Near-Optimal and Practical Algorithms for Graph Scan Statistics

Jose Cadena* Feng Chen† Anil Vullikanti*

Abstract

Scan statistics is a popular approach used for detecting "hotspots" and "anomalies" in spatio-temporal and network data. This methodology involves maximizing a score function over all connected subgraphs, which is NP-hard in general. A number of heuristics have been proposed for these problems, but they do not provide any quality guarantees. In this paper, we develop a framework for designing algorithms for optimizing a large class of scan statistics for networks, subject to connectivity constraints. Our algorithms run in time that scales linearly on the size of the graph and depends on a parameter we call the "effective solution size", while providing rigorous approximation guarantees. In contrast, most prior methods have super-linear running times in terms of graph size. Extensive empirical evidence demonstrates the effectiveness and efficiency of our proposed algorithms in comparison with state-of-the-art methods. Our approach improves on the performance relative to all prior methods, giving up to over 25% increase in the score. Further, our algorithms scale to networks with up to a million nodes, which is 1-2 orders of magnitude larger than all prior applications.

1 Introduction

A number of methods have been proposed for anomaly detection in different applications, but the paradigm of *scan statistics* is among the most powerful and widely used—these typically involve formalizing a notion of "anomalousness" and finding a subset that optimizes this metric (e.g., [23]). These methods were originally developed for spatial data and involve finding clusters that maximize a discrepancy score [11, 13, 16]. Recently, scan statistics have been extended to network data by considering scores for *connected subgraphs* [23, 5] and have been applied to a number of domains, such as epidemiology [22], systems biology [12, 8], and social network analysis [5] (see Section 7 for more details).

Depending on whether the notion of anomalousness is with respect to an underlying model for the data or historical values, the scan statistics can be *parametric* or *non-parametric* (Section 2). As a result of the diversity of applications, a large number of scan statistics have been developed. Maximizing functions over connected subgraphs generalizes *Network Design* problems—this includes well-known graph optimization problems, such as the Steiner Tree problem and its variants, Prize-Collecting Steiner Tree (PCST) and NetWorth, all NP-hard. Heuristics for these problems have been used for network scan statistics [4, 18, 23, 22], but they do not give any rigorous guarantees on the solution quality.

Our contributions are:

1. Rigorous algorithms: We develop a *unified framework for optimizing a large class of parametric and non-parametric scan statistics for networks with connectivity constraints*, which scales linearly in the network size and is a function of a parameter defined as the "effective solution size". We also give rigorous bounds on the solution quality (summarized in Theorem 3.1). In other words, our framework encompasses many different network scan statistics—this contrasts with all prior methods, which are developed for specific statistics; further, our approach also holds for the extensions of these functions with both node and edge weights, which generalize Steiner connectivity problems. *In practice, the effective solution size parameter is very small* (see Section 5.6), making the time complexity of our algorithms better than prior methods, which are super-linear in the network size.

2. Scaling: We develop a preprocessing and refinement technique that reduces the solution size without degrading the quality score beyond a provable constant factor (Section 3.4). The resulting algorithms are able to scale to graphs with over a million nodes in minutes and are significantly faster than state-of-the-art methods (Figure 1), which have only been run on graphs of up to 10^4 nodes.

3. Experimental results: We show that our algorithms give significant improvement over the scores computed by different baselines, with over 25% improvement in some instances, compared to the best baseline method (Section 5.3 and Table 1 below). Better objective scores also translate to higher anomaly detection power with 3% improvement on accuracy and F1 score over state-of-the-art methods. Our algorithmic framework has the added advantage that different score functions can be optimized by just modifying the specific objective function *within the same implementation.*

*Dept. of Computer Science and BI. Virginia Tech
†Dept. of Computer Science. University at Albany - SUNY

Table 1: **Non-parametric scan statistics optimization.** Our algorithm FastColCodeNP obtains higher scores than previous methods in real-world datasets. (Section 5.3)

	Berk-Jones Scan Statistic					
	FastColCodeNP	GS	EL	GL	DFS	NPHGS
CitHepPh	**1138.011**	1135.029	559.176	1118.352	1130.874	1118.353
NEast	**68.525**	64.214	23.366	23.211	55.541	17.696
Traffic	**128.740**	128.722	116.732	125.824	14.412	121.632
Twitter	**1722.790**	1722.388	1722.388	1722.388	1720.243	1457.410
BWSN	**602.164**	599.972	530.850	530.457	536.200	531.280

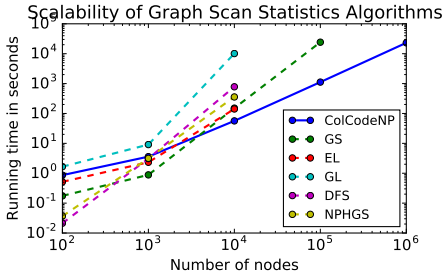

Figure 1: **Scalability.** Running time as a function of number of nodes. (Section 5.5)

For brevity, some of the results, including hardness of the problems, proof details, and pseudocode, are deferred to the supplementary material[1].

2 Preliminaries

We are given a graph $G = (V, E)$, where V is a set of n vertices or nodes, and $E \subseteq V \times V$ is a set of m edges. Each vertex $v \in V$ has two values associated with it: (1) a *population* count, $b(v)$, which indicates the count that we expect to see at the node v, e.g., the number of people in a county, corresponding to node v, and (2) an *event count*, $c(v)$, which indicates how many occurrences of an event of interest are seen at the node, e.g., the number of cases of a disease in a county. These values vary over time, but we will not indicate the time in the notation in order to keep it simple. The notation used in this paper is summarized in Table 3.

2.1 Non-Parametric Scan Statistics. Non-parametric scan statistics do not assume an underlying distribution or process on the graph. Instead, they first estimate a p-value for each vertex based on empirical calibration by comparing the current feature ($c(v)$ and $b(v)$) of this vertex with its features in the historical data. The problem of anomaly detection has been formalized as a hypothesis testing problem for testing whether the empirical p-values are uniformly distributed on $[0, 1]$ [17, 19, 16]. Let $\alpha \in [0, 1]$ be *significance level* and let $w(v, \alpha)$ denote the *weight* of a node v as a function of α. For a set of nodes S, let $W(S, \alpha) = \sum_{v \in S} w(v, \alpha)$ and $N(S)$ be a function of the cardinality of the set. Then, the score functions can be expressed in the following general form:

$$F(S) = \max_{\alpha \leq \alpha_{max}} \phi(W(S, \alpha), N(S), \alpha), \tag{2.1}$$

The significance level α can be optimized between 0 and some constant α_{max}. We use $w(v)$ and $W(S)$ to denote $w(v, \alpha)$ and $W(S, \alpha)$, respectively, whenever α is clear from the context. For clarity, in the rest of the paper, we consider the specific case when $N(S) = |S|$ is the cardinality of S and $w(v, \alpha)$ is 1 if the p-value of v is less than α, 0 otherwise —we say these nodes are *significant at level* α. Then, $W(S, \alpha)$ is the number of significant nodes in S. It is easy to generalize our results to other functions $W(S, \alpha)$ and $N(S)$. Table 2 shows examples of non-parametric scan statistics.

2.2 Parametric Scan Statistics. Parametric scan statistics assume that counts observed at each node are generated from some parameterized distribution and formalize anomaly detection as a hypothesis testing problem [11, 16]. Common choices are distributions from the exponential family, such as Poisson or Normal. The goal is to maximize an appropriate scan statistic function $F(S)$, typically a likelihood ratio. These score functions can be expressed as

$$F(S) = g(C(S), B(S)), \tag{2.2}$$

where $C(S) = \sum_{v \in S} c(v)$, $B(S) = \sum_{v \in S} b(v)$, and the function g is defined depending on the score function considered. We refer to [11, 16, 17] for discussion on the strengths and limitations of parametric scan statistics.

2.3 Problem Formulation. Given a graph $G = (V, E)$, a score function $F(\cdot)$ and the associated counts for the model, the objective is to find a connected subset $S \subseteq V$ that maximizes $F(S)$.

Remark. As we show in the supplementary material, the above problem is NP-hard. In Section 3, we develop algorithms to maximize $F(S)$, restricted to sets with $|S| \leq k$, where k is a parameter that represents the *solution size*. In Section 3.4, we show that we can compress specific subsets of nodes into "supernodes", using a process we refer to as *refinement*. The size of a set S computed in terms of these supernodes will be referred to as the *effective solution size*, and it becomes significantly smaller than the original size of S. Our final algorithms will find solutions with effective solution size at most k.

3 Algorithms for Non-Parametric Scan Statistics

In this section, we present an algorithm for non-parametric functions that are characterized by equation

[1] http://people.cs.vt.edu/~jcadena/sdm17-scan-statistics/sdm17-scan-statistics-supp.pdf

Table 2: A common formulation for parametric and non-parametric scan statistics that can be optimized using our methods, along with their applications. Many other functions are described in the supplementary material.

