A Fast Multi-Resolution Method for Detection of Significant Spatial Overdensities

Daniel B. Neill Andrew W. Moore

June 2003
CMU-CS-03-154

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Given an $N \times N$ grid of squares, where each square s_{ij} has a *count* c_{ij} and an underlying *population* p_{ij}, our goal is to find the square region S with the highest *density*, and to calculate the *significance* of this region by Monte Carlo testing. Any *density measure* D, which depends on the total count and total population of the region, can be used. For example, if each count c_{ij} 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 region of maximum density would be to calculate the density measure for every square region: this requires $O(RN^3)$ calculations, where R is the number of Monte Carlo replications, and hence is generally computationally infeasible. We present a novel multi-resolution algorithm which partitions the grid into overlapping regions, 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 optimal $O(RN^2)$ time, and in practice it results in significant (10-200x) speedups as compared to the naive approach.

Keywords: algorithms, biosurveillance, cluster detection, spatial scan statistic

1 Introduction

This paper develops fast methods for *detection of spatial overdensities*: discovery of spatial regions with high scores according to some density measure, and statistical significance testing in order to determine whether these high-density regions can reasonably have occurred by chance. A major application of this work is in epidemiology: efficient algorithms are needed to identify clusters of disease cases, for purposes ranging from detection of bioterrorist attacks (ex. anthrax) to identification of environmental risk factors for diseases such as childhood leukemia (Openshaw et al, 1988; Waller et al, 1994; Kulldorff and Nagarwalla, 1995). Kulldorff (1999) also discusses many other possible fields of application, including astronomy (identifying star clusters), military reconnaissance, and medical imaging.

We consider the case in which counts are aggregated to a uniform two-dimensional grid. Assume an $N \times N$ grid of squares G, where each square $s_{ij} \in G$ is associated with a *count* c_{ij} and an underlying *population* p_{ij}. For example, in epidemiology, the count of a square may be the number of disease cases occurring in that geographical region in a given time period, while its population may be the total number of people "at-risk" for the disease. Our goal is to find the square region $S^* \subseteq G$ 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. The density measure D must be an increasing function of the total count of the region, $C(S) = \sum_{i=i_0}^{i_0+k-1} \sum_{j=j_0}^{j_0+k-1} c_{ij}$, and a decreasing function of the total population of the region, $P(S) = \sum_{i=i_0}^{i_0+k-1} \sum_{j=j_0}^{j_0+k-1} p_{ij}$. In the simple case of a uniform underlying population, $P(S) \propto k^2$, where k is the size of region S. But we focus on the more interesting case: non-uniform populations.

The problem of finding significant spatial overdensities is distinct from that solved by grid-based hierarchical methods such as CLIQUE (Agrawal et al, 1998), MAFIA (Goil et al, 1999), and STING (Wang et al, 1997), which also look for "dense clusters." There are three main differences: generalization of the density measure, applicability to non-uniform underlying populations, and testing for statistical significance. First, our method is applicable to any density measure D, while the other algorithms are specific to the "standard" density measure $D_1(S) = \frac{C(S)}{P(S)}$. The D_1 measure is the number of points per unit population, or assuming a uniform population, number of points per unit area; in our epidemiological example this corresponds to finding the region with the highest observed disease rate. Unlike many other density measures, D_1 is *monotonic*: if a region S with density d is partitioned into any set of disjoint subregions $S_1 \ldots S_n$, at least one subregion will have density $d' \geq d$. Thus it is not particularly useful to find the "region" with maximum D_1, since this will be the single square with the highest value of $\frac{c_{ij}}{p_{ij}}$. Instead, the other algorithms search for maximally sized connected regions with D_1 density greater than some threshold; they rely heavily on the monotonicity of D_1 by first finding dense "units" (i.e. individual 1×1 squares), then merging adjacent dense units in bottom-up fashion. For a non-monotonic density measure such as $D_r = \frac{C}{P^r}$ (where $0 < r < 1$), it is possible to have a large dense region where none of its subregions are themselves dense, so a bottom-up merging approach may not succeed. We focus on finding a single, maximum density region with respect to an arbitrary non-monotonic density measure, and thus use an approach which is substantially different from CLIQUE, MAFIA, or STING. Our method also differs from these in that it deals with non-uniform underlying populations: this is particularly important for real-world epidemiological applications. Note that we cannot simply "normalize" the problem by computing densities of each individual square; for non-monotonic density measures, an overdensity is more significant if the underlying population is large, so we must take populations into account as well. This brings us to the third difference between our algorithm and those mentioned above: the need for statistical significance testing. Our goal is not only to find the maximum density region, but also to test whether that cluster is a true "overdensity" or if it is likely to have occurred by chance; we discuss this process in more detail in the following subsection.

1.1 The spatial scan statistic

A non-monotonic density measure which is of great interest to epidemiologists is Kulldorff's *spatial scan statistic* (Kulldorff, 1997), which we denote by D_K. The spatial scan 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 the D_1 density). 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.[1] 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 0 otherwise. Kulldorff (1997) demonstrated that the spatial scan statistic is *individually most powerful* for finding a single significant region of elevated disease rate: it is more likely to detect the overdensity than any other test statistic, and thus we focus primarily on this density measure in our experiments. We should note, however, 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 control this by using the D_r measure, with larger values of r corresponding to tests for smaller clusters.

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. To do so, we must deal with the problem of *multiple hypothesis testing*: even in a dataset where all counts are generated independently (and thus, there is no spatial component to the clustering) there will be random variation in the region densities, and by chance some regions will have higher densities than others. This problem is amplified by searching over millions of regions– the region with the highest score is going to appear very significant, even if there is no spatial process in operation. For a simple example, consider a grid where each square has population 1000 and number of cases independently distributed with mean and variance 100. For an 8×8 grid, we would be very surprised to see a square with count 150 (probability 1 in 50000), but for a 1024×1024 grid this would be a fairly common occurrence (probability 1 in 4). 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 perform Monte Carlo simulation for our hypothesis test. 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

[1] In both cases, we assume that q is uniform within the region, and uniform outside the region.

