Adversarial Geospatial Abduction Problems

PAULO SHAKARIAN, United States Military Academy
JOHN P. DICKERSON and V. S. SUBRAHMANIAN, University of Maryland

Geospatial Abduction Problems (GAPs) involve the inference of a set of locations that "best explain" a given set of locations of observations. For example, the observations might include locations where a serial killer committed murders or where insurgents carried out Improvised Explosive Device (IED) attacks. In both these cases, we would like to infer a set of locations that explain the observations, for example, the set of locations where the serial killer lives/works, and the set of locations where insurgents locate weapons caches. However, unlike all past work on abduction, there is a strong adversarial component to this; an adversary actively attempts to prevent us from discovering such locations. We formalize such abduction problems as a two-player game where both players (an "agent" and an "adversary") use a probabilistic model of their opponent (i.e., a mixed strategy). There is asymmetry as the adversary can *choose* both the locations of the observations and the locations of the explanation, while the agent (i.e., us) tries to discover these. In this article, we study the problem from the point of view of both players. We define *reward functions* axiomatically to capture the similarity between two sets of explanations (one corresponding to the locations chosen by the adversary, one guessed by the agent). Many different reward functions can satisfy our axioms. We then formalize the *Optimal Adversary Strategy* (OAS) problem and the *Maximal Counter-Adversary strategy* (MCA) and show that both are NP-hard, that their associated counting complexity problems are #P-hard, and that MCA has no fully polynomial approximation scheme unless P=NP. We show that approximation guarantees are possible for MCA when the reward function satisfies two simple properties (zero-starting and monotonicity) which many natural reward functions satisfy. We develop a mixed integer linear programming algorithm to solve OAS and two algorithms to (approximately) compute MCA; the algorithms yield different approximation guarantees and one algorithm assumes a monotonic reward function. Our experiments use real data about IED attacks over a 21-month period in Baghdad. We are able to show that both the MCA algorithms work well in practice; while MCA-GREEDY-MONO is both highly accurate and slightly faster than MCA-LS, MCA-LS (to our surprise) always completely and correctly maximized the expected benefit to the agent while running in an acceptable time period.

Categories and Subject Descriptors: I.2.3 [**Artificial Intelligence**]: Deduction and Theorem Proving—*Nonmonotonic reasoning and belief revision*; I.2.3 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*; I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems—*Cartography*

General Terms: Algorithms, Experimentation, Theory

Additional Key Words and Phrases: Abduction, spatial reasoning

Some of the authors of this article were funded in part by AFOSR grant FA95500610405 and ARO grants W911NF0910206 and W911NF0910525.
Authors' addresses: P. Shakarian (corresponding author), United States Military Academy, West Point, NY 10996; email: Paulo@shakarian.net; J. P. Dickerson and V. S. Subrahmanian, University of Maryland, College Park, MD 20742.

DOI 10.1145/2089094.2089110 http://doi.acm.org/10.1145/2089094.2089110

ACM Reference Format:
Shakarian, P., Dickerson, J. P., and Subrahmanian, V. S. 2012. Adversarial geospatial abduction problems. ACM Trans. Intell. Syst. Technol. 3, 2, Article 34 (February 2012), 35 pages.
DOI = 10.1145/2089094.2089110 http://doi.acm.org/10.1145/2089094.2089110

1. INTRODUCTION

Geospatial Abduction Problems (GAPs) were introduced by Shakarian et al. [2010] to find a set of locations that "best explain" a given set of locations of observations. We call these inferred sets of locations "explanations." There are many such applications in a wide variety of domains.

— In *criminology*, serial killers carry out murders at various locations; these correspond to the observations we make. The goal of the police is to identify a set of locations that best "explain" the observations. Thus, the police look for the killer's home and office locations. The killer, of course, goes to considerable effort usually to ensure that he cannot be easily found by the police.
— In *military applications*, insurgents (such as those in Iraq and Afghanistan) carry out Improvised Explosive Device (IED) attacks at various locations; these corresponding to our observations. Multinational forces operating in these countries would like to identify many locations associated with these attack locations; one such class of locations corresponds to the locations of weapons caches that provide logistics support for the attacks and enable the attackers to carry them out. As in the case of the serial killer, the insurgents reason carefully about their choice of weapons cache locations to minimize the probability of being detected.
— In a wildlife application, a rare animal or bird might be spotted at several locations (observations). We would like to infer the location of the creature's nest or den. Many animals take considerable care to keep their den/nest hidden as these often hold young ones or eggs and, in some cases, food.

Shakarian et al. [2010] defined *Geospatial Abduction Problems* (GAPs) and studied a version of the problem where the adversary (the "bad guy" or the entity that wishes to evade detection) does not reason about the agent (the "good guy" or the entity that wants to detect the adversary). Despite this significant omission, they were able to accurately predict the locations of weapons caches in real-world data about IED attacks in Baghdad. In this article, we introduce *adversarial* geospatial abduction problems where both the agent and the adversary reason about each other. Specifically, our contributions are as follows.

(1) We axiomatically define *reward functions* to be any functions that satisfy certain basic axioms about the similarity between an explanation chosen by the adversary (e.g., where the serial killer lives and works or where the insurgents put their IED caches) and define notions of expected detriment (to the adversary) and expected benefit (to the agent).
(2) We formally define the *Optimal Adversary Strategy* (OAS) that minimizes chances of detection of the adversary's chosen explanation and the *Maximal Counter-Adversary strategy* (MCA) that maximizes the probability that the agent will detect the adversary's chosen explanation.
(3) We provide a detailed set of results on the computational complexity of these problems, the counting complexity of these problems, and the possibility of approximation algorithms with approximation guarantees for both OAS and MCA.
(4) We develop Mixed Integer Linear Programming algorithms (MILPs) for OAS and two algorithms, MCA-LS and MCA-GREEDY-MONO, to solve MCA with certain

approximation guarantees. MCA-LS has no assumptions, while MCA-GREEDY-MONO assumes monotonicity.

(5) We develop a prototype of our MILP algorithms to solve the OAS problem, using our techniques for variable reduction on top of an integer linear program solver. We demonstrate the ability to achieve near-optimal solutions as well as a correct reduction of variables by 99.6% using a real-world dataset.

(6) We develop a prototype implementation that shows that both MCA-LS and MCA-GREEDY-MONO are highly accurate and have very reasonable time frames. Though MCA-GREEDY-MONO is slightly faster than MCA-LS, we found that on every single run, MCA-LS found the exact optimal benefit even though its theoretical lower-bound approximation ratio is only 1/3. As MCA-LS does not require any additional assumptions and as its running time is only slightly slower than that of MCA-GREEDY-MONO, we believe this algorithm has a slight advantage.

The organization of the article is as follows. Section 2 first reviews the GAP framework of Shakarian et al. [2010]. Section 3 extends GAPs to the adversarial case using an axiomatically-defined reward function (Section 2). Section 4 presents complexity results and several exact algorithms using MILPs for the OAS problem. Section 5 provides complexity results and develops exact and approximate methods MCA, including an approximation technique that provides the best possible guarantee unless P = NP. We then briefly describe our prototype implementation and describe a detailed experimental analysis of our algorithms. Finally, related work is described in Section 7.

2. OVERVIEW OF GAPS

In this section, we briefly describe the theory of GAPs introduced by Shakarian et al. [2010]. With the exception of the counting complexity results (Lemma 2.1 and Theorem 2.1), everything in Section 2 appeared in Shakarian et al. [2010]. Throughout this article, we assume the existence of integers $M, N > 0$ that jointly define a 2-dimensional gridded space. We use $\mathbb{N}, \mathbb{R}, \mathbb{R}^+$ to respectively denote the sets of natural numbers, all real numbers, and nonnegative reals.

Definition 2.1 (*Space*). Suppose $M, N \in \mathbb{N}$. The *space* $\mathcal{S}$ is the set $\{1, \ldots, M\} \times \{1, \ldots, N\}$.

Throughout this article, we assume that $M, N, \mathcal{S}$ are arbitrary, but fixed. This representation of the space $\mathcal{S}$ as a set of integer coordinates is common in most Geospatial Information Systems (GIS). We use $2^{\mathcal{S}}$ to denote the power set of $\mathcal{S}$. We assume that $\mathcal{S}$ has an associated distance function d which assigns a nonnegative distance to any two points and satisfies the usual distance axioms.[1]

Definition 2.2 (*Observation Set*). An *observation set* $\mathcal{O}$ is any finite subset of $\mathcal{S}$.

For instance, in our IED application, an observation set is simply the set of locations where attacks occurred. In the serial killer example, the observation set is the set of locations where the killings occurred.

Definition 2.3 (*Feasibility Predicate*). A *feasibility predicate* is any function $\mathsf{feas} : \mathcal{S} \rightarrow \{\mathsf{TRUE}, \mathsf{FALSE}\}$.

Feasibility predicates encode domain knowledge. For instance, a feasibility predicate in the IED application might rule out the caches being on U.S. bases or in bodies of

[1] $d(x, y) \geq 0; d(x, x) = 0; d(x, y) = d(y, x); d(x, y) + d(y, z) \geq d(x, z)$.

water or (in the case of Baghdad where our dataset contains Shiite attacks) Sunni neighborhoods. Throughout this article, we assume an arbitrary, but fixed function feas that assigns either true or false to every point in $\mathcal{S}$. In our complexity results, we assume feas is computable in constant time.

Definition 2.4 ((α, β)-*explanation*). Given a finite set of observations $\mathcal{O}$ and real numbers $\alpha \geq 0$, $\beta > 0$, a finite set of points $\mathcal{E} \subseteq \mathcal{S}$ is an (α, β)-*explanation* of $\mathcal{O}$ iff:

(1) $(\forall p \in \mathcal{E})$ feas(p) = TRUE;
(2) $(\forall o \in \mathcal{O})(\exists p \in \mathcal{E})\, \alpha \leq d(p, o) \leq \beta$.

Intuitively, $\mathcal{E}$ is an (α, β)-explanation of $\mathcal{O}$ if every point in $\mathcal{E}$ is feasible and every observation in $\mathcal{O}$ is neither too close nor too far from a point in $\mathcal{E}$. For a given observation, o, we will refer to point p as a *partner* iff feas(p) and $d(o, p) \in [\alpha, \beta]$.

α and β are parameters that can be easily learned from historical data (as was done in Shakarian et al. [2010]). Both criminologists Rossmo and Rombouts [2008] and military experts U.S. Army [1994] have noted that partner locations are not too close to an observation location nor are they too far.[2] Note that having α, β actually increases the generality of our approach as users can always opt not to use them by setting $\alpha = 0$ and β to any number exceeding $\sqrt{M^2 + N^2}$. Given an integer $k > 0$, a *k-explanation* is an (α, β)-explanation of cardinality k or less. Often we will fix k; in this situation we will use the terms "k-explanation" and "explanation" interchangeably. Alternatively, another requirement that can be imposed on an explanation is *irredundancy*.

Definition 2.5. An explanation $\mathcal{E}$ is *irredundant* iff no strict subset of $\mathcal{E}$ is an explanation.

Intuitively, if we can remove any element from an explanation, and this action causes it to cease to be a valid explanation, we say the explanation is irredundant.

Example 2.1. Figure 1 shows a map of a drug plantation depicted in a 18×14 grid. The distance between grid squares is 100 meters. Observation set $\mathcal{O} = \{o_1, o_2, o_3, o_4, o_5\}$ represents the center of mass of the poppy fields. Based on an informant or from historical data, drug enforcement officials know that there is a drug laboratory located $150-320$ meters from the center mass of each field (*i.e.*, in a geospatial abduction problem, we can set $[\alpha, \beta] = [150, 320]$). Further, based on the terrain, the drug enforcement officials are able to discount certain areas (shown in black on Figure 1, a feasibility predicate can easily be set up accordingly). Based on Figure 1, the set $\{p_{40}, p_{46}\}$ is an explanation. The sets $\{p_{42}, p_{45}, p_{48}\}$ and $\{p_{40}, p_{45}\}$ are also explanations.

We now formally recall the definition of a GAP from Shakarian et al. [2010].

The k Spatial (α, β) *Explanation Problem (k-**SEP**).*
INPUT: Space $\mathcal{S}$, a set $\mathcal{O}$ of observations, a feasibility predicate feas, reals numbers $\alpha \geq 0$, $\beta > 0$, and natural number k.
OUTPUT: "Yes" if there exists an (α, β) explanation for $\mathcal{O}$ of size k, "no" otherwise.

Shakarian et al. [2010] shows this problem to be NP-complete based on a reduction from the known NP-complete problem *Geometric Covering by Discs* (GCD) seen in

[2]In the case of IED attacks, this is because the location around an IED attack is usually cordoned off and searched and the insurgents do not want their weapons caches to be found, thus leading to α. In contrast, the insurgents do not want their caches to be too far away as they then run the risk of detection at checkpoints and random search points while transporting munitions, leading to β.

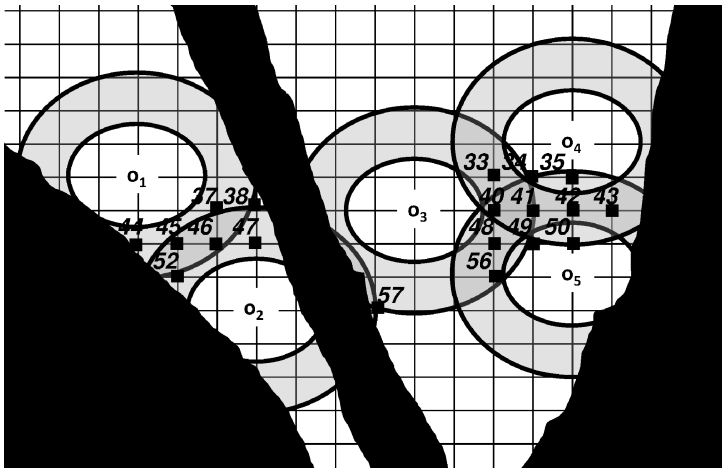

Fig. 1. Map of poppy fields for Example 2.1. For each labeled point p_i, the "p" is omitted for readability.

Johnson [1982], also known as the Euclidean m-center on points in Masuyama and Ibaraki [1981]. The problem is defined as follows.

Geometric Covering by Discs (GCD).
INPUT: A set P of integer-coordinate points in a Euclidean plane, positive integers $b > 0$ and $K < |P|$.
OUTPUT: "Yes" if there exists k discs of diameter b centered on points in P such that there is a disc covering each point in P, "no" otherwise.

As with most decision problems, we define the associated counting problem, #GCD, as the number of "yes" answers to the GCD decision problem. The result that follows, which is new, shows that #GCD is #P-complete and, moreover, that there is no fully polynomial random approximation scheme for #GCD unless NP equals the complexity class RP.[3]

LEMMA 2.1. *#GCD is #P-complete and has no FPRAS unless NP=RP.*

We can leverage the preceding result to derive a complexity result for the counting version of k-**SEP**.

THEOREM 2.1. *The counting version of k-**SEP** is #P-complete and has no FPRAS unless NP=RP.*

3. GEOSPATIAL ABDUCTION AS A TWO-PLAYER GAME

Throughout this article, we view geospatial abduction as a two-player game where an *agent* attempts to find an "explanation" for a set of observations caused by the *adversary* who wants to hide the explanation from the agent.

