Rapid Detection of Significant Spatial Clusters

Daniel B. Neill
Department of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
neill@cs.cmu.edu

Andrew W. Moore
Department of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
awm@cs.cmu.edu

ABSTRACT

Given an $N \times N$ grid of squares, where each square has a *count* c_{ij} and an underlying *population* p_{ij}, our goal is to find the rectangular region with the highest *density*, and to calculate its *significance* by randomization. An arbitrary *density function* D, dependent on a region's total count C and total population P, can be used. For example, if each count represents the number of disease cases occurring in that square, we can use Kulldorff's *spatial scan statistic* D_K to find the most significant spatial disease cluster. A naive approach to finding the maximum density region requires $O(N^4)$ time, and is generally computationally infeasible. We present a multiresolution algorithm which partitions the grid into overlapping regions using a novel *overlap-kd tree* data structure, bounds the maximum score of subregions contained in each region, and prunes regions which cannot contain the maximum density region. For sufficiently dense regions, this method finds the maximum density region in $O((N \log N)^2)$ time, in practice resulting in significant (20-2000x) speedups on both real and simulated datasets.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Apps- Data Mining

General Terms

Algorithms

Keywords

Cluster detection, spatial scan statistics, biosurveillance

1. INTRODUCTION

One of the core goals of data mining is to discover patterns and relationships in data. In many applications, however, it is important not only to discover patterns, but to distinguish those patterns that are *significant* from those that are likely to have occurred by chance. This is particularly important in epidemiological applications, where a rise in the number of disease cases in a region may or may not be indicative of an emerging epidemic. In order to decide whether further investigation is necessary, epidemiologists must know not only the location of a possible outbreak, but also some measure of the likelihood that an outbreak is occurring in that region. More generally, we are interested in spatial data mining problems where the goal is detection of *overdensities*: spatial regions with high scores according to some density measure. The density measure can be as simple as the count (e.g. number of disease cases, or units of cough medication sold) in a given area, or can adjust for quantities such as the underlying population. In addition to discovering these high-density regions, we must perform statistical testing in order to determine whether the regions are significant. As discussed above, a major application is in detecting clusters of disease cases, for purposes ranging from detection of bioterrorism (e.g. anthrax attacks) to identifying environmental risk factors for diseases such as childhood leukemia [8, 11, 6]. [5] discusses many other applications, including mining astronomical data (e.g. identifying star clusters), military reconnaissance, and medical imaging.

We consider the case in which data has been aggregated to a uniform, two-dimensional grid. Let G be an $N \times N$ grid of squares, where each square $s_{ij} \in G$ is associated with a *count* c_{ij} and an underlying *population* p_{ij}. For example, a square's count may be the number of disease cases in that geographical location in a given time period, while its population may be the total number of people "at-risk" for the disease. Our goal is to search over all rectangular regions $S \subseteq G$, and find the region S^* with the highest *density* according to a density measure D: $S^* = \arg\max_S D(S)$. We use the abbreviations *mdr* for the maximum density region S^*, and *mrd* for the maximum region density $D(S^*)$, throughout. We will also find the statistical significance (p-value) of this region by randomization testing, as described below.

The density $D(S)$ of a region S can be an arbitrary function of the total count of the region, $C(S) = \sum_S c_{ij}$, and the total population of the region, $P(S) = \sum_S p_{ij}$. Thus we will often write $D(C,P)$, where C and P are the count and population of the region under consideration. It is important to note that, while the term "density" is typically understood to mean the ratio of count to population, we use the term in a much broader sense, to denote a class of *density functions* D which includes the "standard" density function $D_1(C,P) = \frac{C}{P}$. For our purposes, we assume that the density function D satisfies the following three properties:

1. For a fixed population, density increases monotonically with count: $\frac{\partial D}{\partial C}(C,P) \geq 0$ for all (C,P).
2. For a fixed count, density decreases monotonically with population: $\frac{\partial D}{\partial P}(C,P) \leq 0$ for all (C,P).
3. For a fixed ratio $\frac{C}{P}$, density increases monotonically with population: $\frac{\partial D}{\partial P}(C,P) + \frac{C}{P}\frac{\partial D}{\partial C}(C,P) \geq 0$ for all (C,P).

The first two properties state that an overdensity is present when a large count occurs in a small population. In the case of a uniform population distribution, the population of a region is proportional to its area, and thus an overdensity is present when a large count occurs in a small area. The third property states, in essence, that an overdensity is more significant when the underlying population is large. This is true because smaller populations will have higher variance in densities. However, we also allow D to remain constant as population increases for a fixed ratio $\frac{C}{P}$, thus including the standard density function; we do not, however, allow functions where D decreases in this case. In our discussion below, we will also make one more assumption involving the second partials of D; this fourth property is not strictly necessary but makes our computation easier (eliminating the need to check for local maxima of the density function). A large class of functions satisfy all four properties, including Kulldorff's spatial scan statistic, discussed in detail below.

1.1 Related work

This work builds on our previous work on detection of spatial overdensities [7]. The primary difference is in the statement of the problem: the goal of the present work is to detect the most significant *rectangular* region, as opposed to the most significant *square* region. This extension is extremely important in epidemiological applications because disease clusters are often elongated: airborne pathogens may be blown by wind, creating an ellipsoid "plume," and waterborne pathogens may be carried along the path of a river. In each of these cases, the resulting clusters have high aspect ratios, and a test for squares (or circles, as in Kulldorff's original scan statistic) will have low power for detecting the overdensity. While our discussion below focuses on finding "axis-aligned" rectangular regions, it can be easily extended to find rectangular regions which are not aligned with the coordinate axes. One simple method of doing this is to examine multiple "rotations" of the data, mapping each to a separate grid and computing the maximum density region for each grid. In this case, we must also perform the same rotations on each replica grid, and thus the complexity of the algorithm is multiplied by the number of rotations. However, if we have information about relevant conditions such as wind direction or the flow of a river, we can use this information to align the coordinate axes, reducing or avoiding the need to examine multiple rotations.

While this change from square to rectangular regions has little effect on the underlying statistics, it creates a much more difficult computational problem. While the maximum density square region can be found naively in $O(N^3)$ time for an $N \times N$ grid, finding the maximum density rectangular region requires $O(N^4)$, and thus is computationally infeasible for even moderately sized grids. Our solution is similar to our previous work in that we propose a multiresolution partitioning algorithm: we divide the grid into overlapping regions, bound the maximum score of subregions contained in each region, and prune regions which cannot contain the maximum density region. Within that general framework, however, there are major differences in our multiresolution data structure and the resulting algorithm. Our current "center-based" approach (using a novel "overlap-kd tree" data structure) allows us to achieve tighter upper bounds on the score of a region, allowing much more pruning to take place. As a result, our algorithm gives huge speedups (as compared to the naive approach) without relying on approximation; the non-approximate version of our previous algorithm only resulted in speedups when the maximum density region was sufficiently dense, and actually performed slower than naive in some cases. Our current algorithm is also more robust: while the previous algorithm was severely slowed by non-uniform populations, these are shown to have little or no effect on the current method.