Non-Parametric Scan Statistics $F(S) = \max_{\alpha \leq \alpha_{max}} \phi(W(S,\alpha), N(S), \alpha)$ $W(S,\alpha)$ is the number of significant nodes at level α			
Name	**Original Form**	**General Form**	**Applications**
Berk-Jones [3]	$F(S) = \max_{\alpha \leq \alpha_{max}} \lvert S\rvert KL(\frac{W(S,\alpha)}{\lvert S\rvert}, \alpha)$.	$\phi(a,b,\alpha) = b \cdot KL(a/b,\alpha)$, where KL is the KL-divergence: $KL(x,\alpha) = x\log\left(\frac{x}{\alpha}\right) + (1-x)\log\left(\frac{1-x}{1-\alpha}\right)$	Detection of disease outbreaks, civil unrest events, and human rights events in social media graphs [5] ($\lvert V\rvert$ = 10,000 to 80,000), network intrusion detection [13] ($\lvert V\rvert$ = 1,000 to 10,000), detection of illicit activities in container shipment data [13] ($\lvert V\rvert = 10,000$).
Higher Criticism [6]	$F(S) = \max_{\alpha \leq \alpha_{max}} \frac{W(S,\alpha)-\lvert S\rvert\alpha}{\sqrt{\lvert S\rvert\alpha(1-\alpha)}}$	$\phi(a,b,\alpha) = \frac{a-b\cdot\alpha}{\sqrt{b\cdot\alpha(1-\alpha)}}$	In addition to the same applications above, it was also used in detection of rare and week effects in Genomics and Genetics [9] ($\lvert V\rvert = 20,000$ to 30,000)
Parametric Scan Statistics $F(S) = g(C(S), B(S))$, $C(S) = \sum_{v\in S} c(v)$, $B(S) = \sum_{v\in S} b(v)$			
Elevated Mean Scan Statistic [17]	$F(S) = \sum_{i\in S} x_i/\sqrt{\lvert S\rvert}$	$g(a,b) = a/\sqrt{b}$	Activity detection in social networks, network surveillance, disease outbreak detection, biomedical imaging [17, 19] (N = 129 to 225).
Expectation-based Poisson Statistic [16]	$F(S) = C(S)\log(C(S)/B(S)) + B(S) - C(S)$	$g(a,b) = a\log(a/b) + b - a$	Disease outbreak detection [15] ($\lvert V\rvert$ = 58 to 88), water pollution detection [23] ($\lvert V\rvert = 12,000$).

(2.1) and then discuss techniques to scale it without losing the quality guarantees. Our algorithm relies on two main ideas, namely *monotonicity* and *constraining the solutions*.

Table 3: Definitions and notation used in the paper

Term	**Description**
$b(v)$, $c(v)$	population and event counts of node v
α, α_{max}	significance level, maximum significance level
Significant node (at level α)	a node with p value below α
$Nbr(v)$	set of neighbors of v
$w(v)$, $w(v,\alpha)$	weight of node v, based on its p-value and the significance level α
$W(S)$, $W(S,\alpha)$	denotes $\sum_{v\in S} w(v,\alpha)$
$F(S)$	any of the functions in Table 2
K	The set $\{1,\ldots,k\}$
$col(u)$	color of node u from set K
T	Subset of K (denotes colors)
$M(v,T)$	$\max_S W(S)$, where the maximization is over connected colorful sets $S \subseteq V$, such that $v \in S$ and $\{col(u) : u \in S\} = T$
$\psi_i, \psi_i(\alpha)$	$\max_{T:\lvert T\rvert=i} M(v,T)$. Maximum weight over connected colorful sets of size i
S_i^*, $S_i^*(\alpha)$	Set with weight ψ_i
$OPT(F,k)$	$\max_{S:\lvert S\rvert\leq k} F(S)$, where the maximum is over connected subsets S of size $\leq k$

3.1 First idea: Monotonicity. A key observation is that the functions $\phi(W(S,\alpha), N(S), \alpha)$ are monotonically increasing functions of $W(S,\alpha)$ under some conditions, as described below.

LEMMA 3.1. *The non-parametric scan statistics functions characterized by equation (2.1) are increasing functions of $W(S,\alpha)$ if $\frac{W(S,\alpha)}{N(S)} \geq \alpha$ and $N(S)$ is constant.*

For example, in the Berk-Jones (BJ) statistic from Table 2, the function increases with the number of significant nodes—nodes with p-value less than α. Further, given two node sets of the same size, the set with more significant nodes scores higher according to the BJ statistic. This provides us a way to optimize $F(S)$ by maximizing the function $\phi(\cdot)$ for sets of fixed size, as will be discussed next.

3.2 Second idea: Constraining the solutions. We introduce the idea of a *coloring* of the nodes and only consider connected subgraphs S, in which all nodes have distinct colors. Our approach builds on the color-coding technique of [2], but it involves several new techniques to scale the algorithm up to graphs with millions of nodes.

Let $K = \{1, 2, \ldots k\}$ be a set of colors —where k is a parameter— and let $col(v) \in K$ denote the color for node v. We say that a subgraph induced by set $S \subseteq V$ is *colorful* if $col(u) \neq col(v)$, for all $u, v \in S$. For a node v and subset of colors $T \subseteq K$, we let $M(v,T) = \max_S W(S)$, where the maximization is over all connected and colorful sets $S \subseteq V$, such that $v \in S$, $\lvert S\rvert = \lvert T\rvert$, and $\{col(u) : u \in S\} = T$. In other words, we only consider a set S if each node in the set has a distinct color from T. These definitions are illustrated in Figure 2. $M(v,T)$ can be computed by a dynamic program with a recurrence given in the lemma below.

LEMMA 3.2. *Let $M(v,T)$ be defined as above. For any node v and color s, $M(v,\{s\}) = w(v)$ if $col(v) = s$, else $M(v,\{s\}) = -\infty$. If $\lvert T\rvert \geq 2$:*
$M(v,T) = \max_{\substack{u\in Nbr(v)\\ T_1,T_2\subseteq T}} \{M(v,T_1) + M(u,T_2)\}$, *where the maximum is over all partitions $T_1 \cup T_2 = T$.*

3.3 ColCodeNP In Algorithm 1, we present COLCODENP for optimizing non-parametric statistics. Re-

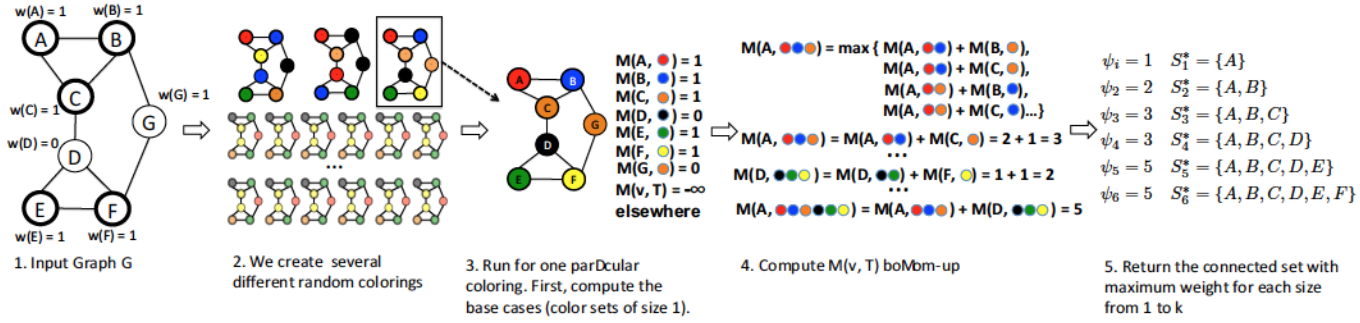

Figure 2: Example illustrating the MaxWeight procedure for $k = 6$ colors.

call the notation in Table 3. Let $F(S)$ denote any of the non-parametric functions in Table 2, and let $OPT(F,k) = \max_{S:|S|\leq k} F(S)$, where the maximum is over all connected subsets S of size $\leq k$, for a given α_{max}. Algorithm ColCodeNP takes the size bound k as input, and an *error parameter* ϵ, which indicates the probability of not finding the optimum solution.

Main steps. We describe the main steps of ColCodeNP connecting with the two ideas from above.

- The set A in line 3 of ColCodeNP denotes the set of distinct p-values of the nodes less than α_{max}; it suffices to find the maximum of $\phi(W(S,\alpha), N(S), \alpha)$ for $\alpha \in A$. The **for** loop in lines 4—6 finds the best solution for each $i \in K$ and any given α (by calling MaxWeight in line 6), and the maximum is computed in line 7.
- MaxWeight finds the best solution S_i^* of size i, for each $i \in K$ using the idea described in Section 3.2. We show an example in Figure 2.
- Each iteration of the outer **for** loop in lines 14—20 starts with a random coloring (line 15). This is Step 2 in Figure 2.
- The inner **for** loop in lines 16—17 computes the base case of the dynamic program from Lemma 3.2; then, we solve the program bottom-up in lines 18–19. These are Steps 3 and 4 in Figure 2.
- ψ_i keeps track of the maximum weight solution restricted to size i, and it is updated if $M(v,T)$ denotes a better solution for size $|T|$.

Theorem 3.1. *For any non-parametric function $F(\cdot)$ in Table 2, algorithm* ColCodeNP *returns solution S^* satisfying $\Pr[F(S^*) = OPT(F,k)] \geq 1-\epsilon$, in time $O(2^k e^k |A| m \log(n^2/\epsilon))$, and using space $O(2^k n)$, where A is the set defined in line 3 of Algorithm 1.*

Proof. (**Sketch**) We start with the proof of correctness of our algorithm, which involves three parts. The first observation is that within the outer for loop in the procedure MaxWeight, for each random coloring $col(\cdot)$, $\max_v M(v, \{1,\ldots,k\})$ is correctly computed. This follows because the algorithm is a dynamic program that computes all $M(v,T)$ for $T \subseteq K$ using the recurrence in Lemma 3.2.

