

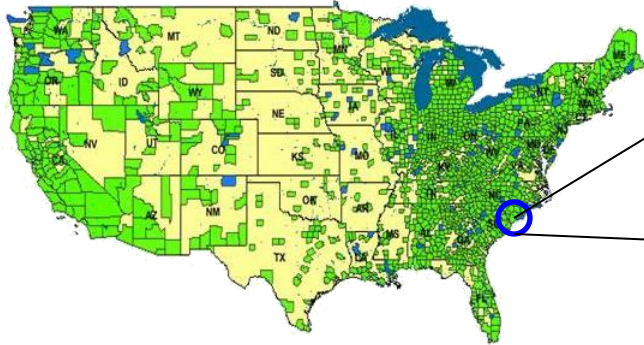
Scalable Detection of Anomalous Patterns with Connectivity Constraints

Skyler Speakman, Ed McFowland III, Daniel B. Neill

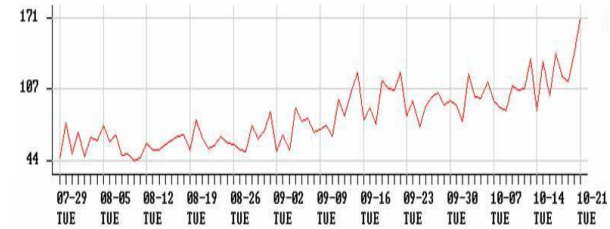
Event and Pattern Detection Lab
H.J. Heinz III College
Carnegie Mellon University

This work was partially supported by NSF grants
IIS-0916345, IIS-0911032, and IIS-0953330





Daily health data from
thousands of hospitals and
pharmacies nationwide



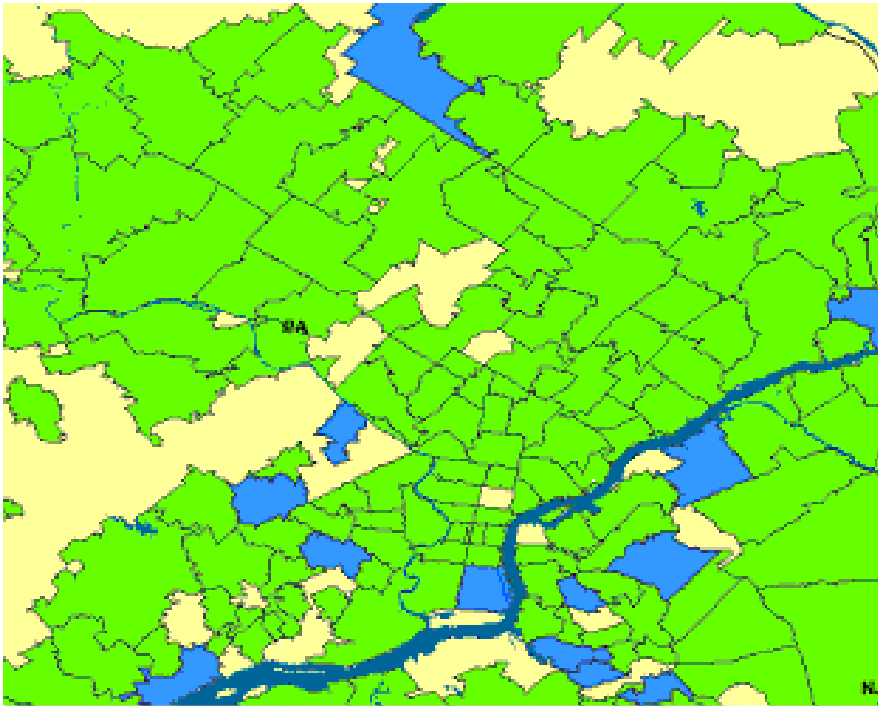
Time series of counts c_i^t
for each zip code s_i

Use this data to detect
anomalous patterns

Detect any emerging events (i.e. outbreaks of disease)
Pinpoint the affected areas

Biosurveillance

(Kulldorff, 1997; Neill and Moore, 2005)



Scan over multiple regions to detect where counts are higher than expected

Aggregate the individual counts from each location within a region

$$C = \sum_S c_i^t \text{ and } B = \sum_S b_i^t$$

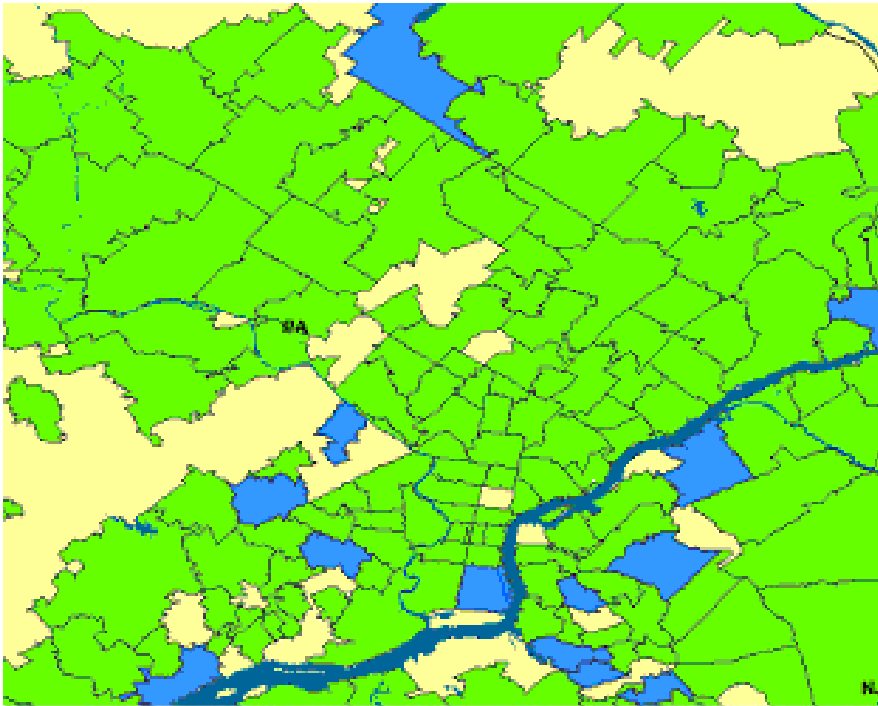
Determine *anomalousness* of region with a scoring function

$$F(S) = \frac{\Pr(\text{Data} | H_1(S))}{\Pr(\text{Data} | H_0)}$$

$$F(S) = \left(\frac{C}{B} \right)^C e^{B-C}$$

Expectation-Based Scan Statistics

(Kulldorff, 1997; Neill and Moore, 2005)