Various other methods for finding "dense clusters" have been proposed in the data mining literature, including grid-based hierarchical methods such as CLIQUE [1], MAFIA [3], and STING [12]. Our work differs from these in three main ways: most importantly, as discussed above, our goal is not only to find the highest scoring cluster, but to determine the statistical significance of that cluster (whether it is a true overdensity, or if it is likely to have occurred by chance). Second, our method deals with non-uniform underlying populations: this is particularly essential for real-world epidemiological applications, in which an overdensity of disease cases is more significant if the underlying population is large, and is also important in many other applications where an "overdensity" is defined relative to some other statistic (e.g. population, baseline score, or covariate). Finally, our method is applicable to a wide class of density measures D, where the other algorithms are specific to the "standard" density measure $D_1(S) = \frac{C(S)}{P(S)}$. The D_1 measure is the ratio of count per unit population; for example, in epidemiology, maximizing D_1 corresponds to finding the region with the highest *observed* disease rate. However, this is not generally the region we are interested in finding, since a region with a high disease rate and very small population (e.g. one person, who happens to be sick) is not likely to be significant. In fact, the "region" with the highest D_1 will be the single square with highest $\frac{c_{ij}}{p_{ij}}$. This is because D_1 density is *monotonic*: if a region S with $D_1(S) = d$ is partitioned into any set of disjoint subregions, at least one subregion S' will have $D_1(S') \geq d$. Thus the other algorithms, rather than maximizing D_1, search for maximally sized regions with D_1 greater than some threshold. There are two disadvantages to this approach: one is the difficulty of measuring statistical significance within this framework. The other is that the algorithms rely heavily on the monotonicity of the D_1 measure by first finding "dense" 1×1 squares, then merging adjacent squares in bottom-up fashion. For a non-monotonic density measure such as Kulldorff's, it is possible to have a large dense region where none of its subregions are themselves dense, so bottom-up methods are not guaranteed to find the correct region. Here, we will optimize with respect to arbitrary non-monotonic density measures, and thus require a different approach from CLIQUE, MAFIA, or STING.

1.2 The spatial scan statistic

A non-monotonic density measure which is of great interest to epidemiologists is Kulldorff's *spatial scan statistic* [4], which we denote by D_K. This statistic is in common use for finding significant spatial clusters of disease cases, which are often indicative of an emerging outbreak. Kulldorff's statistic assumes that counts c_{ij} are generated by an inhomogeneous Poisson process with mean qp_{ij}, where q is the underlying "disease rate" (or expected value of $\frac{C}{P}$). We then calculate the log of the likelihood ratio of two possibilities: that the disease rate q is higher in the region than outside the region, and that the disease rate is identical inside and outside the region. For a region with count C and population P, in a grid with total count C_{tot} and population P_{tot}, we can calculate:

$$D_K = C \log \frac{C}{P} + (C_{tot} - C) \log \frac{C_{tot} - C}{P_{tot} - P} - C_{tot} \log \frac{C_{tot}}{P_{tot}}$$

if $\frac{C}{P} > \frac{C_{tot}}{P_{tot}}$, and $D_K = 0$ otherwise. [4] proved that the spatial scan statistic is *individually most powerful* for finding a single significant region of elevated disease rate: for a fixed false positive rate, and for a given set of regions tested, it is more likely to detect the overdensity than any other test statistic. Again, we note that our algorithm is general enough to use any density measure, and in some cases we may wish to use measures other than Kulldorff's. For instance, if we have some idea of the size of the maximum density region, we can use the D_r measure, $D_r(S) = \frac{C(S)}{P(S)^r}$, $0 < r < 1$, with larger r corresponding to tests for smaller clusters. We have also derived a variant of the D_r measure for normally distributed counts, where the cumulative statistics of a region are not its raw count and population, but a weighted average of squares' z-scores. We are in the process of using this statistic to look for emerging epidemics based on national sales of over-the-counter medications.

1.3 Randomization testing

Once we have found the maximum density region (*mdr*) of grid G according to our density measure, we must still determine the statistical significance of this region. Since the exact distribution of the test statistic is only known in special cases (such as D_1 density with a uniform underlying population), in general we must find the region's p-value by randomization. To do so, we run a large number R of random replications, where a replica has the same underlying populations p_{ij} as G, but assumes a uniform disease rate $q_{rep} = \frac{C_{tot}(G)}{P_{tot}(G)}$ for all squares. For each replica G', we first generate all counts c_{ij} randomly from an inhomogeneous Poisson distribution with mean $q_{rep} p_{ij}$, then compute the maximum region density (*mrd*) of G' and compare this to $\text{mrd}(G)$. The number of replicas G' with $\text{mrd}(G') \geq \text{mrd}(G)$, divided by the total number of replications R, gives us the p-value for our maximum density region. If this p-value is less than .05, we can conclude that the discovered region is statistically significant (unlikely to have occurred by chance) and is thus a "spatial overdensity." If the test fails, we have still discovered the maximum density region of G, but there is not sufficient evidence that this is an overdensity.

1.4 The naive approach

The simplest method of finding the maximum density region is to compute the density of all rectangular regions of sizes $k_1 \times k_2$, where $k_{min} \leq k_1, k_2 \leq k_{max}$.[1] Since there are a total of $(N - k_1 + 1)(N - k_2 + 1)$ regions of each size $k_1 \times k_2$, there are a total of $O(N^4)$ regions to examine. We can compute the density of any region S in $O(1)$, by first finding the count $C(S)$ and population $P(S)$, then applying our density measure $D(C,P)$.[2] This allows us to compute the mdr of an $N \times N$ grid G in $O(N^4)$ time. However, significance testing by randomization also requires us to find the mrd for each replica G', and compare this to $\text{mrd}(G)$. Since calculation of the mrd takes $O(N^4)$ time for each replica, the total complexity is $O(RN^4)$, and R is typically large (we assume $R = 1000$). As discussed in [7], several tricks may be used to speed up this procedure for cases where there is no significant spatial overdensity, but these do not help in cases when an overdensity is found. In general, the $O(N^4)$ complexity of the naive approach makes it infeasible for even moderately sized grids: we estimate a runtime of 45 days for a 256×256 grid on our test system, which is clearly far too slow for real-time detection of disease outbreaks.

While one alternative would be to search for an approximate solution using one of the variety of cluster detection algorithms in the literature, we present an algorithm which is exact (always finds the maximum density region) and yet is much faster than naive search. The key intuition is that, since we only care about finding the maximum density region, we do not need to search over every single rectangular region: in particular, we do not need to search a set of regions if we can prove (based on other regions we have searched) that none of them can be the mdr. As a simple example, if a given region has a very low count, we may be able to conclude that *no* subregion contained in that region can have a score higher than the mrd, and thus we do not need to actually compute the score of each subregion. These observations suggest a top-down, *branch-and-bound* approach: we maintain the current maximum score of the regions we have searched so far, calculate upper bounds on the scores of subregions contained in a given region, and *prune* regions which cannot contain the mdr. Similarly, when we are searching a replica grid, we only care about whether the mrd of the replica is higher than the mrd of the original grid. Thus we can use the mrd of the original grid for pruning on the replicas, and can stop searching a replica if we find a region with score higher than this mrd.

[1] We use $k_{min} = 3$ and $k_{max} = N$ throughout.

[2] An old trick makes it possible to compute the count and population of any rectangular region in $O(1)$: we first form a matrix of the cumulative counts, then compute each region's count by adding/subtracting at most four cumulative counts, and similarly for populations.

2. OVERLAP-MULTIRES PARTITIONING

Our top-down approach to cluster detection can be thought of as a multiresolution search of the space under consideration: we search first at coarse resolutions (large regions), then at successively finer resolutions (smaller regions) as necessary. This suggests that a hierarchical, space-partitioning data structure such as kd-trees [9], mrkd-trees [2], or quadtrees [10] may be useful in speeding up our search. However, our desire for an exact solution makes it difficult to apply these data structures to our problem. In a kd-tree, each spatial region is recursively partitioned into two disjoint "child" regions, each of which can then be further subdivided. The difficulty, however, is that many subregions of the parent are not contained entirely in either child, but overlap partially with each. Thus, in addition to recursively searching each child for the mdr, we must also search over all of these "shared" regions at each level of the tree.[3] Since there are $O(N^4)$ shared regions even at the top level of the tree (i.e. regions partially overlapping both halves of grid G), an exhaustive search over all such regions is too computationally expensive, and thus a different partitioning approach is necessary.