Next, we observe that the algorithm correctly finds $\psi_i(\alpha)$ —the maximum weight among sets of size i for a given α—for each i, α, with probability at least $1-\epsilon/n^2$. The procedure MaxWeight is called with parameter $\epsilon' = \epsilon/n^2$ and $e^k \log(1/\epsilon')$ colorings. Let X_{ij} be the maximum weight found over subsets of size i in the j^{th} random coloring. In the supplementary material, we show that $\Pr[\max_j X_{ij} \neq \psi_i(\alpha)] \leq \epsilon' = \epsilon/n^2$. The number of possible choices for α is $|A|$, which satisfies $|A| \leq n$. Therefore, by a union bound, it follows that for all $i, \alpha \in A$, we have $\Pr[\max_j X_{ij} \neq \psi_i(\alpha)] \leq n^2\epsilon' \leq \epsilon$, and the algorithm correctly computes $\psi_i(\alpha)$ for all i, α with probability $1-\epsilon$.

Finally, for any fixed i, α, by Lemma 3.1, $\phi(W(S), N(S), \alpha)$ is an increasing function of $W(S)$ when $N(S)$ is fixed. This implies that $\max_{S:N(S)=i} \phi(W(S), N(S), \alpha) = \psi_i(\alpha) = F(S_i^*(\alpha))$. Therefore, $\max_{i\in K, \alpha\in A} F(S_i^*(\alpha)) = OPT(F,k)$, and it follows that Algorithm ColCodeNP correctly computes $OPT(F,k)$ with probability at least $1-\epsilon$.

Next, we consider the space and time complexity. The algorithm maintains the array M, indexed by nodes and all possible color sets, which leads to the space complexity of $O(2^k n)$, since there are at most $2^k - 1$ possible non-empty color sets. The running time is the result of solving the recurrence for each node v, and for each color set T; this requires examining each possible partition $T_1 \cup T_2 = T$, and each neighbor $u \in Nbr(v)$, which requires time $O(|Nbr(v)|2^{|T|})$. Therefore, the total running time for each coloring is $O(\sum_v \sum_{i=0}^k |Nbr(v)|2^i) = O(\sum_v |Nbr(v)|2^k) = O(2^k m)$. The algorithm considers $O(e^k \log(n^2/\epsilon))$ colorings, so the running time follows.

3.4 Techniques for Scaling.

ColCodeNP has rigorous guarantees on the quality of the solution; however, if applied directly, it would only be feasible to discover small anomalies due to the exponential depen-

Algorithm 1 ColCodeNP$((G(V,E),\alpha_{max}),k,\epsilon)$.

Input: Instance $(G(V,E),\alpha_{max})$, parameters k, ϵ
Output: Set S^* with score $OPT(F,k)$
Let A be the set of p-values of nodes in V below α_{max}
for $\alpha \in A$
  Let $\mathbf{w}$ be a weight vector with $w(v) = w(v,\alpha)$
  $\{S_i^*(\alpha) : i \in K\} =$ MaxWeight$(G(V,E),\mathbf{w},k,\epsilon/n^2)$
$S^* = argmax_{i\in K,\alpha\in A} F(S_i^*(\alpha))$
return S^*

procedure MaxWeight$(G(V,E),\mathbf{w},k,\epsilon')$
Input: Instance $(G(V,E),\mathbf{w})$ and parameters k,ϵ'
Output: $\{S_i^* : i \in K\}$, such that S_i^* has weight ψ_i
Let $\psi_i = -\infty$ for all $i \in K$
for $j = 1$ to $e^k \log(1/\epsilon')$
  For each node v, pick random color $col(v) \in K$
  for $v \in V, s \in K$
    $M(v,\{s\}) = w(v)$ if $col(v) = s$; $-\infty$ otherwise
  for $v \in V$ and $T \subseteq K$, with $|T| \geq 2$
    Use Lemma 3.2 to compute $M(v,T)$
    If $M(v,T) > \psi_{|T|}$ **update** $\psi_{|T|} = M(v,T)$
return $\{S_i^* : \sum_{v\in S_i^*} w(v) = \psi_i, \text{for } i \in K\}$

dence in k, the solution size. We discuss two techniques to scale ColCodeNP to networks with over a million nodes without losing the approximation guarantees significantly.

Graph refinement and effective solution size. Graph refinement involves compressing subsets of nodes into "supernodes". The size of a set S after refinement is determined in terms of the nodes and supernodes in it. This new size is called *effective size*, and, in practice, it is significantly smaller than the original size of S.

We observe that neighboring significant nodes can be merged without loss in quality. We illustrate with an example in Figure 3a. In the figure, orange nodes are significant and have weight 1. The key idea is that any solution containing node A should also include nodes B and C, since $\phi(W(S,\alpha),N(S),\alpha)$ is increasing in the number of significant nodes. In the figure, we replace 5 nodes of weight 1 for 2 nodes of weights 3 and 2. The effective size of the subgraph A through F is 3. In Section 5.6, we show that this refinement is very effective in real networks, and *we can usually discover large solutions by setting* $k \leq 10$.

We extend this idea to an *approximate refinement*. For a parameter $\beta \in [0,1]$, we keep merging nodes as long as the number of significant nodes remains more than β times the size of the supernode. Figure 3b shows an example for $\beta = 5/6$. By allowing non-significant node D to be merged, we are able to combine nodes A through F in a subgraph of effective size 1 and weight 5. Note that when $\beta < 1$, the solution may not be optimal,

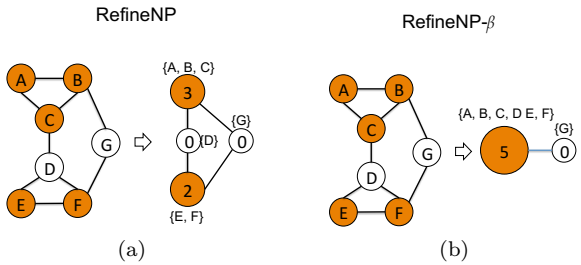

Figure 3: (a) **Graph refinement.** Colored nodes are significant. Nodes A, B, and C are merged into a supernode of weight 3, and E and F form a supernode of weight 2. The effective size of the set $\{A,B,C,D,E,F\}$ is 3. (b) **Graph refinement with** $\beta = 5/6$. By allowing non-significant node D to be merged, the effective size of $\{A,B,C,D,E,F\}$ becomes 1 with an approximation guarantee bounded by β.

but Lemma 9.5 in the supplementary material describes the effect on the approximation bound.

Low radius subgraphs. Let $B_G(v,r)$ (referred to the ball of radius r at v) denote the set of nodes at distance at most r from v in the graph G. It suffices to run the algorithm restricted to the balls centered around significant nodes. The balls are smaller than the full graph, so they can be processed faster; furthermore, they can be processed in parallel.

Remark. We use these two techniques to reduce the size of the input graph before running MaxWeight. Our algorithm, FastColCodeNP, with these addition is shown in the supplementary material.

4 Algorithms for Parametric Scan Statistics and Extensions

In parametric scan statistics, each node v of the input graph has two weights associated with it: $c(v)$ and $b(v)$, which makes the problem more challenging than for non-parametric functions. Furthermore, there exist other score functions for graph anomaly detection where both nodes and edges have weights [4, 18]. Optimizing such functions reduces to the Prize Collecting Steiner Tree (PCST) problem [10], which is NP-Hard. We can extend the methods described above to these settings by keeping additional information in the dynamic program. We propose algorithm FastColCodeP for parametric scan statistics and algorithm ColCodeNW for PCST. Details of these algorithms may be found in the supplementary material, but we show experimental results for both in the next section.

5 Experiments

Our experiments address the following questions.

1. Optimization power. Do our algorithms find high-scoring subgraphs in real networks and synthetic benchmarks? How do they compare with existing methods?

(Section 5.3)
2. Event detection power. Do our algorithms correctly identify anomalous subgraphs? How do precision and recall compare with baselines?(Section 5.4)
3. Scalability. How do our algorithms scale to networks with more than 10^5 nodes? (Section 5.5)
4. Performance in real datasets. How does the performance in real datasets compare with the worst case bounds? (Section 5.6).

For brevity, we focus on one scan statistic from each class as illustrative examples: (1) Berk-Jones (BJ) statistic [3] with $\alpha_{\max} = 0.15$, (2) positively-elevated mean statistic (EMS) [17], and (3) the Heaviest Subgraph (HS) [4] and EventTree+ [18] functions, as examples of non-parametric, parametric, and generalized functions with edge weights, respectively.

5.1 Datasets. We use datasets from different domains, including social networks, infrastructure networks, and standard synthetic benchmarks. For these datasets, we have multiple instances, corresponding to snapshots of the networks at different times, and we have data of events in each snapshot. Additionally, we use datasets with planted anomalies for our scalability experiments. A summary of the datasets is provided in Table 4. See the supplementary material for more details.

