Chapter 4
Geospatial Abduction with Adaptive Adversaries

Paulo Shakarian, V.S. Subrahmanian, John P. Dickerson

Abstract In this chapter, we focus on the problem of geospatial abduction in the presence of an adversary who understands how we are reasoning about his behavior. For instance, consider an insurgent group carrying out Improvised Explosive Device (IED) attacks on US soldiers. Such an adversary may wish to carry out its attacks and select its cache locations (to support those attacks) in a way that it believes will most likely evade detection. How can an agent (*e.g.*, US forces) anticipate this kind of reasoning by the adversary and find optimal locations to search for weapons caches? In this chapter, we develop a framework to express both the adversary's problem and the agent's problem via the paradigm of Stackelberg games. We formally specify the Optimal Adversary Strategy (OAS) problem, allowing the adversary to find a set of cache locations to minimize (what it believes) to be the probability of being discovered. We describe results on the computational complexity of OAS and algorithms to efficiently compute OAS. As the situation is modeled as a Stackelberg game, the agent (*e.g.*, US forces) takes the final action (*e.g.*, search for the IED caches). The agent can decide where to search *after* considering the space of options that the adversary has and after considering how the adversary might act in order to evade detection. We formalize this as the Maximal Counter-Adversary (MCA) strategy. We describe results on the computational complexity of MCA, as well as algorithms to efficiently compute MCA. These include algorithms that provide guaranteed polynomial approximations to MCA. We describe experimental results about the running time, accuracy, and quality of solutions found by the algorithms to compute OAS and MCA.

4.1 Introduction

We begin by reconsidering several example scenarios given in Chapter 1 where geospatial abduction is required. While all of these scenarios involve an agent (who we implicitly treat as the "good guy") and an adversary (the "bad guy"), we see that some of these scenarios, but not all, represent cases where an adversary can

P. Shakarian and V.S. Subrahmanian, *Geospatial Abduction: Principles and Practice*,
DOI 10.1007/978-1-4614-1794-1_4,

intelligently anticipate what an agent might do, and will take steps to avoid any negative effects of the agent.

- **IED Cache Detection Problem.** Here, US forces (the agent) are trying to find the locations of weapons caches used by insurgents to carry out IED terrorist attacks against civilians and/or US forces. The adversary is constantly adapting its tactics by observing what US soldiers do after any given IED attack. It is therefore clear that after some period of observation of how US forces are searching for IED weapons caches, they will understand at least some elements of the general approach described in Chapters 2 and 3. They may not understand the underlying mathematics, and they will certainly not have the feasibility predicates and lower bound and upper bound cutoff distances used, but they will be able to watch what the US military does on the ground and infer that after attacks, US troops search certain regions and not others. The adversary can therefore be expected to *adapt* his attacks to avoid detection, based on the model of search behavior exhibited by US troops that he may be able to monitor. The problem for the agent (US troops) is to *anticipate* how the adversary might adapt, and use that anticipatory knowledge to discover the location(s) of weapons caches supporting the terror attacks carried out by the adversary.
- **The Tiger Detection Problem.** In the tiger detection problem, we are interested in finding preferred locations where the tiger prefers to reside, based on locations of tiger kills and information about the suitability of various regions on the ground for a tiger dwelling. However, the tiger is a solitary and intelligent animal who would vastly prefer to stay away from human contact. Tigers—and other animals—have, in recent years, been found occasionally in habitats that are different from the ones they usually inhabit. Can wildlife conservationists determine how a tiger is likely to adapt its pattern of behavior as we attempt to search for it, based on the location of tiger kills and habitat information? The ability to do this would significantly enhance tiger conservation efforts.
- **The Criminal Identification Problem.** Burglars, serial killers, and other criminals have a clear interest in avoiding being found by geospatial abduction methods. Should they learn and/or understand what tools and investigative techniques law enforcement officials have at their disposal, then they can adapt their own behavior to minimize the probability of being discovered. Of course, law enforcement officers would like to anticipate how criminals might seek to evade them, and accordingly adjust their own strategy to hunt down the criminals involved.

In contrast to the above problems where the agent seeks to solve a geospatial abduction problem in the presence of an adversary which is adapting its behavior, the Virus Host Identification Problem involves an adversary that is certainly changing—but in its physical makeup rather than in a geospatial sense. Hence, the work described in this chapter does not directly apply to this situation.[1]

[1] Some of the results in this chapter may still prove useful in situations like the Virus Host Identification problem. Consider the non-deterministic geospatial abduction algorithms from Chapter 2 such as GREEDY-KSEP-OPT2. Suppose we run one of these algorithms n times, creating n expla-

In the rest of this chapter, we will first describe how geospatial abduction problems can be viewed as two-player games. Then, we will define methods by which an adversary (*e.g.*, insurgents carrying out IED attacks) can best position partner locations (*e.g.*, locations of weapons caches) so as to minimize discovery if the work in the previous sections were to be used. Of course, the agent performing such game theoretic reasoning would like to minimize the probability of being outsmarted by the adversary, so our next section will focus on how to maximize the adversary's probability of being successful. All of these sections will include complexity results, algorithms, and in some cases, approximation algorithms and implementation hints. Finally, we will describe a suite of experiments we have conducted showing that these algorithms work well in a real-life IED cache detection problem using real data drawn from Baghdad.

4.2 Geospatial Abduction as a Two-Player Game

Throughout this chapter, we view geospatial abduction as a two-player game which, like many results in game theory, follows the cyclic outline given below.

1. The *adversary* chooses a set $\mathscr{O}$ of locations where he/it will carry out certain actions that the *agent* can observe.
2. In order to carry out these attacks, the *adversary* tries to find a set of locations that constitute an explanation $\mathscr{E}_1$ of the set $\mathscr{O}$ of observations.
3. The *agent* can detect explanation $\mathscr{E}_1$ using standard geospatial abduction as described in Chapters 2 and 3 (and the *adversary knows this*), so the *adversary* tries to find an explanation $\mathscr{E}_2$ where the *agent* is less likely to detect it.
4. The *agent*, on the other hand, quickly says to himself: "Aha. If the adversary were smart, he would try not to put the explanation at locations in $\mathscr{E}_1$." Instead, putting himself in the adversary's shoes, the *agent* quickly detects that the *adversary* would put the explanation at locations in $\mathscr{E}_2$.
5. The *adversary* can now say "Aha, but if the agent were smart, he would realize that I would not be dumb enough to put the explanation in locations $\mathscr{E}_2$, so he would reason about what I might do and would arrive at the conclusion that I (the *adversary*) would put them at location $\mathscr{E}_3$. At this point, the *agent* would perform a similar analysis (reasoning that the *adversary* would not put it at $\mathscr{E}_2$, etc). This kind of reasoning degenerates into a stage of infinite regress with the *agent* and the *adversary* endlessly trying to stay one step ahead of each other.

Other than purely theoretical interest, the situation where such an infinite regress occurs is neither realistic nor likely. There is so much noise in the real world that

---

nations (some of which may be the same). We can view the virus as having a mixed strategy where it uniformly "selects" one of these n explanations. By framing this as an instance of a Maximal Counter-Adversary Problem (MCA) with an imposed cardinality constraint k (described later in this chapter), an agent can then select the k locations that maximize his payoff with respect to the virus selecting one of the n explanations using a uniform probability distribution.

going too far down this alternating *agent-adversary* reasoning pattern is likely to be extremely complex and not likely to lead to good accuracy or good running time in a real world setting.[2] As a consequence, in this chapter we draw the line at the situation up to point (4) above, *i.e.*, the *adversary* decides to reason up to stage (4) above. But to account for noise, we need to introduce a probabilistic model of how the *adversary* and *agent* reason about each other, together with utilities explaining the value of various situations for them.

4.2.1 Strategies and Rewards

At the end of the day, each player (*agent* or *adversary*) must choose a *strategy* which is merely a subset of the space $\mathscr{S}$.

- When the player considered is the *agent*, the strategy intuitively represents a set of locations that the *agent* believes is the explanation. In the case of the IED detection example, it may be the set of locations that US forces search for an IED weapons cache. In the case of the tiger detection scenario, it may be the set of locations that wildlife experts search for the tiger. The *agent*'s strategy is unknown to the *adversary.*
- When the player considered is the *adversary*, the strategy is the set of locations chosen by the *adversary* to be the true explanation for the observations the agent is causing. For instance, in the tiger detection scenario, the tiger's strategy might be the set of places the tiger dwells before or after making his kills (observations). The *adversary*'s strategy is unknown to the *agent*.

Though "strategy" and "observation" are defined identically, we use separate terms to indicate our intended use. Throughout this chapter, we use $\mathscr{A}$ (resp. $\mathscr{B}$) to denote the strategy of the adversary (resp. agent).

Given a pair $(\mathscr{A},\mathscr{B})$ of adversary-agent strategies, a reward function measures how similar the two sets are. The more similar the two strategies are, 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 4.1 (Reward Function). A *reward function* is any function $\mathbf{rf} : 2^{\mathscr{S}} \times 2^{\mathscr{S}} \rightarrow [0,1]$ that for any k-explanation $\mathscr{A} \neq \emptyset$ and set $\mathscr{B} \subseteq \mathscr{S}$, the function satisfies:

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

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

[2] Our complexity results suggest this may be a #P-hard problem

that if the agent's strategy is the same set as the adversary's, then the reward is the maximum possible. *Thus, the magnitude of the reward function always applies to the reward received by the* agent; if the agent guesses precisely where the adversary's chosen explanation is, then the agent gets the maximum possible reward of 1.

Axiom 2 says that adding a point to $\mathscr{B}$ cannot increase the reward to the agent if that point is already in $\mathscr{B}$, *i.e.*, double-counting of rewards is forbidden.

A reader might wonder why certain natural ideas do not count as valid axioms for a reward function. For instance, why is $\mathbf{rf}(\mathscr{A},\emptyset) = 0$ not an axiom? After all, it could be argued that if the agent does nothing at all, shouldn't there be a zero reward? This is not necessarily true. We can imagine cases where doing something is worse than doing nothing. For instance, in the IED application, the reward to the US military for searching some location (*e.g.*, the house of the Prime Minister of some country) might significantly outweigh the advantage of searching it, especially if there was an error and the hypothetical Prime Minister's house were to be completely empty of any suspicious material. This same argument also explains why reward functions are not necessarily monotonic in the second argument (as the empty strategy for the agent in the preceding discussion could have a higher reward for the agent than the strategy of searching the putative Prime Minister's house).

Nevertheless, there will be cases where some (or many) useful reward functions set $\mathbf{rf}(\mathscr{A},\emptyset) = 0$ and/or are monotonic in nature. We will consider these later in the chapter. Our next step is to state that rewards associate a simple *payoff* for each player.

Observation 4.2.1 *Given adversary strategy $\mathscr{A}$, agent strategy $\mathscr{B}$, and reward function* **rf***, the payoff for the agent is* $\mathbf{rf}(\mathscr{A},\mathscr{B})$ *and the payoff for the adversary is* $-\mathbf{rf}(\mathscr{A},\mathscr{B})$.

Thus, payoffs are positive for the agent and negative for the adversary. It is therefore easy to see that for any reward function and pair $(\mathscr{A},\mathscr{B})$, the corresponding game is a *zero-sum game* [1]. Our complexity analysis assumes all reward functions are polynomially computable. All the specific reward functions we propose in this chapter satisfy this condition.

The following important theorem tells us that every reward function is *submodular*, *i.e.*, the marginal benefit of adding additional points to the agent's strategy decreases as the size of the strategy increases. Submodular functions were defined by us in Chapter 3 — we now adapt this notion of submodularity to the case of binary reward functions and show below that reward functions as defined by us are always submodular.

Proposition 4.1 (Submodularity of Reward Functions). *Every reward function is submodular,* i.e.*, if $\mathscr{B} \subseteq \mathscr{B}'$, and point $p \in \mathscr{S}$ such that $p \notin \mathscr{B}$ and $p \notin \mathscr{B}'$, then* $\mathbf{rf}(\mathscr{A},\mathscr{B}\cup\{p\}) - \mathbf{rf}(\mathscr{A},\mathscr{B}) \geq \mathbf{rf}(\mathscr{A},\mathscr{B}'\cup\{p\}) - \mathbf{rf}(\mathscr{A},\mathscr{B}')$.

Proof. Suppose, by way of contradiction, with $\mathscr{B} \subseteq \mathscr{B}'$, and point $p \in \mathscr{S}$ such that $p \notin \mathscr{B}$ and $p \notin \mathscr{B}'$, then

$$\mathbf{rf}(\mathscr{A},\mathscr{B}\cup\{p\}) - \mathbf{rf}(\mathscr{A},\mathscr{B}) < \mathbf{rf}(\mathscr{A},\mathscr{B}'\cup\{p\}) - \mathbf{rf}(\mathscr{A},\mathscr{B}')$$

We know that $\mathscr{B}' \cup \{p\} = \mathscr{B}' \cup (\mathscr{B} \cup \{p\})$. Hence:

$$\mathbf{rf}(\mathscr{A}, \mathscr{B} \cup \{p\}) - \mathbf{rf}(\mathscr{A}, \mathscr{B}) < \mathbf{rf}(\mathscr{A}, \mathscr{B}' \cup (\mathscr{B} \cup \{p\})) - \mathbf{rf}(\mathscr{A}, \mathscr{B}')$$

Also, we know that $\mathscr{B} = (\mathscr{B} \cup \{p\}) \cap \mathscr{B}'$, so we get:

$$\mathbf{rf}(\mathscr{A}, \mathscr{B} \cup \{p\}) - \mathbf{rf}(\mathscr{A}, (\mathscr{B} \cup \{p\}) \cap \mathscr{B}') < \mathbf{rf}(\mathscr{A}, \mathscr{B}' \cup (\mathscr{B} \cup \{p\})) - \mathbf{rf}(\mathscr{A}, \mathscr{B}')$$

This leads to:

$$\mathbf{rf}(\mathscr{A}, \mathscr{B}') + \mathbf{rf}(\mathscr{A}, \mathscr{B} \cup \{p\}) - \mathbf{rf}(\mathscr{A}, (\mathscr{B} \cup \{p\}) \cap \mathscr{B}') < \mathbf{rf}(\mathscr{A}, \mathscr{B}' \cup (\mathscr{B} \cup \{p\}))$$

which is a clear violation of Axiom 2, hence we have a contradiction.

What this result says is that if the adversary's strategy $\mathscr{A}$ is fixed, then the *marginal benefit* of adding another point to a large agent strategy is not as much as adding the same point to a smaller agent strategy. Simply put, in the case of the IED detection application, if an insurgent group has already placed its weapons at various locations and the agent plans to search either a set $\mathscr{B}$ of points or a superset $\mathscr{B}' \supseteq \mathscr{B}$ of places, and then the agent decides to search one more place p, the increase in overall benefit yielded had they decided to search $\mathscr{B}$ (and p in addition) is greater than the increase in benefit they would get if they searched the superset $\mathscr{B}'$ (and p in addition).

4.2.1.1 Penalizing Reward Function

We explained earlier why $\mathbf{rf}(\mathscr{A}, \emptyset) = 0$ is not an axiom. While this is true of many reward functions, we now give a concrete example of a reward function where we penalize the agent for "bad" strategies because in the real world, executing a bad strategy may have bad consequences. We call this the penalizing reward function.

Definition 4.2 (Penalizing Reward Function). Given a distance *dist*, we define the **penalizing reward function**, $\mathbf{prf}^{dist}(\mathscr{A}, \mathscr{B})$, as follows:

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

The penalizing reward function intuitively works as follows. It starts at 0.5. It then adds to the reward the ratio of the number of points in the adversary's strategy which are within distance *dist* of some point in the agent's strategy to twice the number of points in the adversary's strategy. Intuitively, this ratio is a measure of the effectiveness of the agent in finding locations in the adversary's explanation. After this ratio is added to the reward, the penalizing reward function *penalizes* the agent for points in the agent's explanation that are not within the given distance *dist* of any point in the adversary's explanation. This intuition is captured by the second ratio in the definition of a penalizing reward function.

Let us consider the IED Cache Detection problem and suppose the agent (US forces) chooses to search locations in $\mathscr{B}$, but the actual places where the adversary

has placed his caches are in set $\mathscr{A}$. Intuitively, the $\mathbf{prf}^{dist}$ function gives the agent a reward for all caches within distance *dist* of a cache location contained in the agent's strategy (representing places the agent plans to search), but it penalizes the agent for all locations searched by the agent that are not within distance *dist* of any actual cache placed by the adversary. Thus, it has the effect of forcing the agent to choose where it searches with great care (*e.g.*, to avoid offending local residents in areas that are subject to the search).