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 mrd(G). The number of replicas G' with mrd(G') $\geq$ 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 sufficiently low (for example, 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.

1.2 The naive approach

The simplest method of finding the maximum density region is to compute the density of all square regions of sizes $k = k_{min} \ldots N$.[2] Since there are $(N - k + 1)^2$ regions of size k, this gives us a total of $O(N^3)$ regions to examine. We can compute the density of any region S in $O(1)$ time, by first finding the count $C(S)$ and population $P(S)$, then applying our density measure $D(C(S), P(S))$.[3] From this, it is clear that the mdr of an $N \times N$ grid G can be computed in $O(N^3)$.

However, significance testing by Monte Carlo replication also requires us to find the mrd for each replica G', and compare this to mrd(G). Since calculation of the mrd takes $O(N^3)$ time for each replica, the total complexity is $O(RN^3)$, and R is typically large (we assume $R = 1000$). Several simple tricks may be used to speed up this procedure for cases where there is no significant spatial overdensity. First, we can stop examining a replica G' immediately if we find a region with density greater than mrd(G). In this case, the exact mrd of G' does not matter; we already know that it is greater than mrd(G). Second, we can use the Central Limit Theorem to halt our Monte Carlo testing early if, after a number of replications $R' < R$, we can conclude with high confidence that the region is not significant. For cases where there *is* a significant spatial overdensity, the naive approach is still extremely computationally expensive, and this motivates our search for a faster algorithm.

2 Overlap-multires partitioning

Since the problem of detection of spatial overdensities is closely related to problems such as kernel density estimation and kernel regression, this suggests that multi-resolution partitioning techniques such as kd-trees (Preparata and Shamos, 1985) and mrkd-trees (Deng and Moore, 1995) may be useful in speeding up our search. The main difference of our problem from kernel density estimation, however, is that we are only interested in the maximum density region; thus, we do not necessarily need to build a space-partitioning tree at all resolutions. Also, the assumption that counts are aggregated to a uniform grid simplifies and

[2] We assume that a region must have size at least k_{min} to be significant, for some constant $k_{min} \geq 1$; we use $k_{min} = 3$ here.

[3] An old trick allows us to compute the count of any $k \times k$ region in $O(1)$ time: we first form a matrix of the cumulative counts, requiring time $O(N^2)$, then each region's count can be computed by adding/subtracting at most four cumulative counts. An identical trick can be performed with the populations.

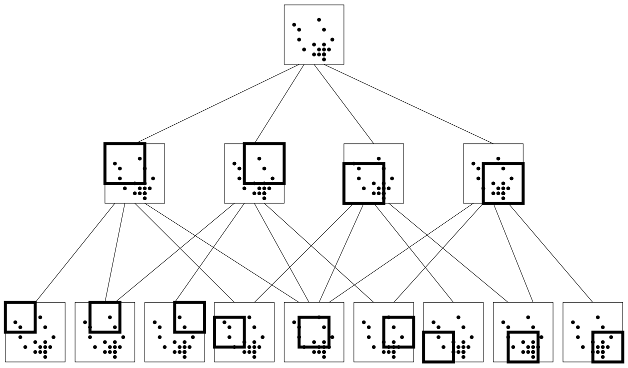

Figure 1: The first two levels of the overlap-multires tree. Each node represents a gridded region (denoted by a thick square) of the entire dataset (thin square and dots).

speeds up partitioning, eliminating the need for a computationally expensive instance-based approach. These observations suggest a top-down multi-resolution partitioning approach, in which we search first at coarse resolutions (large regions), then at successively finer resolutions as necessary. One option would be to use a "quadtree" (Samet, 1990), a hierarchical data structure in which each region is recursively partitioned into its top left, top right, bottom left, and bottom right quarters. However, a simple partitioning approach fails because of the non-monotonicity of our density measure: a dense region may be split by the partitioning process into two or more separate subregions, none of which is as dense as the original region. This problem can be prevented by a partitioning approach in which adjacent regions partially overlap, an approach which we call "overlap-multires partitioning."

To explain how this method works, we first define some notation. We denote a region S by an ordered triple (x, y, k), where (x, y) is the upper left corner of the region and k is its size. Next, we define the ω-children of a region $S = (x, y, k)$ as the four overlapping subregions of size $k - \omega$ corresponding to the top left, top right, bottom left, and bottom right corners of S: $(x, y, k - \omega)$, $(x + \omega, y, k - \omega)$, $(x, y + \omega, k - \omega)$, and $(x + \omega, y + \omega, k - \omega)$.

Next, we define a region as "even" if its size is 2^k for some $k \geq 2$, and "odd" if its size is 3×2^k for some $k \geq 0$. We define the "gridded children" (g-children) of an even region $S = (x, y, k)$ as its ω-children for $\omega = \frac{k}{4}$. Thus the four g-children of an even region are odd, and each overlaps $\frac{2}{3}$ with the child regions directly adjacent to them. Similarly, we define the g-children of an odd region $S = (x, y, k)$ as its ω-children for $\omega = \frac{k}{3}$. Thus the four g-children of an odd region are even, and each overlaps $\frac{1}{2}$ with the directly adjacent child regions. Note that even though a region has four g-children, and each of its g-children has four g-children, it has only nine (not 16) distinct grandchildren, several of which are the child of multiple regions. Figure 1 shows the first two levels of such a tree. Next, we assume that the size of the entire grid is a power of two: $N = 2^n$ for some positive integer n. Thus the entire grid $G = (0, 0, N)$ is an even region. Based on this assumption and the definitions above, we define the set of "gridded" regions of G as G and all of its "gridded descendents" (its g-children, g-grandchildren, etc.). Our algorithm focuses its search on the set of gridded regions, only searching non-gridded regions when necessary. This technique is useful because the total number of gridded regions is $O(N^2)$, as in the simple quadtree partitioning method.[4] This implies that, *if only gridded regions need to be searched*, our total time to find the mdr of a grid is $O(N^2)$. Since it takes $\Omega(N^2)$ time to generate the grid, this time bound is optimal.