Table 4: Datasets used in our experiments

Dataset	Description	Nodes	Edges	Instances
Datasets with real events				
CitHepPh	Citation network	11,895	76,284	4
NEast	Network of counties in Northeastern USA	245	683	10,000
Traffic	Traffic Network of Los Angeles Country, CA	1,870	1,993	1,488
Twitter	Follower network collected through Twitter API	2,645	17,108	182
BWSN	Battle of the Water Sensors	12,527	14,831	22
PCST	Benchmark for the Prize Collecting Steiner Tree Problem	100 to 400	284 to 1,576	34
Datasets with planted anomalies for scalability experiments				
Email-EuAll	Email Network	224,832	340,795	1
Higgs-Retweet	Retweet Network	223,833	307,884	1
RoadNet-PA	Traffic Network of Pennsylvania	1,088,092	1,541,898	1
Random	Erdos-Renyi graphs of with 100 to 1,000,000 nodes	100 to $1,000,000$	$O(100)$ to $O(1,000,000)$	5

5.2 Baseline Methods. We compare our proposed algorithm with 6 state-of-the-art methods that are organized in three categories:
(1) `NPHGS` [5]: A local search heuristic for optimizing the BJ scan statistic. (2) `AdditiveGraphScan` [23] and `DepthFirstScan` [22]: The state-of-the-art algorithms for optimizing parametric scan statistics that satisfy the Linear-Time-Subset-Scanning (LTSS) property. The EMS statistic also belongs to this category.
(3) `GraphLaplacian` [21] and `EdgeLasso` [20]: The representative methods for anomalous subgraph detection that optimize their own specific score functions, but are often considered as baseline methods in connected subgraph detection papers [17].
(4) `EventTree+` [18] and `MEDEN` [4]: For optimizing anomaly score functions with node and edge weights.
Parameter Tuning. The methods under evaluation, including ours, depend on user-specified parameters. We set k to 10 or below for our algorithms. We discuss how we calibrate other parameters in the supplementary material.

5.3 Optimization Power

5.3.1 Non-Parametric Scan Statistics. We compare FastColCodeNP to other algorithms on the BJ statistic. In Table 1, we report the average BJ score obtained by each method, where the average is taken over all the instances in each dataset. We observe that FastColCodeNP achieves higher scores than all other methods. The difference in score is more pronounced in the NEast dataset, where FastColCodeNP more than doubles the score of `EdgeLasso` (EL) and `GraphLaplacian` (GL). We also note that `AdditiveGraphScan` has performance close to our algorithm, which is reasonable, since this method uses a sophisticated heuristic for Steiner connectivity problems.

5.3.2 Parametric Scan Statistics. Next, we compare FastColCodeP to other methods with respect to the EMS function. Table 5 shows the average score for different datasets. We find that FastColCodeP has the best performance in all datasets, except for NEast, where `AdditiveGraphScan` (GS) scores sligthly higher.

Table 5: **Parametric scan statistics optimization.** We evaluate different methods with respect to the EMS. In almost all datasets, FastColCodeP has better performace than existing methods.

	Elevated Mean Scan Statistic			
	FastColCodeP	GS	EL	GL
CitHepPh	**43.611**	8.578	14.959	41.830
NEast	41.903	**42.164**	5.570	7.607
Traffic	**11.763**	9.920	4.526	8.752
Twitter	**23.019**	11.337	22.660	19.110
BWSN	**109.097**	21.64	108.933	107.459

5.3.3 Functions with Node and Edge Weights. We also test our algorithm on two objective functions for event detection that consider edge weights in addition to node weights: the Heaviest Subgraph (HS) [4] and `EventTree+` [18] problems. The methods proposed in these two works are `MEDEN` [4] and `GreedyT` [18], respectively. Both problem formulations reduce to the Prize Collecting Steiner Tree (PCST) problem [10]. We use our framework to design an algorithm for the PCST objective; we call this algorithm ColCodeNW, and

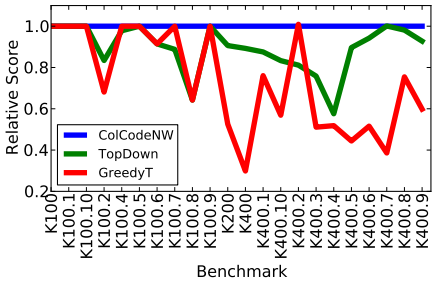

Figure 4: PCST objective score in a set of hard PCST benchmarks. ColCodeNW finds solutions of higher quality than state-of-the-art heuristics.

we compare it in terms of objective score to `MEDEN` and `GreedyT` on the PCST benchmark of [10]. For `MEDEN`, we convert the PCST instances to HS instances and run the TopDown heuristic described in [4]. For `GreedyT`, we convert the instances to a complete graph where an edge between two nodes has weight equal to the shortest path between the nodes, as described in [18]. Figure 4 shows the scores of the two heuristics relative to the score of ColCodeNW. Our algorithm finds subgraphs of higher quality than the heuristics. The score is as much as 4 times higher compared to `GreedyT` and 1.5 times higher compared to `MEDEN`.

5.4 Event Detection Power. Now, we evaluate FastColCodeNP in terms of event detection power. We use the ground truth provided with the BWSN dataset, and we evaluate in terms of accuracy, precision, recall, and the F1 score. Let R be the set of nodes in the anomalous subgraphs and let S be the detected subgraph; then, we define

(1) $\text{Accuracy}(R,S) = \frac{|R \cap S|}{|R \cup S|}$,

(2) $\text{Precision}(R,S) = \frac{|R \cap S|}{|S|}$,

(3) $\text{Recall}(R,S) = \frac{|R \cap S|}{|R|}$, and

(4) $\text{F1 score} = 2\left(\frac{\text{Precision}_{(R,S)} \cdot \text{Recall}_{(R,S)}}{\text{Precision}_{(R,S)} + \text{Recall}_{(R,S)}}\right)$.

In order to assess the performance of our method under noise, we introduce a random percentage of uniform noise in each instance. For a given *noise level* l, each non-significant node becomes significant with probability l.

In Table 6, we compare the performance of FastColCodeNP with other algorithms at different noise levels. As in the previous section, we observe that our algorithm achieves higher objective scores compared to other methods, even when noise is present. As for the event detection power, FastColCodeNP has higher accuracy and recall for all the noise levels and higher F1 score at almost every level. Finally, we note that the results in this section provide evidence that *better objective scores also lead to better detection power, thus the importance of algorithms with good theoretical bounds.*

5.5 Scalability. With the techniques presented in Section 3.4, we are able to run FastColCodeNP in networks with over one million nodes. We note that in previous work only networks of up to $80,000$ nodes have been considered.. In Table 7, we compare our algorithm to existing methods in terms of objective score and running time. First, we compare FastColCodeNP to the best-performing heuristic: `AdditiveGraphScan` (GS). We note that both methods achieve the same score in all datasets, but the running time of the latter is one order of magnitude larger, and it didn't complete after 10 hours for the road network. Second, we observe that `GraphLaplacian` (GL) does not scale to large networks due to the fact high time and space complexity of constrained quadratic programming methods. We also note that our algorithm is much faster on the road network than on the other two because nodes in this planar network have low degree, thus making our low-radius technique more effective. Finally, `EdgeLasso`, `DFS`, and `NPHGS` are faster than FastColCodeNP; however, the scores obtained are significantly lower than using our algorithm.

In Figure 1, we show the scalability of all the algorithms we consider as a function of graph size in the Random dataset. FastColCodeNP is faster than other methods for graph of size 10^4 and above, and it was the only algorithm to run to completion on graphs with one million nodes within 24 hours.

5.6 Performance guarantees in real datasets.
Graph refinement. Our graph refinement operation reduces the number of significant nodes in real datasets to less than a third of the original number. In Table 8, we report the number of significant nodes before and after graph refinement for $\beta \in \{1, 0.95, 0.90\}$. Each dataset initially contains hundreds of significant nodes, with Twitter being close to $1,000$. However, after graph refinement, this number goes down to less than 100. With approximate refinement, we are able to further reduce the effective number of significant nodes, down to a single digit in most cases. Because there are only a few significant supernodes, solutions of high score are small. In fact, if we set k to 10 or less in FastColCodeNP, we are able to discover solutions of higher scores than previous methods (Section 5.3).
Convergence in few iterations. Algorithm FastColCodeNP uses $\ell = e^k \log n^2/\epsilon$ random colorings to guarantee a solution with probability $1 - \epsilon$. In practice, we find the number of colorings needed is much smaller—this is shown in Figure 5 for the PCST benchmark. Each line in the plot represents the solution obtained for one instance of size 100; the y-axis shows the objective value obtained normalized by the objec-

Table 6: Average precision, recall, F1 score, accuracy, and objective value at different levels of noise.