In the same vein, in the tiger identification problem, wildlife conservationists will probably incur a cost for each search they make. Unsuccessful searches may have a cost, both in terms of financial cost, as well as in terms of spooking the tiger and making it harder to find.

The result below states that **prf** satisfies the axioms required for reward functions.

Proposition 4.2. $\mathbf{prf}^{dist}$ *is a valid reward function for any* $dist \geq 0$.

Proof. In this proof, we define $pt1(\mathscr{A},\mathscr{B}), pt2(\mathscr{A},\mathscr{B})$ as follows:

$$pt1(\mathscr{A},\mathscr{B}) = \frac{|\{p \in \mathscr{A} | \exists p' \in \mathscr{B} \text{ s.t. } d(p,p') \leq dist\}|}{2 \cdot |\mathscr{A}|}$$

$$pt2(\mathscr{A},\mathscr{B}) = \frac{|\{p \in \mathscr{B} | \ \nexists p' \in \mathscr{A} \text{ s.t. } d(p,p') \leq dist\}|}{2 \cdot |\mathscr{S}|}$$

Hence, $\mathbf{prf}^{dist}(\mathscr{A},\mathscr{B}) = 0.5 + pt1(\mathscr{A},\mathscr{B}) - pt2(\mathscr{A},\mathscr{B})$. As we know the maximum value of both $pt1(\mathscr{A},\mathscr{B}), pt2(\mathscr{A},\mathscr{B})$ is 0.5, we know that **prf** is in $[0,1]$. As $pt1(\mathscr{A},\mathscr{A}) = 0.5$ and $pt2(\mathscr{A},\mathscr{A}) = 0$, then Axiom 1 is also satisfied. Consider **crf** (Definition 4.5). Later, in Proposition 4.3, we show that this function is submodular, meeting Axiom 2. By Definitions 4.5, we can easily show that $pt1(\mathscr{A},\mathscr{B}) = 0.5 \cdot \mathbf{crf}^{dist}(\mathscr{A},\mathscr{B})$. As $pt1(\mathscr{A},\mathscr{B})$ is a positive linear combination of submodular functions, it is also submodular. Now consider $pt2(\mathscr{A},\mathscr{B})$. Any element added to any set $\mathscr{B}$ has the same effect—it either lowers the value by $\frac{1}{2\cdot|\mathscr{S}|}$ or does not affect it— hence it is trivially submodular. Therefore, it follows that **prf** is submodular as it is a positive-linear combination of submodular functions.

The following example revisits the burglary application studied earlier in Chapter 3.

Example 4.1. Consider the two-dimensional space shown in Figure 4.1. Suppose this diagram shows a set of observations (o_i's) depicting locations where burglaries occurred. Furthermore, the police are convinced based on extra-theoretic considerations that the burglaries were carried out by the same burglar (*e.g.*, by examining fingerprints or the burglar's modus operandi). All points in this figure that are not shown in black are assumed to be feasible, and certain locations p_i are marked with numbers (just the i for readability). Suppose the burglar's actual places of residence (*e.g.*, home and office) are given by $\mathscr{A} = \{p_{40}, p_{46}\}$ while the set $\mathscr{B} = \{p_{38}, p_{41}, p_{44}, p_{56}\}$ represents locations that that the police wish to search. Suppose we consider distance $dist = 100$ meters. There is only one point in $\mathscr{A}$ that

is within 100 meters of a point in $\mathscr{B}$ (point p_{40}) and 3 points in $\mathscr{B}$ more than 100 meters from any point in $\mathscr{A}$ (points p_{38}, p_{44}, p_{56}). These relationships are shown visually in Figure 4.1. Hence, $\mathbf{prf}^{dist}(\mathscr{A}, \mathscr{B}) = 0.5 + 0.25 - 0.011 = 0.739$.

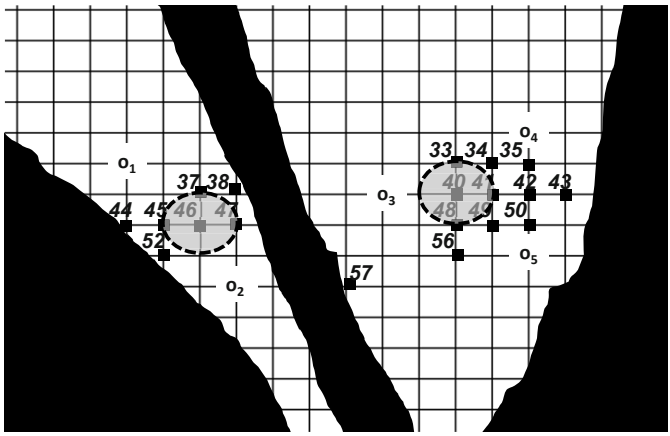

Fig. 4.1 Dashed circles encompass all feasible points within 100 meters from explanation $\{p_{40}, p_{45}\}$. Regions shown in black are deemed infeasible by the supplied feasibility predicate.

Definition 4.3. A reward function is said to be *zero-starting* if $\mathbf{rf}(\mathscr{A}, \emptyset) = 0$.

Thus, a reward function is zero-starting if the agent gets no reward for having an empty strategy. As mentioned earlier in this chapter, not all reward functions should be required to be zero-starting; however, there may be plenty of zero-starting reward functions that are useful in many cases. We now define *monotonic* reward functions.

Definition 4.4. A reward function, **rf**, is **monotonic** if (i) it is zero-starting and (ii) if $\mathscr{B} \subseteq \mathscr{B}'$ then $\mathbf{rf}(\mathscr{A}, \mathscr{B}) \leq \mathbf{rf}(\mathscr{A}, \mathscr{B}')$.

Note that in standard mathematics, monotonic functions in general are not required to satisfy the first condition in the above definition. However, our definition of monotonic reward functions requires them to be zero-starting as well. We now define several example monotonic reward functions, starting with the "cutoff" reward function.

4.2.1.2 Cutoff Reward Function

The intuition behind the *cutoff reward function* **crf** is simple. Suppose we are given a distance *dist* (the "cutoff" distance). The cutoff function looks at the percentage of locations in the adversary's strategy that are within *dist* units of some point in the agent's strategy.

Definition 4.5 (Cutoff Reward Function). Reward function based on a cutoff distance *dist*.

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

Intuitively, the cutoff reward function assumes that if the agent's strategy includes a point p' and there is a location p in the adversary's strategy that is within *dist* units of p', then the agent will discover it. Returning to our IED Cache Detection problem, intuitively this says that the agent will find all caches used by insurgents if those caches are within *dist* units of a location that the agent has decided to search. In the case of the Tiger Identification problem, likewise, this says that if wildlife experts decide to search a point p' that is within *dist* units of an actual tiger dwelling, then the wildlife experts will in fact discover this.

Thus, in the case of the IED Cache Detection application, the cutoff reward function intuitively specifies the percentage of enemy caches actually discovered by the agent's strategy, while in the case of the Tiger Identification problem, it specifies the percentage of tiger dwellings that actually exist. The following proposition shows that the cutoff reward function is a valid, monotonic reward function.

Proposition 4.3. $\mathbf{crf}^{dist}$ *is a valid, monotonic reward function for any* $dist \geq 0$.

Proof. CLAIM 1: **crf** satisfies reward Axiom 1.
Clearly, if $\mathscr{B} = \mathscr{A}$, then the numerator is $|\mathscr{A}|$, which equals the denominator.

CLAIM 2: **crf** satisfies reward function Axiom 2.
Suppose, by way of contradiction, there exists explanations $\mathscr{B}, \mathscr{B}'$ such that $\mathscr{B} \cup \mathscr{B}'$ is an explanation and $\mathbf{crf}^{dist}(\mathscr{A},\mathscr{B} \cup \mathscr{B}') > \mathbf{crf}^{dist}(\mathscr{A},\mathscr{B}) + \mathbf{rf}(\mathscr{A},\mathscr{B}') - \mathbf{rf}(\mathscr{A},\mathscr{B} \cap \mathscr{B}')$. Therefore, $card(\{p \in \mathscr{A} | \exists p' \in \mathscr{B} \cup \mathscr{B}' \text{ s.t. } d(p,p') \leq dist\})$ is greater than $card(\{p \in \mathscr{A} | \exists p' \in \mathscr{B} \text{ s.t. } d(p,p') \leq dist\}) + card(\{p \in \mathscr{A} | \exists p' \in \mathscr{B}' \text{ s.t. } d(p,p') \leq dist\}) - card(\{p \in \mathscr{A} | \exists p' \in \mathscr{B} \cap \mathscr{B}' \text{ s.t. } d(p,p') \leq dist\})$. We have a contradiction; indeed, by basic set theory we see that both sides of this strict inequality are actually equal.

CLAIM 3: **crf** is zero-starting.
Clearly, if $\mathscr{B} = \emptyset$, the numerator must be 0, and the statement follows.

CLAIM 4: **crf** is monotonic.
Suppose, by way of contradiction, there exists $\mathscr{B} \subseteq \mathscr{B}'$ such that $\mathbf{rf}(\mathscr{A},\mathscr{B}) > \mathbf{rf}(\mathscr{A},\mathscr{B}')$. Then $card(\{p \in \mathscr{A} | \exists p' \in \mathscr{B} \text{ s.t. } d(p,p') \leq dist\}) > card(\{p \in \mathscr{A} | \exists p' \in \mathscr{B}' \text{ s.t. } d(p,p') \leq dist\})$. Clearly, this is not possible as $\mathscr{B} \subseteq \mathscr{B}'$ and we have a contradiction.

The following example illustrates how the **crf** reward function works.

Example 4.2. Consider Example 4.1. Here, $\mathbf{crf}^{dist}(\mathscr{A},\mathscr{B})$ returns 0.5 as one element of $\mathscr{A}$ is within 100 meters of an element in $\mathscr{B}$.

4.2.1.3 Falloff Reward Function

The cutoff reward function uses *dist* to decide whether a location in the adversary's strategy will be discovered by the agent or not. Thus, even if an adversary's chosen location is just an inch outside the *dist* bound from the closest point in the agent's strategy, this function would say that the adversary's location would not be found. This may not always be realistic.

In contrast, the falloff reward function **frf** defined below says that for each location p in the adversary's strategy (*e.g.*, where the insurgent group chooses to place its IED weapons locations, or where the tiger actually resides in practice), the probability that the agent will discover this location is inversely proportional to the distance of p from the nearest point p' that is in the agent's strategy.

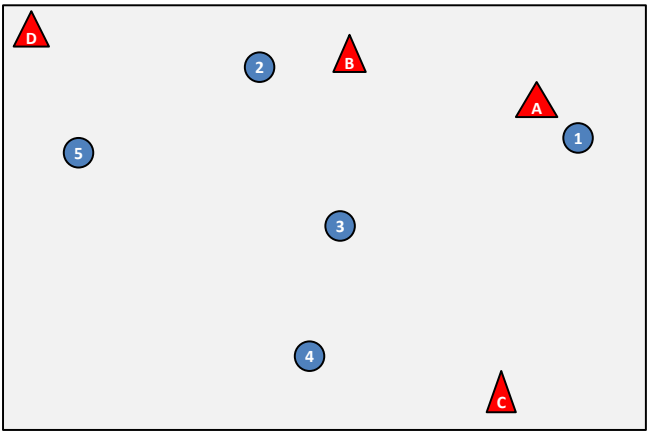

Fig. 4.2 Example adversary and agent strategies for the falloff reward function (**frf**). The agent's strategy consists of points marked by blue circles, while the adversary's strategy consists of the red triangles.

Figure 4.2 shows a simple example. In this figure, we look at each location in the adversary's strategy; in the IED detection application, for example, these are the locations where the insurgents decided to place their weapons caches. For each such adversary location, we find the location in the agent's strategy that is closest to it. The table below summarizes this situation.

Example 4.3.

Adversary Location	Nearest Agent Location	Distance
A	1	1
B	2	2
C	4	5
D	5	4

The falloff reward function assigns a reward to the agent that is inversely proportional to the distances between each *A* and 1, *B* and 2, *C* and 4, and *D* and 5.

As long as the reward increases as the distances in the above table decreases, we have a function that rewards the agent for a strategy which comes "close" (in terms of distance) to the strategy of the adversary. The falloff reward function defined below implements this intuition in one way—many other ways to achieve the same intuition, albeit with slightly different definitions of the falloff reward function, are possible.

Definition 4.6 (Falloff Reward Function). Reward function with value based on minimal distances between points.

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

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.

The example below extends the preceding example.

Example 4.4. Let us continue with the situation in the table shown in Example 4.3. In this case, we see that when the agent strategy is $\{1,2,3,4,5\}$ and the adversary's strategy is $\{A,B,C,D\}$. In this case, $|\mathscr{A}| = 4$ as the adversary has placed four caches. Then the falloff reward function returns the following:

$$\frac{1}{4+1^2} + \frac{1}{4+2^2} + \frac{1}{4+5^2} + \frac{1}{4+4^2}$$

which turns out to be $0.2+0.125+0.034+0.05 = 0.409$ which is the value returned by the falloff reward function.

The following result specifies that the falloff reward function **frf** satisfies the axioms to be a reward function—and, additionally, is monotonic.

Proposition 4.4. frf *is a valid, monotonic reward function.*

Proof. CLAIM 1: **frf** satisfies all reward function axioms (*i.e.*, is valid).

Bounds We must show $\mathbf{rf}(\mathscr{A},\mathscr{B}) \in [0,1]$. For each point $p \in \mathscr{A}$, let $l_p^{\mathscr{B}} = \min_{p'\in\mathscr{B}} d(p,p')^2$. By the definition of the distance function d, we know $0 \leq l_p^{\mathscr{B}} < \infty$. Now let function $f(l_p^{\mathscr{B}}) = \frac{1}{|\mathscr{A}|+\min_{p'\in\mathscr{B}} d(p,p')^2} = \frac{1}{|\mathscr{A}|+l_p^{\mathscr{B}}}$. Since $0 \leq l_p^{\mathscr{B}} < \infty$, we see $0 < f(l_p^{\mathscr{B}}) \leq \frac{1}{|\mathscr{A}|}$. Clearly, the summation over $|\mathscr{A}|$ points $p \in \mathscr{A}$ yields an answer in $\left(0, |\mathscr{A}| \cdot \frac{1}{|\mathscr{A}|}\right] = (0,1] \subset [0,1]$.

Axiom 1 If $\mathscr{B} = \mathscr{A}$, for each $p \in \mathscr{A}$, there exists $p' \in \mathscr{B}$ such that $d(p,p') = 0$. Hence, by the definition of **frf**, $\mathbf{frf}(\mathscr{A},\mathscr{B}) = 1$ in this case.

Axiom 2 We must show that our version of the triangle inequality holds, that is $\mathbf{rf}(\mathscr{A},\mathscr{B}\cup\mathscr{B}') \leq \mathbf{rf}(\mathscr{A},\mathscr{B})+\mathbf{rf}(\mathscr{A},\mathscr{B}')-\mathbf{rf}(\mathscr{A},\mathscr{B}\cap\mathscr{B}')$. From above, $\mathbf{rf}(\mathscr{A},\mathscr{B}\cup\mathscr{B}') = \sum_{p\in\mathscr{A}} f(l_p^{\mathscr{B}\cup\mathscr{B}'})$. For each point $p\in\mathscr{A}$, let $p_* = argmin_{p'\in\mathscr{B}\cup\mathscr{B}'} d(p,p')^2$. Without loss of generality, assume $p_*\in\mathscr{B}$, then $l_p^{\mathscr{B}} = l_p^{\mathscr{B}\cup\mathscr{B}'}$ thus $f(l_p^{\mathscr{B}}) = f(l_p^{\mathscr{B}\cup\mathscr{B}'})$. Since $p_*\in\mathscr{B}$, we have $p_*\in\mathscr{B}\cap\mathscr{B}'$ or $p_*\in\mathscr{B}\cap\bar{\mathscr{B}}'$.

