No-Regret Learning in Stackelberg Security Games with an Unknown, Bounded Set of Attacker Types

Sabil Ahmad* 23b1057@iitb.ac.in
Ishita* 23b0921@iitb.ac.in
Aditi Singh* 23b1053@iitb.ac.in
Swaprava Nath swaprava@iitb.ac.in
Indian Institute of Technology Bombay

Abstract

In a repeated Stackelberg security game, a defender commits in every round to a randomized allocation of its resources, and an attacker, drawn from a set of types, best-responds. Existing no-regret guarantees assume that the defender knows this set of types in advance. We drop this assumption: the set of types is unknown, and its size is at most $K_{\max}$, an upper bound that the defender is not told. We show that what can be learned then depends on what the defender observes. If it observes the type of each attacker, an algorithm that adds types to its list as they appear achieves regret $\tilde{O}(nK_{\max}\sqrt{T})$ over T rounds with n targets, the same $\sqrt{T}$ rate as with known types, at an extra factor of only $\sqrt{K_{\max}}$ and without ever using $K_{\max}$. If it observes only the attacked target and types come from an infinite universe, no algorithm has sublinear regret: in a fixed three-target game every algorithm suffers regret at least $T-2$, even against a single attacker type, because it cannot tell which type it faces. If, in addition, it is told every B rounds which types have appeared so far, sublinear regret returns, at $\tilde{O}(nK_{\max}^{4/3}T^{2/3} + K_{\max}B)$. The first term is the rate of the partial-information learner we use, and we show that the additive cost of the delay cannot be avoided.

1 Introduction

In a Stackelberg security game (SSG), a defender allocates a limited set of resources to protect n targets against a strategic attacker who observes the defender's randomized deployment and then attacks the target it likes best [46]. SSGs underlie several deployed systems: patrols of the US Coast Guard in ports [45], the scheduling of Federal Air Marshals on flights [47], checkpoints and canine patrols at Los Angeles International Airport [40], and ranger patrols against poaching in national parks [16, 50]. In each of them the defender commits to its deployment before any attack, and the attacker's payoffs encode which targets it values and how much it fears being caught.

The classical analysis assumes that the defender knows the attacker's payoffs; the optimal commitment is then the solution of a linear program [14]. In practice the defender faces many attackers whose payoffs it knows only roughly. Balcan et al. [3] address this through online learning. The attacker of each round is one of a finite set K of *types*, chosen by an adversary, and the defender seeks low *regret*, that is, a total payoff close to that of the best fixed deployment in

*Equal contributions.

hindsight. If the defender learns the type after each round, regret $\tilde{O}(n\sqrt{|K|T})$ is achievable; if it sees only which target was attacked, $\tilde{O}(n|K|T^{2/3})$ is, and Bernasconi et al. [6] later improved this to $\tilde{O}(\text{poly}(|K|)\sqrt{T})$. If the number of types is unrestricted, regret is linear in the worst case.

These guarantees require the defender to know the set K in advance, with the payoffs of every type in it. This is rarely the case. A new group of poachers may enter a park and target different species; a new smuggling network may favor different routes through a port; in cyber defense, new threat actors with new objectives appear routinely. What the defender can reasonably assume is that only a limited number of such adversaries are active, without knowing who they are or exactly how many. The information the defender receives also varies across these settings. After an incident it may identify who attacked, as when forensic evidence or captured equipment reveals the poacher group. It may learn only which target was hit, as when an intrusion is detected but not attributed. Or it may receive periodic intelligence reports listing the groups known to be active, without saying which of them carried out which attack. This paper asks:

Can the defender achieve sublinear regret when the set of attacker types is unknown and only its size is bounded, and how does the answer depend on what the defender observes?

The question lies between the two regimes of Balcan et al. [3]: a known set of types, where sublinear regret is possible, and an unrestricted number of types, where it is not.

1.1 Our contributions

We answer the question for each of the three kinds of feedback above. The answers share one message: a bound on the *number* of types helps only if the defender can learn *which* types it faces. Throughout, $K_{\max}$ is an upper bound on the number $|K|$ of types, and the defender is told neither the set K nor $K_{\max}$.

- *Observing the type: no-regret at the known-type rate* (Section 3). Our algorithm Grow-Catalog (Algorithm 1) keeps a list of the types seen so far and runs the known-type learner of Balcan et al. [3] on it, restarting the learner whenever a new type appears. Theorem 1 shows that its regret is $O\big(nK_{\max}\sqrt{T\log(nK_{\max})}\big)$. This is the first no-regret guarantee when an adversary chooses the types from an unknown set. It has the same $\sqrt{T}$ dependence as the known-type bound, and the price of not knowing the types is a factor $\sqrt{K_{\max}}$. The algorithm never uses $K_{\max}$, and its regret scales with the number of types that actually appear (Corollary 1). The $\sqrt{T}$ rate cannot be improved: as in any expert problem, even with two known types and full information, regret $\Omega(\sqrt{T})$ is unavoidable (Proposition 2).

- *Observing only the target: impossibility, even with one type* (Section 4). Theorem 2 shows that when the defender sees only the attacked target and the types come from an infinite universe, every algorithm, randomized or not, suffers regret at least $T-2$ on a fixed three-target game, even when a single type attacks in every round. The linear lower bound of Balcan et al. [3] needs a number of types that grows with T, even when the type is observed; ours needs one type, fixed before the game starts. The obstacle is therefore not how many types there are, but that the defender cannot identify the one it faces. The types form a continuum, and each coverage vector catches at most one of them, so the defender must find a good deployment by exact-match search. Each round reveals at most one bit about the type, and the counting argument of Lemma 5 shows that this is too little to win such a search in time. The result also differs from the impossibility of Bollini et al. [10], where the follower's type is drawn from a distribution; here the attacker is one fixed type.

- *Periodic lists of types: sublinear regret again* (Section 5). Suppose the defender sees only the attacked target, but every B rounds it is told the set of types that have attacked so far, not when or how often. Our algorithm Block-Grow-Catalog (Algorithm 2) restores sublinear regret, $O\big(nK_{\max}^{4/3}T^{2/3}\log^{1/3}(nK_{\max})+K_{\max}B\big)$ (Theorem 4). The key tool is a partial-information learner with an anytime guarantee (Lemma 7), built with the doubling trick [1, 11]; we need it because the defender does not know how long each epoch will last. The $T^{2/3}$ rate is that of this learner, built on the partial-information learner of Balcan et al. [3], and our analysis uses it as a black box; any anytime learner with a better rate for known types gives a correspondingly better bound. The additive term is the cost of the delay, and it is tight: Theorem 5 shows that every algorithm must pay $K_{\max}(B-2)$. Reports are needed rarely: $O(T^{1/3})$ of them over the whole horizon leave the rate unchanged (Corollary 3).

Experiments in Appendix C illustrate the behavior predicted by the analysis. Table 1 places these results among the known ones.

Set of types	Regret	Result	What is new
The defender observes the type of the attacker			
known	$\tilde{O}(n\sqrt{\lvert K\rvert T})$	Balcan et al. [3, Thm. 5.1]	
known, $\lvert K\rvert$ unrestricted	$\Omega(T)$	Balcan et al. [3, Thm. 7.1]	
known, $\lvert K\rvert = 2$	$\Omega(\sqrt{T})$	Proposition 2	standard expert lower bound, here in an SSG
unknown, $\lvert K\rvert \le K_{\max}$	$\tilde{O}(nK_{\max}\sqrt{T})$	Theorem 1 and Corollary 1	first bound for adversarial unknown types; $K_{\max}$ not used; adapts to the types that appear
The defender observes only the attacked target			
known	$\tilde{O}(n\lvert K\rvert T^{2/3})$	Balcan et al. [3, Thm. 6.1]	
known	$\tilde{O}(\mathrm{poly}(\lvert K\rvert)\sqrt{T})$	Bernasconi et al. [6]	
unknown, even $\lvert K\rvert = 1$	$\Omega(T)$	Theorem 2	one fixed type suffices; Balcan et al. [3, Thm. 7.1] need $\lvert K\rvert$ growing with T even when types are observed
The defender observes the attacked target, and every B rounds the types seen so far			
unknown, $\lvert K\rvert \le K_{\max}$	$\tilde{O}(nK_{\max}^{4/3}T^{2/3}+K_{\max}B)$	Theorem 4	sublinear for $B = o(T)$; works with any anytime learner (Lemma 7)
unknown, $\lvert K\rvert \le K_{\max}$	$\Omega(K_{\max}B)$	Theorem 5	the delay cost is tight; $O(T^{1/3})$ reports suffice

Table 1: Regret in the repeated SSG, grouped by what the defender observes. n: targets, T: rounds, $K_{\max}$: upper bound on $\lvert K\rvert$, not known to the defender; $\tilde{O}$ hides logarithmic factors; Ω bounds hold for every algorithm. Shaded rows are the results of this paper.