Each agent chooses a *strategy* which is merely a subset of $\mathcal{S}$. Though "strategy" and "observation" are defined identically, we use separate terms to indicate our intended use. In the IED example, the adversary's strategy is a set of points where to place his

[3] RP is the class of decision problems for which there is a randomized polynomial algorithm that, for any instance of the problem, returns "false" with probability 1 when the correct answer to the problem instance is false, and returns "true" with probability $(1 - \epsilon)$ for a small $\epsilon > 0$ when the correct answer to the problem instance is "true."

cache, while the agent's strategy is a set of points that he thinks hold the weapons caches. Throughout this article, we use $\mathcal{A}$ (respectively $\mathcal{B}$) to denote the strategy of the adversary (respectively agent).

Given a pair $(\mathcal{A}, \mathcal{B})$ of adversary-agent strategies, a reward function measures how similar the two sets are. The more similar, the better it is for the agent. As reward functions can be defined in many ways, we choose an axiomatic approach so that our framework applies to many different reward functions including ones that people may invent in the future.

Definition 3.1 (*Reward Function*). A *reward function* is any function $\mathbf{rf} : 2^{\mathcal{S}} \times 2^{\mathcal{S}} \rightarrow [0, 1]$ that for any k-explanation $\mathcal{A} \not\equiv \emptyset$ and set $\mathcal{B} \subseteq \mathcal{S}$, the function satisfies:

(1) If $\mathcal{B} = \mathcal{A}$, then $\mathbf{rf}(\mathcal{A}, \mathcal{B}) = 1$.
(2) For $\mathcal{B}, \mathcal{B}'$ then
$\mathbf{rf}(\mathcal{A}, \mathcal{B} \cup \mathcal{B}') \leq \mathbf{rf}(\mathcal{A}, \mathcal{B}) + \mathbf{rf}(\mathcal{A}, \mathcal{B}') - \mathbf{rf}(\mathcal{A}, \mathcal{B} \cap \mathcal{B}')$.

We now define the payoffs for the agent and adversary.

OBSERVATION 3.1. *Given adversary strategy $\mathcal{A}$, agent strategy $\mathcal{B}$, and reward function* **rf**, *the payoff for the agent is* $\mathbf{rf}(\mathcal{A}, \mathcal{B})$ *and the payoff for the adversary is* $-\mathbf{rf}(\mathcal{A}, \mathcal{B})$.

It is easy to see that for any reward function and pair $(\mathcal{A}, \mathcal{B})$, the corresponding game is a *zero-sum game* [Leyton-Brown and Shoham 2008]. Our complexity analysis assumes all reward functions are polynomially computable. All the specific reward functions we propose in this article satisfy this condition.

The basic intuition behind the reward function is that the more the strategy of the agent resembles that of the adversary, the closer the reward is to 1. Axiom 1 says that if the agent's strategy is the same set as adversary's, then the reward is 1. Axiom 2 says that adding a point to $\mathcal{B}$ cannot increase the reward to the agent if that point is already in $\mathcal{B}$, that is, double-counting of rewards is forbidden.

The following theorem tells us that every reward function is *submodular*, that is, the marginal benefit of adding additional points to the agent's strategy decreases as the cardinality of the strategy increases.

PROPOSITION 3.1 (SUBMODULARITY OF REWARD FUNCTIONS). *Every reward function is submodular, that is, if $\mathcal{B} \subseteq \mathcal{B}'$, and point $p \in \mathcal{S}$ such that $p \notin \mathcal{B}$ and $p \notin \mathcal{B}'$, then* $\mathbf{rf}(\mathcal{A}, \mathcal{B} \cup \{p\}) - \mathbf{rf}(\mathcal{A}, \mathcal{B}) \geq \mathbf{rf}(\mathcal{A}, \mathcal{B}' \cup \{p\}) - \mathbf{rf}(\mathcal{A}, \mathcal{B}')$.

Some readers may wonder why $\mathbf{rf}(\mathcal{A}, \emptyset) = 0$ is not an axiom. While this is true of many reward functions, there are reward functions where we may wish to penalize the agent for "bad" predictions. Consider the following reward function.

Definition 3.2 (*Penalizing Reward Function*). Given a distance *dist*, we define the *penalizing reward function*, $\mathbf{prf}^{dist}(\mathcal{A}, \mathcal{B})$, as follows.

$$\frac{1}{2} + \frac{|\{p \in \mathcal{A} | \exists p' \in \mathcal{B} \text{ s.t. } d(p, p') \leq dist\}|}{2 \cdot |\mathcal{A}|} - \frac{|\{p \in \mathcal{B} | \nexists p' \in \mathcal{A} \text{ s.t. } d(p, p') \leq dist\}|}{2 \cdot |\mathcal{S}|}$$

PROPOSITION 3.2. **prf** *is a valid reward function.*

Example 3.1. Consider Example 2.1 and the explanation $\mathcal{A} \equiv \{p_{40}, p_{46}\}$ (resembling actual locations of the drug labs), the set $\mathcal{B} \equiv \{p_{38}, p_{41}, p_{44}, p_{56}\}$ (representing areas that the drug enforcement officials wish to search), distance *dist* = 100 meters. There is only one point in $\mathcal{A}$ that is within 100 meters of a point in $\mathcal{B}$ (point p_{40}) and 3 points in $\mathcal{B}$ more than 100 meters from any point in $\mathcal{A}$ (points p_{38}, p_{44}, p_{56}). These relationships are shown visually in Figure 2. Hence, $\mathbf{prf}^{dist}(\mathcal{A}, \mathcal{B}) = 0.5 + 0.25 - 0.011 = 0.739$.

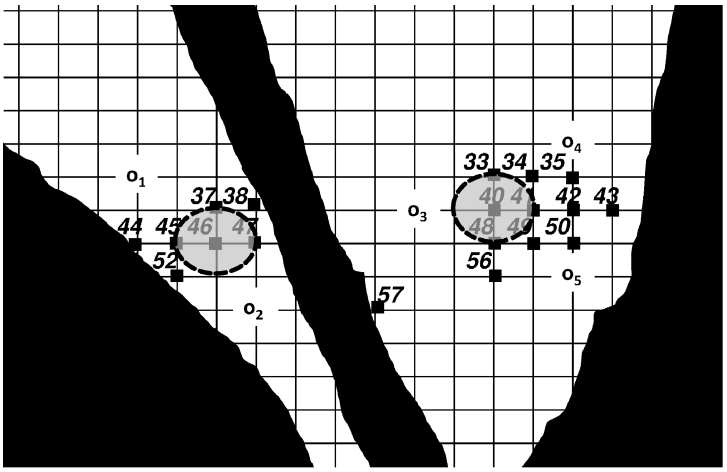

Fig. 2. Dashed circles encompass all feasible points within 100 meters from explanation $\{p_{40}, p_{45}\}$.

prf penalizes the agent if he poorly selects points in $\mathcal{S}$. The agent starts with a reward of 0.5. The reward increases if he finds points close to elements of $\mathcal{A}$; otherwise, it decreases.

A reward function is *zero-starting* if $\mathbf{rf}(\mathcal{A}, \emptyset) = 0$, that is, the agent gets no reward if he infers nothing.

Definition 3.3. A reward function, **rf**, is *monotonic* if (i) it is zero-starting and (ii) if $\mathcal{B} \subseteq \mathcal{B}'$ then $\mathbf{rf}(\mathcal{A}, \mathcal{B}) \leq \mathbf{rf}(\mathcal{A}, \mathcal{B}')$.

We now define several example monotonic reward functions.

The intuition behind the *cutoff reward function* **crf** is simple: For a given distance *dist* (the "cut-off" distance), if for every $p \in \mathcal{A}$, there exists $p' \in \mathcal{B}$ such that $d(p, p') \leq dist$, then p' is considered "close to" p.

Definition 3.4 (*Cutoff Reward Function*). Reward function based on a cut-off distance, *dist*.

$$\mathbf{crf}^{dist}(\mathcal{A}, \mathcal{B}) := \frac{card(\{p \in \mathcal{A} | \exists p' \in \mathcal{B} \; s.t. \; d(p, p') \leq dist\})}{card(\mathcal{A})}$$

The following proposition shows that the cutoff reward function is a valid, monotonic reward function.

PROPOSITION 3.3. **crf** *is a valid, monotonic reward function.*

Example 3.2. Consider Example 3.1. Here, $\mathbf{crf}^{dist}(\mathcal{A}, \mathcal{B})$ returns 0.5 as one element of $\mathcal{A}$ is within 100 meters of an element in $\mathcal{B}$.

By allowing a more general notion of "closeness" between points $p \in \mathcal{A}$ and $p' \in \mathcal{E}$, we are able to define another reward function, the *falloff reward function*, **frf**. This function provides the most reward if $p = p'$ but, unlike the somewhat binary **crf**, gently lowers this reward to a minimal zero as distances $d(p, p')$ grow.

Definition 3.5 (*Falloff Reward Function*). Reward function with value based on minimal distances between points.

$$\mathbf{frf}(\mathcal{A},\mathcal{B}) := \begin{cases} 0 & \text{if } \mathcal{B}=\emptyset \\ \sum_{p\in\mathcal{A}} \frac{1}{|\mathcal{A}|+\min_{p'\in\mathcal{B}}(d(p,p')^2)} & \text{otherwise} \end{cases}$$

with $d(p,p') := \sqrt{(p_x-p'_x)^2+(p_y-p'_y)^2}$. In this case, the agent's reward is inversely proportional to the square of the distance between points, as the search area required grows proportionally to the square of this distance.

PROPOSITION 3.4. **frf** *is a valid, monotonic reward function.*

In practice, an agent may assign different weights to points in $\mathcal{S}$ based on the perceived importance of their partner observations in $\mathcal{O}$. The "weighted reward function" **wrf** gives greater reward for being "closer" to points in $\mathcal{A}$ that have high weight than those with lower weights.

Definition 3.6 (*Weighted Reward Function*). Given weight function $W : \mathcal{S} \to \mathbb{R}^+$, and a cut-off distance *dist* we define the *weighted reward function* to be:

$$\mathbf{wrf}^{(W,dist)}(\mathcal{A},\mathcal{B}) := \frac{\sum_{\{p\in\mathcal{A}|\exists p'\in\mathcal{B}\ s.t.\ d(p,p')\le dist\}} W(p)}{\sum_{p'\in\mathcal{A}} W(p')}.$$

PROPOSITION 3.5. **wrf** *is a valid, monotonic reward function.*

It is easy to see that the weighted reward function is a generalization of the cutoff reward function where all weights are 1.

It is important to note that we have introduced reward functions axiomatically. There are numerous other reward functions that satisfy the axioms given in Definition 3.1 that can be defined in an application. There is no guarantee that the few specific instances of a reward function we have defined are the only good ones; application developers are welcome to use their own.

3.1. Incorporating Mixed Strategies

In this section, we introduce pdfs over strategies (or "mixed strategies" [Leyton-Brown and Shoham 2008]) and introduce the notion of "expected reward." We first present *explanation/strategy functions* which return an explanation (respectively strategy) of a certain size for a given set of observations.

Definition 3.7 (*Explanation/Strategy Function*). An *explanation (respectively strategy) function* is any function $\mathsf{ef} : 2^{\mathcal{S}} \times \mathbb{N} \to 2^{\mathcal{S}}$ (respectively $\mathsf{sf} : 2^{\mathcal{S}} \times \mathbf{N} \to 2^{\mathcal{S}}$) that, given a set $\mathcal{O} \subseteq \mathcal{S}$ and $k \in \mathbb{N}$, returns a set $\mathcal{E} \subseteq \mathcal{S}$ such that $\mathcal{E}$ is a k-sized explanation of $\mathcal{O}$ (respectively $\mathcal{E}$ is a k-sized subset of $\mathcal{S}$). Let **EF** be the set of all explanation functions.

Example 3.3. Following from Example 2.1, we shall define two functions $\mathsf{ef}_1, \mathsf{ef}_2$, which for set $\mathcal{O}$ (defined in Example 2.1) and $k \le 3$, give the following sets.

$$\mathsf{ef}_1(\mathcal{O},3) = \{p_{42}, p_{45}, p_{48}\}$$
$$\mathsf{ef}_2(\mathcal{O},3) = \{p_{40}, p_{46}\}$$

These sets may correspond to explanations from various sources. Perhaps they correspond to the answer of an algorithm that drug-enforcement officials use to solve GAPs. Conversely, they could also be the result of a planning session by the drug cartel to determine optimal locations for the drug labs.

In theory, the set of all explanation functions can be infinitely large; however, it makes no sense to look for explanations containing more points than $\mathcal{S}$, so we assume explanation functions are only invoked with $k \leq M \times N$.

A strategy function is appropriate for an agent who wants to select points resembling what the adversary selected, but is not required to produce an explanation. Our results typically do not depend on whether an explanation or strategy function is used (when they do, we point it out). Therefore, for simplicity, we use "explanation function" throughout the article. In our complexity results, we assume that explanation/strategy functions are computable in constant time.

Both the agent and the adversary do not know the explanation function (where is the adversary going to put his weapons caches? Where will US forces search for them?) in advance. Thus, they use a pdf over explanation functions to estimate their opponent's behavior, yielding a "mixed" strategy.

Definition 3.8 (*Explanation Function Distribution*). Given a space $\mathcal{S}$, real numbers α, β, feasibility predicate feas, and an associated set of explanation functions $\mathbf{EF}$, an *explanation function distribution* is a finitary[4] probability distribution $\mathsf{efd} : \mathbf{EF} \rightarrow [0, 1]$ with $\sum_{\mathsf{ef} \in \mathbf{EF}} \mathsf{efd}(\mathsf{ef}) = 1$. Let $\mathbf{EFD}$ be a set of explanation function distributions.

We use $|\mathsf{efd}|$ to denote the cardinality of the set $\mathbf{EF}$ associated with efd.

Example 3.4. Following from Example 3.3, we shall define the explanation function distribution efd_{drug} that assigns a uniform probability to explanation functions in the set $\mathsf{ef}_1, \mathsf{ef}_2$ (*i.e.*, $\mathsf{efd}_{drug}(\mathsf{ef}_1) = 0.5$).

We now define an "expected reward" that takes into account these mixed strategies specified by explanation function distributions.

Definition 3.9 (*Expected Reward*). Given a reward function $\mathbf{rf}$, and explanation function distributions $\mathsf{efd}_{adv}, \mathsf{efd}_{ag}$, the *expected reward* is the function $\mathsf{EXR}^{\mathbf{rf}} : \mathbf{EFD} \times \mathbf{EFD} \rightarrow [0, 1]$ defined as follows.

$$\mathsf{EXR}^{\mathbf{rf}}(\mathsf{efd}_{adv}, \mathsf{efd}_{ag}) = \sum_{\mathsf{ef}_{adv} \in \mathbf{EF}_{adv}} \left(\mathsf{efd}_{adv}(\mathsf{ef}_{adv}) \cdot \sum_{\mathsf{ef}_{ag} \in \mathbf{EF}_{ag}} \mathsf{efd}_{ag}(\mathsf{ef}_{ag}) \cdot \mathbf{rf}(\mathsf{ef}_{adv}, \mathsf{ef}_{ag}) \right)$$

However, in this article, we will generally not deal with expected reward directly, but two special cases (expected adversarial detriment and expected agent benefit) in which the adversary's and agent's strategies are *not* mixed respectively. We explore these two special cases in the next two sections.

