Chapter 3
Region-Based Geospatial Abduction

Abstract Given a set $\mathscr{O}$ of observations, in the previous chapter, we developed a set of methods to find sets $\mathscr{E}$ of explanations. However, these explanations consisted of points. When the geospatial resolution of the space $\mathscr{S}$ is small, the non-determinism in our point-based geospatial abduction algorithms is negligible—making many points more or less "equivalent" as far as being potential partner locations. As such, users might want to get regions back as output to their geospatial abduction queries. Moreover, users might want to reason about real-valued points rather than points that are integer-valued. In this chapter, we develop the theory and algorithms required for reasoning in the real-valued domain with regions being returned to the user rather than points.

3.1 Introduction

In Chapter 2, we developed a theory of geospatial abduction in which a set of points was returned as an answer (or explanation) to the user. For instance, in our IED cache detection problem, we returned a set of points consisting of potential locations of IED caches in Baghdad. In our tiger detection problem, we returned sets of points where a tiger might dwell, given the locations of various kills attributed to the tiger. In the same vein, in our virus host detection problem, we returned sets of points where the host of a virus causing a disease such as monkey-pox might conceivably reside. And in our burglar detection problem, the explanations generated by point-based geospatial abduction identified explanations consisting of points where a burglar might reside (*e.g.*, his house, his office, his significant other's house, etc.).

However, when our space $\mathscr{S}$ has a fine-grained resolution, many points might be more or less "equivalent" as far as being potential IED cache locations is concerned. As a consequence, point-based geospatial abduction yields many potential points that could be included in an explanation, and sometimes, preferring one point to another is merely a matter of non-deterministic choice, rather than rational preference of one point over another.

In this chapter, we try to return explanations for geospatial abduction that consist of sets of *regions* rather than points so that such non-determinism can be significantly reduced. Thus, our definition of an explanation in this chapter returns a set or regions. Each region in an explanation says a potential IED weapons cache (or a tiger dwelling, or a region supporting a virus host, or a set of locations corresponding to a burglar's residence) might be somewhere (*i.e.*, at any point) within the region.

In addition, in this chapter, we focus on the real-valued domain. While most real-world GIS systems only use integer-valued coordinates, real-valued coordinates are interesting both from a theoretical perspective and often from the perspective of better computation—for example, solving linear constraints over the continuous, real-valued domain is polynomial, while solving the same linear constraints over the domain of the integers is well known to be NP-hard.

3.2 Technical Preliminaries

To address the problem of region-based geospatial abduction, we introduce a framework that resembles that of Chapter 2—but differs in several important aspects. These include the use of a continuous space and multiple types of explanations. In Chapter 4, we return to the original framework of Chapter 2.

Unlike the previous chapter, we assume the existence of a real-valued $M \times N$ space $\mathcal{S}$ whose elements are pairs of *real* numbers (rather than integers) from the set $[0,M] \times [0,N]$. An observation is any member of $\mathcal{S}$—thus, unlike the preceding chapter, observations are pairs of real values. We use $\mathcal{O}$ to denote an arbitrary, but fixed, finite set of *observations*. We assume there are real numbers $\alpha \leq \beta$ such that for each observation o , there exists a partner p_o (to be found) whose distance from o is in the interval $[\alpha, \beta]$.[1] Without loss of generality, we also assume that all elements of $\mathcal{O}$ are over β distance away from the edge of $\mathcal{S}$. Example 3.1 presents a neighborhood as a space and locations tiger dwellings.

Example 3.1 (Tiger Example). A tiger in the Achanakamar Wildlife Sanctuary (AMWLS) has made many kills. Suppose the AMWLS sanctuary is the space $\mathcal{S}$ depicted in Figure 3.1. Tiger kills were found by wildlife rangers at points $\mathcal{O} = \{o_1, \ldots, o_{13}\}$. Tiger conservation experts, on the basis of historical data, suggest that favored tiger dwellings are located within 5km of these kills (*i.e.*, $\alpha = 0$ and $\beta = 5$km). Note that in Figure 3.1, circles of radius 5km are drawn around the observation points. The tiger conservation experts are interested in the locations of such dwellings.

Throughout this chapter, we assume the notion of a *distance function* d on $\mathcal{S}$ satisfying the usual properties of such distance functions introduced in Chapter 2. The methods used in this chapter apply to *any notion of distance between two points as long as the three distance axioms described in Chapter 2 are satisfied.*

[1] Chapter 2 describes methods to learn α, β automatically from historical data.

We now define a region and how they relate to the set of observations. Our intuition is simple—a region *explains* an observation if that region contains a partner point for that observation.

Definition 3.1 (Region / Super-Explanation / Sub-Explanation). A *region* r is a subset of $\mathcal{S}$ such that for any two points $(x,y),(x',y') \in r$, there is sequence a of line segments from (x,y) to (x',y') s.t. no line segment lies outside r.

1. A region r **super-explains** point o in $\mathcal{S}$ iff there exists a point $p \in r$ such that $d(o,p) \in [\alpha,\beta]$.
2. A region r **sub-explains** some point o in $\mathcal{S}$ iff $(\forall p \in r)\ d(o,p) \in [\alpha,\beta]$.

Thus, intuitively, a region r as defined above is connected in the sense that one can travel from any point in a region to any other point in the region without leaving the region r. In addition, regions can have any shape and may overlap.

Informally speaking, region r super-explains an observation o if and only if there is at least one partner in region r for the observation o. On the other hand, region r sub-explains an observation o if and only if every point in the region explains observation o. Throughout this chapter, we assume that checking if some point o is sub-explained (super-explained) by region r can be performed in constant (*i.e.*, $O(1)$) time. This is a reasonable assumption for most regular shaped regions like circles, ellipses and polygons. The following result follows immediately from Definition 3.1.

Observation 3.2.1 *If region $r \neq \emptyset$ sub-explains point o, then r super-explains point o.*

This observation follows immediately from the definitions. If r sub-explains point o then the distance of every point in r from observation o lies within the interval $[\alpha,\beta]$. Thus, as long as r is non-empty, at least one point in r is at a distance d_0 from the observation o where $d_o \in [\alpha,\beta]$.

We would like to explain observations by finding regions containing a partner. In some applications, the user may be able to easily search the entire region—hence a super-explaining region would suffice. In other applications, we may want to be sure that any point within the region can be a partner as not to waste resources—so only a sub-explanation would make sense in such a case. Often, these situations may depend on the size of the regions. We shall discuss the issue of restricting region size later in this section. For now, we shall consider regions of any shape or size. Example 3.2 shows regions that super- or sub-explain various observations.

Example 3.2. Consider the scenario from Example 3.1 and the regions $R = \{r_a, r_b, r_c, r_d, r_e, r_f, r_g\}$ shown in Figure 3.1. Suppose these regions correspond with feasible regions for the tiger to live in—*i.e.*, places that have the right amount of ground cover and the right amount of prey for a tiger to consider this to be a good habitat. Consider region r_a. As it totally lies within the α,β distance of o_1, it both sub-explains and super-explains this observation. Conversely, region r_d super-explains both o_6 and o_7 but sub-explains neither.

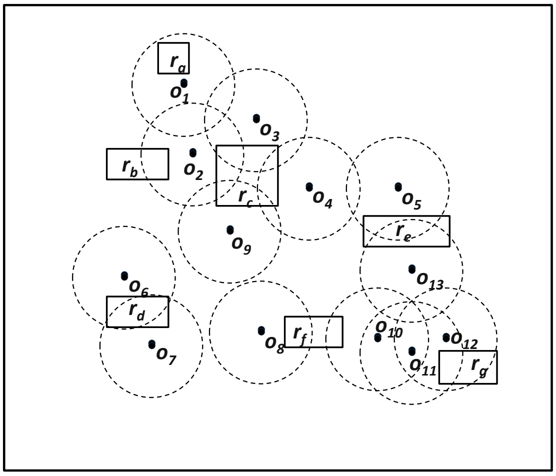

Fig. 3.1 Locations of tiger kills and feasible locations $\{r_a, r_b, r_c, r_d, r_e, r_f, r_g\}$ where the tiger can potentially dwell. The β distance for each observation is shown with a dashed circle.

This chapter studies following decision problems.

Sub-(Super-) Region Explanation Problem (Sub/Sup-REP)
INPUT: A space $\mathcal{S}$, distance interval $[\alpha, \beta]$, set $\mathcal{O}$ of observations, set R of regions, and natural number $k \in [1, |\mathcal{O}|]$.
OUTPUT: Set $R' \subseteq R$, where $|R'| \leq k$ and for each $o \in \mathcal{O}$, there is an $r \in R$ such that r sub-(super-) explains o.

The Sub-(Super) Region Explanation Problem asks us to find all sub-explanations (resp. super-explanations) R of size k or fewer where size is defined as the number of regions in R which sub-explain (resp. or super-explain) our set of observations $\mathcal{O}$.

The fact that a set R of regions is part of the input is not an assumption, but a feature. A user might set R to be all the regions associated with $\mathcal{S}$ in which case he is really making no assumption at all. Alternatively, he might use his knowledge of the application (*e.g.*, IED cache locations or tiger hangouts or virus host information or burglary-related information) to define regions, taking into account, the terrain and/or known aspects of the population living in the area of interest. For instance, when trying to identify regions containing IED caches in Baghdad used for attacks by Shi'ite groups, he might define regions to be places that are not predominantly Sunni and that do not contain US bases or bodies of water. On the other hand, in the tiger detection application, he might define regions to be places where the tiger has ample ground cover and ample amount of prey to hunt. In the virus host detection problem, he might decide based on his knowledge of biology and his knowledge of the geography of the terrain, that certain regions are feasible locations for the virus host, while others are not. And finally, the St. Paul, MN, police detective might use knowledge of the criminal to decide that the criminal could not live in certain areas (*e.g.*, there was a police chase not known to the geospatial abduction system where the perpetrator disappear in a reasonably narrow region, allowing the detective to eliminate other regions from consideration). Other kinds of logical conditions may be used when dealing with burglaries or drug trafficking.

Thus, the set R of regions allows an analyst to specify any knowledge he has, and allows the system to benefit from that knowledge. In short, the set R is similar to the feasibility predicate in Chapter 2 by saying that only regions in R can be returned as part of the answer by the region-based geospatial abduction system. If no such knowledge is available, R can be taken to be the set of all regions associated with $\mathscr{S}$, and thus, allowing the user to specify R as part of the input leads to no loss of generality; moreover, it allows the user greater flexibility in specifying where the regions he is looking for could possibly be. R can also be used to restrict the size of the region (*e.g.*, only considering regions whose area is less than 5 sq. km.).

There are two different associated optimization problems associated with both the Sub-REP and Sup-REP problems. The first deals with finding a subset of regions of minimal cardinality that explains all observations.

Sub-(Super-)Region Explanation Problem-Minimum Cardinality (Sub/Sup-REP-MC)
INPUT: A space, $\mathscr{S}$, distance interval $[\alpha,\beta]$, set of observations $\mathscr{O}$, and set of regions R.
OUTPUT: Set $R' \subseteq R$ of minimum cardinality, where for each $o \in \mathscr{O}$, there is an $r \in R$ s.t. r sub-(super-) explains o.

The Sub/Sup-REP-MC problems therefore support the principles of Occam's razor, long present in research on abduction[3, 6]. Only a minimal-sized set of regions can be returned—no more regions than strictly necessary should be returned.

Our second optimization problem fixes the number of regions returned in the solution, but maximizes the number of observations that are explained.

Sub-(Super-)Region Explanation Problem-Maximum Explaining (Sub/Sup-REP-ME)
INPUT: Given a space $\mathscr{S}$, distance interval $[\alpha,\beta]$, set $\mathscr{O}$ of observations, set R of regions, and natural number $k \in [1,|\mathscr{O}|]$.
OUTPUT: Set $R' \subseteq R$, where $|R'| \leq k$ such that the number of $o \in \mathscr{O}$ where there is an $r \in R$ s.t. r sub-(super-) explains o is maximized.

Sub-(Super-)Region Explanation Problem-Maximum Explaining (Sub/Sup-REP-ME) problems are similar in spirit to the k-**SEP** problem by requiring that no more than k regions be returned as the answer by the geospatial abduction system in response to a user request. Consider the following example.