2.1 Top-down pruning

This result leads us to the necessary question, when can we search only gridded regions, or alternatively, when does a given non-gridded region need to be searched? Our basic method is branch-and-bound: we perform a top-down search, and speed up this search by *pruning* regions which cannot possibly contain the mdr. Our first step is to derive an upper bound $D_{max}(S,k)$ on the density of subregions of minimum size k contained in a given region S; we examine how these bounds can be calculated in the next subsection. Then we can compare $D_{max}(S,k)$ to the density $D(S^*)$ of the best region found so far: if $D_{max}(S,k) < D(S^*)$, we know that no subregion of S with size k or more can be the mdr.

We can use this information for two types of pruning. First, if $D_{max}(S,k_{min}) < D(S^*)$, we know that *no* subregion of S can be optimal; we can prune the region completely, and not search its (gridded or non-gridded) children. Second, we can show that (for $0 < k < n$) any region of size 2^k+1 or less is contained entirely in some odd gridded region of size $\frac{3}{2}\times 2^k$. Thus, if $D_{max}(G, 2^{n-1}+2) < D(S^*)$ for the entire grid G, any optimal non-gridded region must be contained in an odd gridded region. Similarly, if $D_{max}(S, 2^k+2) < D(S^*)$ for an odd gridded region S of size 3×2^k, then any optimal non-gridded subregion of S must be contained in an odd gridded subregion of S. Thus we can search only gridded regions if two conditions hold: 1) no subregion of G of size $2^{n-1}+2$ or more can be optimal, and 2) for each odd gridded region of size 3×2^k, no subregion of size 2^k+2 or more can be optimal.

2.2 Bounding subregion density

To bound the maximum subregion density $D_{max}(S,k)$, we must find the highest possible score $D(S')$ of a subregion $S'\subseteq S$ of size k or more. Let $C = C(S)$, $P = P(S)$, and $K = \text{size}(S)$. We assume that all of these are known, as well as lower and upper bounds $[d_{min}, d_{max}]$ on the D_1 density of subregions of S. Let $c = C(S')$, $p = P(S')$, and $d = \frac{C(S')}{P(S')}$; these are presently unknown. Our first observation is that the maximum possible density of S' occurs when all of $S - S'$ has the minimum allowable D_1 density d_{min}. This gives us $pd + (P-p)d_{min} = C$; solving for p, we obtain $p = \frac{C-Pd_{min}}{d-d_{min}}$ and $c = dp = \frac{C-Pd_{min}}{1-d_{min}/d}$. Note that since both p and c decrease with increasing d, we are *not* guaranteed that the maximum of $D(S')$ occurs at $d = d_{max}$. However, we can prove (for both the D_K and D_r density measures) that no local maxima occur for d in $(\frac{C}{P}, d_{max})$. Thus the maximum occurs either at $d=\frac{C}{P}$ or $d = d_{max}$. If $d = \frac{C}{P}$, we obtain $c = C$ and $p = P$, giving a density identical to the parent region. Hence, we know that if $D(S') > D(S)$, the maximum value of $D(S')$

[4] This result is surprising since there are twice as many levels of recursion as for simple partitioning ($2\log_2 N$, as opposed to $\log_2 N$). But in fact, we find that the overall branching factor per two levels is four, identical to the branching factor per level for simple partitioning. For a more detailed proof, we first note that the number of regions at level $2i$ is $(2^{i+1}-1)^2$, and the number of regions at level $2i+1$ is $4\times(2^{i+1}-1)^2$. Thus the sum is $O(\sum_{i=1}^{\log_2 N} 5\times 2^{2i+2}) = O(\sum_{i=1}^{\log_2 N} 4^i) = O(N^2)$.

occurs when $d = d_{max}$. Thus the corresponding population and count are $p = \frac{C - Pd_{min}}{d_{max} - d_{min}}$ and $c = d_{max}p$, and then $D(c,p)$ is an upper bound on $D_{max}(S,k)$.

We can place tighter bounds on $D_{max}(S,k)$ if we also have a lower bound $p_{min}(S,k)$ on the population of a size k subregion $S' \subseteq S$: in this case, if the value calculated for p in the equation above is less than p_{min}, we know that $D(c', p_{min})$, where $c' = C - (P - p_{min})d_{min}$, is a tighter upper bound for $D_{max}(S,k)$. We can bound p_{min} in several ways. First, if we know the minimum population $p_{s,min}$ of a single square $s \in S$, then $p_{min} \geq k^2 p_{s,min}$. Second, if we know the maximum population $p_{s,max}$ of a single square $s \in S$, then $p_{min} \geq P - (K^2 - k^2)p_{s,max}$.

At the beginning of our algorithm, we calculate $p_{s,max}(S) = \max p_{ij}$ and $p_{s,min}(S) = \min p_{ij}$ (where $s_{ij} \in S$) for each gridded region S. This calculation can be done recursively (in bottom-up fashion) in $O(N^2)$ time, since $p_{s,max/min}(S)$ is equal to the maximum/minimum $p_{s,max/min}$ of its gridded children. The resulting population statistics are used for the original grid and for all replicas. For non-gridded regions, we use the population statistics of the region's gridded parent (either an odd gridded region or the entire grid G); these bounds will be looser for the child region than for the gridded parent, but are still correct.