An initial step toward our partitioning can be seen by considering two divisions of a rectangular spatial region S: first, into its left and right halves (which we denote by S_1 and S_2), and second, into its top and bottom halves (which we denote by S_3 and S_4). Assuming that S has size $k_1 \times k_2$, this means that S_1 and S_2 have size $\frac{1}{2}k_1 \times k_2$, and S_3 and S_4 have size $k_1 \times \frac{1}{2}k_2$. Considering these four (overlapping) halves, we can show that any subregion of S either a) is contained entirely in (at least) one of $S_1 \ldots S_4$, or b) contains the centroid of S. Thus one possibility would be to search S by exhaustively searching all regions containing its centroid, then recursing the search on its four "children" $S_1 \ldots S_4$. Again, there are $O(N^4)$ "shared" regions at the top level of the tree (i.e. regions containing the centroid of grid G), so an exhaustive search is infeasible.

Our solution, as in our previous work [7], is a partitioning approach in which adjacent regions partially overlap, a technique we call "overlap-multiresolution partitioning," or "overlap-multires" for short. Again we consider the division of S into its left, right, top, and bottom "children." However, while in the discussion above each child contained exactly half the area of S, now we let each child contain *more* than half the area. We again assume that region S has size $k_1 \times k_2$, and we choose fractions $f_1, f_2 > \frac{1}{2}$. Then S_1 and S_2 have size $f_1 k_1 \times k_2$, and S_3 and S_4 have size $k_1 \times f_2 k_2$. This

[3] Note that an attempt to find the two "pieces" of the mdr, one in each child, and then merge the two, fails because of the non-monotonicity of the density measure: the mdr may have a *higher* score than either of its two pieces!

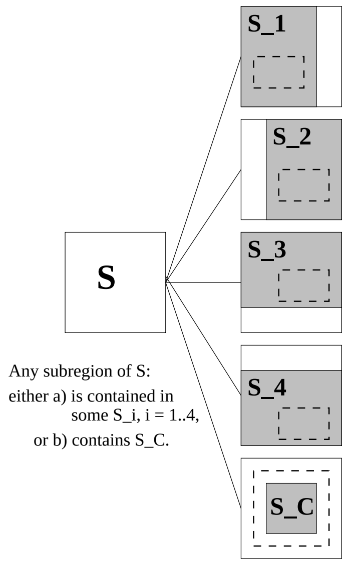

Figure 1: Overlap-multires partitioning of region S

partitioning (for $f_1 = f_2 = \frac{3}{4}$) is illustrated in Figure 1. Note that there is a region S_C common to all four children; we call this region the *center* of S. The size of S_C is $((2f_1 - 1)k_1 \times (2f_2 - 1)k_2)$, and thus the center has non-zero area. When we partition region S in this manner, it can be proved that any subregion of S either a) is contained entirely in (at least) one of $S_1 \ldots S_4$, or b) contains the center region S_C. Figure 1 illustrates each of these possibilities.

Now we can search S by recursively searching $S_1 \ldots S_4$, then searching all of the regions contained in S which contain the center S_C. Unfortunately, at the top level there are still $O(N^4)$ regions contained in grid G which contain its center G_C. However, since we know that each such region contains the large region G_C, we can place very tight bounds on the score of these regions, often allowing us to prune most or all of them. (We discuss how these bounds are calculated in the following subsection.) Thus the basic outline of our search procedure (ignoring pruning, for the moment) is:

```
overlap-search(S)
{
  call base-case-search(S)
  define child regions S_1..S_4, center S_C as above
  call overlap-search(S_i) for i=1..4
  for all S′ such that S′ is contained in S and contains S_C,
    call base-case-search(S′)
}
```

Now we consider how to select the fractions f_1 and f_2 for each call of overlap-search, and characterize the resulting set Φ of regions S on which overlap-search(S) is called. Regions $S \in \Phi$ are called *gridded regions*, and regions $S \notin \Phi$ are called *outer regions*. For simplicity, we assume that the grid G is square, and that its size N is a power of two. We begin the search by calling overlap-search(G). Then for each recursive call to overlap-search(S), where the size of S is $k_1 \times k_2$, we set $f_1 = \frac{3}{4}$ if $k_1 = 2^r$ for some integer r, and $f_1 = \frac{2}{3}$ if $k_1 = 3 \times 2^r$ for some integer r. We define f_2 identically in terms of k_2, and then the child regions $S_1 \ldots S_4$ and the center region S_C are defined in terms of f_1 and f_2 as above. This choice of f_1 and f_2 has the useful property that all gridded regions have sizes 2^r or 3×2^r for some integer r. For instance, if the original grid G has size 64×64, then the children of G will be of sizes 64×48 and 48×64, and the grandchildren of G will be of sizes 64×32, 48×48, and 32×64. This process can be repeated recursively down to regions of size $k_{min} \times k_{min}$, forming a structure that we call an *overlap-kd tree*. The first two levels of the overlap-kd tree are shown in Figure 2. Note that even though grid G has four child regions, and each of its child regions has four children, G has only ten (not 16) distinct grandchildren, several of which are the child of multiple regions.

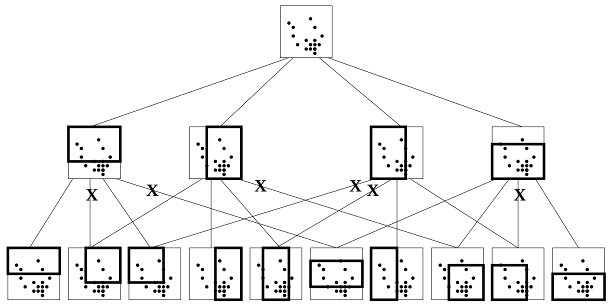

Figure 2: The first two levels of the overlap-kd tree. Each node represents a gridded region (denoted by a thick rectangle) of the entire dataset (thin square and dots).

Our overlap-kd tree has several nice properties, which we present here without proof. First, for every rectangular region $S \subseteq G$, either S is a gridded region (contained in the overlap-kd tree), or there exists a unique gridded region S' such that S is an outer region of S' (i.e. S is contained in S', and contains the center region of S'). This means that, if overlap-search is called exactly once for each gridded region, and no pruning is done, then base-case-search will be called exactly once for every rectangular region $S \subseteq G$. In practice, we will prune many regions, so base-case-search will be called *at most once* for every rectangular region, and every region will be either searched or pruned. The second nice property of our overlap-kd tree is that the total number of gridded regions $|\Phi|$ is $O((N \log N)^2)$ rather than $O(N^4)$. This implies that, if we are able to prune (almost) all outer regions, we can find the mdr of an $N \times N$ grid in $O((N \log N)^2)$ time. In fact, we may not even need to search all gridded regions, so in many cases the search will be even faster.