	Precision				Recall				F1 Score				Accuracy				$F(S)$			
	GS	EL	GL	FastColCodeNP	GS	EL	GL	FastColCodeNP	GS	EL	GL	FastColCodeNP	GS	EL	GL	FastColCodeNP	GS	EL	GL	FastColCodeNP
0%	**0.980**	0.999	0.901	0.977	0.943	0.856	0.856	**0.955**	0.948	0.895	0.820	**0.952**	0.966	0.855	0.820	**0.973**	599.972	530.850	530.457	**602.164**
2%	**0.974**	0.991	0.995	0.973	0.967	0.796	0.772	**0.975**	**0.970**	0.854	0.842	0.957	0.946	0.789	0.769	**0.950**	579.197	427.984	437.783	**580.977**
4%	0.945	0.985	0.984	**0.966**	0.955	0.687	0.663	**0.971**	0.952	0.775	0.757	**0.963**	0.912	0.678	0.652	**0.929**	565.363	393.930	387.914	**571.231**
6%	**0.959**	0.964	0.973	0.954	0.937	0.567	0.542	**0.953**	0.946	0.683	0.664	**0.953**	0.901	0.558	0.536	**0.912**	522.694	318.593	300.118	**531.497**
8%	0.928	0.960	0.966	**0.931**	0.888	0.561	0.502	**0.919**	0.905	0.670	0.626	**0.923**	0.830	0.544	0.490	**0.860**	483.127	315.720	291.227	**491.657**

Table 7: Performace-runtime tradeoff in large datasets for non-parametric scan statistic evaluation.

	Berk-Jones Scan Statistic (Time in seconds)					
	FastColCodeNP	GS	EL	GL	DFS	NPHGS
Email-EuAll	**420.46** (2,376)	420.46 (20,254)	275.08 (679)	-	392.53 (3,671)	275.08 (10)
Higgs-Retweet	**839.18** (1,015)	839.18 (32,340)	421.16 (585)	-	721.70 (3,213)	421.16 (5)
RoadNet-PA	**24.66** (22)	-	24.66 (7,919)	-	21.22 (1,584)	13.28 (15)

Table 8: Number of significant nodes with graph refinement.

Dataset	$W(S,\alpha)$ ($\alpha = 0.15$)	**Graph Refinement** $\beta = 1$	$\beta = 0.95$	$\beta = 0.90$
CitHepPh	624	20	3	1
NEast	65	24	24	21
Traffic	157	61	61	60
Twitter	991	80	2	1
BWSN	333	4	2	2

tive obtained in the last iteration of the algorithm. The theoretical bound requires $3,285$ random colorings to guarantee 95% probability. However, the best solution is found in less than 20 iterations for all instances, and sooner for most of them. We observed similar results in the other networks in Table 4.

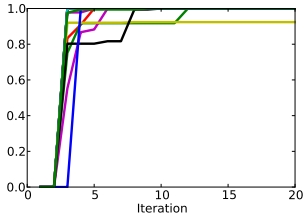

Figure 5: Score of the solution found (normalized by that of the optimum) as a function of the number of iterations.

6 Application

We use our methods for event detection in Twitter follower graphs of Mexico and Venezuela. We model interactions —i.e. retweets and replies—between users as Poisson counts. The weight of an edge is given by $-\log(p(n_e^t|n_e^{[1,t-1]})/\mu$, where $p(n_e^t|n_e^{[1,t-1]})$ is the posterior probability of seeing n_e^t interactions given the counts in the previous days, and $\mu = 0.05$ is a significance threshold. This weighing function [14] is positive-increasing if the posterior probability is less than μ and negative-decreasing otherwise. After assigning weights to edges, we solve the Heaviest Dynamic Subgraph problem [4] using algorithm ColCodeNW (See Section 5.3.3). For Venezuela, the temporal anomalous subgraph spans the time period between January 4, 2014 and March 31, 2014, which was a time of nationwide protests against the central government of that country. We also extracted the tweets corresponding to the heavy subgraphs, and we find that these tweets contain chatter about important national events.Figure 6 shows one such event. The predominant terms of the tweets relate to a protest organized in the city of Tachira demanding the liberation of local students, who had been put in jail in the previous days for protesting. In the supplementary material, we show results for the Twitter follower network of Mexico.

(a) (b)

Figure 6: Examples of events found by solving the HDS formulation in Twitter networks using our methods.

7 Related Work

There is a very large body of work related to our paper because of the wide range of applications. For brevity, we only discuss some of the specific papers that deal with scan statistics in Table 9, and give more detailed comparison in the online supplementary material. We also refer to the comprehensive survey by Akoglu et al. [1] for general discussion on anomaly detection. As shown in Table 9, our method is the first to give *rigorous guarantees* for a large class of scan statistics.

8 Conclusions

We present a unified framework for optimizing a broad class of graph scan statistics with connectivity constraints, with the following novel characteristics: (1) it gives rigorous guarantees for a large class of parametric and non-parametric score functions, (2) it can be scaled to large graphs with over a million nodes, and Our methods will lead to direct improvements in performance quality for other uses of graph scan statistics in different applications.

Acknowledgements. The work of Jose Cadena and Anil Vullikanti has been partially supported by the following grants: DTRA CNIMS Contract HDTRA1-11-D-0016-0010, NSF BIG DATA Grant IIS-1633028 and NSF DIBBS Grant ACI-1443054.

References

Table 9: Summary of the related work and how it compares to this paper.

Algorithms based on optimization of parametric scan statistics in networks		
Exact algorithms	Exhaustive search over connected subgraphs [24], branch-and-bound method methods for Kulldorff's and `DepthFirstScan` for upper level set scan statistic [22]	Do not scale to graphs with more than 1000 nodes
Simulated annealing	Based on a concept of "non-compactness" for penalizing clusters [7]	Exponential time, No guarantee
`AdditiveGraphScan`	Connects clusters based on shortest path distances [23], used for nonlinear score functions	$O(mn+n^2 \log n)$ time, no quality guarantees
`EdgeLasso`	Sparse learning method based on edge-lasso regularization [20], handles quadratic score functions	$O(l \cdot n^3)$ time, no guarantees
`GraphLaplacian`	Spectral scan method based on graph Laplacian regularization [21], handles quadratic score functions	$O(l \cdot n^3)$ time, no guarantees
Algorithms based on optimization of nonparametric scan statistics		
`NPHGS`	Heuristic method for optimizing nonparametric scan statistics on general graphs [5], considers nonlinear functions	$O(n \log n)$ time, no quality guarantees
Reduction to variants of Network Design and using methods for Prize-Collecting Steiner Tree (PCST)		
`EventTree+`	Use PCST method [18], linear function	$O(n^2 \log n)$, 2-approximation for PCST from [18]
`Meden`	Greedy algorithm for the Heaviest Dynamic Subgraph (HDS) problem, equivalent to the NetWorth objective, which is a linear function [4]	$O(n^2 \log n)$ time, no quality guarantees
Our algorithm: gives optimal solution for a large number of parametric and non-parametric scan statistics with effective solution size parameter τ, with probability at least $(1-\epsilon)$, and runs in time $O((2e)^\tau \cdot m \log \frac{n}{\epsilon})$		

[1] L. Akoglu, H. Tong, and D. Koutra. Graph based anomaly detection and description: a survey. *Data Mining and Knowledge Discovery*, 2014.

[2] N. Alon, R. Yuster, and U. Zwick. Color-coding. *Journal of the ACM (JACM)*, 1995.

[3] R. H. Berk and D. H. Jones. Goodness-of-fit test statistics that dominate the kolmogorov statistics. *Z. Wahrsch. Verw. Gebiete*, 1979.

[4] P. Bogdanov, M. Mongiovì, and A. Singh. Mining heavy subgraphs in time-evolving networks. In *ICDM*, 2011.

[5] F. Chen and D. Neill. Non-parametric scan statistics for event detection and forecasting in heterogeneous social media graphs. In *KDD*, 2014.

[6] D. Donoho and J. Jin. Higher criticism for large-scale inference, especially for rare and weak effects. *Statist. Sci.*, 30(1), 2015.

[7] L. Duczmal, M. Kulldorff, and L. Huang. Evaluation of spatial scan statistics for irregularly shaped clusters. *Journal of Computational and Graphical Statistics*, 2006.

[8] T. Hansen and F. Vandin. Finding mutated subnetworks associated with survival in cancer. *arXiv preprint arXiv:1604.02467*, 2016.

[9] S. Iyengar et al. The genetic basis of complex traits: rare variants or "common gene, common disease"? *Methods Mol Biol.*, 2007.

[10] D. Johnson, M. Minkoff, and S. Phillips. The prize collecting steiner tree problem: Theory and practice. In *ACM SODA*, 2000.

[11] M. Kulldorff. A spatial scan statistic. *Communications in Statistics: Theory and Methods*, 1997.

[12] M. Leiserson et al. Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. *Nature genetics*, 47(2):106–114, 2015.

[13] E. McFowland, S. Speakman, and D. B. Neill. Fast generalized subset scan for anomalous pattern detection. *JMLR*, 14(1), 2013.

[14] M. Mongiovì, P. Bogdanov, R. Ranca, A. Singh, E. Papalexakis, and C. Faloutsos. Netspot: Spotting significant anomalous regions on dynamic networks. In *SDM*, 2013.

[15] D. B. Neill. An empirical comparison of spatial scan statistics for outbreak detection. *International Journal of Health Geographics*, 2009.

[16] D. B. Neill. Fast subset scan for spatial pattern detection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2012.

[17] J. Qian, V. Saligrama, and Y. Chen. Connected subgraph detection. In *AISTATS*, 2014.

[18] P. Rozenshtein, A. Anagnostopoulos, A. Gionis, and N. Tatti. Event detection in activity networks. In *KDD*, 2014.

[19] J. Sharpnack, A. Krishnamurthy, and A. Singh. Near-optimal anomaly detection in graphs using lovasz extended scan statistic. In *NIPS*, 2013.