Scan over multiple regions to detect where counts are higher than expected

Aggregate the individual counts from each location within a region

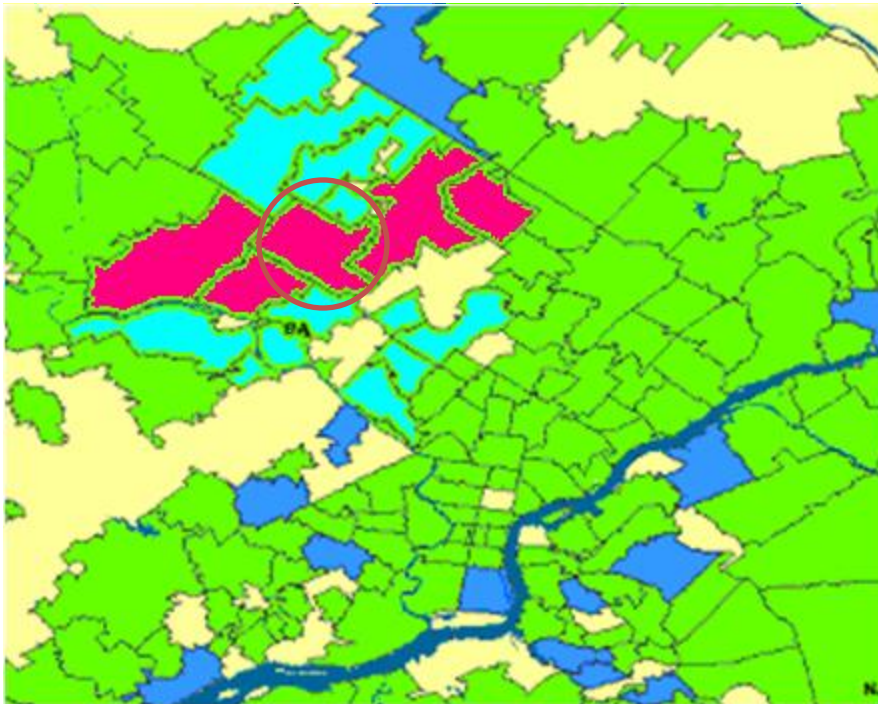
Circles

Choose a center location s_c and its k nearest neighbors

Find the circle that maximizes the score function of the aggregated counts and baselines

Expectation-Based Scan Statistics

(Kulldorff, 1997; Neill and Moore, 2005)




Power to Detect

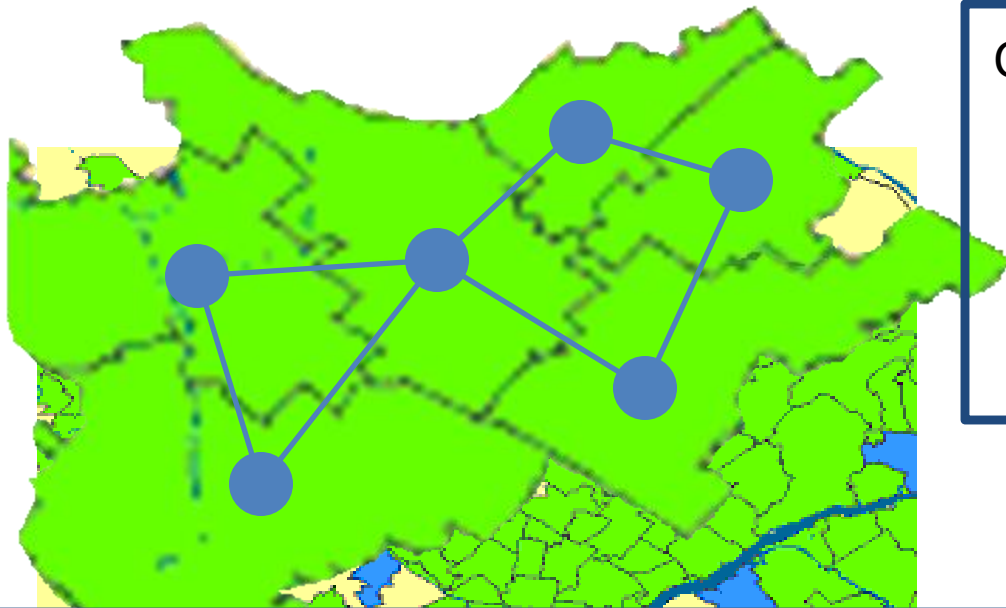
Circles are useful for detecting tightly clustered outbreaks

However, they lose power to detect abnormally shaped clusters

 Affected locations

 Un-affected locations contributing to region score

Expectation-Based Scan Statistics



Create an adjacency graph of the locations and score ***connected subsets***

Increase power to detect non-circular clusters

Upper Level Set Scan Statistic (ULS)

Patil & Taillie, 2004

Uses a heuristic to determine high scoring connected subsets
Is not guaranteed to find the highest scoring connected subset

Flexible Scan statistic (FlexScan)

Tango & Takahashi, 2005

Naively scores all connected subsets
Infeasible for regions of >30 locations

Connectivity Constraints

PROBLEM:	The number of subsets grows exponentially with the size of the region 2^N
This makes it computationally infeasible for regions with more than ~ 30 locations	
SOLUTION:	Exploit a property of scoring functions to rule out subsets that cannot obtain the highest score
This reduction in the search space allows for exact and efficient calculation of the highest scoring <i>unconstrained subset</i>	
EXTENSION:	Use this same property for exact and efficient calculation of the highest scoring <i>connected subset</i>

Subset Scanning



(Neill, 2008)

We wish to maximize a scoring function

$$F(S) = F\left(\sum_{s_i \in S} c_i, \sum_{s_i \in S} b_i\right)$$

over all possible subsets, S

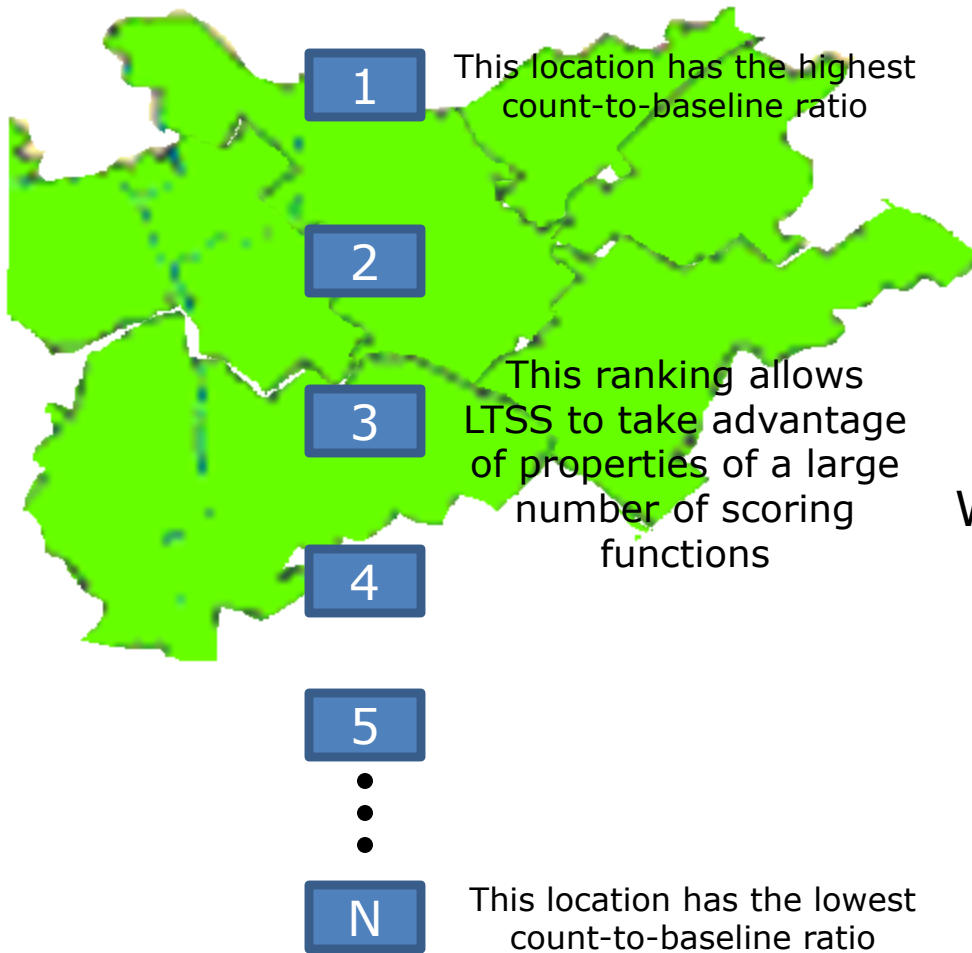
Sort the locations according to a priority function