4. SELECTING A STRATEGY FOR THE ADVERSARY

In this section, we study how an adversary would select points (set $\mathcal{A}$) in the space he would use to cause observations $\mathcal{O}$. For instance, in the IED example, the adversary needs to select $\mathcal{A}$ and O so that $\mathcal{A}$ is an explanation for $\mathcal{O}$. We assume the adversary has a probabilistic model of the agent's behavior (an explanation function distribution) and that he wants to eventually find an explanation (e.g., where to put his weapons caches). Hence, though he can use expected reward to measure how close the agent will be to his explanation, only the agent's strategy is mixed. The adversary's actions are concrete. Hence, we introduce a special case of expected reward: expected adversarial detriment.

[4]That is, efd assigns nonzero probabilities to only finitely many explanation functions.

Definition 4.1 (*Expected Adversarial Detriment*). Given any reward function **rf** and explanation function distribution efd, the *expected adversarial detriment* is the function $\mathsf{EXD}^{\mathbf{rf}} : \mathbf{EFD} \times 2^{\mathcal{S}} \rightarrow [0, 1]$ defined as follows.

$$\mathsf{EXD}^{\mathbf{rf}}(\mathsf{efd}, \mathcal{A}) = \sum_{\mathsf{ef} \in \mathbf{EF}} \mathbf{rf}(\mathcal{A}, \mathsf{ef}(\mathcal{O}, k)) \cdot \mathsf{efd}(\mathsf{ef})$$

Intuitively, the expected adversarial detriment is the expected number of partner locations the agent may uncover if efd is correct. Consider the following example.

Example 4.1. Following from the previous examples, suppose the drug cartel is planning three drug labs. Suppose they have information that drug-enforcement agents will look for drug labs using efd_{drug} (Example 3.4). One suggestion the adversary may consider is to put the labs at locations p_{41}, p_{52} (see Figure 1). Note that this explanation is optimal with respect to cardinality. With $dist$ = 100 meters, they wish to compute $\mathsf{EXD}^{\mathbf{crf}}(\mathsf{efd}_{drug}, \{p_{41}, p_{52}\})$. We first need to find the reward associated with each explanation function (see Example 3.3).

$$\mathbf{crf}^{dist}(\{p_{41}, p_{52}\}, \mathsf{ef}_1(\mathcal{O}, 3)) = 1$$
$$\mathbf{crf}^{dist}(\{p_{41}, p_{52}\}, \mathsf{ef}_2(\mathcal{O}, 3)) = 0.5$$

Thus, $\mathsf{EXD}^{\mathbf{crf}}(\mathsf{efd}_{drug}, \{p_{41}, p_{52}\}) = 0.5 \cdot 1 + 0.5 \cdot 0.5 = 0.75$. Hence, this is probably not the best location for the cartel to position the labs with respect to **crf** and efd, because the expected adversarial detriment of the drug-enforcement agents is large.

The expected adversarial detriment is a quantity that the adversary would seek to minimize. This is now defined as an *optimal adversarial strategy* next.

Definition 4.2 (*Optimal Adversarial Strategy*). Given a set of observations $\mathcal{O}$, natural number k, reward function **rf**, and explanation function distribution efd, an *optimal adversarial strategy* is a k-sized explanation $\mathcal{A}$ for $\mathcal{O}$ such that $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{efd}, \mathcal{A})$ is minimized.

4.1. The Complexity of Finding an Optimal Adversarial Strategy

In this section, we formally define the Optimal Adversary Strategy (OAS) problem and study its complexity.

OAS Problem.
INPUT: Space $\mathcal{S}$, feasibility predicate feas, real numbers α, β, set of observations $\mathcal{O}$, natural number k, reward function **rf**, and explanation function distribution efd.
OUTPUT: Optimal adversarial strategy $\mathcal{A}$.

We show that the known NP-hard problem *Geometric Covering by Discs* (see Section 2) is polynomially reducible to OAS, which establishes NP-hardness.

THEOREM 4.1. *OAS is NP-hard.*

The proof of the previous theorem yields two insights. First, OAS is NP-hard even if the reward function is monotonic (or antimonotonic). Second, OAS remains NP-hard even if the cardinality of **EF** is small; in the construction we only have one explanation function. Thus, we cannot simply pick an "optimal" function from **EF**. To show an upper bound, we define OAS-DEC to be the decision problem associated with OAS. If the reward function is computable in polynomial time, OAS-DEC is in NP.

OAS-DEC.
INPUT: Space $\mathcal{S}$, feasibility predicate feas, real numbers α, β, set of observations $\mathcal{O}$, natural number k, reward function **rf**, explanation function distribution efd, and number $R \in [0, 1]$.
OUTPUT: "Yes" if there exists an adversarial strategy $\mathcal{A}$ such that $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{efd}, \mathcal{A}) \leq R$, "no" otherwise.

THEOREM 4.2. *If the reward function is computable in PTIME, then OAS-DEC is NP-complete.*

Suppose we have an NP oracle that can return an optimal adversarial strategy; let's call it $\mathcal{A}$. Quite obviously, this is the *best response* of the adversary to the mixed strategy of the agent. Now, how does the agent respond to such a strategy? If we were to assume that such a solution were unique, then the agent would simply have to find an strategy $\mathcal{B}$ such that $\mathbf{rf}(\mathcal{A}, \mathcal{B})$ is maximized. This would be a special case of the problem we discuss in Section 5. However, this is not necessarily the case. A natural way to address this problem is to create a uniform probability distribution over all optimal adversarial strategies and optimize the expected reward, again a special case of what is to be discussed in Section 5. However, obtaining the set of explanations is not an easy task. Even if we had an easy way to exactly compute an optimal adversarial strategy, finding *all* such strategies is an even more challenging problem. In fact, it is at least as hard as the counting version of GCD, which we already have shown #P-hard and difficult to approximate. This is shown in the following theorem.

THEOREM 4.3. *Finding the set of all adversarial optimal strategies that provide a "yes" answer to OAS-DEC is #P-hard.*

4.2. Preprocessing and Naive Approach

In this section, we present several algorithms to solve OAS. We first present a simple routine for preprocessing followed by a naive enumeration-based algorithm.

We use Δ to denote the maximum number of partners per observation and f to denote the maximum number of observations supported by a single partner. In general, Δ is bounded by $\pi(\beta^2 - \alpha^2)$, but may be lower depending on the feasible points in $\mathcal{S}$. Likewise, f is bounded by $\min(|\mathcal{O}|, \Delta)$ but may be much smaller depending on the sparseness of the observations.

Preprocessing Procedure. Given a space $\mathcal{S}$, a feasibility predicate feas, real numbers $\alpha \geq 0, \beta > 0$, and a set $\mathcal{O}$ of observations, we create two lists (similar to a standard inverted index) as follows.

— *Matrix M.* M is an array of size $\mathcal{S}$. For each feasible point $p \in \mathcal{S}$, $M[p]$ is a list of pointers to observations. $M[p]$ contains pointers to each observation o such that $\mathsf{feas}(p)$ is true and such that $d(o, p) \in [\alpha, \beta]$.
— *List L.* List L contains a pointer to position $M[p]$ in the array M iff there exists an observation $o \in \mathcal{O}$ such that $\mathsf{feas}(p)$ is true and such that $d(o, p) \in [\alpha, \beta]$.

It is easy to see that we can compute M and L in $O(|\mathcal{O}| \cdot \Delta)$ time. The next example shows how M, L apply to our running drug example.

Example 4.2. Consider our running example concerning the location of drug laboratories that started with Example 2.1. The set L consists of $\{p_1, \ldots, p_{67}\}$. The matrix M returns lists of observations that can be associated with each feasible point. For example, $M(p_{40}) = \{o_3, o_4, o_5\}$ and $M(p_{46}) = \{o_1, o_2\}$.

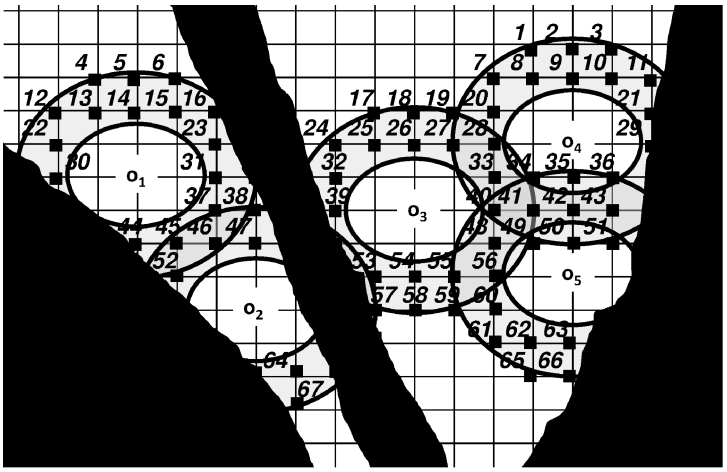

Fig. 3. Set L of all possible partners for our drug laboratory location example.

Naive Approach. After preprocessing, a straightforward exact solution to OAS would be to enumerate all subsets of L that have a cardinality less than or equal to k. Let us call this set L^*. Next, we eliminate all elements of L^* that are not valid explanations. Finally, for each element of L^*, we compute the expected adversarial detriment and return the element of L^* for which this value is the least. Clearly, this approach is impractical as the cardinality of L^* can be very large. Further, this approach does not take advantage of the specific reward functions. We now present Mixed Integer Linear Programs (MILPs) for **wrf** and **frf** and later look at ways to reduce the complexity of solving these MILPs.

4.3. Mixed Integer Linear Programs for OAS under wrf, crf, frf

We present Mixed Integer Linear Programs (MILPs) to solve OAS exactly for some specific reward functions. First, we present a mixed integer linear program for the reward function **wrf**. Later, in Section 4.4, we show how to improve efficiency (while maintaining optimality) by reducing the number of variables in the MILP. Note that these constraints can also be used for **crf** as **wrf** generalizes **crf**. We also define a MILP for the **frf** reward function.

While these mixed integer programs may appear nonlinear, Proposition 4.4 gives a simple transformation to standard linear form. For readability, we define the MILPs before discussing this transformation.

Definition 4.3 (**wrf** *MILP*). Given real number $dist > 0$ and weight function W, associate a constant w_i with the weight $W(p_i)$ of each point $p_i \in L$. Next, for each $p_i \in L$ and $\mathsf{ef}_j \in \mathbf{EF}$, let constant $c_{i,j} = 1$ iff $\exists p' \in \mathsf{ef}(\mathcal{O}, k)$ such that $d(p', p_i) \leq dist$ and 0 otherwise. Finally, associate an integer-valued variable X_i with each $p_i \in L$.
Minimize:

$$\sum_{\mathsf{ef}_j \in \mathbf{EF}} \left(\mathsf{efd}(\mathsf{ef}_j) \cdot \sum_{p_i \in L} \left(X_i \cdot \frac{w_i \cdot c_{i,j}}{\sum_{p_i \in L} w_i \cdot X_i} \right) \right)$$

subject to:

(1) $X_i \in \{0, 1\}$;
(2) Constraint $\sum_{p_i \in L} X_i \leq k$;

(3) For each $o_j \in \mathcal{O}$, add constraint $\sum_{p_i \in L_{d(o_j,p_i)\in[\alpha,\beta]}} X_i \geq 1$.

Example 4.3. Continuing from Examples 4.1 and 4.2, suppose the drug cartel wishes to produce an adversarial strategy $\mathcal{A}$ using **wrf**. Consider the case where we use **crf**, $k \leq 3$, and *dist* = 100 meters as before (see Example 4.1). Clearly, there are 67 variables in these constraints, as this is the cardinality of set L (as per Example 4.2). The constants $c_{i,1}$ are 1 for elements in the set $\{p_{35}, p_{40}, p_{41}, p_{42}, p_{43}, p_{44}, p_{45}, p_{46}, p_{49}, p_{49}, p_{50}, p_{52}, p_{56}\}$ (and 0 for all others). The constants $c_{i,2}$ are 1 for elements in the set $\{p_{33}, p_{37}, p_{40}, p_{41}, p_{45}, p_{46}, p_{47}, p_{48}\}$ (and 0 for all others).

We can create a MILP for **frf** as follows.

Definition 4.4 (*frf MILP*). For each $p_i \in L$ and $\mathsf{ef}_j \in \mathbf{EF}$, let constant $c_{i,j} = \min_{p' \in \mathsf{ef}_{(\mathcal{O},k)}}(d(p_i, p')^2)$. Associate an integer-valued variable X_i with each $p_i \in L$.
Minimize:

$$\sum_{\mathsf{ef}_j \in \mathbf{EF}} \left(\mathsf{efd}(\mathsf{ef}_j) \cdot \sum_{p_i \in L} \left(X_i \cdot \frac{1}{c_{i,j} + \sum_{p_i \in L} X_i} \right) \right)$$

subject to:

(1) $X_i \in \{0, 1\}$;
(2) Constraint $\sum_{p_i \in L} X_i \leq k$;
(3) For each $o_j \in \mathcal{O}$, add constraint $\sum_{p_i \in L_{d(o_j,p_i)\in[\alpha,\beta]}} X_i \geq 1$.

The following theorem tells us that solving the preceding MILPs correctly yields a solution for the OAS problem under both **wrf** or **frf**.

PROPOSITION 4.1. *Suppose $\mathcal{S}$ is a space, $\mathcal{O}$ is an observation set, real numbers $\alpha \geq 0$, $\beta > 0$, and suppose the* **wrf** *and* **frf** *MILPs are defined as earlier.*

(*1*) *Suppose $\mathcal{A} \equiv \{p_1, \ldots, p_n\}$ is a solution to OAS with* **wrf***(respectively* **frf***). Consider the assignment that assigns* 1 *to each $X_1, \ldots, X_n$ corresponding to the p_i's and* 0 *otherwise. This assignment is an optimal solution to the MILP.*
(*2*) *Given the solution to the constraints, if for every $X_i = 1$, we add point p_i to set $\mathcal{A}$, then $\mathcal{A}$ is a solution to OAS with* **wrf***(respectively* **frf***).*

Setting up either set of constraints can be performed in polynomial time, where computing the $c_{i,j}$ constants is the dominant operation.

PROPOSITION 4.2. *Setting up the* **wrf**/**frf** *constraints can be accomplished in $O(|\mathbf{EF}| \cdot k \cdot |\mathcal{O}| \cdot \Delta)$ time (provided the weight function W can be computed in constant time).*

The number of variables for either set of constraints is related to the size of L, which depends on the number of observations, spacing of $\mathcal{S}$, and α, β.