Example 3.3. Consider the scenario from Example 3.2. Consider an instance of Sup-REP with $k = 7$. The set $\{r_a, r_b, r_c, r_d, r_e, r_f, r_g\}$ is a solution to this problem. Now consider Sup-REP-MC: the set $\{r_a, r_c, r_d, r_e, r_f, r_g\}$ is a solution to this problem. Finally, consider Sup-REP-ME with $k = 2$. The set $\{r_c, r_d\}$ is a solution to this problem.

We now consider a special case of these problems that arises when the set R of regions is created by a partition of the space based on the set of observations ($\mathcal{O}$) and concentric circles of radii α and β drawn around each $o \in \mathcal{O}$. We can associate regions in such a case with subsets of $\mathcal{O}$. For a given subset $\mathcal{O}'$, we say that there is an associated set of *induced regions* (denoted $R_{\mathcal{O}'}$), defined as follows:

$$R_{\mathcal{O}'} = \{\{x | \ \forall o \in \mathcal{O}', d(x,o) \in [\alpha,\beta] \wedge \\ \forall o' \notin \mathcal{O}', d(x,o') \notin [\alpha,\beta]\} \ \}$$

We note that for a given subset of observations, it is possible to have a set of induced regions, $R_{\mathcal{O}'}$ that has more than one element. For example, consider set $R_{\emptyset} = \{r_1, r_{12}\}$ in Figure 3.2. For a given set of observations $\mathcal{O}$, we will use the notation $R_{\mathcal{O}}$ do denote the set of all induced regions. Formally:

$$R_{\mathcal{O}} = \bigcup_{\substack{\mathcal{O}' \in 2^{\mathcal{O}} \\ R_{\mathcal{O}'} \neq \emptyset}} R_{\mathcal{O}'}$$

We illustrate the idea of induced regions in the following example.

Example 3.4. In order to identify where the tiger resides, tiger conservation experts may create 33 **induced** regions in $\mathcal{S}$ by drawing circles of 5km radius around all observations (see Figure 3.2), the set of which is denoted $R_{\mathcal{O}} = \{r_1, \ldots, r_{33}\}$.

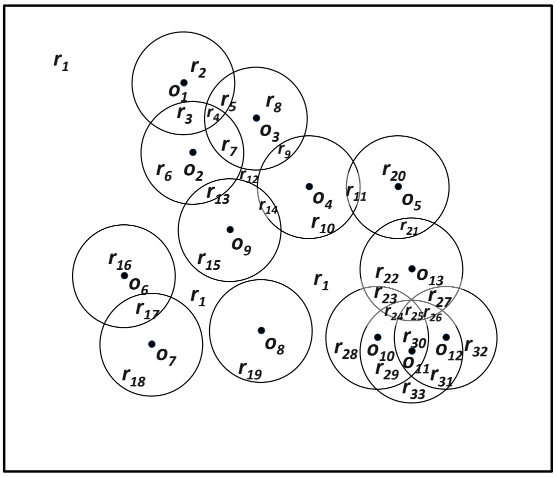

Fig. 3.2 Space $\mathcal{S}$ and the regions in set $R_{\mathcal{O}}$.

For the special case where $R_{\mathcal{O}}$ is the set of all possible regions of $\mathcal{S}$, we have the following result.

Lemma 3.1. *Suppose $\mathcal{O}$ is a set of observations and $R_{\mathcal{O}}$ is the set of induced regions. A region $r \in R_{\mathcal{O}}$ sub-explains an observation $o \in \mathcal{O}$ if and only if it super-explains o.*

Proof. CLAIM 1: Any point in a region $r \in R_{\mathcal{O}}$ is either within distance $[\alpha, \beta]$ or outside the distance $[\alpha, \beta]$ from each $o \in \mathcal{O}$.
As $R_{\mathcal{O}}$ is created by drawing circles of radii α, β around each observation, the statement follows by the definition of $R_{\mathcal{O}}$.

CLAIM 2: ($\Leftarrow$) There is no $r \in R_{\mathcal{O}}$ that super-explains some $o \in \mathcal{O}$ but does not sub-explain the observation.
Suppose, by way of contradiction, there is some $r \in R_{\mathcal{O}}$ that super-explains some $o \in \mathcal{O}$ but does not sub-explain it. Then, there must be at least one point in r that can be partnered with $\mathcal{O}$ and at least one point in r that cannot be partnered with o. However, by Claim 1, this is not possible, hence a contradiction.
CLAIM 3: ($\Rightarrow$) There is no $r \in R_{\mathcal{O}}$ that sub-explains some $o \in \mathcal{O}$ but does not super-explain the observation.
Follows directly from Observation 3.2.1.

By this result, for the special case of induced regions, we only need one decision problem.

Induced Region Explanation Problem (I-REP)
INPUT: Given a space, $\mathcal{S}$, distance interval $[\alpha, \beta]$, set $\mathcal{O}$ of observations, and natural number $k \in [1, |\mathcal{O}|]$.
OUTPUT: Set $R' \subseteq R_{\mathcal{O}}$, where $|R'| \leq k$ and for each $o \in \mathcal{O}$, there is an $r \in R$ s.t. r sub-explains o.

As mentioned earlier, the sizes of regions can be regulated by our choice of R. However, we may also explicitly require that all regions must be less than a certain area. Consider the following variant of Sup-REP.

Area-Constrained Super-Region Explanation Problem (AC-Sup-REP)
INPUT: Given a space, $\mathcal{S}$, distance interval $[\alpha, \beta]$, set $\mathcal{O}$ of observations, set R of regions, area A, and natural number $k \in [1, |\mathcal{O}|]$.
OUTPUT: Set $R' \subseteq R$, where $|R'| \leq k$ and each $r \in R'$ has an area $\leq A$ and for each $o \in \mathcal{O}$, there is an $r \in R$ such that r super-explains o.

The following proposition tells us that AC-Sup-REP is at least as hard as I-REP, yet no harder than Sup-REP (an analogous result can easily be shown for an area-constrained version of Sub-REP). We note that essentially, we eliminate the regions whose area is above area A, which gives us an instance of Sup-REP. To go the other direction, we directly encode I-REP into an instance of AC-Sup-REP and have A be larger than the area of any region.

Theorem 3.1. *I-REP is polynomially reducible to AC-Sup-REP.*
AC-Sup-REP is polynomially reducible to Sup-REP.

Proof. CLAIM 1: I-REP $\leq_p$ AC-Sup-REP.
Set up an instance of AC-Sup-REP with the input for I-REP plus the parameter $A = \pi \cdot (\beta^2 - \alpha^2)$. For direction $\Leftarrow$, note that a solution to this instance of I-REP is also a solution to AC-Sup-REP, as any region that sub-explain an observation also super-explains it for the set of region $R_{\mathscr{O}}$ (Lemma 3.1) and the fact that, by definition, all regions in the set $R_{\mathscr{O}}$ must have an area less than A. For direction $\Rightarrow$, we know that only regions that can be partnered with observations are considered by the area restriction, and by Lemma 3.1, all regions in the solution are also super-explanations for their corresponding observation.

CLAIM 2: AC-Sup-REP $\leq_p$ Sup-REP.
Consider the set R from AC-Sup-REP and let set $R' = \{r \in R | \textit{ the area of } r \leq A\}$. Set up an instance of Sup-REP where the set of regions is R' and the rest is the input from AC-Sup-REP. For direction $\Leftarrow$, it is obvious that any solution to AC-Sup-REP is also a solution to Sup-REP, as $R - R'$ are all regions that cannot possibly be in the solution to the instance of AC-Sup-REP. Going the other direction ($\Rightarrow$), we observe that by the definition of R', all regions in the result of the instance of Sup-REP meet all the requirements of the AC-Sup-REP problem.

In the final observation of this section, we note that the set $R_{\mathscr{O}}$ can be used as a "starting point" in determining regions. For instance, supplemental information on areas that may be restricted from being partnered with an observation may also be considered and reduce the area of (or eliminate altogether) some regions in the set. Consider the following example.

Example 3.5. Consider the tiger scenario from Example 3.4. Tiger conservation experts may eliminate an open meadow in the area and certain other areas with small amounts of prey from their search. These "restricted areas" are depicted in Figure 3.3. Note that several regions from Figure 3.2 are either eliminated or have decreased in size. However, by eliminating these areas, tiger conservation experts have also pruned some possibilities from their search. For example, regions r_9, r_{13} were totally eliminated from consideration.

3.3 Complexity

In this section, we study the computational complexity of problems related to region-based geospatial abduction. In particular, we show that Sub-REP, Sup-REP, and I-REP are NP-Complete and that the associated optimization problems are NP-Hard. We also show that the optimization problems Sub-REP-MC, Sup-REP-MC, and I-REP-MC cannot be approximated by a fully polynomial-time approximation scheme (FPTAS) unless $P = NP$. In particular, this means that there are no

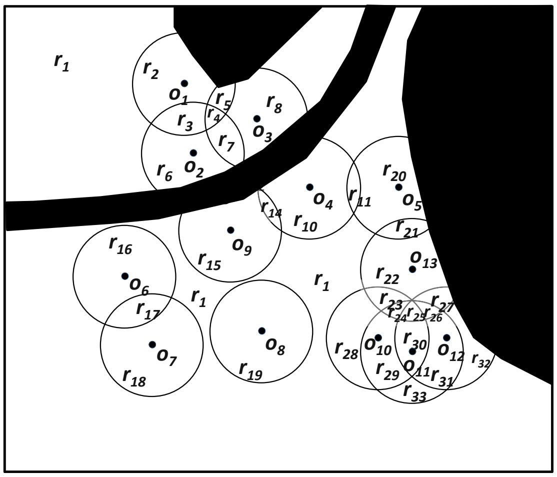

Fig. 3.3 A set of regions in $\mathcal{S}$ created based on the distance $\beta = 5$km as well as restricted areas (shown in black).

polynomial-time algorithms to approximate these problems with guarantees of approximation unless $P = NP$ (the latter, of course, is a central unsolved problem in computer science and it is widely believed that in fact $P \neq NP$). We also note that the complexity of the area-constrained versions of these problems follows directly from the results of this section by the reduction of Theorem 3.1 (page 64).

We first prove that I-REP is NP-complete, which then allows us to correctly identify the complexity classes of the other problems by leveraging Lemma 3.1. First, we introduce the problem of "circle covering" (CC) that was proven to be NP-complete in [11].

Circle Covering (CC)
INPUT: A space $\mathcal{S}'$, set P of points, real number β', natural number k'.
OUTPUT: "Yes" if there is a set of points, Q in $\mathcal{S}'$ such that all points in P are covered by discs centered on points in Q of radius β' where $|Q| \leq k'$—"no" otherwise.

The theorem below establishes that I-REP is NP-complete.

Theorem 3.2. *I-REP is NP-Complete.*

Proof. CLAIM 1: I-REP is in-NP.
Given a set of regions, $R' \subseteq R_{\mathcal{O}}$ we can easily check in polynomial time that for each $o \in \mathcal{O}$ there is an $r \in R$ that is a partner for o. Simply check if each r falls within the distance $[\alpha, \beta]$ for a given $o \in \mathcal{O}$. The operation will take time $O(|\mathcal{O}| \cdot |R'|)$—which is polynomial.

CLAIM 2: I-REP is strongly NP-hard.
We show that for an instance of the known strongly NP-complete problem, circle covering (CC), $CC \leq_p I - REP$ by the following transformation.

- Set $\mathscr{S} = \mathscr{S}'$
- Set $\mathscr{O} = P$
- Set $\beta = \beta'$
- Set $\alpha = 0$
- Set $k = k'$

This transformation obviously takes polynomial time. We prove correctness with the following two sub-claims.

CLAIM 2.1: If there is a k-sized solution R' for I-REP, then there is a corresponding k'-sized solution for CC.
Consider some $r \in R'$. Let $\mathscr{O}'$ be the subset of $\mathscr{O}$ (also of P) such that all points in $\mathscr{O}'$ are partnered with r. By definition, all points enclosed by r are of distance β or less away from each point in $\mathscr{O}'$. Hence, we can pick some point enclosed by r and we have the center of a circle that covers all elements in $\mathscr{O}'$. The statement follows.