For example,

$$G(s_i) = \frac{c_i}{b_i}$$

Works for expectation-based Poisson (EBP)

Linear Time Subset Scanning



(Neill, 2008)

We wish to maximize a scoring function

$$F(S) = F\left(\sum_{s_i \in S} c_i, \sum_{s_i \in S} b_i\right)$$

over all possible subsets, S

We sort the locations according to a relevance criteria

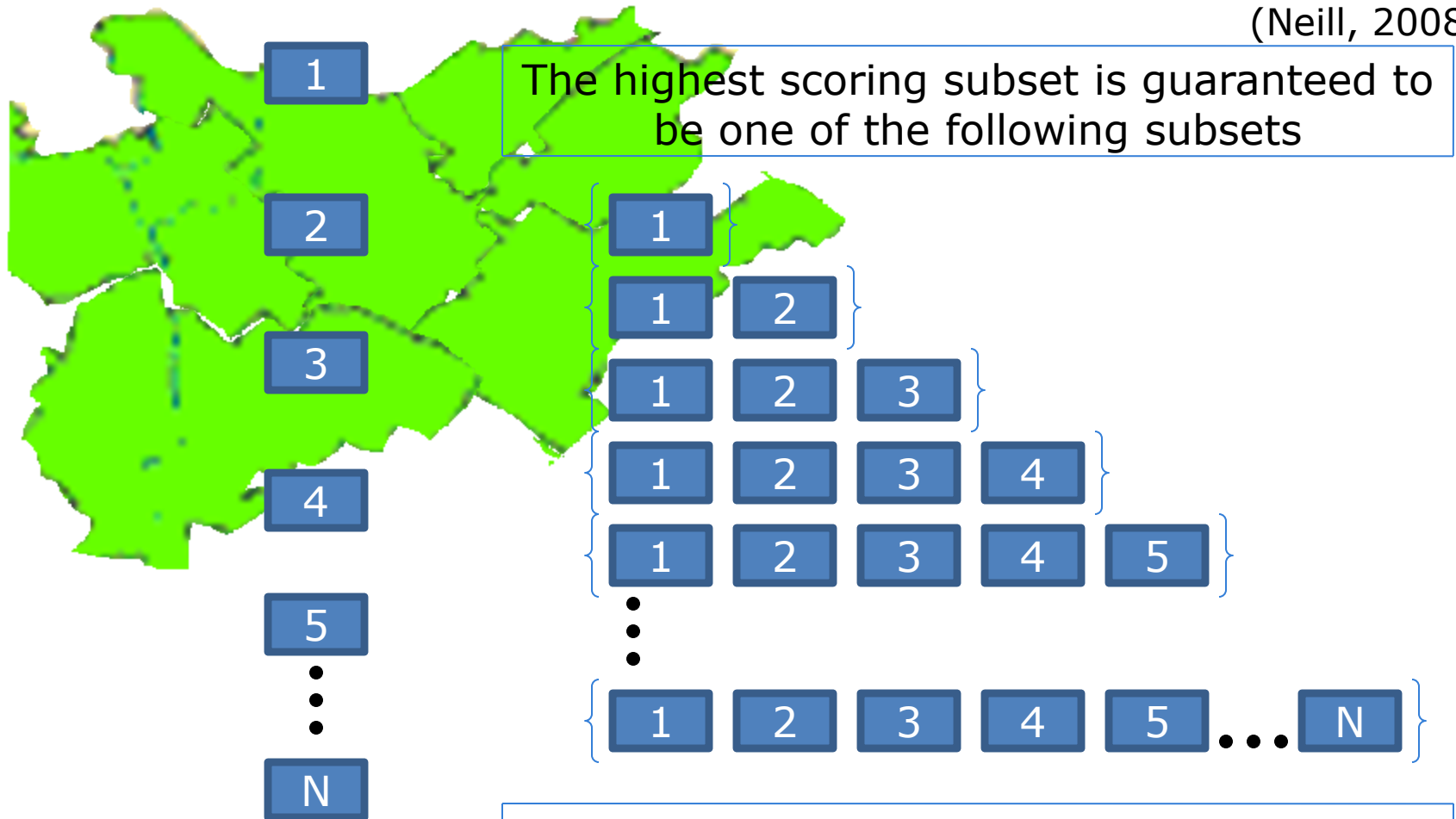
For example,

$$G(s_i) = \frac{c_i}{b_i}$$

works for Expectation-based Poisson (EBP)

Linear Time Subset Scanning

(Neill, 2008)

