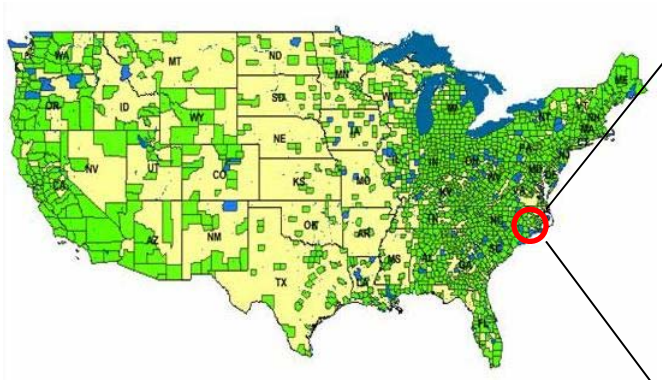
Fast Multivariate Subset Scanning for Scalable Cluster Detection

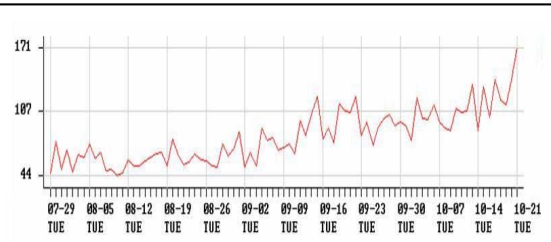
Daniel B. Neill*, Edward McFowland III, Skyler Speakman
Event and Pattern Detection Laboratory
Carnegie Mellon University
***E-mail: neill@cs.cmu.edu**

We gratefully acknowledge funding support from the National Science Foundation, grants IIS-0916345, IIS-0911032, and IIS-0953330.

Multivariate event detection



Spatial time series data from spatial locations s_i (e.g. zip codes)



Time series of counts $c_{i,m}{}^t$ for each zip code s_i for each data stream d_m.

<u>Outbreak detection</u>

d_1 = respiratory ED

d_2 = constitutional ED

d_3 = OTC cough/cold

d_4 = OTC anti-fever

(etc.)

<u>Main goals</u>:

Detect any emerging events.

Pinpoint the affected subset of locations and time duration.

Characterize the event by identifying the affected streams.

<u>Compare hypotheses</u>:

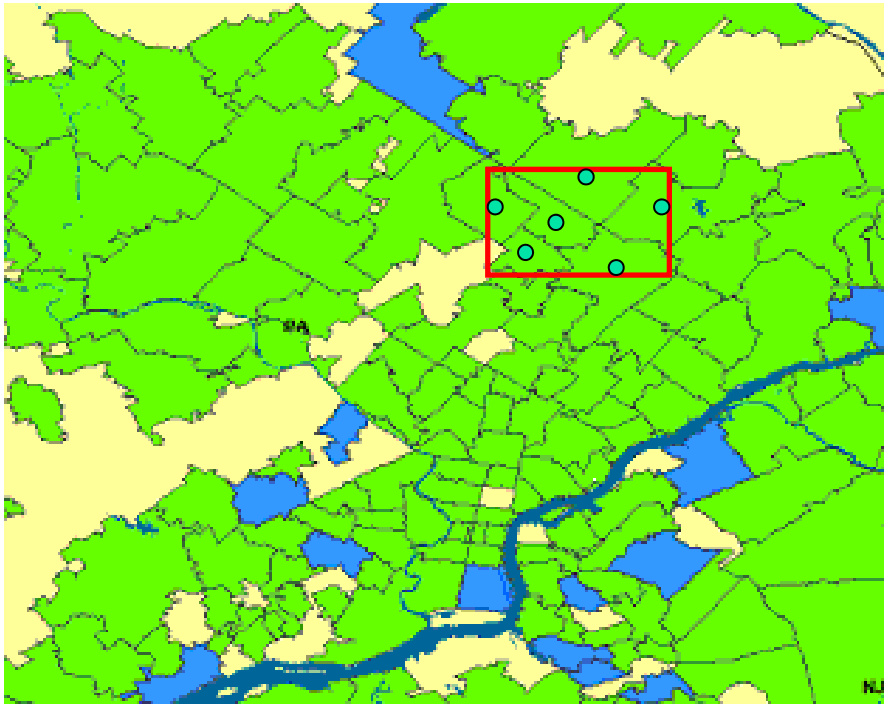
$H_1(D, S, W)$

D = subset of streams
S = subset of locations
W = time duration

vs. H_0: no events occurring

Expectation-based scan statistics

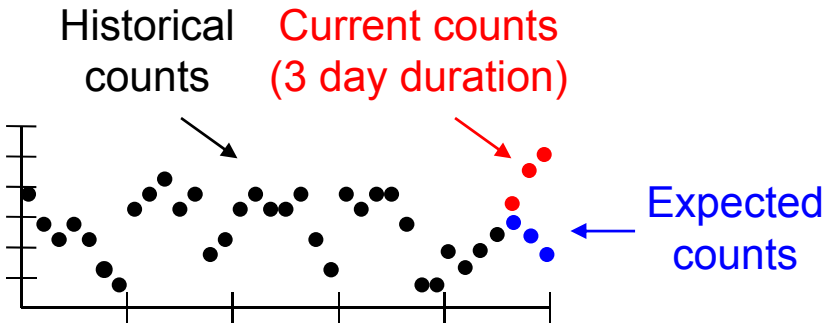
(Kulldorff, 1997; Neill and Moore, 2005)



We search for spatial regions (subsets of locations) where the recently observed counts for some subset of streams are significantly higher than expected.

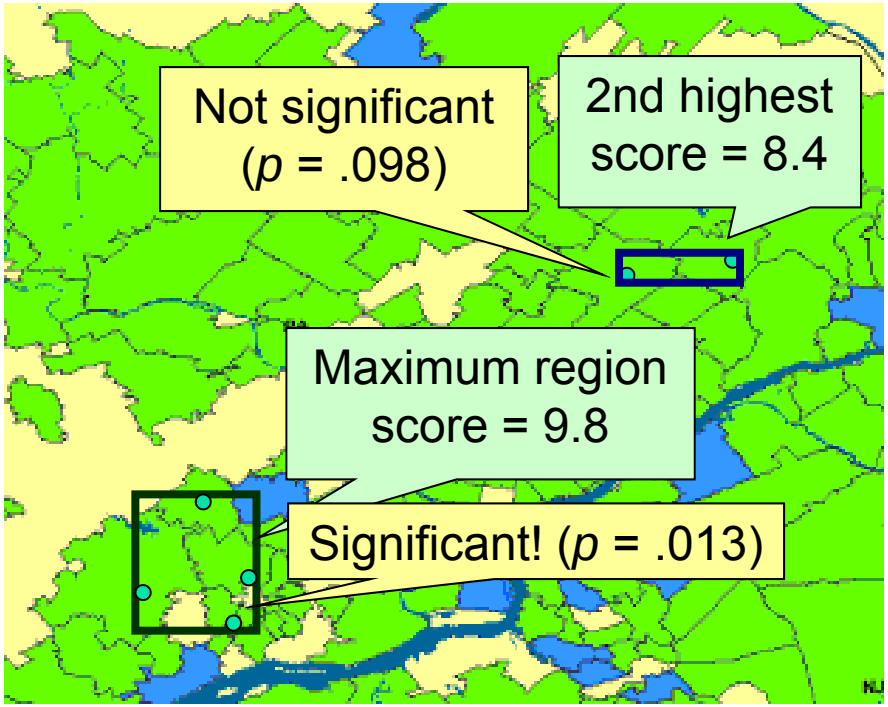
We perform time series analysis to compute expected counts ("baselines") for each location and stream for each recent day.