We also initially calculate d_{max} and d_{min}. This is done simply by finding the global maximum and minimum values of the D_1 density: $d_{max} = \max \frac{C(S')}{P(S')}$ (where $S' \subseteq G$ and $\text{size}(S') = k_{min}$), and $d_{min} = \min \frac{c_{ij}}{p_{ij}}$ (where $s_{ij} \in G$).[5] Alternatively, we could compute d_{max} and d_{min} recursively (bottom-up) for each gridded region S. This procedure is still $O(N^2)$, but takes significantly longer than computing global values for d_{min} and d_{max}, and values must be recomputed for each replica. Use of the more accurate approximation of d_{max} results in fewer regions being searched, but typically the reduction in search time is outweighed by the extra cost of computing densities over all gridded regions. In practice, we find that the global values are sufficient for good performance on most test cases.

2.3 The algorithm

Our algorithm, based on the overlap-multires partitioning scheme above, is a top-down, best-first search of the set of gridded regions, followed by a top-down, best-first search of any non-gridded regions as necessary. We use priority queues (q1,q2) for our search: each step of the algorithm takes the "best" (i.e. highest density) region from a queue, examines it, and (if necessary) adds its children to queues. The ω-children and g-children of a region S are defined above; note that the 1-children of S are its ω-children with $\omega = 1$. We also assume that regions are "marked" once added to a queue, so that a region will not be searched more than once. Finally, we use the rules and density bounds derived above to speed up our search, by pruning subregions when $D_{max}(S,k) \leq D(S^*)$. The basic pseudocode outline of our method is as follows:

```
Add G to q1.
If D_max(G,N/2+2)>mrd, add 1-children(G) to q2 with k1=N/2+2.
While q1 not empty:
```

[5] We can use the tighter bound for d_{max} since we are using it to bound the density of a square region S' of size at least k_{min}; we cannot use the tighter bound for d_{min} since we are using it to bound the density of $S - S'$, which is not square.

```
  Get best region S from q1.
  If D(S)>mrd, set mdr=S and mrd=D(S).
  If D_max(S,k_min)>mrd, add g-children(S) to q1.
  If size(S)=3(2^k) and D_max(S,2^k+2)>mrd,
    add 1-children(S) to q2 with k1=2^k+2.
While q2 not empty:
  Get best region S and value k1(S) from q2.
  If D(S)>mrd, set mdr=S and mrd=D(S).
  If D_max(S,k1(S))>mrd, add 1-children(S) to q2 with same k1.
```

These steps are first performed for the original grid, allowing us to calculate its mdr and mrd. We then perform these steps to calculate the mrd of each replica; however, several techniques allow us to reduce the amount of computation necessary for a replica. First, as discussed above, we can stop examining a replica G' immediately if we find a region with density greater than mrd(G). This is especially useful in cases where there is no significant spatial overdensity in G. Second, we can use mrd(G) for pruning our search on a replica G': if $D_{max}(S, k) < \text{mrd}(G)$ for some $S \subseteq G'$, we know that no subregion of S of size k or more can have a greater density than the mdr of the original grid, and thus we do not need to examine any of those subregions. This is especially useful where there is a significant spatial overdensity in G: a high mrd will allow large amounts of pruning on the replica grids.

We also note an important correctness property of our algorithm. Assuming all lower and upper bounds on population and density are correct (as is the case for the version of the algorithm presented so far), the algorithm will search all regions that could possibly be the mdr of G. Moreover, for each replica G', the algorithm will search all regions that could possibly have density at least mrd(G). Thus, if $D(S)$ is computed exactly, the algorithm will achieve the correct answer for the mdr $S^* = \arg\max_S D(S)$ and the mrd $D(S^*)$, as well as a low-expected-error Monte Carlo estimate of the significance of the maximum density region. Note that we are *not* guaranteed that the algorithm will search the minimum number of regions necessary to find the mdr, and in fact, we can improve the performance of the algorithm by deriving tighter bounds on the maximum score of a subregion. This is the subject of the next section.

3 Improving the algorithm

The exact version of the algorithm uses conservative estimates of the D_1 densities of S' and $S - S'$ (d_{max} and d_{min} respectively), and a loose lower bound on the population of S', to calculate $D_{max}(S, k)$. This results in an upper bound on D_{max} which is guaranteed to be correct, but is very loose and allows little pruning to be done. We can derive tighter bounds on D_{max} in two ways: by using a closer approximation to the D_1 density of $S - S'$, and by using a tighter lower bound on the population of S'. These improvements will be discussed in the following two subsections.

3.1 The outer density approximation

To derive tighter bounds on the maximum density of a subregion S' contained in a given region S, we first note that (under both the null hypothesis and the alternative hypothesis) we assume 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: $E\left[\frac{C-c}{P-p}\right] = \frac{C_{tot}-C}{P_{tot}-P} = d_{out}$. Assuming that the D_1 density of $S - S'$ is equal to its expected value d_{out}, we obtain the equation $pd_{max} + (P-p)d_{out} = C$. Solving for p, we find $p = \frac{C-Pd_{out}}{d_{max}-d_{out}}$. Then $D_{max}(S,k) = D(c,p)$, where $c = d_{max}p$.

The problem with this approach is that we have not compensated for the variance in densities: thus our calculated value of D_{max} is an upper bound for the maximum subregion density $D(S')$ only in the most approximate probabilistic sense. We would expect the D_1 density of $S - S'$ to be less than its expected value half the time, and thus we would expect $D(S')$ to be less than D_{max} at least half the time; in practice, our bound will be correct more often than this, since we are still using a conservative approximation of the D_1 density of S'. Note also that we expect to underestimate D_{max} if the disease cluster S_{dc} is not contained entirely in S: this is acceptable (and desirable) since a region not containing S_{dc} does not need to be expanded.