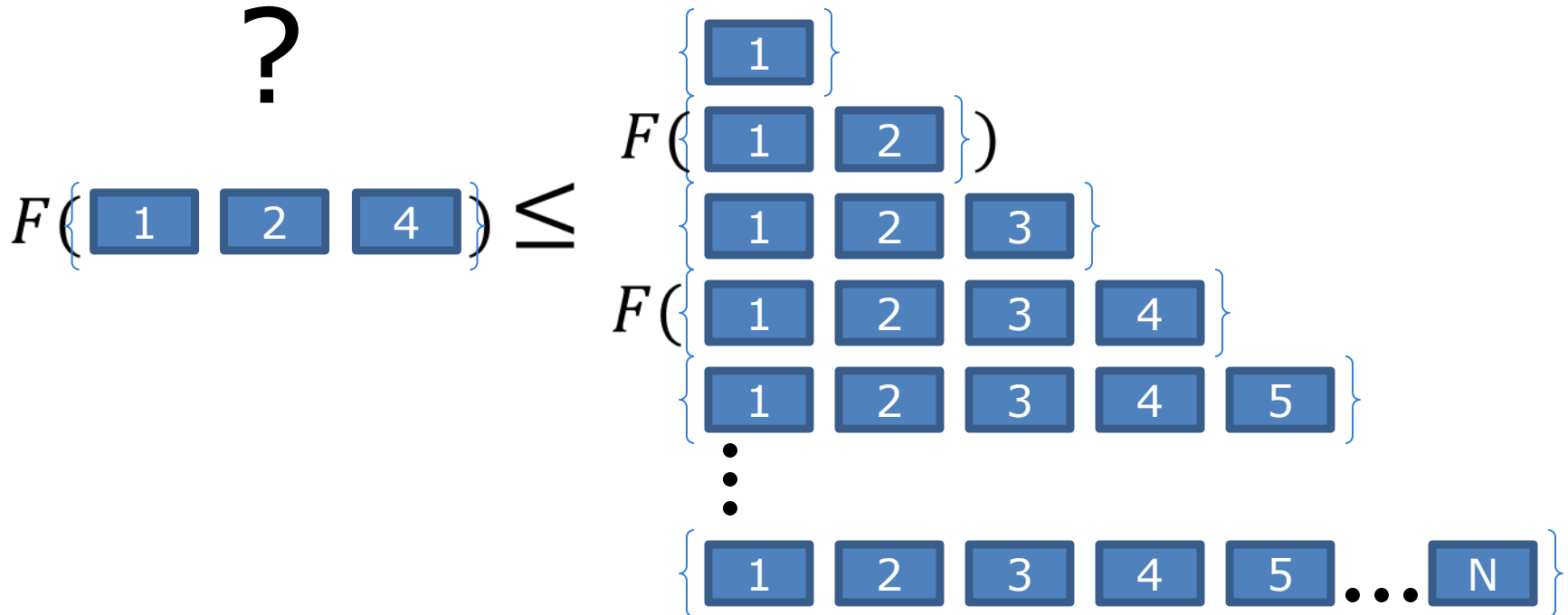


Decreases the search space from 2^N to N

Linear Time Subset Scanning

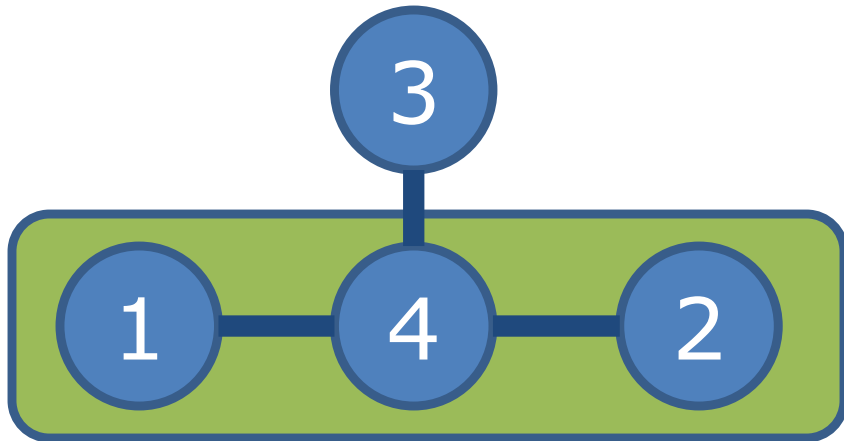
Some Quick Intuition... (Neill, 2008)

?



Linear Time Subset Scanning

If the k^{th} priority location is contained in the optimal subset
and if removing the location does not disconnect the subset
then all higher priority adjacent locations
must also be in the optimal subset.



C=41 B=3

Priority	Count	Baseline
1	20	1
2	20	1
3	1001	1000
4	1	1

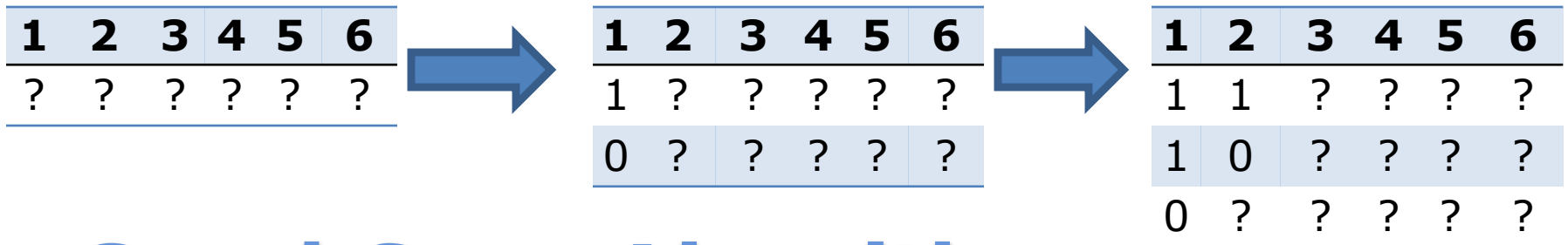
LTSS with Connectivity Constraints

We represent groups of subsets as a string of 0's, 1's, and ?'s

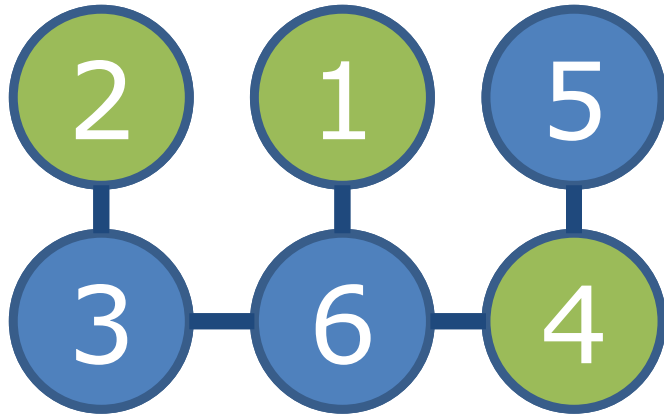
Priority Ranking	1	2	3	4	5	6
Bit String	1	0	0	1	?	?

The above bit string represents 4 possible subsets:
 {1,4} {1,4,5} {1,4,6} {1,4,5,6}

A Naïve approach would search all 2^N subsets and is computationally infeasible



GraphScan Algorithm



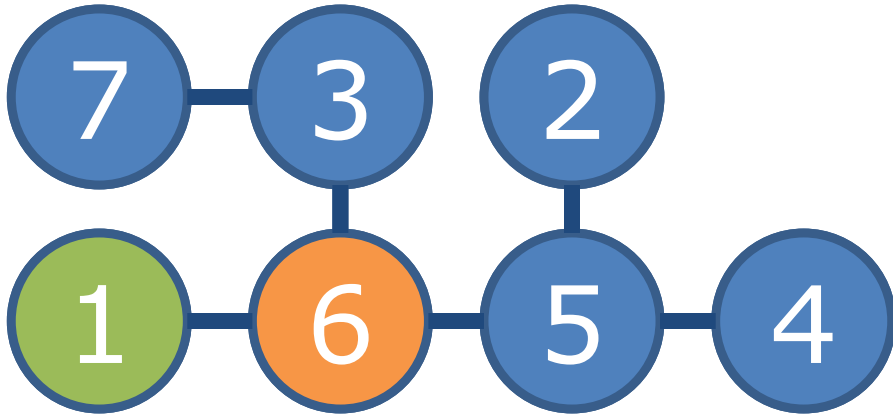
Seed nodes have higher priority than all of their neighbors

We can rule out bit strings whose highest priority node is not a seed node

Seed nodes provide starting locations for the following depth first search

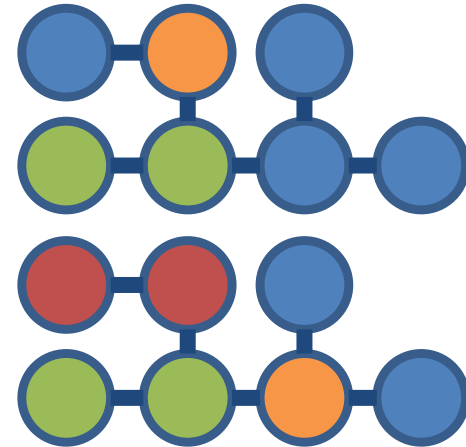
	1	2	3	4	5	6
S_1	1	?	?	?	?	?
S_2	0	1	?	?	?	?
S_3	0	0	1	?	?	?
S_4	0	0	0	1	?	?
S_5	0	0	0	0	1	?
S_6	0	0	0	0	0	1