If $p_*\in\mathscr{B}\cap\mathscr{B}'$: Then $f(l_p^{\mathscr{B}\cap\mathscr{B}'}) = f(l_p^{\mathscr{B}})$. However, since $p_*\in\mathscr{B}'$ we have, as above, $f(l_p^{\mathscr{B}'}) = f(l_p^{\mathscr{B}}) = f(l_p^{\mathscr{B}\cup\mathscr{B}'})$. Thus

$$\sum_{p\in\mathscr{A}}\left[f(l_p^{\mathscr{B}})+f(l_p^{\mathscr{B}'})-f(l_p^{\mathscr{B}\cap\mathscr{B}'})\right] \tag{4.1}$$

$$=\sum_{p\in\mathscr{A}}\left[f(l_p^{\mathscr{B}\cup\mathscr{B}'})+f(l_p^{\mathscr{B}\cup\mathscr{B}'})-f(l_p^{\mathscr{B}\cup\mathscr{B}'})\right] \tag{4.2}$$

$$=\sum_{p\in\mathscr{A}} f(l_p^{\mathscr{B}\cup\mathscr{B}'}) \tag{4.3}$$

So $\mathbf{rf}(\mathscr{A},\mathscr{B}\cup\mathscr{B}') = \mathbf{rf}(\mathscr{A},\mathscr{B})+\mathbf{rf}(\mathscr{A},\mathscr{B}')-\mathbf{rf}(\mathscr{A},\mathscr{B}\cap\mathscr{B}')$ for this case.

If $p_*\in\mathscr{B}\cap\bar{\mathscr{B}}'$: From above, we are still guaranteed that $f(l_p^{\mathscr{B}}) = f(l_p^{\mathscr{B}\cup\mathscr{B}'})$, thus $\mathbf{rf}(\mathscr{A},\mathscr{B}\cup\mathscr{B}') = \mathbf{rf}(\mathscr{A},\mathscr{B})$. This reduces our problem to showing $\mathbf{rf}(\mathscr{A},\mathscr{B}')-\mathbf{rf}(\mathscr{A},\mathscr{B}\cap\mathscr{B}')\geq 0$. However, **rf** is monotonic (shown below); since $\mathscr{B}\cap\mathscr{B}'\subseteq\mathscr{B}'$, then $\mathbf{rf}(\mathscr{A},\mathscr{B}\cap\mathscr{B}')\leq\mathbf{rf}(\mathscr{A},\mathscr{B}')$ and our claim holds.

A similar proof holds for the case $p_*\in\mathscr{B}'$.

CLAIM 2: **frf** is monotonic and zero-starting. The property of zero-starting follows directly from the definition of **frf**.

By way of contradiction, assume there is some $\mathscr{B}\subset\mathscr{B}'$ such that $\mathbf{rf}(\mathscr{A},\mathscr{B}) > \mathbf{rf}(\mathscr{A},\mathscr{B}')$. Then, as above, $\sum_{p\in\mathscr{A}} f(l_p^{\mathscr{B}}) > \sum_{p\in\mathscr{A}} f(l_p^{\mathscr{B}'})$. However, since $\mathscr{B}\subset\mathscr{B}'$, we have $l_p^{\mathscr{B}}\geq l_p^{\mathscr{B}'}$ for each $p\in\mathscr{A}$. Similarly, $f(l_p^{\mathscr{B}})\leq f(l_p^{\mathscr{B}'})$ and thus $\sum_{p\in\mathscr{A}} f(l_p^{\mathscr{B}})\leq\sum_{p\in\mathscr{A}} f(l_p^{\mathscr{B}'})$, which is our contradiction.

4.2.1.4 Weighted Reward Function

In all the specific examples of reward functions presented thus far, all locations in both the agent's strategy and the adversary's strategy are considered to be equally important (though there is a hint that this may not be the case when we discussed searching a Prime Minister's house in the discussion on penalizing reward functions). We now define *weighted reward functions*, where each location p' in the agent's strategy has an associated weight.

Returning to the IED detection example, the weight of searching the Prime Minister's house might be very low, while a national security analyst may set the weight of searching a mosque or the grounds of an extremist *madrasah* (religious school) to be much higher. Likewise, in the case of the tiger detection example, the wildlife conservation expert might set the weight of searching a particular location in a way

that is consistent with the suitability of the habitat (*e.g.*, density of ground cover or forest, abundance of prey) for the tiger to inhabit. Thus, the weighted reward function **wrf** assigns a greater reward for being "closer" to points in $\mathscr{A}$ that have high weight than those with lower weights.

Definition 4.7 (Weighted Reward Function). Given weight function $W : \mathscr{S} \to \mathbb{R}^+$, and a cutoff distance *dist* we define the *weighted reward function* to be:

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

Thus, the weighted reward function proceeds as follows. For a given cutoff distance *dist*, it considers each location p in the adversary's strategy and checks if there exists a location p' in the agent's strategy. If so, it adds the weight of p (which intuitively indicates the importance of the adversary's location p from the point of view of the GAP application) to a running total. Once all such locations $p \in \mathscr{A}$ have been considered, it divides the total obtained by the sum of weights of all points in $\mathscr{A}$ to obtain a fractional value of the locations that the agent is expected to discover, should the agent and the adversary use strategies $\mathscr{B}$ and $\mathscr{A}$, respectively.

The following result establishes that the weighted reward function satisfies the axioms required to be a reward function—and, moreover, is monotonic.

Proposition 4.5. $\mathbf{wrf}^{(W,dist)}(\mathscr{A},\mathscr{B})$ *is a valid, monotonic reward function.*

Proof. CLAIM 1: **wrf** satisfies all reward function axioms (*i.e.*, is valid).

Domain We must show $\mathbf{wrf}^{(W,dist)}(\mathscr{A},\mathscr{B}) \in [0,1]$. As $(\mathscr{B}\cap\mathscr{A}) \subseteq \mathscr{A}$ and W only returns positive values, this function can only return values in $[0,1]$.

Axiom 1 If $\mathscr{B} = \mathscr{A}$, then for each $p \in \mathscr{A}$, there exists $p' \in \mathscr{B}$ such that $d(p,p') = 0$. This causes the numerator to equal $\sum_{p\in\mathscr{B}} W(p)$. As $\mathscr{B} = \mathscr{A}$, the numerator is equivalent to the denominator, so $\mathbf{wrf}(\mathscr{A},\mathscr{B}) = 1$ in this case.

Axiom 2 We must show the inequality $\mathbf{wrf}^{(W,dist)}(\mathscr{A},\mathscr{B}\cup\mathscr{B}') \leq \mathbf{wrf}^{(W,dist)}(\mathscr{A},\mathscr{B}) + \mathbf{wrf}^{(W,dist)}(\mathscr{A},\mathscr{B}') - \mathbf{wrf}^{(W,dist)}(\mathscr{A},\mathscr{B}\cap\mathscr{B}')$. This proof is similar to the proof of Axiom 2 in Proposition 4.3.

CLAIM 2: **wrf** is monotonic and zero-starting.
The property of zero-starting if shown by when $\mathscr{B} = \emptyset$, the numerator must be 0, hence, $\mathbf{wrf}(\mathscr{A},\emptyset) = 0$. By way of contradiction, assume there is some $\mathscr{B} \subset \mathscr{B}'$ such that $\mathbf{wrf}^{(W,dist)}(\mathscr{A},\mathscr{B}) > \mathbf{wrf}^{(W,dist)}(\mathscr{A},\mathscr{B}')$. Then

$$\frac{\sum_{\{p\in\mathscr{A} \mid \exists p'\in\mathscr{B}\ s.t.\ d(p,p')\leq dist\}} W(p)}{\sum_{p'\in\mathscr{A}} W(p')} > \frac{\sum_{\{p\in\mathscr{A} \mid \exists p'\in\mathscr{B}'\ s.t.\ d(p,p')\leq dist\}} W(p)}{\sum_{p'\in\mathscr{A}} W(p')}$$

Since $\mathscr{B} \subset \mathscr{B}'$, we have

$$\frac{\sum_{\{p\in\mathscr{A} \mid \exists p'\in\mathscr{B}\ s.t.\ d(p,p')\leq dist\}} W(p)}{\sum_{p'\in\mathscr{A}} W(p')} >$$

$$\frac{\sum_{\{p \in \mathscr{A} | \exists p' \in \mathscr{B} \text{ s.t. } d(p,p') \leq dist\}} W(p)}{\sum_{p' \in \mathscr{A}} W(p')} + \frac{\sum_{\{p \in \mathscr{A}' | \exists p' \in (\mathscr{B}' \cap \mathscr{B}) \text{ s.t. } d(p,p') \leq dist\}} W(p)}{\sum_{p' \in \mathscr{A}} W(p')}$$

Where $\mathscr{A}' = \{p \in \mathscr{A} \mid \nexists p' \in \mathscr{B} \text{ s.t. } d(p,p') \leq dist\}$. Hence,

$$0 > \mathbf{wrf}^{(W,dist)}(\mathscr{A}', \mathscr{B}' \cap \mathscr{B})$$

Which violates the first axiom, which was shown to apply to $\mathbf{wrf}^{(W,dist)}$ by Claim 1—a contradiction.

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

4.2.2 Incorporating Mixed Strategies

Of course, neither the adversary nor the agent wants to be entirely predictable, as predictability (in the case of the adversary) would mean that the agent has an easy way to uncover its hidden locations, while predictability (in the case of the agent) means the adversary can easily avoid being uncovered. To achieve unpredictability, they are not likely to pick one strategy and stick with it; rather, they are likely to pick strategies in accordance with some probability distribution. In this section, we introduce probability density functions (pdfs) over strategies (or *mixed strategies* as they are commonly referred to in game theory [1]) and introduce the notion of expected reward. We first present *explanation/strategy functions* which return an explanation (resp. strategy) of a certain size for a given set of observations.

Definition 4.8 (Explanation/Strategy Function). An *explanation (resp. strategy) function* is any function $\mathsf{ex\_fcn} : 2^{\mathscr{S}} \times \mathbb{N} \rightarrow 2^{\mathscr{S}}$ (resp. $\mathsf{sf} : 2^{\mathscr{S}} \times \mathbb{N} \rightarrow 2^{\mathscr{S}}$) that, given a set $\mathscr{O} \subseteq \mathscr{S}$ and $k \in \mathbb{N}$, returns a set $\mathscr{E} \subseteq \mathscr{S}$ such that $\mathscr{E}$ is a k-sized explanation of $\mathscr{O}$ (resp. $\mathscr{E}$ is a k-sized subset of $\mathscr{S}$). Let $\mathbf{EF}$ be the set of all explanation functions.

Intuitively, all that an explanation function does is to return an explanation of size k, given a set of observations and an integer k as input. In contrast, a strategy function just returns a strategy of size k.

Example 4.5. Continuing with Example 4.1, we now define two functions $\mathsf{ex\_fcn}_1$ and $\mathsf{ex\_fcn}_2$. Given the set $\mathscr{O}$ (defined in Example 4.1) as input and $k \leq 3$, these functions give the following results:

$$\mathsf{ex\_fcn}_1(\mathscr{O}, 3) = \{p_{42}, p_{45}, p_{48}\}$$
$$\mathsf{ex\_fcn}_2(\mathscr{O}, 3) = \{p_{40}, p_{46}\}$$

These sets may correspond to explanations from various sources. Perhaps they correspond to the answer of an algorithm that police officials use to solve GAPs. Con-

versely, they could also be places the burglar thinks would make most sense for him to inhabit.

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 $\mathscr{S}$ and so we assume explanation functions are only invoked with $k \leq (M+1) \times (N+1)$.

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 chapter. 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 (*e.g.*, answers to the questions "where is the adversary going to put his weapons caches?" and "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 4.9 (Explanation Function Distribution). Given a space $\mathscr{S}$, real numbers α, β, feasibility predicate feas, and an associated set of explanation functions $\mathbf{EF}$, an *explanation function distribution* is a finitary[3] probability distribution $\mathsf{exfd} : \mathbf{EF} \rightarrow [0,1]$ with $\sum_{\mathsf{ex\_fcn} \in \mathbf{EF}} \mathsf{exfd}(\mathsf{ex\_fcn}) = 1$. Let $\mathbf{EFD}$ be a set of explanation function distributions.

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

Example 4.6. Following from Example 4.5, we define the explanation function distribution $\mathsf{exfd}_{burglar}$ that assigns a uniform probability to explanation functions in the set $\mathsf{ex\_fcn}_1, \mathsf{ex\_fcn}_2$ (*i.e.*, $\mathsf{exfd}_{burglar}(\mathsf{ex\_fcn}_1) = 0.5$).

We now define an *expected reward* that takes mixed strategies specified by explanation function distributions into account to compute an expected value for the reward function to return.

Definition 4.10 (Expected Reward). Given a reward function **rf**, and explanation function distributions $\mathsf{exfd}_{adv}, \mathsf{exfd}_{ag}$ for the adversary and agent respectively, the *expected reward* is a function $\mathsf{EXR}^{\mathbf{rf}} : \mathbf{EFD} \times \mathbf{EFD} \rightarrow [0,1]$. For some explanation function distributions $\mathsf{exfd}_{adv}, \mathsf{exfd}_{ag}$, we define $\mathsf{EXR}^{\mathbf{rf}}(\mathsf{exfd}_{adv}, \mathsf{exfd}_{ag})$ as follows:

$$\sum_{\mathsf{ex\_fcn}_{adv} \in \mathbf{EF}_{adv}} \left(\mathsf{exfd}_{adv}(\mathsf{ex\_fcn}_{adv}) \cdot \sum_{\mathsf{ex\_fcn}_{ag} \in \mathbf{EF}_{ag}} \mathsf{exfd}_{ag}(\mathsf{ex\_fcn}_{ag}) \cdot \mathbf{rf}(\mathsf{ex\_fcn}_{adv}, \mathsf{ex\_fcn}_{ag}) \right)$$

This definition can be explained as follows. We consider each possible explanation function $\mathsf{ex\_fcn}_{ad}$ that the adversary might use. For each possible explanation function $\mathsf{ex\_fcn}_{ag}$ used by the agent, we find the *expected reward* to the agent, which

[3] That is, exfd assigns non-zero probabilities to only finitely many explanation functions.

is the probability of the agent using explanation function $\mathsf{exfd}_{ag}(\mathsf{ex\_fcn}_{ag})$ times the reward to the agent if he uses $\mathsf{ex\_fcn}_{ag}$ and the adversary uses $\mathsf{ex\_fcn}_{adv}$—this is the product $\mathsf{exfd}_{ag}(\mathsf{ex\_fcn}_{ag}) \cdot \mathbf{rf}(\mathsf{ex\_fcn}_{adv}, \mathsf{ex\_fcn}_{ag})$ in the above formula. However, this product must be multiplied by the probability that the adversary uses explanation function $\mathsf{ex\_fcn}_{adv}$, yielding $\mathsf{exfd}_{adv} \times \mathsf{exfd}_{ag}(\mathsf{ex\_fcn}_{ag}) \cdot \mathbf{rf}(\mathsf{ex\_fcn}_{adv}, \mathsf{ex\_fcn}_{ag})$. This expression is then summed up over all possible explanation functions $\mathsf{ex\_fcn}_{adv}$ that the adversary might use to give the final expected reward (for the agent).

In this chapter, we will generally not deal with expected reward directly. Rather, we handle 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.

4.3 Selecting a Strategy for the Adversary

In this section, we consider the problem where the adversary has already decided what the set $\mathscr{O}$ of observations should be (*e.g.*, in the case of the insurgents, this would correspond to the insurgents having selected the targets of their terror attacks, while in the case of the burglar, this corresponds to the set of targets the burglar plans to break into), and now he wants to choose the best strategy $\mathscr{A}$ to carry out his nefarious deeds. Of course, the strategy $\mathscr{A}$ needs to be an explanation for $\mathscr{O}$ with respect to a given feasibility predicate. 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.*, the set of locations for his weapons caches). Even though he can use "expected reward" to measure how close the agent will be to the adversary's explanation, only the agent's strategy is mixed because the adversary must physically select his strategy once and for all (*e.g.*, the burglar must decide where to live/work, while the terrorist must decide where to place his weapons caches). In other words, the adversary's strategy is not mixed—it is concrete. In order to account for this, we introduce a special case of expected reward called the *expected adversarial detriment* (of a given strategy $\mathscr{A}$ that he chooses).

Definition 4.11 (Expected Adversarial Detriment). Given any reward function **rf** and explanation function distribution exfd, the *expected adversarial detriment* is the function $\mathsf{EXD}^{\mathbf{rf}} : \mathbf{EFD} \times 2^{\mathscr{S}} \rightarrow [0,1]$ defined as follows:

$$\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{A}) = \sum_{\mathsf{ex\_fcn} \in \mathbf{EF}} \mathbf{rf}(\mathscr{A}, \mathsf{ex\_fcn}(\mathscr{O}, k)) \cdot \mathsf{exfd}(\mathsf{ex\_fcn})$$

Intuitively, the expected adversarial detriment is the expected number of partner locations the agent may uncover *according to the explanation function distribution* exfd that the adversary uses to model the agent. To compute this, we consider each explanation function $\mathsf{ex\_fcn}$ and identify the adversary's reward—which we call "detriment", since it is a measure the adversary wishes to minimize. The product of the two gives the expected detriment if in fact the explanation function distribution

is correct and the sum of these products, one for each possible explanation function ex_fcn, gives the total expected detriment.