Before we consider how to calculate score bounds and use them for pruning, we must first deal with an essential issue in searching overlap-kd trees. Since a child region may have multiple parents, how do we ensure that each gridded region is examined only once, rather than being called recursively by each parent? One simple answer is to keep a hash table of the regions we have examined, and only call overlap-search(S) if region S has not already been examined. The disadvantage of this approach is that it requires space proportional to the number of gridded regions, $O((N \log N)^2)$, and spends a substantial amount of time doing hash queries and updates. A more elegant solution is what we call *lazy expansion*: rather than calling overlap-search(S_i) on all four children of a region S, we selectively expand only certain children at each stage, in such a way that there is exactly one path from the root of the overlap-kd tree to any node of the tree. One such scheme is shown in Figure 2: if the path between a parent and child is marked with an X, lazy expansion does not make that recursive call. No extra space is needed by this method; instead, a simple set of rules is used to decide which children of a node to expand. A child is expanded if it has no other parents, or if the parent node has the highest *priority* of all the child's parents. We give parents with lower aspect ratios priority over parents with higher aspect ratios: for example, a 48×48 parent would have priority over a 64×32 parent if the two share a 48×32 child. This rule allows us to perform variants of the search

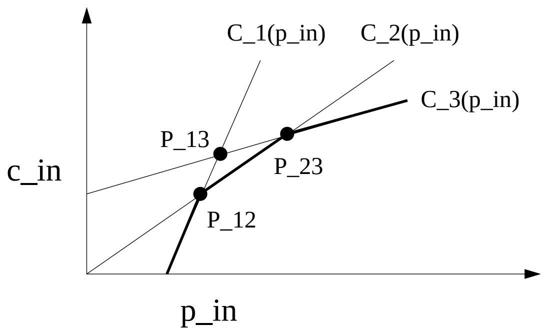

Figure 3: Maximizing count c_{in} for a given population p_{in}. Count must be less than $C_1(p_{in})$, $C_2(p_{in})$, and $C_3(p_{in})$.

where regions with very high aspect ratios are not included; an extreme case would be to only search for squares, as in our previous work. Within an aspect ratio, we fix an arbitrary priority ordering. Since we maintain the property that every node is accessible from the root, the correctness of our algorithm is maintained: every gridded region will be examined (if no pruning is done), and thus every region $S \subseteq G$ will be either searched or pruned.

2.1 Score bounds

We now consider which regions can be *pruned* (discarded without searching) during our multiresolution search procedure. First, given some region S, we must calculate an upper bound on the scores $D(S')$ for regions $S' \subset S$. More precisely, we are interested in two upper bounds: a bound on the score of *all* subregions $S' \subset S$, and a bound on the score of the *outer* subregions of S (those regions contained in S and containing its center S_C). If the first bound is less than or equal to the mrd, we can prune region S completely; we do not need to search any (gridded or outer) subregion of S. If only the second bound is less than or equal to the mrd, we do not need to search the outer subregions of S, but we must recursively call overlap-search on the gridded children of S. If both bounds are greater than the mrd, we must both recursively call overlap-search and search the outer regions.

Now we will explain the calculation of the second bound (on subregions containing the center); the calculation of the first bound (on *all* subregions) can be treated as a special case where the population, count, and area of the center are zero. We begin by assuming as known various pieces of information about the subregions of S; we discuss below how these are obtained. This information includes: upper and lower bounds p_{max}, p_{min} on the population of subregions S'; an upper bound d_{max} on the D_1 density of S'; an upper bound d_{inc} on the D_1 density of $S' - S_C$; and a lower bound d_{min} on the D_1 density of $S - S'$. We also know the count C and population P of region S, and the count c_{center} and population p_{center} of region S_C. Let c_{in} and p_{in} be the count and population of S'; these are presently unknown. To find an upper bound on $D(S')$, we must calculate the values of c_{in} and p_{in} which maximize $D(c_{in}, p_{in})$, subject to the given constraints:

1. $\frac{c_{in}-c_{center}}{p_{in}-p_{center}} \leq d_{inc}$
2. $\frac{c_{in}}{p_{in}} \leq d_{max}$
3. $\frac{C-c_{in}}{P-p_{in}} \geq d_{min}$
4. $p_{min} \leq p_{in} \leq p_{max}$

This potentially difficult maximization problem could be solved by convex programming, but is made much easier by the properties of

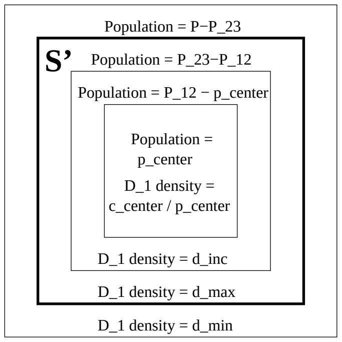

Figure 4: Division of region S into layers of differing density. In the typical case, subregion S' includes all but the outer layer.

the density function D. Since $\frac{\partial D}{\partial C} \geq 0$, we know that the maximum value of D for a given p_{in} occurs when c_{in} is maximized subject to the constraints. We solve the first three constraints for c_{in}, giving us $c_{in} = \min(C_1, C_2, C_3)$, where:

$$C_1 = d_{inc}p_{in} - (d_{inc}p_{center} - c_{center}) = d_{inc}p_{in} - B_1$$

$$C_2 = d_{max}p_{in}$$

$$C_3 = d_{min}p_{in} + (C - d_{min}P) = d_{min}p_{in} + B_3$$

In the typical case,[4] we have $d_{min} \leq d_{max} \leq d_{inc}$, $B_1 > 0$, and $B_3 > 0$: this means that $c_{in} = C_1$ for small p_{in}, $c_{in} = C_2$ for moderate p_{in}, and $c_{in} = C_3$ for large p_{in}, as illustrated in Figure 3. Thus we can solve for the intersection points P_{12}, P_{13}, and P_{23}, where $C_i \leq C_j$ for $p_{in} \leq P_{ij}$, and we use these quantities to find the maximum allowable count c_{in} for a given p_{in}. Solving the equations, we find that $P_{12} = \frac{B_1}{d_{inc}-d_{max}}$, $P_{13} = \frac{B_1+B_3}{d_{inc}-d_{min}}$, and $P_{23} = \frac{B_3}{d_{max}-d_{min}}$. In the typical case,[5] we have $0 < P_{12} \leq P_{13} \leq P_{23} < \infty$. In this case, we use the values of P_{12} and P_{23}, and the value P_{13} is not needed. Then the count $c_{in} = c_{center} + d_{inc}(p_{in} - p_{center})$ for $p_{center} \leq p_{in} \leq P_{12}$, $c_{in} = d_{max}p_{in}$ for $P_{12} \leq p_{in} \leq P_{23}$, and $c_{in} = d_{max}P_{23} + d_{min}(p_{in} - P_{23})$ for $p_{in} \geq P_{23}$. This is illustrated by Figure 4: the region S is separated into four "layers" of differing densities. Starting from the inside, we have the center (with a known population p_{center} and count c_{center}), a layer of high D_1 density d_{inc}, a layer of moderate D_1 density d_{max}, and a layer of low D_1 density d_{min}.

Now we can write c_{in} as a function of p_{in}, and thus the score D becomes a function of the single variable p_{in}. Where does the maximum of this function occur? Again we rely on properties of the function $D(C,P)$, and a case-by-case analysis is necessary. In the typical case $d_{inc} > d_{max} > \frac{c_{center}}{p_{center}}$, we know that the score increases with population in the "high density" and "moderate density" layers. This follows from two properties of our density function: $\frac{\partial D}{\partial C} \geq 0$ and $\frac{\partial D}{\partial P} + \frac{C}{P}\frac{\partial D}{\partial C} \geq 0$. In the high density layer, the D_1 density of S' $\left(\frac{c_{in}}{p_{in}}\right)$ increases from $\frac{c_{center}}{p_{center}}$ to d_{max} as we add more population, so the score D is monotonically increasing with population. In the moderate density layer, the D_1 density of S' stays constant (at d_{max}) as population increases, so again D is monotonically increasing. In the low density layer, D_1 density of S' *decreases* as population increases: in this case, since count and population are both

[4]We must also handle a variety of special cases where one or more of these inequalities are violated, and some constraints may not be relevant. We omit the details of this case-by-case analysis.
[5]See previous note.