GraphScan Algorithm: Seeds

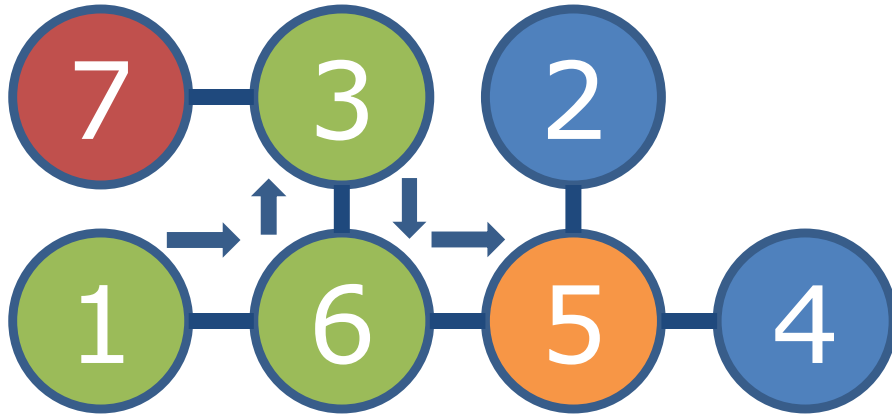


Q
U
E
U
E

	1	2	3	4	5	6	7
1	1	?	?	?	?	1	?
	1	?	0	?	1	1	0



GraphScan Algorithm: Propagation

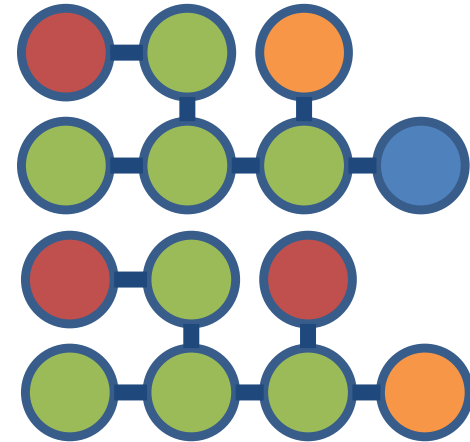
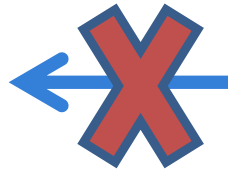


Notice that 3 can be removed and not disconnect the subset

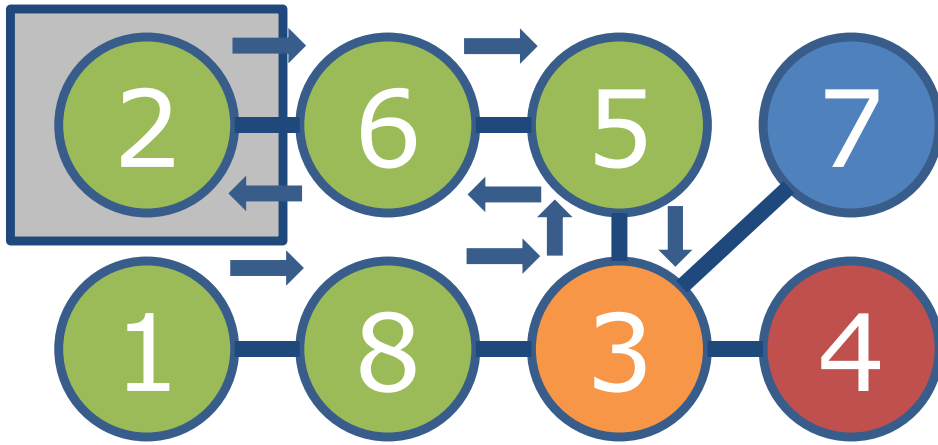
Q
U
E
U
E

1	2	3	4	5	6	7
1	?	1	?	1	1	0

Provably sub-optimal
by LTSS Property



GraphScan Algorithm: Propagation



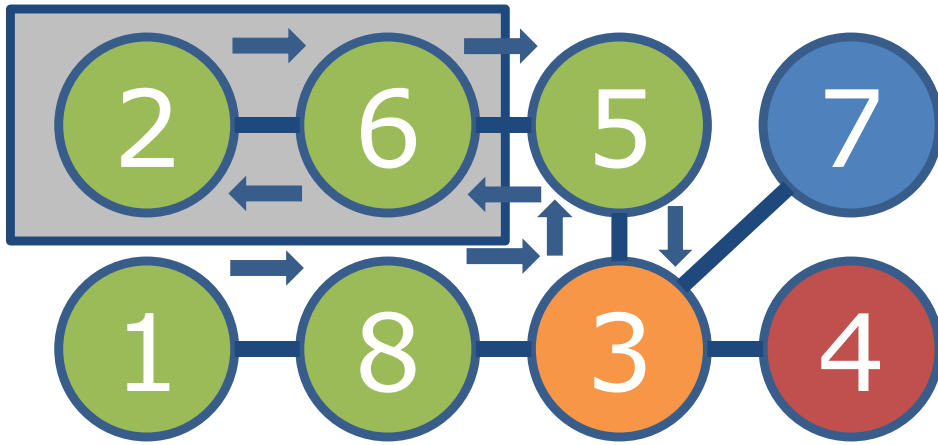
Q
U
E
U
E

1	2	3	4	5	6	7	8
1	1	1	0	1	1	?	1

2 is the lowest priority record that can be removed without disconnecting the subset

However, 2 is not low enough to rule out this subset (compared to 4)

GraphScan Algorithm: Backtrack



Priority	Count	Baseline
1	9	1
2	7	1
3	9	3
4	3	1
5	3	3
6	1	1
7	2	3
8	1	2

Q
U
E
U
E

1	2	3	4	5	6	7	8
1	1	1	0	1	1	?	1

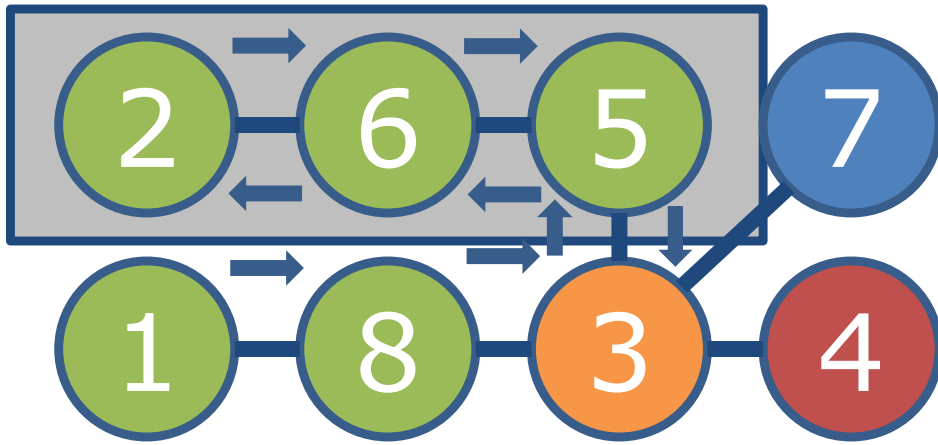
2 and 6 may be removed simultaneously without disconnecting the subset