We then compare the actual and expected counts for each subset (D, S, W) under consideration.



Expectation-based scan statistics

(Kulldorff, 1997; Neill and Moore, 2005)

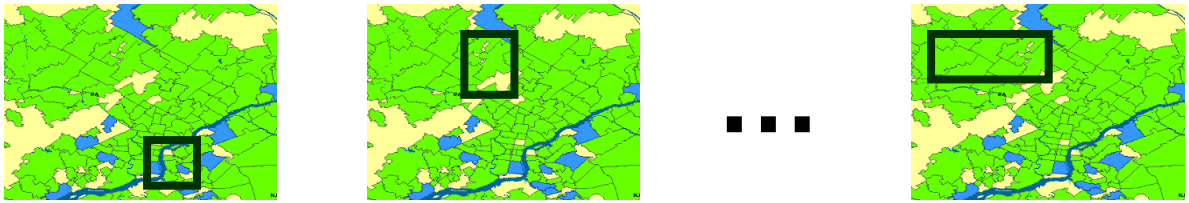


We find the regions with highest values of a likelihood ratio statistic, and compute the *p*-value of each region by randomization testing.

$$\mathrm{F}(S) = \frac{\Pr(\mathrm{Data} \mid H_1(D,S,W))}{\Pr(\mathrm{Data} \mid H_0)}$$

<u>To compute p-value</u>
Compare region score to maximum region scores of simulated datasets under H_0.

$F_1^* = 2.4$ $F_2^* = 9.1$ ... $F_{999}^* = 7.0$



Likelihood ratio statistics

The univariate log-likelihood ratio statistic F(C, B) is a function of the aggregate count and baseline.

For the **expectation-based Poisson** (EBP) statistic:
F(C, B) = C log (C / B) + B – C, if C > B, and 0 otherwise.

Burkom's multivariate spatial scan statistic

Assumes a **constant effect** over all affected data streams, computed by maximum likelihood estimation.

$F(D, S, W) = F(C, B)$

C and B are aggregated over all affected data streams $d_m \in D$ and all affected spatial locations $s_i \in S$, for the most recent W days.

Kulldorff's multivariate spatial scan statistic

Assumes **independent effects** on each data stream, each estimated separately by maximum likelihood.

$F(D, S, W) = \sum_m F(C_m, B_m)$

C_m and B_m are aggregated over all affected spatial locations $s_i \in S$, for the given data stream d_m and for the most recent W days.

Likelihood ratio statistics

The univariate log-likelihood ratio statistic F(C, B) is a function of the aggregate count and baseline.

For the **expectation-based Poisson** (EBP) statistic:
F(C, B) = C log (C / B) + B – C, if C > B, and 0 otherwise.

Burkom's multivariat… …variate
spati… …tic

Assumes … …**s** on
affecte… …ated
m… …od.

C ar… …ed over all
affecte… …ons $s_i \in S$, for
all affecte… …eam d_m and for
for… …e most recent W days.

Two main goals of this work:

Enable both Burkom and Kulldorff multivariate scan statistics to **scale up** to massive and high dimensional datasets.

Compare the performance of these two methods for event detection and characterization.

Which regions to search?

- Typical approach: each search region S is a subregion of the search space.
  - Choose some region shape (e.g. circles, rectangles) and consider all regions of that shape and varying size.
  - Low power for true events that do not correspond well to the chosen set of search regions (e.g. irregular shapes).
- Alternate approach: each search region S represents a distinct subset of the N locations.
  - Find the highest scoring subset, subject to some constraints (e.g. spatial proximity, connectivity).
  - For multivariate, also optimize over subsets of the M monitored data streams.
  - Exponentially many possible subsets, $O(2^N \times 2^M)$: computationally infeasible for naïve search.

The LTSS property

- In certain cases, we can search over the exponentially many subsets in linear time!
- Many commonly used scan statistics have the property of linear-time subset scanning:
  - Just sort the data records from highest priority to lowest priority according to some criterion…
  - … then search over groups consisting of the top-k highest priority records, for k = 1..N.

The highest scoring subset is guaranteed to be one of these!

The LTSS property

- Example: Poisson statistics (Kulldorff, EBP)
  - Sort locations s_i by the ratio of observed to expected count, c_i / b_i.
  - Given the ordering $s_{(1)} \ldots s_{(N)}$, we can **prove** that the top-scoring subset F(S) consists of the locations $s_{(1)} \ldots s_{(k)}$ for some k, $1 \leq k \leq N$.
- Also holds for Gaussian, nonparametric, …
- LTSS gives highest-scoring subset by evaluating **N** subsets instead of $\mathbf{2^N}$ for naïve search.
  - Sample result: we can find the most anomalous subset of 97 western PA zip codes in **.03 sec** vs. $\mathbf{10^{24}}$ **years**.
  - How to incorporate spatial proximity constraints?