1.2 Related work

Computing the optimal commitment against a single known attacker is a linear program [14], and compact formulations scale it to large deployments [25, 30]. Uncertainty about the attacker has been handled by Bayesian SSGs, which optimize against a known prior over types [26, 37], by robust formulations over payoff intervals [27, 41], and by behavioral models fitted to past attacks [20, 36].

Each of these assumes a known population of attackers, given as a prior, an uncertainty set or a parametric model.

A second line learns the optimal commitment against one fixed but unknown attacker from its best responses [9, 31, 38], or from revealed preferences [15, 43]. The number of queries these methods need depends on the geometry of the attacker's best-response regions. This is consistent with our impossibility result, whose adversary chooses a type with an exponentially thin region (Section 4.4).

Online learning against a sequence of attackers was introduced by Balcan et al. [3], whose setting we build on: an oblivious adversary picks each round's type from a finite set that the defender knows, together with every type's payoffs. They obtain $\tilde{O}(n\sqrt{|K|T})$ regret under full information and $\tilde{O}(n|K|T^{2/3})$ under partial information. Bernasconi et al. [6] improve the partial-information rate to $\tilde{O}(\mathrm{poly}(|K|)\sqrt{T})$ through a reduction to online linear optimization, and Harris et al. [22] and Balcan et al. [4] extend the setting to side information observed by both players, reaching $\tilde{O}(\sqrt{T})$ under bandit feedback. Chen et al. [13] allow several attackers and varying attack intensities in each round. Other work studies attackers that are not myopic [21], or combines bandit algorithms with equilibrium computation when the defender has no prior knowledge [29, 34, 49]. Every upper bound in this line requires the set of types, and their payoffs, to be known before the first round; our results remove that requirement.

The work closest to ours is that of Bollini et al. [10], who study Bayesian Stackelberg games in which the leader knows neither the follower's payoffs nor the distribution of its types. They show that no-regret learning is impossible when the leader observes only the follower's action, and give an $\tilde{O}(\sqrt{T})$ algorithm when it also observes the type. Personnat et al. [39] study several followers whose utilities are known but whose type distribution is not. Both papers draw types i.i.d. from a fixed distribution, whereas our types form an adversarial sequence and our benchmark is the best fixed strategy on that sequence. Our impossibility result holds in a security game with a single type, where theirs uses three types in a general Stackelberg game; theirs, in turn, needs only polynomial precision in the payoffs, where ours needs precision exponential in T. Neither paper considers revelation of types after a delay.

Our algorithms run an expert algorithm [11, 12, 18, 33] over a set of experts that grows when a new type appears. Expert sets that change over time have been studied as sleeping experts [19, 28], growing sets [35] and dying experts [44]. There the learner is told when the set changes, and the losses of existing experts do not. In our setting a new type changes the loss of every existing expert, since an expert is a coverage vector and its payoff depends on how the new type responds to it, so we restart the learner instead. Under block revelation the defender learns of a new type only after a delay, which relates our additive term to online learning with delayed feedback [23, 42, 48]; there the delay applies to the losses, here to the identity of the attacker.

2 Preliminaries

A defender protects n targets with limited resources against a sequence of attackers over T rounds. In each round the defender commits to a randomized allocation of its resources; the attacker of that round sees the allocation, but not its random outcome, and attacks the target it likes best; both players then receive payoffs that depend on whether the attacked target was covered. Attackers differ in their payoffs, and each attacker is one of finitely many *types*. The first four paragraphs below define this model, and the last recalls how the defender learns when the set of types is known.

Game primitives. There are n targets $N = \{1, \dots, n\}$. The defender has R resources and a collection $\mathcal{D} \subseteq 2^N$ of *schedules*, each a set of targets that one resource can protect at the same time. An *assignment function* $A : R \to 2^{\mathcal{D}}$ lists the schedules each resource may take. A *pure strategy* assigns every resource to one of its allowed schedules; let m be the number of pure strategies. A *mixed strategy* is a distribution $\mathbf{s} \in \Delta_m$ over pure strategies. Its *coverage vector* is $\mathbf{p} = M\mathbf{s} \in [0,1]^n$, where $M \in \{0,1\}^{n \times m}$ has $M_{ij} = 1$ if and only if the j-th pure strategy covers target i, so p_i is the probability that target i is covered. Let $\mathcal{P}$ be the set of coverage vectors of all mixed strategies. It is the convex hull of the coverage vectors of the pure strategies, and hence a convex polytope. For example, with $n = 3$ targets and one guard who can stand at any single target, the pure strategies are the three targets, and $\mathcal{P} = \{\mathbf{p} \in [0,1]^3 : p_1 + p_2 + p_3 = 1\}$. Payoffs and the attacker's behavior depend on a mixed strategy only through its coverage vector, so from now on the defender is taken to choose $\mathbf{p} \in \mathcal{P}$ directly.

Defender payoffs. For each target i, the defender receives $u_d^c(i) \in [-1,1]$ if i is attacked and covered, and $u_d^u(i) \in [-1,1]$ if i is attacked and uncovered. Averaging over the defender's own randomization, under which i is covered with probability p_i, the defender's utility when target i is attacked is

$$U_d(i, \mathbf{p}) \;=\; u_d^c(i)\, p_i \;+\; u_d^u(i)\,(1 - p_i), \tag{1}$$

which is linear in $\mathbf{p}$ for each fixed target i.

Attacker types and best response. An attacker *type* α is a pair $(u_\alpha^c, u_\alpha^u) \in [-1,1]^n \times [-1,1]^n$ of per-target payoffs, for a covered and an uncovered attack respectively. Its utility for attacking target i is $U_\alpha(i, \mathbf{p}) = u_\alpha^c(i)\, p_i + u_\alpha^u(i)\,(1 - p_i)$. Seeing $\mathbf{p}$, type α attacks a target in $\arg\max_{i \in N} U_\alpha(i, \mathbf{p})$, breaking ties by a fixed order over the targets, which may depend on the type; we write $b_\alpha(\mathbf{p})$ for the target it attacks. The map b_α is piecewise constant, so the defender's payoff $U_d(b_\alpha(\mathbf{p}), \mathbf{p})$ jumps where the attacker switches targets. For example, with one guard and two targets, an attacker who values both targets equally attacks whichever is less covered, and the defender's payoff jumps as p_1 crosses $1/2$. These jumps are why standard methods for online linear optimization do not apply directly, and the last paragraph of this section explains how they are handled.

Repeated game and regret. There is a set K of attacker types; throughout, $|K|$ denotes its size. An *oblivious* adversary fixes a sequence $\mathbf{a} = (a_1, \dots, a_T) \in K^T$ before the game starts, and the defender does not know it. In round $t = 1, \dots, T$ the defender commits to $\mathbf{p}_t \in \mathcal{P}$, the attacker a_t attacks $b_{a_t}(\mathbf{p}_t)$, and the defender receives $U_d(b_{a_t}(\mathbf{p}_t), \mathbf{p}_t)$. Under *full-information feedback*, at the end of round t the defender learns the type a_t, and the first time a type is seen its payoffs (u_α^c, u_α^u) are revealed as well. The benchmark is the best fixed coverage vector in hindsight,[1]

$$\mathbf{p}^* \;=\; \arg\max_{\mathbf{p} \in \mathcal{P}} \; \sum_{t=1}^{T} U_d\big(b_{a_t}(\mathbf{p}), \mathbf{p}\big), \tag{2}$$

and the regret of a defender that plays $\mathbf{p}_1, \dots, \mathbf{p}_T$ is

$$\mathrm{Regret}(T) \;=\; \sum_{t=1}^{T} U_d(b_{a_t}(\mathbf{p}^*), \mathbf{p}^*) \;-\; \mathbb{E}\Big[\sum_{t=1}^{T} U_d(b_{a_t}(\mathbf{p}_t), \mathbf{p}_t)\Big], \tag{3}$$

[1] Because of tie-breaking the maximum in Equation (2) may not be attained; as in Balcan et al. [3, footnote 1], $\mathbf{p}^*$ then denotes a coverage vector whose total payoff is arbitrarily close to the supremum.

where the expectation is over the defender's internal randomization. Up to this point the model is that of Balcan et al. [3, Sec. 2], with the same notation; the difference, studied in the rest of the paper, is that the defender does not know K.