CLAIM 2.2: If there is a k'-sized solution Q for CC, then there is a corresponding k-sized set solution for I-REP.
Consider some point $q \in Q$. Let P' be the subset of P (also of $\mathscr{O}$) such that all points in P' are of distance β' from q. As p is within β of an element of $\mathscr{O}$, it is in some region of the set $R_{\mathscr{O}}$. Hence, the region that contains p is a partner region for all elements of P'. The statement follows.

Further, as the optimization version of circle covering is known to have no FPTAS unless $P = NP$ [18], by the nature of the construction in Theorem 3.2, we can be assured of the same result for I-REP-MC.

Corollary 3.1. *I-REP-MC cannot be approximated by a fully polynomial-time approximation scheme (FPTAS) unless $P = NP$.*

Proof. Follows directly from [11] and Theorem 3.2.

So, from the above Theorem and Corollary and Lemma 3.1, we get the following results:

Corollary 3.2. *1. Sub-REP and Sup-REP are NP-Complete.*
2. Sub-REP-MC, Sup-REP-MC, I-REP-MC, Sub-REP-ME, Sup-REP-ME, and I-REP-ME are NP-Hard.
3. Sub-REP-MC, Sup-REP-MC cannot be approximated by a FPTAS unless $P = NP$.

Proof. All follow directly from Lemma 3.1, Theorem 3.2, and Corollary 3.1.

3.4 Algorithms

In this section we devise algorithms to address the optimization problems associated with Sup-REP, Sub-REP, and I-REP. First, we show that these optimization problems reduce to either instances of set-cover (for Sub/Sup-REP-MC) or max-k-cover (for Sub/Sup-REP-ME). These problems are well-studied and there are algorithms that provide exact and approximate solutions. We then provide a new greedy-algorithm for Sub/Sup-REP-MC that also provides an approximation guarantee. This is followed by a discussion of approximation for I-REP-ME for the case where $\alpha = 0$. Finally, we discuss some practical issues dealing with implementation.

3.4.1 Exact and Approximate Solutions by Reduction

In this section we show that the -MC problems introduced earlier in this chapter can be reduced to set-cover and that the -ME problems can reduce to the well-known max-k-cover problem. As these problems have been extensively studied in the core computer science algorithms community, they offer the potential to solve the various region-based geospatial abduction problems introduced earlier in this chapter. Set cover has already been introduced earlier on in Chapter 2. We now present max-k-cover [7], which is often regarded as the dual of set-cover.

Max-k-Cover
INPUT: Set of elements S, family of subsets of S, $\mathscr{H} = H_1, \ldots, H_m$, natural number $k \leq |S|$.
OUTPUT: Subset $\mathscr{H}' \subseteq \mathscr{H}$ s.t. $|\mathscr{H}'| \leq k$ where $|\bigcup_{H_i \in \mathscr{H}'} H_i \cap S|$ is maximized.

The key to showing that Sub/Sup-REP optimization problems can reduce to one of these problems is to determine the family of subsets. We accomplish this as follows: for each region $r \in R$, we find the subset of $\mathscr{O}$ that can be partnered with r. We shall refer to this set as $\mathscr{O}_r$. This gives us the following algorithm for the optimization problems (we simply omit the k parameter for the -MC problems that reduce to Set-Cover):

```
REDUCE-TO-COVERING($\mathscr{O}$ set of observations, R set of regions, k natural number) returns
instance of covering problem $\langle S, \mathscr{H}, k \rangle$

1. For each $r \in R$, find $\mathscr{O}_r$ (i.e., $o$ is in $\mathscr{O}_r$ iff $r$ sub/super-explains $o$)
2. Return $\langle \mathscr{O}, \bigcup_{r \in R}\{\mathscr{O}_r\}, k \rangle$
```

This algorithm is the analog of the naive KSEP algorithm introduced in Chapter 2. It essentially says that we must perform the following steps.

- For each feasible region $r \in R$, find all the observations "supported" by R (depending on whether we are interested in sub/super-explanations, this means we

want to find all observations in R that are within a distance of $[\alpha,\beta]$ of at least one point in R or all points in R). These are the sets $\mathcal{O}_r$ for each $r \in R$.
- We then return the union of all these sets $\mathcal{O}_r$.

It is clear that this algorithm is potentially wasteful, returning regions that can be fed as input to a set covering problem because any set that "covers" all the sets $\mathcal{O}_r$ thus computed yields a potential region that explains the observations. The following result describes the complexity of this algorithm.

Proposition 3.1. *REDUCE-TO-COVERING requires $O(|\mathcal{O}| \cdot |R|)$ time.*

Proof. Follows directly from Line 1.

The following theorem shows that REDUCE-TO-COVERING correctly reduces a Sub/Sup-REP optimization problem to set-cover or max-k-cover as appropriate.

Theorem 3.3. *Sub/Sup-REP-MC polynomially reduces to Set-Cover and Sub/Sup-REP-ME polynomially reduces to Max-k-Cover.*

Proof. CLAIM 1: Sub/Sup-REP-MC $\leq_p$ Set-Cover
Consider the instance of set-cover $\langle \mathcal{O}, \bigcup_{r\in R}\{\mathcal{O}_r\}\rangle$ obtained from REDUCE-TO-COVERING$(\mathcal{O},R)$.
Let $\mathcal{H}'$ be a solution to this instance of set-cover. ($\Leftarrow$) If R' is a solution to the instance of Sub/Sup-REP-MC, then the set $\bigcup_{r\in R'}\{\mathcal{O}_r\}$ is a solution to set-cover. Obviously, it must cover all elements of $\mathcal{O}$ and a smaller solution to set-cover would indicate a smaller R'—a contradiction. ($\Rightarrow$) Given set $\mathcal{H}'$, let $R'' = \{r \in R | \mathcal{O}_r \in \mathcal{H}'\}$. Obviously, R'' provides a partner for all observations in $\mathcal{O}$. Further, a smaller solution to Sub/Sup-REP-MC would indicate a smaller $\mathcal{H}'$ is possible—also a contradiction.

CLAIM 2: Sub/Sup-REP-ME $\leq_p$ Max-k-Cover
Consider the instance of max-k-cover $\langle \mathcal{O}, \bigcup_{r\in R}\{\mathcal{O}_r\}, k\rangle$ obtained from REDUCE-TO-COVERING$(\mathcal{O},R,k)$. Let $\mathcal{H}'$ be a solution to this instance of max-k-cover. ($\Leftarrow$) If R' is a solution to the instance of Sub/Sup-REP-ME, then the set $\bigcup_{r\in R'}\{\mathcal{O}_r\}$ is a solution to max-k-cover. Obviously, both have the same cardinality requirement. Further, if there is a solution to max-k-cover that covers more elements in $\mathcal{O}$, this would imply a set of regions that can be partnered with more observations in $\mathcal{O}$—which would be a contradiction. ($\Rightarrow$) Given set $\mathcal{H}'$, let $R'' = \{r \in R | \mathcal{O}_r \in \mathcal{H}'\}$. Obviously, R'' meets the cardinality requirement of k. Furthermore, a solution to Sub/Sup-REP-ME that allows more observations in $\mathcal{O}$ to be partnered with a region would indicate a more optimal solution to max-k-cover—a contradiction.

This result allows us to leverage any exact approach to the above optimization problems to obtain a solution to an optimization problem associated with Sub/Sup-REP. A straightforward algorithm for any of the optimization problems would run in time exponential in $|\mathcal{O}|$ or k and consider every $|\mathcal{O}|$ or k sized subset of $\bigcup_{r\in R}\{\mathcal{O}_r\}$. Clearly, this is not practical for real-world applications.

Fortunately, there are several well-known approximation techniques for both these problems. First, we address the Sub/Sup-REP-ME problems, both of which reduce to Max-k-Cover. As the Max-k-Cover problem reduces to the maximization of a submodular function over uniform matroid[2], we can leverage the greedy approximation algorithm of [12] to solve our problem. We do so below.

Formally, an arbitrary function $f : X \to \mathbb{R}$ from some set X to the reals is submodular if and only if for all $X_1, X_2 \subseteq X$, it is the case that if $x \in X - X_2$, then $f(X_1 \cup \{x\}) - f(X_1) \geq f(X_2 \cup \{x\}) - f(X_2)$. Figure 3.4 explains the notion of submodularity; an easy way to explain such functions is given via an intuitive example. Suppose you have a poor man with very few possessions (X_1) and a rich man with many more possessions (X_2). Suppose neither possesses a Ferrari car (x). Giving the poor man the Ferrari would make a greater difference to his net worth (computed via f as a function of the person's possessions) than giving it to the rich man.

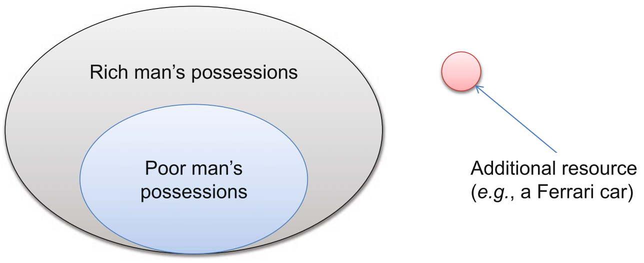

Fig. 3.4 Example of a submodular function. The addition of an expensive vehicle to a rich man's set of possessions would yield a relative increase in net worth far less than the same addition to a poorer man's set of possessions.

```
GREEDY-REP-ME($\mathcal{O}$ set of observations, $R$ set of regions, $k$ natural number) returns $R' \subseteq R$

1. Let $\mathbf{O} = \bigcup_{r \in R} \{\mathcal{O}_r\}$ (obtained by REDUCE-TO-COVERING)
2. Let $\mathcal{O}' = \mathcal{O}$, set $R' = \emptyset$
3. While $k \neq 0$ loop

   a. Let the element $\mathcal{O}_r$ be the member of $\mathbf{O}$ s.t. $|\mathcal{O}_r \cap \mathcal{O}'|$ is maximized.
      $R' = R' \cup r$
      $\mathcal{O}' = \mathcal{O}' - (\mathcal{O}_r \cap \mathcal{O}')$
      $k--$

4. Return $R'$
```

[2] A matroid is a pair (X, I) where X is some set and I is a set of subsets of X (called independent sets) satisfying the following axioms: (i) $\emptyset \in I$, (ii) If $Y \in I$ and $Y' \subseteq Y$, then $Y' \in I$, and (iii) If $Y, Y' \in I$ and $Y' \subset Y$, then there is an element $y \in Y$ such that $(Y' \cup \{y\}) \in I$.

The GREEDY-REP-ME algorithm basically starts by finding all the observations $\mathcal{O}_r$ covered by each region $r \in R$ where R is the set of regions deemed feasible. In order to find a subset of regions of R of cardinality k or less, the algorithm looks at all $\mathcal{O}_r$'s. It initially adds that r into the answer such that $\mathcal{O}_r \cap \mathcal{O}$ is maximized, *i.e.*, in the first iteration of the loop of algorithm GREEDY-REP-ME, it finds an r such that $\mathcal{O}_r$ covers the maximal number of observations in $\mathcal{O}$. In this sense, this algorithm is greedy. This r is added into the solution. As all elements in $\mathcal{O}_r$ have now been "covered" by the insertion of r into the solution, we now only consider elements in $\mathcal{O}' - (\mathcal{O}_r \cap \mathcal{O}')$. The same process is repeated till either $\mathcal{O}'$ is empty or the bound k is reached.

Suppose f denotes the maximum number of observations that can be partnered with a given region. The following result shows an approximation guarantee for our algorithm.

Proposition 3.2. *GREEDY-REP-ME runs in $O(k \cdot |R| \cdot f)$ time and returns a solution such that the number of observations in $\mathcal{O}$ that have a partner region in R' is within a factor $\left(\frac{e}{e-1}\right)$ of optimal.*

Proof. Follows directly from Line 1.