Fast localized scan

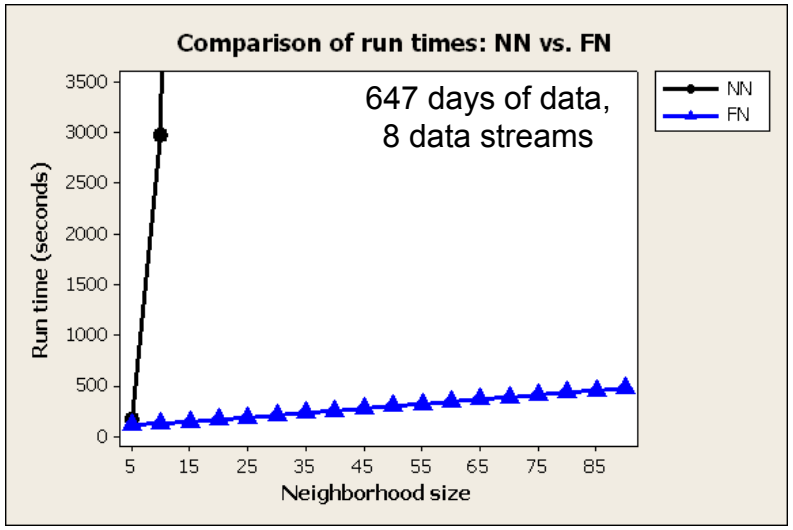
- Maximize the spatial scan statistic over all subsets of the "local neighborhoods" consisting of a center location s_i and its k – 1 nearest neighbors, for a fixed neighborhood size k.
- This is similar to Tango and Takahashi's flexible scan statistic, but may find a disconnected region.
- Naïve search requires $O(N \cdot 2^k)$ time and is computationally infeasible for k > 25.
- For each center, we can search over all subsets of its local neighborhood in O(k) time using LTSS, thus requiring a total time complexity of O(Nk) + O(N log N) for sorting the locations.

Fast multivariate scans

How can we efficiently search over all subsets of data streams and over all proximity-constrained subsets of locations? Let's start with Burkom's multivariate scan.

Option 1 (fast/naïve, or FN): for each of the 2^M subsets of streams, aggregate counts and apply LTSS to efficiently search over subsets of locations.

Guaranteed to find the highest scoring subset!



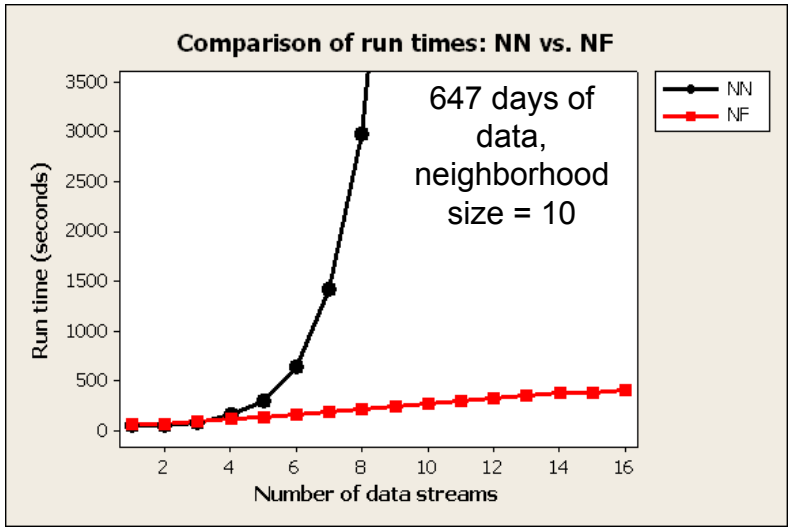
For a fixed number of streams, FN fast localized scan scales linearly (not exponentially) with neighborhood size.

8 streams: <1 sec/day of data.

Fast multivariate scans

How can we efficiently search over all subsets of data streams and over all proximity-constrained subsets of locations? Let's start with Burkom's multivariate scan.

Option 2 (naïve/fast, or NF): exhaustively search over spatial regions. For each, perform efficient LTSS search over subsets of streams.



Guaranteed to find the highest scoring subset!

For a fixed neighborhood size k, NF fast localized scan scales linearly (not exponentially) with number of streams.

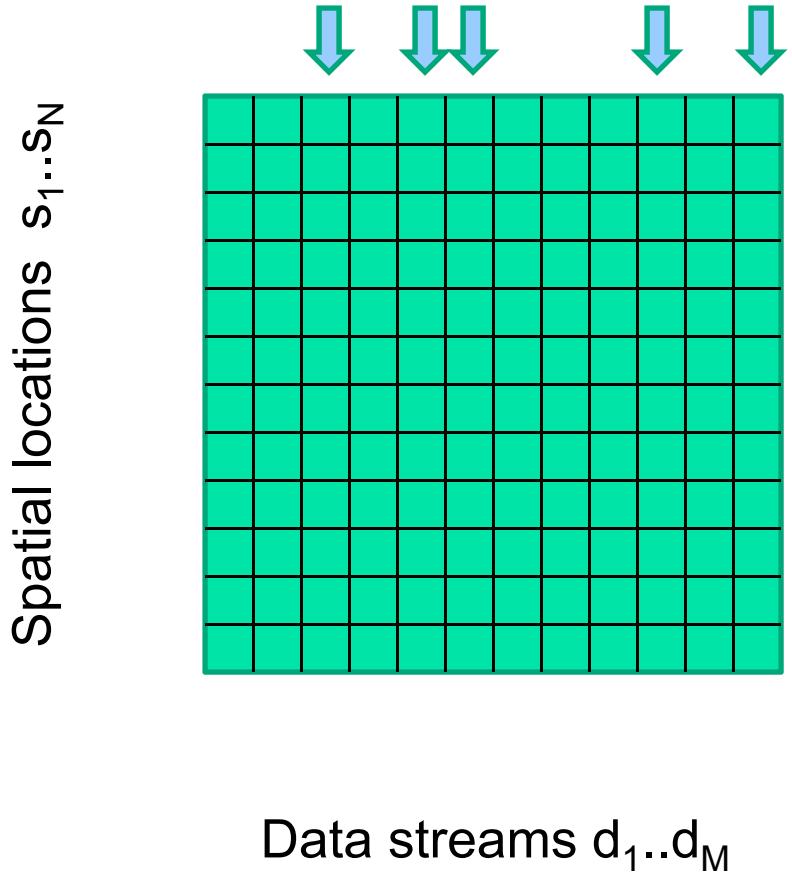
For k = 10: <1 sec/day of data

Fast multivariate scans

What if we have a large set of search regions and many data streams?

