Machine Learning Algorithms for Graph-Based Representations of Fracture Networks

Los Alamos National Laboratory

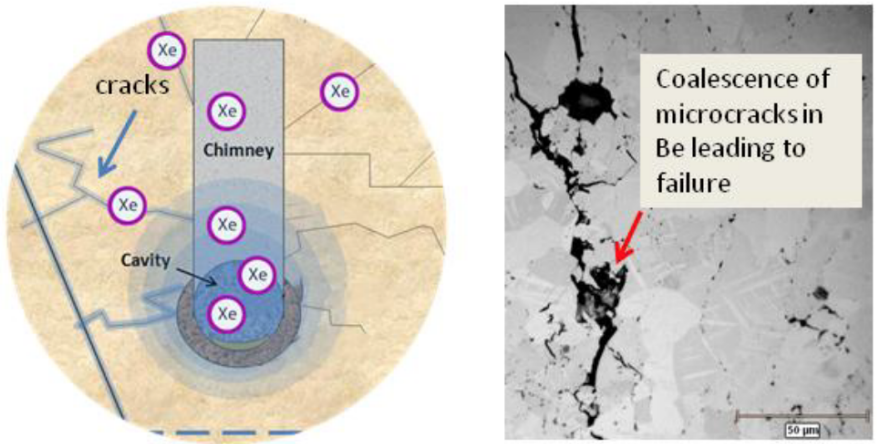
Students: Adrian Cantu, Michael Guo, Priscilla Kelly, Sean Matz, Manuel Valera[PM]
Faculty Advisor: Allon Percus
Liaisons: Jeffrey Hyman, Gowri Srinivasan, and Hari Viswanathan

Claremont Graduate University

November 29, 2016

Background

- Fracture development important to reserves which can develop flow[1, 2]
  - Water Reservoirs
  - Shale Gas extraction
  - Nuclear Test Detection
- Determine dominating physics for fracture propagation and flow through fractures in rock



Gas Migration from a Nuclear Test
- Static fractures with gas flow

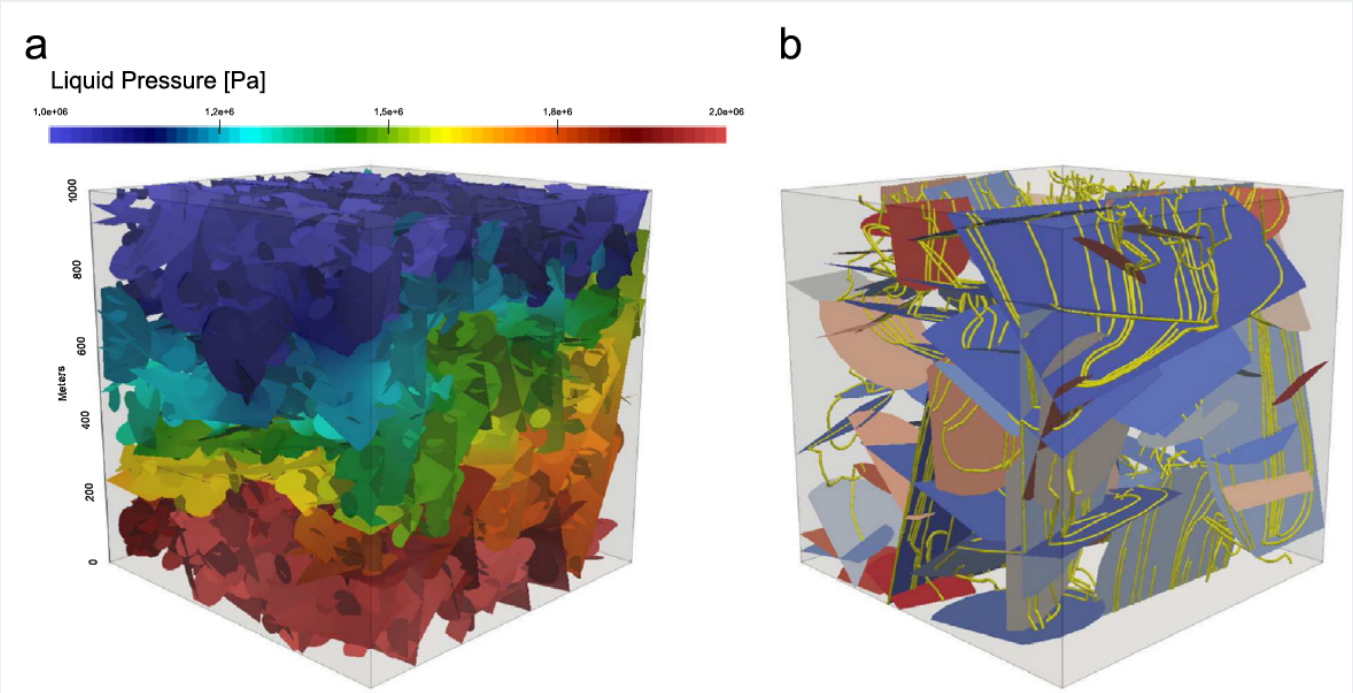
LANL Brittle Fracture Experiment
- Dynamic fracture growth

Figure 1. Gas migration and brittle failure both involve fractures.

[1] Jordan, et al., Nat. Sci. Rep. 5, 2015
[2] Cady, et al., EPJ Web of Conf., 2012

Background

- To model fractures **Los Alamos National Lab** developed two high-resolution high performance computing models: dfnWorks[3] and HOSS
  - Networks based on statistics taken from real fracture networks



[3] Hyman, et al., Comp. and Geosci., 84, 2015

Background

LANL Suite

- **dfnWorks:** a static DFN generation code coupled to massively parallel flow and transport simulation tool PFLOTRAN
- **HOSS:** a high-fidelity mesoscale implementation of FDEM (Finite Element Discrete Element Method) for crack propagation in brittle geomaterials
- In subsurface applications, fracture networks could typically contain 10^6 fractures with an average of 1000 elements needed to represent each fracture resulting in billions of unknowns.

Background

To reduce computational complexity, the physics of gas migration through a 3D fracture network can be represented as a flow problem on a graph without compromising the essential geometry.[4]

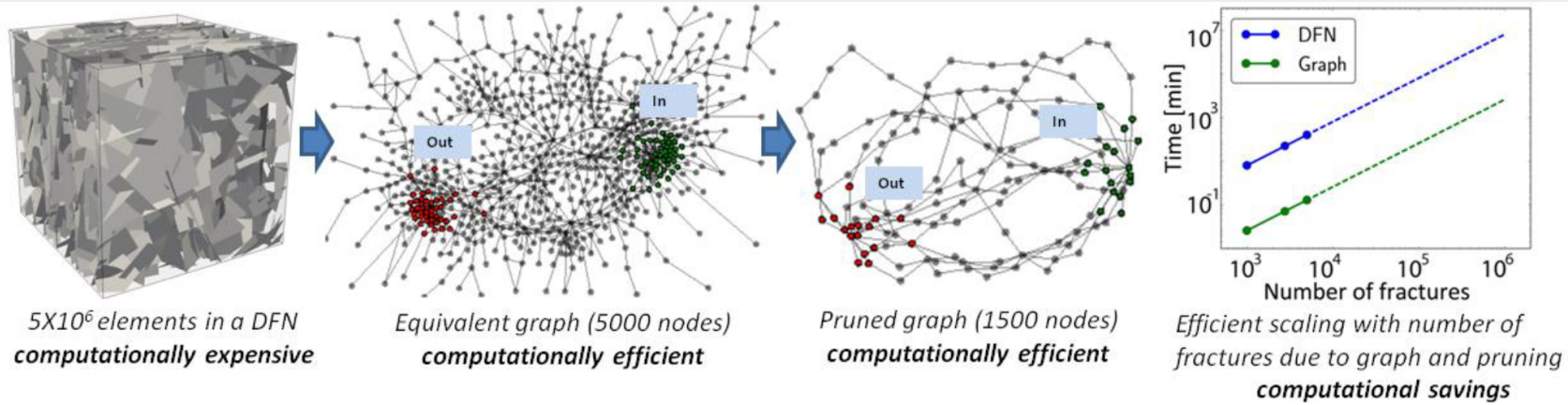


Figure 2. Preliminary results from the Forsmark site, Sweden to identify the critical region in the Discrete Fracture Network (DFN). The plot shows 3 orders of magnitude improvement for the graph-based flow simulation. Dashed lines show estimated scaling as number of fractures increases.