We can improve the correctness of our probabilistic bound by also considering the variance of $\frac{C-c}{P-p} - \frac{C_{tot}-C}{P_{tot}-P}$. Assuming that all counts outside S_{dc} are generated by a inhomogeneous Poisson distribution with parameter qp_{ij}, we obtain:

$$\sigma^2\left[\frac{C-c}{P-p} - \frac{C_{tot}-C}{P_{tot}-P}\right] = \sigma^2\left[\frac{\mathrm{Po}(q(P-p))}{P-p} - \frac{\mathrm{Po}(q(P_{tot}-P))}{P_{tot}-P}\right] = \frac{q}{P-p} + \frac{q}{P_{tot}-P} = \frac{q(P_{tot}-p)}{(P-p)(P_{tot}-P)}.$$

Since the actual value of the parameter q is not known, we use a conservative empirical estimate: $q = \frac{C_{tot}}{P_{tot}-p}$.[6] From this, we obtain $\sigma\left[\frac{C-c}{P-p} - \frac{C_{tot}-C}{P_{tot}-P}\right] = \sqrt{\frac{C_{tot}}{(P-p)(P_{tot}-P)}}$. Then we can compute p by solving the equation $pd_{max} + (P-p)(d_{out} - b\sigma) = C$, and obtain $c = d_{max}p$ and $D_{max} = D(c,p)$ as before.

By adjusting our approximation of the minimum density in this manner, we compute a higher score D_{max}, reducing the likelihood that we will underestimate the maximum subregion density and prune a region that should not necessarily be pruned. Assuming the normal approximation to the Poisson distribution, the D_1 density of $S - S'$ will be greater than $d_{out} - b\sigma$ with probability $P(Z < b)$, where Z is chosen randomly from the unit normal distribution. Thus for $b = 2$, there is an 98% chance that we will underestimate the D_1 density of $S - S'$, and thus have a guaranteed correct upper bound for the maximum subregion density. In practice, the true value of the maximum subregion density will be lower than our computed value of D_{max} more often, since we are using conservative estimates for d_{max} and q. From this calculation, we can compute a (very conservative) lower bound on the probability that our algorithm will find the correct region: assuming that there is

[6] For Monte Carlo replicas of a grid G, we could also use the exact value $q = q_{rep} = \frac{C_{tot}(G)}{P_{tot}(G)}$, though this is not done in our experiments.

only one top-down path through the multi-resolution hierarchy to the correct region, and assuming independent probabilities of pruning at each resolution, we obtain an accuracy of $(.98)^L$, where L is the number of levels of hierarchy we must search to obtain the correct region (worst case $O(N)$, but typically proportional to $\log N$). In practice, however, our accuracy is much higher than this for three reasons: conservative estimates of d_{max} and q, high correlation between probabilities of pruning at different resolutions, and multiple paths through the multi-resolution hierarchy. In fact, our experiments demonstrate that $b = 1$ is sufficient to obtain the correct region with over 90% probability (approaching 100% for sufficiently dense regions). Thus, though our algorithm is approximate (and thus, correctness is not guaranteed), it is very likely to converge to the globally optimal mdr. Hence it differs from approximate methods such as "bump hunting" (Friedman and Fisher, 1999) or greedy search techniques, which typically converge to a local maximum without any probabilistic guarantees on global optimality.

3.2 Cached population statistics

Though the approximate algorithm as presented above achieves high accuracy and fast performance for most test cases, it does not perform well on test cases where there is a large spatial variation in the square populations p_{ij}. To solve this problem, we cache the *sufficient statistics* (Moore and Lee, 1998) for populations needed to perform pruning in our algorithm. For each gridded region S and each size k that is relevant for S, we cache the minimum population $p_{min}(S, k)$ of size k subregions $S' \subseteq S$. For every gridded region S, we must know $p_{min}(S, k_{min})$ for pruning; since $p_{min}(S, k_{min})$ is equal to the minimum p_{min} of the gridded children of S, we can compute these values recursively (bottom-up) in $O(N^2)$ time. For the entire grid G, we must know $p_{min}(G, \frac{N}{2} + 2)$; this can be found by searching over the $O(N^2)$ regions (gridded and non-gridded) of size $\frac{N}{2} + 2$. Finally, for each odd gridded region S of size k, we must find $p_{min}(S, \frac{k}{3} + 2)$. To do this, we search exhaustively over all regions (gridded and non-gridded) of size $\frac{k}{3} + 2$ for each value $k = 3 \times 2^x$. Since there are $O(\log N)$ values of k to search, and each search takes $O(N^2)$ time, the total complexity is $O(N^2 \log N)$. At first glance, the additional $\log N$ factor makes this method undesirable. However, since each Monte Carlo replica has the same underlying population statistics as the original grid, we only need to cache the population statistics once, then can use them for all replicas. Under the reasonable assumption that $\log N < R$, we obtain total complexity $O(RN^2)$, and thus an amortized $O(N^2)$ per replica. In practice, we find that caching population statistics can be performed in under one minute for a 512×512 grid, while speeding up performance (on highly non-uniform populations) by a factor of 500.

4 Results

We first present two sets of experiments on artificially generated grids (for the exact and approximate versions of the algorithm), then present results on a grid generated from real-world case data. An artificial grid is generated from a set of parameters (N, k, μ, σ, q', q''). The grid generator first creates an $N \times N$ grid, and randomly selects a $k \times k$ "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.