Option 3 (fast/fast, or FF):

1. Start with a randomly chosen subset of streams.



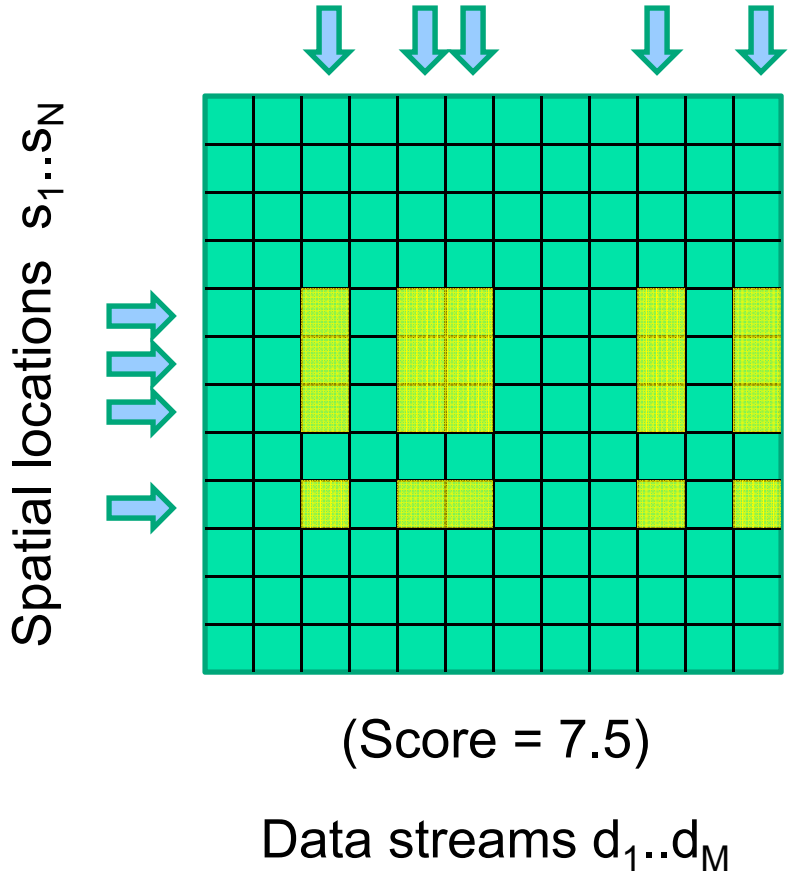
Fast multivariate scans

What if we have a large set of search regions and many data streams?

Option 3 (fast/fast, or FF):

1. Start with a randomly chosen subset of streams.

2. Use LTSS to efficiently find the highest-scoring subset of locations for the given streams.



Fast multivariate scans

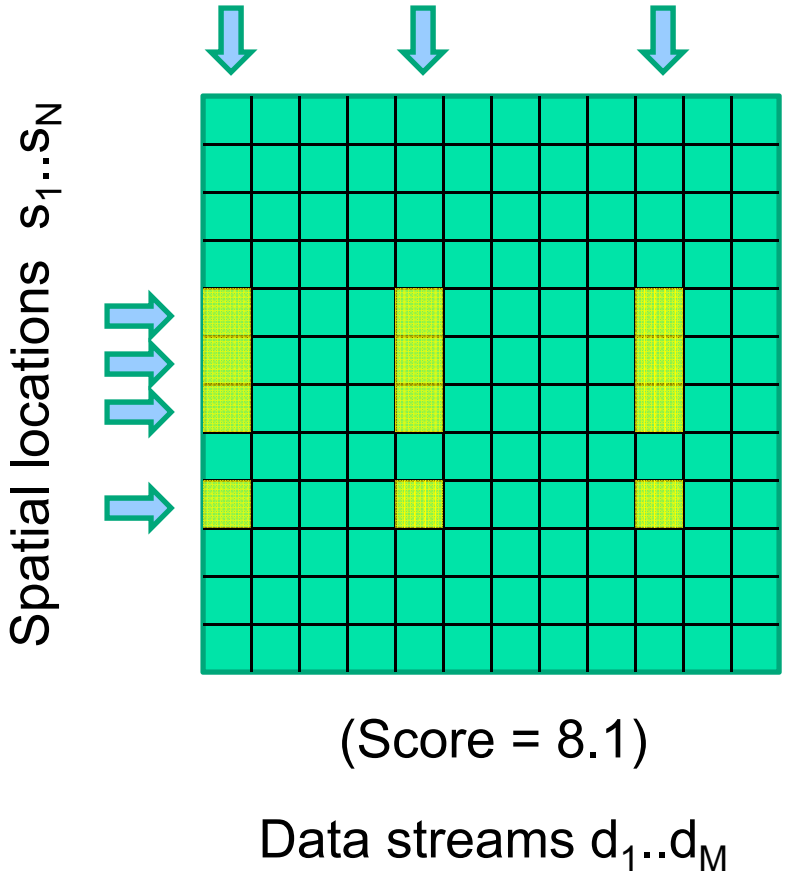
What if we have a large set of search regions and many data streams?

Option 3 (fast/fast, or FF):

1. Start with a randomly chosen subset of streams.

2. Use LTSS to efficiently find the highest-scoring subset of locations for the given streams.

3. Use LTSS to efficiently find the highest-scoring subset of streams for the given locations.



Fast multivariate scans

What if we have a large set of search regions and many data streams?

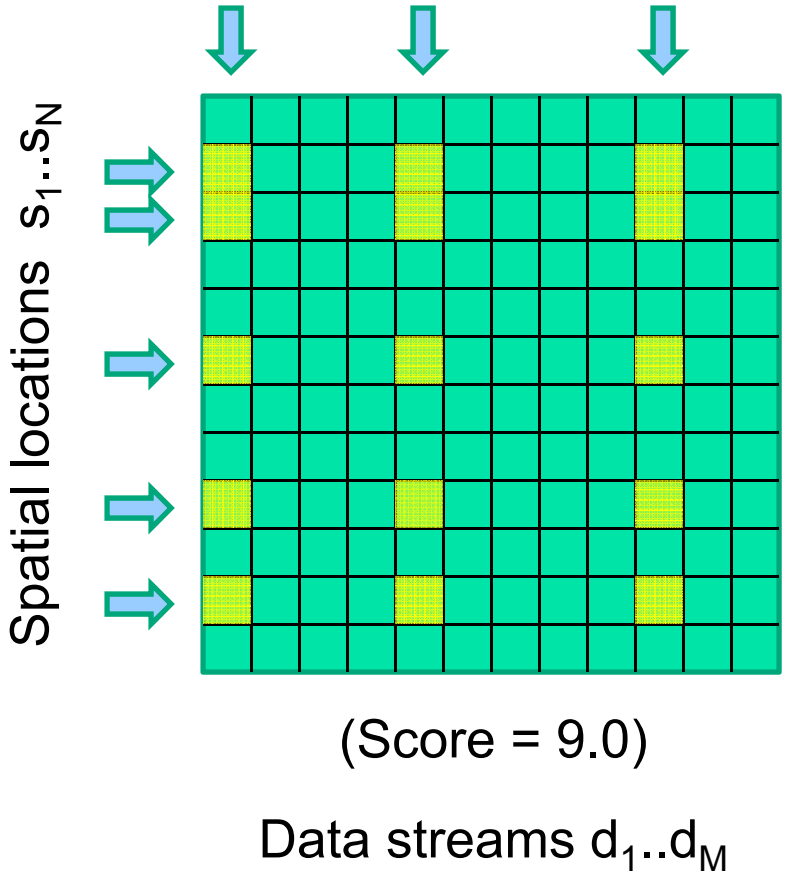
Option 3 (fast/fast, or FF):

1. Start with a randomly chosen subset of streams.

2. Use LTSS to efficiently find the highest-scoring subset of locations for the given streams.

3. Use LTSS to efficiently find the highest-scoring subset of streams for the given locations.

4. Iterate steps 2-3 until convergence.



Fast multivariate scans

What if we have a large set of search regions and many data streams?