We illustrate the expected adversary detriment via the following example.

Example 4.7. Following from the previous examples, suppose the burglar is planning to have three safe locations (*e.g.*, his house, his office, and his significant other's house). Suppose, from prior experience of the police (or by appropriate scouting), he expects that police detectives will look for his safe houses using $\mathsf{exfd}_{burglar}$ (see Example 4.6). One suggestion the burglar may consider is to choose safe houses at locations p_{41}, p_{52} (see Figure 4.1). Note that this explanation is optimal with respect to cardinality. With $dist = 100$ meters, he wishes to compute $\mathsf{EXD}^{\mathbf{crf}}(\mathsf{exfd}_{burglar}, \{p_{41}, p_{52}\})$. We first need to find the reward associated with each explanation function (see Example 4.5):

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

Thus, $\mathsf{EXD}^{\mathbf{crf}}(\mathsf{exfd}_{burglar}, \{p_{41}, p_{52}\}) = 0.5 \cdot 1 + 0.5 \cdot 0.5 = 0.75$. Hence, this is probably not the best location for the burglar to position his safe houses with respect to **crf** and exfd, as the expected adversarial detriment associated with this set of locations 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* below.

Definition 4.12 (Optimal Adversarial Strategy). Given a set of observations $\mathscr{O}$, natural number k, reward function **rf**, and explanation function distribution exfd, an **optimal adversarial strategy** is a k-sized explanation $\mathscr{A}$ for $\mathscr{O}$ such that $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{A})$ is minimized.

4.3.1 The Complexity of Finding an Optimal Adversarial Strategy

In this section, we formally define the optimal adversarial strategy (OAS) problem and study its complexity.

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

The result below demonstrates that the known NP-hard problem *Geometric Covering by Discs* [2] is polynomially reducible to OAS. This establishes NP-hardness.

Theorem 4.1. *OAS is NP-hard.*

Proof. CONSTRUCTION: Given an input $\langle P, b, K \rangle$ of GCD, we create an instance of **OAS** in PTIME as follows:

- Set $\mathscr{S}$ to be a grid large enough that all points in P are also points in $\mathscr{S}$.
- $\mathsf{feas}(p) = \mathsf{TRUE}$ if and only if $p \in P$
- $\alpha = 0$, $\beta = b$, $\mathscr{O} = P$, $k = |P|$
- Let $\mathbf{rf}(\mathscr{E}_1, \mathscr{E}_2) = 1$ if $\mathscr{E}_1 \subseteq \mathscr{E}_2$, and $\frac{|\mathscr{E}_1|}{|\mathscr{S}|}$ otherwise.
  This satisfies reward Axiom 1 as $\mathscr{E}_1 \subseteq \mathscr{S}$, Axiom 2 by definition, and the satisfaction of Axiom 3, along with monotonicity (with respect to the second argument) can easily be shown by the fact that explanations that are not supersets of $\mathscr{E}_1$ (called $\mathscr{E}_2, \mathscr{E}_3$) satisfy $\mathbf{rf}(\mathscr{E}_1, \mathscr{E}_2) = \mathbf{rf}(\mathscr{E}_1, \mathscr{E}_3)$.
- Let $\mathsf{ex\_fcn}(O, num)$ that returns set O when $num = |O|$ and is otherwise undefined. Let $\mathsf{exfd}(\mathsf{ex\_fcn}) = 1$ and 0 otherwise.

CLAIM 1: If $\mathscr{A}$ as returned by **OAS** has a cardinality of $\leq K$, then the answer to GCD is "yes".
Suppose, by way of contradiction, that $card(\mathscr{A}) \leq K$ and GCD answers "no." This is an obvious contradiction as $\mathscr{A}$ is a subset of P (by how feasibility was defined) where all elements of P are within a radius of b and $\mathscr{A}$ also meets the cardinality requirement of GCD.

CLAIM 2: If the answer to GCD is "yes" then $\mathscr{A}$ as returned by **OAS** has a cardinality of less than or equal to K.
Suppose, by way of contradiction, GCD returns "yes" but $\mathscr{A}$ returned by **OAS** has a cardinality greater than K. By the result of GCD, there exists a set P' of cardinality K such that each point in P (hence $\mathscr{O}$) is of a distance $\leq \beta$ from a point in P'. This, along with the definition of feasibility, makes P' a valid K-explanation for $\mathscr{O}$. We note that $\mathsf{ex\_fcn}(P, |P|) = P$ and that exfd assigns this reward function a probability of one. Hence, the expected adversarial detriment for any explanation $\mathscr{A}'$ is $\mathbf{rf}(\mathscr{A}', P)$. As P' is an explanation of cardinality less than $\mathscr{A}$, it follows that $\mathbf{rf}(P', P) < \mathbf{rf}(\mathscr{A}, P)$, which is a contradiction.

The proof of the above theorem yields two insights, stated below as a corollary.

Corollary 4.1. *1. OAS is NP-hard even if the reward function is monotonic (or anti-monotonic).*
2. OAS is NP-hard even if the cardinality of ***EF*** *is 1.*

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, then the following result says that OAS-DEC is in the complexity class NP.

OAS-DEC
INPUT: Space $\mathscr{S}$, feasibility predicate feas, real numbers α, β, set of observations $\mathscr{O}$, natural number k, reward function **rf**, explanation function distribution exfd, and number $R \in [0, 1]$.

OUTPUT: "Yes" if there exists an adversarial strategy $\mathscr{A}$ such that $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{A}) \leq R$, and "no" otherwise.

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

Proof. NP-hardness follows from Theorem 4.1. To show NP-completeness, a witness simply consists of $\mathscr{A}$. We note that, as the reward function is computable in PTIME, finding the expected adversarial detriment for $\mathscr{A}$ and comparing it to R can also be accomplished in PTIME.

Suppose we have an NP oracle that can return an optimal adversarial strategy and suppose this NP oracle returns $\mathscr{A}$. 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 a strategy $\mathscr{B}$ such that $\mathbf{rf}(\mathscr{A}, \mathscr{B})$ is maximized. This is a special case of the problem we discuss in Section 4.4. 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 we will discuss later in Section 4.4. 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 to be #P-hard and difficult to approximate (see Chapter 2). The following theorem shows that finding the set of all optimal strategies (for the adversary) that have an expected adversarial detriment below a certain threshold is #P-hard.

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

Proof. Let us assume that we know one optimal adversarial strategy and can compute the expected adversarial detriment from such a set. Let us call this value D. Given an instance of GCD, we can create an instance of **OAS-DEC** as in Theorem 4.1, where we set $R = D$. Suppose we have an algorithm that produces all adversarial strategies. If we iterate through all strategies in this set, and count all strategies with a cardinality $\leq K$ (the K from the instance of GCD), we have counted all solutions to GCD—thereby solving the counting version of GCD, a #P-hard problem that is difficult to approximate by Lemma 2.1.

This theorem says that it is infeasible for the adversary to find all strategies that are optimal for him.

4.3.2 Pre-Processing and Naive Approach

In this section, we present several algorithms to solve OAS. We first present a simple routine for pre-processing 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 $\mathscr{S}$. Likewise, f is bounded by $\min(|\mathscr{O}|, \Delta)$ but may be much smaller depending on the sparseness of the observations.

Pre-Processing Procedure. Given a space $\mathscr{S}$, a feasibility predicate feas, real numbers $\alpha, \beta \in [0,1]$, and a set $\mathscr{O}$ of observations, we create two lists (similar to a standard inverted index) as follows.

- **Matrix M.** M is an array of size $\mathscr{S}$. For each point $p \in \mathscr{S}$, $M[p]$ 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 if and only if there exists an observation $o \in \mathscr{O}$ such that $\mathsf{feas}(p)$ is true and such that $d(o,p) \in [\alpha, \beta]$.

Thus, $M[p]$ points to all observations that are both feasible and which are within the appropriate lower and upper bounds $[\alpha, \beta]$ in distance from point p. In the case of the insurgent cache detection problem, $M[p]$ specifies the set of attacks that the insurgent terror group wants to carry that could be carried out if the insurgent group had a weapons cache at location p. In contrast, list L points to all locations that can be used to carry out at least one of the attacks that the insurgent group wants to carry out. What is important to note is that if a point p is *not* in L, then point p is not a location where the adversary might want to put his weapons cache.

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

Example 4.8. Consider our running example concerning the burglaries and the burglar's dwellings that started with Example 4.1. The set L consists of $\{p_1, \ldots, p_{67}\}$. The matrix M returns lists of observations that can be associated with each point. For example, $M(p_{40}) = \{o_3, o_4, o_5\}$ and $M(p_{46}) = \{o_1, o_2\}$.

Naive Approach. After pre-processing, 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^*. Furthermore, suppose we eliminate all elements of L^* that are not valid explanations. Finally, for each element of L^*, suppose 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. Furthermore, this approach does not take advantage of the specific reward functions. We now present mixed integer programs (MIPs) to compute the minimal expected adversary detriment when the associated reward function is either **wrf** or **frf**. We first write these mixed integer programs—later, we develop methods to reduce the complexity of solving these MILPs.

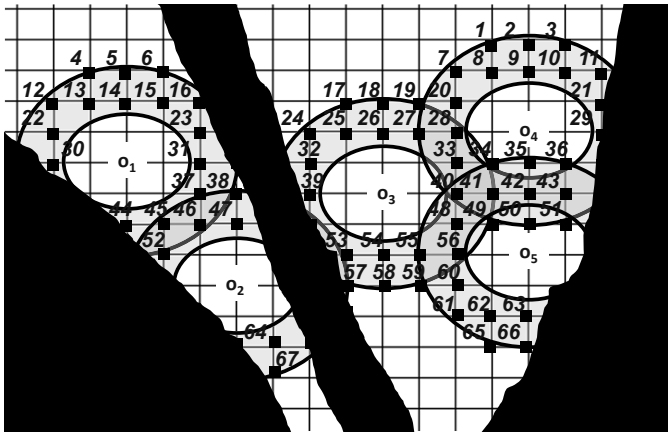

Fig. 4.3 Set L of all possible partners for our burglar's dwelling locations.

4.3.3 Mixed Integer 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 4.3.4, 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 4.9 gives a simple transformation to standard linear form. For readability, we define the MILPs before discussing this transformation.

We start by associating an integer-valued variable X_i with each point $p_i \in L$. Intuitively, X_i is an (unknown) variable which has the value 0 if the adversary chooses not to locate a partner location there, and 1 otherwise.

Definition 4.13 (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 $\mathsf{ex\_fcn}_j \in \mathbf{EF}$, let constant $c_{i,j} = 1$ if and only if $\exists p' \in \mathsf{ex\_fcn}(\mathscr{O}, k)$ 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:

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

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
   $\sum_{p_i \in L_{d(o_j,p_i)\in[\alpha,\beta]}} X_i \geq 1$

We note that the above definition does not define purely linear constraints. Fortunately, these constraints can be easily linearized using the well-known Charnes-Cooper transformation [3].

Example 4.9. Continuing from Examples 4.7 (page 109) and 4.8, suppose the burglar wishes to produce an adversarial strategy $\mathscr{A}$ using **wrf**. Consider the case where we use **crf**, $k \leq 3$, and $dist = 100$ meters as before (see Example 4.7). Clearly, there are 67 variables in these constraints, as this is the cardinality of set L (as per Example 4.8). 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).

As in the case of the weighted reward function above, we can create a MILP for the falloff reward function **frf** as follows, where X_i has a meaning identical to that in the preceding case.

Definition 4.14 (frf MILP). For each $p_i \in L$ and $\mathsf{ex\_fcn}_j \in \mathbf{EF}$, let constant $c_{i,j} = \min_{p' \in \mathsf{ex\_fcn}(\mathcal{O},k)}(d(p_i,p')^2)$. Associate an integer-valued variable X_i with each $p_i \in L$.
Minimize:

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

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 above MILPs correctly yields a solution for the OAS problem under both **wrf** or **frf**.

Proposition 4.6. *Suppose $\mathscr{S}$ is a space, $\mathcal{O}$ is an observation set, real numbers $0 \leq \alpha < \beta \leq 1$, and suppose the* **wrf** *and* **frf** *MILPs are defined as above.*

1. *Suppose* $\mathscr{A} = \{p_1, \ldots, p_n\}$ *is a solution to OAS with* **wrf** *(resp.* **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* $\mathscr{A}$*, then* $\mathscr{A}$ *is a solution to OAS with* **wrf** *(resp.* **frf***).*

Proof. PART 1: Suppose, by way of contradiction, that there is a set of variables $X'_1, \ldots, X'_m$ that is a solution to the constraints such that the value of the objective function is less than if variables $X_1, \ldots, X_n$ were used. Then, there are points $p'_1, \ldots, p'_m$ in set L that correspond with the X_i's such that they cover all observations and that the expected adversarial detriment is minimized. Clearly, this is a contradiction.
PART 2: Suppose, by way of contradiction, that there is a set of points $\mathscr{A}'$ such that the expected adversarial detriment is less than $\mathscr{A}$. Clearly, $\mathscr{A}$ is a valid explanation that minimizes the expected adversarial detriment by the definition of the constraints—hence a contradiction.

The result below states that setting up either set of constraints can be performed in polynomial time, where computing the $c_{i,j}$ constants is the dominant operation.

Proposition 4.7. *Setting up the* **wrf/frf** *constraints can be accomplished in* $O(|\mathbf{EF}| \cdot k \cdot |\mathscr{O}| \cdot \Delta)$ *time (provided the weight function* W *can be computed in constant time).*

Proof. First, we must run POSS-PART, which requires $O(|\mathscr{O}| \cdot \Delta)$ operations. This results in a list of size $O(|\mathscr{O}| \cdot \Delta)$. For each explanation function, ex_fcn, we must compare every element in L with each element of $\mathsf{ex\_fcn}(\mathscr{O})$, which would require $O(k \cdot |\mathscr{O}| \cdot \Delta)$ time. As there are $|\mathbf{EF}|$ explanation functions, the statement follows.

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 $\mathscr{S}$, and α, β.

Proposition 4.8. *The* **wrf/frf** *constraints have* $O(|\mathscr{O}| \cdot \Delta)$ *variables and* $1 + |\mathscr{O}|$ *constraints.*

Proof. As list L is of size $O(|\mathscr{O}| \cdot \Delta)$, and there is one variable for every element of L, there are $O(|\mathscr{O}| \cdot \Delta)$ variables. As there is a constraint for each observation, plus a constraint to ensure the cardinality requirement (k) is met, there are $1 + |\mathscr{O}|$ constraints.

The MILPs for **wrf** and **frf** appear nonlinear as the objective function is fractional. However, as the denominator is non-zero and strictly positive, the Charnes-Cooper transformation [3] 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 and hence, this transformation is very standard.

Proposition 4.9. *The* **wrf/frf** *constraints can be transformed into a purely linear-integer form in* $O(|\mathscr{O}|^2 \cdot \Delta)$ *time.*

Proof. Obviously, in both sets of constraints, the denominator of the objective function is strictly positive and non-zero. Hence, we can directly apply the Charnes-Cooper transformation [3] to obtain a purely integer-linear form. This transformation requires $O(\textit{number of variables} \times \textit{number of constraints})$. Hence, the $O(|\mathscr{O}|^2 \cdot \Delta)$ time complexity of the operation follows immediately from Proposition 4.8.

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

Proposition 4.10. *Given the constraints of Definition 4.13 or Definition 4.14, 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 $O(|\mathscr{O}|^{3.5} \cdot \Delta^{3.5})$ time.*

Proof. CLAIM 1: The linear relaxation of Definition 4.13 or Definition 4.14 provides a lower bound on the objective function value for the full integer-linear constraints. As an optimal value returned by the integer-linear constraints would also be a solution, optimal with respect to minimality, for the linear relaxation, the statement follows.
CLAIM 2: The lower bound can be obtained in $O(|L|^{3.5})$ time.
As there is a variable for each element of L, the size of L is $O(|\mathscr{O}| \cdot \Delta)$, and the claim follows immediately from the result of [4].

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 4.11. *Consider the MILPs in Definition 4.13 and Definition 4.14. 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.*

Proof. Suppose, by way of contradiction, that the solution for the objective function on the reduced MILP would be less than the actual MILP. Let $X_1, \ldots, X_n$ be the variables set to 1 for the reduced MILP in this scenario. We note, that setting the same variables to the full MILP would also be a solution, and could not possibly be less than a minimal solution. This is a contradiction.