[4] Srinivasan, Tech. Rep., LANL 2016

Math Clinic Goals

How can we fit in?

We want to reduce the complexity and computational costs of physics simulations by representing fracture networks as graphs.

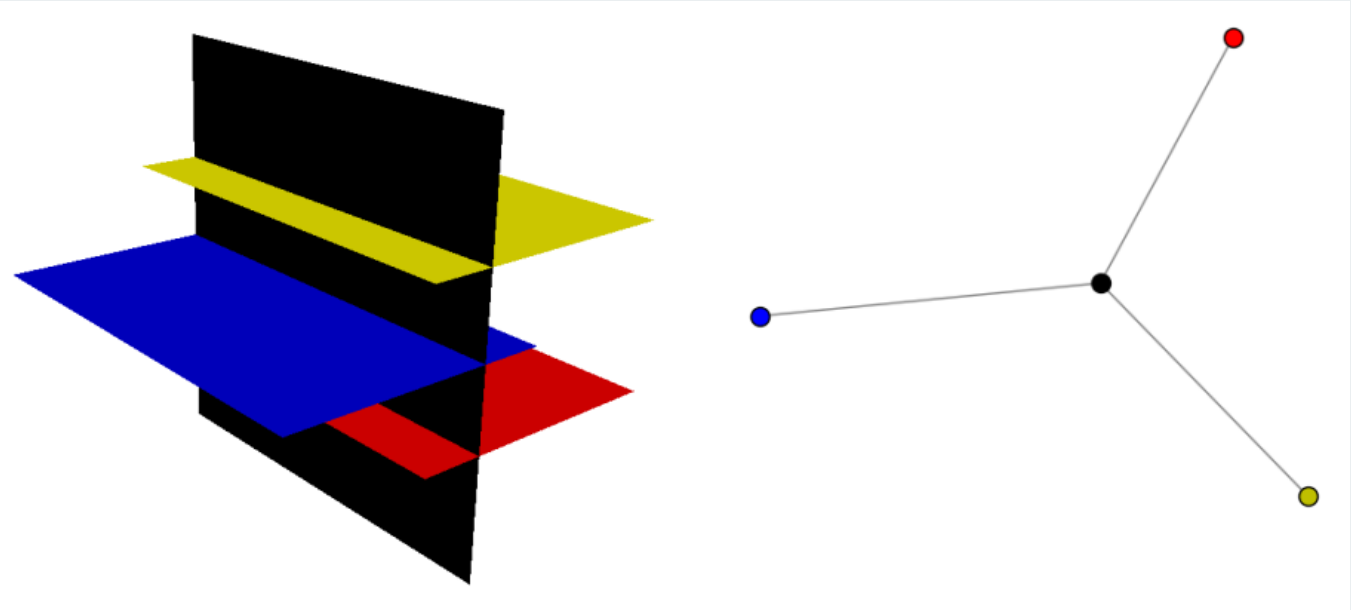
- Predict flow backbone based on graph topology and underlying network geometry
- Predict fracture propagation using dynamic graph methods

Our tasks:

- Define appropriate node-based quantities and weights for prediction of physical phenomena on the graph, validated using dfnWorks and HOSS
- Use machine learning algorithms to predict and develop these phenomena, based on training data from high-fidelity simulations

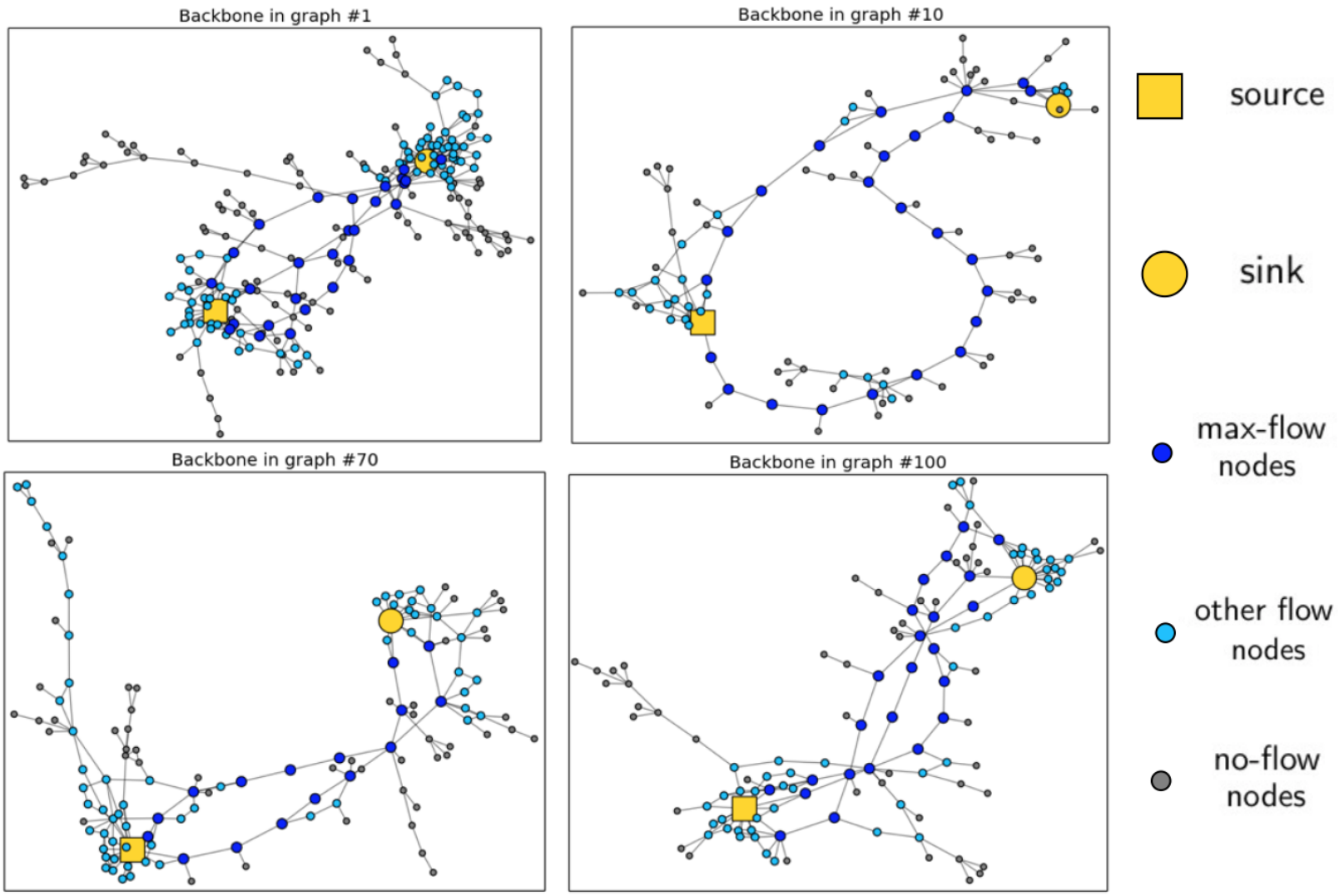
DFN Graphs Representation

- To translate generated networks to graphs[5]
  - Fractures becomes nodes
  - Intersections between fractures are edges
- In this form, flow calculation is at least 3 o.m. faster