First, we used artificial grids to measure the performance of the exact algorithm, as compared to the naive approach. The exact algorithm was tested for grids of sizes $N = 64$, 128, and 256. For each trial, we used $\mu = 10^4$ and $\sigma = 10^3$; a test region of size $k = \frac{N}{16}$ was randomly selected, and the disease rate q was set to $r \times .001$ inside the test region and .001 outside the test region. By observing the performance of the algorithm for varying r (see Table 1), we found that the exact algorithm performs faster than naive only for sufficiently dense regions. We observed three distinct cases: for very dense regions ($r \geq 25$), the algorithm's runtime (and number of regions searched) increased optimally as $O(N^2)$, as compared to the naive algorithm's complexity of $O(N^3)$, and as a result significant speedups were observed. For $r = 30$, we observed speedups of x15/x29/x60 for $N = 64$, $N = 128$, and $N = 256$ respectively. For less dense regions ($8 \leq r \leq 25$), the exact algorithm's runtime increased approximately as $O(N^3)$, but performed faster than naive by a constant factor. For $r = 10$, a speedup of 1.8-3.5x was observed. For even less dense regions ($r \leq 8$), the exact algorithm took *longer* to run than naive: for $r = 5$ a slowdown of 2.0-2.6x was observed, and for $r = 2$ a slowdown of 9.0-9.2x was observed. From these results, it is clear that the loose population and density bounds used in the exact algorithm prevent it from doing large amounts of pruning unless the maximum density region is obvious, and thus the exact algorithm is insufficient for our purposes.

Second, we used artificial grids to measure the performance and accuracy of the approximate algorithm. We tested six variants of the algorithm: three different adjustments for density variance ($b = 0, 1, 2$), each with and without cached population statistics. The approximate algorithm was tested for grids of size $N = 512$; test region sizes of $k = 16$ and $k = 4$ were used, and the disease rate q was set to .002 inside the test region and .001 outside the test region. 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 "city region" with size 16: inside the city, square populations were generated with $\mu = 10^7$ and $\sigma = 10^6$, and outside the city, square populations were generated with $\mu = 10^4$ and $\sigma = 10^3$. For the "high-σ" distribution, we generated all square populations with $\mu = 10^4$ and $\sigma = 5 \times 10^3$, giving many squares zero population. We first compared the performance of each variant of the algorithm to the naive approach for the six test cases; see Table 2 for results. For large test regions, on the standard and city distributions, all variants of the algorithm had runtimes of ~20 minutes, as compared to 44 hours for the naive approach, a speedup of 122-155x. For the other test cases, we observed two main phenomena: performance generally decreased with increasing b, and the use of cached population statistics (cps) significantly improved performance. For $b = 1$, the algorithm with cps achieved a 151x speedup for the large test region/high-σ case, while the algorithm without cps achieved only a 15x speedup. Similar results were observed for the small test region/standard case (88x speedup as opposed to 30x), and the small test region/city case (20x speedup as opposed to 6x). For the small test region/high-σ case, the algorithm with cps achieved high speedups (146x/75x/13x for $b = 0, 1, 2$), but without cps it took significantly *longer* to run than the naive algorithm (we stopped the runs after 150 hours, so these results are not included in the table).

Next, we tested accuracy by generating 50 artificial grids for each population distribution, and computing the percentage of test grids on which the algorithm was able to find the correct mdr (see Table 3). For the large test region ($k = 16$), all variants were able to find the correct mdr with high accuracy (97% for $b = 0$, 100% for $b = 1$ and $b = 2$). For the small test region, accuracy improved significantly wth increasing b: the non-variance adjusted version ($b = 0$) achieved only 45% accuracy, while the variance adjusted versions ($b = 1$ and $b = 2$) achieved 89% and 99% accuracy respectively. These results demonstrate that the approximate algorithm (with variance adjustment and cached population statistics) is able to achieve high performance and accuracy even for very small test regions and highly non-uniform population distributions.

Finally, we measured the performance of the approximate algorithm on a grid generated from real-world data. We used 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). For each record, the latitude L and longitude l were converted to a grid square s_{ij} by $i = \frac{L-L_{min}}{.005}$ and $j = \frac{l-l_{min}}{.005}$; this created a 512×512 grid. We tested for spatial clustering of "recent" disease cases: the "count" of each square was the number of ED visits in that square in the last two months, and the "population" of that square was the total number of ED visits in that square. See Figure 2 for a picture of this dataset, including the highest scoring region.

We tested six variants of the approximate algorithm on the ED dataset; the presence/absence of cached population statistics did not significantly affect the performance or accuracy for this test, so we focus on the variation in b. All three variants ($b = 0, 1, 2$), as well as the naive algorithm, found the maximum density region (of size 101) and found it statistically significant (p-value 0/1000). The major difference, of course, was in runtime and number of regions searched (see Table 4). The naive algorithm took 2.7 days to find the mdr and perform 1000 Monte Carlo replications, while each of the variants of the approximate algorithm performed the same task in $\sim$2 hours or less. The approximate algorithm took 19 minutes (a speedup of 209x) for $b = 0$, 47 minutes (a speedup of 85x) for $b = 1$, and 126 minutes (a speedup of 31x) for $b = 2$. Thus we can see that all three variants find the correct region in much less time than the naive approach.[7] This is very important for applications such as 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 allowing many lives to be saved.

[7] It should also be noted that alternative algorithms such as those presented in Kulldorff (1999) would be infeasible for this dataset, since there are $M = O(N^2)$ population points, and $G = O(N^2)$ grid points, so Kulldorff's algorithm 14.3.6 is $O(GM \log M + RGM) = O(N^4 \log N + RN^4)$.

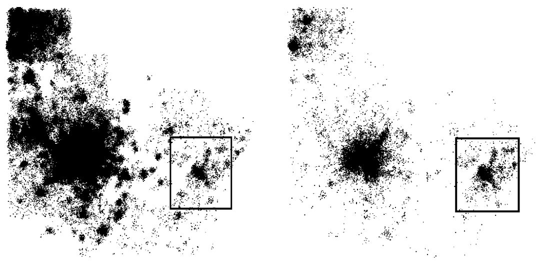

Figure 2: The left picture shows the "population" distribution within Western PA and the right picture shows the "counts" distribution. The winning region is shown as a square.