4.3.4 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 above with 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.[4] It also turns out that finding these special explanations can be accomplished using a MILP which will often have significantly less variables than the MILP's 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:

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

where for each $p_i \in L$ and $\mathsf{ex\_fcn}_j \in \mathbf{EF}$, $c_{i,j} = 1$ if and only if $\exists p' \in \mathsf{ex\_fcn}_j(\mathscr{O}, k)$ 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 $const_i$:

$$const_i = \sum_{\mathsf{ex\_fcn}_j \in \mathbf{EF}} \mathsf{exfd}(\mathsf{ex\_fcn}_j) \cdot c_{i,j}.$$

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 4.10. Continuing from Example 4.9, $const_i = 0.5$ 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.

4.3.4.1 Relationship with Covering Problems

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 can not be easily applied. The reason for this is because an optimal adversarial explanation is not necessarily irredundant (see Definition 2.7, page 24). 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

[4] Thus, this describes a class of OAS problems that can be solved exactly in polynomial time.

$m < const' \cdot (k-2)$, which clearly contradicts $const' < \frac{m}{k-2}$. It is clear that a solution to OAS need not be irredundant.

However, we do leverage the idea of an irredundant cover in a different exact approach in this section which may provide a speedup over the exact algorithms of the previous section. The main intuition is that each OAS solution contains an irredundant cover, and if we find such a cover, we can build an optimal adversarial strategy in polynomial time. First, we define a *core* explanation. Before doing so, we recall that L is the set of all points in the space $\mathscr{S}$ that are feasible and that explain at least one observation (*i.e.*, is within the $[\alpha, \beta]$ distance bounds from at least one explanation).

Definition 4.15 (Core Explanation). Given an observation set $\mathscr{O}$ and set L of possible partners, an explanation $\mathscr{E}_{core}$ is a **core explanation** if and only if for any $p_i \in \mathscr{E}_{core}$, there does not exist $p_j \in L$ such that:

1. $\forall o \in \mathscr{O}$ 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 $\mathscr{A}$ is an optimal adversarial strategy, there exists a core explanation $\mathscr{E}_{core} \subseteq \mathscr{A}$.*

Proof. CLAIM 1: For any explanation $\mathscr{E}$, there is an explanation $\mathscr{E}' \subseteq \mathscr{E}$ such that there are no two elements $p, p' \in \mathscr{E}'$ such that $\forall o \in \mathscr{O}$ such that o, p are partners, then o, p' are also partners.
Consider $\mathscr{E}$. If it does not already have the quality of Claim 1, then by simple induction, we can remove elements until the resulting set does.
CLAIM 2: If $\mathscr{A}$ is an optimal adversarial strategy, there is a no $p_j \in L - \mathscr{A}$ such that there exists $p_i \in \mathscr{A}$ where $const_j < const_i$ and $\forall o \in \mathscr{O}$ such that o, p_i are partners, then o, p_j are also partners.
Suppose, by way of contradiction, there is a $p_j \in L - \mathscr{A}$ such that there exists $p_i \in \mathscr{A}$ where $const_j < const_i$ and $\forall o \in \mathscr{O}$ such that o, p_i are partners, then o, p_j are also partners. Consider the set $(\mathscr{A} - \{p_i\} \cup \{p_j\}$. This set is still an explanation and $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, (\mathscr{A} - \{p_i\} \cup p_j) < \mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{A})$—which would be a contradiction as $\mathscr{A}$ is an optimal adversarial strategy.
CLAIM 3: There is an explanation $\mathscr{E} \subseteq \mathscr{A}$ such that condition 1 of Definition 4.15 holds.
Consider the set $\mathscr{E} = \{p_i \in \mathscr{A} \mid \nexists p_j \in \mathscr{A}$ *s.t.* $(const_j < const_i) \wedge$
$(\forall o \in \mathscr{O}$ *s.t.* o, p_i *are partners, then* o, p_j *are also partners*$)\}$. By Claim 1, this set is contained in an OAS. Note that any observation covered by a point in $\mathscr{A} - \mathscr{E}$ is covered by a point in $\mathscr{E}$, so $\mathscr{E}$ is an explanation. Further, by the definition of $\mathscr{E}$ and Claim 2, this set meets condition 1 of Definition 4.15.
CLAIM 4: Set $\mathscr{E}$ from Claim 3 is a core explanation.
By Claim 3, $\mathscr{E}$ is a valid explanation and meets condition 1 of Definition 4.15.

Example 4.11. Continuing from Example 4.10, consider the set $\mathscr{A} = \{p_{34}, p_{38}, p_{57}\}$ (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 $\mathscr{A}$ 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 $\mathscr{A}$.

Suppose we have an oracle that, for a given k, $\mathscr{O}$, and exfd returns a core explanation $\mathscr{E}_{core}$ that is guaranteed to be a subset of the optimal adversarial strategy associated with k, $\mathscr{O}$, and exfd. 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 $\mathscr{E}_{core}$ is an explanation. The algorithm BUILD-STRAT follows from this theorem.

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

Proof. CLAIM 1: For explanation $\mathscr{E}$ and point $p_i \in L - \mathscr{E}$, $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E}) > \mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E} \cup \{p_i\})$ if and only if $const_i < \mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E})$.
If: Suppose $const_i < \mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E})$. Let $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E}) = \frac{a}{b}$. Hence, $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E} \cup \{p_i\}) = \frac{a+const_i}{b+1}$. Suppose, by way of contradiction, $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E}) \leq \mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E} \cup \{p_i\})$. Then, $\frac{a}{b} \leq \frac{a+const_i}{b+1}$. This give us $a \cdot b + a \leq a \cdot b + const_i \cdot b$, which give us $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E}) \leq const_i$—a contradiction.
Only-if: Suppose $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E}) > \mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E} \cup \{p_i\})$. Let $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E}) = \frac{a}{b}$. Hence, $\frac{a}{b} > \frac{a+const_i}{b+1}$, which proves the claim.
CLAIM 2: For explanation $\mathscr{E}$ and points $p_i, p_j \in L - \mathscr{E}$ if $const_i < const_j$, then $\mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E} \cup \{p_i\}) > \mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E} \cup \{p_j\})$.
Straightforward algebra similar to Claim 1.
CLAIM 3: Algorithm BUILD-STRAT returns an optimal adversarial strategy.
We know that $\mathscr{E}_{core}$ must be in the optimal adversarial strategy. Hence, we suppose BWOC that for the remaining elements there is a better set of elements—cardinality between 0 and $k - |\mathscr{E}_{core}|$ such that the expected adversarial detriment is lower. However, this contradicts Claims 1–2.
CLAIM 4: Algorithm BUILD-STRAT runs in time $O(\Delta \cdot |\mathscr{O}| \cdot \log(\Delta \cdot |\mathscr{O}|) + (k - |\mathscr{E}_{core}|)^2)$.
Sorting the set $L - \mathscr{E}_{core}$ can be accomplished in $O(\Delta \cdot |\mathscr{O}| \cdot \log(\Delta \cdot |\mathscr{O}|))$ time. The remainder can be accomplished in $O((k - |\mathscr{E}_{core}|)^2)$ time.

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

Definition 4.16. Given an integer $\delta > 0$, an explanation distribution function exfd, and a reward function **rf**, a core explanation $\mathscr{E}_{core}$ is **δ-core optimal** if and only if:

Algorithm 11 BUILD-STRAT

INPUT: Partner list L, core explanation $\mathscr{E}_{core}$, natural number k, explanation function distribution exfd
OUTPUT: Optimal adversarial strategy $\mathscr{A}$

1. If $|\mathscr{E}_{core}| = k$, return $\mathscr{E}_{core}$
2. Set $\mathscr{A} = \mathscr{E}_{core}$. Let $k' = |\mathscr{E}_{core}|$
3. Sort the set $L - \mathscr{E}_{core}$ 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 $ans = \min_{p_i \in L'}(\{\frac{k' \cdot \mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E}_{core}) + S_i}{k'+i}\})$
7. Let P_{ans} be the P_i associated with ans
8. If $ans \geq \mathsf{EXD}^{\mathbf{rf}}(\mathsf{exfd}, \mathscr{E}_{core})$, return $\mathscr{E}_{core}$, else return $\mathscr{E}_{core} \cup P_{ans}$

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

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 a MILP of reduced size.

Definition 4.17 (Reduced Partner Set). Given observations $\mathscr{O}$ and set of possible partners L, we define the reduced partner set L^{**} as follows:

$$L^{**} = \{p_i \in L |\ \nexists p_j \in L \text{ s.t. } (const_j < const_i) \wedge (\forall o \in \mathscr{O} \text{ s.t. } o, p_i \text{ are partners, } o, p_j \text{ are also partners})\}$$

We define L^* as follows:

$$L^* = \{p_i \in L^{**} |\ \nexists p_j \in L^{**} \text{ s.t. } (const_j = const_i) \wedge (\forall o \in \mathscr{O} \text{ s.t. } o, p_i \text{ are partners, } o, p_j \text{ are also partners})\}$$

Lemma 4.1. *1. If explanation $\mathscr{E}$ is a core explanation, then $\mathscr{E} \subseteq L^{**}$.*
2. If explanation $\mathscr{E}$ is δ-core optimal, then $\mathscr{E} \subseteq L^{**}$.
3. If for some natural number δ, there exists an explanation of size δ, then there exists a δ-core optimal explanation $\mathscr{E}$ such that $\mathscr{E} \subseteq L^*$.

Proof. Proof of Part 1:
Suppose, BWOC, $\mathscr{E}$ is a core explanation and $\mathscr{E} \not\subseteq L^{**}$. Then, there is some element $p_i \in \mathscr{E} \cap (L - L^{**})$. Moreover, by the definition of a core explanation, there does not exist $p_j \in L$ such that $\forall o \in \mathscr{O}$ such that o, p_i are partners, then o, p_j are also partners and $const_j < const_i$. This would also put the element in L^{**} by the definition of that set—which is a contradiction.

Proof of Part 2:
Suppose, BWOC, there exists explanation $\mathscr{E}$ such that for some δ, $\mathscr{E}$ is δ-core optimal and $\mathscr{E} \not\subseteq L^{**}$. Then, there exists some $p_i \in \mathscr{E} \cap (L - L^{**})$. By the definition of L^{**}, there exists a $p_j \in L^{**}$ such that $const_j < const_i$ and $\forall o \in \mathscr{O}$ *s.t.* o, p_i are partners, then o, p_j are also partners. Hence, the set $(\mathscr{E} - \{p_i\}) \cup \{p_j\}$ is also an explanation of size δ and has a lower expected detriment. From the definition of δ-core optimal, this is a contradiction.

Proof of Part 3:
Suppose, BWOC, for some δ such that there is an explanation of this size, there does not exist a δ-core optimal explanation $\mathscr{E}$ such that $\mathscr{E} \subseteq L^*$. By the proof of Part 2, we know that a δ-core optimal explanation must be within L^{**}. Further, by the definition of L^*, for any point $p_i \in L^{**} - L^*$, there exists point $p_j \in L^*$ such that $const_j = const_i$ and $\forall o \in \mathscr{O}$ *s.t.* o, p_i are partners, o, p_j are also partners. Hence, for some δ-core explanation that is not a subset of L^*, any $p_i \in \mathscr{E} \cap (L^{**} - L^*)$ can be replaced by some $p_j \in L^*$, and the resulting set is still an explanation, optimal, and of cardinality δ—a contradiction.

The reduced partner set can be computed in polynomial time. We also note that under the assumption that $|\mathscr{O}| \ll |L|$, which we have found to be true in practice, determining the set L^{**} or L^* can be accomplished faster (in terms of time complexity) than solving even a relaxation of the original MILP.

Proposition 4.12. *Given set L, set L^* and L^{**} can be found in $O(|L|^2 \cdot |\mathscr{O}|^2)$ time.*

Proof. Given sets $L, \mathscr{O}$, set L^{**} can be found with the following steps.

1. For each $p_i \in L$, let $\mathscr{O}_i$ be the subset of $\mathscr{O}$ that can be partnered with p_i
2. For each $p_i \in L$, let $elim_i$ be a boolean variable set to *FALSE*
3. For each $p_i \in L^{**}$, do the following
   a. If not $elim_i$
      i. For each $p_j \in L^{**} - \{p_j\}$, if $\mathscr{O}_j \subseteq \mathscr{O}_i$ and $const_i < const_j$ then set $elim_j = TRUE$
4. Return the set $\{p_i \in L | elim_i = FALSE\}$.

Clearly, the correctness of the above procedure follows directly from the definition of set L^{**}. Further, the complexity of the operation is $O(|L|^2 \cdot |\mathscr{O}|^2)$, as we have two nested loops, each iterating at most $|L|$ times and a comparison where for some p_i, p_j, we examine the elements of $\mathscr{O}_i, \mathscr{O}_j$. To determine the set L^*, we can simply adjust line 3(a)i of the above procedure and change the $<$ to a $\leq$. The correctness again follows from the definition and the time complexity remains the same.

Example 4.12. Let us continue from Example 4.11. Based on pre-processing and the computation of $const_i$, we can easily produce the data of Table 4.1 in polynomial time. Based on this, we obtain a **reduced partner set** $L^* = \{p_{34}, p_{38}, p_{57}\}$.

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

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}

Table 4.1 The set L partitioned by $const_i$ and supported observations.

Lemma 4.2. *Given an optimal adversarial strategy $\mathscr{A}$, there exists some $\delta \leq |\mathscr{A}|$ such that there is a δ-core optimal explanation that is a subset of $\mathscr{A}$ (using the **crf** reward function).*

Proof. By Theorem 4.4, $\mathscr{A}$ must contain a core explanation and by Lemma 4.1, any core explanation must be a subset of L^{**}. Therefore, $\mathscr{A} \cap L^{**}$ is a core explanation. Let $B = \mathscr{A} - (\mathscr{A} \cap L^{**})$ and $\delta = |\mathscr{A} \cap L^{**}|$. Suppose $\mathscr{A} \cap L^{**}$ is not δ-core optimal. Then there is some set Q that is a subset of L^{**}, is disjoint from $\mathscr{A} \cap L^{**}$, and is δ-core optimal. Note that $Q \cap B = \emptyset$ as Q must be a subset of L^{**} and B is not. Hence, since it has a lower expected detriment than $\mathscr{A} \cap L^{**}$ and $|Q \cup B| = |\mathscr{A}|$, the set $Q \cup B$ will have a lower expected detriment than $\mathscr{A}$—which is clearly a contradiction as $\mathscr{A}$ is an OAS.

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 a 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 a 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 4.2 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 4.18 (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 $o_j \in \mathscr{O}$, add constraint $\sum_{p_i \in L^* d(o_j,p_i) \in [\alpha,\beta]} X_i \geq 1$

Example 4.13. Using set L^* from Example 4.12, 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 $L^* = L$. Hence, we can assert that:

Proposition 4.13. *The δ-core constraints require $O(\Delta \cdot |\mathscr{O}|)$ variables and $1 + |\mathscr{O}|$ constraints.*

Proof. Mirrors proposition 4.6.

Proposition 4.14. *Given δ-core constraints:*

1. *Given set δ-core optimal explanation $\mathscr{E}_{core} = \{p_1, \ldots, p_n\}$, if variables $X_1, \ldots, X_n$—corresponding with elements in $\mathscr{A}$—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 $\mathscr{E}_{core}$, then $\mathscr{E}_{core}$ is a δ-core optimal solution.*

Proof. From Lemma 4.1, we know that for any δ such that there exists and explanation of that size, there is a δ-core explanation $\mathscr{E}$ that is a subset of L^*. Hence, the rest of the proof mirrors the proof of Proposition 4.6

We now have all the pieces required to leverage core explanations and reduced partner sets to find an optimal adversarial strategy. By Theorem 4.11, we know that any optimal adversarial strategy must have a core explanation. Further, by Lemma 4.2, 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 Chapter 2; see also[9]), 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 4.14. Continuing from Example 4.13, 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 Chapter 2 - see also [9]) 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 4.13—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.

4.4 Finding a Counter-Adversary Strategy

The preceding section explains how an intelligent adversary can try to keep its "partner" locations associated with a given set of observations hidden. To do this, the adversary uses an explanation function distribution—but unfortunately, the agent may not know what this distribution is. The agent is thus confronted with the problem of creating a strategy to discover the adversary's strategy. When attempting to find an "optimal" strategy for the agent, we first need to understand what benefit each possible strategy brings to the agent. More formally, we use a special case of expected reward (Definition 4.2.2 from Section 4.10) defined as the agent's *expected benefit* below.