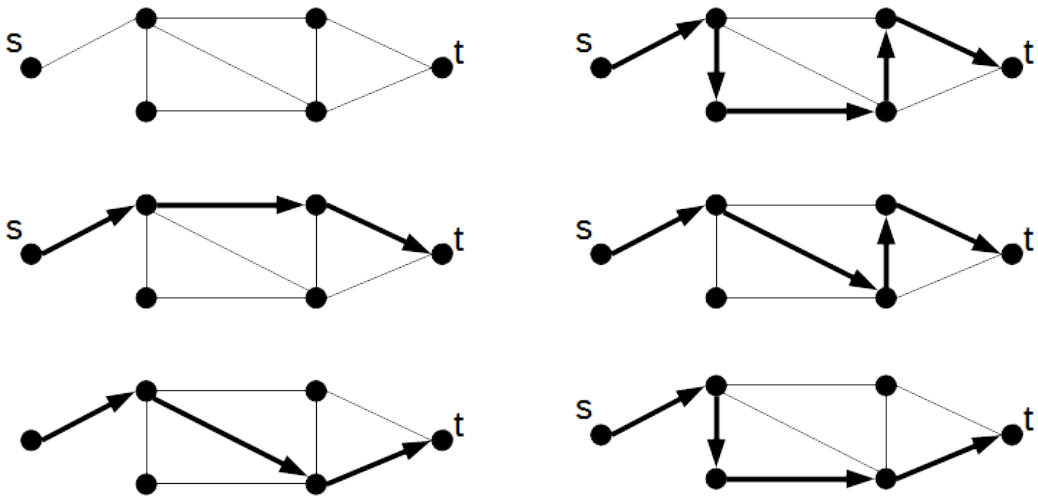
[5] Hyman, 2016

Sample dfnWorks Graphs

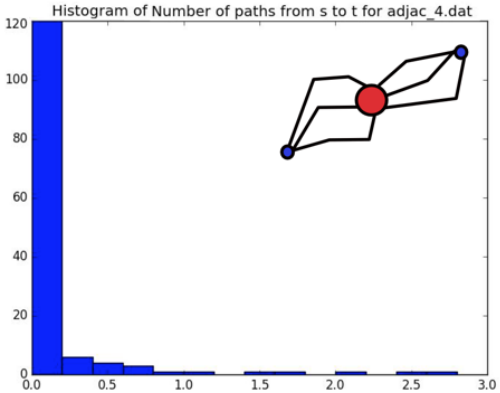


Max-Flow: How the backbone was found ?

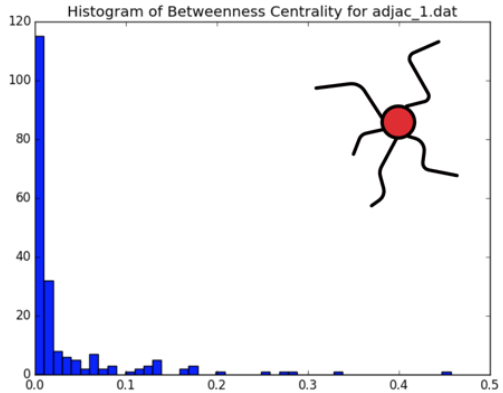
- Max-flow backbone nodes are determined once the graphs are generated as shown to the left.
- This backbone is **arbitrary**
  - Edges have a weight of 1
- This is a caveat for ML methods.



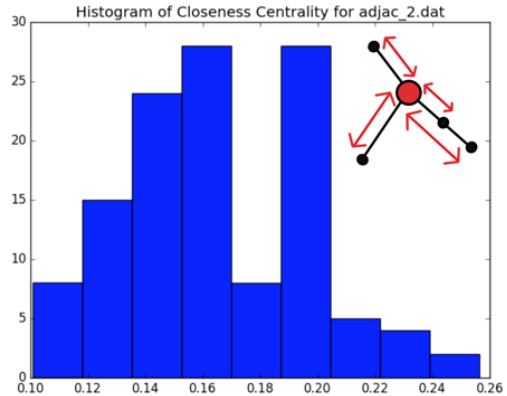
Node Metrics



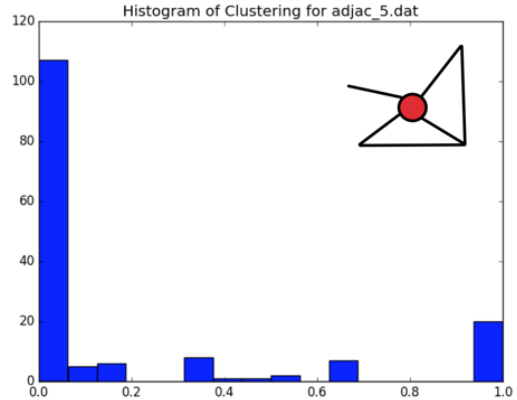
- Source-Sink Paths



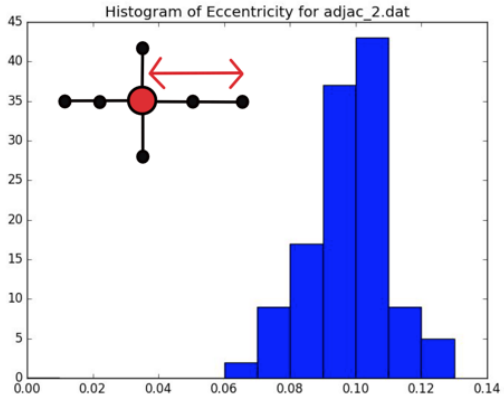
- Betweenness Centrality



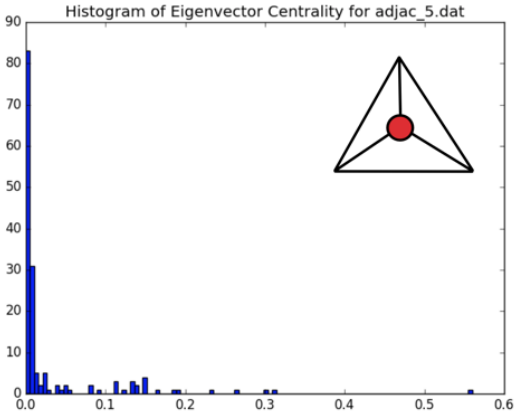
- Closeness Centrality



- Clustering Coefficient



- Eccentricity

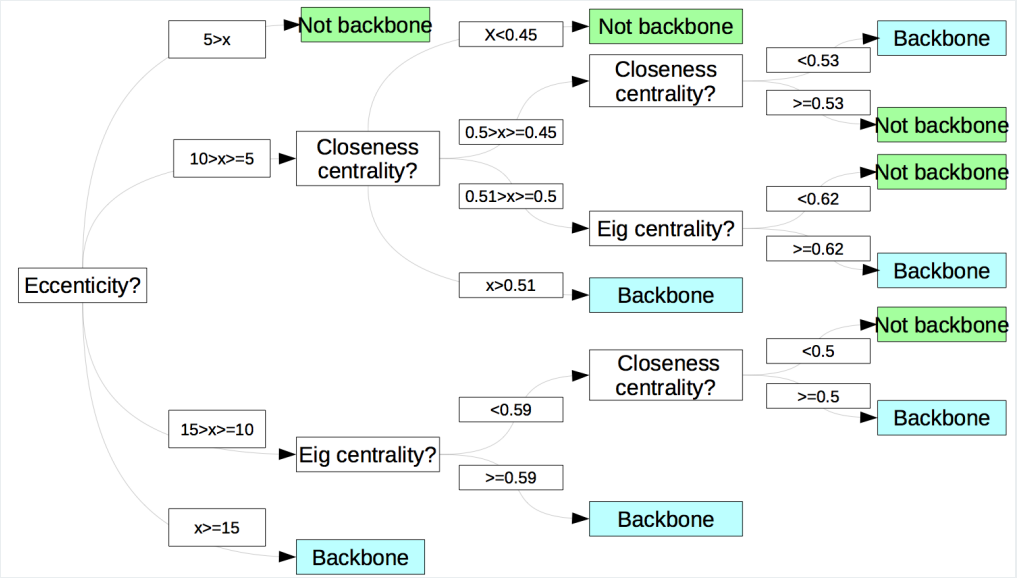


- Eigenvector Centrality

Machine Learning Methods

We use ML to study which nodes and edges will support flow and/or develop into new fractures based on the topological and physical characteristics of the nodes and edges.
The techniques we will use are:

- Random Forest
- Support Vector Machine

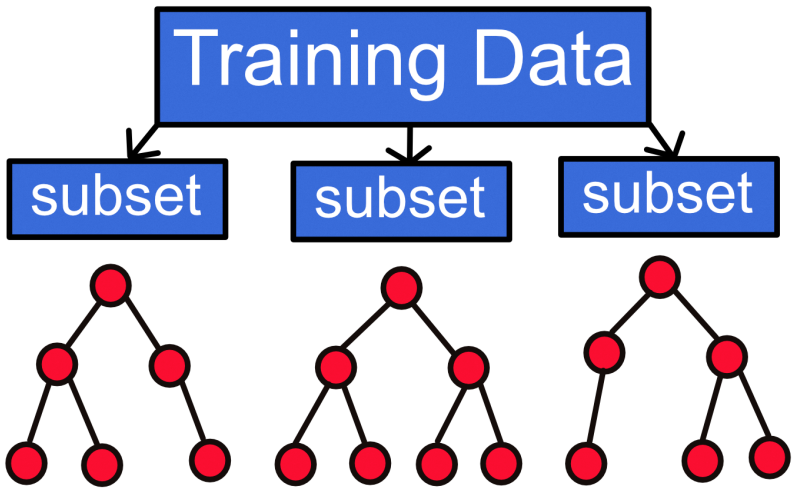


Random Forest

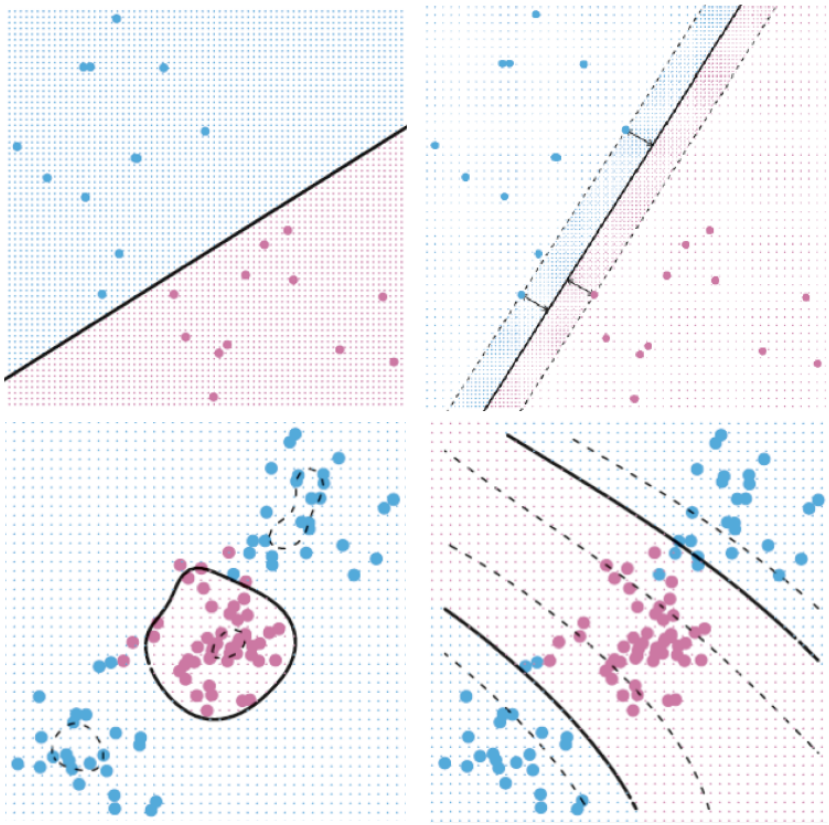
A decision tree is a graph where the nodes represent details about a decision or classification.

- Variations in training set can yield substantially different trees
- The best tree is not guaranteed

These can be addressed by generating a large collection of trees from sub-samples in the training set.



Support Vector Machine (SVM)



SVM separates the data into two groups using a maximal margin classifier. There are many types of kernels which can generate the classifier [6]

- linear
- polynomial
- radial

In order to generate the classifiers, the SVM algorithm needs to identify boundary limits based on metrics.

[6] James, et al., An Introduction to Statistical Learning, Chap. 9, 2013

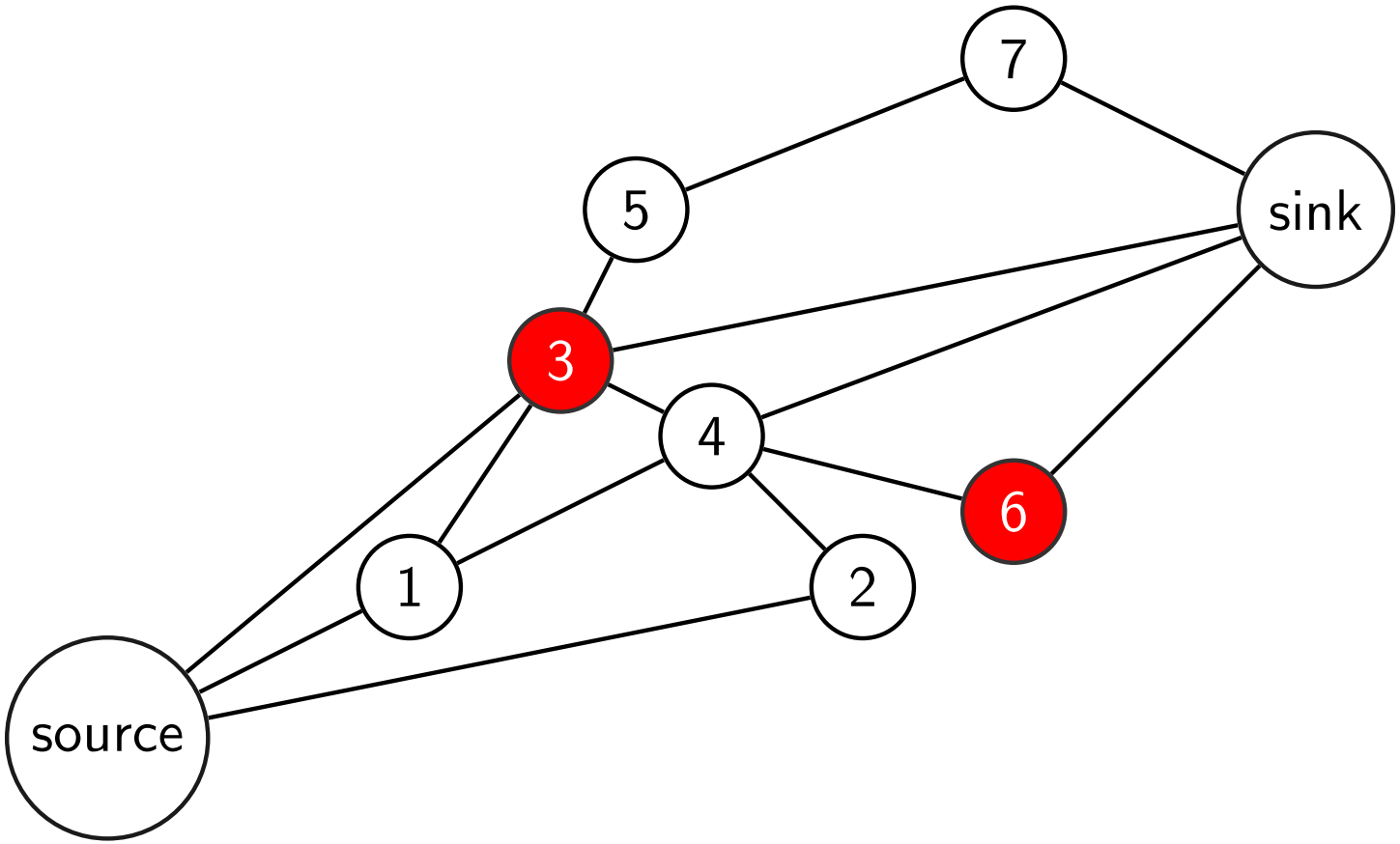
How can we best use these techniques?