Example 3.6. Consider Example 3.2 (page 59), where the set of regions is $R = \{r_a, r_b, r_c, r_d, r_e, r_f, r_g\}$. Suppose tiger conservationists want to run GREEDY-REP-ME to solve an instance of Sup-REP-ME associated with this situation with $k = 3$. Initially set $\mathcal{O}' = \{o_1, \ldots, o_{13}\}$. On the first iteration of the outer loop, it identifies set $\mathcal{O}_{r_c} = \{o_2, o_3, o_4, o_9\}$ where the cardinality of $\mathcal{O}_{r_c} \cap \mathcal{O}'$ is maximum. Hence, it picks region r_c. The set $\mathcal{O}' = \{o_1, o_5, \ldots, o_8, o_{10}, \ldots o_{13}\}$. On the second iteration, it identifies $\mathcal{O}_{r_e} = \{o_5, o_{13}\}$, which intersected with $\mathcal{O}'$ provides a maximum cardinality, causing r_e to be picked. Set $\mathcal{O}'$ is now $\{o_1, o_6, \ldots, o_8, o_{10}, \ldots, o_{12}\}$. On the last iteration, it identifies $\mathcal{O}_{r_g} = \{o_{11}, o_{12}\}$, again the maximum cardinality when intersected with $\mathcal{O}'$. The element is picked and the solution is r_c, r_e, r_g, and the observations super-explained are $\{o_2, o_3, o_4, o_5, o_9, o_{11}, o_{12}, o_{13}\}$.

Likewise, we can leverage the greedy algorithm for set-cover [26] applied to Sub/Sup-REP-MC. This algorithm is identical to the GREEDY-REP-ME algorithm except in Step (3) where the bound of k is eliminated.

```
GREEDY-REP-MC($\mathcal{O}$ set of observations, $R$ set of regions, ) returns $R' \subseteq R$

1. Let $\mathbf{O} = \bigcup_{r \in R}\{\mathcal{O}_r\}$ (obtained by REDUCE-TO-COVERING)
2. Let $\mathcal{O}' = \mathcal{O}$, set $R' = \emptyset$
3. While $\mathcal{O}' \neq \emptyset$ loop
   a. Let the element $\mathcal{O}_r$ be the member of $\mathbf{O}$ s.t. $|\mathcal{O}_r \cap \mathcal{O}'|$ is maximized.
      $R' = R' \cup r$
      $\mathcal{O}' = \mathcal{O}' - (\mathcal{O}_r \cap \mathcal{O}')$
4. Return $R'$
```

The following result provides approximation guarantees on the solution to the region-based geospatial abduction problem found by the GREEDY-REP-MC algorithm.

Proposition 3.3. *GREEDY-REP-MC runs in* $O(|\mathcal{O}| \cdot |R| \cdot f)$ *time and returns a solution whose cardinality is within a factor of* $1 + \ln(f)$ *of optimal.*

Proof. The outer loop of the algorithm iterates no more than $|\mathcal{O}|$ times, while the inner loop iterates no more than $|R|$ times. The time to compare the number of elements in a set $\mathcal{O}_r$ is $O(f)$.

The approximation factor of $1 + \ln(f)$ follows directly from [26].

Example 3.7. Consider the scenario from Example 3.6. To explain all points where a tiger kill has been observed, tiger conservation experts can create an instance of Sup-REP-MC and use GREEDY-REP-MC. The algorithm proceeds just as GREEDY-REP-ME in the first three steps (as in Example 3.6), but will continue on until all observations are super-explained. So, GREEDY-REP-MC proceeds for three more iterations, selecting r_f ($\mathcal{O}_{r_f} = \{o_8, o_{10}\}$), r_d ($\mathcal{O}_{r_d} = \{o_6, o_7\}$), and finally r_a ($\mathcal{O}_{r_a} = \{o_1\}$). The solution returned is:

$$\{r_c, r_e, r_g, r_f, r_d, r_a\}$$

We now focus on speeding up the set-cover reduction via the GREEDY-REP-MC2 algorithm below.

```
GREEDY-REP-MC2($\mathcal{O}$ set of observations, $R$ set of regions, ) returns $R' \subseteq R$

1. Let $\mathbf{O} = \bigcup_{r \in R}\{\mathcal{O}_r\}$ (obtained by REDUCE-TO-COVERING)
2. For each observation $o \in \mathcal{O}$, let $GRP_o = \{\mathcal{O}_r \in \mathbf{O} | o \in \mathcal{O}_r\}$
3. For each observation $o \in \mathcal{O}$, let $REL_o = \{o' \in \mathcal{O} | o' \in \bigcup_{\mathcal{O}_r \in GRP_o} \mathcal{O}_r\}$ and let $key_o = |REL_o|$
4. Let $\mathcal{O}' = \mathcal{O}$, set $R' = \emptyset$
5. While $\mathcal{O}' \neq \emptyset$ loop
   a. Let $o$ be the element in $\mathcal{O}$ where $key_o$ is minimal.
   b. Let the element $\mathcal{O}_r$ be the member of $GRP_o$ s.t. $|\mathcal{O}_r \cap \mathcal{O}'|$ is maximized.
   c. If there are more than one set $\mathcal{O}_r$ that meet the criteria of line 5b, pick the set with the
      greatest cardinality.
   d. $R' = R' \cup r$
   e. For each $o' \in \mathcal{O}_r \cap \mathcal{O}'$, do the following:
      i. $\mathcal{O}' = \mathcal{O}' - o'$
      ii. For each $o'' \in \mathcal{O}' \cap REL_{o'}$, decrement $key_{o''}$
6. Return $R'$
```

In the GREEDY-REP-MC2 algorithm, we proceed as follows.

- For any observation $o \in \mathcal{O}$, the set GRP_o is the set of all $\mathcal{O}_r$ where r is a feasible region (*i.e.*, a member of R) that explains o. Thus, $\mathcal{O}_r$ is the set of observations $\mathcal{O}_r$ that contain o and that are explained by some region r$\in R$.

- REL_o is the set of all observations that are contained in sets $\mathscr{O}_r$ that are found in the previous step. Thus, if an observation $o' \in REL_o$, there is at least one region $r \in R$ which explains both o and o'.
- key_o is the size of Rel_o.
- We pick the o such that key_o is minimal, *i.e.*, an o that is "co-explained" as poorly as possible.
- We then find a region r that explains o and that overlaps the set of observations as much as possible. Let $\mathscr{O}_r$ be the set of observations explained by r—if multiple such r's exist, pick the one with the highest cardinality.
- Add r to the "current" answer, and eliminate o from $\mathscr{O}$ as it no longer needs to be explained.
- For every observation $o'' \in \mathscr{O}$ that is explained already by $\mathscr{O}$, we reduce $key_{o''}$ by 1 as one explanation for it has already been found.
- This loop is repeated until all observations are explained.

In the rest of this section, we use Δ to denote the maximum number of different regions that can be partnered with a given observation.

Proposition 3.4. *GREEDY-REP-MC2 runs in $O(\Delta \cdot f^2 \cdot |\mathscr{O}| + |\mathscr{O}| \cdot \ln(|\mathscr{O}|)$ time and returns a solution whose cardinality is within a factor of $1 + \ln(f)$ of optimal.*

Proof. CLAIM 1: GREEDY-REP-MC2 runs in $O(\Delta \cdot f^2 \cdot |\mathscr{O}| + |\mathscr{O}| \cdot \ln(|\mathscr{O}|)$ time.
The pre-processing in lines 1-4 can be accomplished in $O(\Delta + \Delta \cdot f)$ as the size of each GRP_o is bounded by Δ and the size of each REL_o is bounded by $\Delta \cdot f$.

The outer loop of the algorithm iterates $\mathscr{O}$ times. In each loop, the selection of the minimal element (line 5a) can be accomplished in constant time by use of a Fibonacci heap [13] (*i.e.*, storing observations in $\mathscr{O}'$ organized by the value key_o). The next lines of the inner loop (lines 5b-5c) can be accomplished in $O(\Delta)$ time. The next line (line 5d) requires $O(\ln(|\mathscr{O}|)$ time per observation using a Fibonacci heap. However, we can be assured that, during the entire run of the algorithm, this operation is only performed $|\mathscr{O}|$ times (hence, we add an $|\mathscr{O}| \cdot \ln(|\mathscr{O}|)$). The final loop at line 5e occurs after the inner loop and iterates, at most f times. At each iteration, it considers, at most $f \cdot \Delta$ elements. Hence, the overall complexity is:

$$O(|\mathscr{O}| \cdot \left(\Delta + f^2 \cdot \Delta\right) + |\mathscr{O}| \cdot \ln(|\mathscr{O}|))$$

The statement of the claim follows.

CLAIM 2: GREEDY-REP-MC2 returns a solution whose cardinality is within a factor of $1 + \ln(f)$ of optimal.
The proof of this claim resembles the approximation proof of the standard greedy algorithm for set-cover (see [5] page 1036).

Let $r_1, \ldots, r_i, \ldots, r_n$ be the elements of R', the solution to GREEDY-REP-MC2, numbered by the order in which they were selected. For each iteration (of the outer loop), let set COV_i be the subset of observations that are partnered for the first time with region r_i. Note that each element of $\mathcal{O}$ is in exactly one COV_i. For each $o_j \in \mathcal{O}$, we define $cost_j$ to be $\frac{1}{|COV_i|}$ where $o_j \in COV_i$. Let R^* be an optimal solution to the instance of Sub/Sup-REP-MC.

CLAIM 2.1: $\sum_{r_i \in R^*} \sum_{o_j \in \mathcal{O}_{r_i}} cost_j \geq |R|$
By the definition of $cost_j$, exactly one unit of cost is assigned every time a region is picked for the solution R. Hence,

$$COST(R) = |R| = \sum_{o_j \in \mathcal{O}} cost_j$$

The statement of the claim follows.

CLAIM 2.2: For some region $r \in R$, $\sum_{o_j \in \mathcal{O}_r} cost_j \leq 1 + \ln(f)$.
Let P be the subset of $\mathcal{O}$ that can be partners with p. At each iteration i of the algorithm, let $uncov_i$ be the number elements in P that do not have a partner. Let $last$ be the smallest number such that $uncov_{last} = 0$. Let $R_P = \{r_i \in R | (i \leq last) \wedge (COV_i \cap P \neq \emptyset)\}$. From here on, we shall renumber each element in R_P as $r_1, \ldots, r_{|R_P|}$ by the order they are picked in the algorithm (*i.e.*, if an element is picked that cannot partner with anything in P, we ignore it and continue numbering with the next available number, we will COV_i and the iterations of the algorithm as well, but do not re-define the set based on the new numbering).
We note that for each iteration i, the number of items in P that are partnered is equal to $uncov_{i-1} - uncov_i$. Hence,

$$\sum_{o_j \in \mathcal{O}_r} cost_j = \sum_{i=1}^{last} \frac{uncov_{i-1} - uncov_i}{|COV_i|}$$

At each iteration of the algorithm, let $PCOV_i$ be the subset of observations that are covered for the first time if region p is picked instead of region r_i. We note, that for all iterations in $1, \ldots, last$, the region p is considered by the algorithm as one of its options for greedy selection. Therefore, as p is not chosen, we know that $|COV_i| \leq |PCOV_i|$. Also, by the definition of $ucov_i$, we know that $|PCOV_i| = ucov_{i-1}$. This gives us:

$$\sum_{o_j \in \mathcal{O}_r} cost_j \leq \sum_{i=1}^{last} \frac{uncov_{i-1} - uncov_i}{ucov_{i-1}}$$

Using the algebraic manipulations of [5] (page 1037), we get the following:

$$\sum_{o_j \in \mathcal{O}_r} cost_j \leq H_{|P|}$$

Where H_j is the jth harmonic number. By definition of the symbol f (maximum number of observations supported by a single partner), we obtain the statement of the claim.

(Proof of Claim 2): Combining claims 1–2, we get $|R| \leq \sum_{r_i \in R^*}(1+\ln(f))$, which gives us the statement.

While GREEDY-REP-MC2 considers regions in a different order than GREEDY-REP-MC, it maintains the same approximation ratio. This is because the region (in set GRP_o) that is partnered with the greatest number of uncovered observations is selected at each iteration, allowing us to maintain the approximation guarantee. There are two selections at each step: the selection of the observation (in which we use a minimal key value based on related observations) and a greedy selection in the inner loop. Any selection of observations can be used at each step and the approximation guarantee is still maintained. This allows for a variety of different heuristics. Furthermore, the use of a data structure such as a Fibonacci Heap allows us to actually obtain a better time complexity than GREEDY-REP-MC.