5 Conclusions

Thus we have presented a fast overlap-multires partitioning algorithm for detection of spatial overdensities, and demonstrated that this method results in significant (10-200x) speedups on both real and artificially generated datasets. Our current focus is application of this algorithm to the real-time monitoring and detection of disease outbreaks, based on national-level hospital and pharmacy data. We are currently examining how to detect statistically significant indications of a disease outbreak based on changes in the spatial clustering of disease cases; this is similar to, but distinct from, the space-time statistic of Kulldorff (2001), since Kulldorff's method detects clusters which persist over time rather than "new" outbreaks. Application of a fast partitioning method based on the techniques presented here may allow us to achieve the difficult goal of automatic real-time detection of disease outbreaks.

References

• R. Agrawal, et al. 1998. Automatic subspace clustering of high dimensional data for data mining applications. *Proc. ACM-SIGMOD Intl. Conference on Management of Data*, 94-105.

• K. Deng and A. W. Moore. 1995. Multiresolution instance-based learning. *Proc. 12th Intl. Joint Conference on Artificial Intelligence*, 1233-1239.

• J. Friedman and N. Fisher. 1999. Bump hunting in high-dimensional data. *Statistics and Computing* **9**(2), 1-20.

• S. Goil, et al. 1999. MAFIA: efficient and scalable subspace clustering for very large data sets. *Northwestern University, Technical Report No. CPDC-TR-9906-010.*

• M. Kulldorff. 1997. A spatial scan statistic. *Communications in Statistics: Theory and Methods* **26**(6), 1481-1496.

• M. Kulldorff. 1999. Spatial scan statistics: models, calculations, and applications. In Glaz and Balakrishnan, eds. *Scan Statistics and Applications.* Birkhauser, Boston, MA, 303-322.

• M. Kulldorff. 2001. Prospective time periodic geographical disease surveillance using a scan statistic. *Journal of the Royal Statistical Society A* **164**(1), 61-72.

• M. Kulldorff and N. Nagarwalla. 1995. Spatial disease clusters: detection and inference. *Statistics in Medicine* **14**, 799-810.

• A. W. Moore and M. S. Lee. 1998. Cached sufficient statistics for efficient machine learning with large datasets. *J. Artificial Intelligence Research* **8**, 67-91.

Table 1: Performance of exact algorithm

method	test	time (orig+1000 reps)	regions (orig+1000 reps)	speedup
naive	$N = 64$	$0:00.28 + 4:39$	$8.9 \times 10^4 + 8.9 \times 10^7$	x1
exact	$N = 64$, $r = 30$	$0:00.80 + 0:18$	$1.2 \times 10^4 + 1.8 \times 10^3$	x15
exact	$N = 64$, $r = 20$	$0:00.96 + 0:18$	$1.5 \times 10^4 + 4.6 \times 10^4$	x15
exact	$N = 64$, $r = 10$	$0:02 + 1:18$	$3.2 \times 10^4 + 1.3 \times 10^6$	x3.5
exact	$N = 64$, $r = 5$	$0:03 + 9:22$	$4.6 \times 10^4 + 1.0 \times 10^7$	x0.49
exact	$N = 64$, $r = 2$	$0:04 + 41:40$	$6.3 \times 10^4 + 4.3 \times 10^7$	x0.11
naive	$N = 128$	$0:02 + 38:50$	$7.1 \times 10^5 + 7.1 \times 10^8$	x1
exact	$N = 128$, $r = 30$	$0:07 + 1:14$	$1.2 \times 10^5 + 3.2 \times 10^4$	x29
exact	$N = 128$, $r = 20$	$0:09 + 1:22$	$1.6 \times 10^5 + 4.0 \times 10^5$	x26
exact	$N = 128$, $r = 10$	$0:15 + 14:54$	$2.6 \times 10^5 + 1.5 \times 10^7$	x2.6
exact	$N = 128$, $r = 5$	$0:20 + 1:33:36$	$3.4 \times 10^5 + 9.8 \times 10^7$	x0.41
exact	$N = 128$, $r = 2$	$0:28 + 5:54:19$	$4.7 \times 10^5 + 3.4 \times 10^8$	x0.11
naive	$N = 256$	$0:20 + 5:37:03$	$5.6 \times 10^6 + 5.6 \times 10^9$	x1
exact	$N = 256$, $r = 30$	$0:58 + 4:38$	$9.6 \times 10^5 + 3.6 \times 10^5$	x60
exact	$N = 256$, $r = 20$	$1:16 + 10:08$	$1.3 \times 10^6 + 6.3 \times 10^6$	x30
exact	$N = 256$, $r = 10$	$1:51 + 3:10:30$	$1.9 \times 10^6 + 1.9 \times 10^8$	x1.8
exact	$N = 256$, $r = 5$	$2:35 + 14:30:50$	$2.7 \times 10^6 + 8.2 \times 10^8$	x0.39
exact	$N = 256$, $r = 2$	$3:50 + 51:24:00$	$3.6 \times 10^6 + 2.8 \times 10^9$	x0.11

- S. Openshaw, et al. 1988. Investigation of leukemia clusters by use of a geographical analysis machine. *Lancet* **1**, 272-273.
- F. P. Preparata and M. I. Shamos. 1985. *Computational Geometry: An Introduction.* Springer-Verlag, New York, NY.
- H. Samet. 1990. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, Reading, MA.
- L. A. Waller, et al. 1994. Spatial pattern analysis to detect disease clusters. In Nicholas Lange, et al, eds. *Case Studies in Biometry.* Wiley, New York, NY, 3-23.
- W. Wang, et al. 1997. STING: a statistical information grid approach to spatial data mining. *Proc. 23rd Conference on Very Large Databases*, 186-195.