The ML algorithms will use node metrics and material properties as features to determine paths carrying significant flow. The learning samples have been generated by LANL using dfnWorks and the HOSS suites.

- Determine which nodes support flow in a static graph generated by dfnWorks
- Predict where new edges, or new connections to existing fractures, would appear in the graph over time by comparing to HOSS simulations

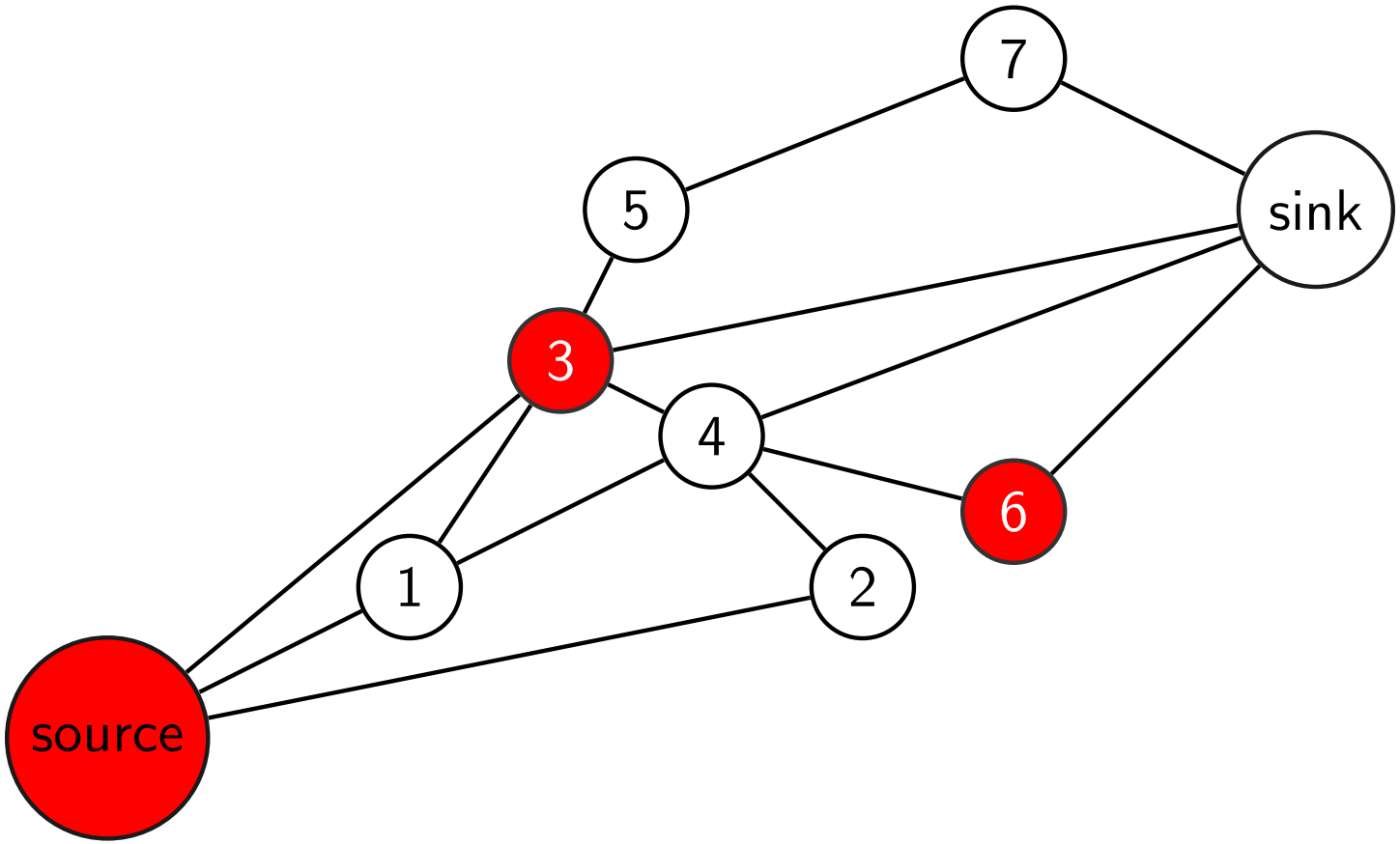
Staged SVM Method

Stage 1: Identify nodes, independently of neighbors, that have the metrics of a backbone or non-backbone. Backbone nodes pictured here in red.



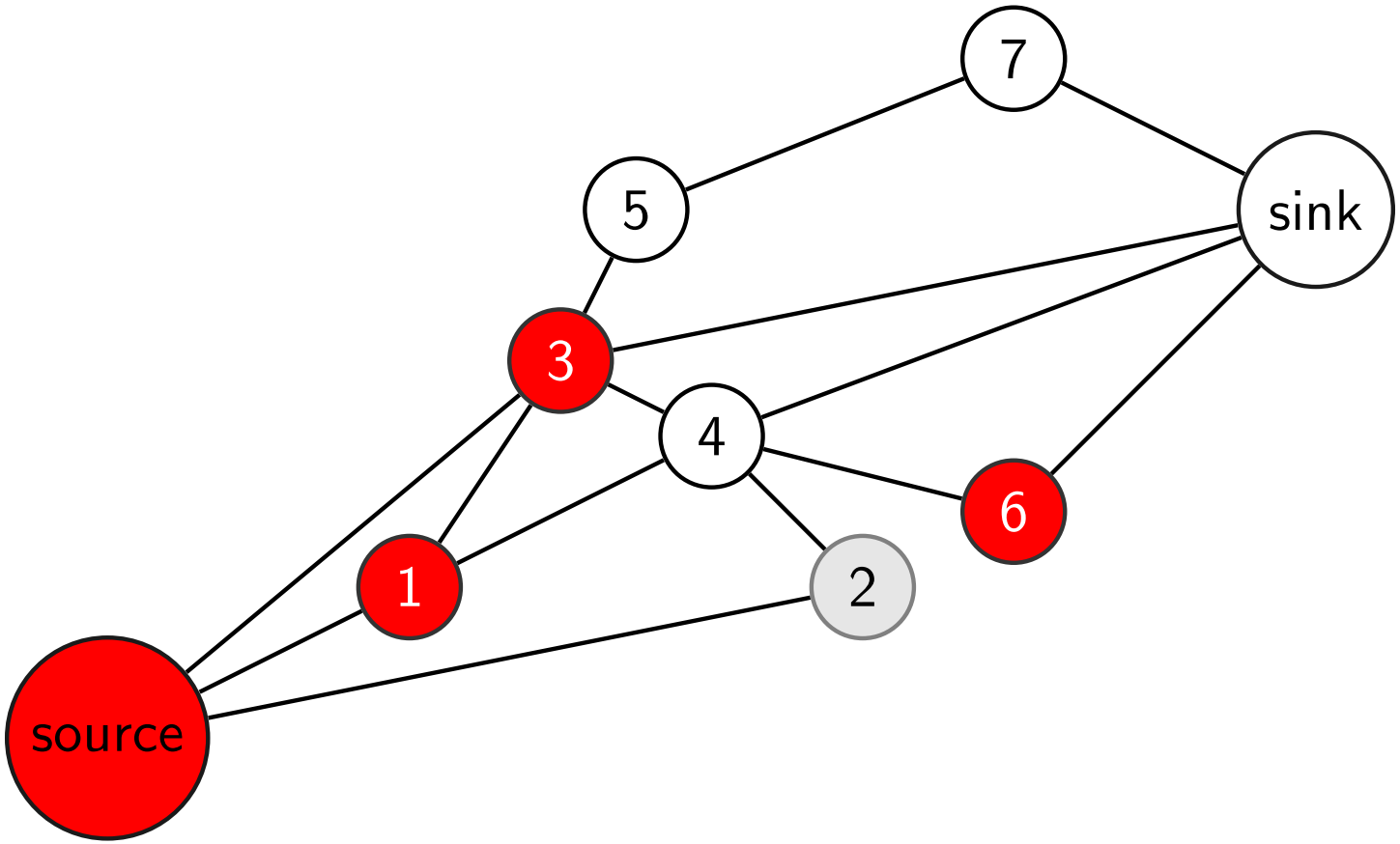
Staged SVM Method

Stage 2: Iteratively classify neighbors of identified backbone nodes starting at source, adding new nodes so as to form a connected backbone.