Example 3.8. Consider the situation in Example 3.4 where tiger conservation experts are considering regions $R_{\mathcal{O}} = \{r_1, \ldots, r_{33}\}$ that are induced by the set of observations and wish to solve I-REP-MC using GREEDY-REP-MC. On the first iteration of the loop at line 5, the algorithm picks o_8, as $key_{o_8} = 1$. The only possible region to pick is r_{19}, which can only be partnered with o_8. There are no observations related to o_8 other than itself, so it proceeds to the next iteration. It then selects o_6 as $key_{o_6} = 2$ because $REL_{o_6} = \{o_6, o_7\}$. It then greedily picks r_{17} which sub-explains both o_6, o_7. As all observations related to o_6 are now sub-explained, the algorithm proceeds with the next iteration. The observation with the lowest key value is o_5 as $key_{o_5} = 3$ and $REL_{o_5} = \{o_4, o_5, o_{13}\}$. It then greedily picks region r_{21} which sub-explains o_5, o_{13}. The algorithm then reduces the key value associated with o_4 from 4 to 3 and decrements the keys associated with o_{10}, o_{11}, o_{12} (the un-explained observations related to o_{13}) also from 4 to 3. In the next iteration, the algorithm picks o_9 as $key_{o_9} = 3$. It greedily picks r_{12} which sub-explains o_9, o_2. It then decreases key_{o_4} to 2 and also decreases the keys associated with o_1 and o_3. At the next iteration, it picks o_1 as $key_{o_1} = 2$. It greedily selects r_4, which sub-explains o_1, o_3 and decreases the key_{o_4} to 1 which causes o_4 to be selected next, followed by a greedy selection of r_{11}—no keys are updated at this iteration. In the final iteration, it selects o_{10} as $key_{o_{10}} = 3$. It greedily selects r_{25}, which sub-explains all un-explained observations. The algorithm terminates and returns $\{r_{11}, r_{12}, r_{17}, r_{19}, r_{21}, r_{25}\}$.

3.4.2 Approximation for a Special Case

In Section 3.3, we showed that circle covering is polynomially reducible to I-REP-MC. Let us consider a special (but natural) case of I-REP-MC where $\alpha = 0$, *i.e.*, there is no minimum distance between an observation and a partner point that caused it. We shall call this special case I-REP-MCZ. There is a great similarity between this problem and circle-covering. It is trivial to modify our earlier complexity proof to obtain the following result.

Corollary 3.3. *I-REP-MCZ is polynomially reducible to CC.*

Proof. Follows directly from Theorem 3.2.

Furthermore, we can adopt any algorithm that provides a constructive result for circle covering to provide a result for I-REP-MCZ in polynomial time with the following algorithm. Given any point p, it identifies the set $\mathscr{O}_r$ associated with the region that encloses that point.

```
FIND-REGION($\mathscr{S}$ space, $\mathscr{O}$ observation set, $\beta$ real, $p$ point) returns set $\mathscr{O}_r$
1. Set $\mathscr{O}_r = \emptyset$
2. For each $o \in \mathscr{O}$, if $d(p,o) \leq \beta$ then $\mathscr{O}_r = \mathscr{O}_r \cup \{o\}$
3. Return $\mathscr{O}_r$.
```

What FIND-REGION does is initially set $\mathscr{O}_r$ to the empty set. It then looks at all observations $o \in \mathscr{O}$. If the observation o is within β units or less from point p, it inserts p into the set $\mathscr{O}_r$.

Proposition 3.5. *The algorithm FIND-REGION runs in $O(|\mathscr{O}|)$ time, and region r (associated with the returned set $\mathscr{O}_r$) contains p.*

Proof. PART 1: FIND-REGION consists of a single loop that iterates $|\mathscr{O}|$ times.

PART 2: Suppose, the region enclosing point p has a different label. Then, there is either a bit in *label* that is incorrectly set to 1 or 0. As only observations which are at a distance of β or less from p have the associated bit position set to 1, then all 1 bits are correct. As we exhaustively consider all observations, the 0 bits are correct. Hence, we have a contradiction.

By pre-processing the regions, we can compute $\mathscr{O}_r$ *a priori* and simply pick a region r associated with the output for FIND-REGION. While there may be more than one such region, any one can be selected as, by definition, they would support the same observations.

Example 3.9. Paleontologists working in a 30×26km area represented by space $\mathscr{S}$ have located scattered fossils of prehistoric vegetation at $\mathscr{O} = \{o_1, o_2, o_3, o_4\}$. Previous experience has led the paleontologists to believe that a fossil site will be within 3km of the scattered fossils. In Figure 3.5, the observations are labeled and circles with radius of 3km are drawn (shown with solid lines). Induced regions $r_1, \ldots, r_6$ are also labeled. As the paleontologists have no additional information, and $\alpha = 0$, they can model their problem as an instance of I-REP-MCZ with $k = 3$. They can solve this problem by reducing it to an instance of circle-covering. The circle-covering algorithm returns three points - p_1, p_2, p_3 (marked with an 'x' in Figure 3.5). Note that each point in the solution to circle-covering falls in exactly one region (when using induced regions). The algorithm FIND-REGION returns the set $\{o_1, o_2\}$ for point p_1, which corresponds with region r_2. It returns set $\{o_3\}$ for p_2, corresponding with r_6 and returns set $\{o_4\}$ for p_3, corresponding with r_5. Hence, the algorithm returns regions r_2, r_6, r_5, which explains all observations.

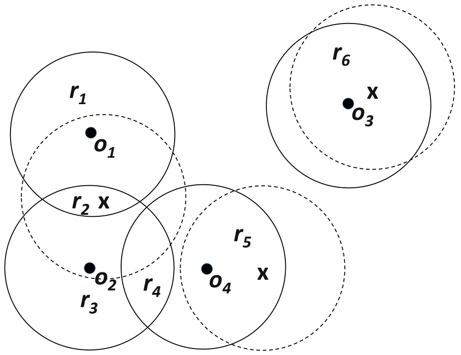

Fig. 3.5 Given the instance of I-REP-MCZ for Example 3.9 as input for circle-covering, a circle-covering algorithm returns points p_1, p_2, p_3 (points are denoted with an "x", dashed circles represent all points within $3km$ from the point).

Any algorithm that provides a constructive result for CC can provide a constructive result for I-REP-MCZ. Because of this one-to-one mapping between the problems, we can also be assured that we maintain an approximation ratio of any approximation technique.

Corollary 3.4. *An a−approximation algorithm for CC is an a-approximation for I-REP-MCZ.*

Proof. Follows directly from Theorem 3.2.

This is useful as we can now use approximation algorithms for CC on I-REP-MCZ. Perhaps the most popular approximation algorithms for CC are based on the "shifting strategy" [18]. To leverage this strategy, we would divide the space, $\mathcal{S}$, into strips of width $2 \cdot \beta$. The algorithm considers groups of ℓ consecutive strips—ℓ is called the "shifting parameter." A local algorithm **A** is applied to each group of strips. The union of all solutions is a feasible solution to the problem. The algorithm then shifts all strips by $2 \cdot \beta$ and repeats the process, saving the feasible solution. This can be done a total of $\ell - 1$ times, and the algorithm simply picks the feasible solution with minimal cardinality. In [18], the following lemma is proved (we state it in terms of I-REP-MCZ—which is done by an application of Corollary 3.4):

Lemma 3.2 (Shifting Lemma [18]). *Let $\boldsymbol{a}_{S(\boldsymbol{A})}$ be the approximation factor of the shifting strategy applied with local algorithm* **A** *and $\boldsymbol{a}_{\boldsymbol{A}}$ be the approximation factor for the local algorithm. Then:*

$$\boldsymbol{a}_{S(\boldsymbol{A})} = \boldsymbol{a}_{\boldsymbol{A}} \cdot \left(1 + \frac{1}{\ell}\right).$$

Furthermore, the shifting strategy can actually be applied twice, solving the local algorithm in squares of size $2 \cdot \beta \cdot \ell \times 2 \cdot \beta \cdot \ell$. This gives the following result:

$$\boldsymbol{a}_{S(S(\mathbf{A}))} = \boldsymbol{a}_{\mathbf{A}} \cdot \left(1 + \frac{1}{\ell}\right)^2 .$$

A good survey of results based on the shifting strategy can be found in [8], which also provides a linear-time algorithm (this result is later generalized by [9] for multiple dimensions). The following result leverages this for I-REP-MCZ by Corollary 3.4 (and is proved in [9]).

Proposition 3.6. *I-REP-MCZ can be solved with an approximation ratio of $\boldsymbol{x} \cdot \left(1 + \frac{1}{\ell}\right)^2$ in $O(K_{\ell,\rho} \cdot |\mathscr{O}|)$ time. Where p is the maximum number of points in a finite lattice over a square of side length $2 \cdot \beta \cdot \ell$ s.t. each observation in such a square lies directly on a point in the lattice and $\boldsymbol{x} \in \{3,4,5,6\}$ (and is determined by β, see [8] for details) and $K_{\ell,\rho}$ is defined as follows.*

$$K_{\ell,\rho} = \ell^2 \cdot \sum_{i=1}^{\lceil \ell \cdot \sqrt{2} \rceil^2 - 1} \binom{p}{i} \cdot i$$

An alternative to the shifting strategy leverages techniques used for the related problem of geometric dominating set. In [4], the authors present a $1+\varepsilon$ approximation that runs in $O(|\mathscr{O}|^{O(\frac{1}{\varepsilon^2} \cdot \lg^2(\frac{1}{\varepsilon}))})$ time.

3.4.3 Practical Considerations for Implementation

We now describe some practical implementation issues. Our primary aim is to find a set of regions that resembles the set of induced regions, $R_{\mathscr{O}}$. There are several reasons for doing this. One reason is to implement a fast heuristic to deal with I-REP optimization problems, specifically when $\alpha \neq 0$. Another, is that such a set of induced regions in the space may be a starting point for creating a set of regions that may include other data, such as that shown in Example 3.5.

As most GIS systems view space as a set of discrete points, we discretized the space using the REGION-GEN algorithm below. The parameter g is the spacing of a square grid that overlays the space.

The result below specifies the running time complexity of the REGION-GEN algorithm.

Proposition 3.7. *REGION-GEN has a time complexity $\Theta(|\mathscr{O}| \cdot \frac{\pi \cdot \beta^2}{g^2})$.*

Proof. For any given observation, the number of points in the grid that can be in a partnered region is $\frac{\pi \cdot \beta^2 - \alpha^2}{g^2}$. Hence, the first loop of the algorithm and the size of L are both bounded by $|\mathscr{O}| \cdot \frac{\pi \cdot \beta^2}{g^2}$. We note that the lookup and insert operations for the

```
REGION-GEN($\mathscr{S}$ space, $\mathscr{O}$ observation set, $\alpha,\beta,g$ reals) returns set $R$

1. Overlay a grid of spacing $g$ on space $\mathscr{S}$. With each grid point, $p$, associate set $\mathscr{O}_p = \emptyset$. This
   can easily be represented with an array.
2. Initialize list $L$ of pointers to grid-points.
3. For each $o \in \mathscr{O}$, identity all grid points within distance $[\alpha,\beta]$. For each point $p$ meeting this
   criteria, if $\mathscr{O}_p = \emptyset$, add $p$ to $L$. Also, set $\mathscr{O}_p = \mathscr{O}_p \cup \{o\}$
4. For some subset $\mathscr{O}' \subset \mathscr{O}$, let $str(\mathscr{O}')$ be a bit string of length $|\mathscr{O}|$ where every position corre-
   sponding to an element of $\mathscr{O}'$ is 1 and all other positions are 0.
5. Let $T$ be a hash table of size $\lceil |\mathscr{O}| \cdot \frac{\pi \cdot \beta^2}{g^2} \rceil$ regions indexed by bit-strings of length $|\mathscr{O}|$
6. For each $p \in L$, do the following:
   a. If $T[str(\mathscr{O}_p)] =$ null then initialize this entry to be a rectangle that encloses point $p$.
   b. Else, expand the region at location $T[str(\mathscr{O}_p)]$ to be the minimum-enclosing rectangle that
      encloses $p$ and region $T[str(\mathscr{O}_p)]$.
7. Return all entries in $T$ that are not null.
```

hash table T do not affect the average-case complexity. We assume these operations take constant time based on [5], hence the statement follows.