Table 2: Performance of approximate algorithm, $N = 512$

method	test	time (orig+1000 reps)	regions (orig+1000 reps)	speedup
naive	all	$2:37+43:36:40$	$4.5\times10^{7}+4.5\times10^{10}$	x1
$b=0$	std, $k=16$	$0:18+16:35$	$6.6\times10^{3}+1.1\times10^{4}$	x155
$b=1$	std, $k=16$	$0:17+17:05$	$7.5\times10^{3}+2.1\times10^{4}$	x151
$b=2$	std, $k=16$	$0:17+16:50$	$8.7\times10^{3}+8.6\times10^{4}$	x153
cps, $b=0$	std, $k=16$	$0:42+16:40$	$6.7\times10^{2}+9.4\times10^{3}$	x151
cps, $b=1$	std, $k=16$	$0:43+16:20$	$7.5\times10^{2}+1.4\times10^{4}$	x154
cps, $b=2$	std, $k=16$	$0:41+17:00$	$8.8\times10^{2}+3.2\times10^{4}$	x148
$b=0$	std, $k=4$	$0:17+17:50$	$2.4\times10^{3}+1.1\times10^{6}$	x145
$b=1$	std, $k=4$	$0:18+1:26:00$	$1.4\times10^{4}+7.0\times10^{7}$	x30
$b=2$	std, $k=4$	$0:25+10:10:30$	$1.5\times10^{5}+5.9\times10^{8}$	x4.3
cps, $b=0$	std, $k=4$	$0:41+17:00$	$1.8\times10^{3}+4.0\times10^{5}$	x148
cps, $b=1$	std, $k=4$	$0:41+29:10$	$7.5\times10^{3}+1.5\times10^{7}$	x88
cps, $b=2$	std, $k=4$	$0:42+1:13:00$	$2.5\times10^{4}+6.7\times10^{7}$	x36
$b=0$	city, $k=16$	$0:19+18:55$	$6.6\times10^{3}+1.5\times10^{4}$	x136
$b=1$	city, $k=16$	$0:19+19:35$	$7.8\times10^{3}+3.7\times10^{4}$	x132
$b=2$	city, $k=16$	$0:18+18:40$	$1.0\times10^{4}+1.2\times10^{5}$	x138
cps, $b=0$	city, $k=16$	$0:42+16:30$	$9.0\times10^{2}+1.2\times10^{4}$	x153
cps, $b=1$	city, $k=16$	$0:46+20:40$	$9.8\times10^{2}+2.1\times10^{4}$	x122
cps, $b=2$	city, $k=16$	$0:41+18:40$	$1.3\times10^{3}+5.3\times10^{4}$	x135
$b=0$	city, $k=4$	$0:17+28:00$	$1.7\times10^{4}+5.5\times10^{7}$	x93
$b=1$	city, $k=4$	$0:19+7:14:40$	$3.4\times10^{4}+4.0\times10^{8}$	x6.0
$b=2$	city, $k=4$	$0:27+28:19:00$	$1.7\times10^{5}+1.5\times10^{9}$	x1.5
cps, $b=0$	city, $k=4$	$0:43+24:30$	$4.3\times10^{3}+3.3\times10^{7}$	x104
cps, $b=1$	city, $k=4$	$0:44+2:11:00$	$1.9\times10^{4}+1.2\times10^{8}$	x20
cps, $b=2$	city, $k=4$	$0:47+7:06:50$	$1.3\times10^{5}+4.5\times10^{8}$	x6.1
$b=0$	high-σ, $k=16$	$1:29+28:00$	$1.2\times10^{6}+1.1\times10^{7}$	x89
$b=1$	high-σ, $k=16$	$2:22+2:54:20$	$2.0\times10^{6}+1.0\times10^{8}$	x15
$b=2$	high-σ, $k=16$	$4:22+12:33:50$	$3.8\times10^{6}+6.9\times10^{8}$	x3.5
cps, $b=0$	high-σ, $k=16$	$0:41+17:00$	$8.1\times10^{2}+9.8\times10^{3}$	x148
cps, $b=1$	high-σ, $k=16$	$0:41+16:40$	$9.8\times10^{2}+1.5\times10^{4}$	x151
cps, $b=2$	high-σ, $k=16$	$0:41+17:00$	$1.8\times10^{3}+4.6\times10^{4}$	x148
cps, $b=0$	high-σ, $k=4$	$0:44+17:15$	$3.2\times10^{3}+5.5\times10^{6}$	x146
cps, $b=1$	high-σ, $k=4$	$0:45+34:10$	$6.2\times10^{4}+1.9\times10^{7}$	x75
cps, $b=2$	high-σ, $k=4$	$1:08+3:20:00$	$5.2\times10^{5}+2.1\times10^{8}$	x13

Table 3: Accuracy of approximate algorithm

method	test	accuracy ($k = 16$)	accuracy ($k = 4$)
$b = 0$	standard	96%	52%
$b = 0$	city	98%	36%
$b = 0$	high-σ	98%	46%
$b = 1$	standard	100%	90%
$b = 1$	city	100%	88%
$b = 1$	high-σ	100%	90%
$b = 2$	standard	100%	98%
$b = 2$	city	100%	98%
$b = 2$	high-σ	100%	100%

Table 4: Performance on Emergency Dept. dataset

method	time (orig+1000 reps)	regions (orig+1000 reps)	speedup
naive	$4:05 + 65:50:00$	$4.5 \times 10^7 + 4.5 \times 10^{10}$	x1
$b = 0$	$4:20 + 14:36$	$3.5 \times 10^6 + 7.0 \times 10^6$	x209
$b = 1$	$4:22 + 42:20$	$3.5 \times 10^6 + 3.6 \times 10^7$	x85
$b = 2$	$4:36 + 2:01:12$	$3.8 \times 10^6 + 1.1 \times 10^8$	x31