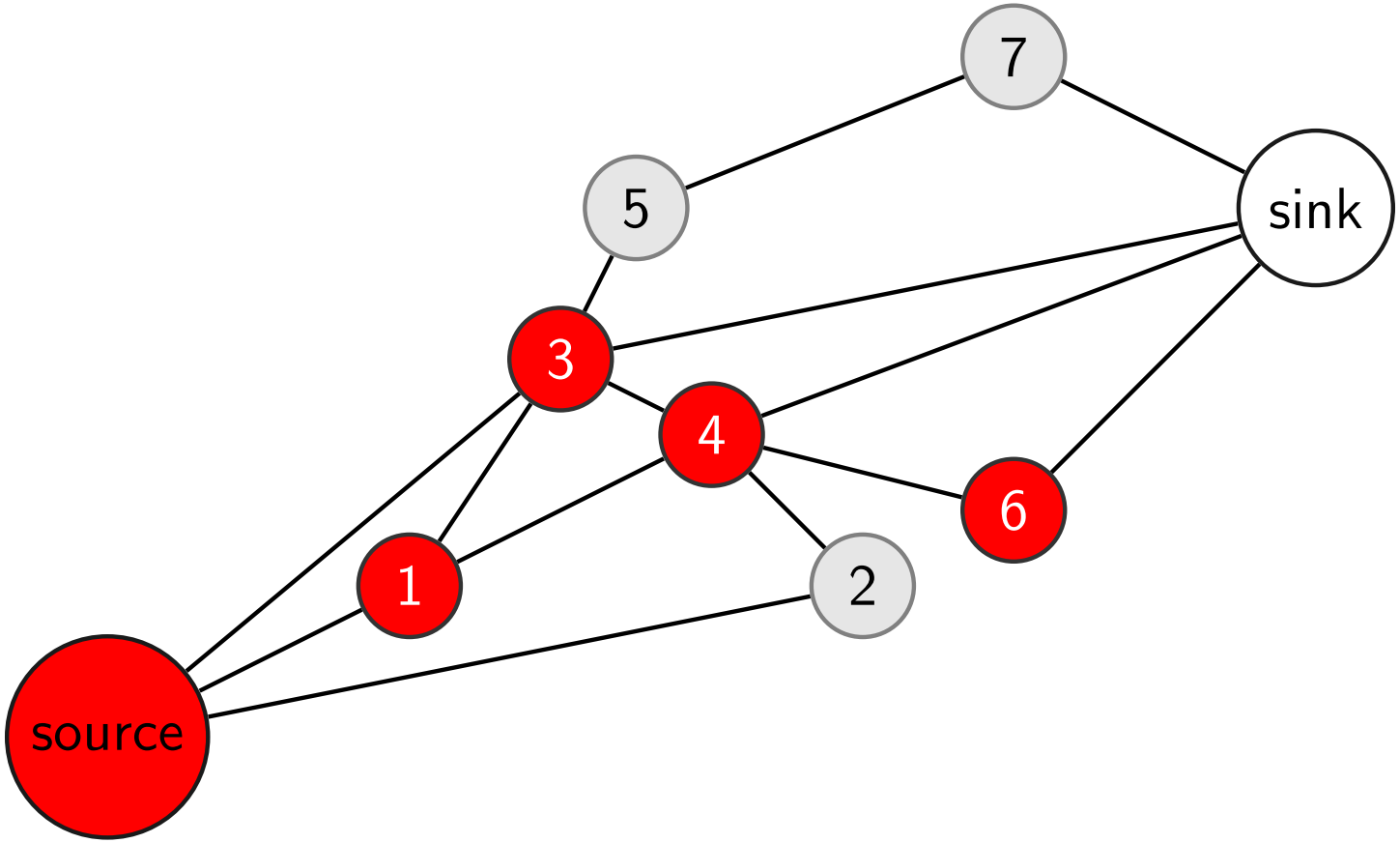
Staged SVM Method

Stage 2: Iteratively classify neighbors of identified backbone nodes starting at source, adding new nodes so as to form a connected backbone.



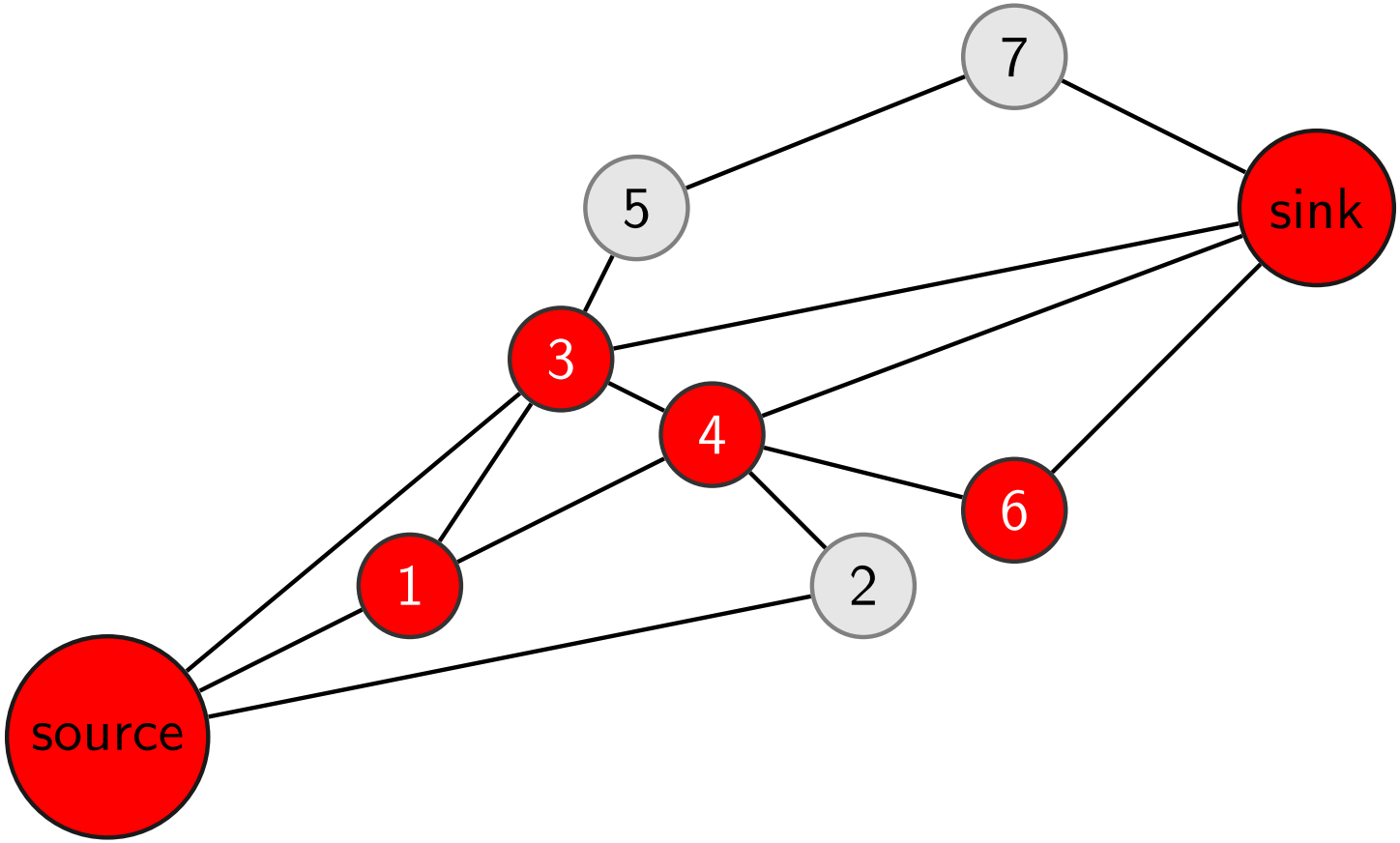
Staged SVM Method

Stage 2: Iteratively classify neighbors of identified backbone nodes starting at source, adding new nodes so as to form a connected backbone.