Definition 4.19 (Expected Agent Benefit). Given a reward function **rf** and explanation function distribution exfd, the *expected agent benefit* is the function $\mathsf{EXB}^{\mathbf{rf}} : 2^{\mathscr{S}} \times \mathbf{EFD} \rightarrow [0,1]$ defined as follows:

$$\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd}) = \sum_{\mathsf{ex\_fcn} \in \mathbf{EF}} \mathbf{rf}(\mathsf{ex\_fcn}(\mathscr{O}, k), \mathscr{B}) \cdot \mathsf{exfd}(\mathsf{ex\_fcn})$$

Suppose an agent uses an explanation function distribution exfd to estimate how the adversary is assigning probabilities to specific explanation functions. $\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd})$ is computed by looking at each explanation function ex_fcn, identifying the probability of ex_fcn according to the explanation function distribution, and then finding

the reward for the counter-adversary strategy $\mathscr{B}$ used by the agent if ex_fcn were really the explanation function used. The product of the probability of ex_fcn and the reward to the agent of the counter-adversary strategy $\mathscr{B}$ yields an "expected reward" if ex_fcn is the actual explanation function—the sum of such products across all possible explanation functions yields the total expected reward. Thus, this definition is exactly identical to that of expected adversary detriment—except that we now consider the agent instead of the adversary.

Example 4.15. Following from Examples 4.1 and 4.6, suppose police detectives have information (*e.g.*, from a tipster) that the burglar is choosing safe locations according to exfd_{drug}. (Such information could also come from multiple runs of the GREEDY-KSEP-OPT2 algorithm of Chapter 2 (see also [9]). The police detectives wish to consider the set $\mathscr{B} = \{p_{41}, p_{52}\}$. First, they must calculate the reward associated with each explanation function (note that $k = 3, dist = 100$ and $\mathbf{rf} = \mathbf{crf}$).

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

(As an aside, we would like to point out the asymmetry in **crf**—compare these computations with the results of Example 4.7). Hence, $\mathsf{EXB}^{\mathbf{crf}}(\{p_{41}, p_{52}\}, \mathsf{exfd}_{drug}) = 0.634$.

We now define a counter-adversary strategy that the agent can use to nullify the agent's behavior with maximal effectiveness.

Definition 4.20 (Maximal Counter-Adversary Strategy (MCA)). Given a reward function **rf** and explanation function distribution exfd, a **maximal counter-adversary strategy**, $\mathscr{B}$, is a subset of $\mathscr{S}$ such that $\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd})$ is maximized.

Simply put, the maximal counter-adversary strategy is merely any strategy that yields the highest expected benefit to the agent. Note that in theory, there could be zero, one, or many potential maximal counter-adversary strategies.

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 $\mathscr{S}$ and be assured of a solution that maximizes the expected agent benefit. Therefore, for the monotonic case, we include an extra parameter $B \in \{1, \ldots, |\mathscr{S}|\}$ (for "budget") which will serve as a cardinality requirement for $\mathscr{B}$. This cardinality requirement for $\mathscr{B}$ is not necessarily the same as for $\mathscr{A}$ as the agent and adversary may have different sets of resources. Also, we do not require that $\mathscr{B}$ be an explanation. For a discussion of the special case where the solution to the MCA problem is required to be an explanation, see the appendix to [12].

4.4.1 The Complexity of Finding a Maximal Counter-Adversary Strategy

In this section, we develop complexity results for finding a maximal counter-adversary strategy. We start by formally defining the problem of finding a maximal counter-adversary strategy.

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

The result below shows that MCA is NP-hard via a reduction of the GCD problem.

Theorem 4.6. *MCA is NP-hard.*

Proof. Consider an instance of GCD consisting of set of points P, integer b, and integer K. We construct an instance of MCA as follows:
CONSTRUCTION:

- Set $\mathscr{S}$ to be a grid large enough that all points in P are also points in $\mathscr{S}$. We will use M, N to denote the length and width of $\mathscr{S}$.
- $\mathsf{feas}(p) = \mathsf{TRUE}$ if and only if $p \in P$
- $\alpha = 0$, and $\beta = \sqrt{M^2 + N^2}$, $\mathscr{O} = P$, $k = K$, and $B = K$
- Let $\mathbf{rf}(\mathscr{E}_1, \mathscr{E}_2)$ be **crf** where $dist = b$.
- Let functions $\mathsf{ex\_fcn}_1, \ldots, \mathsf{ex\_fcn}_{|P|}$ be explanation functions, with each $\mathsf{ex\_fcn}_i$ corresponding to a unique $p_i \in P$. Let $\mathsf{ex\_fcn}_i(\mathscr{O}, num) = \{p_i\}$ for all $num > 0$. Note that each p_i is an explanation for the set P as it is of cardinality $\leq k$, is feasible, and is guaranteed to be with $[\alpha, \beta]$ from all other points in P as $[\alpha, \beta] = [0, \sqrt{M^2 + N^2}]$
- Let $\mathsf{exfd}(\mathsf{ex\_fcn}_i) = \frac{1}{|P|}$ for all i.

CLAIM 1: $\mathbf{crf}^{dist}(\{p_i\}, \mathscr{B}) = 1$ if and only if there exists $p' \in \mathscr{B}$ such that a disc of radius b (note $b = dist$) centered on p' covers p_i. $\mathbf{crf}^{dist}(\{p_i\}, \mathscr{B}) = 0$ if and only if there does not exist $p' \in \mathscr{B}$ such that a disc of radius b centered on p' covers p_i.
Follows directly from the definition of **crf**.

CLAIM 2: If the expected agent benefit is 1, then for all i, $\mathbf{crf}^{dist}(\{p_i\}, \mathscr{B}) = 1$.
Suppose, by way of contradiction, that the expected agent benefit is 1 and there exists some p_i such that $\mathbf{crf}^{dist}(\{p_i\}, \mathscr{B}) \neq 1$. Then, for a singleton set, $\mathbf{crf}^{dist}(\{p_i\}, \mathscr{B}) = 0$. Hence, for the $\mathsf{ex\_fcn}_i$ associated with p_i, $\mathbf{crf}^{dist}(\mathsf{ex\_fcn}_i(\mathscr{O}), \mathscr{B}) = 0$. So, by the definition of expected agent benefit, it is not possible for the expected agent benefit to be 1—a contradiction.

CLAIM 3: If MCA returns an optimal counter-adversary strategy with an expected expected agent benefit of 1, then GCD must also return "yes."
Suppose, by way of contradiction, MCA returns a strategy with an expected agent benefit of 1 and the corresponding of GCD returns "no." Then there does not exist a K-sized cover for the points in P. However, the set $\mathscr{B}$ is of cardinality K and by Claims 1–2 covers all points in P. Hence, a contradiction.

CLAIM 4: If GCD return "yes" then MCA must return an optima counter-adversary strategy with an expected agent benefit of 1.
Suppose, by way of contradiction, GCD returns "yes" and MCA returns a an optimal strategy with an expected agent benefit < 1. However, by the answer to GCD, there must exist $P' \subseteq P$ of cardinality k that is within distance b of all points in P. Hence, for all i, $\mathbf{crf}^{dist}(\{p_i\}, \mathscr{B}) = 1$ (as $b = dist$). So, the expected agent benefit must also be 1. Hence, a contradiction.

Proof of theorem: Follows directly from Claims 3–4.

The result below follows immediately from the proof of Theorem 4.6 and shows that MCA is NP-hard even if the reward function is monotonic.

Corollary 4.2. *MCA is NP-hard even if the reward function is monotonic.*

Later, in Section 4.4.4, we also show that MCA can encode the NP-hard MAX-K-COVER problem [6] 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 $\mathscr{S}$, feasibility predicate feas, real numbers α, β, set of observations $\mathscr{O}$, natural numbers k, B, reward function **rf**, explanation function distribution exfd, and number $R \in [0,1]$.
OUTPUT: Counter-adversary strategy $\mathscr{B}$ such that $\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd}) \geq R$.

The following result says that as long as the reward function can be evaluated in polynomial time, the **MAX-DEC** decision problem is NP-complete. We note that all the example reward functions we have presented in this chapter are all polynomially computable and hence, the result below applies to them.

Theorem 4.7. ***MCA-DEC*** *is NP-complete, provided the reward function can be evaluated in PTIME.*

Proof. CLAIM 1: Membership in NP.
Given an explanation, $\mathscr{B}$, we can evaluate it reward and if it is an explanation in PTIME.

CLAIM 2: MCA-DEC is NP-hard.
Follows directly from Theorem 4.6

4.4.2 The Complexity of Counting MCA Strategies

Not only is **MCA-DEC** NP-hard, under the same assumptions as above, the result below establishes that counting version of the problem is #P-complete. Moreover, it has no fully polynomial random approximation scheme.

Theorem 4.8. *Counting the number of strategies that provide a "yes" answer to* ***MCA-DEC*** *is #P-complete and has no fully polynomial randomized approximation scheme (FPRAS for short) unless NP=RP.*

Proof. Theorem 4.6 shows a parsimonious reduction from GCD to **MCA**. Hence, we can simply apply Lemma 2.1 and the statement follows.

Theorem 4.8 tells us that MCA may not have a unique solution. Therefore, setting up a mixed strategy across all MCAs to determine the "best response" to the MCA of an agent by an adversary would be an intractable problem. This mirrors the result we presented in the preceding section (Theorem 4.3, page 111).

4.4.3 MCA in the General Case: Exact and Approximate Algorithms

In this section, we first describe an exact algorithm to find a maximal counter-adversary strategy for the agent. In the case of the IED detection example, for instance, a maximal counter-adversary strategy would correspond to the best places for US forces to search for IED weapons caches, given the presence of an adversary who is trying to conceal the locations of his caches. As the results above show, computing MCA is intractable computationally. Therefore, in this chapter, we also develop approximation algorithms that the agent could use to find a maximal counter-adversary strategy in the general case. Note that throughout this section (as well as in Section 4.4.4), we assume that the same pre-processing for **OAS** is used (cf. Section 4.3.2). We 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 $O(\sum_{i=0}^{|\mathscr{S}|} \binom{|L|}{i})$ 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 exfd is fixed, the expected agent benefit is submodular.[5]

[5] Recall that a function $f : 2^X \rightarrow \mathbb{R}$ is *submodular* if and only if for all subsets $X_1 \subseteq X_2 \subseteq X$ and for all $x \notin X_2$, it is the case that $f(X_1 \cup \{x\}) - f(X_1) \geq f(X_2 \cup \{x\}) - f(X_2)$.

Algorithm 12 (MCA-LS)

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

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

Theorem 4.9. *For a fixed $\mathscr{O}, k,$ exfd, the expected agent benefit, $EXB^{\mathbf{rf}}(\mathscr{B}, exfd)$ has the following properties:*

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

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

*(*i.e.*, expected agent benefit is sub-modular 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 basic intuition behind MCA-LS is the following.

1. Start with the set of all possible partners (the set L) and for each possible partners location p_i, find the *difference* of the expected benefit that occurs if we choose partner p_i to put in the agent's strategy as compared to not putting it in the agent's strategy. This value is the "incremental benefit" of adding p_i to the agent's strategy (when the agent's strategy is empty) and is denoted by inc_i. This is what happens in Line 1 of the MCA-LS algorithm.

2. In Line 2 of the MCA-LS algorithm, we sort L in descending order of the inc_i values.
3. In Line 3, we put the p_i with the highest inc_i value into the agent's strategy and remove it from consideration.
4. From Line 4 onwards, we execute a loop. In each iteration, we consider the next possible partner p_i from L—this is always the partner with the highest possible incremental benefit. In Line 4(a), we compute the new value of the agent's strategy if the incremental value of p_i can be directly added to the agent's strategy (the sum of the agent's strategy value plus inc_i is just an estimate). If this value is large enough (Line 4(b)), we add it to the agent's strategy; otherwise we do not add it.
5. The process is repeated several times until we are done.

We will explain what is meant by "large enough" via an example shortly.

The following two propositions leverage Theorem 4.9 and Theorem 3.4 of [5].

Proposition 4.15. *MCA-LS has time complexity of* $O(\frac{1}{\varepsilon} \cdot |L|^3 \cdot F(\mathsf{exfd}) \cdot \lg(|L|))$ *where* $F(\mathsf{exfd})$ *is the time complexity to compute* $\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd})$ *for some set* $\mathscr{B} \subseteq L$.

Proof. We note that one iteration of the algorithm requires $O(|L| \cdot F(\mathsf{exfd}) + |L| \cdot \lg(|L|))$ time. We shall assume that $O(|L| \cdot F(\mathsf{exfd})$ dominates $O(|L| \cdot \lg(|L|))$. By Theorem 3.4 of [5], the number of iterations of the algorithm is bounded by $O(\frac{1}{\varepsilon} \cdot |L|^2 \cdot \lg(|L|)$ where $F(\mathsf{exfd})$, hence the statement follows.

The result below now states that MCA-LS is a $\frac{1}{3}$-approximation algorithm for MCA and thus provides approximation guarantees.

Proposition 4.16. *MCA-LS is an* $(\frac{1}{3} - \frac{\varepsilon}{|L|})$*-approximation algorithm for MCA.*

Proof. By Theorem 4.9, we can be assured that when the "if" statement at line 4c is TRUE, then there are no further elements in $\mathscr{B}^*$ that will afford an incremental increase of $> (1 + \frac{\varepsilon}{|L|^2}) \cdot \mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd})$, even if the last element is not yet reached. Hence, we can apply Theorem 3.4 of [5] and the statement follows.

We now return to our burglary example and use it to illustrate the running of our MCA-LS example.

Example 4.16. Let us consider our running example where law enforcement agents are attempting to find where a burglar resides in the area depicted in Figure 2.4. The agents may guess that there are at most k locations where the burglar might dwell (*e.g.*, home, office, significant other's house) and they know that these locations somehow support the burglaries that he carries out (set of observations $\mathscr{O}$). Furthermore, police assume that the burglar chose his safe locations using some explanation function distribution $\mathsf{exfd}_{burglar}$ (see Example 4.6, page 107).

The law enforcement agents wish to find a maximal counter-adversarial strategy using the **prf** reward function (see page 4.2). They decide to use MCA-LS to find

such a strategy with $\varepsilon = 0.1$. Initially (at line 3), 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 2.5 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 4 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 $\mathscr{B}$ 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 picked elements p_{48}, p_{41}, p_{45} would cause 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 $\mathscr{B} = \{p_{48}, p_{41}, p_{45}\}$ which is optimal.

4.4.4 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 $\mathscr{B}$ (as described in Section 4.4). We first show that expected agent benefit is monotonic when the reward function is.

Corollary 4.3. *For a fixed $\mathscr{O}, k,$ exfd, if the reward function is monotonic, then the expected agent benefit, $EXB^{\mathbf{rf}}(\mathscr{B},$ exfd$)$ is also monotonic.*

Proof. The zero-starting aspect of expected agent benefit follows directly from the definitions of zero-starting and expected agent benefit.

Consider the definition of $\mathsf{EXB}^{\mathbf{rf}}$:

$$\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}\cup\{p\},\mathsf{exfd})-\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B},\mathsf{exfd}) \geq \mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}'\cup\{p\},\mathsf{exfd})-\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}',\mathsf{exfd})$$

As **rf** is monotonic by the statement, and exfd is fixed, $\mathsf{EXB}^{\mathbf{rf}}$ is a positive linear combination of monotonic functions, so the statement follows.

Algorithm 13 (MCA-GREEDY-MONO)

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

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

Thus, when we have a monotonic reward function, the MCA problem reduces to the maximization of a monotonic, normalized[6] submodular function with respect to a uniform matroid[7]—this is a direct consequence of Theorem 4.9 and Corollary 4.3. Therefore, we can leverage the result of [7], to develop the MCA-GREEDY-MONO algorithm below. 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 non-trivial 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).

The basic outline of the MCA-GREEDY-MONO algorithm is as follows. As in the case of MCA-LS, we first assume the agent uses the empty strategy (he hasn't decided what to search as yet). We compute the expected benefit to the agent of using this strategy. We then iteratively add partners to the agent's strategy till the agent's strategy reaches the requisite size B. The key part of the MCA-GREEDY-MONO algorithm is in how we decide which points to add to the agent's strategy. In each iteration of the loop, we consider all remaining members of L and find the one, which, if added to the agent's strategy, gives the highest incremental benefit. Thus member of L then gets added to the agent's strategy, and the procedure is iteratively repeated until the agent's strategy reaches the desired size.