PROPOSITION 4.3. *The* **wrf**/**frf** *constraints have $O(|\mathcal{O}| \cdot \Delta)$ variables and $1 + |\mathcal{O}|$ constraints.*

The MILPs for **wrf** and **frf** appear nonlinear as the objective function is fractional. However, as the denominator is nonzero and strictly positive, the Charnes-Cooper

transformation [Charnes and Cooper 1962] allows us to quickly (in the order of number of constraints multiplied by the number of variables) transform the constraints into a purely integer linear form. Many linear and integer linear program solvers include this transformation in their implementation.

PROPOSITION 4.4. *The* **wrf**/**frf** *constraints can be transformed into a purely linear integer form in* $O(|\mathcal{O}|^2 \cdot \Delta)$ *time.*

We note that a linear relaxation of any of the aforesaid three constraints can yield a lower bound on the objective function in $O(|L|^{3.5})$ time.

PROPOSITION 4.5. *Given the constraints of Definition* 4.3 *or Definition* 4.4*, if we consider the linear program formed by setting all* X_i *variables to be in* $[0, 1]$*, then the value returned by the objective function will be a lower bound on the value returned by the objective function for the mixed integer linear constraints, and this value can be obtained in* $O(|\mathcal{O}|^{3.5} \cdot \Delta^{3.5})$ *time.*

Likewise, if we solve the mixed integer linear program with a reduced number of variables, we are guaranteed that the solution will cause the objective function to be an upper bound for the original set of constraints.

PROPOSITION 4.6. *Consider the MILPs in Definition 4.3 and Definition 4.4. Suppose* $L' \subset L$ *and every variable* X_i *associated with some* $p_i \in L'$ *is set to* 0*. The resulting solution is an upper bound on the objective function for the constraints solved on the full set of variables.*

4.4. Correctly Reducing the Number of Variables for crf

As the complexity of solving MILPs is closely related to the number of variables in the MILP, the goal of this section is to reduce the number of variables in the MILP associated before with the **crf** reward function. We note that all results in this section apply only for the **crf** reward function. In this section, we show that if we can find a certain type of explanation called a δ-core optimal explanation, then we can "build-up" an optimal adversarial strategy in polynomial time. It also turns out that finding these special explanations can be accomplished using an MILP which will often have significantly fewer variables than the MILPs of the last section. First, we consider the **wrf** constraints applied to **crf** which is a special case of **wrf**. The objective function for this case is

$$\sum_{\mathsf{ef}_j \in \mathbf{EF}} \left(\mathsf{efd}(\mathsf{ef}_j) \cdot \sum_{p_i \in L} \left(X_i \cdot \frac{c_{i,j}}{\sum_{p_i \in L} X_i} \right) \right),$$

where for each $p_i \in L$ and $\mathsf{ef}_j \in \mathbf{EF}$, $c_{i,j} = 1$ iff $\exists p' \in \mathsf{ef}_j(\mathcal{O}, k)$ such that $d(p', p_i) \leq dist$ and 0 otherwise. If we rearrange the objective function, we see that with each X_i variable associated with point $p_i \in L$, there is an associated constant $const_i$.

$$const_i = \sum_{\mathsf{ef}_j \in \mathbf{EF}} \mathsf{efd}(\mathsf{ef}_j) \cdot c_{i,j}$$

This lets us rewrite the objective function as

$$\frac{\sum_{p_i \in L} X_i \cdot const_i}{\sum_{p_i \in L} X_i}.$$

Example 4.4. Continuing from Example 4.3, $const_i = 0.5$ for the following elements: $\{p_{33}, p_{35}, p_{37}, p_{42}, p_{43}, p_{44}, p_{47}, p_{49}, p_{50}, p_{52}, p_{56}\}$; $const_i = 1$ for these elements: $\{p_{40}, p_{41}, p_{45}, p_{46}, p_{48}\}$, and 0 for all others.

In many covering problems where we wish to find a cover of minimal cardinality, we could reduce the number of variables in the integer program by considering equivalent covers as duplicate variables. However, for OAS, this technique cannot be easily applied. The reason for this is because an optimal adversarial explanation is not necessarily irredundant (see Definition 2.5). Consider the following. Suppose we wish to find an optimal adversarial strategy of size k. Let P be an irredundant cover of size $k-1$. Suppose there is some element $p' \in P$ that covers only one observation o'. Hence, there is no $p \in P - \{p'\}$ that covers o' by the definition of an irredundant cover. Suppose there is also some $p'' \notin P$ that also covers o'. Now, let $m = \sum_{p_i \in P - p'} const_i$. In our construction of an example solution to OAS that is not irredundant, we let $const'$ be the value associated with both p' and p''. Consider the scenario where $const' < \frac{m}{k-2}$. Suppose by way of contradiction that the optimal irredundant cover is also the optimal adversarial strategy. Then, by the definition of an optimal adversarial strategy we know that the set P is more optimal than $P \cup \{p''\}$. This would mean that $\frac{m+const'}{k-1} < \frac{m+2 \cdot const'}{k}$. This leads us to infer that $m < const' \cdot (k-2)$, which clearly contradicts $const' < \frac{m}{k-2}$. It is clear that a solution to OAS need not be irredundant.

Even though an OAS is not necessarily irredundant, we are able to reduce the size of the set L by looking at certain aspects of an OAS. Our intuition is that each OAS contains a *core* explanation which has fewer redundant elements than the OAS and low values of *const* for each element in that set. Once this type of explanation is found, we can build an optimal adversarial strategy in polynomial time. First, we define a *core* explanation.

Definition 4.5 (*Core Explanation*). Given an observation set $\mathcal{O}$ and set L of possible partners, an explanation $\mathcal{E}_{core}$ is a *core explanation* iff for any $p_i \in \mathcal{E}_{core}$, there does not exist $p_j \in L$ such that:

(1) $\forall o \in \mathcal{O}$ if o, p_i are partners, then o, p_j are also partners.
(2) $const_j < const_i$.

We now show that any optimal adversarial strategy contains a subset that is a core explanation.

THEOREM 4.4. *If $\mathcal{A}$ is an optimal adversarial strategy, there exists a core explanation $\mathcal{E}_{core} \subseteq \mathcal{A}$.*

Example 4.5. Continuing from Example 4.4, consider the set $\mathcal{A} \equiv \{p_{34}, p_{38}, p_{57}\}$ (which would correspond to drug lab locations as planned by the cartel). Later, we show that this is an optimal adversarial strategy (the expected adversarial detriment associated with $\mathcal{A}$ is 0). Consider the subset p_{34}, p_{38}. As p_{34} explains observations o_3, o_4, o_5, and p_{38} explains observations o_1, o_2, this set is also an explanation. Obviously, it is of minimal cardinality. Hence, the set $\{p_{34}, p_{38}\}$ is a **core explanation** of $\mathcal{A}$.

Suppose we have an oracle that, for a given k, $\mathcal{O}$, and efd returns a core explanation $\mathcal{E}_{core}$ that is guaranteed to be a subset of the optimal adversarial strategy associated with k, $\mathcal{O}$, and efd. The following theorem says we can find the optimal adversarial strategy in polynomial time. The key intuition is that we need not concern ourselves with covering the observations as $\mathcal{E}_{core}$ is an explanation. The algorithm BUILD-STRAT follows from this theorem.

ALGORITHM 1: BUILD-STRAT

INPUT: Partner list L, core explanation $\mathcal{E}_{core}$, natural number k, explanation function distribution efd

OUTPUT: Optimal adversarial strategy $\mathcal{A}$

(1) If $|\mathcal{E}_{core}| = k$, return $\mathcal{E}_{core}$
(2) Set $\mathcal{A} = \mathcal{E}_{core}$. Let $k' = |\mathcal{E}_{core}|$
(3) Sort the set $L - \mathcal{E}_{core}$ by $const_i$. Let $L' = \{p_1, \ldots, p_{k-k'}\}$ be the $k - k'$ elements of this set with the lowest values for $const_i$, in ascending order
(4) For each $p_i \in L'$ let P_i be the set $\{p_1, \ldots, p_i\}$
(5) For each P_i let $S_i = \sum_{j \leq i} const_j$
(6) Let $ans = \min_{p_i \in L'}(\{\frac{k' \cdot \mathsf{EXD}^{\mathbf{rf}}(\mathsf{efd}, \mathcal{E}_{core}) + S_i}{k' + i}\})$
(7) Let P_{ans} be the P_i associated with ans
(8) If $ans \geq \mathsf{EXD}^{\mathbf{rf}}(\mathsf{efd}, \mathcal{E}_{core})$, return $\mathcal{E}_{core}$, else return $\mathcal{E}_{core} \cup P_{ans}$

THEOREM 4.5. *If there is an oracle that for any given k, $\mathcal{O}$, and efd returns a core explanation $\mathcal{E}_{core}$ that is guaranteed to be a subset of the optimal adversarial strategy associated with k, $\mathcal{O}$, and efd, then we can find an optimal adversarial strategy in $O(\Delta \cdot |\mathcal{O}| \cdot \log(\Delta \cdot |\mathcal{O}|) + (k - |\mathcal{E}_{core}|)^2)$ time.*

We now introduce the notion of δ-core optimal. Intuitively, this is a core explanation of cardinality exactly δ that is optimal with respect to the expected adversarial detriment compared to all other core explanations of that cardinality.

Definition 4.6. Given an integer $\delta > 0$, an explanation distribution function efd, and a reward function **rf**, a core explanation $\mathcal{E}_{core}$ is *δ-core optimal* iff:

— $|\mathcal{E}_{core}| = \delta$.
— There does not exist another core explanation $\mathcal{E}'_{core}$ of cardinality exactly δ such that $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{efd}, \mathcal{E}'_{core}) < \mathsf{EXD}^{\mathbf{rf}}(\mathsf{efd}, \mathcal{E}_{core})$.

We now define some subsets of the set L that are guaranteed to contain core explanations and δ-core optimal explanations as well. In practice, these sets will be much smaller than L and will be used to create an MILP of reduced size.

Definition 4.7 (*Reduced Partner Set*). Given observations $\mathcal{O}$ and set of possible partners L, we define the reduced partner set L^{**} as follows.

$$L^{**} \equiv \{p_i \in L | \nexists p_j \in L \text{ such that } (const_j < const_i) \wedge (\forall o \in \mathcal{O} \text{ such that } o, p_i \text{ are partners, } o, p_j \text{ are also partners})\}$$

We define L^* as follows.

$$L^* \equiv \{p_i \in L^{**} | \nexists p_j \in L^{**} \text{ such that } (const_j = const_i) \wedge (\forall o \in \mathcal{O} \text{ such that } o, p_i \text{ are partners, } o, p_j \text{ are also partners})\}$$

LEMMA 4.6. *(1) If explanation $\mathcal{E}$ is a core explanation, then $\mathcal{E} \subseteq L^{**}$.*
(2) If explanation $\mathcal{E}$ is δ-core optimal, then $\mathcal{E} \subseteq L^{**}$.
(3) If for some natural number δ, there exists an explanation of size δ, then there exists a δ-core optimal explanation $\mathcal{E}$ such that $\mathcal{E} \subseteq L^*$.

The reduced partner set can be computed in polynomial time. We also note that under the assumption that $|\mathcal{O}| << |L|$, which we have found true in practice, determining

Table I. The Set L Partitioned by $const_i$ and Supported Observations

Supported Observations	$const_i = 0$	$const_i = 0.5$	$const_i = 1$
o_1	$p_4 - p_6$, $p_{12} - p_{16}$, $p_{22} - p_{23}$, $p_{30} - p_{31}$	p_{44}	
o_1, o_2	p_{38}	p_{37}, p_{52}	p_{45}, p_{46}
o_2	p_{64}, p_{67}	p_{47}	
o_2, o_3	p_{57}		
o_3	$p_{17} - p_{19}$, $p_{24} - p_{26}$, p_{32}, p_{39}, $p_{58} - p_{59}$		
o_3, o_4	$p_{27} - p_{28}$	p_{33}	
o_4	$p_1 - p_3$, $p_7 - p_{11}$, $p_{20} - p_{21}$, p_{29}, p_{51}	p_{50}	
o_3, o_4, o_5	p_{34}, $p_{53} - p_{54}$	p_{49}	$p_{40} - p_{41}$
o_5	p_{36}, $p_{60} - p_{66}$	p_{35}	
o_4, o_5		$p_{42} - p_{43}$	
o_3, o_5	p_{55}	p_{56}	p_{48}

the set L^{**} or L^* can be accomplished faster (in terms of time complexity) than solving even a relaxation of the original MILP.

PROPOSITION 4.7. *Given set L, set L^* and L^{**} can be found in $O(|L|^2 \cdot |\mathcal{O}|^2)$ time.*

Example 4.6. Let us continue from Example 4.5. Based on preprocessing and the computation of $const_i$, we can easily produce the data of Table I in polynomial time. Based on this, we obtain a *reduced partner set* $L^* \equiv \{p_{34}, p_{38}, p_{57}\}$.

Next, the following lemma tells us that an OAS must contain a core explanation that is δ-core optimal.

LEMMA 4.6. *Given an optimal adversarial strategy $\mathcal{A}$, there exists some $\delta \leq |\mathcal{A}|$ such that there is a δ-core optimal explanation that is a subset of $\mathcal{A}$ (using the **crf** reward function).*

Thus, if we can find the δ-core optimal explanation that is contained in an OAS, we can then find the OAS. If we know δ, such an explanation can be found using an MILP. We now present a set of integer linear constraints to find a *δ-core optimal* explanation. Of course we can easily adopt the constraints of the previous section, but this would offer us no improvement in performance. We therefore create an MILP that should have a significantly smaller number of variables in most cases.

To create this MILP, we take a given set of possible partners L and calculate the set L^* (the reduced partner set), which often will have a cardinality much smaller than L. Next, we use L^* to form a new set of constraints to find a δ-core optimal explanation. We now present these δ-core constraints. Notice that the cardinality requirement in these constraints is "=" and not "$\leq$". This is because Lemma 4.6 ensures a core explanation that is δ-core optimal, meaning that the core explanation must have cardinality exactly δ. This also allows us to eliminate variables from the denominator of the objective function, as the denominator must equal δ as well.

Definition 4.8 (*δ-core MILP*). Given parameter δ and reduced partner set L^*, we define the δ-core constraints by first associating a variable X_i with each point $p_i \in L^*$, then solving:
Minimize:

$$\frac{1}{\delta} \sum_{p_i \in L^*} X_i \cdot const_i$$

subject to:

(1) $X_i \in \{0, 1\}$.
(2) Constraint $\sum_{p_i \in L} X_i = \delta$.
(3) For each $o_j \in \mathcal{O}$, add constraint $\sum_{p_i \in L^* d(o_j, p_i) \in [\alpha, \beta]} X_i \geq 1$.