Learning with a known type set. Suppose for the moment that the defender knows a finite set Θ of types containing every attacker. Balcan et al. [3] solve this case, and our algorithms use their solution as a subroutine, with Θ the set of types seen so far. It combines two ingredients.

An expert algorithm. An *expert* is a fixed action that the learner may choose; here it will be a fixed coverage vector. An *expert algorithm* keeps a weight on each of finitely many experts, plays an expert drawn with probability proportional to its weight, then sees the loss of every expert and lowers the weights of those that did badly. Polynomial Weights [12], for instance, multiplies each weight by $1 - \eta \cdot \text{loss}$ for a learning rate η.

Proposition 1 (Balcan et al. [3, Proposition 1], after Blum and Mansour [8]; anytime form by the doubling trick [11, Sec. 2.3])**.** *Consider a learner that in each round chooses a distribution over a finite set $\mathcal{M}$ of experts, after which every expert receives a loss in $[-\kappa, \kappa]$. There is an expert algorithm whose expected regret after T' rounds, against the best fixed expert in hindsight, is $O(\sqrt{T'\kappa \log |\mathcal{M}|})$ for every T', and which is not told T'.*

A finite set of experts. Proposition 1 cannot use every coverage vector as an expert, since $\mathcal{P}$ is infinite. For a *response profile* $\sigma : \Theta \to N$, which names a target for every type, let

$$\mathcal{P}_\sigma(\Theta) \;=\; \{\mathbf{p} \in \mathcal{P} :\; b_\alpha(\mathbf{p}) = \sigma(\alpha) \text{ for every } \alpha \in \Theta\} \tag{4}$$

be the region where every type attacks the target σ names. It is cut out of $\mathcal{P}$ by linear inequalities, so it is a convex polytope, and the nonempty regions partition $\mathcal{P}$. Inside one region every attacker's target is fixed, so by Equation (1) the defender's total payoff is linear in $\mathbf{p}$ and is maximized at a corner. The corners of all regions would therefore be enough experts, except that tie-breaking makes some defining inequalities strict: a corner may lie just outside its region, where some attacker already switches target. Each such corner is replaced by a point of its region within ℓ_1-distance ϵ of it. The resulting finite set $\mathcal{E}(\Theta; \epsilon)$ of *ϵ-approximate extreme points* loses little and is small.

Lemma 1 (Balcan et al. [3, Lemma 4.3])**.** *For every finite Θ, every $\epsilon > 0$ and every sequence $(a'_1, \ldots, a'_{T'}) \in \Theta^{T'}$,*

$$\max_{\mathbf{p} \in \mathcal{E}(\Theta;\epsilon)} \sum_{t=1}^{T'} U_d(b_{a'_t}(\mathbf{p}), \mathbf{p}) \;\geq\; \sup_{\mathbf{p} \in \mathcal{P}} \sum_{t=1}^{T'} U_d(b_{a'_t}(\mathbf{p}), \mathbf{p}) - 2\epsilon\, T'.$$

Lemma 2 (Balcan et al. [3, Lemma 4.4])**.** *For every finite Θ and every $\epsilon > 0$, $\log |\mathcal{E}(\Theta; \epsilon)| = O(n^2 |\Theta| \log(n|\Theta|))$, a bound that does not depend on ϵ.*

Putting them together. Under full information the defender knows a_t and the payoffs of every type in Θ, so it can compute the loss $\ell_t(\mathbf{p}) = -U_d(b_{a_t}(\mathbf{p}), \mathbf{p})$ of every expert in $\mathcal{E}(\Theta; \epsilon)$. Running an expert algorithm on $\mathcal{E}(\Theta; \epsilon)$ with these losses is Balcan et al. [3, Theorem 5.1]. We use it with two changes, both needed later when the learner is restarted at times the defender cannot foresee. First, since the size of $\mathcal{E}(\Theta; \epsilon)$ does not depend on ϵ (Lemma 2), ϵ can be made as small as we like at no cost; we fix $\epsilon = 1/T$ from the global horizon, so that L rounds lose at most $2L/T$ to the approximation, and any collection of disjoint stretches of rounds loses at most 2 in total. Second, by Proposition 1 the bound holds after every number of rounds L, without L being supplied.

Lemma 3 (Known type set; adapted from Balcan et al. [3, Theorem 5.1])**.** *Let Θ be a finite set of types, $\epsilon = 1/T$, and $\mathcal{A}(\Theta)$ an algorithm satisfying Proposition 1, run on the experts $\mathcal{E}(\Theta;\epsilon)$ with the losses ℓ_t. There is an absolute constant c_1 such that, for every $L \in \{1,\dots,T\}$ and every sequence $(a_1,\dots,a_L) \in \Theta^L$ fixed in advance,*

$$\sup_{\mathbf{p}\in\mathcal{P}} \sum_{t=1}^{L} U_d\big(b_{a_t}(\mathbf{p}),\mathbf{p}\big) - \mathbb{E}\Big[\sum_{t=1}^{L} U_d\big(b_{a_t}(\mathbf{p}_t),\mathbf{p}_t\big)\Big] \;\le\; c_1\, n\sqrt{|\Theta|\, L\, \log(n|\Theta|)} \;+\; \frac{2L}{T}. \tag{5}$$

The learner $\mathcal{A}(\Theta)$ is not told L.

Proof. The left side of Equation (5) splits into two gaps. The first is between the best expert and $\mathcal{A}(\Theta)$. Losses lie in $[-1,1]$, so Proposition 1 with $\kappa = 1$ bounds it by $O\big(\sqrt{L \log|\mathcal{E}(\Theta;\epsilon)|}\big)$, which by Lemma 2 is $O\big(n\sqrt{|\Theta| L \log(n|\Theta|)}\big)$. The second is between the supremum over $\mathcal{P}$ and the best expert, at most $2\epsilon L = 2L/T$ by Lemma 1. Adding the two gives Equation (5). □

Lemma 3 needs every attacker to lie in Θ. When K is unknown, the defender still knows the types it has seen, and until a new type appears every attacker is among them. The next section builds an algorithm on exactly this observation.

3 Unknown Type Set and Full Information: Sublinear Regret

This section treats the case in which the set of attacker types is unknown but the defender learns, after each round, which type attacked.

Problem 1 (Unknown-K repeated SSG)**.** A repeated SSG as in Section 2 is played under the following conditions.

1. The set K of attacker types is finite, and $|K| \le K_{\max}$ for some constant $K_{\max} \ge 1$. Neither K nor $K_{\max}$ is revealed to the defender.
2. Feedback is full information: after round t the defender observes a_t, and the first time any type α appears, its utility vectors (u^c_α, u^u_α) are revealed.

The adversary is oblivious, and the benchmark $\mathbf{p}^*$ and regret $\mathrm{Regret}(T)$ are as in Equations (2) and (3).

Remark 1 (a bound on $|K|$ is necessary). Balcan et al. [3, Theorem 7.1] exhibit, for every T, a game with $|K| < 2^{T+1}$ on which every online algorithm has regret at least $T/2$. Hence sublinear regret needs some bound on $|K|$ that does not grow with T. Problem 1 asks only that such a bound exist, not that the defender know it.

The Grow-Catalog algorithm. The defender does not know K, but at every round it knows which types it has observed so far, together with their utilities. We call this set the *catalog* and write $C_t \subseteq K$ for the catalog at round t. It is the only information about the attacker population on which the choice of $\mathbf{p}_t$ can depend, and it is needed because the defender can optimize only against types it knows. Algorithm 1 treats the catalog as if it were the full type set. The rounds at which a type appears for the first time cut the horizon into *epochs*. Within an epoch the catalog does not change and every attacker that appears belongs to it, so the defender faces a game with a known type set $\Theta = C_t$ and runs the learner of Lemma 3 on it. When a new type appears, the catalog grows, the learner is restarted on the enlarged catalog, and the next epoch begins.

Formally, $\mathcal{A}(C)$ denotes the learner of Lemma 3 with $\Theta = C$: an expert algorithm satisfying Proposition 1, for instance Polynomial Weights, run over the expert set $\mathcal{E}(C;\epsilon)$ with $\epsilon = 1/T$ and

Algorithm 1 GROW-CATALOG (full information).