Let us return to our paleontology example from Example 3.9.

Example 3.10. Consider the scenario from Example 3.9. Suppose the paleontologists now want to generate regions using REGION-GEN instead of using induced regions. The algorithm REGION-GEN overlays a grid on the space in consideration. Using an array representing the space, it records the observations that can be explained by each grid point (Figure 3.6, top). As it does this, any grid point that can explain an observation is stored in list L. The algorithm then iterates through list L, creating entries in a hash table for each subset of observations, enclosing all points that explain the same observation with a minimally-enclosing rectangle. Figure 3.6 (bottom) shows the resulting regions $r_1,\ldots,r_6$.

One advantage to using REGION-GEN is that we already have the observations that a region super-explains stored—simply consider the bit-string used to index the region in the hash table. Another thing that can be done, for use in an algorithm such as GREEDY-MC2, where the regions are organized by what observation they support, can also be easily done during the running of this algorithm at an additional cost of f (the number of observations that can be partnered with a given region). This is done by updating an auxiliary data structure, shown at line 6a.

3.5 Experimental Results

We implemented REGION-GEN and GREEDY-MC2 in approximately 3000 lines of Java code and conducted several experiments on a Windows-based computer with an Intel x86 processor. Our goal was to show that solving the optimization problem

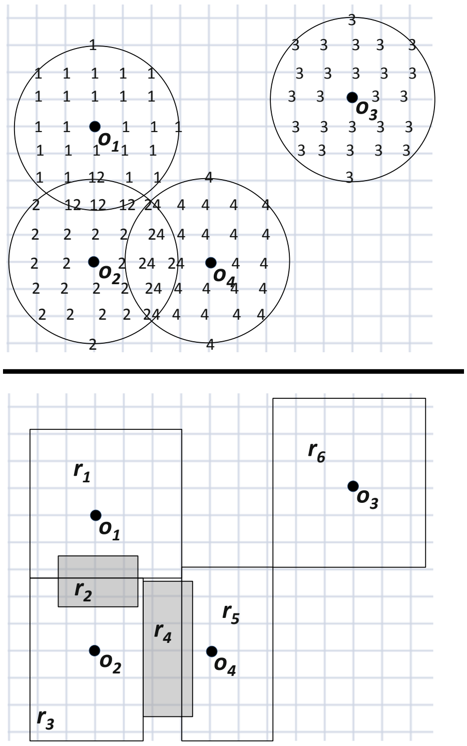

Fig. 3.6 REGION-GEN applied to the paleontology example (Example 3.9). First, it identifies observations associated with grid points (top). It then creates minimally-enclosing rectangles around points that support the same observations (bottom).

Sup-REP-MC would provide useful results in a real-world scenario. We looked at counter-insurgency data from [38] that included data on improvised-explosive device attacks in Baghdad and cache sites where insurgents stored weapons. Under the assumption that the attacks required support of a cache site a certain distance away, could we use attack data to locate cache sites using an instance of Sup-REP-MC solved with GREEDY-MC2 using regions created with REGION-GEN? In our framework, the observations were attacks associated with a cache (which was a partner). The goal was to find relatively small regions that enclosed partners (caches). We evaluated our approach based on the following criteria:

1. Do the algorithms run in a reasonable amount of time?
2. Does GREEDY-MC2 return regions of a relatively small size?
3. Do the regions returned by GREEDY-MC2 usually contain a partner (cache)?
4. Is the partner (cache) density within regions returned by GREEDY-MC2 significantly greater than the partner density of the space?

5. How does the spacing between grid points affect the runtime and accuracy of the algorithms?

Overall, the experiments indicate that REGION-GEN and GREEDY-MC2 satisfactorily meet the requirements above. For example, for our trials considering locating regions with weapons cache sites (partners) in Baghdad given recent IED attacks (observations), with a grid spacing of $g = 100$m, the combined (mean) run-time on a Windows-based laptop was just over 2 seconds. The algorithm produced (mean) 15.54 regions with an average area of 1.838 km^2. Each region, on average, enclosed 1.739 cache sites. If it did not contain a cache site, it was (on average) 275m away from one. The density of caches within returned regions was 8.09 caches/km^2—significantly higher than the overall density for Baghdad of 0.488 caches/km^2.

The rest of this section is organized as follows. Section 3.5.1 describes the data set we used for our tests and experimental set-up. Issue 1 is addressed in Section 3.5.2. We shall discuss the area (issue 2) of the regions returned in Section 3.5.3 and follow this with a discussion of issue 3 in Section 3.5.4. We shall discuss issue 4 in Section 3.5.5. Throughout all the sections, we shall describe results for a variety of different grid spacings, hence addressing issue 5.

3.5.1 Experimental Setup

We used the *Map of Special Groups Activity in Iraq* available from the Institute for the Study of War [38]. The map plots over 1000 insurgent activities attributed to what are termed as "Special Groups"—groups with access to certain advanced weaponry. This data set—the same one used in Chapter 2—contains events for 21 months between February 2007 and November 2008. The activity types include the following categories.

1. Attacks with probable links to Special Groups
2. Discoveries of caches containing weapons associated with Special Groups
3. Detainments of suspected Special Groups criminals
4. Precision strikes against Special Groups personnel

We use this data for two geographic areas: the Baghdad urban area and the Sadr City district. In our experiment, we will view the attacks by the special groups (item 1) as observations and attempt to determine the minimum set of cache sites (item 3), which we shall view as partners. Hence, a region returned by GREEDY-MC2 encloses a partner iff a cache falls within the region.

For distance constraints, we used a simple algorithm to learn the parameter β (α was set to zero). This was done using the first 7 months of attack data ($\frac{1}{3}$ of the available months) and 14 months of cache data. We used the following simple algorithm, FIND-BETA, to determine these values. Note we set β_{max} to 2.5km.

We ran the experiments on a Lenovo T400 ThinkPad laptop with a 2.53 GHz Intel Core 2 Duo T9400 processor and 4GB of RAM. The computer was running Windows Vista 64-bit Business edition with Service Pack 1 installed.

Algorithm 10 Determines β value from historical data

```
FIND-BETA($\mathscr{O}_h$ historical, time-stamped observations,
$\mathscr{E}_h$ historical, time-stamped partners, $\beta_{max}$ real)

1. Set $\beta = \beta_{max}$
2. Set Boolean variable $flag$ to TRUE
3. For each $o \in \mathscr{O}_h$, do the following:
   a. For each $p \in \mathscr{E}_h$ that occurs after $o$, do the following.
      i. Let $d$ be the Euclidean distance function.
      ii. If $flag$, and $d(o,p) \leq \beta_{max}$ then set $\beta = d(o,p)$
      iii. If not $flag$, then do the following:
         A. If $d(o,p) > \beta$ and $d(o,p) \leq \beta_{max}$ then set $\beta = d(o,p)$
4. Return real $\beta$
```

As the relationship between attacks and cache sites may differ varied on terrain, we ran tests with two different geographic areas. First, we considered the entire Baghdad urban area. Then, we considered just the Sadr City district. We ran FIND-BETA with a β_{max} of 2.5 km on both areas prior to testing the algorithms. There were 73 observations (attacks) for Baghdad and 40 for Sadr City. Table 3.1 shows the exact locations and dimensions of the areas considered.

Area	Lower-Left Latitude	Lower-Left Longitude	E-W Distance	N-S Distance
Baghdad	33.200° N	44.250° E	27 km	25 km
Sadr City	33.345° N	44.423° E	7 km	7 km

Table 3.1 Locations and dimensions of areas considered

We conducted two types of tests: tests focusing on GREEDY-MC2 and tests focusing on REGION-GEN.

For the tests of GREEDY-MC2, we used multiple settings for the grid spacing g. We tested grid spacings at every 10 meter interval in the range of $[70, 1000]$ meters, giving a total of 93 different values for g. Due to the fact that REGION-GEN produces a deterministic result, we ran that algorithm only once per grid setting. However, we ran 100 trials of GREEDY-MC2 for each parameter g. This was done for both Baghdad and Sadr City, giving a total of $18,600$ experiments.

To study the effects of grid-spacing on the run-time of REGION-GEN, we also ran 25 trials for each grid spacing setting for both geographic areas, yielding a total of $4,650$ experiments. To compare the algorithms running with different settings for g in a statistically valid manner, we used ANOVA [14] to determine if the differences among grid spacings were statistically significant. For some test results, we conducted linear regression analysis.

3.5.2 Running Time

Overall, the run times provided by the algorithms were quite reasonable. For example, for the Baghdad trials, 73 attacks were considered for an area of 675km^2. With a grid spacing $g = 100$m, REGION-GEN ran in 2340ms and GREEDY-MC2 took less than 30ms.

For GREEDY-MC2, we found that run-time generally decreased as g increased. For Baghdad, the average run times ranged over $[1.39, 34.47]$ milliseconds. For Sadr City, these times ranged over $[0.15, 4.97]$ milliseconds. ANOVAs for both Baghdad and Sadr City run-times gave p-values of $2.2 \cdot 10^{-16}$, which suggests with well over 99% probability that the algorithm run with different grid settings will result in different run times. We also recorded the number of regions considered in each experiment (resulting from the output of REGION-GEN). Like run-times, we found that the number of regions considered also decreased as the grid spacing increased. For Baghdad, the number of considered regions ranged over $[88, 1011]$. For Sadr City, this number ranged over $[25, 356]$. ANOVAs for both Baghdad and Sadr City number of considered regions gave p-values of $2.2 \cdot 10^{-16}$, which suggests with well over 99% probability that the algorithm run with different grid settings will result in different numbers of considered regions. Note that this is unsurprising as REGION-GEN run deterministically. We noticed that, generally, only grid spacings that were near the same value would lead to the same number of considered regions.

The most striking aspect of the run time/number of regions considered results for GREEDY-MC2 is that these two quantities seem closely related (see Figure 3.7). This most likely results from the fact that the number of regions that can be associated with a given observation (Δ) increases as the number of regions increases. This coincides with our analysis of GREEDY-MC2 (see Proposition 3.4).

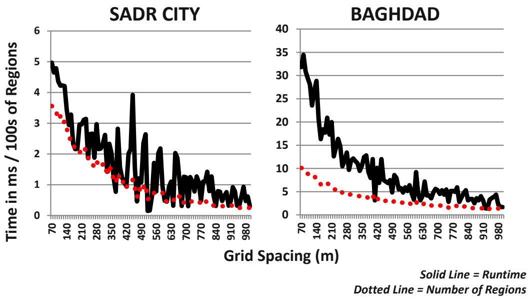

Fig. 3.7 The run time of GREEDY-MC2 in ms compared with the number of regions considered.

We also studied the average run-times for REGION-GEN for various different settings of the grid space g. For Baghdad, the average run times ranged over $[16.84, 9184.72]$ms. For Sadr City, these times ranged over $[0.64, 308.92]$ms. ANOVAs for both Baghdad and Sadr City run-times gave p-values of $2.2 \cdot 10^{-16}$, which suggests with well over 99% probability that the algorithm run with different grid settings will result in different run times. Our analysis of REGION-GEN (See Proposition 3.7) states that the algorithm runs in time $O(\frac{1}{g^2})$. We found striking similarities with this analysis and the experimental results (see Figure 3.8).

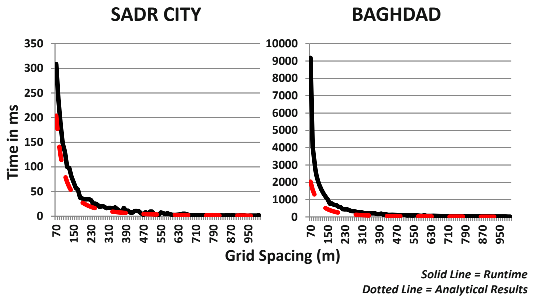

Fig. 3.8 A comparison between analytical ($O(\frac{1}{g^2})$) and experimental results for the run time of REGION-GEN compared with grid spacing (g).