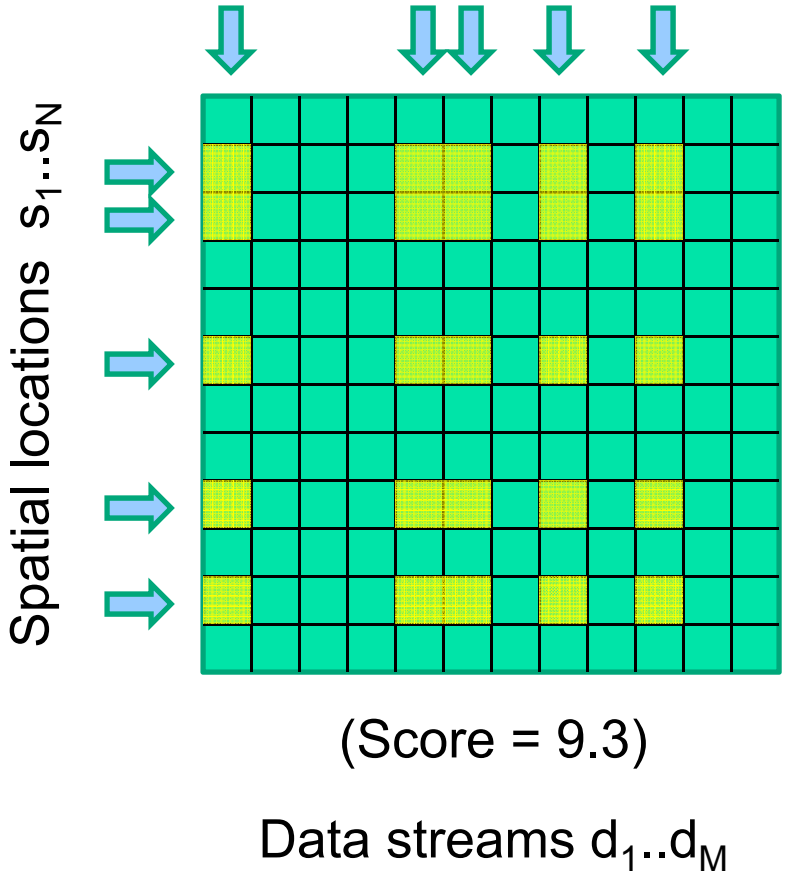
Option 3 (fast/fast, or FF):

1. Start with a randomly chosen subset of streams.

2. Use LTSS to efficiently find the highest-scoring subset of locations for the given streams.

3. Use LTSS to efficiently find the highest-scoring subset of streams for the given locations.

4. Iterate steps 2-3 until convergence.



Fast multivariate scans

What if we have a large set of search regions and many data streams?

Option 3 (fast/fast, or FF):

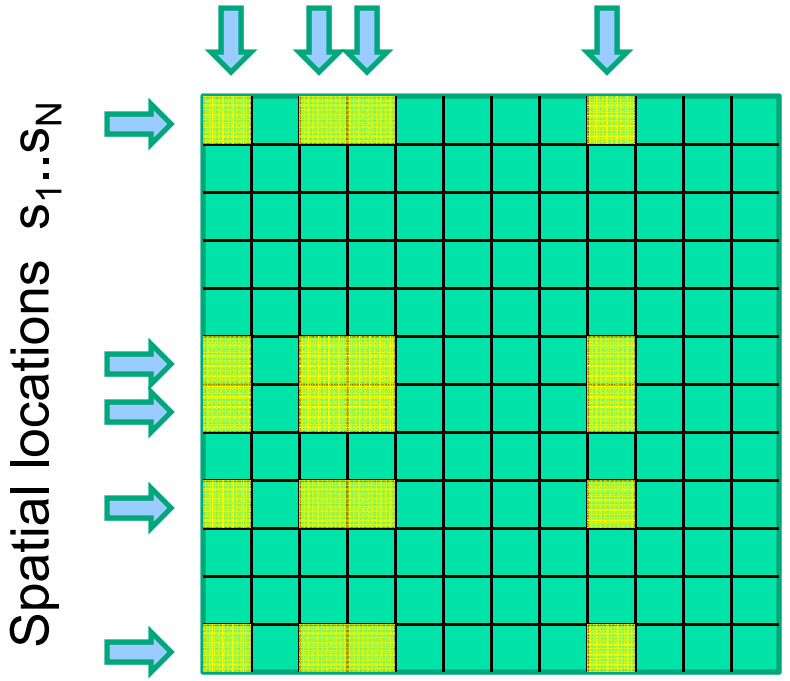
1. Start with a randomly chosen subset of streams.

2. Use LTSS to efficiently find the highest-scoring subset of locations for the given streams.

3. Use LTSS to efficiently find the highest-scoring subset of streams for the given locations.

4. Iterate steps 2-3 until convergence.

5. Repeat steps 1-4 for 50 random restarts.



Fast multivariate scans

What if we have a large set of search regions and many data streams?

Option 3 (fast/fast, or FF):

1. Start with a randomly chosen subset of streams.

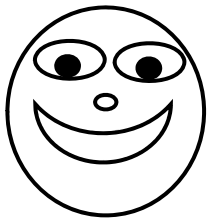
2. Use LTSS to efficiently find the highest-scoring subset of locations for the given streams.

3. Use LTSS to efficiently find the highest-scoring subset of streams for the given locations.

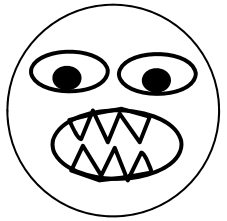
4. Iterate steps 2-3 until convergence.

5. Repeat steps 1-4 for 50 random restarts.

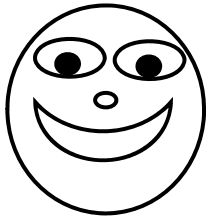
GOOD NEWS:
Run time is linear in number of locations & number of streams.



BAD NEWS:
Not guaranteed to find global maximum of the score function.



MORE GOOD NEWS:
200x faster than FN for 16 streams, and >98% approximation ratio.



Fast multivariate scans

What if we have a large set of search regions and many data streams?

Kulldorff's multivariate scan treats each stream independently, so it already scales efficiently with the number of streams…

… but searching over the exponentially many irregularly shaped regions (subsets of locations) is more difficult.