2 and 6 have combined priority of

$$\frac{7 + 1}{1 + 1} = \frac{8}{2}$$

still not lower than 4's $\frac{3}{1}$ priority

GraphScan Algorithm: Backtrack



Priority	Count	Baseline
1	9	1
2	7	1
3	9	3
4	3	1
5	3	3
6	1	1
7	2	3
8	1	2

Q
U
E
U
E

1	2	3	4	5	6	7	8
1	1	1	0	1	1	?	1

2,6, and 5 may be removed simultaneously without disconnecting the subset

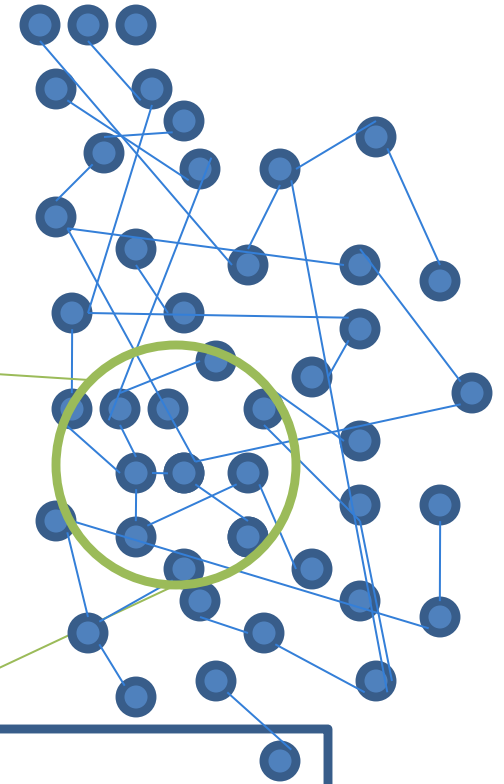
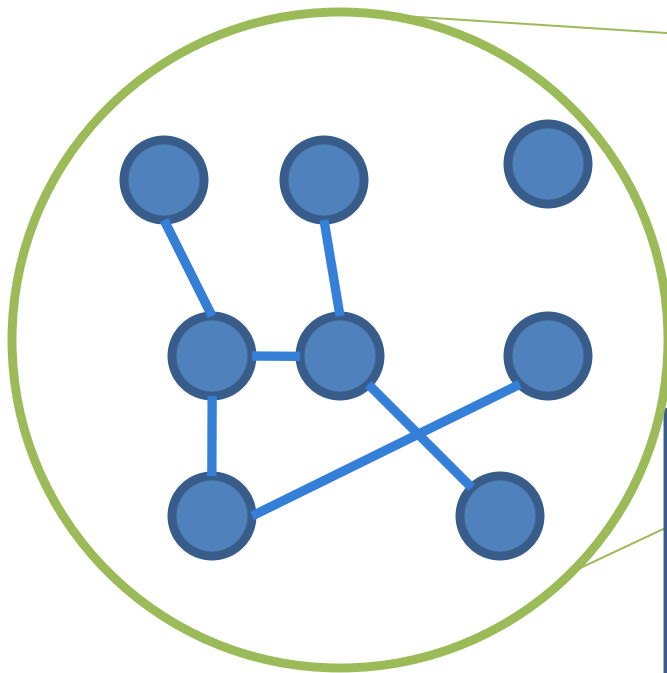
2, 6, and 5 have combined priority of

$$\frac{7 + 1 + 3}{1 + 1 + 3} = \frac{11}{5} = 2.2$$

lower than 4's $\frac{3}{1}$ priority.

GraphScan Algorithm: Backtrack

If the domain provides *spatial* information, we may use both proximity and connectivity constraints simultaneously



Forming a neighborhood of the 'k nearest neighbors'

Proximity Constraints

Two years of admissions from
10 different Allegheny County
Emergency Departments

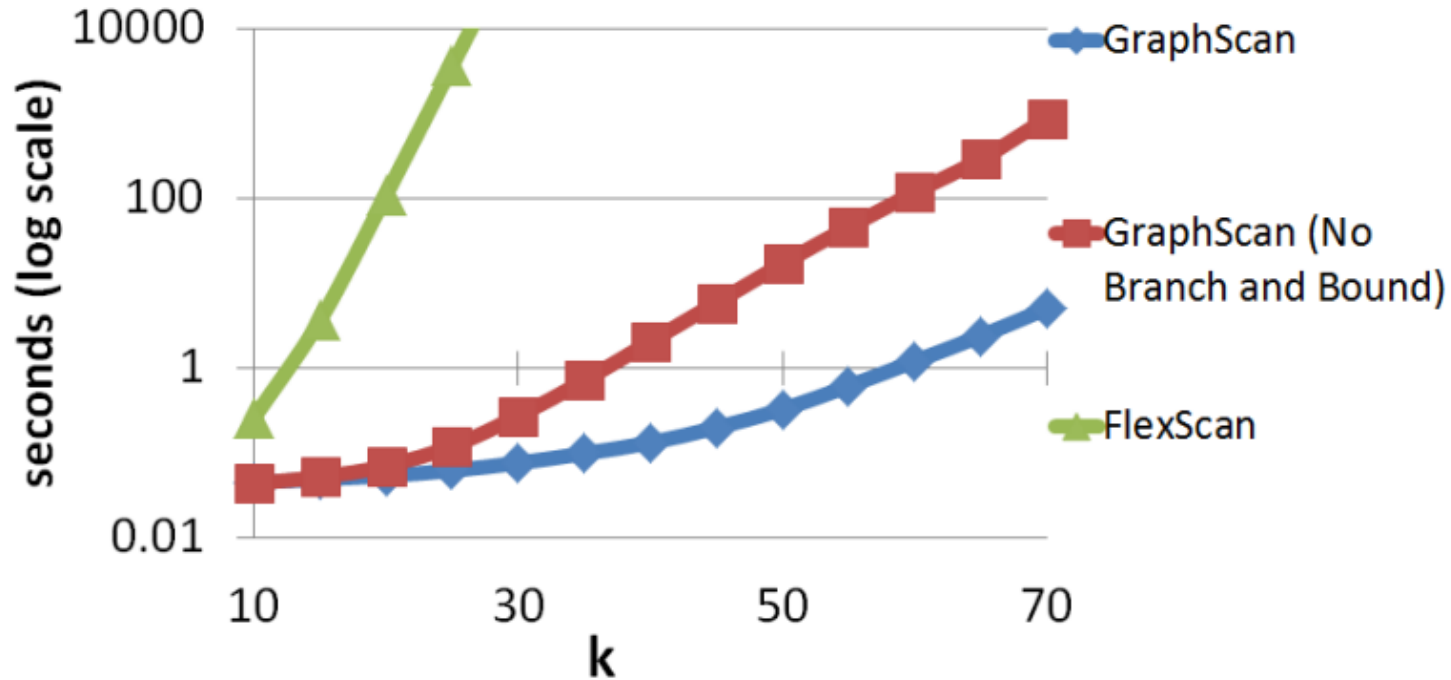
The patient's home zip code
is used to tally the counts at
each location (node)