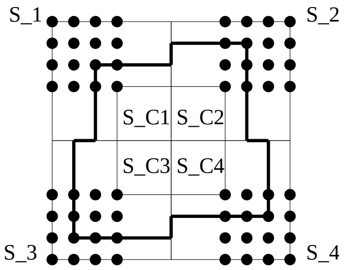

Figure 5: Quartering of region S

increasing, the score may increase or decrease. We assume that the score function D has no local maxima in the interval (P_{23}, P), and thus that the maximum occurs either at $(c_{in}, p_{in}) = (d_{max}P_{23}, P_{23})$ or at $(c_{in}, p_{in}) = (C, P)$.[6] We are only interested in finding subregions with scores *higher* than the parent, so we can ignore the latter case. Thus our upper bound on $D(S')$ is $D(c_{in}, p_{in})$, where $p_{in} = P_{23}$ and $c_{in} = d_{max}p_{in}$. The various special cases, where one or more of the inequalities above are violated, are handled similarly using the intersection points P_{12}, P_{13}, and P_{23} as necessary. We also must adjust our value of p_{in} if it violates the inequality $p_{min} \leq p_{in} \leq p_{max}$, adjusting c_{in} accordingly given the density of the layers being added or subtracted.

We now consider how the bounds on populations and D_1 densities are obtained. The simplest method of doing so is to use global values: first, we precompute the minimum and maximum population and D_1 density of all single squares s_{ij} in the grid. This gives us usable (though very conservative) values for d_{min}, d_{max}, and d_{inc}. We can also use the minimum and maximum square populations, together with the minimum and maximum area of a region, to obtain bounds p_{min} and p_{max}. Slightly less conservative bounds can be obtained using our assumption of a minimum region size $k_{min} = 3$. Any $k_1 \times k_2$ region, where $k_1, k_2 \geq 3$, can be tiled with rectangles of sizes $3 \leq k_1, k_2 \leq 5$. Thus we can precompute the minimum and maximum D_1 density and population per square of all such rectangles in $O(N^2)$ time, and use these rather than the single square bounds when allowable. For example, when bounding maximum score of the outer regions of S, we can use the less conservative bound for d_{max}; when bounding maximum score of *all* subregions of S, we can also use the less conservative bound for d_{inc}.

These global bounds on populations and D_1 densities are inexpensive to compute (we need only compute them once per grid), but are very conservative estimates of the densities of squares in a given region. We use these bounds in our algorithm as a sort of "first pass" which prunes many regions but also leaves many unpruned. If a region survives this round of pruning, we compute much tighter bounds on subregion scores in a "second pass," which is also more computationally expensive. To do so, we obtain tighter bounds on the population and D_1 density of S' using a novel technique we term *quartering*, then use these constraints to bound $D(S')$ as above.

Given a region S of size $k_1 \times k_2$, with a (non-zero) center region S_C, the quartering procedure calculates bounds on subregion population and D_1 density in $O(k_1k_2)$ time. The first step of quartering is to divide S into its four (non-overlapping) quadrants $S_1 \ldots S_4$, as in Figure 5. We now consider each S_i separately, together with the quarter of the center (S_{Ci}) which overlaps that quadrant. For each quadrant, we consider all rectangles S'_i with one corner at the centroid of S, and one corner outside S_{Ci} (i.e. on one of the dots in Figure 5). Note that there are $O(k_1k_2)$ such rectangles, and thus we can search over all of these regions S'_i in quadratic time. Our search procedure is very simple: given a region S'_i, let p_{in}, c_{in}, and A_{in} denote its population, count, and area; let p_{out}, c_{out}, and A_{out} denote the population, count, and area of $S_i - S'_i$; and let p_{dif}, c_{dif}, and A_{dif} denote the population, count, and area of $S'_i - S_{Ci}$. We then calculate the D_1 density d and the average population per square p_s for each of S'_i, $S_i - S'_i$, and $S'_i - S_{Ci}$: $d_{in} = \frac{c_{in}}{p_{in}}$, $p_{s,in} = \frac{p_{in}}{A_{in}}$, and the other quantities (d_{out}, d_{dif}, $p_{s,out}$, $p_{s,dif}$) are defined similarly. We then set d_{max} equal to the maximum of all d_{in}, d_{inc} equal to the maximum of all d_{dif}, and d_{min} equal to the minimum of all d_{out}. Similarly, we take the minimum and maximum values of $p_{s,in}$, $p_{s,out}$, and $p_{s,dif}$; we can use these to calculate bounds p_{min} and p_{max} once we are given the minimum and maximum area of S'. In essence, what we have done is bounded the D_1 densities and populations for the piece of region S' contained in each quadrant. Then since D_1 density is monotonic, we know that the D_1 density of the entire region S' is bounded by the maximum of the max-densities and the minimum of the min-densities computed for all regions S'_i. Population per square is also monotonic, so an identical argument applies. Another way to think of this is that we are calculating bounds on population and D_1 density for all the irregular (but rectangle-like) regions containing the center C_S and consisting of one rectangle in each quadrant, as drawn in Figure 5; then these quantities are also bounds on the population and density of all *rectangles* which contain C_S.

We do not provide a formal proof here, but we note that the bounds on population and density derived by quartering are exact (i.e. no rectangle $S' \subset S$, such that $S_C \subseteq S'$, can have density or population outside these bounds) and that they are much tighter than the global population and density bounds, allowing many more regions to be pruned. However, as noted above, quartering is significantly more computationally expensive than using the global bounds, taking time quadratic in the size of region S, and thus $O(N^2)$ for large regions. This is why we first use the global bounds for pruning outer regions, and only use quartering on regions that this initial pruning does not eliminate. We also note that quartering can be done in linear time in the special case where the parent region S and its center S_C have the same row size k_1 or the same column size k_2; in this case we need only divide the region into two halves, and each half can be searched linearly and the bounds combined. We apply this linear-time "halving," as well as the standard quadratic-time quartering, in our algorithm presented below.

2.2 The algorithm

We now possess all of the algorithmic and statistical tools needed to present our algorithm in full. The basic structure is similar to the top-down "overlap-search" routine presented above, with several important differences. First, we use a best-first search (implemented using a pair of priority queues q_1 and q_2) rather than a recursive depth-first search. Our algorithm has two stages: in the first stage we examine only gridded regions, and in the second stage we search outer regions if necessary. In both stages, we prune regions whenever possible, calculating increasingly tight bounds on subregions' population and D_1 density, and using these to calculate upper bounds D_{max} on $D(S')$ as above. The first stage of our algorithm proceeds as follows, using the (loose) global bounds on population and D_1 density to calculate D_{max}:

[6]Formally, we assume the following constraint on the first and second partials of D: $D_P^2 D_{CC} + D_C^2 D_{PP} - 2D_C D_P D_{CP} \geq 0$. This is true for a large class of functions, including Kulldorff's statistic. If this constraint is violated, we must also calculate $D(C,P)$ at each local maximum, which is not difficult if the number of maxima is small and each maximum is easy to calculate.

```
Add G to q_1.
While q_1 not empty:
  Get region S with highest D(S) from q_1.
  If D(S) > mrd, set mdr = S and mrd = D(S).
  If D_max(S′ in S) > mrd,
    add gridded children of S to q_1 (using lazy expansion).
  If D_max(S′ in S containing S_C) > mrd, add S to q_2.
```