Example 4.7. Using set L^* from Example 4.6, we can create δ-core constraints as follows:
Minimize:

$$\frac{1}{\delta}(X_{34} \cdot const_{34} + X_{38} \cdot const_{38} + X_{57} \cdot const_{57})$$

subject to:

(1) $X_{34}, X_{38}, X_{57} \in \{0, 1\}$
(2) $X_{34} + X_{38} + X_{57} = \delta$
(3) $X_{38} \geq 1$ (for observation o_1)
(4) $X_{38} + X_{57} \geq 1$ (for observation o_2)
(5) $X_{34} + X_{57} \geq 1$ (for observation o_3)
(6) $X_{34} \geq 1$ (for observations o_4, o_5)

In the worst case, the set $L^* \equiv L$. Hence, we can assert the following:

PROPOSITION 4.8. *The δ-core constraints require $O(\Delta \cdot |\mathcal{O}|)$ variables and $1 + |\mathcal{O}|$ constraints.*

PROPOSITION 4.9. *Given δ-core constraints:*

(*1*) *Given set δ-core optimal explanation $\mathcal{E}_{core} \equiv \{p_1, \ldots, p_n\}$, if variables $X_1, \ldots, X_n$—corresponding with elements in $\mathcal{A}$—are set to 1 and the rest of the variables are set to 0, the objective function of the constraints will be minimized.*
(*2*) *Given the solution to the constraints, if for every $X_i = 1$, we add point p_i to set $\mathcal{E}_{core}$, then $\mathcal{E}_{core}$ is a δ-core optimal solution.*

We now have all the pieces required to leverage core explanations and reduced partner sets to find an optimal adversarial strategy. By Theorem 4.5, we know that any optimal adversarial strategy must have a core explanation. Further, by Lemma 4.6, such a core explanation is δ-core optimal. Using a (usually) much smaller mixed integer linear program, we can find such an explanation. We can then find the optimal adversarial strategy in polynomial time using BUILD STRAT. Though we do not know what δ is, we know it must be in the range $[1, k]$. Further, using a relaxation of the OPT-KSEP-IPC constraints for solving geospatial abduction problems (as presented in Shakarian et al. [2010]), we can easily obtain a lower bound tighter than 1 on δ. Hence, if we solve k such (most likely small) mixed integer linear programs, we are guaranteed that at least one of them must be a core explanation for an optimal adversarial strategy. We note that these k MILPs can be solved in parallel (and the following k instances of BUILD-STRAT can also be run in parallel as well). An easy comparison of the results of the parallel processes would be accomplished at the end. As L^* is likely to be significantly smaller than L, this could yield a significant reduction in complexity. Furthermore, various relaxations of this technique can be used (e.g., only using one value of δ).

Example 4.8. Continuing from Example 4.7, where the cartel members are attempting to find an OAS to best position drug laboratories, suppose they used the

relaxation of OPT-KSEP-IPC (from Shakarian et al. [2010]) to obtain a lower bound on the cardinality of an explanation and found it to be 2. With $k = 3$, they would solve two MILPs of the form of Example 4.7: one with $\delta = 2$ and one with $\delta = 3$. The solution to the first MILP would set X_{34} and X_{38} both to 1 while the second MILP would set X_{34}, X_{38}, and X_{57} all to 1. As the expected adversarial detriment for both solutions is 0, they are both optimal and running BUILD-STRAT is not necessary. Either $\{p_{34}, p_{38}\}$ or $\{p_{34}, p_{38}, p_{57}\}$ can be returned as an OAS.

5. FINDING A COUNTER-ADVERSARY STRATEGY

Now that we have examined ways in which the adversary can create a strategy based on probabilistic knowledge of the agent, we consider how the agent can devise an "optimal" strategy to counter the adversary. As before, we use a special case of expected reward (Definition 3.1 from Section 3.9).

Definition 5.1 (*Expected Agent Benefit*). Given a reward function **rf** and explanation function distribution efd, the *expected agent benefit* is the function $\mathsf{EXB}^{\mathbf{rf}} : 2^{\mathcal{S}} \times \mathbf{EFD} \rightarrow [0, 1]$ defined as follows.

$$\mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd}) = \sum_{\mathsf{ef} \in \mathbf{EF}} \mathbf{rf}(\mathsf{ef}(\mathcal{O}, k), \mathcal{B}) \cdot \mathsf{efd}(\mathsf{ef})$$

Example 5.1. Following from Examples 2.1 and 3.4, suppose drug-enforcement agents have information that the cartel is placing drug labs according to efd_{drug}. (Such information could come from multiple runs of the GREEDY-KSEP-OPT2 algorithm of Shakarian et al. [2010]). The drug-enforcement agents wish to consider the set $\mathcal{B} \equiv \{p_{41}, p_{52}\}$. First, they must calculate the reward associated with each explanation function (note that $k = 3$, $dist = 100$, and $\mathbf{rf} = \mathbf{crf}$).

$$\mathbf{crf}^{dist}(\mathsf{ef}_1(\mathcal{O}, 3), \{p_{41}, p_{52}\}) = 0.67$$
$$\mathbf{crf}^{dist}(\mathsf{ef}_2(\mathcal{O}, 3), \{p_{41}, p_{52}\}) = 0.5$$

(As an aside, we would like to point out the asymmetry in **crf**; compare these computations with the results of Example 4.1). Hence, $\mathsf{EXB}^{\mathbf{crf}}(\{p_{41}, p_{52}\}, \mathsf{efd}_{drug}) = 0.634$.

We now define a maximal counter-adversary strategy.

Definition 5.2 (*Maximal Counter-Adversary Strategy (MCA)*). Given a reward function **rf** and explanation function distribution efd, a *maximal counter-adversary strategy*, $\mathcal{B}$, is a subset of $\mathcal{S}$ such that $\mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$ is maximized.

Note that MCA does not include a cardinality constraint. This is because we do not require reward functions to be monotonic. In the monotonic case, we can trivially return all feasible points in $\mathcal{S}$ and be assured of a solution that maximizes the expected agent benefit. Therefore, for the monotonic case, we include an extra parameter $B \in \{1, \ldots, |\mathcal{S}|\}$ (for "budget") which will serve as a cardinality requirement for $\mathcal{B}$. This cardinality requirement for $\mathcal{B}$ is necessarily the same as for $\mathcal{A}$ as the agent and adversary may have different sets of resources. Also, we do not require that $\mathcal{B}$ be an explanation. We discuss the special case where the solution to the MCA problem is required to be an explanation in the Appendix.

5.1. The Complexity of Finding a Maximal Counter-Adversary Strategy

We now formally define the problem of finding a maximal counter-adversary strategy.

MCA Problem.
INPUT: Space $\mathcal{S}$, feasibility predicate feas, real numbers α, β, set of observations $\mathcal{O}$, natural numbers k, B, reward function **rf**, and explanation function distribution efd.
OUTPUT: Maximal counter-adversary strategy $\mathcal{B}$.

MCA is NP-hard via a reduction of the GCD problem.

THEOREM 5.1. *MCA is NP-hard.*

The proof of the preceding result shows that MCA is NP-hard even if the reward function is monotonic. Later, in Section 5.3, we also show that MCA can encode the NP-hard MAX-K-COVER problem [Feige 1998] as well (which provides an alternate proof for NP-hardness of MCA). We now present the decision problem associated with MCA and show that it is NP-complete under reasonable conditions.

MCA-DEC.
INPUT: Space $\mathcal{S}$, feasibility predicate feas, real numbers α, β, set of observations $\mathcal{O}$, natural numbers k, B, reward function **rf**, explanation function distribution efd, and number $R \in [0, 1]$.
OUTPUT: Counter-adversary strategy $\mathcal{B}$ such that $\mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd}) \geq R$.

THEOREM 5.2. ***MCA-DEC*** *is NP-complete, provided the reward function can be evaluated in PTIME.*

Not only is **MCA-DEC** NP-hard, under the same assumptions as earlier, the counting version of the problem is #P-complete and moreover, it has no fully polynomial random approximation scheme.

THEOREM 5.3. *Counting the number of strategies that provide a "yes" answer to* ***MCA-DEC*** *is #P-complete and has no FPRAS unless NP=RP.*

Theorem 5.3 tells us that MCA may not have a unique solution. Therefore, setting up a mixed strategy of all MCAs to determine the "best response" to the MCA of an agent by an adversary would be an intractable problem. This mirrors our result of the previous section (Theorem 4.3).

5.2. MCA in the General Case: Exact and Approximate Algorithms

We now describe exact and approximate algorithms for finding a maximal counter-adversary strategy in the general case. Note that throughout this section (as well as in Section 5.3), we assume that the same preprocessing for **OAS** is used (refer to Section 4.2). We will use the symbol L to refer to the set of all possible partners.

An Exact Algorithm For MCA. A naive, exact, and straightforward approach to the MCA problem would simply consider all subsets of L and pick the one which maximizes the expected agent benefit. Obviously, this approach has a complexity $O\left(\sum_{i=0}^{|\mathcal{S}|} \binom{|L|}{i}\right)$ and is not practical. This is unsurprising as we showed this to be an NP-complete problem.

Approximation in the General Case. Despite the impractical time complexity associated with an exact approach, it is possible to approximate MCA with guarantees, even in the general case. This is due to the fact that when efd is fixed, the expected agent benefit is submodular.

ALGORITHM 2: (MCA-LS)

INPUT: Reward function **rf**, set $\mathcal{O}$ of observations, explanation function distribution efd, possible partner set L, real number $\epsilon > 0$
OUTPUT: Set $\mathcal{B} \subset \mathcal{S}$

(1) Set $\mathcal{B}^* = L$, for each $p_i \in \mathcal{B}^*$ let $inc_i = \mathsf{EXB}^{\mathbf{rf}}(\{p\}, \mathsf{efd}) - \mathsf{EXB}^{\mathbf{rf}}(\emptyset, \mathsf{efd})$.
(2) Sort the p_i's in $\mathcal{B}^*$ from greatest to least by inc_i (*i.e.*, p_1 is the element with the greatest inc_i).
(3) $\mathcal{B} = \{p_1\}$, $\mathcal{B}^* = \mathcal{B}^* - \{p_1\}$, $cur\_val = inc_1 + \mathsf{EXB}^{\mathbf{rf}}(\emptyset, \mathsf{efd})$, $flag1 = \mathsf{true}$, $i = 2$
(4) While $flag1$
  (a) $new\_val = cur\_val + inc_i$
  (b) If $new\_val > (1 + \frac{\epsilon}{|L|^2}) \cdot cur\_val$ then
    i.If $\mathsf{EXB}^{\mathbf{rf}}(\mathcal{B} \cup \{p_i\}, \mathsf{efd}) > (1 + \frac{\epsilon}{|L|^2}) \cdot \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$ then:
      $\mathcal{B} = \mathcal{B} \cup \{p_i\}$, $\mathcal{B}^* = \mathcal{B}^* - \{p_i\}$, $cur\_val = \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B} \cup \{p_i\}, \mathsf{efd})$
  (c) If $new\_val \leq (1 + \frac{\epsilon}{|L|^2}) \cdot cur\_val$ **or** if p_i is the last element then
    i. $j = 1$, $flag2 = \mathsf{true}$, number each $p_j \in \mathcal{B}$
    ii. While $flag2$
      A. If $\mathsf{EXB}^{\mathbf{rf}}(\mathcal{B} - \{p_j\}, \mathsf{efd}) > (1 + \frac{\epsilon}{|L|^2}) \cdot \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$ then:
        $\mathcal{B} = \mathcal{B} - \{p_j\}$, $cur\_val = \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B} - \{p_j\}, \mathsf{efd})$
        For each $p_i \in \mathcal{B}^*$ let $inc_i = \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B} \cup \{p_i\}, \mathsf{efd}) - \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$.
        Sort the p_i's in $\mathcal{B}^*$ from greatest to least by inc_i
        $i = 0$, $flag2 = \mathsf{false}$
      B. Else,
        If p_j was the last element of $\mathcal{B}$ then set $flag1$, $flag2 = \mathsf{false}$
        Otherwise, $j{+}{+}$
  (d) $i{+}{+}$
(5) If $\mathsf{EXB}^{\mathbf{rf}}(L - \mathcal{B}, \mathsf{efd}) > \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$ then set $\mathcal{B} = L - \mathcal{B}$
(6) Return $\mathcal{B}$

THEOREM 5.4. *For a fixed $\mathcal{O}$, k, efd, the expected agent benefit, $EXB^{\mathbf{rf}}(\mathcal{B}, efd)$ has the following properties:*

(*1*) $EXB^{\mathbf{rf}}(\mathcal{B}, efd) \in [0, 1]$.
(*2*) *For $\mathcal{B} \subseteq \mathcal{B}'$ and some point $p \in \mathcal{S}$ where $p \notin \mathcal{B}'$, the following is true:*

$$EXB^{\mathbf{rf}}(\mathcal{B} \cup \{p\}, efd) - EXB^{\mathbf{rf}}(\mathcal{B}, efd) \geq EXB^{\mathbf{rf}}(\mathcal{B}' \cup \{p\}, efd) - EXB^{\mathbf{rf}}(\mathcal{B}', efd)$$

(i.e., *expected agent benefit is submodular for MCA*).

It follows immediately that MCA reduces to the maximization of a submodular function. We now present the MCA-LS algorithm that leverages this submodularity.

The following two propositions leverage Theorem 5.4 and Theorem 3.4 of Feige et al. [2007].

PROPOSITION 5.1. *MCA-LS has time complexity of $O(\frac{1}{\epsilon} \cdot |L|^3 \cdot F(efd) \cdot \lg(|L|)$ where $F(efd)$ is the time complexity to compute $EXB^{\mathbf{rf}}(\mathcal{B}, efd)$ for some set $\mathcal{B} \subseteq L$.*

PROPOSITION 5.2. *MCA-LS is an $(\frac{1}{3} - \frac{\epsilon}{|L|})$-approximation algorithm for MCA.*

Example 5.2. Let us consider our running example where drug-enforcement agents are attempting to locate illegal drug laboratories in the area depicted in Figure 1.

The agents have information that there are k or fewer drug laboratories that support the poppy fields (set of observations $\mathcal{O}$) and that they are positioned according to efd_{drug} (see Example 3.4). The agents wish to find a maximal counter-adversarial strategy using the **prf** reward function. They decide to use MCA-LS to find such a strategy with $\epsilon = 0.1$. Initially (at line 2), the algorithm selects point p_{48} (renumbering as p_1, note that in this example we shall use p_i and inc_i numbering based on Example 2.1 rather than what the algorithm uses). Hence, $inc_{40} = 0.208$ and $cur\_val = 0.708$. As the elements are sorted, the next point to be considered in the loop at line 2 is p_{40} which has an incremental increase of 0, so it is not picked. It then proceeds to point p_{41}, which gives an incremental increase of 0.084 and is added to $\mathcal{B}$ so $cur\_val = 0.792$. Point p_{45} is considered next, which gives an incremental increase of 0.208 and is picked, so now $cur\_val = 1.0$. The algorithm then considers point p_{46}, which does not afford any incremental increase. After considering points $p_{33}, p_{35}, p_{37}, p_{42}, p_{43}, p_{44}, p_{47}, p_{49}, p_{50}, p_{52}, p_{56}$, and finding they all give a negative incremental increase (and thus, are not picked), the algorithm finds that the old incremental increase of the next element, p_1, would cause the "if" statement at line 4c to be true, thus proceeding to the inner loop inside that "if" statement (line 4(c)iiA). This loop considers if the removal of any of the picked elements p_{48}, p_{41}, p_{45} causes the expected agent benefit to increase. However, in this example, if any of the elements are removed, the expected agent benefit decreases. Hence, the boolean $flag1$ is set to false and the algorithm exits the outer loop. The algorithm then returns the set $\mathcal{B} \equiv \{p_{48}, p_{41}, p_{45}\}$ which is optimal.