Only consider patients from
within Allegheny County



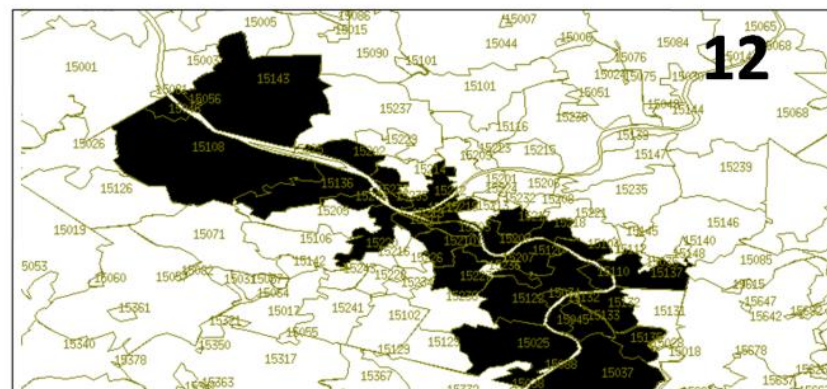
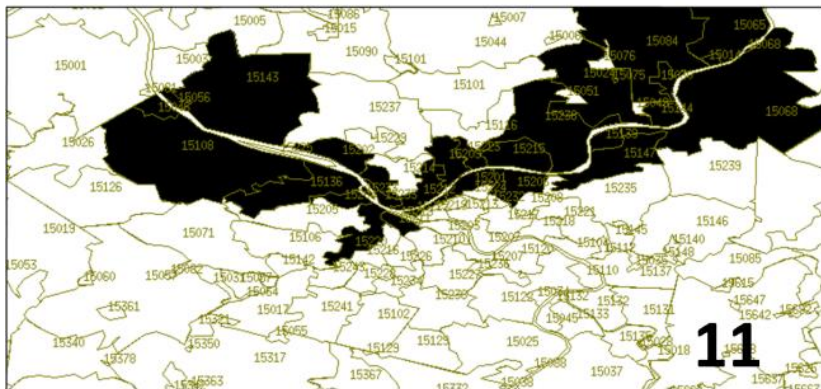
Evaluation: Emergency Department Data

Run times per single day of Emergency Department data



Run time analysis for FlexScan and GraphScan with and without Branch and Bounding. The x-axis denotes the “neighborhood size” as various values of k .

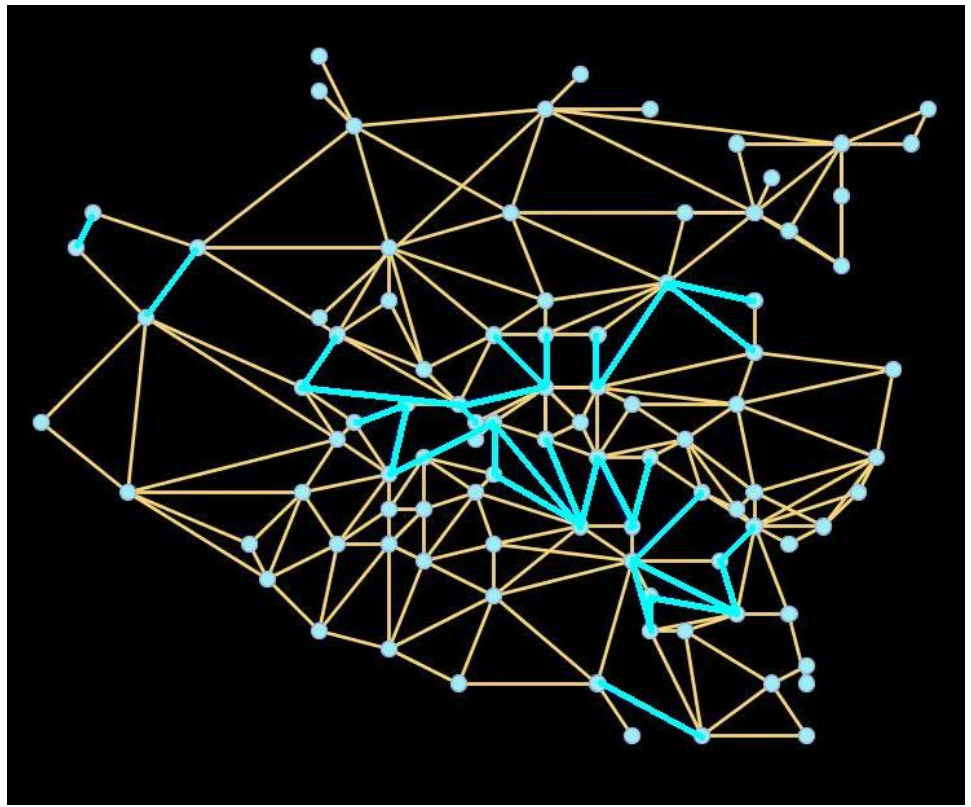
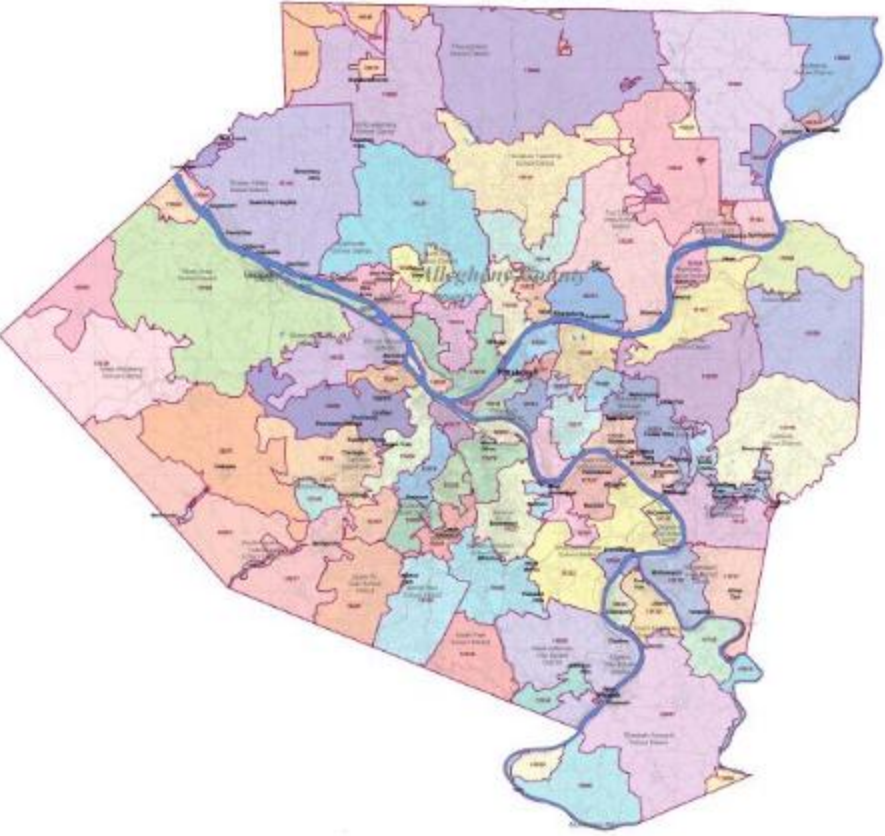
Evaluation: Run Times