Thus, after the first stage of our algorithm, we have searched or pruned all gridded regions (requiring at most $O((N \log N)^2)$ time), and the current mdr is the gridded region with highest $D(S)$. q_2 now contains the subset of gridded regions whose outer regions have not yet been pruned, prioritized by their upper bounds D_{max}. The second stage of our algorithm proceeds as follows:

```
While q_2 not empty (and some S on q_2 has D_max(S) > mrd):
  Get region S with highest D_max(S) from q_2.
  Use quartering to calculate tighter pop and density bounds for S.
  Recalculate D_max(S) using these bounds.
  If D_max(S) > mrd, then search-outer-regions(S).
```

Now the only question left is how to perform the search-outer-regions procedure. We first note that a rectangular region requires four coordinates for specification: we use the row size k_1, the column size k_2, the minimum row x_1, and the minimum column x_2. Then a naive search of the outer regions of S could be done using four nested loops, stepping over each legal combination of these four coordinates (i.e. such that the resulting region S' is in S and contains S_C). We also use four nested loops (in the order k_1, x_1, k_2, x_2), but take several more opportunities for pruning. Once we have fixed $k_1(S')$ and $x_1(S')$, we can obtain a very tight bound on $D_{max}(S')$ by *expanding* the center region S_C and *contracting* the parent region S such that $k_1(S) = k_1(S_C) = k_1(S')$ and $x_1(S) = x_1(S_C) = x_1(S')$. We then recalculate bounds on the D_1 density and population using the new S and S_C (this can be done in linear time using "halving," the special case of quartering, since the parent and center have the same k_1), and finally recompute D_{max} for the new parent and center. Only if D_{max} is greater than the mrd do we need to loop over k_2 and x_2 for that combination of k_1 and x_1. Finally, once we have fixed k_2, we can recompute D_{max} again, since we now precisely know the area of S', giving us much tighter population bounds. Only if D_{max} is greater than the mrd must we search all x_2 for that combination of k_1, x_1, and k_2.

Thus the second stage of our algorithm can be seen as a series of "screens" that an outer region must pass through if it is to be searched. The first screen is whether the parent region is taken off q_2 and examined, the second screen is whether the parent region passes the "quartering" test, the third screen is whether the new parent region (formed after k_1 and x_1 are fixed) passes the "halving" test, and the fourth screen is whether the new parent region passes the "halving" test once the area of S' is fixed. We now examine the complexity of this procedure, given a large parent region (i.e. one containing $O(N^4)$ outer regions S'). If the parent region does not pass the first screen, we have spent only $O(1)$ to search these $O(N^4)$ regions; if the parent does not pass the second screen, we have spent only the $O(N^2)$ time required by quartering. If the parent passes the second screen, but none of the new parent regions pass through the third and fourth screens, we have spent only $O(N^2) \times O(N)$ (for halving, given each k_1 and x_1) $+ O(N^3)$ (for bounding, given each k_1, x_1, and k_2) $= O(N^3)$ time. Thus only if all four screens fail will the algorithm have $O(N^4)$ complexity; typically well over 90% of regions are eliminated at each screen, and thus we search only a small fraction of possible regions.

3. A USEFUL APPROXIMATION

As opposed to our previous results [7], the algorithm presented above gives large speedups as compared to the naive approach *without* approximation: the algorithm is guaranteed to find the maximum density rectangular region. In some cases, however, very rapid detection may be more important than guaranteed accuracy; thus we present an approximate version of the algorithm which finds the mdr 5-20x faster while maintaining over 90% accuracy.

As noted above, the "first pass" of our algorithm uses very conservative bounds on the D_1 densities of S', $S' - S_C$, and $S - S'$, derived from the global minimum and maximum density values. Thus one way to increase the speed of the algorithm is to use a closer approximation of these densities as a bound. The disadvantage is that if we use an estimate which is not guaranteed to bound the densities, we may underestimate the score of a region, and hence possibly prune away the mdr and find an incorrect region. Here we consider an approximate lower bound on the D_1 density of $S - S'$: using this bound instead of a guaranteed but much more conservative bound typically results in large speedups with minimal loss of accuracy.

To derive tighter bounds on the maximum density of a subregion S' contained in a given region S, we consider the assumptions being made by our statistical test. Kulldorff's statistic assumes, both under the null hypothesis and the alternative hypothesis, that at most one disease cluster S_{dc} exists, and that the disease rate q is expected to be uniform outside S_{dc} (or uniform everywhere, if no disease cluster exists). Thus, if S_{dc} is contained entirely in the region under consideration S, we would expect that the maximum density subregion S' of S is S_{dc}, and that the disease rate of $S - S'$ is equal to the disease rate outside S: $E\left[\frac{C-c_{in}}{P-p_{in}}\right] = \frac{C_{tot}-C}{P_{tot}-P} = d_{out}$. Assuming that the D_1 density of $S - S'$ is equal to its expected value d_{out}, we can use the derivation above (using d_{out} in place of d_{min}) to find the maximum subregion score.

The problem with this approach is that we have not compensated for the *variance* in densities. Our calculated value of d_{out} is only a lower bound for the D_1 density of $S - S'$ in the most approximate probabilistic sense, in that we expect $D_1(S - S') > d_{out}$ half the time. We can improve the accuracy of our probabilistic bound by also considering the variance of $\frac{C-c_{in}}{P-p_{in}} - \frac{C_{tot}-C}{P_{tot}-P}$. Assuming that all counts outside S_{dc} are generated by a inhomogeneous Poisson distribution with parameter qp_{ij}, we obtain: $\sigma^2\left[\frac{C-c_{in}}{P-p_{in}} - \frac{C_{tot}-C}{P_{tot}-P}\right] = \sigma^2\left[\frac{\text{Po}(q(P-p_{in}))}{P-p_{in}} - \frac{\text{Po}(q(P_{tot}-P))}{P_{tot}-P}\right] = \frac{q}{P-p_{in}} + \frac{q}{P_{tot}-P} = \frac{q(P_{tot}-p_{in})}{(P-p_{in})(P_{tot}-P)}$. Since the actual value of the parameter q is not known, we use a conservative empirical estimate: $q = \frac{C_{tot}}{P_{tot}-p_{in}}$. From this, we obtain $\sigma\left[\frac{C-c_{in}}{P-p_{in}} - \frac{C_{tot}-C}{P_{tot}-P}\right] = \sqrt{\frac{C_{tot}}{(P-p_{in})(P_{tot}-P)}}$. Then we can compute the maximum subregion density by using $d_{out} - b\sigma$, for some constant b, in place of d_{min} in the derivation above. One minor complication is that, since σ is dependent on p_{in}, we must solve equations for P_{13} and P_{23} which are quadratic rather than linear; we omit the details of this calculation.

By adjusting our approximation of d_{min} in this manner, we compute a higher score D, reducing the likelihood that we will underestimate the maximum subregion density and prune a region that should not necessarily be pruned.[7] Given a constant b, the D_1 density of $S - S'$ will be greater than $d_{out} - b\sigma$ with probability $\Pr(Z < b)$, where Z is chosen randomly from the unit normal. For $b = 2$, there is an 98% chance that we will correctly bound

[7]This also increases the number of regions searched, and thus we have a tradeoff between speed and accuracy.

$D_1(S - S')$, giving a guaranteed correct upper bound for the maximum subregion score. In practice, the maximum score will be lower than our approximate bound more often than this, since our estimates for the other parameters are conservative. Thus, though our algorithm is approximate, it is very likely to converge to the globally optimal mdr.