Require: horizon T, game primitives, default $\mathbf{p}_{\text{def}} \in \mathcal{P}$

$\epsilon \leftarrow 1/T$ ▷ fixed once, from the global horizon
$C \leftarrow \emptyset$; $\mathcal{A} \leftarrow$ NULL ▷ empty catalog, no inner learner yet
for $t = 1, \dots, T$ **do**
  Play $\mathbf{p}_t \leftarrow \mathbf{p}_{\text{def}}$ if $C = \emptyset$, else the strategy $\mathcal{A}$ prescribes for round t
  Observe the attacking type a_t
  if $a_t \notin C$ **then** ▷ discovery round: new attacker type
    $C \leftarrow C \cup \{a_t\}$; record $(u^c_{a_t}, u^u_{a_t})$
    recompute $\mathcal{E}(C; \epsilon)$ ▷ the enlarged expert set, Lemma 1
    $\mathcal{A} \leftarrow \mathcal{A}(C)$ ▷ restart the learner of Lemma 3 on $\mathcal{E}(C; \epsilon)$
  else ▷ routine round
    pass a_t to $\mathcal{A}$ as its round-t feedback
  end if
end for

the losses $\ell_t(\mathbf{p}) = -U_d(b_{a_t}(\mathbf{p}), \mathbf{p})$. Full-information feedback reveals a_t and the utilities of every type in the catalog, so these losses can be evaluated at every expert. The accuracy ϵ is fixed once from the horizon T, which the defender knows, and $\mathcal{A}(C)$ does not need the length of an epoch in advance; the proof of Theorem 1 uses both facts. The round of first contact with a new type falls outside that guarantee and is charged separately in the proof. Algorithm 1 never uses $K_{\max}$.

Our main result for full information is that this simple scheme keeps the $\sqrt{T}$ rate of the known-type case.

Theorem 1 (unknown K with full information is learnable)**.** *Let the defender play Algorithm 1 in Problem 1. For any oblivious sequence $\mathbf{a} \in K^T$,*

$$\text{Regret}(T) \;\le\; O\Big(nK_{\max}\sqrt{T\log(nK_{\max})}\Big). \tag{6}$$

In particular, if $K_{\max} = O(1)$ then $\text{Regret}(T) = \tilde{O}(n\sqrt{T})$, matching the known-type bound of Lemma 3 [3, Theorem 5.1] up to logarithmic factors.

Proof. Idea. Cut the horizon at the rounds where a type attacks for the first time. Between two cuts the catalog is fixed and contains every attacker, so the defender faces the known-type problem and Lemma 3 bounds its regret. Each cut round is charged the largest possible loss, and the Cauchy–Schwarz inequality adds the per-epoch bounds.

Fix an oblivious sequence $\mathbf{a} = (a_1, \dots, a_T) \in K^T$, and let $K' = \{a_1, \dots, a_T\} \subseteq K$ be the set of types that appear in it, so that $|K'| \le |K| \le K_{\max}$. For a set of rounds $S \subseteq [T]$ write

$$R(S) \;=\; \sum_{t \in S} U_d(b_{a_t}(\mathbf{p}^*), \mathbf{p}^*) - \mathbb{E}\Big[\sum_{t \in S} U_d(b_{a_t}(\mathbf{p}_t), \mathbf{p}_t)\Big]$$

for the regret accumulated on S against the benchmark $\mathbf{p}^*$ of Equation (2). Then $R([T]) = \text{Regret}(T)$, and R adds up over disjoint sets of rounds.

Epochs. Let $\tau_1 < \cdots < \tau_{|K'|}$ be the *discovery rounds*, the rounds t with $a_t \notin \{a_1, \dots, a_{t-1}\}$, and set $\tau_{|K'|+1} = T + 1$. The i-th *epoch* is the stretch strictly between two consecutive discovery rounds,

$$\text{Ep}_i = \{\tau_i + 1, \dots, \tau_{i+1} - 1\}, \qquad L_i = |\text{Ep}_i|, \qquad i = 1, \dots, |K'|. \tag{7}$$

Every round is either a discovery round or in exactly one epoch, so with $\mathcal{D} = \{\tau_1, \dots, \tau_{|K'|}\}$,

$$\text{Regret}(T) \;=\; R(\mathcal{D}) + \sum_{i=1}^{|K'|} R(\text{Ep}_i), \qquad \sum_{i=1}^{|K'|} L_i \le T.$$

Since $\mathbf{a}$ is oblivious, the discovery rounds, and hence the epochs, are fixed before play starts and do not depend on what the defender does.

Each epoch. Throughout Ep_i the catalog is $C_i = \{a_{\tau_1}, \dots, a_{\tau_i}\}$, every attacker of the epoch belongs to C_i, and Algorithm 1 runs one copy of $\mathcal{A}(C_i)$, started at the first round of the epoch. The attackers of the epoch form a sequence in C_i fixed in advance, so Lemma 3 with $\Theta = C_i$ and $L = L_i$ applies. It bounds the regret against the best coverage vector *for this epoch alone*. The global benchmark $\mathbf{p}^*$ is one of the candidates in that supremum, so its payoff on the epoch is no larger, and the same bound holds against $\mathbf{p}^*$:

$$R(\mathrm{Ep}_i) \;\le\; c_1\, n\sqrt{|C_i|\, L_i \log(n|C_i|)} + \frac{2L_i}{T} \;\le\; c_1\, n\sqrt{|K'|\log(n|K'|)}\,\sqrt{L_i} + \frac{2L_i}{T}. \tag{8}$$

The second inequality uses $|C_i| \le |K'|$ and that $x \mapsto x\log(nx)$ is increasing. The learner does not need L_i, which is unknown when the epoch starts.

Discovery rounds. Payoffs lie in $[-1, 1]$, so a single round contributes at most 2 to R, whatever the defender plays. There are $|K'|$ discovery rounds, hence

$$R(\mathcal{D}) \;\le\; 2|K'|. \tag{9}$$

Adding up. Summing Equation (8) over the epochs, using $\sum_i 2L_i/T \le 2$, and adding Equation (9) gives $\mathrm{Regret}(T) \le 2|K'| + 2 + c_1 n\sqrt{|K'|\log(n|K'|)}\,\sum_i \sqrt{L_i}$. The Cauchy–Schwarz inequality, applied to $(1, \dots, 1)$ and $(\sqrt{L_1}, \dots, \sqrt{L_{|K'|}})$, bounds the last sum:

$$\sum_{i=1}^{|K'|} \sqrt{L_i} \;\le\; \sqrt{|K'|\textstyle\sum_i L_i} \;\le\; \sqrt{|K'|\,T}. \tag{10}$$

Therefore

$$\mathrm{Regret}(T) \;\le\; c_1\, n|K'|\sqrt{T\log(n|K'|)} \;+\; 2|K'| + 2, \tag{11}$$

and $|K'| \le K_{\max}$ gives Equation (6). Applied once to the whole horizon with the type set known, Lemma 3 would give $c_1 n\sqrt{|K'|\,T\log(n|K'|)}$; the extra factor $\sqrt{|K'|}$ comes from Equation (10) and is the price of not knowing the types in advance. □

Since Algorithm 1 never reads $K_{\max}$, the proof gives more than Theorem 1: the bound Equation (11) depends only on the types that actually appear.

Corollary 1 (instance-adaptive bound)**.** *Let $K' = \{a_1, \dots, a_T\}$ be the set of attacker types that appear in the sequence. Then Algorithm 1 achieves*

$$\mathrm{Regret}(T) \;\le\; O\Big(n|K'|\sqrt{T\log(n|K'|)}\Big).$$

Proof. This is Equation (11), the bound reached in the proof of Theorem 1 before $|K'|$ is replaced by $K_{\max}$. □

The dependence on T in Theorem 1 is the best possible: Proposition 2 in Section 4.5 shows that two types already force regret of order $\sqrt{T}$, even when both are known in advance. What the defender needed in this section, however, was to see the type of each attacker. The next section shows what happens without it.

4 Unknown Type Set and Partial Information: An Impossibility Result

Theorem 1 shows that not knowing K is affordable: under full-information feedback it costs at most a factor $\sqrt{K_{\max}}$ over the known-type rate of Balcan et al. [3, Theorem 5.1], as the last paragraph of its proof records. Not observing a_t is affordable too: with K known and only the attacked target observed, Balcan et al. [3, Theorem 6.1] achieve $O(T^{2/3} n|K| \log^{1/3}(n|K|))$, and Bernasconi et al. [6] improve this to $\tilde{O}(\mathrm{poly}(|K|)\sqrt{T})$.