5.3. Finding a Maximal Counter-Adversary Strategy, the Monotonic Case

In the previous section we showed a $\frac{1}{3}$ approximate solution to MCA can be found in polynomial time even without any monotonicity restriction. In this section, we show that under the additional assumptions of monotonicity of reward functions, we can obtain a better 63% approximation ratio with a faster algorithm. Here, we also have the additional cardinality requirement of B for the set $\mathcal{B}$ (as described in Section 5). We first show that expected agent benefit is monotonic when the reward function is.

COROLLARY 5.1. *For a fixed $\mathcal{O}, k$, efd, if the reward function is monotonic, then the expected agent benefit, $EXB^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$ is also monotonic.*

Thus, when we have a monotonic reward function, the MCA problem reduces to the maximization of a monotonic, normalized[5] submodular function with respect to a uniform matroid[6]; this is a direct consequence of Theorem 5.4 and Corollary 5.1. Therefore, we can leverage the result of Nemhauser et al. [1978], to develop the MCA-GREEDY-MONO algorithm that follows. We improve performance by including "lazy evaluation" using the intuition that the incremental increase caused by some point p at iteration i of the algorithm is greater than or equal to the increase caused by that point at a later iteration. As with MCA-LS, we also sort elements by the incremental increase, which may allow the algorithm to exit the inner loop earlier. In most nontrivial instances of MCA, this additional sorting operation will not affect the complexity of the algorithm (i.e., under the assumption that the time to compute $\mathsf{EXB}^{\mathbf{rf}}$ is greater than $\lg(|L|)$, we make this same assumption in MCA-LS as well).

[5]As we include zero-starting in our definition of monotonic.
[6]In our case, the uniform matroid consists of all subsets of L of size B or less.

ALGORITHM 3: (MCA-GREEDY-MONO)

INPUT: Monotonic reward function **rf**, set $\mathcal{O}$ of observations, real number $B > 0$, explanation function distribution efd, possible partner set L, real number $\epsilon > 0$
OUTPUT: Set $\mathcal{B} \subset \mathcal{S}$

(1) Initialize $\mathcal{B} = \emptyset$ and $\mathcal{B}^* = L$
(2) For each $p_i \in \mathcal{B}^*$, set $inc_i = 0$
(3) Set $last\_val = \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$
(4) While $|\mathcal{B}| \leq B$
  (a) $p_{best} = \mathsf{null}$, $cur\_inc = 0$
  (b) For each $p_i \in \mathcal{B}^*$, do the following
    i. If $inc_i < cur\_inc$, break loop and goto line 3.
    ii. Let $inc_i = \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B} \cup \{p\}, \mathsf{efd}) - last\_val$
    iii. If $inc_i \geq cur\_inc$ then $cur\_inc = inc_i$ and $p_{best} = p$
  (c) $\mathcal{B} = \mathcal{B} \cup \{p_{best}\}$, $\mathcal{B}^* = \mathcal{B}^* - \{p_{best}\}$
  (d) Sort $\mathcal{B}^*$ in descending order by inc_i.
  (e) Set $last\_val = \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$
(5) Return $\mathcal{B}$

PROPOSITION 5.3. *The complexity of MCA-GREEDY-MONO is* $O(B \cdot |L| \cdot F(efd))$ *where* $F(efd)$ *is the time complexity to compute* $EXB^{\mathbf{rf}}(\mathcal{B}, efd)$ *for some set* $\mathcal{B} \subseteq L$ *of size B. In the first iteration of the algorithm, we have the next corollary.*

COROLLARY 5.2. *MCA-GREEDY-MONO is an* $(\frac{e}{e-1})$*-approximation algorithm for MCA (when the reward function is monotonic).*

In addition to the fact that MCA-GREEDY-MONO is an $(\frac{e}{e-1})$-approximation algorithm for MCA, it also provides the best possible approximation ratio unless $P = NP$. This is done by a reduction of MAX-K-COVER [Feige 1998].

THEOREM 5.5. *MCA-GREEDY-MONO provides the best approximation ratio for MCA (when the reward function is monotonic) unless* $P = NP$.

The following example illustrates how MCA-GREEDY-MONO works.

Example 5.3. Consider the situation from Example 5.2, where the drug-enforcement agents are attempting to locate illegal drug labs. Suppose they want to locate the labs, but use the **crf** reward function, which is monotonic and zero-starting. They use the cardinality requirement $B = 3$ in MCA-GREEDY-MONO. After the first iteration of the loop at line 3, the algorithm selects point p_{48} as it affords an incremental increase of 0.417. On the second iteration, it selects point p_{46}, as it also affords an incremental increase of 0.417, so $last\_val = 0.834$. Once p_{46} is considered, the next point considered is p_{33}, which had a previous incremental increase (calculated in the first iteration) of 0.25, so the algorithm can correctly exit the loop to select the final element. On the last iteration of the outer loop, the algorithm selects point p_{35}, which gives an incremental increase of 0.166. Now the algorithm has a set of cardinality 3, so it exits the outer loop and returns the set $\mathcal{B} = \{p_{48}, p_{46}, p_{35}\}$, which provides an expected agent benefit of 1, which is optimal. Note that this would not be an optimal solution for the scenario in Example 5.2 which uses **prf** as p_{35} would incur a penalty (which it does not when using **crf** as in this example).

6. IMPLEMENTATION AND EXPERIMENTS

In this section, we describe prototype implementations and experiments for solving the OAS and MCA problems. For OAS, we create an MILP for the **crf** case and reduce the number of variables with the techniques we presented in Section 4. For MCA, we implement both the MCA-LS and MCA-GREEDY-MONO.

We carried out all experiments for MCA on an Intel Core2 Q6600 processor running at 2.4 GHz with 8GB of memory available, using code written in Java 1.6; all runs were performed in Windows 7 Ultimate 64-bit using a 64-bit JVM, and made use of a single core. We also used functionality from the previously implemented SCARE software [Shakarian et al. 2009] to calculate, for example, the set of all possible partners L and to perform preprocessing (see the discussion in Section 4.2).

Our experiments are based on 21 months of real-world Improvised Explosive Device (IED) attacks in Baghdad[7] [Shakarian et al. 2009]. The IED attacks in this 25×27 km region constitute our observations. The data also includes locations of caches associated with those attacks discovered by U.S. forces. These constitute partner locations. We used data from the International Medical Corps to define feasibility predicates based on ethnic makeup, location of U.S. bases, and geographic features. We overlaid a grid of 100m $\times$ 100m cells, about the size of a standard U.S. city block. We split the data into two parts; the first 7 months of data were used as a "training" set to learn the $[\alpha, \beta]$ parameters and the next 14 months of data were used for the observations. We created an explanation function distribution based on multiple runs of the GREEDY-KSEP-OPT2 algorithm described in Shakarian et al. [2010].

6.1. OAS Implementation

We now present experimental results for the version of OAS, with the **crf** reward function, based on the constraints in Definition 4.3 and variable reduction techniques of Section 4.4. First, we discuss promising real-world results for the calculation of the reduced partner set L^*, described in Definition 4.5. Then, we show that an optimal adversarial strategy can be computed quite tractably using the methods discussed in Section 4.4. Finally, we compare our results to a set of real-world data, showing a significant decrease in the adversary's expected detriment across various parameter settings. Our implementation was written on top of the QSopt[8] MILP solver and used 900 lines of Java code.

Reduced Partner Set. As discussed in Section 4.2, producing an optimal adversarial strategy for any reward function relies heavily on efficiently solving a (provably worst-case intractable) integer linear program. The number of integer variables in these programs is based solely on the size of the partner set L; as such, the ability to experimentally solve OAS relies heavily on the size of this set.

Our real-world data created a partner set L with cardinality 22,692. We then applied the method from Definition 4.5 to reduce this original set L to a smaller subset of possible partners L^*, while retaining the optimality of the final solution. This simple procedure, while dependent on the explanation function distribution efd as well as the cutoff distance for **crf**, always returned a reduced partner set L^* with cardinality between 64 and 81. This represents around a 99.6% decrease in the number of variables required in the subsequent integer linear programs!

Figure 4 provides more detailed accuracy and timing results for this reduction. Most importantly, regardless of parameters chosen, our real-world data is reduced by orders

[7]Attack and cache location data provided by the Institute for the Study of War.
[8]http://www2.isye.gatech.edu/~wcook/qsopt/index.html

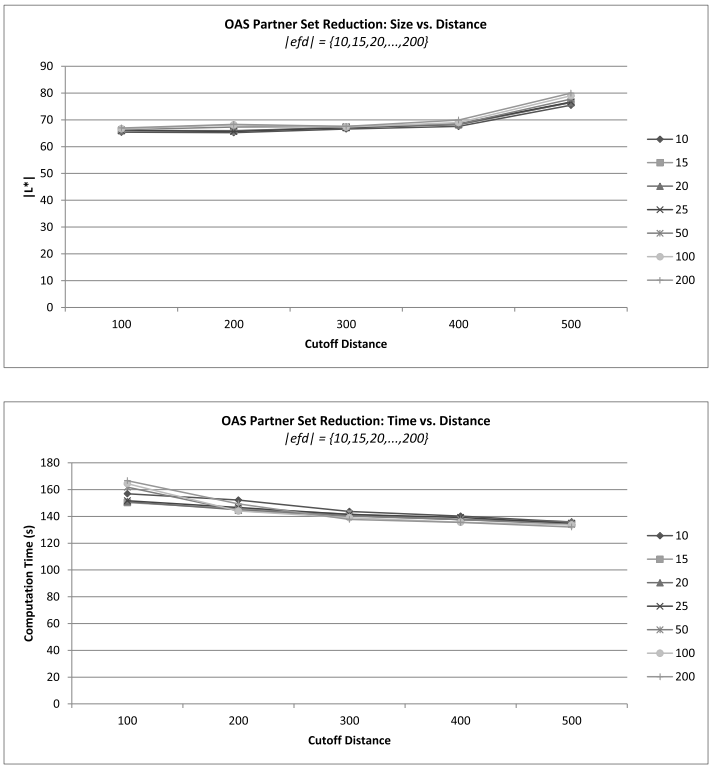

Fig. 4. The size of the reduced partner set L^* (left) and the time required to compute this reduction (right). Regardless of parameters chosen, we see a 99.6% decrease in possible partners (as well as integer variables in our linear program) in under 3 minutes.

of magnitude across the board. Of note, we see a slight increase in the size of the reduced set L^* as the size of the explanation function distribution efd increases. This can be traced back to the strict inequality in Definition 4.7. As we increase the number of nontrivial explanation functions in efd, the number of nonzero constants $const_i$ increases. This results in a higher number of candidates for the intermediary set L^{**}. We see a similar result as we increase the penalizing cutoff distance. Again, this is a factor of the strict inequality in Definition 4.7 in conjunction with a higher fraction of nonzero $const_i$ constants.

Interestingly, Figure 4 shows a slight decrease in the runtime of the reduction as we increase the penalizing cutoff distance. Initially, this seems counterintuitive; with more nontrivial constants $const_i$, the construction of the intermediary set L^{**} requires more work. However, this extra work pays off during the computation of the final reduced set L^*. In our experiments, the reduction from L to L^{**} took less time than the final reduction from L^{**} to L^*. This is due to frequent short circuiting in the computation of the right-hand side of the conjunction during L^{**} creation. As we increase the penalizing cutoff distance, the size of L^{**} actually decreases, resulting in a decrease in the longer computation of L^*. As seen before, this decrease in L^{**} did not correspond to a decrease in the size of L^*.

Optimal Adversarial Strategy. Using the set L^*, we now present results to find an optimal adversarial strategy using δ-core optimal explanations. This is done by minimizing the MILP of Section 4.4, then feeding this solution into BUILD-STRAT. Since we do

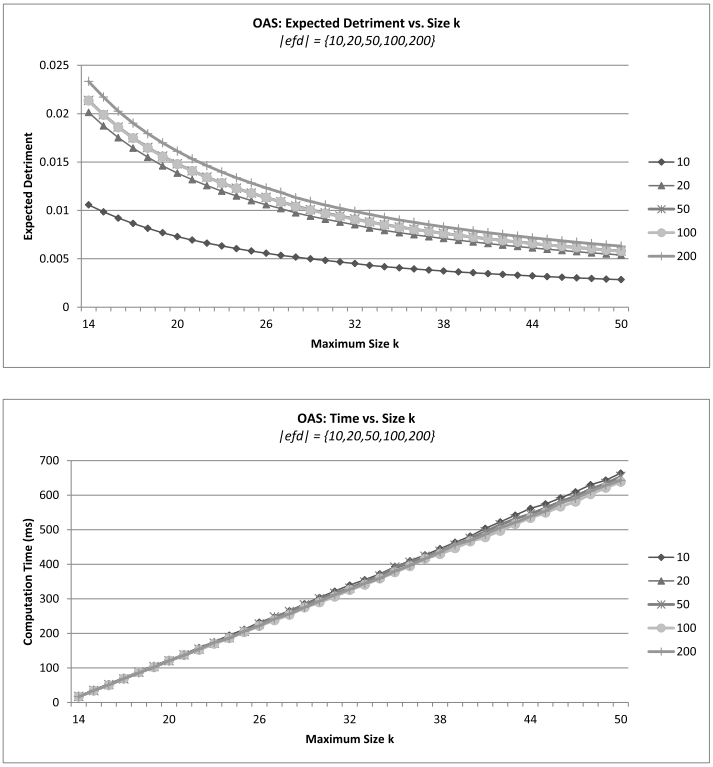

Fig. 5. Expected detriment of the optimal adversarial strategy (left, lower is better) and the runtime of the integer linear program required to produce this strategy in milliseconds (right). Note the smooth decrease toward zero detriment as k increases, corresponding with a near-linear increase in total runtime.

not know the value of δ in advance, we must perform this combined operation multiple times, choosing the best (lowest expected detriment) adversarial strategy as optimal.

A note on the lower bound for δ: As shown by Shakarian et al. [2009], finding a minimum-cardinality explanation is NP-hard. Because of this, it is computationally difficult to find a tight lower bound for δ. However, this lower bound can be estimated empirically. For instance, for our set of real-world data from Baghdad, an explanation of cardinality below 14 has never been returned, even across tens of thousands of runs of GREEDY-KSEP-OPT2. Building on this strong empirical evidence, the minimum δ used in our experiments is 14.

Figure 5 shows both timing and expected detriment results as the size of the explanation function |efd| and maximum strategy cardinality k are varied. Note that a lower expected detriment is better for the adversary, with zero representing no probability of partner discovery by the reasoning agent. As the adversary is allowed larger and larger strategies, its expected detriment smoothly decreases toward zero. Intuitively, as the number of nontrivially-weighted explanation functions in efd increases, the expected detriment increases as well. This is a side-effect of a larger |efd| allowing the reasoning agent to cover a larger swath of partner locations.

