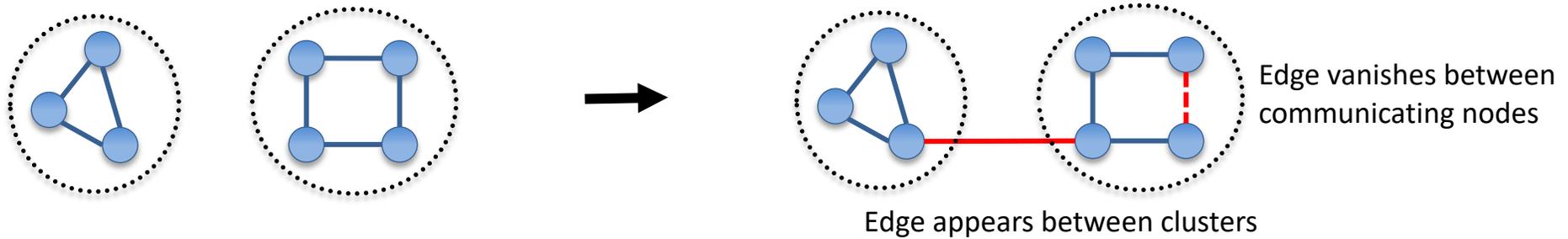
Option 2: Trigger Functions to Slice at Change Points



- Place a slice only when the graph is changing
- Ensures slices are homogenous periods of time
- Minimizes slices to cluster

Correlates of War Project: www.correlatesofwar.org

Automatic Triggering via Cluster Structure



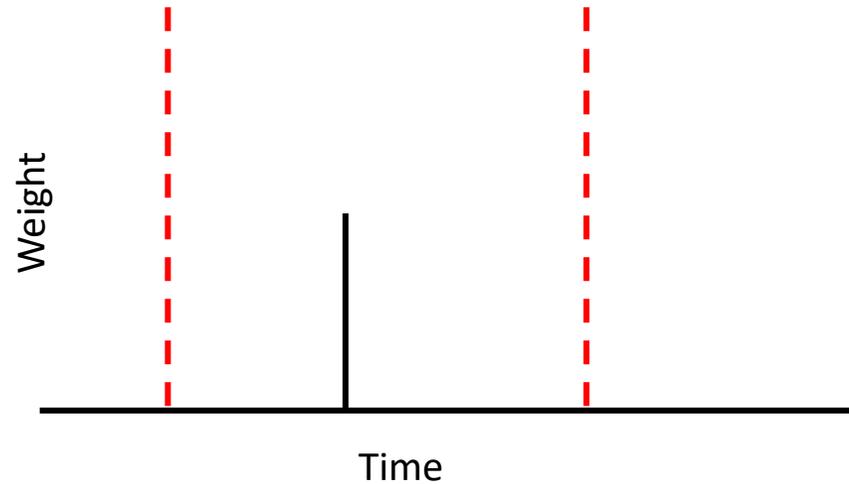
Tally events that indicate structure change – Cross-Cluster Edges and Edge Timeout

Cross-Cluster Edges are any edges appearing between existing clusters and might indicate a merge or a migration

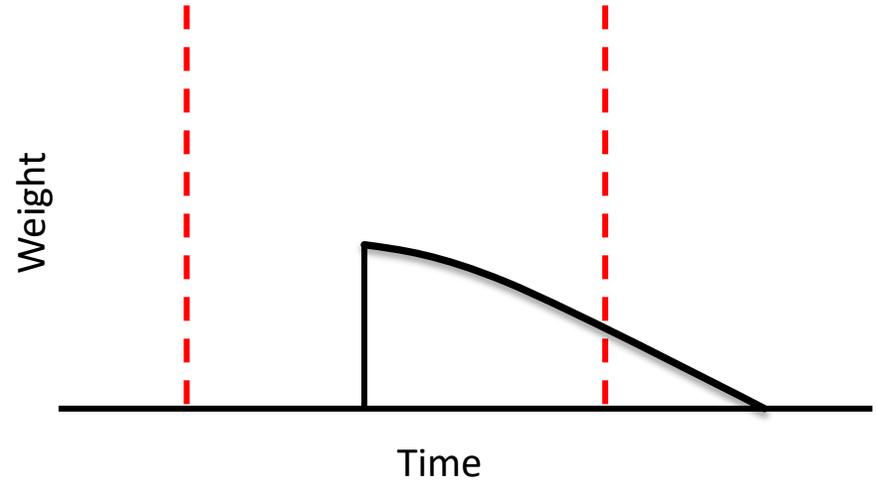
Edge Timeout is when a pair of nodes in the same cluster don't communicate again after a certain period of time, and might indicate a partition has formed