Our solution (FK) is similar to the FF algorithm for Burkom's multivariate scan, except that we must condition not on the affected subset of streams, but on the assumed relative risk for each stream.

We can efficiently optimize over subsets of locations for a given set of risks, using the LTSS property.

We can easily compute the maximum likelihood risk estimates for a given subset of locations.

Iterate between these two steps until convergence!

Burkom vs. Kulldorff comparison

Using our new, fast algorithms, we evaluated the Burkom and Kulldorff multivariate scans on semi-synthetic outbreak detection tasks for 16 streams of Emergency Department data from Allegheny County, PA.

For both methods, searching over proximity-constrained subsets of locations resulted in 1 to 2 days faster detection, and significantly improved spatial accuracy (overlap), as compared to circular scan.

Comparing Burkom vs. Kulldorff methods, we found similar run times (Burkom 2-3x faster), and spatial accuracy was almost identical.

We observed an interesting tradeoff between the two methods' detection power and ability to characterize the affected streams.

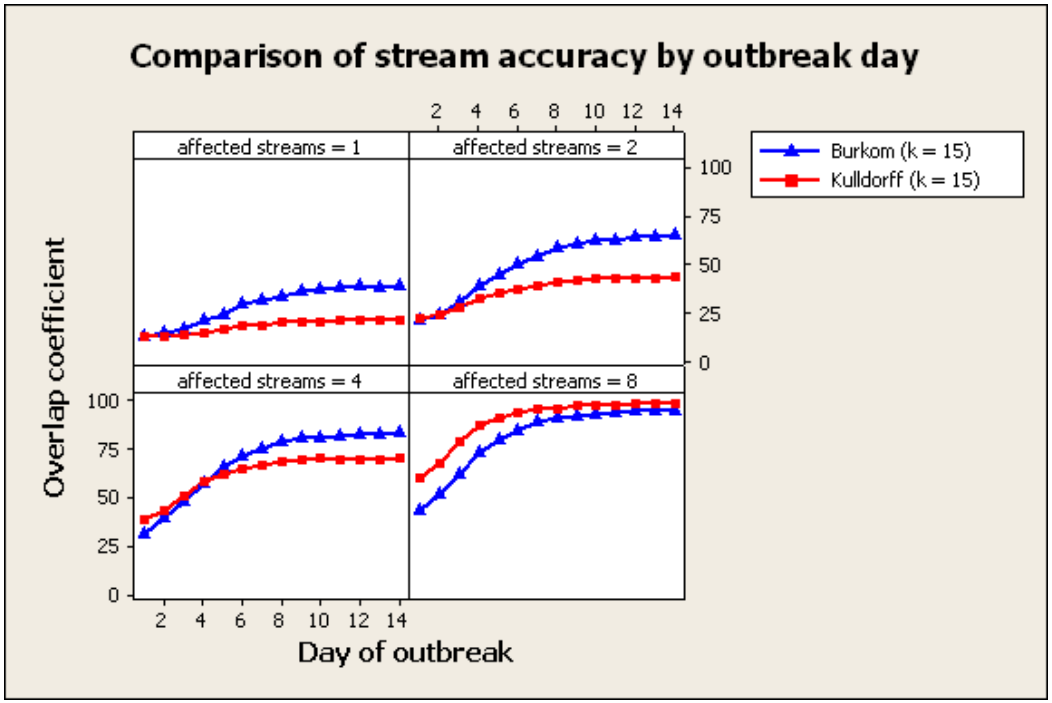
Kulldorff's method tended to detect slightly faster than Burkom's: 0.5 days for M = 2 streams, and 0.2 to 0.3 days for larger values of M.

Burkom vs. Kulldorff comparison

Using our new, fast algorithms, we evaluated the Burkom and Kulldorff multivariate scans on semi-synthetic outbreak detection tasks for 16 streams of Emergency Department data from Allegheny County, PA.

However, Burkom's method was better able to identify the affected subset of streams.

Kulldorff's tended to report many unaffected streams as affected.



Discussion

The choice between the Burkom and Kulldorff versions of the multivariate scan statistic depends on whether our primary goal is **early detection** or **accurate characterization** of outbreaks.

Our fast algorithms, based on extensions of linear-time subset scanning to the multivariate case, enable either version to be computed efficiently, even for many locations and many streams.

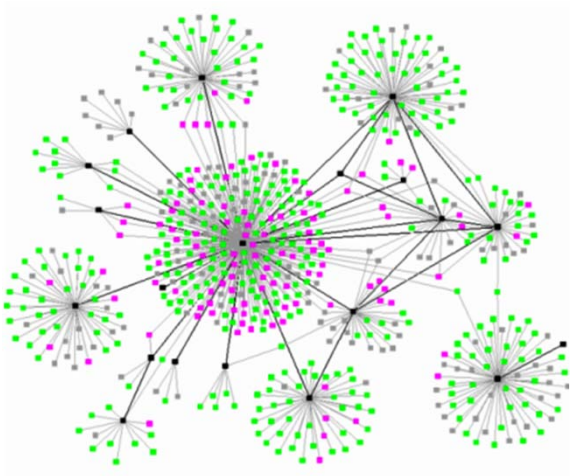
By scanning over all subsets of streams, and over all proximity-constrained subsets of locations, we can dramatically improve our ability to detect and characterize emerging outbreaks of disease.

For Burkom's multivariate scan, we have recently extended our FF algorithm to graph/network and tensor data, allowing us to scan over **connected** subsets of locations, **related** subsets of data streams, and **subpopulations** with different sets of demographic characteristics.

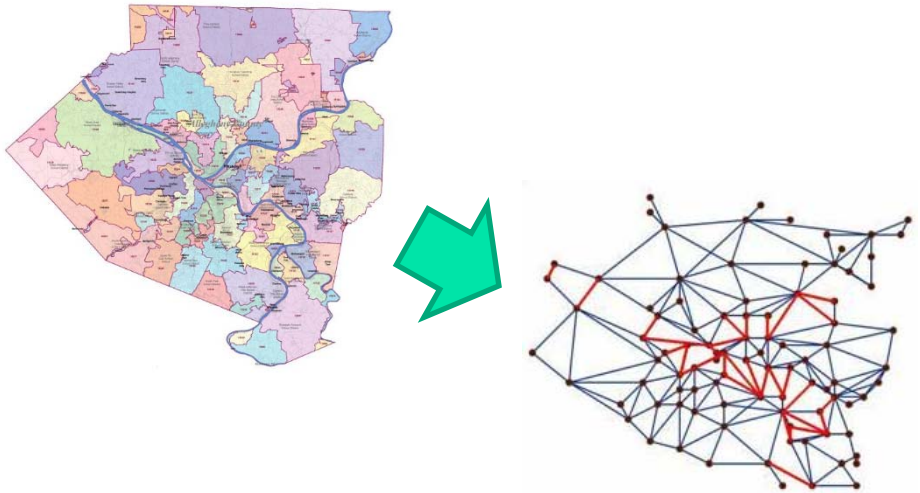
Incorporating connectivity constraints

Proximity-constrained subset scans may return a disconnected subset of the data.