One interesting feature of this approximation is that we expect to *underestimate* the maximum subregion score if the disease cluster S_{dc} is not contained entirely in S, since we are calculating d_{out} based on a region which includes this region of higher density. In cases where there is only a single disease cluster, this is acceptable (and desirable) since a region not containing S_{dc} does not need to be expanded. In applications where multiple disease clusters are present, however, there is a risk that the presence of one significant disease cluster will cause the approximate algorithm to miss another more significant cluster. This phenomenon is visible in our results below: in two of the three real-world datasets, the approximate algorithms did not find the maximum density region, though they did find another cluster that was also significant. In the cases where only one disease cluster was present, as in our simulated trials, the approximate algorithms achieved high accuracy. We present results for the exact and approximate versions of the algorithm below.

4. RESULTS

We first describe results with artificially generated grids and then real-world case data. An artificial grid is generated from a set of parameters (N, k_1, k_2, μ, σ, q', q''). The grid generator first creates an $N \times N$ grid, and randomly selects a $k_1 \times k_2$ "test region." Then the population of each square is chosen randomly from a normal distribution with mean μ and standard deviation σ (populations less than zero are set to zero). Finally, the count of each square is chosen randomly from a Poisson distribution with parameter qp_{ij}, where $q = q'$ inside the test region and $q = q''$ outside the test region.

For all our simulated tests, we used grid size $N = 256$, and a background disease rate of $q'' = .001$. We tested for four different combinations of test region parameters ($k_1 \times k_2$, q'): (7×9, .01), (11×5, .002), (4×3, .002), and (0×0, .001). These represent the cases of an extremely dense disease cluster, large and small disease clusters which are significant but not extremely dense, and no disease cluster respectively. We used three different population distributions for testing: the "standard" distribution ($\mu = 10^4$, $\sigma = 10^3$), and two types of "highly varying" populations. For the "city" distribution, we randomly selected a 10×10 "city region": square populations were generated with $\mu = 5 \times 10^4$ and $\sigma = 5 \times 10^3$ inside the city, and $\mu = 10^4$ and $\sigma = 10^3$ outside the city. For the "high-σ" distribution, we generated all square populations with $\mu = 10^4$ and $\sigma = 5 \times 10^3$. For each combination of test region parameters and population distribution, run times were averaged over 20 random trials, and an additional 90 trials (for a total of 110) were used to test accuracy. A trial was counted "correct" if the algorithm either found the test region S_{dc}, or another region S with $D(S) \geq D(S_{dc})$; we emphasize that this is always the case for the exact version of our algorithm, which always finds the maximum density region.[8] We also recorded the average number of regions examined; for our algorithm, this includes calculation of score bounds as well as scores of individual regions. Separate results are presented for the original grid and for each replica; for a large number of random replications ($R = 1000$) the results per replica dominate, since total run time is $t_{orig} + R(t_{rep})$ to search the original grid and perform randomization testing. See Table 1 for results.

[8]In additional to our theoretical argument for correctness, we confirmed this empirically by running a large number of tests on smaller grids, for all possible test region sizes. These results are not given here, but we note that the algorithm found the correct region in all cases.

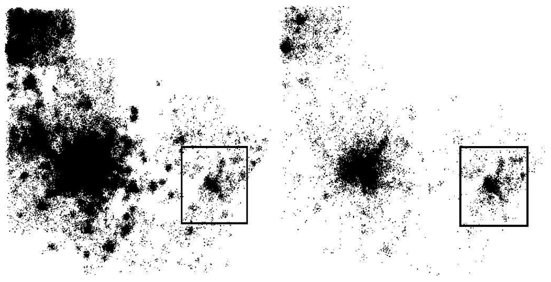
Figure 6: Emergency Department dataset. The left picture shows the "population" distribution and the right picture shows the "counts." The winning region is shown as a rectangle.

Our first observation was that the run time and number of regions searched were not significantly affected by the underlying population distribution; typically the three results differed by only 5-10%, and in many cases test regions were found *faster* for the highly varying distributions than the standard distribution. Thus Table 1, rather than presenting separate results for each population distribution, presents the average performance over all three population distributions for each test. This result demonstrates the robustness of the algorithm to highly non-uniform populations; this is very different than our previous work [7], where the algorithm was severely slowed by highly varying populations. The exact algorithm achieved average speedups ranging from 35x (for no test region), to 2300x (for an extremely dense test region) as compared to the naive approach. We note that, for the case of no test region, it is typically not necessary to run more than 10-20 randomizations before vconcluding that the discovered region is not significant; thus our true average "worst-case" results will be closer to the 95x speedup on small, significant (but not extremely dense) test regions. Since the naive approach requires approximately 45 days for a 256×256 grid with $R = 1000$, this suggests that our exact algorithm can complete the same task in less than 12 hours. We also tested two approximate variants of the algorithm, "approx-2" and "approx-3," with adjustments for density variance $b = 2$ and $b = 3$ respectively. These variants achieved up to 5000x speedups, with over 1000x speedups whenever a test region was present, and 84-173x speedups in the "no test region" case, enabling us to find the maximum density region and its significance in less than 1 hour. While all variants of the algorithm achieved 100% accuracy on the very dense test regions, the approximate versions missed some of the less dense test regions. For the larger (11×5) test regions, approx-2 and approx-3 achieved 98.5% and 99.7% accuracy respectively (averaged over the 330 trials); for smaller (4×3) test regions, these accuracies were reduced to 93.0% and 98.2% respectively. In some cases, the guaranteed accuracy of the exact algorithm may be more necessary than the additional speedups gained by the approximate algorithm; in other cases, extremely fast results are needed, and an approximation may be sufficient.

We now discuss the performance of the algorithm on various real-world datasets. Our first test set was a database of anonymized Emergency Department data collected from Western Pennsylvania hospitals in the period 1999-2002. This dataset contained a total of 630,000 records, each representing a single ED visit and giving the latitude and longitude of the patient's home location to the nearest .005 degrees ($\sim \frac{1}{3}$ mile, a sufficiently low resolution to ensure anonymity). These locations were mapped to three grid sizes:

Table 1: Performance of algorithm, simulated datasets, $N = 256$

test	method	sec/orig	speedup	sec/rep	speedup	regions (orig)	regions (rep)	accuracy
all	naive	3864	x1	3864	x1	1.03B	1.03B	100%
7x9, 0.01	exact	5.47	x706	1.68	x2300	100K	1.20K	100%
	approx-2	0.83	x4659	0.74	x5245	2.69K	16	100%
	approx-3	0.86	x4475	0.76	x5091	2.76K	16	100%
11x5, 0.002	exact	21.72	x178	12.43	x311	1.03M	196K	100%
	approx-2	1.39	x2780	0.77	x4992	87.2K	1.59K	98.5%
	approx-3	1.72	x2246	0.74	x5207	134K	2.29K	99.7%
4x3, 0.002	exact	42.96	x90	40.57	x95	2.59M	1.87M	100%
	approx-2	3.10	x1248	1.80	x2143	346K	94.7K	93.0%
	approx-3	5.75	x672	3.20	x1209	738K	227K	98.2%
no region	exact	189.68	x20	110.25	x35	27.4M	12.7M	-
	approx-2	36.67	x105	22.27	x173	5.75M	4.34M	-
	approx-3	89.97	x44	45.90	x84	16.95M	9.65M	-

$N = 128$, 256, and 512. For each grid, we tested for spatial clustering of "recent" disease cases: the "count" of a square was the number of ED visits in that square in the last two months, and the "population" of a square was the total number of ED visits in that square. See Figure 6 for a picture of this dataset, including the highest scoring region. For each of these grids, the exact and approximate versions of our algorithm found the same, statistically significant region (p-value 0/1000) as the naive approach. The major difference, of course, was in runtime and number of regions searched (see Table 2). Our algorithms found the mdr of the original grids 22-31x faster than the naive approach; however, much faster performance was achieved when searching the replica grids. The exact algorithm achieved speedups increasing from 450x to 4700x as grid size increased from 128 to 512; the approximate versions did even better, achieving 2300-24000x speedups.