This section shows that the two relaxations cannot be combined. Under partial information with an unknown type set, no algorithm achieves regret $o(T)$, *even when* $K_{\max} = 1$. Here $K_{\max} = 1$ means that a single type attacks in every round; the defender knows that this type comes from a given infinite universe of types, but not which one it is. The obstruction is exactly this: the defender sees only a target index, and infinitely many types produce the same index, so it can never learn *which* type it faces. What the defender lacks is the identity of its opponent, and Section 5 studies the cheapest way of supplying it.

Problem 2 (Unknown-K repeated SSG, partial information)**.** A repeated SSG as in Section 2 is played, with three modifications.

1. There is a set $\mathfrak{U}$ of attacker types, called the *universe*, known to the defender and not assumed finite.
2. The set $K \subseteq \mathfrak{U}$ is finite, and $|K| \le K_{\max}$ for some constant $K_{\max} \ge 1$. Neither K nor $K_{\max}$ is revealed to the defender.
3. Feedback is *partial information* [3, Section 3]: at the end of round t the defender observes only the attacked target $b_{a_t}(\mathbf{p}_t)$. It never observes a_t, and utility vectors are never revealed.

The adversary is oblivious, and $\mathbf{p}^*$ and $\mathrm{Regret}(T)$ are as in Equations (2) and (3).

The defender's own payoff is not extra feedback: it knows u_d^c and u_d^u, so its payoff $U_d(b_{a_t}(\mathbf{p}_t), \mathbf{p}_t)$ follows from the observed target and its own play $\mathbf{p}_t$. Moreover, Problem 2 is at least as hard as each setting it weakens: revealing a_t recovers Problem 1, and revealing K before the first round recovers Balcan et al. [3, Section 6]. Our lower bounds hold even for a defender that is told $K_{\max}$.

4.1 The lower-bound game G_3

All of our lower bounds are proved in a single small game, which we call G_3; it is the game Balcan et al. [3, Theorem 7.1] use for their unrestricted-K lower bound. There are three targets, $N = \{1, 2, 3\}$, and the defender has one resource that may guard any one of them. The schedules are therefore $\mathcal{D} = \{\emptyset, \{1\}, \{2\}, \{3\}\}$: guard nothing, or guard a single target. The induced coverage vector $\mathbf{p} = (p_1, p_2, p_3)$ records the probability with which each target is guarded, and the set of feasible coverage vectors is

$$\mathcal{P} = \{\mathbf{p} \in [0,1]^3 \ : \ p_1 + p_2 + p_3 \le 1\}. \tag{12}$$

Every coverage vector constructed in the proofs satisfies $p_1 + p_2 + p_3 = 1$, so the arguments apply verbatim when $\mathcal{P}$ is restricted to the simplex.

The defender's payoffs. In G_3 the defender loses one unit whenever target 1 or target 3 is attacked, loses nothing when target 2 is attacked, and in each case the loss is the same whether or not the attacked target happened to be guarded:

$$u_d^c(i) = u_d^u(i) = -1 \ \ (i \in \{1,3\}), \qquad u_d^c(2) = u_d^u(2) = 0. \tag{13}$$

Because the two payoffs coincide for each target, the defender's expected utility Equation (1) does not depend on the coverage at all:

$$U_d(1, \mathbf{p}) = U_d(3, \mathbf{p}) = -1, \qquad U_d(2, \mathbf{p}) = 0 \qquad \text{for every } \mathbf{p} \in \mathcal{P}. \tag{14}$$

Coverage has not become irrelevant; it has become purely *indirect*. The attacker's utilities do depend on $\mathbf{p}$, so by choosing coverage the defender changes which target the attacker prefers, and that choice is the defender's entire problem: it must arrange for the attacker to prefer target 2. Such payoffs are allowed by the model of Section 2, which requires only that all utilities lie in $[-1, 1]$, never that guarding a target reduce the damage of an attack. A round is thus worth 0 if target 2 is attacked and -1 otherwise. We say the defender *steers successfully* at round t if $b_{a_t}(\mathbf{p}_t) = 2$ and *fails* otherwise, so that the regret against a benchmark which always steers successfully is exactly the expected number of failures. Ties in the attacker's best response are broken toward the lowest-indexed target, following Balcan et al. [3, Section 7]; target 2 is the strict maximizer at every benchmark we exhibit, so none of our bounds depends on this convention.

The attacker types. Each attacker type we use in G_3 is described by a pair of thresholds $0 \le r_1 < r_2 \le 1$. The type α_{r_1,r_2} has utilities

$$u^c_\alpha = (r_1 - 1,\ 0,\ -r_2), \qquad u^u_\alpha = (r_1,\ 0,\ 1 - r_2); \tag{15}$$

that is, attacking target 1 pays the attacker $r_1 - 1$ if the target is covered and r_1 if it is not, target 2 pays 0 either way, and target 3 pays $-r_2$ or $1 - r_2$. All six entries lie in $[-1, 1]$. Substituting them into $U_\alpha(i, \mathbf{p}) = u^c_\alpha(i)\, p_i + u^u_\alpha(i)\,(1 - p_i)$ gives

$$U_\alpha(1, \mathbf{p}) = r_1 - p_1, \quad U_\alpha(2, \mathbf{p}) = 0, \quad U_\alpha(3, \mathbf{p}) = (1 - r_2) - p_3. \tag{16}$$

In words: attacking target 1 is attractive when target 1 is lightly guarded, attacking target 3 is attractive when target 3 is lightly guarded, and target 2 is a neutral fallback worth exactly zero.

A one-dimensional picture. It is convenient to look at coverage vectors that split the single resource between targets 1 and 3 and never guard target 2. Write these as $\mathbf{p} = (p, 0, 1 - p)$ for a scalar $p \in [0, 1]$. Substituting $p_1 = p$ and $p_3 = 1 - p$ into Equation (16) turns the three utilities into $r_1 - p$, 0 and $p - r_2$. Comparing them, and breaking ties toward the lowest index, gives

$$b_{\alpha_{r_1,r_2}}((p, 0, 1 - p)) = \begin{cases} 1 & p \in [0, r_1], \\ 2 & p \in (r_1, r_2], \\ 3 & p \in (r_2, 1]. \end{cases} \tag{17}$$

So along this segment the type α_{r_1,r_2} may be pictured as the interval $(r_1, r_2]$: it strikes target 2 precisely when p falls inside the interval, and strikes target 1 or target 3 according to whether p falls short of the interval or overshoots it. For example, the type $\alpha_{0.3,0.5}$ attacks target 1 under $(0.2, 0, 0.8)$, target 2 under $(0.4, 0, 0.6)$ and target 3 under $(0.7, 0, 0.3)$. Locating an unknown type means locating its interval.

An online algorithm need not restrict its plays to this segment: it may also guard target 2 or leave some probability unassigned. Since the lower bounds must apply to all possible plays, the following lemma describes the attacker's behavior over the whole of $\mathcal{P}$; its proof is in Appendix A.1.

Lemma 4 (when a threshold type strikes target 2)**.** *Let $0 \le r_1 < r_2 \le 1$ and $\alpha = \alpha_{r_1,r_2}$. For every $\mathbf{p} = (p_1, p_2, p_3) \in \mathcal{P}$,*

$$b_\alpha(\mathbf{p}) = 2 \iff p_1 > r_1 \quad \text{and} \quad p_3 \ge 1 - r_2. \tag{18}$$

Moreover, whenever $b_\alpha(\mathbf{p}) = 2$, the coverage of target 1 satisfies $p_1 \in (r_1, r_2]$.

The second assertion is the key to the lower bounds. Although the condition on p_3 may hold for several types at once, the single-resource constraint forces $p_1 \le r_2$, so the type α_{r_1,r_2} can strike target 2 only when $p_1 \in (r_1, r_2]$. It is therefore the coordinate p_1 that separates the types.

4.2 A family of types with disjoint intervals

The lower bounds use families of threshold types whose intervals do not overlap. The basic one chops the unit interval into $M \ge 2$ equal pieces:

$$I_j = \left(\tfrac{j-1}{M}, \tfrac{j}{M}\right], \qquad \alpha^{(j)} = \alpha_{\frac{j-1}{M}, \frac{j}{M}}, \qquad j \in [M], \tag{19}$$

and we write $\mathfrak{U}_M = \{\alpha^{(1)}, \ldots, \alpha^{(M)}\}$. By Equation (17) these M types respond differently to coverage vectors on the segment, so they are distinct. By Lemma 4, disjoint intervals mean that no coverage vector can serve two of them at once.