The result below specifies the running time of the MCA-GREEDY-MONO algorithm.

[6] As we include zero-starting in our definition of monotonic.

[7] In our case, the uniform matroid consists of all subsets of L of size B or fewer.

Proposition 4.17. *The complexity of* MCA-GREEDY-MONO *is* $O(B \cdot |L| \cdot F(\mathsf{exfd}))$ *where* $F(\mathsf{exfd})$ *is the time complexity to compute* $\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd})$ *for some set* $\mathscr{B} \subseteq L$ *of size* B.

Proof. The outer loop at line 4 iterates B times, the inner loop at line 4b iterates $O(|L|)$ times, and at each inner loop, at line 4(b)ii, the function $\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd})$ is computed with cost $F(\mathsf{exfd})$. There is an additional $O(|L| \cdot \lg(|L|))$ sorting operation after the inner loop which, under most non-trivial cases, is dominated by the $O(|L| \cdot F(\mathsf{exfd}))$ cost of the loop. The statement follows.

The result below shows us that MCA-GREEDY-MONO provides a 0.63 approximation ration for MCA when the reward function is monotonic.

Corollary 4.4. MCA-GREEDY-MONO *is an* $(\frac{e}{e-1})$*-approximation algorithm for MCA (when the reward function is monotonic).*

Proof. We need a definition of the notion of "incremental increase" in our proof:

Definition 4.21. For a given $p_i \in L$ at some iteration j of the outer loop of GREEDY-MONO (the loop starting at line 4), the *incremental increase*, $inc_i^{(j)}$, is defined as follows:

$$inc_i^{(j)} = \mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}^{(j-1)} \cup \{p_i\}, \mathscr{A}) - \mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}^{(j-1)}, \mathscr{A})$$

Where $\mathscr{B}^{(j-1)}$ is the set of points in L selected by the algorithm after iteration $j-1$.

We now continue with the proof of Corollary 4.4.
CLAIM 1: For any given iteration j of GREEDY-MONO and any $p_i \in L$, $inc_i^{(j)} \geq inc_i^{(j+1)}$
By Definition 4.21, the statement of the proposition is equivalent to the following:

$$\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}^{(j-1)} \cup \{p_i\}, \mathscr{A}) - \mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}^{(j-1)}, \mathscr{A}) \geq \mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}^{(j)} \cup \{p_i\}, \mathscr{A}) - \mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}^{(j)}, \mathscr{A})$$

Obviously, as $\mathscr{B}^{(j-1)} \subseteq \mathscr{B}^{(j)}$, this has to be true by the submodularity of $\mathsf{EXB}^{\mathbf{rf}}$, as proved in Theorem 4.9.

By Claim 1, we can be assured that any point not considered by the inner loop will not have a greater incremental increase than some point already considered in that loop. Hence, our algorithm provides the same result as the greedy algorithm of [7]. We know that the results of [7] state that a greedy algorithm for a non-decreasing, submodularity function F such that $F(\emptyset) = 0$ is a $\frac{e}{e-1}$ approximation algorithm for the associated maximization problem. Theorem 4.9 and Corollary 4.3 show that these properties hold for finding a maximal counter-adversary strategy when the reward function is monotonic. Hence, by [7], the statement follows.

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 $P = NP$. In particular, the following result shows that there is not other polynomial algorithm that can provide an approximation ration which is strictly better than $(\frac{e}{e-1})$ unless $P = NP$. This is done by a reduction of MAX-K-COVER [6].

Theorem 4.10. *MCA-GREEDY-MONO provides the best approximation ratio for MCA (when the reward function is monotonic) unless $P = NP$.*

Proof. The MAX-K-COVER [6] is defined as follows.
INPUT: Set of elements, S and a family of subsets of S, $\mathscr{H} = \{H_1, \ldots, H_{max}\}$, and positive integer K.
OUTPUT: $\leq K$ subsets from $\mathscr{H}$ such that the union of the subsets covers a maximal number of elements in S.
In [6], the author proves that for any $\alpha' < \frac{e}{e-1}$, there is no α'-approximation algorithm for MAX-K-COVER unless $P = NP$. We show that an instance of MAX-K-COVER can be embedded into an instance of MCA where the reward function is monotonic and zero-starting in PTIME. By showing this, we can leverage the result of [6] and Corollary 4.4 to prove the statement. We shall define the reward function $\textbf{\textit{srf}}(\mathscr{A}, \mathscr{B}) = 1$ if and only if $|\mathscr{A} \cap \mathscr{B}| \geq 1$ and $\textbf{\textit{srf}}(\mathscr{A}, \mathscr{B}) = 0$ otherwise. Clearly, this reward function meets all the axioms, is zero-starting, and monotonic. We create a space $\mathscr{S}$ such that the number of points in $\mathscr{S}$ is greater than or equal to $|\mathscr{H}|$. For each subset in $\mathscr{H}$, we create an observation at some point in the space. We shall call this set $\mathscr{O}_{\mathscr{H}}$ and say that o_H is the element of $\mathscr{O}_{\mathscr{H}}$ that corresponds with set $H \in \mathscr{H}$. We set $\mathsf{feas}(p) = \mathsf{true}$ if and only if $p \in \mathscr{O}_{\mathscr{H}}$. We set $\alpha = 0$, β to be equal to the diagonal of the space, and $k = |\mathscr{O}_{\mathscr{H}}|$. Hence, any non-empty subset of $\mathscr{O}_{\mathscr{H}}$ is a valid explanation for $\mathscr{O}$. For each $x \in S$, we define explanation function $\mathsf{ex\_fcn}_x$ such that $\mathsf{ex\_fcn}_x(\mathscr{O}_{\mathscr{H}}, k) = \{o_H \in \mathscr{O}_{\mathscr{H}} | x \in H\}$. We define the explanation function distribution exfd to be a uniform distribution over all $\mathsf{ex\_fcn}_x$ explanation functions. We set the budget $B = K$. Clearly, this construction can be accomplished in PTIME. We note that any solution to this instance of MCA must be subset of $\mathscr{O}_{\mathscr{H}}$, for if it is not, we can get rid of the extra elements and have no change to the expected agent benefit. Hence, each $p \in \mathscr{B}$ will correspond to an element of $\mathscr{H}$, so we shall use the notation p_H to denote a point in the solution that corresponds with some $H \in \mathscr{H}$ (as each $o \in \mathscr{O}_{\mathscr{H}}$ corresponds with some $H \in \mathscr{H}$).

CLAIM 1: Given a solution $\mathscr{B}$ to MCA, the set $\{H \in \mathscr{H} | p_H \in \mathscr{B}\}$ is a solution to MAX-K-COVER.
Clearly, this solution meets the cardinality constraint, as there is exactly one element in $\mathscr{O}_{\mathscr{H}}$ for each element of $\mathscr{H}$ and $\mathscr{B}$ is a subset of $\mathscr{O}_{\mathscr{H}}$. Suppose, by way of contradiction, there is some other subset of $\mathscr{H}$ that covers more elements in S. Let $\mathscr{H}'$ be this solution to MAX-K-COVER and $\mathscr{B}'$ be the subset of $\mathscr{O}_{\mathscr{H}}$ that corresponds with it. We note that for some $x \in S$ in $\mathscr{B}'$, $\textbf{\textit{srf}}(\mathsf{ex\_fcn}_x(\mathscr{O}_{\mathscr{H}}, k), \mathscr{B}') = 1$ if and only if there is some $H \in \mathscr{H}'$ such that $x \in H$ and $\textbf{\textit{srf}}(\mathsf{ex\_fcn}_x(\mathscr{O}_{\mathscr{H}}, k), \mathscr{B}') = 0$ otherwise. Hence, the expected agent benefit is the fraction of elements in S covered by $\mathscr{H}'$. If $\mathscr{H}'$ is the optimal solution to MAX-K-COVER, then $\mathscr{B}'$ must provide a greater expected agent benefit than $\mathscr{B}$, which is clearly a contradiction.

CLAIM 2: Given a solution $\mathscr{H}'$ to MAX-K-COVER, the set $\{o_H \in \mathscr{O}_{\mathscr{H}} | H \in \mathscr{H}'\}$ is a solution to MCA.
Again, that the solution meets the cardinality requirement is trivial (mirrors that part of Claim 1). Suppose, by way of contradiction, there is some set $\mathscr{B}$ that provides a

greater maximum benefit than $\{o_H \in \mathscr{O}_{\mathscr{H}} | H \in \mathscr{H}'\}$. Let $\mathscr{H}'' = \{H \in \mathscr{H} | p_H \in \mathscr{B}\}$. As with Claim 1, the expected agent benefit for $\mathscr{B}$ is equal to the fraction of elements in S covered by $\mathscr{H}''$, which is a contradiction as $\mathscr{H}'$ is an optimal solution to MAX-K-COVER.

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

Example 4.17. Consider the situation from Example 4.16, where the law enforcement agents are attempting to locate the burglar's places of residence. Suppose they want to locate these location, but use the **crf** reward function, which is monotonic (and hence also zero-starting). They use the cardinality requirement $B = 3$ in MCA-GREEDY-MONO. After the first iteration of the loop at Line 4, 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 $\mathscr{B} = \{p_{48}, p_{46}, p_{35}\}$, 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 4.16 which uses **prf** as p_{35} would incur a penalty (which it does not when using **crf** as in this example).

4.5 Implementation and Experiments

In this section, we describe prototype implementations and experiments for solving the OAS and MCA problems. For OAS, we create a MILP for the **crf** case and reduce the number of variables with the techniques we presented in Section 4.3. 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 2.4GHz 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 from Chapter 2 to calculate, for example, the set of all possible partners L and to perform pre-processing (see the discussion in Section 4.3.2, page 111).

Our experiments are based on 21 months of real-world Improvised Explosive Device (IED) attacks in Baghdad[8] (see Chapter 2). 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 US forces. These constitute partner locations. We used data from the International Medical Corps to define feasibility predicates

[8] Attack and cache location data provided by the Institute for the Study of War.

based on ethnic makeup, location of US bases, and geographic features. We overlaid a grid of 100m $\times$ 100m cells—about the size of a standard US 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 GREEDY-KSEP-OPT2 algorithm described in Chapter 2.

4.5.1 OAS Implementation

We now present experimental results for the version of OAS, with the **crf** reward function, based on the constraints in Definition 4.13 and variable-reduction techniques of Section 4.3.4. First, we discuss promising real-world results for the calculation of the reduced partner set L^*, described in Definition 4.15. Then, we show that an optimal adversarial strategy can be computed quite tractably using the methods discussed in Section 4.3.4. 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[9] MILP solver and used 900 lines of Java code.

Reduced Partner Set. As discussed in Section 4.3.2, 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 4.15 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 exfd 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 4.4 provides more detailed accuracy and timing results for this reduction. Most importantly, regardless of parameters chosen, our real-world data is reduced by orders 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 exfd increases. This can be traced back to the strict inequality in Definition 4.17. As we increase the number of nontrivial explanation functions in exfd, 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 4.17 in conjunction with a higher fraction of nonzero $const_i$ constants.

[9] http://www2.isye.gatech.edu/~wcook/qsopt/index.html

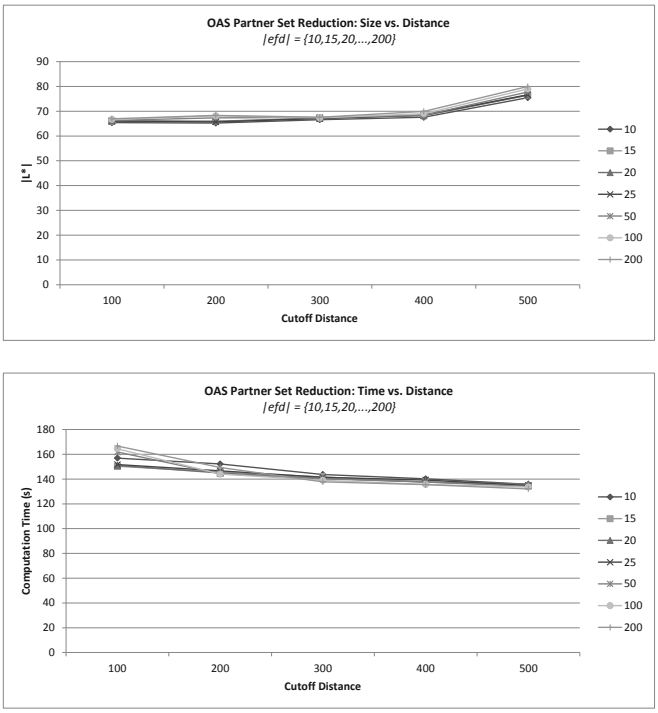

Fig. 4.4 The size of the reduced partner set L^* (top) and the time required to compute this reduction (bottom). Regardless of parameters chosen, we see a 99.6% decrease in possible partners—as well as integer variables in our linear program—in under 3 minutes.

Interestingly, Figure 4.4 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*, resulted in a decrease in the longer computation of L^*. As seen above, 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 4.3.4, then feeding this solution into BUILD-STRAT. Since we do 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 [8], 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.

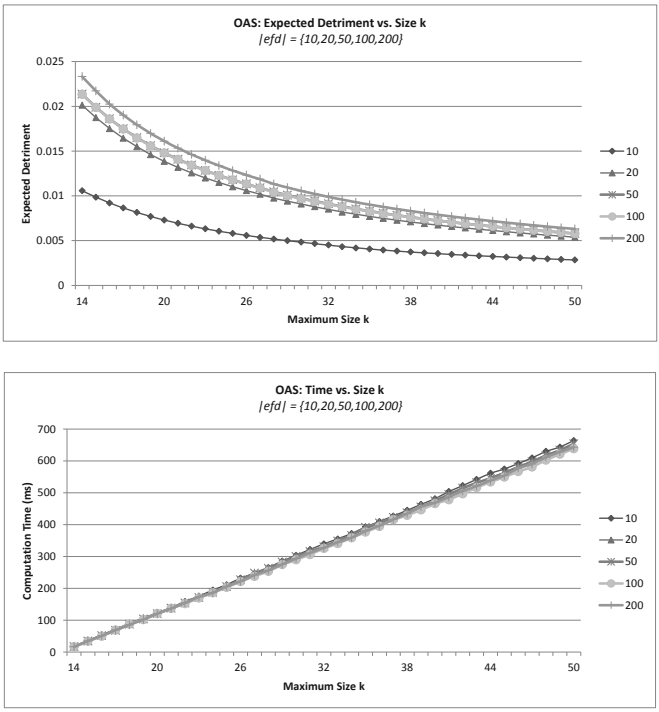

Fig. 4.5 Expected detriment of the optimal adversarial strategy (top) and the runtime of the integer linear program required to produce this strategy in milliseconds (bottom). Note the smooth decrease toward zero detriment as k increases, corresponding with a near-linear increase in total runtime.

Figure 4.5 shows both timing and expected detriment results as the size of the explanation function $|\mathsf{exfd}|$ 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 exfd increases, the expected detriment increases as well. This is a side effect of a larger $|\mathsf{exfd}|$ 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 4.5, 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 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.

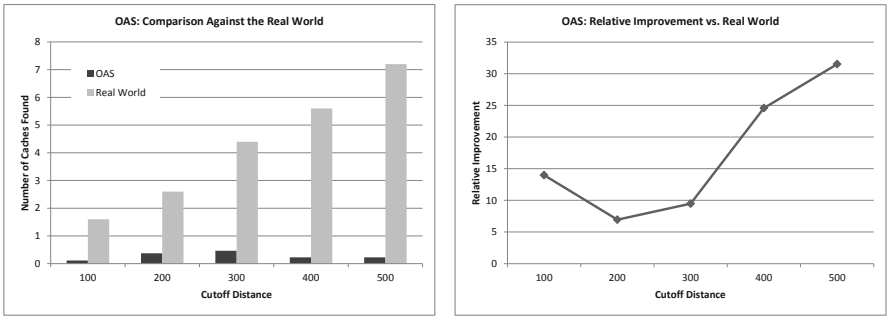

Fig. 4.6 Expected number of caches found when the adversary uses our strategy instead of the current state of the art (left - it is better for the adversary if fewer caches are found). Relative improvement of the OAS strategy versus the current state of the art (right). We assume the reasoning agent is using the Spatio-Cultural Abductive Reasoning Engine (SCARE) to provide information on cache locations.

OAS Performance w.r.t. Real-World Adversarial Strategy. Figure 4.6 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 paper. 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 [8] 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.