In some cases this may be undesirable, or we might have non-spatial data so proximity constraints cannot be used.



Example: tracking disease spread from person-to-person contact.



Example: identifying a **connected** subset of zip codes (Allegheny County, PA)

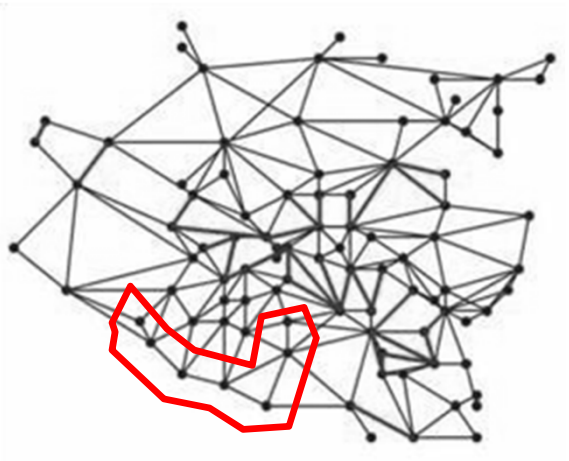
Incorporating connectivity constraints

Proximity-constrained subset scans may return a disconnected subset of the data.

In some cases this may be undesirable, or we might have non-spatial data so proximity constraints cannot be used.

Our **GraphScan** algorithm* can efficiently and exactly identify the highest-scoring connected subgraph:

- Can incorporate multiple data streams
  - With or without proximity constraints
  - Graphs with several hundred nodes



We can use the LTSS property to rule out subgraphs that are provably suboptimal, dramatically reducing our search space.

*Speakman and Neill, 2009; Speakman et al., 2011

Incorporating connectivity constraints

We represent groups of subsets as strings of 0's, 1's, and ?'s.

Assume that the graph nodes are sorted from highest priority to lowest priority.

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

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

LTSS property <u>without</u> connectivity constraints:
"If node $x \in S$ and node $y \notin S$, for $x > y$,
then subset S cannot be optimal."

We can use the LTSS property to rule out subgraphs that are provably suboptimal, dramatically reducing our search space.

Incorporating connectivity constraints

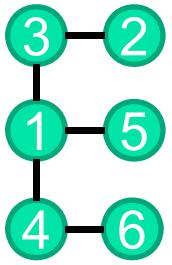
We represent groups of subsets as strings of 0's, 1's, and ?'s.

Assume that the graph nodes are sorted from highest priority to lowest priority.

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

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

LTSS property with connectivity constraints:
"If node x ∈ S and node y ∉ S, for x > y,
and S – {x} and S + {y} are both connected,
then subset S cannot be optimal."



We can use the LTSS property to rule out subgraphs that are provably suboptimal, dramatically reducing our search space.

Incorporating connectivity constraints

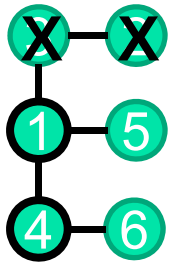
We represent groups of subsets as strings of 0's, 1's, and ?'s.

Assume that the graph nodes are sorted from highest priority to lowest priority.

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

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

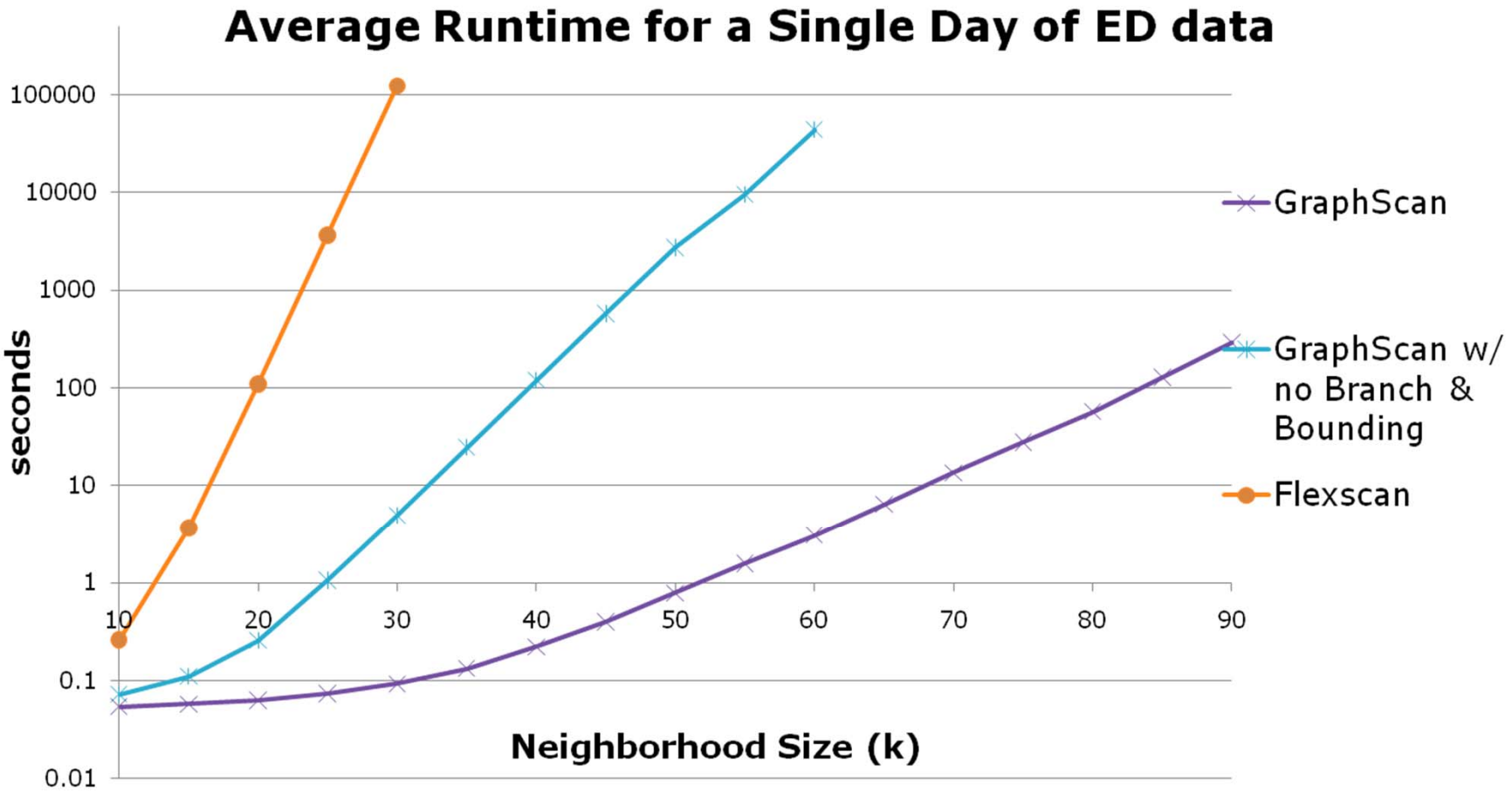
LTSS property with connectivity constraints:
"If node x ∈ S and node y ∉ S, for x > y,
and S – {x} and S + {y} are both connected,
then subset S cannot be optimal."