[20] J. Sharpnack, A. Singh, and A. Rinaldo. Sparsistency of the edge lasso over graphs. In *AISTATS*, 2012.

[21] J. Sharpnack, A. Singh, and A. Rinaldo. Changepoint detection over graphs with the spectral scan statistic. In *AISTATS*, 2013.

[22] S. Speakman et al. Scalable detection of anomalous patterns with connectivity constraints. *Jl Comp Graphical Stat*, 2015.

[23] S. Speakman, Y. Zhang, and D. B. Neill. Dynamic pattern detection with temporal consistency and connectivity constraints. In *ICDM*, 2013.

[24] K. Takahashi, M. Kulldorff, T. Tango, and K. Yih. A flexibly shaped space-time scan statistic for disease out-

break detection and monitoring. *International Journal of Health Geographics*, 2008.

9 Supplementary Material

We present proof details, pseudocode, and extensions of the methods presented in Section 3. For brevity, some proofs have been omitted, but they can be found in an online extended version of this paper [**?**], along with the code and datasets used in the experiments.

9.1 Additional Details for Section 2

9.1.1 Scan Statistics Functions. Table 10 lists examples of parametric and non-parametric scan statistics that can be optimized using our framework. We list more functions that fall under our framework in the online version.

9.1.2 Hardness Because of connectivity constraints, optimizing scan statistics on graphs is challenging in general. Note that this contrasts with the case without any connectivity requirement, where the optimal value of the scan statistic can be computed in polynomial time because of a linear ordering property [16].

LEMMA 9.1. *Given an instance $(G = (V, E), \alpha_{max})$ of non-parametric scan statistics, finding a connected set $S \subseteq V$ that maximizes the score $F(S)$ is NP-complete.*

LEMMA 9.2. *Given an instance $(G = (V, E), b(\cdot), c(\cdot))$ of parametric scan statistics, finding a connected set $S \subseteq V$ that maximizes the score $F(S)$ is NP-complete.*

9.2 Additional details for Section 3

Proof. (of Lemma 3.2) Suppose $M(v, T)$ is achieved for a connected set S, such that $|S| = |T|$ and $\{col(u) : u \in S\} = T$, with $M(v, T) = \sum_{i \in S} w(i, \alpha)$. We claim that there exists $u \in Nbr(v)$, and partitions $T = T_1 \cup T_2$, and $S = S_1 \cup S_2$, such that: (1) $M(v, T) = M(v, T_1) + M(u, T_2)$, (2) $\{col(i) : i \in S_1\} = T_1$ and $\{col(i) : i \in S_2\} = T_2$, and (3) the subsets of nodes S_1 and S_2 are connected. Since S is connected, there exists a tree H that spans S and contains node v. Further, there must exist a node $u \in Nbr(v)$ such that $(u, v) \in T$, since $|T| = |H| \geq 2$. Let H_1 and H_2 be the trees rooted at nodes v and u, respectively, that result when edge (u, v) is deleted in H. Let S_1 and S_2 denote the sets of nodes in H_1 and H_2, respectively. Let T_1 and T_2 be the colors used by S_1 and S_2, respectively. By construction, we have $M(v, T) = M(v, T_1) + M(u, T_2)$, so that the partitions $S = S_1 \cup S_2$ and $T = T_1 \cup T_2$ satisfy the requirements mentioned earlier. Therefore, the recurrence follows.

9.2.1 Scaling We take advantage of a *locality* property of non-parametric scan statistics. Namely, if we are given a set S with score $F(S)$, we can increase the score of the set by adding any significant node that is a neighbor of a node already in S.

LEMMA 9.3. *Let $G(V, E)$ be a network, and let $F(\cdot)$ be a non-parametric scan statistic function. Given a set $S \subseteq V$, suppose there exists a significant node $u \notin S$ and an edge $(u, v) \in E$, for some $v \in S$. Then, $F(S \cup \{u\}) \geq F(S)$.*

Proof. Non-parametric scan statistics are increasing on $\frac{W(S)}{N(S)}$. Since u is significant, we have that

$$\frac{W(S \cup \{u\})}{N(S \cup \{u\})} = \frac{W(S) + 1}{N(S) + 1} \geq \frac{W(S)}{N(S)},$$

and the proof follows.

Exact refinement. An implication of Lemma 9.3 is that we can collapse components of significant nodes into a single node prior to running ColCodeNP. We propose a *graph refinement* to reduce the total number of significant nodes in a graph. Given a network $G(V, E)$, p-values for the nodes in V, and a significance level α, we define $V_1, \ldots, V_r$ as the connected components (or *supernodes*) of G induced by the set of significant nodes. We create a new graph $H(V', E')$, whose node set consists of $V_1, \ldots, V_r$ and the non-significant nodes in G, $V \setminus \bigcup_{i=1}^{r} V_i$. Edges between non-significant nodes are preserved in H, and we put an edge from a non-significant node u to V_i if the edge (u, v) exists, for some $v \in V_i$. Finally, we create a vector of weights, $\mathbf{w}'$. The weight of a node in H is $|V_i|$ for each supernode V_i, and 0 otherwise. This procedure is equivalent to removing all the significant nodes and replacing them with the respective supernode V_i. Figure 3a in the main text illustrates the procedure. After this preprocessing step, we run procedure MaxWeight for the instance $(H(V', E'), \mathbf{w}')$.

LEMMA 9.4. *Let $((G(V, E), \alpha), k, \epsilon)$ be an input to* ColCodeNP*, and let S^*_G be the solution returned by the algorithm. Similarly, let $((H(V', E'), \alpha), k, \epsilon)$ with weights $\mathbf{w}'$ be the corresponding instance generated by graph refinement, and let S^*_H be the solution returned by* ColCodeNP *in this instance. Then, $F(S^*_H) = F(S^*_G)$. Furthermore, suppose $|S^*_G| = k$; then, $|S^*_H| = k'$, for some $k' \leq k$, so it is possible to execute* ColCodeNP *with parameter k' and still obtain $F(S^*_H) = F(S^*_G)$.*

Proof. The lemma follows from Lemma 9.3.

Approximate refinement. It is possible to further compress the graph by including non-significant nodes into the components described above. As before, we first obtain connected components of significant nodes, $V_1, \ldots, V_r$, but now, we keep adding nodes to a component V_i as long as the number of significant nodes is at least $\beta|V_i|$, where β is a parameter between 0 and 1.

Table 10: Scan Statistics Functions that can be optimized with our framework.

Non-Parametric Scan Statistics (The following definitions are by default, unless otherwise indicated) $F(S) = \max_{\alpha \le \alpha_{max}} \phi(W(S,\alpha), N(S), \alpha)$, $p(v)$ refers to the p-value of node v, $N(S) = \lvert S \rvert, W(S,\alpha) = \sum_{v \in S} I(p(v) \le \alpha)$, where $I(\text{True}) = 1$ and $I(\text{False}) = 1$.		
Name	**Original Form**	**General Form**
Berk-Jones [3]	$F(S) = \max_{\alpha \le \alpha_{max}} N(S) KL(\frac{W(S,\alpha)}{N(S)}, \alpha)$	$\phi(a,b,\alpha) = b \cdot KL(a/b, \alpha)$, where $KL(x,\alpha) = x \log\left(\frac{x}{\alpha}\right) + (1-x)\log\left(\frac{1-x}{1-\alpha}\right)$
Higher Criticism [?]	$F(S) = \max_{\alpha \le \alpha_{max}} \frac{W(S,\alpha) - N(S)\alpha}{\sqrt{N(S)\alpha(1-\alpha)}}$	$\phi(a,b,\alpha) = (a - b\cdot\alpha)/\sqrt{b \cdot \alpha(1-\alpha)}$
Kolmogorov-Smirnov [?]	$F(S) = \max_{\alpha \le \alpha_{max}} \sqrt{N(S)} \cdot \left(\frac{W(S,\alpha)}{N(S)} - \alpha\right)$	$\phi(a,b,\alpha) = \sqrt{b}\left(\frac{a}{b} - \alpha\right)$
Anderson-Darling [?]	$F(S) = \max_{\alpha \le \alpha_{max}} \sqrt{N(S)} \cdot \left(\frac{W(S,\alpha)}{N(S)} - \alpha\right) / \sqrt{\frac{W(S,\alpha)}{N(S)} \cdot \left(1 - \frac{W(S,\alpha)}{N(S)}\right)}$	$\phi(a,b,\alpha) = \sqrt{b}\left(\frac{a}{b} - \alpha\right) / \sqrt{\frac{a}{b} \cdot \left(1 - \frac{a}{b}\right)}$
Stochastic Ordering of p-Values [?]	$F(S) = N(S) \int_0^{\alpha_{max}} \frac{(W(S,\alpha)/N(S) - \alpha)^2}{\alpha(1-\alpha)} \mathrm{d}\alpha$	$\phi(a,b,\alpha) = b \int_0^{\alpha_{max}} \frac{(a/b - \alpha)^2}{\alpha(1-\alpha)} \mathrm{d}\alpha$
Fisher's Test [?]	$F(S) = -\sum_{v \in S} \log p(v)/N(S)$	$W(S,\alpha) = \sum_{v \in S} \log p(v)$, $\phi(a,b,\alpha) = -a/b$
Truncated Fisher's Test	$F(S) = \max_{\alpha \le \alpha_{max}} -\frac{\sum_{v \in S} I(p(v) \le \alpha) \log p(v)}{N(S)}$	$W(S,\alpha) = \sum_{v \in S} I(p(v) \le \alpha) \log p(v)$, $\phi(a,b,\alpha) = -a/b$
Stouffer's Test [?]	$F(S) = -\frac{\sum_{v \in S} \Phi^{-1}(1-p(v))}{\sqrt{N(S)}}$	$W(S,\alpha) = \sum_{v \in S} \Phi^{-1}(1-p(v))$, $\phi(a,b,\alpha) = -a/b$, where $\Phi^{-1}(\cdot)$ refers to the inverse cumulative density function of standard Gaussian distribution
Parametric Scan Statistics (The following defintions are by default, unless otherwise indicated) $F(S) = g(C(S), B(S))$, $C(S) = \sum_{v \in S} c(v)$, $B(S) = \sum_{v \in S} b(v)$		
Positive Elevated Mean Scan Statistic [17]	$F(S) = \sum_{i \in S} x_i / \sqrt{N(S)}$	$g(a,b) = a/\sqrt{b}$
Elevated Mean Scan Statistic [17]	$F(S) = (\sum_{i \in S} x_i)^2 / N(S)$	$g(a,b) = a^2/b$
Expectation-based Poisson Scan Statistic [16]	$F(S) = C(S)\log(C(S)/B(S)) + B(S) - C(S)$	$g(a,b) = a\log(a/b) + b - a$
Kulldorff Scan Statistic [11]	$F(S) = C(S)\log\left(\frac{C(S)}{B(S)}\right) + (C - C(S))\log\left(\frac{C - C(S)}{B - B(S)}\right) - C\log\left(\frac{C}{B}\right)$, where $C = \sum_{v \in \mathbb{V}} c(v)$ and $B = \sum_{v \in \mathbb{V}} b(v)$.	$g(a,b) = a\log(\frac{a}{b}) + (C-a)\log(\frac{C-a}{B-b}) - C\log\left(\frac{C}{B}\right)$
Expectation-based Gaussian Scan Statistic [16]	$F(S) = (C(S) - B(S))^2/(2B(S))$, where $\sigma(v)$ refers to the standard deviation of $c(v)$ that is calibrated based on its historical observations, $C(S) = \sum_{v \in S}(c(v)b(v))/\sigma(v)^2$, and $B(S) = \sum_{v \in S} b(v)/\sigma(v)^2$	$g(a,b) = (a-b)^2/(2b)$