Corollary 2 (one coverage vector steers at most one type)**.** *Let $\alpha^{(1)}, \ldots, \alpha^{(M)}$ be threshold types in G_3 whose intervals are pairwise disjoint, for instance those of Equation* (19)*. For every $\mathbf{p} \in \mathcal{P}$, at most one of them strikes target 2 under $\mathbf{p}$.*

The proof, two lines from Lemma 4, is in Appendix A.1.

4.3 Identification is a comparison search

Suppose the true instance is $K = \{\alpha^{(j)}\}$ for one unknown index $j \in [M]$, so that the same type attacks in every round. Then the defender's situation is a search problem. In each round it chooses a coverage vector and observes the attacked target, which, for coverage vectors on the segment $(p, 0, 1-p)$, tells it by Equation (17) whether p fell below, inside, or above the unknown interval I_j. Two of the three possible answers, attacks on targets 1 or 3, merely narrow the range of candidates, like a comparison in binary search; the third, an attack on target 2, identifies j exactly, and by Corollary 2 it is the only answer that can do so. The defender pays one unit of regret for every round in which it has not yet received that third answer.

The next lemma bounds how quickly any such search can succeed. It sharpens the classical fact that a worst-case comparison search over M candidates needs $\Omega(\log_2 M)$ queries in two respects: it charges the defender for every unsuccessful round rather than counting rounds until success, and it controls the whole distribution of the success time. In the lemma the attacker's type is drawn uniformly at random. This is a choice of hard instance, not an assumption about the adversary: a lower bound need only exhibit instances on which the defender does badly. If the regret averaged over the draw is large, then some fixed type makes it at least as large, and that fixed type is the instance used in Theorem 2. Drawing the type at random also handles randomized defenders, since their coin tosses are independent of the draw; this is the easy direction of Yao's minimax principle [51].

Lemma 5 (a search over disjoint intervals cannot succeed quickly)**.** *Let $\alpha^{(1)}, \ldots, \alpha^{(M)}$ be threshold types in G_3 whose intervals are pairwise disjoint. Draw j uniformly at random from $[M]$, and*

suppose $\alpha^{(j)}$ attacks in every round. Consider a defender whose coverage vector $\mathbf{p}_t$ in round t is determined by the targets $i_1, \ldots, i_{t-1}$ it saw attacked in the earlier rounds, where $i_s = b_{\alpha^{(j)}}(\mathbf{p}_s)$. Let τ be the first round in which the attacker strikes target 2, with $\tau = \infty$ if this never happens. Then for every $u \geq 1$ and every $T \geq 1$,

$$\Pr[\tau \leq u] \;\leq\; \frac{2^u - 1}{M}, \tag{20}$$

$$\mathbb{E}[\min\{\tau - 1, T\}] \;\geq\; T - \frac{2^{T+1} - T - 2}{M}. \tag{21}$$

The defender never sees j. It sees only its own coverage vectors and the attacked targets, and j enters solely through the attacker's best response $b_{\alpha^{(j)}}(\mathbf{p}_s)$. The intuition is binary search in which only an exact hit counts. Each failed round tells the defender one bit, whether target 1 or target 3 was struck, so its possible plays form a binary tree with at most 2^{t-1} branches in round t, and each branch can catch at most one candidate. With $M = 2^T$ candidates, half of them can be caught only in the last round, a quarter only in the round before, and so on. This is the mechanism behind the impossibility of online learning of thresholds, whose Littlestone dimension is infinite [5, 32]; Lemma 5 adapts it to G_3, where a coverage vector yields one comparison and success needs an exact hit.

Proof sketch. Before its first success the defender has seen only attacks on targets 1 or 3, so at the start of round t its history is one of at most 2^{t-1} strings over $\{1, 3\}$, and each history determines one coverage vector. By Corollary 2 each coverage vector steers at most one candidate onto target 2. Hence at most 2^{t-1} values of j have $\tau = t$, so $\Pr[\tau = t] \leq 2^{t-1}/M$. Summing over $t \leq u$ gives Equation (20), and the tail-sum formula for $\min\{\tau - 1, T\}$ gives Equation (21). The full proof is in Appendix A.2. □

4.4 The impossibility theorem

Taking $M = 2^T$ in Lemma 5 makes the search too long for the horizon, which is our main result for partial information.

Theorem 2 (unknown K with partial information is not learnable)**.** *Let G_3 be as in Equation* (13) *and let $\mathfrak{U} = \{\alpha_{r_1,r_2} : 0 \leq r_1 < r_2 \leq 1\}$ be the infinite universe of threshold types Equation* (15)*. For every $T \geq 1$ and every $K_{\max} \geq 1$, every online algorithm for Problem 2, randomized or deterministic, even one knowing T, $\mathfrak{U}$, $K_{\max}$ and the construction, suffers, on some instance with $|K| = 1$,*

$$\mathrm{Regret}(T) \;\geq\; T - 2 + \frac{T+2}{2^T} \;>\; T - 2. \tag{22}$$

The game is fixed; only the adversary's choice of K within it depends on T.

Proof sketch. Set $M = 2^T$, draw j uniformly from $[M]$ and let $\alpha^{(j)}$ of Equation (19) attack in every round. The coverage vector whose first coordinate is the midpoint of I_j steers $\alpha^{(j)}$ in every round, with both conditions of Equation (18) holding strictly, so the benchmark earns 0 and regret counts failed rounds. Under partial information a deterministic algorithm chooses each coverage vector from the targets attacked so far, so Lemma 5 with $M = 2^T$ applies directly and yields Equation (22) on average over j. Averaging over the algorithm's coins extends this to randomized algorithms, and some fixed j attains the average. The full proof is in Appendix A.3. □

Remark 2 (the bound is exactly tight). The constant cannot be improved. The defender that keeps the set of indices consistent with its observations, which by Equation (17) is an integer interval,

and plays $\mathbf{p}_t = (k/M, 0, 1 - k/M)$ for its midpoint k attains regret exactly $T - 2 + (T + 2)2^{-T}$ on average over j. Indeed, up to round T every history still holds a nonempty interval, so each history catches exactly one candidate and Equation (20) holds with equality; Appendix C.4 confirms this in simulation. Since $\mathrm{Regret}(T) \le T$ always in G_3, the defender loses all but two of the T rounds.

Letchford et al. [31] show that a defender facing a *single* unknown attacker type can learn a near-optimal commitment from a small number of best-response queries, which at first sight is in tension with Theorem 2. There is no contradiction. Any such query bound must depend on the geometry of the best-response regions, and the family $\mathfrak{U}_M$ of Equation (19) drives the region that has to be located down to width $1/M = 2^{-T}$. Bollini et al. [10] prove a related impossibility for Bayesian Stackelberg games with three types drawn i.i.d.; there, payoffs of polynomial precision suffice, whereas a single type, as in Theorem 2, requires the universe to be exponentially large in T. Lemma 5 measures this cost of precision exactly. With the finite universe $\mathfrak{U}_M$ and a single type, summing $\Pr[\tau > u] \ge 1 - 2^u/M$ from Equation (20) over $u \le \lfloor \log_2 M \rfloor$ gives regret at least $\min\{T, \lfloor \log_2 M \rfloor\} - 2$, and the defender of Remark 2 identifies the type within $\lceil \log_2 M \rceil$ rounds. A single type therefore costs $\Theta(\min\{T, \log M\})$, and Theorem 2 is the case of payoffs given to T bits of precision.

4.5 Two types already cost $\sqrt{T}$

Theorem 2 is driven entirely by the universe being infinite. With a single type from a finite universe $\mathfrak{U}_M$, the defender of Remark 2 identifies the type after $\lceil \log_2 M \rceil$ rounds and loses nothing afterwards, so its regret does not grow with T. Two types are different: even when the defender knows both of them and learns the type after every round, it must keep guessing which one attacks next, and this alone costs $\sqrt{T}$.

Proposition 2 (two types already cost $\sqrt{T}$)**.** *Consider G_3 with the two types $\alpha^{(1)} = \alpha_{0,1/2}$ and $\alpha^{(2)} = \alpha_{1/2,1}$, so that $|\mathfrak{U}| = |K| = K_{\max} = 2$. For every $T \ge 2$ and every online algorithm, randomized or deterministic, under either feedback model, and even one told K and the utilities of both types before the first round, there is a sequence $\mathbf{a} \in K^T$ with*

$$\mathrm{Regret}(T) \;\ge\; \tfrac{1}{2}\sqrt{\lfloor T/2 \rfloor}. \tag{23}$$

In fact the bound holds on average over $\mathbf{a}$ drawn i.i.d. uniformly from K, so some fixed sequence attains it.