4.5.2 MCA Implementation

First, we briefly discuss an implementation of the naive MCA algorithm discussed in section 4.4.3. 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 $k_{\mathscr{B}}$ or more and picking the one which maximizes the expected agent benefit—is inherently intractable. This approach has a complexity $O(\binom{|L|}{k_{\mathscr{B}}})$, and is made worse by the large magnitude of the set L. In our experimental setup, we typically saw $|L| > 20,000$; as such, for even the trivially small $k_{\mathscr{B}} = 3$, we must enumerate and rank over a trillion subsets. For any realistic value of $k_{\mathscr{B}}$, 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 above, 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 non-monotonic submodular maximization approximation algorithm of [5].

Since the expected benefit was maximal for every strategy $\mathscr{B}$ returned, we move to analyzing the particular structure of these strategies. Figure 4.7 shows a relationship between the size $|\mathscr{B}|$, the cutoff distance *dist*, and the cardinality of the expectation function distribution |exfd|. Recall that **prf** penalizes any strategy that does not completely cover its input set of observations; as such, intuitively, we see that MCA-LS 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 $\mathscr{B}$ only roughly twice the size as minimum-sized explanation found by GREEDY-KSEP-OPT2 (which, based on the analysis of Chapter 2 and [9], 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 [9], $|\mathscr{E}|$. 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 4.8, we see results comparing overall computation time to both the distance *dist* and the cardinality of exfd. For more strict (*i.e.*, smaller) values of

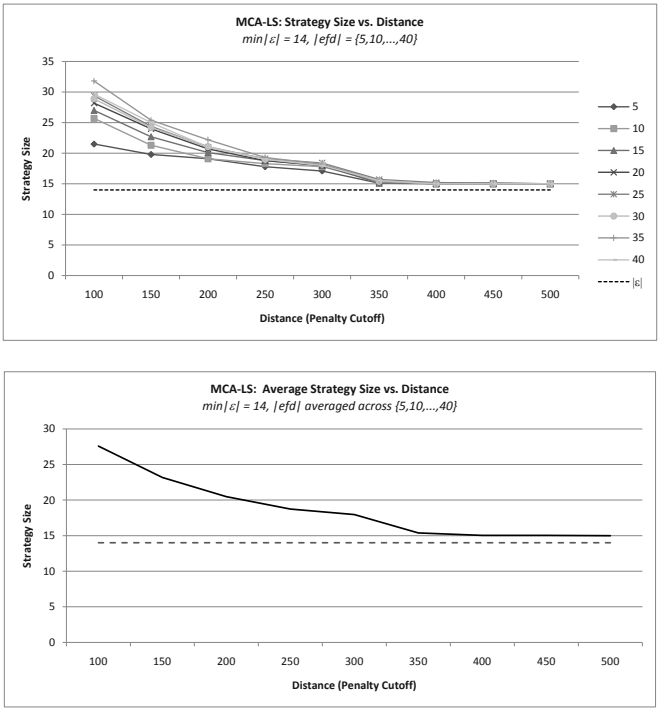

Fig. 4.7 The average size of the strategy recommended by MCA-LS decreases as the distance cutoff increases. For these experiments, the minimum cardinality for a given explanation $\mathscr{E}$ considered is exfd was 14, which gives us a natural lower bound on the expected size of a strategy. Note the convergence to this bound at cutoff distances at and above 300 meters.

dist, the algorithm—which, under **prf**, is penalized for all uncovered observations across exfd—must spend more time forming a strategy $\mathscr{B}$ that minimizes penalization. Similarly, as the distance constraint is loosened, the algorithm completes more quickly. Finally, an increase in $|\mathsf{exfd}|$ results in higher computational cost; as explained in Proposition 4.15, this is due to an increase in $F(\mathsf{exfd})$, the time complexity of computing $\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd})$. Comparing these results to Figure 4.7, we see that the runtime of MCA-LS is correlated to the size of the returned strategy $\mathscr{B}$.

MCA-GREEDY-MONO. As discussed in Section 4.4.4, 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 reward function **crf**. Recall that a trivial solution to MCA given a monotonic reward function is $\mathscr{B} = L$; as such, MCA-GREEDY-MONO uses a budget B to limit the maximum size $|\mathscr{B}| \ll |L|$. We varied this parameter $B \in \{1, \ldots, 28\}$.

Figure 4.9 shows the expected benefit $\mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd})$ increase as the maximum allowed $|\mathscr{B}|$ increases. In general, the expected benefit of $\mathscr{B}$ increases as the dis-

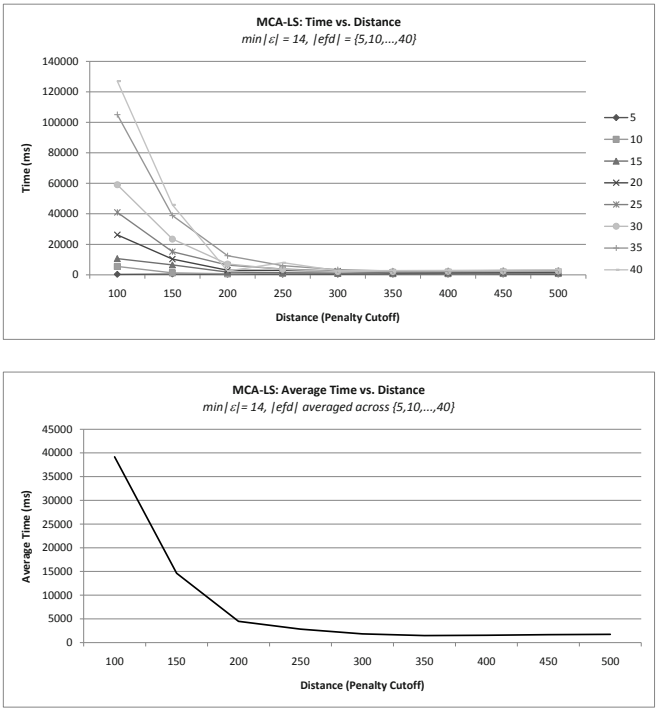

Fig. 4.8 The runtime of MCA-LS decreases as the penalizing cutoff distance is relaxed. Note the relation to Figure 4.7; intuitively, larger recommended strategies tend to take longer to compute.

tance 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 $|\mathsf{exfd}|$ increases, the expected benefit of $\mathscr{B}$ converges more slowly to 1.0. This is intuitive, as a wider spread of possible partner positions will, in general, require a larger $|\mathscr{B}|$ to provide coverage.

Figure 4.10 shows that the runtime of MCA-GREEDY-MONO increases as predicted by Proposition 4.15. In detail, as we linearly increase budget B, we also linearly increase the runtime of our $F(\mathsf{exfd}) = \mathsf{EXB}^{\mathbf{rf}}(\mathscr{B}, \mathsf{exfd})$. In turn, the overall runtime $O(B \cdot |L| \cdot F(\mathsf{exfd}))$ increases quadratically in B, for our specific reward function. Finally, note the increase in runtime as we increase $|\mathsf{exfd}| = 10$ to $|\mathsf{exfd}| = 100$. Theoretically, this increases $F(\mathsf{exfd})$ linearly; in fact, we see almost exactly a ten-fold increase in runtime given a ten-fold increase in $|\mathsf{exfd}|$.

MCA Algorithms and SCARE. We now compare the efficacy of the two MCA algorithms proposed in this paper to SCARE [8] 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 $|\mathsf{exfd}| = 100$ static for the MCA-based algorithms. Figure 4.11 plots the average number of predicted points within 500

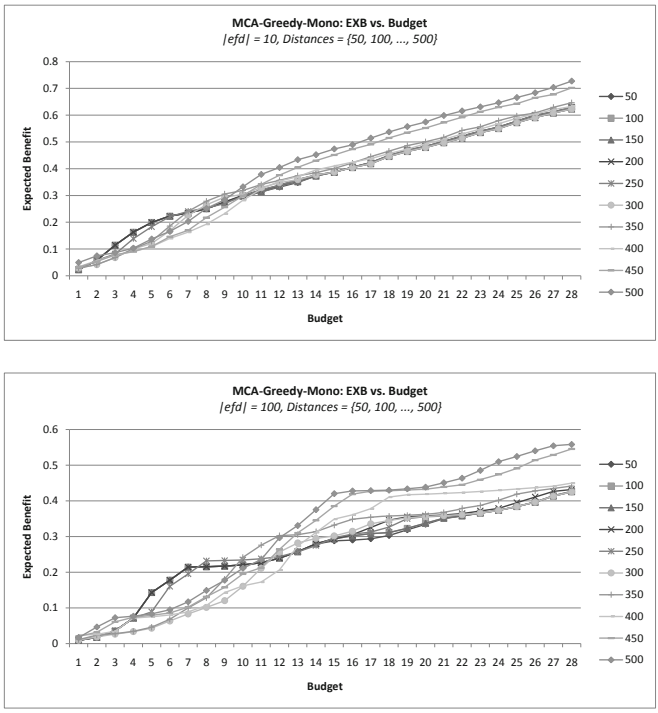

Fig. 4.9 Expected benefit of the strategy returned by MCA-GREEDY-MONO as the budget increases, with $|\mathsf{exfd}| = 10$ (top) and $|\mathsf{exfd}| = 100$ (bottom). Note the decrease in expected benefit due to the increase in $|\mathsf{exfd}|$. Similarly, note the increase in expected benefit given a larger cutoff distance.

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 a low penalizing cutoff distances, and steadily converges to SCARE's baseline as the penalizing distance increases (as expected). As shown earlier in Figure 4.7, 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

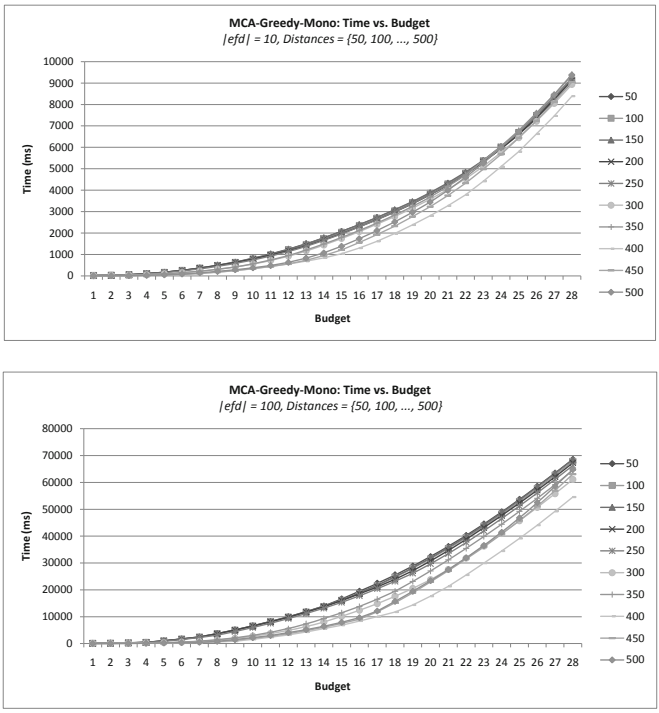

Fig. 4.10 Runtime of MCA-GREEDY-MONO as the budget increases, with $|\mathsf{exfd}| = 10$ (top) and $|\mathsf{exfd}| = 100$ (bottom). Note the increase in runtime due to the extra determinism of a larger exfd.

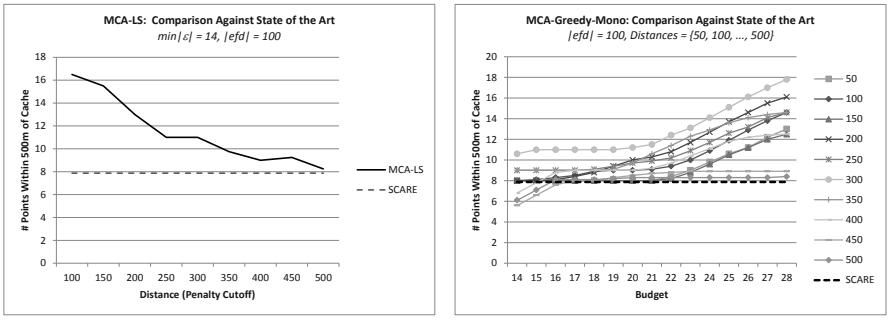

Fig. 4.11 Expected number of points within 500 meters of an actual cache returned by MCA-LS (left) and MCA-GREEDY-MONO (right) compared against an agent using SCARE (higher is better). Note that the SCARE software always returns an explanation of size 14, while both MCA algorithms benefit from the ability to adjust this explanation size.

meters; for these settings, the algorithm returns an average strategy size of 18, with 11 points (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 2 of its resources under the strategy provided by SCARE.

4.6 Conclusion

In this chapter, we recognized that adversaries are not going to sit by passively while the agent adapts to their behavior. Instead, the adversary is going to adapt its tactics in response to what the agent does as well. In our IED weapons cache detection application, for example, US forces observe what the adversary does, and use that information (using the techniques defined in Chapters 2 and 3) to determine which regions or locations to search for IED weapons caches. However, the work in those chapters assume that the adversary does not change his tactics, based on the searches that US forces carry out (that are very easily visible to them).

This chapter recognizes this reality and describes a mathematical framework, based on game theory, to determine how the adversary might adapt to his observations of the agent. We define this problem via notions of reward functions, leading to the definition of expected adversarial detriment (for the adversary). The adversary then tries to find a strategy that minimizes the expected adversarial detriment. For instance, in the case of the IED weapons cache location application, the adversary wants to find locations that minimize his expected adversarial detriment and, intuitively, minimizes the probability that his weapons cache locations will be found. We study the complexity of this problem and develop both exact and approximation algorithms to solve them.

The good news, for the agent, is that the adversary must move first. In the IED weapons cache detection application, the adversary must first decide where to put his weapons caches. The goal of the agent is to come up with a strategy (which corresponds to locations to search for weapons caches in the IED weapons cache detection application) which uncovers a maximal set of IED weapons caches. We formalize this problem in terms of expected benefit to the agent and find a strategy that maximizes the agent's expected benefit.

Our experiments to evaluate both the OAS algorithm to find an optimal adversary strategy and the MCA-Greedy-MONO algorithm to find the maximal counter-adversary strategy have been tested on real-world data involving IED attacks on US and Coalition forces in Iraq and have proven to be highly accurate.

References

1. Leyton-Brown, K., Shoham, Y. 2008. *Essentials of Game Theory: A Concise, Multidisciplinary Introduction*. Morgan and Claypool Publishers.
2. Johnson, D. 1982. The NP-Completeness Column: An Ongoing Guide. *Journal of Algorithms 3,* 2, 182–195.
3. Charnes, A., Cooper, W. 1962. Programming with linear fractional functionals. *Naval Research Logistics Quarterly 9,* 3, 163–297.
4. Karmarkar, N. 1984. A new polynomial-time algorithm for linear programming. *Combinatorica 4,* 4, 373–395.
5. Feige, U., Mirrokni, V. S., Vondrak, J. 2007. Maximizing non-monotone submodular functions. In *FOCS '07: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society, Washington, DC, USA, 461–471.
6. Feige, U. 1998. A threshold of ln n for approximating set cover. *J. ACM 45,* 4, 634–652.
7. Nemhauser, G., Wolsey, L., Fisher, M. 1978. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming 14*, 265–294.
8. Shakarian, P., Subrahmanian, V.S., Sapino, M.L. SCARE: A Case Study with Baghdad, Proc. 2009 Intl. Conf. on Computational Cultural Dynamics (eds. D. Nau, A. Mannes), Dec. 2009, AAAI Press.
9. Shakarian, P., Subrahmanian, V.S., Sapino, M.L. 2012. GAPS: Geospatial Abduction Problems, ACM Transactions on Intelligent Systems and Technology (TIST), 3, 1, to appear.
10. Shakarian, P., Subrahmanian, V.S. Region-based Geospatial Abduction with Counter-IED Applications, accepted for publication in: Wiil, U.K. (ed.).Counterterrorism and Open Source Intelligence, Springer Verlag Lecture Notes on Social Networks, to appear, 2011.
11. Shakarian, P., Nagel, M., Schuetzle, B., Subrahmanian, V.S. 2011. Abductive Inference for Combat: Using SCARE-S2 to Find High-Value Targets in Afghanistan, in Proc. 2011 Intl. Conf. on Innovative Applications of Artificial Intelligence, Aug. 2011, AAAI Press.
12. Shakarian, P., Dickerson, J., Subrahmanian, V.S. 2012. Adversarial Geospatial Abduction Problems, ACM Transactions on Intelligent Systems and Technology (TIST), to appear.