We show an example in Figure 3b. By doing this, we are able to reduce the number of anomalous supernodes. We may not find the optimal solution now; however, we can control the error with the parameter β.

LEMMA 9.5. *Denote the number of nodes signicant at level α in a set S as $N_\alpha(S)$. Let S^* be the set that maximizes F and let $r(S^*) = \frac{N_\alpha(S^*)}{N(S^*)}$. There is a solution on the instance H with ratio $r(S) \ge \beta r(S^*)$.*

Proof. We split the set S^* into significant nodes, $N_\alpha(S^*)$, and non-significant nodes, $N_\alpha^-(S^*)$, such that $N(S) = N_\alpha(S^*) + N_\alpha^-(S^*)$. We now show that, in the instance H, there exists a set S with ratio

$$r(S) = \frac{N_\alpha(S)}{N(S)} \ge \frac{N_\alpha(S^*)}{N_\alpha(S^*) + N_\alpha^-(S^*) + \frac{1-\beta}{\beta} N_\alpha(S^*)}.$$

We define S' as the set formed by the supernodes V_i corresponding to the significant nodes in S^*, and we note that $N_\alpha(S') \ge N_\alpha(S^*)$; for simplicity, we assume equality. By construction, the cardinality of S' is $N(S') = N_\alpha(S') + \frac{1-\beta}{\beta} N_\alpha(S') = N_\alpha(S^*) + \frac{1-\beta}{\beta} N_\alpha(S^*)$. Note that the nodes in S' may be disconnected; however, we can connect them using a set of anomalous nodes S'' of size at most $N_\alpha^-(S^*)$. Finally, we form a set $S = S' \cup S''$ that has the desired ratio.

To conclude the proof, we compute $r(S)/r(S^*)$:

$$\frac{r(S)}{r(S^*)} = \frac{\frac{N_\alpha(S)}{N(S)}}{\frac{N_\alpha(S^*)}{N(S^*)}} \ge \frac{\frac{N_\alpha(S^*)}{N(S^*) + \frac{1-\beta}{\beta} N_\alpha(S^*)}}{\frac{N_\alpha(S^*)}{N(S^*)}} = \frac{1}{1 + \frac{1-\beta}{\beta} \times \frac{N_\alpha(S^*)}{N(S^*)}}.$$

Noticing that $\frac{N_\alpha(S^*)}{N(S^*)} \le 1$, we obtain $r(S) \ge \beta r(S^*)$.

9.3 Extensions to Parametric Scan Statistics. In Algorithm 3, we describe COLCODEP for parametric scan statistics maximization. For these functions, each node v of the input graph has two weights associated with it: $C(v)$ and $B(v)$. Therefore, we need a more

Algorithm 2 FastColCodeNP$(G(V,E), \alpha_{max}, k, \epsilon, \beta)$.

Input: Instance $(G(V,E), \alpha_{max})$, parameters k, ϵ and β
Output: Set S^* with score $OPT(F,k)$
Let A be the set of p-values of nodes in V below α_{max}
for$\alpha \in A$
  Perform approximate refinement with parameter β.
  Let $H = (V', E')$ be the refined graph with weights $\mathbf{w}'$
  $\{S_i^*(\alpha) : i \in K\} = \textsc{MaxWeight}(H(V', E'), \mathbf{w}', k, \epsilon/n^2)$
$S^* = argmax_{i\in[1,k],\alpha\in A} F(S_i^*(\alpha))$
return S^*

general algorithm than ColCodeNP. Analogous to Lemma 3.1, we make use of the following property:

Lemma 9.6. *The parametric scan statistics functions characterized by equation (2.2) are increasing functions of $C(S)$ if $C(S) > B(S)$ and $B(S)$ is constant.*

Given a graph $G(V,E)$ and vectors $\mathbf{C}$ and $\mathbf{B}$, let $M(v,T,j)$ be the maximum value $C(S)$ over all connected subsets S, such that (1) $v \in S$, (2) S is colorful with respect to T, and (3) $B(S) = j$. Here j ranges from 1 to $B(V)$. $M(v,T,j)$ can be computed by a dynamic program with the following recurrence.

Lemma 9.7. *Let $M(v,T,j)$ be defined as above. For any node v and color s, $M(v,\{s\},j) = c(v)$ if $col(v) = s$ and $b(v) = j$, else $M(v,\{s\},j) = -\infty$. If $|T| \geq 2$:*

$$M(v,T,j) = \max_{\substack{u\in Nbr(v)\\ T_1,T_2\subseteq T\\ j_1+j_2=j}} \{M(v,T_1,j_1) + M(u,T_2,j_2)\}.$$

where the maximum is over all partitions $T_1 \cup T_2$ of the set T, all integers j_1, j_2 with $j_1 + j_2 = j$. and all neighbors u of v.

Proof. Analogous to Lemma 3.2.

9.3.1 ColCodeP Our algorithm for maximizing parametric scan statistics, ColCodeP, is presented in Algorithm 3. The procedure MaxWeightP uses Lemma 9.7 to compute $\psi_i = \max_{T:|T|=i} M(v,T,j)$ for all $i \leq k$.

Theorem 9.1. *Let $F(\cdot)$ be any of the parametric scan statistics in Table 10, and let $OPT(F,k) = \max_{S:|S|\leq k} F(S)$, where the maximum is over all connected subsets S of size $\leq k$.* ColCodeP *returns solution S^* satisfying $\Pr[F(S^*) = OPT(F,k)] \geq 1 - \epsilon$, in time $O(2^k e^k m B_{max} \log(n/\epsilon))$, and using space $O(2^k n B_{max})$, where $B_{max} = B(V)$.*

Proof. Analogous to Theorem 3.1.

Algorithm 3 ColCodeP$((G(V,E), \mathbf{C}, \mathbf{B}), k, \epsilon)$.