Recall that, as the maximum k increases, we must solve linear programs for each $\delta \in \{k_{low}, k\}$. This is mirrored in the timing results in Figure 5, which assumes $k_{low} = 14$. As k increases, we see a near-linear increase in the total runtime of the set of integer programs. Due to the reduced set L^*, we are able to solve dozens of

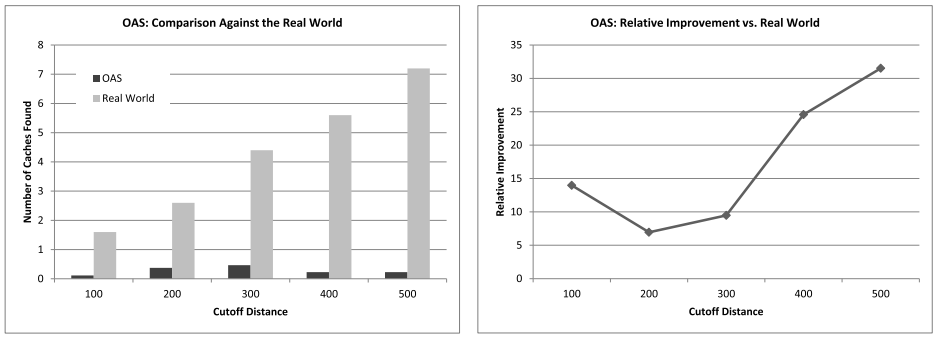

Fig. 6. Expected number of caches found when the adversary uses our strategy instead of the current state-of-the-art (left, lower is better). Relative improvement of the OAS strategy versus the current state-of-the-art (right, higher is better). We assume the reasoning agent is using the Spatial Cultural Abductive Reasoning Engine (SCARE) to provide information on cache locations.

integer programs in less than 800ms; were we to use the unreduced partner set L, this would be intractable. Note that the runtime graph includes that of BUILD-STRAT which always ran in under sixteen milliseconds.

OAS Performance with respect to Real-World Adversarial Strategy. Figure 6 compares the expected number of caches found under the current, state-of-the-art—IED cache locations based on 21 months of real-world data from Baghdad, Iraq—against the OAS strategy proposed in this article. We hold the cardinality of the adversary's solution (i.e., the number of possible caches) to 14 to match the real-world data. We assume the reasoning agent uses the Spatial Cultural Abductive Reasoning Engine (SCARE) introduced in Shakarian et al. [2009] to provide partner locations to these attacks. SCARE is the state-of-the-art method for finding IED caches.

When tested against real-world adversaries based on real-world Baghdad data, OAS significantly outperforms what adversaries have done so far in the real world (fortunately this is balanced by later experiment results showing that MCA-LS and MCA-GREEDY-MONO significantly outperform SCARE). The expected number of caches found by SCARE against an opponent using OAS is significantly lower than against present-day insurgents in Iraq. For instance, while SCARE (using a cutoff distance of 100 meters) detects 1.6 of the 14 possible caches against a real-world adversary, it is expected to detect only 0.11 of the caches against an adversary using OAS. This roughly order of magnitude improvement is seen across all five cutoff distances, from a minimum of approximately 7x at a cutoff distance of 200m to a maximum of over 31x at a distance of 500m. Thus, OAS significantly improves the adversary's performance.

6.2. MCA Implementation

First, we briefly discuss an implementation of the naive MCA algorithm discussed in Section 5.2. Next, we provide promising results for the MCA-LS algorithm using the **prf** reward function. Finally, we give results for the MCA-GREEDY-MONO using the monotonic **crf** reward function, and qualitatively compare and contrast the results from both algorithms.

MCA-Naive. The naive, exact solution to MCA (considering all subsets of L with cardinality $k_{\mathcal{B}}$ or more and picking the one which maximizes the expected agent benefit) is inherently intractable. This approach has a complexity $O(\binom{|L|}{k_{\mathcal{B}}})$, and is made worse by the large magnitude of the set L. In our experimental setup, we typically

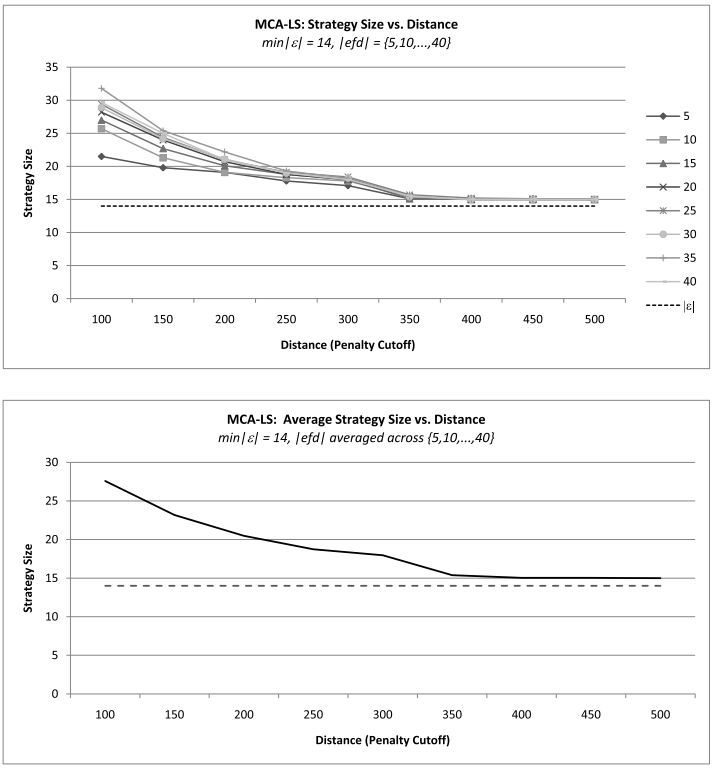

Fig. 7. The average size of the strategy recommended by MCA-LS decreases as the distance cutoff increases. For these experiments, the minimum cardinality for a given explanation $\mathcal{E}$ considered is efd was 14, which gives us a natural lower bound on the expected size of a strategy. Note the convergence to this bound at cutoff distances at and above 300 meters.

saw $|L| > 20{,}000$; as such, for even the trivially small $k_{\mathcal{B}} = 3$, we must enumerate and rank over a trillion subsets. For any realistic value of $k_{\mathcal{B}}$, this approach is simply unusable. Luckily, we will see that both MCA-LS and MCA-GREEDY-MONO provide highly tractable and accurate alternatives.

MCA-LS. In sharp contrast to the naive algorithm described previously, the MCA-LS algorithm provides (lower-)bounded approximate results in a tractable manner. Interestingly, even though MCA-LS is an approximation algorithm, in our experiments on real-world data from Baghdad using the **prf** reward function, the algorithm returned strategies with an expected benefit of 1.0 on every run. Put simply, on our practical test data, MCA-LS always completely maximized the expected benefit. This significantly outperforms the lower-bound approximation ratio of 1/3. We would also like to point out that this is the first implementation (to the best of our knowledge) of the nonmonotonic submodular maximization approximation algorithm of Feige et al. [2007].

Since the expected benefit was maximal for every strategy $\mathcal{B}$ returned, we move to analyzing the particular structure of these strategies. Figure 7 shows a relationship between the size $|\mathcal{B}|$, the cutoff distance *dist*, and the cardinality of the expectation function distribution |efd|. Recall that **prf** penalizes any strategy that does not completely cover its input set of observations; as such, intuitively, we see that MCA-LS

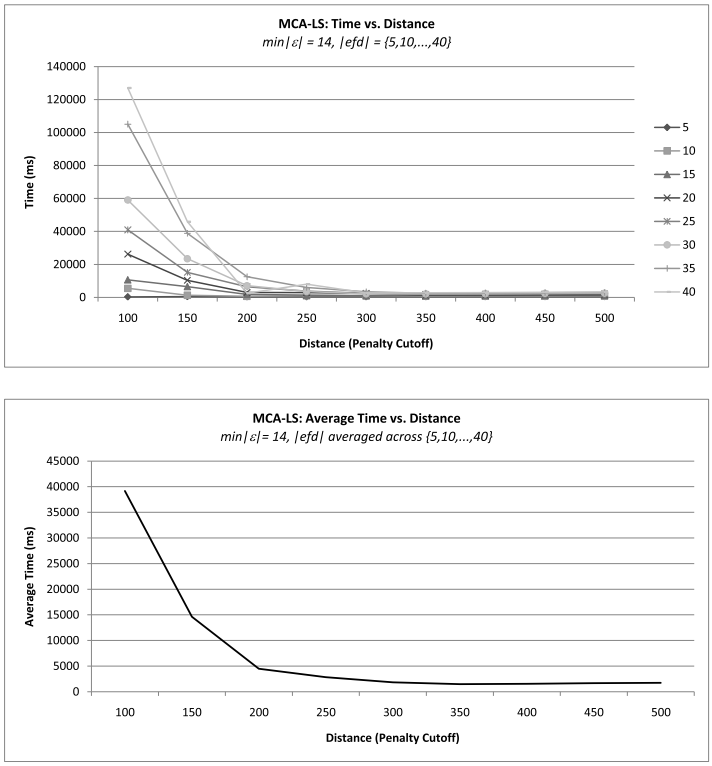

Fig. 8. The runtime of MCA-LS decreases as the penalizing cutoff distance is relaxed. Note the relation to Figure 7; intuitively, larger recommended strategies tend to take longer to compute.

returns larger strategies as the penalizing cutoff distance decreases. If the algorithm can cover all possible partners across all expectation functions, it will not receive any penalty. Still, even when *dist* is 100m, the algorithm returns $\mathcal{B}$ only roughly twice the size as minimum-sized explanation found by GREEDY-KSEP-OPT2 (which, based on the analysis of Shakarian et al. [2010], is very close to the minimum possible explanation). As the cutoff *dist* increases, the algorithm returns strategies with sizes converging, generally, to a baseline: the smallest-sized explanation found by the algorithm of Shakarian et al. [2010], $|\mathcal{E}|$. This is an intuitive soft lower bound; given enough leeway from a large distance *dist*, a single point will cover all expected partners. This is not a strict lower bound in that, given two extremely close observations with similar expected partners, a single point may sufficiently cover both.

In Figure 8, we see results comparing overall computation time to both the distance *dist* and the cardinality of efd. For more strict (i.e., smaller) values of *dist*, the algorithm (which, under **prf**, is penalized for all uncovered observations across efd) must spend more time forming a strategy $\mathcal{B}$ that minimizes penalization. Similarly, as the distance constraint is loosened, the algorithm completes more quickly. Finally, an increase in |efd| results in higher computational cost; as explained in Proposition 5.1, this is due to an increase in $F(\text{efd})$, the time complexity of computing $\text{EXB}^{\mathbf{rf}}(\mathcal{B}, \text{efd})$. Comparing these results to Figure 7, we see that the runtime of MCA-LS is correlated to the size of the returned strategy $\mathcal{B}$.

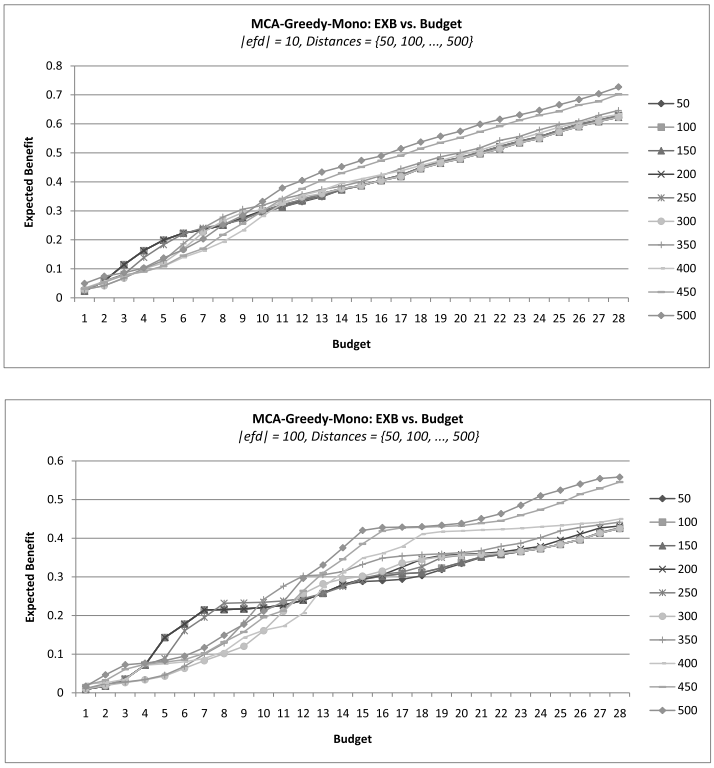

Fig. 9. Expected benefit of the strategy returned by MCA-GREEDY-MONO as the budget increases, with |efd| = 10 (left) and |efd| = 100 (right). Note the decrease in expected benefit due to the increase in |efd|. Similarly, note the increase in expected benefit given a larger cutoff distance.

MCA-GREEDY-MONO. As discussed in Section 5.3, MCA-GREEDY-MONO provides tighter approximation bounds than MCA-LS at the cost of a more restrictive (monotonic) reward function. For these experiments, we used the monotonic **rf** = **crf**. Recall that a trivial solution to MCA given a monotonic reward function is $\mathcal{B} = L$; as such, MCA-GREEDY-MONO uses a budget B to limit the maximum size $|\mathcal{B}| \ll |L|$. We varied this parameter $B \in \{1, \ldots, 28\}$.

Figure 9 shows the expected benefit $\mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$ increases as the maximum allowed $|\mathcal{B}|$ increases. In general, the expected benefit of $\mathcal{B}$ increases as the distance constraint *dist* is relaxed. However, note the points with $B \in \{3, \ldots, 9\}$; we see that $dist \leq 100$ performs better than $dist > 100$. We believe this is an artifact of our real-world data. Finally, as |efd| increases, the expected benefit of $\mathcal{B}$ converges more slowly to 1.0. This is intuitive, as a wider spread of possible partner positions will, in general, require a larger $|\mathcal{B}|$ to provide coverage.

Figure 10 shows that the runtime of MCA-GREEDY-MONO increases as predicted by Proposition 5.1. In detail, as we linearly increase budget B, we also linearly increase the runtime of our $F(\mathsf{efd}) = \mathsf{EXB}^{\mathbf{rf}}(\mathcal{B}, \mathsf{efd})$. In turn, the overall runtime $O(B \cdot |L| \cdot F(\mathsf{efd}))$ increases quadratically in B, for our specific reward function. Finally, note the increase in runtime as we increase |efd| = 10 to |efd| = 100. Theoretically, this increases $F(\mathsf{efd})$ linearly; in fact, we see almost exactly a ten-fold increase in runtime given a tenfold increase in |efd|.