Proof sketch. Draw the attackers i.i.d. uniformly from K, and let N_1 and N_2 count the rounds of each type. The two intervals are disjoint, so by Corollary 2 a fixed coverage vector steers at most one type; the best one steers the more frequent type and fails only on the rounds of the other, so the benchmark earns $-\min\{N_1, N_2\}$. An algorithm, whatever it has seen, cannot predict the next type, so its coverage vector steers the attacker with probability at most $1/2$ in each round, and it earns at most $-T/2$ in expectation. The gap is $\frac{1}{2}\mathbb{E}|N_1 - N_2|$, half the expected distance from the origin of a simple random walk after T steps, which is at least $\frac{1}{2}\sqrt{\lfloor T/2 \rfloor}$. The full proof is in Appendix A.4. □

So the $\sqrt{T}$ dependence of Theorem 1 cannot be improved, even with the type set known. Returning to Theorem 2: the defender failed there because it could never learn the identity of its opponent. The next section asks how little information about identities is enough to restore sublinear regret.

5 Block Revelation of Attacker Types: Sublinear Regret

Theorem 2 shows that under partial information the defender must receive some information about the identities of the attackers; observing only the attacked target is not enough. This section studies a minimal form of such information. Every B rounds, an oracle gives the defender the *set* of attacker types seen so far, together with their utility vectors. The oracle does not reveal when a type attacked, how often it attacked, or whether it will attack again, so the defender still cannot tell which type was responsible for any particular round. For example, an anti-poaching unit may receive a monthly intelligence report listing the poaching groups known to be active in a park, without saying which incident each group was responsible for.

Identifying the attacker from its observed best responses is exactly the difficulty considered by Balcan et al. [3, Section 6]. Our additional information does not remove this difficulty; it only tells the defender which types are present. Our algorithm therefore runs a partial-information learner for a known type set on the types reported so far, and its regret is that learner's rate, summed over the stretches between new types, plus an additive cost for the delay. With the learner of Balcan et al. [3] this gives a bound of order $T^{2/3}$.

Problem 3 (Unknown-K repeated SSG with block revelation)**.** Fix a *revelation length* $B \in \mathbb{N}$, known to the defender, and divide the horizon into $N_B = \lceil T/B \rceil$ *revelation blocks*

$$\mathcal{B}_\nu = \{(\nu-1)B+1,\ \ldots,\ \min\{\nu B, T\}\}, \qquad \nu \in [N_B]. \tag{24}$$

The game is Problem 2 with the following additional feedback: at the end of $\mathcal{B}_\nu$, the defender receives the set

$$D_\nu = \{a_t :\ t \le \min\{\nu B, T\}\} \subseteq K \tag{25}$$

of distinct types that have attacked in the first ν blocks, together with the utility vectors of all types in D_ν. We set $D_0 = \emptyset$. No other information is revealed, and within each round the feedback remains partial.

This model is weaker than Problem 1: even at $B = 1$ the defender learns a round's type only when that type is new. Taking $B \ge T$ makes the revelation useless and recovers Problem 2, so by Theorem 2 some bound on B is necessary.

5.1 The algorithm

Algorithm 2 keeps the outer skeleton of Algorithm 1: it maintains a catalog C of the types the defender has been told about, runs an inner learner on the experts $\mathcal{E}(C;\epsilon)$, and restarts the inner learner whenever C grows. It differs from Algorithm 1 in two places. First, the catalog can grow only at the checkpoints $t = B, 2B, \ldots$, so restarts happen at block boundaries rather than at the exact round a new type first attacks. Second, the full-information learner $\mathcal{A}(C)$ cannot be used: it updates every expert with the loss $-U_d(b_{a_t}(\mathbf{p}), \mathbf{p})$, which requires knowing a_t. We therefore use the partial-information learner of Balcan et al. [3, Algorithm 1], which we denote $\mathcal{A}^{\mathrm{pi}}(C)$ and which needs only the attacked target.

We treat $\mathcal{A}^{\mathrm{pi}}(C)$ as a black box. Its input is a catalog C, which fixes the expert set $\mathcal{E}(C;\epsilon)$; at each round it receives the observed target i_t and outputs the coverage vector $\mathbf{p}_t$ to play. Internally it groups its rounds into estimation blocks, which are unrelated to the revelation blocks $\mathcal{B}_\nu$, explores on a few rounds of each block, uses them to form an unbiased estimate of every expert's average payoff on the block, and feeds that estimate to an expert algorithm whose output drives the next block. Appendix B.1 describes it in more detail. The analysis uses its regret bound, which Balcan

Algorithm 2 BLOCK-GROW-CATALOG (partial information, revelation length B).

Require: horizon T; revelation length B; game primitives; default strategy $\mathbf{p}_{\text{def}} \in \mathcal{P}$

$C \leftarrow \emptyset$; $\epsilon \leftarrow 1/T$; $\mathcal{A}^{\text{pi}} \leftarrow$ NULL ▷ empty catalog
for $t = 1, \ldots, T$ **do**
  Play $\mathbf{p}_t \leftarrow \mathbf{p}_{\text{def}}$ if $C = \emptyset$, else the strategy $\mathcal{A}^{\text{pi}}$ prescribes for round t
  Observe attacked target $i_t = b_{a_t}(\mathbf{p}_t)$
  pass i_t to $\mathcal{A}^{\text{pi}}$ as its round-t feedback
  if $t = \nu B$ for some ν, or $t = T$ **then**
    receive $D_\nu \subseteq K$ and the utilities of each $\alpha \in D_\nu$
    if $D_\nu \neq C$ **then** ▷ new type(s) appeared during $\mathcal{B}_\nu$
      $C \leftarrow D_\nu$
      $\mathcal{A}^{\text{pi}} \leftarrow \mathcal{A}^{\text{pi}}(C)$ ▷ restart on enlarged catalog
    end if
  end if
end for

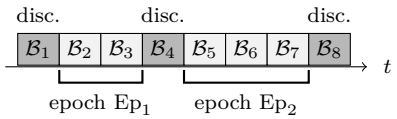


Figure 1: The block decomposition. Shaded blocks are the *discovery blocks*: their closing checkpoint enlarges the catalog. The run of unshaded blocks strictly between two consecutive discovery blocks forms an *epoch*; inside an epoch the catalog is fixed and Algorithm 2 runs one uninterrupted inner learner.

et al. [3] prove for a known set K of types, together with one further fact shown in the proof of Lemma 7: the bound also holds at every prefix of a run.

Theorem 3 (Balcan et al. [3, Theorem 6.1])**.** *Given a repeated security game with partial-information feedback, n targets, a known set K of attacker types, and time horizon T, there is an online algorithm that for any oblivious sequence of attackers $\mathbf{a}$ plays at time t a randomly chosen coverage vector $\mathbf{p}_t$ and satisfies*

$$\mathbb{E}\Big[\sum_{t=1}^{T} U_d(b_{a_t}(\mathbf{p}_t), \mathbf{p}_t)\Big] \;\geq\; \sum_{t=1}^{T} U_d(b_{a_t}(\mathbf{p}^*), \mathbf{p}^*) - O\big(T^{2/3}\, n|K| \log^{1/3}(n|K|)\big),$$

where the expectation is over the algorithm's internal randomization.

In our use the known set is the catalog C and the algorithm is $\mathcal{A}^{\text{pi}}(C)$. Its horizon must be known in advance; Lemma 7 below removes this requirement.

5.2 Discovery blocks and epochs

The analysis cuts the horizon into two kinds of stretches, shown in Figure 1: *discovery blocks*, during which a type the defender has not yet been told about may attack, and *epochs*, the runs of blocks between them, during which every attacker is already in the catalog. The proof charges discovery blocks the maximum regret and applies the learner's guarantee on each epoch.

Definition 1 (discovery blocks and epochs)**.** A revelation block $\mathcal{B}_\nu$ is a *discovery block* if its report names a type not named before, that is, $D_\nu \neq D_{\nu-1}$. Let $\mathcal{B}_{\beta_1}, \ldots, \mathcal{B}_{\beta_m}$ be the discovery blocks, in

order. The j-th *epoch* Ep_j consists of the blocks strictly between $\mathcal{B}_{\beta_j}$ and $\mathcal{B}_{\beta_{j+1}}$, or up to the end of the horizon when $j = m$. We write L_j for the number of rounds in Ep_j and $C_j \coloneqq D_{\beta_j}$ for the catalog during it.