Input: Instance $(G(V,E), \mathbf{C}, \mathbf{B})$, parameters k and ϵ
Output: Set S^* with score $OPT(F,k)$
$\{S_i^* : i \in K\} = \textsc{MaxWeightP}(G(V,E), \mathbf{C}, \mathbf{B}, k, \epsilon/n)$
$S^* = argmax_{i\in[1,k]} F(S_i^*)$
return S^*

procedure MaxWeightP$(G(V,E), \mathbf{C}, \mathbf{B}, k, \epsilon')$
Input: Instance $(G(V,E), \mathbf{C}, \mathbf{B})$ and parameter k
Output: Set S_i^* with maximal weight ψ_i for all $i \in [1,k]$
Let $\psi_i = -\infty$ for all $i \in [1,k]$
for $t = 1$ to $e^k \log(1/\epsilon')$
  For each node v, pick random color $col(v) \in K$
  for $v \in V, s \in K, j \leq B(V)$
    $M(v,\{s\},j) = c(v)$ if $col(v) = s$ and $j = b(v)$; $-\infty$ otherwise
  for $v \in V$, $T \subseteq K$, with $|T| \geq 2$, $j \leq B(V)$
    Use Lemma 9.7 to compute $M(v,T,j)$
    If $M(v,T,j) > \psi_{|T|}$ **update** $\psi_{|T|} = M(v,T,j)$
return $\{S_i^* : \sum_{v\in S_i^*} c(v) = \psi_i, \text{for } i \in K\}$

We note that by approximating $B(S)$ within a factor of $(1+\delta)$, the running time in Theorem 9.1 can be improved to $O(2^k e^k m \frac{n^2}{\delta} \log(n/\epsilon))$, while losing a constant factor in terms of the approximation. We define $\mu = \delta B_{max}/n$, for some $\delta > 0$. Then, we define a vector $\mathbf{B}'$, where $b'(v) = \lfloor b(v)/\mu \rfloor$, for each node v. By invoking ColCodeP on the instance $(G(V,E), \mathbf{C}', \mathbf{B}')$, we obtain a $(1+\delta)$ approximation on the weight of S^*.

9.3.2 Scaling There is a notion of graph refinement for parametric scan statistics analogous to the one presented above. We note that the parametric functions in Table 10 are increasing on the ratio $r(S) = C(S)/B(S)$. Therefore, if we are given a set S with score $F(S)$, we can increase the score of the set by adding any node that is a neighbor of a node already in S as long as $r(S)$ does not decrease. This idea is formalized in the following lemma.

Lemma 9.8. *Let $G(V,E)$ be a network, and let $F(\cdot)$ be a parametric scan statistic function. Given a set $S \subseteq V$, suppose there exists an node $u \notin S$ and an edge $(u,v) \in E$, for some $v \in S$, such that $C(S\cup\{u\})/B(S\cup\{u\}) \geq C(S)/B(S)$; then, $F(S\cup\{u\}) \geq F(S)$.*

Exact refinement. For parametric scan statistics, the exact refinement consists of merging nodes into components as long as the ratio $r(S)$ of the component does not decrease. Given a network $G(V,E)$ and event $(c(v))$ and population $(b(v))$ counts for every $v \in V$, we maintain a list of components or supernodes $\mathcal{V} = V_1, \ldots, V_r$ and the graph $H(V', E')$ induced by those.

There is an edge between V_i and V_j if there exists nodes $v_i \in V_i$ and $v_j \in V_j$, such that v_i and v_j are neighbors in G. Initially, every node is its own component. The exact refinement iterates through the list of current components trying to merge them until no more merges are possible. In each iteration, we first sort the current components in descending order of ratio $r(S)$. Then, in that order, we merge a component with its neighbors if the ratio does not decrease. After this exact refinement, we run Algorithm 3 for the instance $(H(V', E'), \mathbf{C}', \mathbf{B}')$, where $\mathbf{C}'$ and $\mathbf{B}'$ are the event and population counts of the supernodes, respectively.

Approximate refinement. We can obtain larger components by allowing merges that decrease the ratio of the component. Our approximate refinement procedure takes two parameters: $\beta \in [0, 1]$ and $\delta > 1$. We allow a component V_i to grow as long as two constraints are not violated:

1. **Population size constraint.** The population size of V_i is at most δ times the smallest population size in the component: $B(V_i) \leq \delta \min_{v \in V_i} b(v)$.

2. **Event size constraint.** The event size of V_i is at least $\beta\delta$ times the largest event size in the component: $\frac{C(V_i)}{\delta} \geq \beta \max_{v \in V_i} c(v)$.

LEMMA 9.9. *Let S^* be the set that maximizes a parametric scan statistic G and let $r(S^*) = \frac{C(S^*)}{B(S^*)}$. There is a solution S on the instance H constructed as above with ratio $r(S) \geq \beta r(S^*)$.*

Proof. Analogous to Lemma 9.5.

9.3.3 Functions with Node and Edge Weights. Both the Heaviest Subgraph [4] and EVENTTREE+ problems [18] reduce to Prize Collecting Steiner Tree (PCST) with NetWorth objective [10]. In PCST, we are given a graph $G(V, E)$ with non-negative node prizes, $\boldsymbol{\pi}$, and non-negative edge costs, $\mathbf{w}$, and the goal is to find a tree $S(V(S), E(S))$ that maximizes the NetWorth objective:

$$W(S) = \sum_{v \in V(S)} \pi(v) - \sum_{e \in E(S)} w(e).$$

Using our framework, we can design an algorithm to find a tree with maximal NetWorth and size up to k, where k is a parameter. This implies an algorithm for HS and EVENTTREE+.

Let $M(v, T) = \max_S W(S)$, where the maximization is over all connected and colorful sets $S \subseteq V$, such that $v \in S$, $|S| = |T|$, and $\{col(u) : u \in S\} = T$. $M(v, T)$ can be computed by a dynamic program:

LEMMA 9.10. *Let $M(v, T)$ be defined as above. For any node v and color s, $M(v, \{s\}) = \pi(v)$ if $col(v) = s$, else $M(v, \{s\}) = -\infty$. If $|T| \geq 2$:*
$M(v, T) = \max_{\substack{u \in Nbr(v) \\ T_1, T_2 \subseteq T}} \{M(v, T_1) + M(u, T_2)\}$, *where the maximum is over all partitions $T_1 \cup T_2$ of the set T and all neighbors u of v.*

Proof. Analogous to Lemma 3.2.

9.4 Additional details for Section 5

9.4.1 Datasets. For the optimization event detection power experiments, we use the following datasets, for which we have temporal event data:

CitHepPh[2]. This is a network of scientific collaborations between authors of papers submitted to the High Energy Physics - Phenomenology category of arXiv. The p-value of each node v for a specific snapshot was calculated as the ratio of nodes in the current graph snapshot whose citations are greater than or equal to the citations of this node.

NEast [?]. The Northeastern USA Benchmark is a well-known dataset in the spatial scan statistics community. The benchmark contains census information of 245 counties as well as synthetically generated cases of a disease.

Traffic[3]. The highway network of Los Angeles County, California and its activity on May, 2014. Nodes in the graph are sensors that record traffic statistics, such as average speed and the number of vehicles passing through. We assume a normal distribution for the average speed recorded by each sensor. In each snapshot t, the p-value of a node v is the cumulative distribution function of a normal distribution with mean $x_v^{[1,t-1]}$ and standard deviation $\sigma_v^{[1,t-1]}$, where $x_v^{[1,t-1]}$ and $\sigma_v^{[1,t-1]}$ are, respectively, the sample mean and standard deviation for node v from snapshots 1 to $t-1$.

Twitter. A sample of the follower graph of Venezuela collected between July 1, 2013 and December 31, 2013. We assign p-values based on the tweeting behavior of the users. Formally, let x_u^t be the number of tweets generated by node u at time t; we model x_u^t as a draw from a Poisson distribution with parameter λ_u. We take a Bayesian approach and consider λ_u to be drawn from a Gamma distribution with parameters α_u and β_u. These parameters are updated as we see new data every snapshot. The p-value of a node at time t is its posterior probability $p(n_e^t | n_e^{[1,t-1]})$, which follows a negative binomial distribution by our choice of prior.

Battle of the Water Sensor Networks (BWSN) [?]. This dataset is a benchmark originally used to

[2] https://snap.stanford.edu/data/cit-HepPh.html
[3] http://pems.dot.ca.gov/

(a)

(b)

Figure 7: Examples of events found in the heavy subgraphs for Venezuela (top) and Mexico (bottom)

evaluate different sensor network designs in terms of early detection of contaminants in a water system. The dataset includes "ground truth" subgraphs representing parts of the network that are contaminated, which we use for evaluation in Section 5.4.

PCST [10]. A standard benchmark to evaluate algorithms for the Prize Collecting Steiner Tree Problem. We use the "K" instances of the benchmark to evaluate methods that reduce to PCST (Section 5.3.3).

For the scalability experiments (Section 5.5), we consider three large networks (i.e., over 10^5 nodes) from the SNAP repository [**?**]. Since we do not have data of events in these networks, we plant events according to the statistical assumptions of the BJ scan statistic. First, we assign p-values uniformly at random to all the nodes in the graph. Then, we select 5 seed nodes and their neighbors in the graph, and we assign a p-values less than 0.15 to these nodes. We also generated a set of Erdos-Renyi graphs (Random in Table 4) with number of nodes $n \in \{10^2, 10^3, 10^4, 10^5, 10^6\}$ and number of edges $m = O(n)$. We first assign random p-values to all the nodes. Then, we plant an anomaly by selecting a node and its ego network at random and setting their p-values below 0.15.

9.4.2 Parameter tuning for baseline methods. When possible, we use the values prescribed by the authors of the method; this is the case with `NPHGS`, `EventTree+`, and `MEDEN`. In the case of `GraphLaplacian` and `EdgeLasso`, we tune the parameters separately for each dataset. In particular, we take a sample of 20 instances for each dataset and choose a parameter from $\{0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1\}$ that maximizes the average score. Notice that the parameter for each dataset may be different.

9.4.3 Additional Details for Section 6. For Mexico, the two most common words in the extracted tweets form the phrase "energy reform" referring to a bill recently proposed by Mexican president Enrique Pena Nieto that would have a significant impact in the economy of the country (Figure 7).