Our second test set was a nationwide database of retail sales of over-the-counter cough and cold medication. Sales figures were reported by zip code; the data covered 5000 zip codes across the U.S., with highest coverage in the Northeast. In this case, our goal was to see if the spatial distribution of sales on a given day (2/14/2004) was significantly different than the spatial distribution of sales a week before (2/7/2004), and to identify a significant cluster of increased sales if one exists. Thus we used the sales on 2/7 as our underlying population distribution, and the sales on 2/14 as our count distribution. Slight modifications to Kulldorff's statistic were necessary to deal with regions with zero population and nonzero count (i.e. sales on 2/14 but not 2/7). We created four grids from this data, two using all of the national data, and two using only data from the Northeast (where a greater proportion of zip codes report sales data). For both "national" and "regional" over-the-counter data, we created grids of sizes $N = 128$ and $N = 256$, converting each zip code's centroid to a latitude and longitude. For each of these four grids, our exact algorithm found the same statistically significant region (p-value 0/1000) as the naive approach, and achieved speedups of 96-132x on the 128×128 grids and 440-739x on the 256×256 grids. The approximate versions of the algorithm did not find the correct region on these four grids, and thus we do not include these in Table 2. We note, however, that they did find another statistically significant region, though with a lower score than the mdr; it is possible that the presence of this region caused the algorithms to miss the most significant region, as discussed above.

Thus the exact version of our algorithm found the maximum density region in all of our simulated and real-world trials, while achieving speedups of at least 20x (and typically much larger) as compared to the naive spatial scan. The approximate versions of the algorithm achieved much larger speedups, though at the cost of occasionally failing to find the correct region. This speedup is extremely important for the real-time detection of disease outbreaks: if a system is able to detect an outbreak in minutes rather than days, preventive measures or treatments can be administered earlier, possibly saving many lives. We believe that our algorithm will be useful for rapid detection of significant spatial clusters in a variety of other applications as well.

5. CONCLUSIONS AND FUTURE WORK

Thus we have presented a fast multiresolution partitioning algorithm for detection of significant spatial overdensities, and demonstrated that this method results in significant (20-2000x) speedups on real and artificially generated datasets. We are currently applying this algorithm to national-level hospital and pharmacy data, attempting to detect disease outbreaks based on statistically significant changes in the spatial clustering of disease cases. Our eventual goal is the automatic real-time detection of outbreaks, and application of a fast partitioning method using the techniques presented here may allow us to achieve this difficult goal.

Additionally, we are extending the algorithm (and the underlying overlap-kd tree data structure) in various ways, making it usable for a broader range of application domains. Most importantly, overlap-kd trees can be extended to higher dimensions, as can the other techniques (e.g. quartering) used in our multiresolution search. We note, however, that various quantities (for example, number of children of a node) grow exponentially with dimension, so overlap-kd trees are probably not appropriate for very high dimensional data. Nevertheless, we hope that our techniques will be useful for various 3-D (and higher dimensional) applications, including the discovery of regions of significantly increased brain activity (corresponding to given cognitive tasks) using fMRI data. As discussed above, we also are actively engaged in deriving more powerful statistical tests for overdensities (and the corresponding density functions) under a variety of application-specific models (for example, normally distributed counts inferred from the time series of previous counts, applied to the over-the-counter retail data). As long as the density function satisfies the simple conditions described above, our algorithm can be used to rapidly find the maximum density region according to this function. Finally, we are interested in extending our search for overdensities to more general classes of multivariate density functions, thus allowing the discovery of clusters which are significant even after adjusting for multiple covariates.

Table 2: Performance of algorithm, real-world datasets

test	method	sec/orig	speedup	sec/rep	speedup	regions (orig)	regions (rep)
ED ($N = 128$)	naive	72	x1	68	x1	62.0M	62.0M
	exact	3	x24	0.15	x453	5.12M	15.9K
	approx-2	3	x24	0.03	x2266	4.50M	510
	approx-3	3	x24	0.03	x2266	4.55M	364
ED ($N = 256$)	naive	1207	x1	1185	x1	1.03B	1.03B
	exact	55	x22	1.2	x988	95.9M	74.7K
	approx-2	41	x29	0.14	x8464	69.8M	2.58K
	approx-3	42	x29	0.14	x8464	71.0M	2.10K
ED ($N = 512$)	naive	19146	x1	18921	x1	16.8B	16.8B
	exact	854	x22	4.0	x4730	1.51B	120K
	approx-2	626	x31	0.8	x23651	1.10B	13.1K
	approx-3	626	x31	0.8	x23651	1.12B	13.2K
national OTC ($N = 128$)	naive	71	x1	77	x1	62.0M	62.0M
	exact	2	x36	0.8	x96	682K	200K
national OTC ($N = 256$)	naive	1166	x1	1232	x1	1.03B	1.03B
	exact	14	x96	2.8	x440	3.24M	497K
regional OTC ($N = 128$)	naive	78	x1	79	x1	62.0M	62.0M
	exact	2	x39	0.6	x132	783K	101K
regional OTC ($N = 256$)	naive	1334	x1	1330	x1	1.03B	1.03B
	exact	13	x103	1.8	x739	3.10M	168K

6. ACKNOWLEDGEMENTS

This work was supported in part by NSF Grant IIS-0325581.

7. REFERENCES

[1] R. Agrawal, J. Gehrke, D. Gunopulus, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. ACM-SIGMOD Intl. Conf. on Mgmt. of Data*, pages 94–105, 1998.

[2] K. Deng and A. W. Moore. Multiresolution instance-based learning. In *Proc. 12th Intl. Joint Conf. on Artificial Intelligence*, pages 1233–1239, 1995.

[3] S. Goil, H. Nagesh, and A. Choudhary. MAFIA: efficient and scalable subspace clustering for very large data sets. Technical Report CPDC-TR-9906-010, Northwestern University, 1999.

[4] M. Kulldorff. A spatial scan statistic. *Communications in Statistics: Theory and Methods*, 26(6):1481–1496, 1997.

[5] M. Kulldorff. Spatial scan statistics: models, calculations, and applications. In J. Glaz and N. Balakrishnan, editors, *Scan Statistics and Applications*, pages 303–322. Birkhauser, 1999.

[6] M. Kulldorff and N. Nagarwalla. Spatial disease clusters: detection and inference. *Statistics in Medicine*, 14:799–810, 1995.

[7] D. B. Neill and A. W. Moore. A fast multi-resolution method for detection of significant spatial disease clusters. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

[8] S. Openshaw, M. Charlton, A. Craft, and J. Birch. Investigation of leukemia clusters by use of a geographical analysis machine. *Lancet*, 1:272–3, 1988.

[9] F. Preparata and M. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, 1985.

[10] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, Reading, MA, 1990.

[11] L. Waller, B. Turnbull, L. Clark, and P. Nasca. Spatial analysis to detect disease clusters. In N. Lange, editor, *Case Studies in Biometry*, pages 3–23. Wiley, 1994.

[12] W. Wang, J. Yang, and R. Muntz. STING: a statistical information grid approach to spatial data mining. In *Proc. 23rd Conference on Very Large Databases*, pages 186–195, 1997.