Tally two cumulative sums of each of these events and re-cluster whenever either exceeds a threshold

Option 3: Decay Function on Edges



Original: Edge only applies to the time slice it falls in

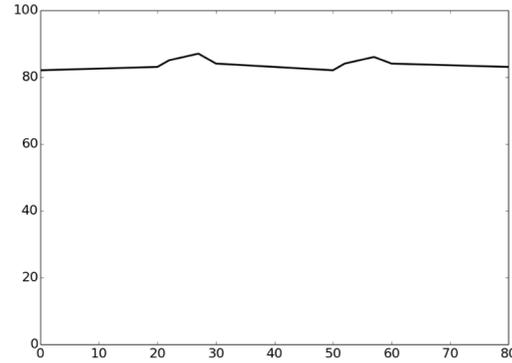
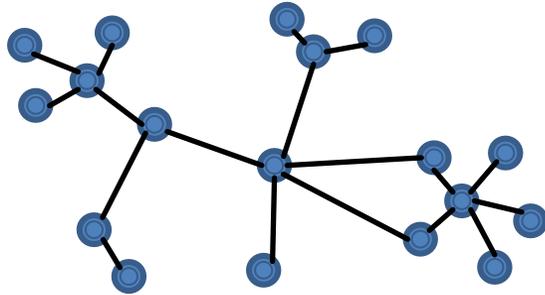


Decay Function: Edge has (decaying) weight which applies to other time slices

Cut off edge when weight has decayed too low

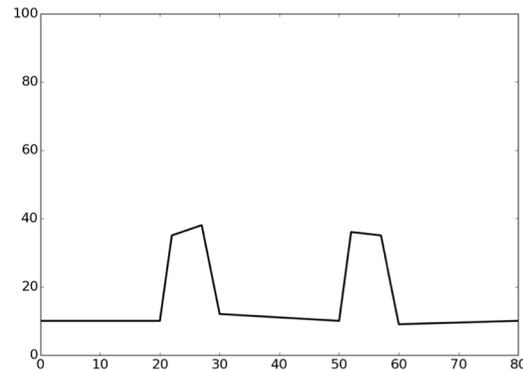
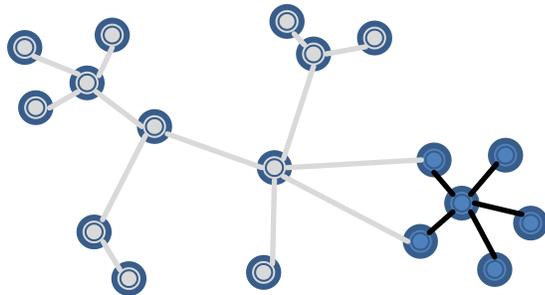
Using an appropriate decay function allows for arbitrarily narrow time slices

Decay Functions Can Accommodate Data with Multiple Time Scales



Example: Humans and Machines in network

Slices wide enough for one are too narrow for other and vice versa



Need local time representation

Give less active entities slower decays on their edges to avoid sparsity

Summary

Choice of how to handle time dimension greatly affects analysis of dynamic networks

Ideally represent all changes in network with fine granularity without errors due to sparsity

Presented multiple methods here for placing time slices but ultimately they are difficult to tune and very data-dependent

Local regions operating at different time scales present an additional challenge

Best approach is still unclear; remains a difficult problem



CASC

Center for Applied
Scientific Computing



**Lawrence Livermore
National Laboratory**

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.