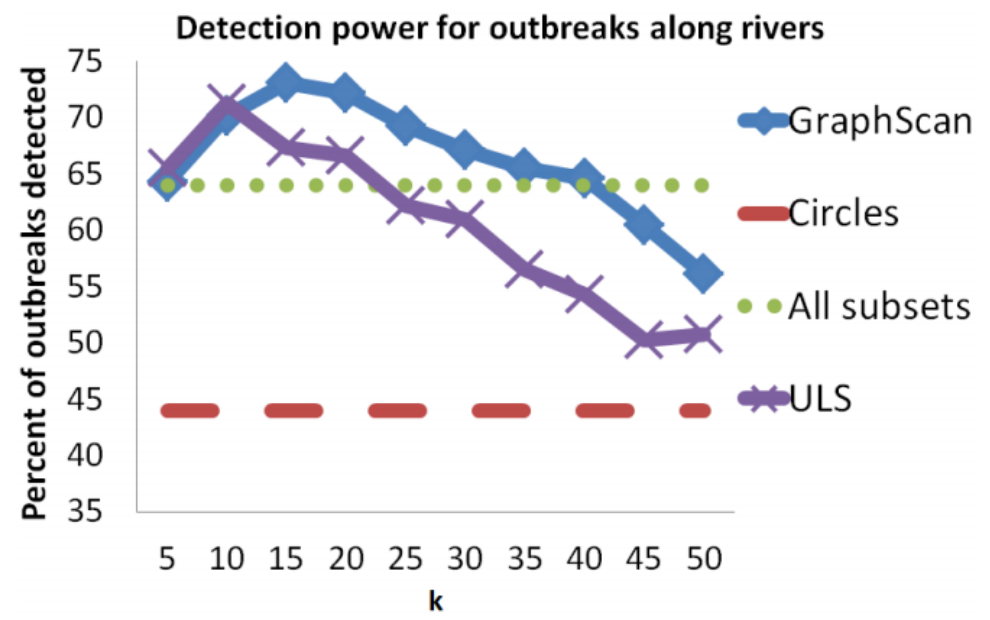
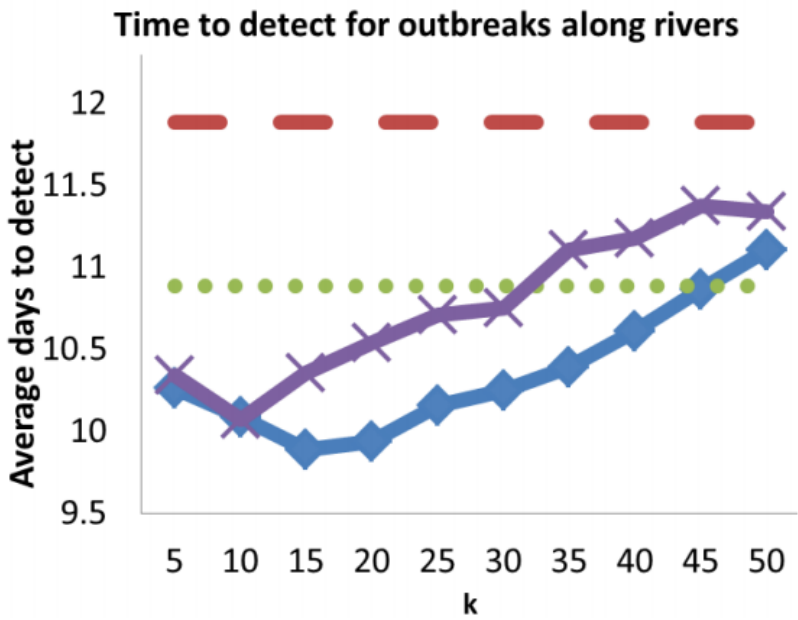


Semi-synthetic injects were created by artificially increasing the observed counts in selected zip codes. Zip codes adjacent to rivers were selected as an example of realistic yet abnormally-shaped cluster.

Compare performance on detection power and time to detect for a fixed false positive rate of 1 per month.

Evaluation: Injects

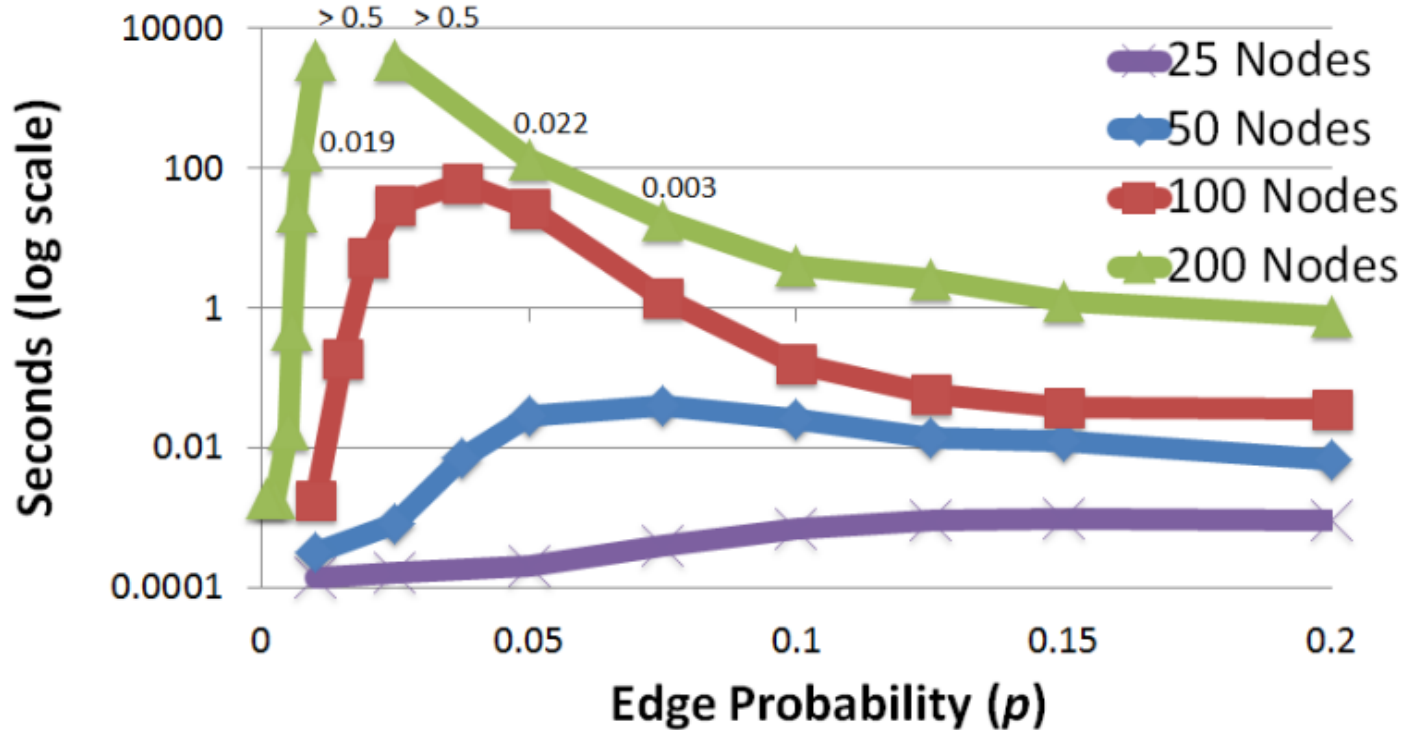




Average detection time and detection power for outbreaks along the rivers.

Results

Average run time on random graphs



Run time performance of GraphScan on randomly generated Erdos-Renyi graphs of varying size and edge probability. Labeled data points mark the proportion of graphs where run time exceeded a 1-hour threshold.

Results: Random Graphs

This work provides...

Theoretical framework for ruling out connected subsets that are provably suboptimal according to the LTSS property

Practical implementation of LTSS with connectivity constraints through the GraphScan Algorithm

GraphScan has shown...

Extremely large speed improvements over FlexScan, while still guaranteeing to identify the highest scoring connected subset

Using connected subsets can increase detection power for irregularly shaped disease clusters

Conclusions