3.5.3 Area of Returned Regions

In this section, we examine how well the REGION-GEN/GREEDY-MC2 suite of algorithms address the issue of returning regions that are generally small. Although not inherently part of the algorithm, our intuition is that the Sup-REP-MC optimization problem will generally return small regions based on the set R produced by REGION-GEN. The reason for this is that we would expect that smaller regions generally support more observations (note that this is not always true, even for induced regions, but our conjecture is that it is often the case for induced regions or the output of REGION-GEN).

To define "small" we look at the area of a circle of radius β as a basis for comparison. As different grid settings led to different values for β, we looked at the smallest areas. For a given trial, we looked at the average area of the returned regions.

For Baghdad, the average areas ranged over $[0.611, 2.985]\text{km}^2$. For Sadr City, these times ranged over $(0.01, 0.576]\text{km}^2$. ANOVAs for both Baghdad and Sadr City run-times gave p-values of $2.2 \cdot 10^{-16}$, which suggests with over a 99% probability that the algorithm run with different grid settings will result in different average areas. Plotting the areas compared with the established "minimum area" described earlier in this section clearly shows that REGION-GEN with GREEDY-MC2 produce solutions with an average area that is about half of this value (refer to Figure 3.9).

Overall, there seemed to be little relation between grid spacing and average area of the returned set of regions—based on grid spacings in $[70, 1000]$m. As an example, we provide screenshots of GREEDY-MC2 for $g = 100$ and $g = 1000$ (Figure 3.10). Anecdotally, we noticed that larger grid spacing led to more "pinpoint" regions—regions encompassing only one point in the grid (and viewed as having an area of 0). This is most likely due to the fact that overlaps in the circles around observations points would overlap on fewer grid points for larger values of g. Another factor is that different settings for g led to some variation of the value β—which also affects accuracy (note for our analysis we considered only the smallest values of β as an upper bound for the area (see Figure 3.9).

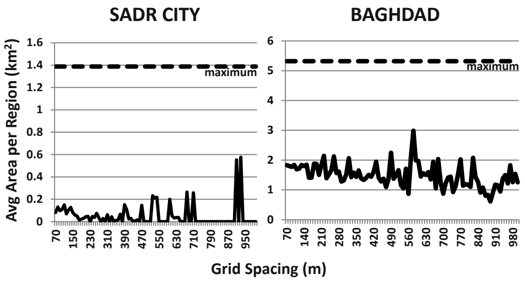

Fig. 3.9 Average areas for solutions provided by REGION-GEN with GREEDY-MC2 for Baghdad and Sadr City.

3.5.4 Regions that Contain Caches

In this section we discuss the issue of ensuring that most of the returned regions enclose at least one partner (cache in the case of our experiments). One measure of this aspect is to look at the average number of caches enclosed per region in a given result. We found that for Baghdad, we generally enclosed more than 1 cache per region in a given result—this number was in the range $[0.764, 3.25]$. The results for Sadr City were considerably lower—in the range $[0, 0.322]$. ANOVAs for both Baghdad and Sadr City gave p-values of $2.2 \cdot 10^{-16}$, which suggests with over a 99% probability that the algorithm run with different grid settings will result in different average number of enclosed caches. However, we did not observe an obvious trend in the data (see Figure 3.11).

As an alternative metric, we look at the number of regions provided by GREEDY-MC2 that contain at least one partner. Figure 3.13 shows the number of regions returned in the output. For Baghdad, generally fewer than half the regions in the output will enclose a cache—the number of enclosing regions was in $[1, 8]$, while the total number of regions was in $[10.49, 22]$. This result, along with the average number

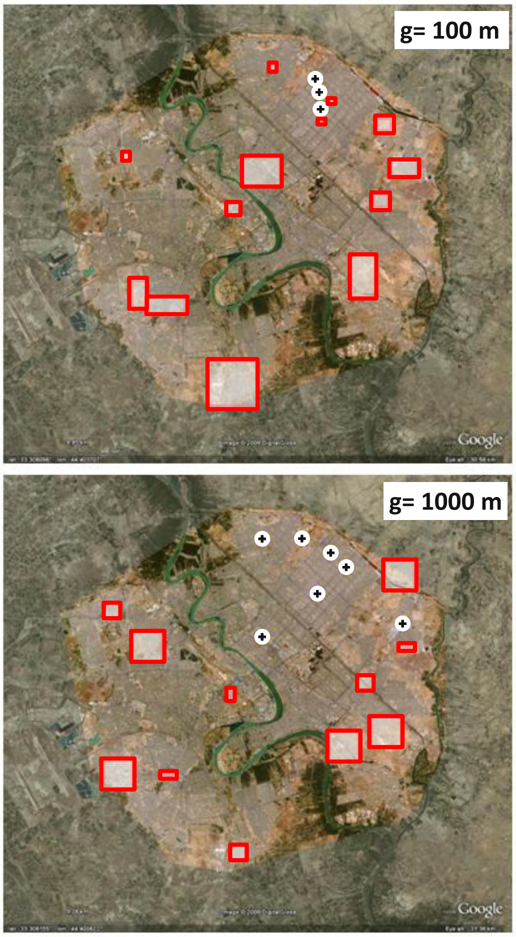

Fig. 3.10 Results from two runs of GREEDY-MC2 - $g = 100m$ (top), $g = 1000m$ (bottom). Pinpoint-regions are denoted with plus-signs. Notice that the average areas of the results are comparable.

of caches enclosed by a region, may indicate that while sometimes GREEDY-MC2 may find regions that enclose many caches, there are often regions that enclose no caches as well. This may indicate that for Baghdad, some attacks-cache relationships conform to our model and others do not. Perhaps there is another discriminating attribute about the attacks not present in the data that may account for this phenomenon. For example, perhaps some attacks were performed by some group that had the capability to store weapons in a cache located further outside the city, or perhaps some groups had the capability to conduct attacks using cache sites that were never found. We illustrate this phenomenon with an example output in Fig-

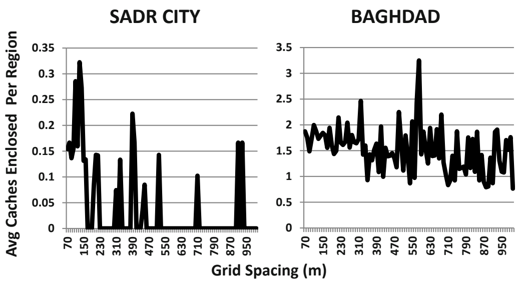

Fig. 3.11 Average caches enclosed per region for Baghdad and Sadr City for various grid-spacing settings.

ure 3.12. Note that in the figure, regions A–E do not contain any cache sites while regions G–I all contain numerous cache sites.

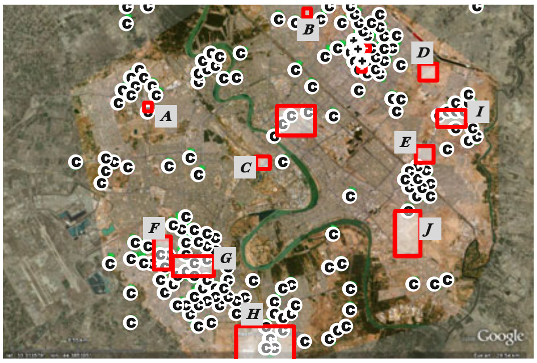

Fig. 3.12 The output of GREEDY-MC2 for Baghdad with $g = 100$m compared with the locations of actual cache sites (denoted with a "C"). Notice that regions A–E do not contain any cache sites while regions G–I all contain numerous cache sites.

For Sadr City, the number of caches that contain one cache was significantly lower—in the range $[0,2]$—while the total number of returned regions was in $[3,9.8]$. ANOVAs for both Baghdad and Sadr City gave p-values of $2.2 \cdot 10^{-16}$, which suggests with well over 99% probability that the algorithm, run with different grid settings, will result in different number of regions that enclose a cache location.

We believe that the low numbers for caches enclosed by regions for Sadr City were directly related to the smaller areas of regions. However, the mean of the average area of a returned set of regions was 0 for 49 of the 94 different grid settings (for Sadr City). This means that for the majority of grid settings, the solution consisted only of pinpoint regions (see Section 3.5.3 for a description of pinpoint regions).

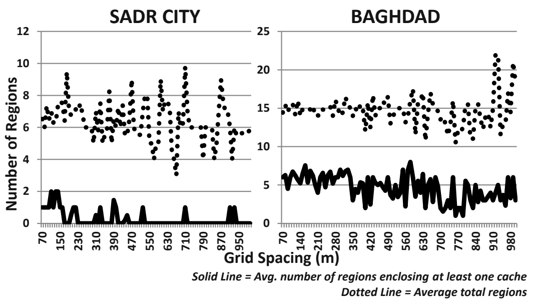

Fig. 3.13 Regions in the output that enclose at least one partner (cache) and total number of regions returned for Baghdad and Sadr City.

Obviously, it is unlikely for a pinpoint region to contain a cache site merely due to its infinitesimally small area. To better account for this issue, we develop another metric: distance to nearest cache. If a region contains a cache, the value for this metric is 0. Otherwise, it is the distance to the closest cache outside of the region. For Baghdad, we obtained distances in $[0.246, 0.712]$km, for Sadr City, $[0.080, 0.712]$km. ANOVAs for both Baghdad and Sadr City gave p-values of $2.2 \cdot 10^{-16}$, which suggests with well over 99% probability that the algorithm run with different grid settings will result in different distances to the nearest cache. Using linear regression, we observed that this distance increases as grid spacing increases. For Baghdad, we obtained $R^2 = 0.2396$ and $R^2 = 0.2688$ for Sadr City. See Figure 3.14 for experimental results and the results of the liner regression analysis.

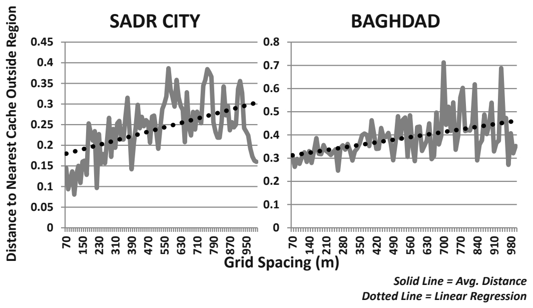

Fig. 3.14 Distance to nearest cache versus grid spacing (in meters).

3.5.5 Partner Density

To consider the density of partners in the regions, we compare the number of enclosed partners to the overall partner density of the area in question. For Baghdad, there were 303 caches in an area measuring 27×24km, giving a density of 0.488 caches/km^2. For Sadr City, there were 64 caches in an area of 7×7km, giving a density of 1.306 caches/km^2. In our experiments, we looked at the cache density for each output. For Baghdad, the density was significantly higher, ranging in $[0.831, 34.9]$ cache/km^2. If we consider $g \in [70, 200]$, the density is between $[7.19, 32.9]$ cache/km^2. For $g = 100$, the density was 8.09 caches/km^2. Most likely due to the issue of pinpoint regions described in Section 3.5.3, the results for Sadr City were often lower than the overall density (in $[0, 31.3]$ cache/km^2). For $g = 100$, the density was 2.08 caches/km^2. We illustrate these results compared with overall cache density in Figure 3.15.

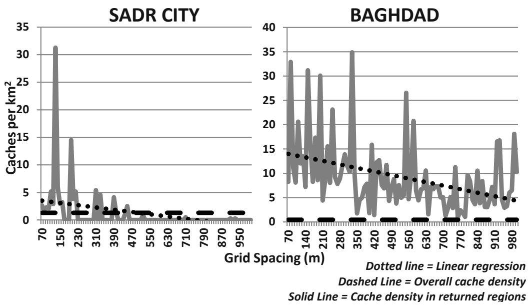

Fig. 3.15 Cache density of outputs produced by GREEDY-MC2 for Baghdad and Sadr City compared with overall cache density and linear-regression analysis.

ANOVAs for both Baghdad and Sadr City gave p-values of $2.2 \cdot 10^{-16}$, which suggests with well over 99% probability that the algorithm run with different grid settings will result in different cache densities. Using linear regression, we observed that this cache density decreases as grid spacing increases. For Baghdad, we obtained $R^2 = 0.1614$ and $R^2 = 0.1395$ for Sadr City. See Figure 3.15 for experimental results and the results of the linear regression analysis.