Staged SVM Method

Stage 2: Iteratively classify neighbors of identified backbone nodes starting at source, adding new nodes so as to form a connected backbone.



Test set

We used the particle-flow data with the first 20 networks as testing set.

- Out of 100 graphs:
  - 80 graphs (??? nodes) were chosen as training data, 6% of the samples are backbone
  - 20 graphs (9238 nodes) were chosen as test data 7.5% of the samples are backbone
- We show 8 models (4 RF, 4 SVM) with different Precision/Recall ratios.

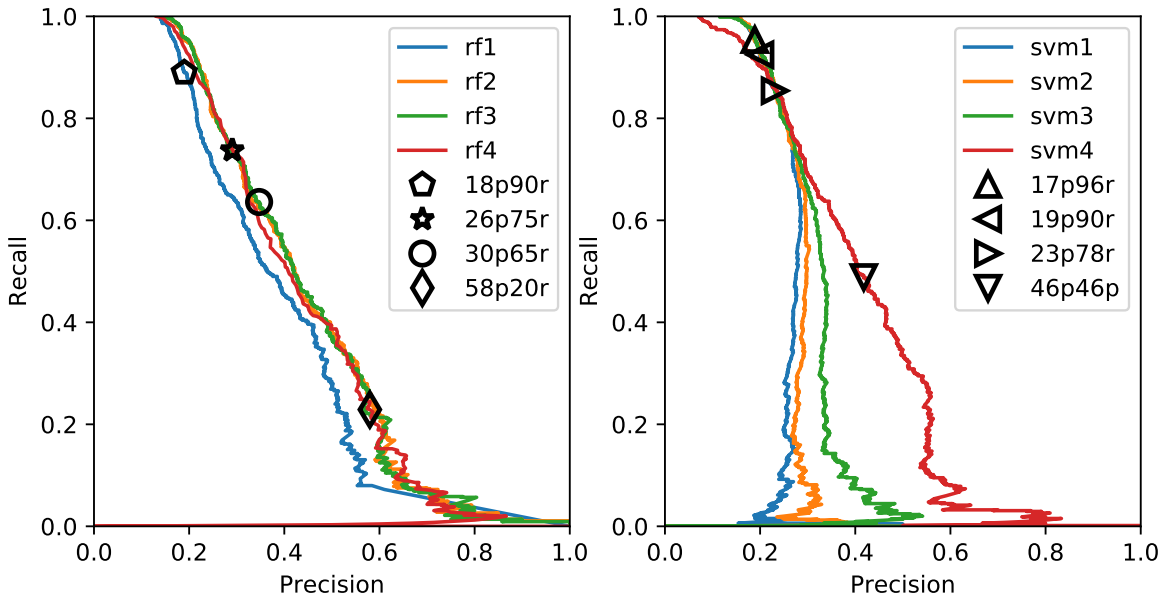
P/R Curves and Critical Parameters

Random Forest

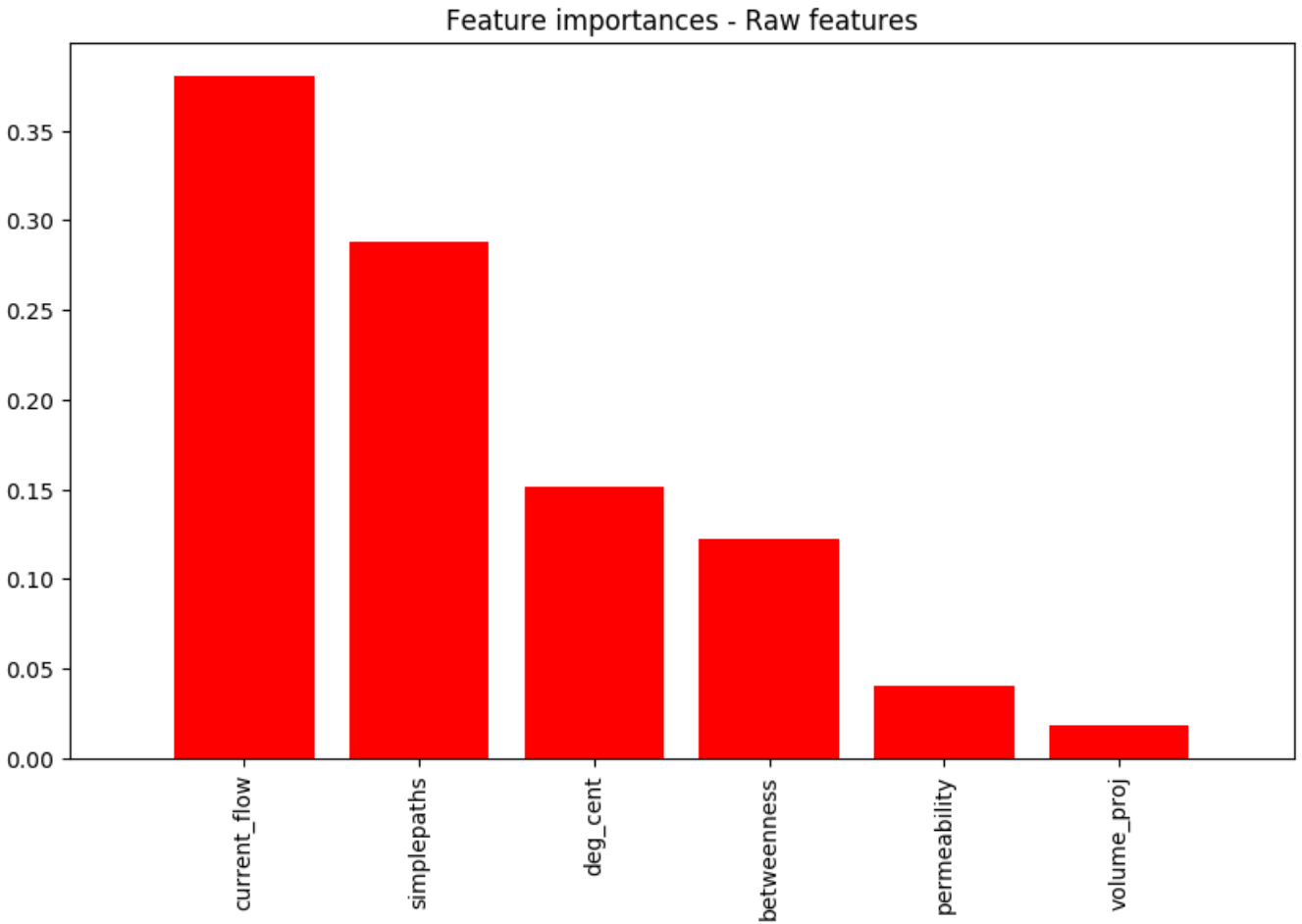
Model	min. samples leaf
18p-90r-RF	370
26p-75r-RF	8
30p-65r-RF	4
58p-20r-RF	1