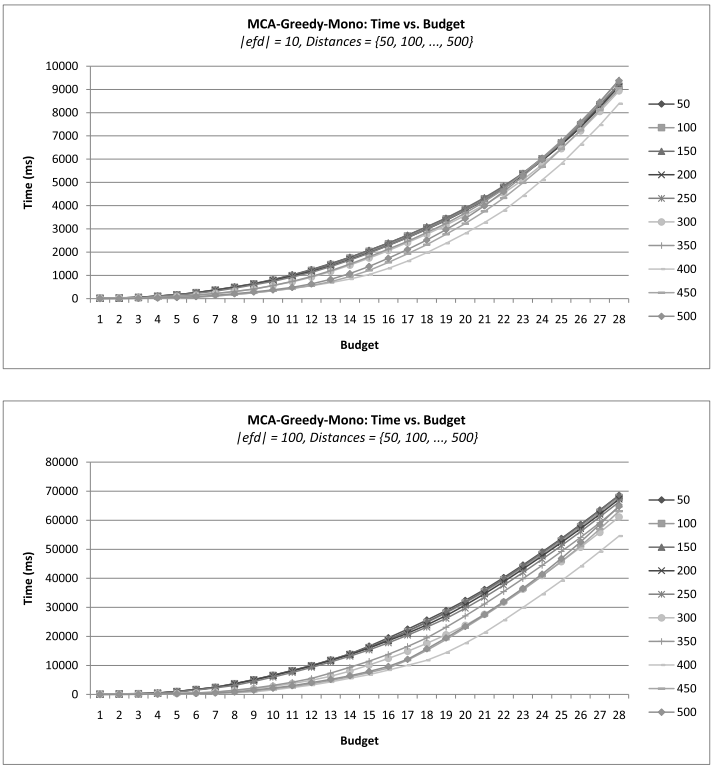

Fig. 10. Runtime of MCA-GREEDY-MONO as the budget increases, with |efd| = 10 (left) and |efd| = 100 (right). Note the increase in runtime due to the extra determinism of a larger efd.

MCA Algorithms and SCARE. We now compare the efficacy of the two MCA algorithms proposed in this article to SCARE [Shakarian et al. 2009] which represents the current state-of-the-art as far as IED cache detection is concerned. Again, our experiments are based on real-world data from Baghdad, Iraq. For these experiments, we average results across 100 runs of SCARE; as such, we hold |efd| = 100 static for the MCA-based algorithms. Figure 11 plots the average number of predicted points within 500 meters of an actual cache for both MCA-LS and MCA-GREEDY-MONO. SCARE, plotted as a horizontal line, predicts an average of 7.87 points within 500 meters of caches. MCA-LS finds over twice as many points at low penalizing cutoff distances, and steadily converges to SCARE's baseline as the penalizing distance increases (as expected). As shown earlier in Figure 7, MCA-LS tends to find larger strategies given a smaller penalizing cutoff distance; in turn, these larger strategies yield more close points to actual caches. MCA-GREEDY-MONO shows similar behavior; as we increase the allowable budget (i.e., maximum strategy size), more points are within 500 meters of a real-world cache location. Thus, MCA-LS and MCA-GREEDY-MONO both outperform SCARE, enabling more caches to be discovered.

We note that while the number of points in the strategy close to a real-world cache location is higher in the MCA-based algorithms than SCARE, the fraction of close points stays consistently close. SCARE returns a solution of size 14, with approximately half ($7.87/14 \approx 56\%$) of these points within 500 meters of cache. Compare this to, for instance, MCA-LS with a penalizing cutoff distance of 300 meters; for these settings, the algorithm returns an average strategy size of 18, with 11 points

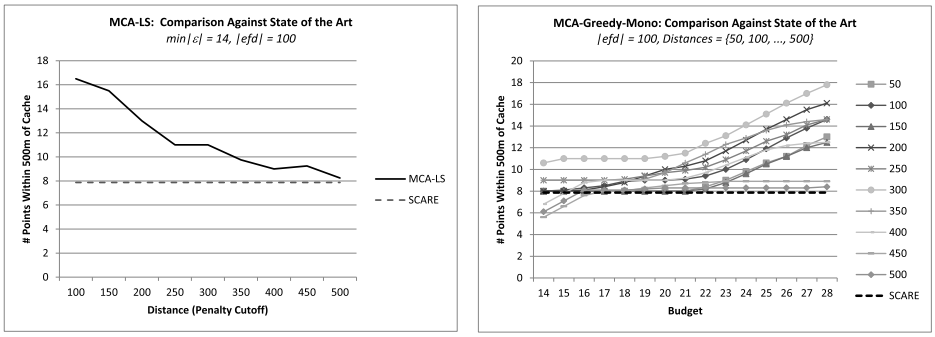

Fig. 11. Expected number of points within 500 meters of an actual cache returned by MCA-LS (left) and MCA-GREEDY-MONO (right) compared against an agent using SCARE (higher is better). Note that the SCARE software always returns an explanation of size 14, while both MCA algorithms benefit from the ability to adjust this explanation size.

(approximately 60%) within 500 meters of a cache location. This behavior is a product of the strategy size flexibility built into the MCA-based algorithms, and is beneficial to the reasoning agent. For example, assume the minimal solution to a problem is of size 2 and the reasoning agent has a budget of size 4. Now assume SCARE finds $1/2 = 50\%$ of the points near caches, while MCA-GREEDY-MONO finds $2/4 = 50\%$ of its points near caches. Both algorithms returned the same fraction of points near caches; however, the reasoning agent will spend its budget of 4 resources more effectively under MCA-GREEDY-MONO, instead of wasting two of its resources under the strategy provided by SCARE.

7. RELATED WORK

Geospatial abduction was introduced in Shakarian et al. [2010] and used to infer a set of partner locations from a set of observations, given a feasibility predicate and an interval $[\alpha, \beta] \subseteq [0, 1]$. The authors developed exact and approximate algorithms for GAPs. In particular, no adversary was assumed to exist there. In this article, we study the case of geospatial abduction where there is an explicit adversary who is interested in ensuring that the agent does not detect the partner locations. This is the case with real-world serial killers and insurgents who launch IED attacks. In this article, we develop a game-theoretic framework for reasoning about the best strategy that an adversary might adopt (based on minimizing the adversary's detriment) and the best strategy that the agent could adopt to counter the adversary's strategy. All this is uncharted territory and represents a novel contribution of this article. In fact, everything from Section 3 onwards in this article is new.

Although abduction [Peirce 1955] has been studied in a variety of different contexts—medicine [Peng 1986; Peng and Reggia 1990], fault diagnosis [Console et al. 1991], belief revision [Pagnucco 1996], database updates [Console et al. 1995; Kakas and Mancarella 1990], and AI planning [do Lago Pereira and de Barros 2004]—we are not aware of any work in abduction where an adversary selects a ground-truth explanation with respect to a probability distribution over explanation functions that an agent would consider. Additionally, we are not aware of any related work dealing with the problem of an agent finding elements of an adversarially selected explanation (with respect to a probability distribution). However, we do believe that many of the techniques introduced here for adversarial geospatial abduction may be generalized to other forms of abduction as well.

In the field of operations research, the *facility location* problem [Stollsteimer 1963] is a well-studied problem dealing with optimal placement of facilities in a plane, network, or multidimensional space. The facilities must be positioned to optimize some sort of distance to the "demand points", most likely resembling consumers of the items being produced at the facility. In Shakarian et al. [2010], the authors outline numerous differences between facility location and geospatial abduction (difference in optimality criteria, use of feasibility predicate, nonconvexitivity of covers, etc.), even when no adversaries are present. However, facility location with adversaries has not really been studied, and that is the focus of this article.

Similar motivation exists in the field of (multi-)agent security, where the central idea is to protect a set of targets from adversaries. These games are typically modeled on top of graphs, with agents and adversaries competing to protect or penetrate a set of targets. Paruchuri et al. [2006] represent the adversary's behavior through a probability distribution over states, indicating the probability of that state being targeted; no real graph structure is considered, much less a geospatial model. Agmon et al. [2008, 2009] consider an environment with more hidden information, and attempt to detect adversarial penetrations across the routes (represented as paths on a graph) of patrolling agents. Pita et al. [2009] solve Stackelberg (leader-follower) games under the assumption of bounded reasoning rationality, again on a graph network. Dickerson et al. [2010] explores protecting dynamic targets from rational adversaries on real-world road networks.

8. CONCLUSION

Geospatial abduction was introduced in Shakarian et al. [2010] and used to infer a set of partner locations from a set of observations, given a feasibility predicate and reals $\alpha \geq 0, \beta > 0$. Shakarian et al. [2010] developed exact and approximate algorithms for GAPs. In particular, no adversary was assumed to exist there. In this article, we study the case of geospatial abduction where there is an explicit adversary who is interested in ensuring that the agent does not detect the partner locations. This is the case with real-world serial killers and insurgents who launch IED attacks. We develop a game-theoretic framework for reasoning about the best strategy that an adversary might adopt (based on minimizing the adversary's detriment) and the best strategy that the agent could adopt to counter the adversary's strategy.

We consider the adversarial geospatial abduction problem to be a two-player game: an agent ("good" guy) and an adversary ("bad" guy). The adversary is attempting to cause certain observable events to occur (e.g., murders or IED attacks) but make it hard to detect the associated set of partner locations (e.g., location of the serial killer's home/office, or the locations of weapons caches supporting the IED attacks). We use an axiomatically-defined "reward function" to determine how similar two explanations are to each other. We study the problems of finding the best response for an agent and adversary to a mixed strategy (based on a probability distribution over explanations) of the opponent. We formalize these problems as the Optimal Adversarial Strategy (OAS) and Maximal Counter-Adversary strategy (MCA) problem. We show both OAS and MCA to be NP-hard and provide exact and approximate methods for solving them.

When reasoning about the best possible strategy for the adversary, we present a mixed-integer-programming-based algorithm and show that the MILP in question can be greatly reduced through the elimination of many variables using the concept of a δ-core explanation. Our experiments are carried out on real-world data about IED attacks over a period of 21 months in Baghdad.

When reasoning about the best possible strategy for the adversary, we present two algorithms. The MCA-LS algorithm is very general and leverages submodularity of reward functions. The MCA-GREEDY-MONO algorithm assumes the reward function is

monotonic. Both MCA-LS and MCA-GREEDY-MONO are highly accurate and have very reasonable time frames. Though MCA-GREEDY-MONO is slightly faster than MCA-LS, we found that on every single run, MCA-LS found the exact optimal benefit even though its theoretical lower-bound approximation ratio is only 1/3, a truly remarkable performance. As MCA-LS does not require any additional assumptions and as its running time is only slightly slower than that of MCA-GREEDY-MONO, we believe this algorithm has a slight advantage.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

ACKNOWLEDGMENTS

We thank the reviewers for their many excellent comments that have significantly improved the article.

REFERENCES

AGMON, N., KRAUS, S., AND KAMINKA, G. 2008. Multi-robot perimeter patrol in adversarial settings. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'08)*. 2339–2345.

AGMON, N., KRAUS, S., KAMINKA, G., AND SADOV, V. 2009. Adversarial uncertainty in multi-robot patrol. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*. 1811–1817.

CHARNES, A. AND COOPER, W. 1962. Programming with linear fractional functionals. *Naval Res. Logistics Quart. 9*, 3, 163–297.

CONSOLE, L., PORTINALE, L., AND DUPRÉ, D. T. 1991. Focusing abductive diagnosis. *AI Comm. 4*, 2–3, 88–97.

CONSOLE, L., SAPINO, M. L., AND DUPRÉ, D. T. 1995. The role of abduction in database view updating. *J. Intell. Info. Syst. 4*, 3, 261–280.

DICKERSON, J., SIMARI, G., SUBRAHMANIAN, V., AND KRAUS, S. 2010. A graph-theoretic approach to protect static and moving targets from adversaries. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'10)*. 299–306.

DO LAGO PEREIRA, S. AND DE BARROS, L. N. 2004. Planning with abduction: A logical framework to explore extensions to classical planning. *Advances in Artificial Intelligence*: In *Proceedings of the Brazilian Symposium on Artificial Intelligence (SBIA)*. 106–118.

DYER, M., GOLDBERG, L. A., GREENHILL, C., AND JERRUM, M. 2000. On the relative complexity of approximate counting problems. Tech. rep., Coventry, UK.

FEIGE, U. 1998. A threshold of ln n for approximating set cover. *J. ACM 45*, 4, 634–652.

FEIGE, U., MIRROKNI, V. S., AND VONDRAK, J. 2007. Maximizing non-monotone submodular functions. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*. IEEE Computer Society. 461–471.

HUNT III, H. B., MARATHE, M. V., RADHAKRISHNAN, V., AND STEARNS, R. E. 1998. The complexity of planar counting problems. *SIAM J. Comput. 27*, 4, 1142–1167.

JOHNSON, D. 1982. The NP-completeness column: An ongoing guide. *J. Algorithms 3*, 2, 182–195.

KAKAS, A. C. AND MANCARELLA, P. 1990. Database updates through abduction. In *Proceedings of the International Conference on Very Large Databases (VLDB)*.

KARMARKAR, N. 1984. A new polynomial-time algorithm for linear programming. *Combinatorica 4*, 4, 373–395.

LEYTON-BROWN, K. AND SHOHAM, Y. 2008. *Essentials of Game Theory: A Concise, Multidisciplinary Introduction*. Morgan and Claypool Publishers.

MASUYAMA, S. T. AND IBARAKI, T. H. 1981. The computational complexity of the m-center problems on the plane. *Trans. IECE Japan E84*, 57–64.

NEMHAUSER, G., WOLSEY, L., AND FISHER, M. 1978. An analysis of the approximations for maximizing submodular set functions. *Math. Program. 14*, 265–294.

PAGNUCCO, M. 1996. The role of abductive reasoning within the process of belief revision. Ph.D. thesis, Basser Department of Computer Science, University of Sydney, Australia.

PARUCHURI, P., TAMBE, M., ORDÓÑEZ, F., AND KRAUS, S. 2006. Security in multiagent systems by policy randomization. In *Proceedings of the 5th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'06)*.

PEIRCE, C. S. 1955. *Philosophical Writings of Peirce*, Selected and Edited with an Introduction by Justus Buchler. Dover Publications, New York.

PENG, Y. AND REGGIA, J. A. 1990. *Abductive Inference Models for Diagnostic Problem-Solving*. Springer, New York.

PITA, J., JAIN, M., ORDÓÑEZ, F., TAMBE, M., KRAUS, S., AND MAGORI-COHEN, R. 2009. Effective solutions for real-world stackelberg games: When agents must deal with human uncertainties. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'09)*. 369–376.

ROSSMO, D. K. AND ROMBOUTS, S. 2008. Geographic profiling. In *Enviromental Criminology and Crime Analysis*, R. Wortley and L. Mazerolle Eds., 136–149.

SHAKARIAN, P., SUBRAHMANIAN, V., AND SAPINO, M. L. 2009. SCARE: A case study with Baghdad. In *Proceedings of the 3rd International Conference on Computational Cultural Dynamics*. AAAI.

SHAKARIAN, P., SUBRAHMANIAN, V., AND SAPINO, M. L. 2010. GAPs: Geospatial Abduction Problems. *ACM Trans. Intell. Syst. Technol.* http://www.di.unito.it/ mlsapino/Conferma-I-fascia/TIST.pdf.

STOLLSTEIMER, J. F. 1963. A working model for plant numbers and locations. *J. Farm Econ. 45*, 3, 631–645.

U.S. ARMY. 1994. *Intelligence Preparation of the Battlefiled (US Army Field Manual)* (FM 34-130 Ed).

PENG, Y. J. R. 1986. Plausibility of diagnostic hypotheses. In *Proceedings of the 5th National Conference on Artificial Intelligence (AAAI'86)*. 140–145.

Received January 2011; revised July 2011; accepted August 2011