Although partner density is a useful metric, it does not tell us anything about partners that lie close to a region, although still outside. For example, consider Figure 3.12. Although region A does not enclose any caches, there is a cache just outside. Region B is similar. Also consider the cluster of caches south of region E and north of region J—in this situation it appears as though GREEDY-MC2 mispositioned a region. We include a close-up of region F in Figure 3.16, which encloses a cache, but there are also 4 other caches at a distance of 250m or less.

In order to account for such phenomena, we created an area-quadrupling metric—that is we uniformly double the sides of each region in the output. Then, we cal-

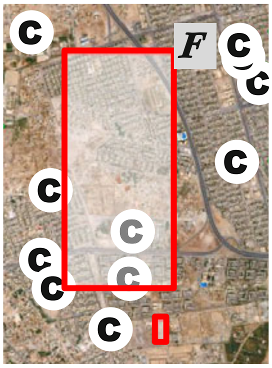

Fig. 3.16 Close-up of region F from Figure 3.12. While region F contains 1 cache, there are 4 other caches < 250m from the boundary of that region. The area-quadrupling metric helps us account for such scenarios.

culated the density of the output with area-quadrupled regions. For Baghdad, this density was in $[0.842, 30.3]$ caches/km^2. For Sadr City, this density was in $[0, 12.3]$ caches/km^2. These results are depicted in Figure 3.17.

As the regions for Sadr City were often smaller than those in Baghdad, we found that the cache density for area-quadrupled regions was often higher for Sadr City (*i.e.*, a region in Sadr City would have nearby cache sites). An example is shown in Figure 3.16.

ANOVAs for both Baghdad and Sadr City gave p-values of $2.2 \cdot 10^{-16}$, which suggests with well over 99% probability that the algorithm run with different grid settings will result in different cache densities for area-quadrupled regions. We also conducted linear regression analysis, and, like the normal partner density, we found that cache density decreases as grid spacing increases. However, this linear analysis was more closely correlated with the data than the analysis for non-area-quadrupled density. For Baghdad, we obtained $R^2 = 0.3171$ (for non-area-quadrupled, we obtained $R^2 = 0.1614$) and $R^2 = 0.3983$ (for non-area-quadrupled, we obtained $R^2 = 0.1395$) for Sadr City. See Figure 3.17 for experimental results and the results of the liner regression analysis.

3.6 Conclusion

In Chapter 2, we developed a formulation of the geospatial abduction problem that assumed that:

- Space was discretized into integer-valued coordinates and handled in a discretized manner as is the case with most real-world GISs; and
- The user desires a set of *points* coming back to him as an explanation.

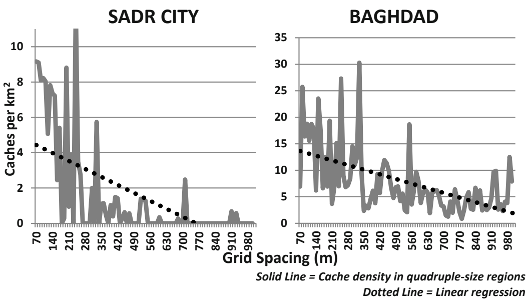

Fig. 3.17 Area quadrupled cache density of output produced by GREEDY-MC2 with linear-regression analysis.

In contrast, in this chapter, we have developed a formalism that:

- Treats space as a continuous two-dimensional set as in this real world (but unlike the way most GIS systems on the market treat space); and
- Returned a set of *regions* to the user.

Returning a set of regions to the user can be highly actionable because each region deemed feasible (and represented as region $r \in R$) may be just large enough so the entity interested in geospatial abduction can act upon the results. For instance, a US military commander looking for weapons caches may know that he cannot search certain regions (for one reason or another). Then he may specify the set R in the input to the region-based geospatial abduction problem to *ignore* such unsearchable regions. In this chapter, R operates much like a feasibility predicate in Chapter 2. However, it goes further in two ways: first, it can be used to specify that *regions* (as opposed to points) can be feasible or infeasible, and second, it can be used to regulate the sizes of the regions that an analyst may want to find. The system would return regions that he can search (*i.e.*, members of R) that offer the best probability of finding a weapons cache. Likewise, in the virus host location detection problem, a public health analyst may set R to consist of only some regions (*e.g.*, a public health expert looking at monkey pox in Rwanda may know that he cannot cross into neighboring countries like Uganda or the Democratic Republic of Congo to eradicate virus hosts). In this case, he may choose only regions $r \in R$ that are within Rwanda and ask the region-based geospatial abduction system to find the best regions in Rwanda for him to target for public health interventions *even though there may be better regions in Uganda or the Democratic Republic of Congo* for him to target with public health interventions.

Thus far, in Chapter 2 and Chapter 3, we have assumed that the adversary is nonchalant and is ignoring our efforts to locate it. This may be reasonable in the case of the virus host detection problem where perhaps mosquitoes and ticks do not have the cognitive capabilities to outwit us. But it is 100% certain that insurgents and terrorists, burglars and other criminals, and even the innocent, but much maligned tiger, have the cognitive capabilities to see what we are doing and adjust their strategy to attempt to outwit us. Tigers are likely to move away from areas of human intervention, just as insurgents in war zones track what we do and react in ways intended to outwit us. This is the focus of the next chapter.

References

1. Alpaydin, E.: 2010. *Introduction to Machine Learning*. MIT Press, 2 edition, 2010.
2. Brantingham, P., Brantingham, P.: 2008. Crime Pattern Theory. In Enviromental Criminology and Crime Analysis, R. Wortley and L. Mazerolle, Eds., pages 78–93.
3. , Bylander, T., Allemang, D., Tanner, M., Josephson, J.R.: 1991. The Computational Complexity of Abduction, Artificial Intelligence.
4. Liao, C., Hu, S.: 2009. Polynomial time approximation schemes for minimum disk cover problems, Journal of Combinatorial Optimization.
5. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: 2001. Introduction to Algorithms. MIT Press, second edition, 2001.
6. Eiter, T., Gottlob, G.: 1995. The complexity of logic-based abduction, J. ACM, 42, 1, pages 3–42.
7. Feige, U.: 1998. A threshold of ln n for approximating set cover, J. ACM, 45, 4, pages 634–652.
8. Franceschetti, M., Cook, M., Bruck, J.: 2004. A Geometric Theorem for Network Design, IEEE Transactions on Computers, 53, 4, pages 483–489.
9. Fu, B, Chen, Z., Abdelguerfi, M.: 2007. An Almost Linear Time 2.8334-Approximation Algorithm for the Disc Covering Problem, AAIM '07: Proceedings of the 3rd international conference on Algorithmic Aspects in Information and Management, pages 317–326, Springer-Verlag.
10. Hochbaum,D.S.,Maass, W.: 1985. Approximation schemes for covering and packing problems in image processing and VLSI, J. ACM, 32, pages 130–136.
11. Megiddo, N., Supowit,K.J.: 1984. On the Complexity of Some Common Geometric Location Problems, SIAM Journal of Computing, 13, 1, pages 182–196.
12. , Nemhauser, G. L., Wolsey, L. A., Fisher, M.L.: 1978. An analysis of approximations for maximizing submodular set functions I, Mathematical Programming, 14, 1, pages 265–294.
13. Fredman, M.L., Tarjan, R.E.: 1987. Fibonacci heaps and their uses in improved network optimization algorithms. Journal of the ACM, 34(3):596–615, July 1987.
14. Freedman, D., Purves, R., Pisani, R.: 2007. Statistics. W.W. Norton and Co., 4 edition.
15. Garey, M.R., Johnson, D.S.: 1979. Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, USA.
16. Hochbaum, D.S.: 1982. Approximation Algorithms for the Set Covering and Vertex Cover Problems. SIAM Journal on Computing, 11(3):555–556.
17. Hochbaum, D.S.: 1997. *Approximation Algorithms for NP-Complete Problems*. PWS Publishing Co., 1997.
18. Hochbaum, D.S., Maass, W.: 1985 Approximation schemes for covering and packing problems in image processing and vlsi. Journal of the ACM, 32:130–136.
19. Jia, L., Rajaraman, R. Suel, T.: 2002. An efficient distributed algorithm for constructing small dominating sets. Distrib. Comput., 15(4):193–205.
20. Johnson, D.S.: 1982. The np-completeness column: An ongoing guide. Journal of Algorithms, 3(2):182–195, 1982.
21. Karp, R.: 1972. Reducibility Among Combinatorial Problems. In R. E. Miller and J. W. Thatcher, editors, Complexity of Computer Computations, page 85-103.
22. Kuhn, F., Wattenhofer, R.: 2003. Constant-time distributed dominating set approximation. In In Proc. of the 22 nd ACM Symposium on the Principles of Distributed Computing (PODC, pages 25–32.

23. Lu, J., Nerode, A., Subrahmanian, V.S.: 1996. Hybrid Knowledge Bases, IEEE Transactions on Knowledge and Data Engineering, 8, 5, pages 773-785.
24. Lund, C., Yannakakis, M.: 1994. On the hardness of approximating minimization problems. Journal of the ACM, 41(5):960–981.
25. Papadimitriou, C.H.: 1981. Worst-Case and Probabilistic Analysis of a Geometric Location Problem, *SIAM J. Comput.*, 10(3):542–557.
26. Paschos, V.T.: 1997. A survey of approximately optimal solutions to some covering and packing problems. ACM Comput. Surv., 29(2):171–209.
27. Reggia, J.A., Peng, Y.: 1990. Abductive inference models for diagnostic problem-solving. Springer-Verlag New York, Inc., New York, NY, USA.
28. Rimoin, A. *et al.*: Endemic Human Monkeypox, Democratic Republic of Congo, 2001-2004, Emerging Infectious Diseases, 13, 6, pages 934–937, 2007.
29. Rossmo, D. K., Rombouts, S.: 2008. Geographic Profiling. In Enviromental Criminology and Crime Analysis, R. Wortley and L. Mazerolle, Eds. pages 136-149.
30. H. Samet.: The Design and Analysis of Spatial Data Structures, Addison Wesley, 1989.
31. Shakarian, P., Subrahmanian, V.S., Sapino, M.L. SCARE: A Case Study with Baghdad, Proc. 2009 Intl. Conf. on Computational Cultural Dynamics (eds. D. Nau, A. Mannes), Dec. 2009, AAAI Press.
32. Shakarian, P., Subrahmanian, V.S., Sapino, M.L. 2012. GAPS: Geospatial Abduction Problems, ACM Transactions on Intelligent Systems and Technology (TIST), 3, 1, to appear.
33. Shakarian, P., Subrahmanian, V.S. Region-based Geospatial Abduction with Counter-IED Applications, accepted for publication in: Wiil, U.K. (ed.).Counterterrorism and Open Source Intelligence, Springer Verlag Lecture Notes on Social Networks, to appear, 2011.
34. Shakarian, P., Nagel, M., Schuetzle, B., Subrahmanian, V.S. 2011. Abductive Inference for Combat: Using SCARE-S2 to Find High-Value Targets in Afghanistan, in Proc. 2011 Intl. Conf. on Innovative Applications of Artificial Intelligence, Aug. 2011, AAAI Press.
35. Shakarian, P., Dickerson, J., Subrahmanian, V.S. 2012. Adversarial Geospatial Abduction Problems, ACM Transactions on Intelligent Systems and Technology (TIST), to appear.
36. Singh, M., Joshi, P.K., Kumar,M., Dash, P.P., Joshi, B.D.: Development of tiger habitat suitability model using geospatial tools: a case study in Achankmar Wildlife Sanctuary (AMWLS), Chhattisgarh India, Env. Monitoring and Assessment journal, Vol. 155, pages 555-567, 2009.
37. US Army: *Intelligence Preparation of the Battlefiled (US Army Field Manual)*, FM 34-130 edition, 1994.
38. ”Map of Special Groups Activity in Iraq, Institute for the Study of War”, Institute for the Study of War, 2008.
39. Vazirani, V.V.: 2004. Approximation Algorithms. Springer, March 2004.