SVM

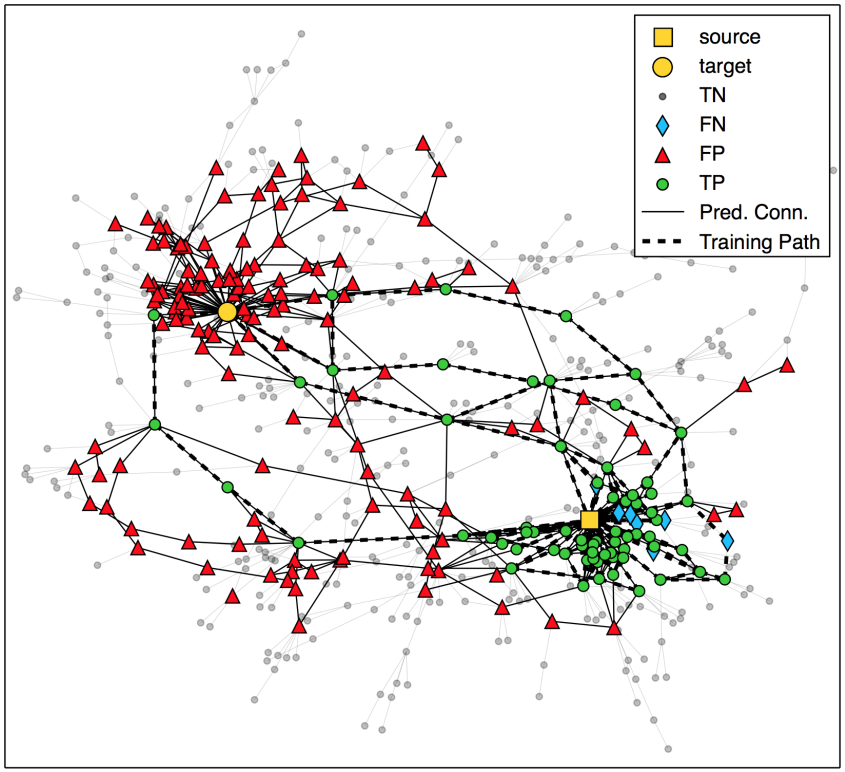
Model	weight NB	weight BB
17p-96r-SVM	0.54	9.00
19p-90r-SVM	0.63	9.00
23p-78r-SVM	0.70	7.00
46p-46r-SVM	1.90	7.00

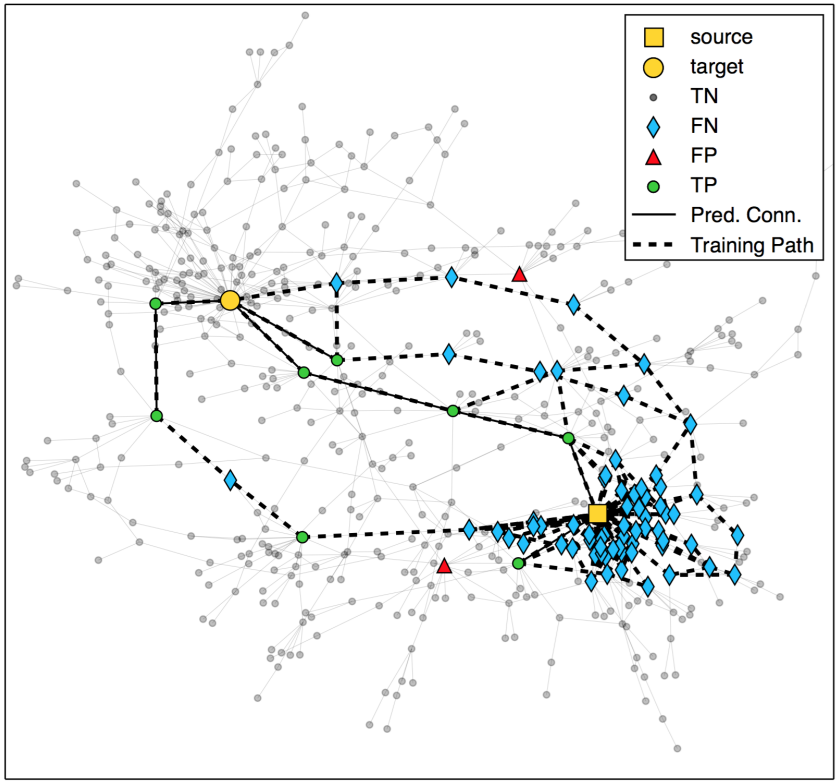


Random Forest Importance



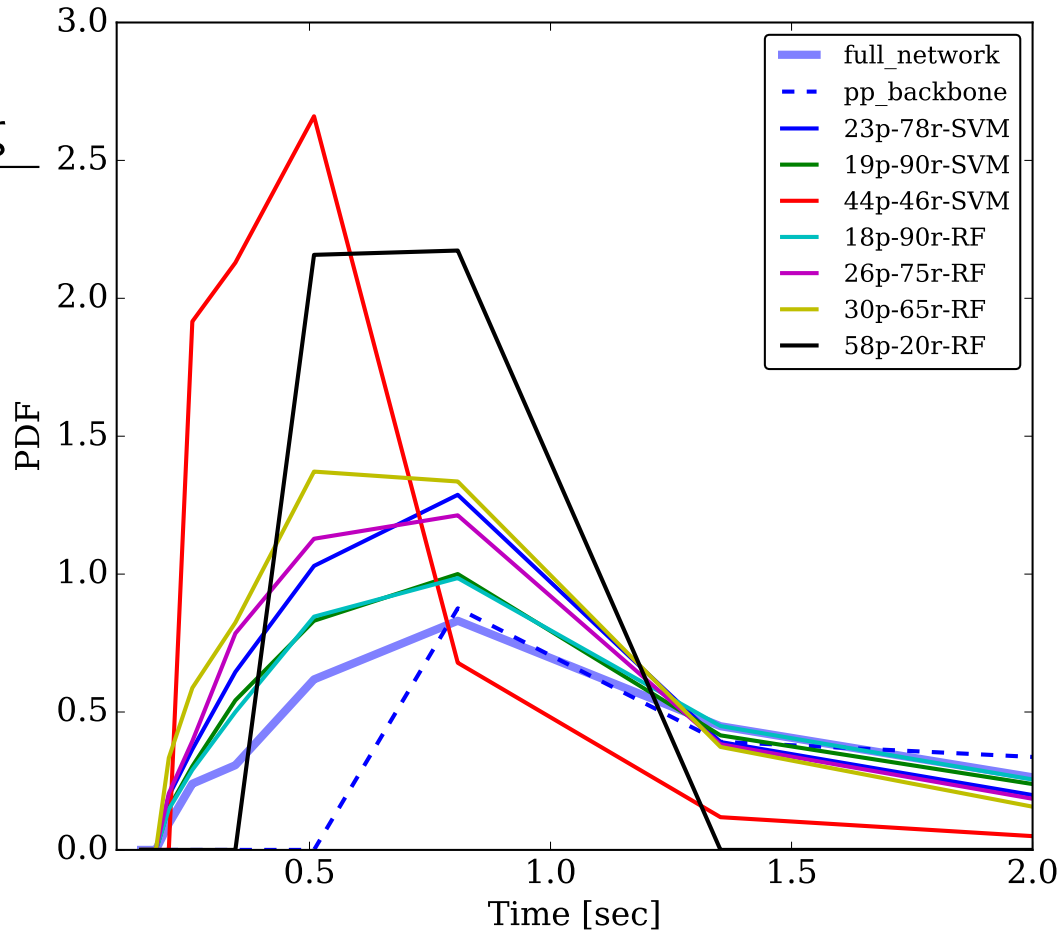
Extreme Cases





Validation - Breaktrough Curves

Model	Remaining
17p-96r-SVM	39%
18p-90r-RF	36%
19p-90r-SVM	34%
23p-78r-SVM	24%
26p-75r-RF	21%
30p-65r-RF	15%
46p-46r-SVM	7%
58p-20r-RF	3%



Final Remarks

- Flow through static graph
  - Success in using topological properties as features to classify a backbone nodes set
  - Arbitrary results enable us to apply it to different problems
  - Remaining graphs from 39% // Savings of 61% of nodes
  - Instantaneous predictions, validated by BTCs.
  - The limitations of the training set are overcome by our approach.
- We tried much more... ! (talk about it)