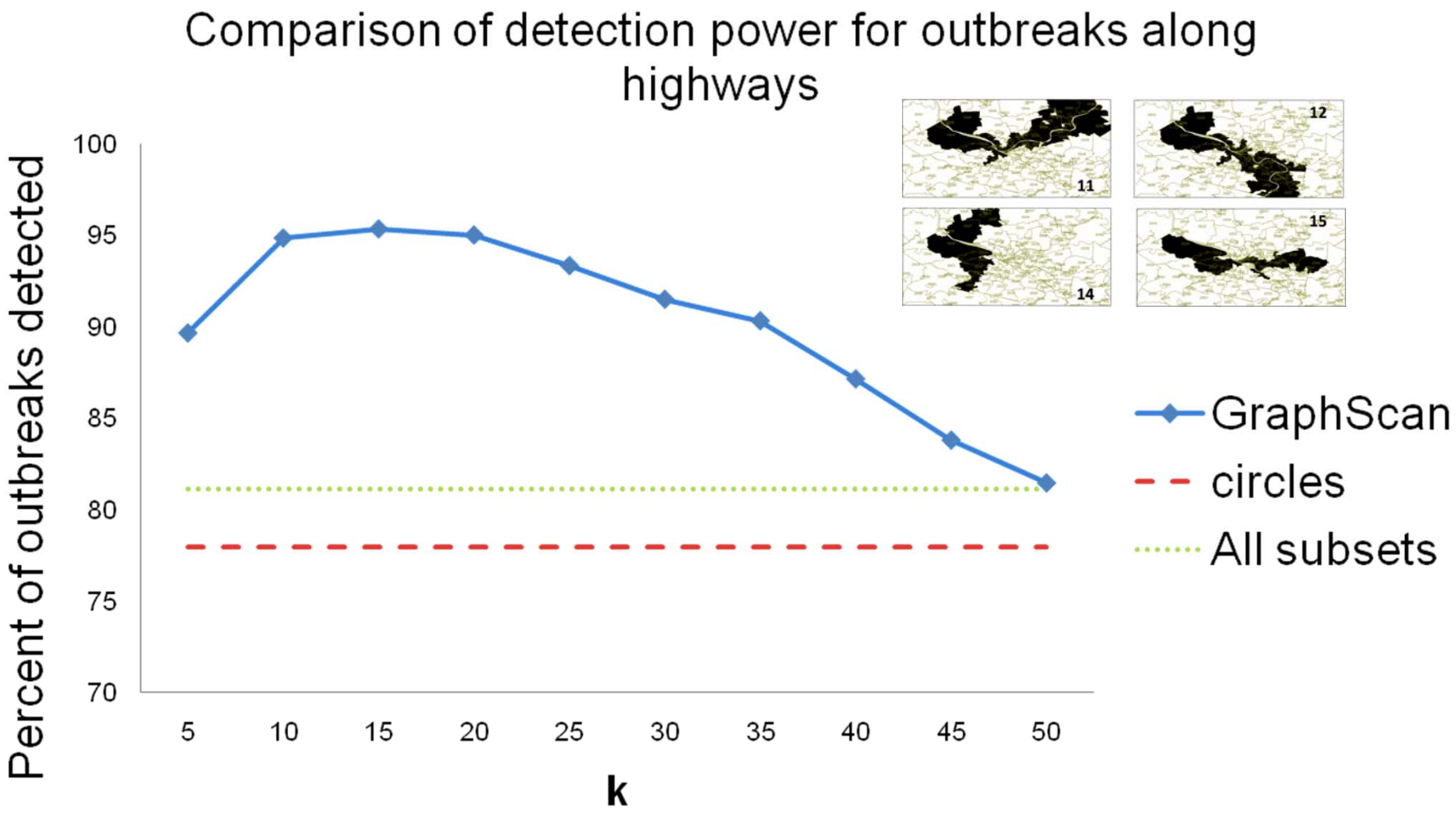
suboptimal

We can use the LTSS property to rule out subgraphs that are provably suboptimal, dramatically reducing our search space.

Evaluation: run times



Evaluation: detection power



Current and future work

Speeding up the GraphScan Algorithm

Exponentially many multiple paths between nodes represents a significant bottleneck.

Better handling of this will allow us to scale to even larger graphs.

Non-spatial Applications (e.g. spread of innovations)

Cell phone and SMS network:

Early results have shown that we can detect the most active connected group of ‘texters’ in a graph of 300 users in 16 minutes.

Dynamic Graphs

Allow edges and nodes to enter and leave the graph over time.

Enables us to apply GraphScan to more complex settings, e.g. supply and transportation networks.

References

- **D. B. Neill, E. McFowland III, and H. Zheng. Fast subset scan for multivariate event detection. Submitted for publication, 2011.**
- S. Speakman, E. McFowland III, and D. B. Neill. Scalable detection of anomalous patterns with connectivity constraints. In preparation.
- S. Speakman and D. B. Neill. Fast graph scan for scalable detection of arbitrary connected clusters. *Proc. ISDS Annual Conference*, 2009.
- D. B. Neill. Fast subset scan for spatial pattern detection. *Journal of the Royal Statistical Society* (Series B: Statistical Methodology), accepted for publication, 2011.
- D. B. Neill. Fast and flexible outbreak detection by linear-time subset scanning. *Advances in Disease Surveillance* 5:48, 2008.
- H. S. Burkom. Biosurveillance applying scan statistics with multiple disparate data sources. *J. Urban Health* 80: i57-i65, 2003.
- M. Kulldorff, F. Mostashari, L. Duczmal, K. Yih, K. Kleinman, and R. Platt. Multivariate spatial scan statistics for disease surveillance. *Statistics in Medicine* 26:1824-1833, 2007.