Lemma 6 (decomposition)**.** *Let $K' = \{a_1, \dots, a_T\}$ be the set of types appearing in $\mathbf{a}$.*

(a) There are $m \le |K'|$ discovery blocks, and together with the epochs they partition $[1, T]$; in particular $\sum_j L_j \le T$.

(b) Throughout Ep_j the catalog is C_j, every attacker belongs to C_j, and a single instance of $\mathcal{A}^{\mathrm{pi}}(C_j)$ runs, started at the first round of Ep_j.

(c) The discovery blocks and the epochs depend only on $\mathbf{a}$, not on the defender's play.

Proof. (a) The first block is a discovery block, since $D_0 = \emptyset$, and every discovery block names at least one of the $|K'|$ types for the first time. (b) A block that is not a discovery block names no new type, so every type attacking in it had already been named; and Algorithm 2 changes its catalog, and restarts its learner, only after a discovery block. (c) Whether a block is a discovery block depends only on which types attack, and the oblivious adversary fixes these before play begins. □

Part (c) is where the obliviousness of the adversary is used: it makes the attackers within an epoch an oblivious sequence, as the learner's guarantee requires. What the learner does during discovery blocks does not matter, since the proof charges those rounds the maximum regret.

The proof of Theorem 4 applies Theorem 3 on each epoch. That theorem needs its horizon in advance, whereas an epoch ends only when a new type appears, so its length is unknown when it starts. The following lemma removes this requirement.

Lemma 7 (anytime partial-information learner)**.** *For every finite catalog C there is a variant of $\mathcal{A}^{\mathrm{pi}}(C)$ that does not require its horizon in advance, and an absolute constant c_4, such that on any oblivious sequence $a'_1, a'_2, \dots$ with $a'_t \in C$ and for every $L \le T$ simultaneously,*

$$\max_{\mathbf{p}\in\mathcal{P}} \sum_{t=1}^{L} U_d(b_{a'_t}(\mathbf{p}), \mathbf{p}) - \mathbb{E}\Big[\sum_{t=1}^{L} U_d(b_{a'_t}(\mathbf{p}_t), \mathbf{p}_t)\Big] \;\le\; c_4\, L^{2/3}\, n\, |C|\, \log^{1/3}(n|C|). \tag{26}$$

The benchmark on the left is the best fixed coverage vector for the sequence $a'_1, \dots, a'_L$ alone; the proof of Theorem 4 passes from it to the global benchmark $\mathbf{p}^*$. The variant is built with the doubling trick of Auer et al. [1] [see also 11, Sec. 2.3], which turns a learner tuned to a known horizon into one that needs none.

Proof sketch. (i) A run tuned to a horizon H obeys its bound at every prefix. The proof of Theorem 3 bounds the regret of $\mathcal{A}^{\mathrm{pi}}(C)$ by two error terms: the payoff lost on exploration rounds, and the regret of its internal expert algorithm on the estimated payoffs. Stopping that argument after $L \le H$ rounds bounds the regret at round L by the same two terms evaluated at L (a block cut in the middle adds only a lower-order term). Both terms are nondecreasing in L, so at every prefix they are at most their values at H: the regret at any round $L \le H$ is at most $c_0 H^{2/3}\phi$, the bound of Theorem 3, where $\phi \coloneqq n|C|\log^{1/3}(n|C|)$. *(ii) Doubling.* The variant runs fresh copies of $\mathcal{A}^{\mathrm{pi}}(C)$ tuned to horizons $1, 2, 4, \dots$, one after another, so it needs no horizon. Given L, let S be the stage containing round L; the earlier stages fill $2^S - 1$ rounds, so $2^S \le L$. Only stage S can be cut off before its end, and by (i) its bound still holds. The best fixed vector over $[1, L]$ earns at most the sum of the stages' best vectors, so the regret over $[1, L]$ is at most the sum of the stage bounds,

$$c_0\phi \sum_{s=0}^{S} 2^{2s/3} \;\le\; \frac{2^{2/3}}{2^{2/3}-1}\, c_0\phi\, 2^{2S/3} \;\le\; 2.71\, c_0\phi\, L^{2/3},$$

which is Equation (26) with $c_4 = 2.71\, c_0$. The full proof is in Appendix B.2. □

With these pieces in place we can state our main result for block revelation, in which Algorithm 2 is run with the learner of Lemma 7.

Theorem 4 (block revelation restores no-regret)**.** *Let the defender play Algorithm 2 in Problem 3, and let $K' = \{a_1, \dots, a_T\}$ be the set of types appearing in the oblivious sequence $\mathbf{a} \in K^T$. Then*

$$\mathrm{Regret}(T) \;\le\; c\, n\, |K'|^{4/3}\, T^{2/3} \log^{1/3}(n|K'|) \;+\; 2|K'|B \tag{27}$$

for an absolute constant c; in particular $\mathrm{Regret}(T) = O(nK_{\max}^{4/3} T^{2/3} \log^{1/3}(nK_{\max}) + K_{\max} B)$.

Proof. Adopt the notation of Definition 1 and Lemma 6. For a set of rounds $S \subseteq [1, T]$ let

$$\mathrm{Regret}(S) = \sum_{t \in S} U_d(b_{a_t}(\mathbf{p}^*), \mathbf{p}^*) - \mathbb{E}\Big[\sum_{t \in S} U_d(b_{a_t}(\mathbf{p}_t), \mathbf{p}_t)\Big] \tag{28}$$

denote the regret incurred on the rounds in S; it is additive over disjoint sets of rounds. The idea is to split the rounds into two kinds, bound the regret on each kind separately, and add.

Splitting the rounds. By Lemma 6(a), the horizon is the disjoint union of the $m \le |K'|$ discovery blocks $\mathcal{B}_{\beta_j}$ and the m epochs Ep_j, so by additivity

$$\mathrm{Regret}(T) = \sum_{j=1}^{m} \mathrm{Regret}(\mathcal{B}_{\beta_j}) + \sum_{j=1}^{m} \mathrm{Regret}(\mathrm{Ep}_j). \tag{29}$$

Discovery blocks. Inside a discovery block the defender may face a type it has not yet been told about, so we make no use of what the learner does there and charge each round the largest possible regret. Since all payoffs lie in $[-1, 1]$, a round contributes at most 2 to Equation (28); each discovery block holds at most B rounds, and there are at most $|K'|$ of them, so

$$\sum_{j=1}^{m} \mathrm{Regret}(\mathcal{B}_{\beta_j}) \;\le\; 2|K'|B. \tag{30}$$

Epochs. Fix an epoch Ep_j. By Lemma 6(b), a single instance of $\mathcal{A}^{\mathrm{pi}}(C_j)$ runs throughout it and every attacker in it belongs to C_j; by Lemma 6(c), the attackers in it form an oblivious sequence. These are exactly the hypotheses of Lemma 7, which moreover does not require the epoch length L_j to be known in advance. The lemma bounds the regret against the best fixed coverage vector *for this epoch*; since $\mathbf{p}^*$ is one of the candidates in that maximum, the same bound holds against $\mathbf{p}^*$. Using also $|C_j| \le |K'|$, as the catalog only ever contains types that appear,

$$\mathrm{Regret}(\mathrm{Ep}_j) \;\le\; c_4\, n\, |C_j| \log^{1/3}(n|C_j|)\, L_j^{2/3} \;\le\; c_4\, n\, |K'| \log^{1/3}(n|K'|)\, L_j^{2/3}. \tag{31}$$

Adding up. The adversary decides when new types arrive and so controls the epoch lengths, subject only to $m \le |K'|$ and $\sum_j L_j \le T$. By Hölder's inequality with exponents 3 and 3/2, applied to the vectors $(1, \dots, 1)$ and $(L_1^{2/3}, \dots, L_m^{2/3})$,

$$\sum_{j=1}^{m} L_j^{2/3} \;\le\; m^{1/3} \Big(\sum_{j=1}^{m} L_j\Big)^{2/3} \;\le\; |K'|^{1/3}\, T^{2/3}, \tag{32}$$

with equality when all epochs have the same length: the adversary does the most damage by spreading the arrivals of new types evenly. Substituting Equations (30) to (32) into Equation (29) gives

$$\mathrm{Regret}(T) \;\le\; c_4\, n\, |K'|^{4/3}\, T^{2/3} \log^{1/3}(n|K'|) + 2|K'|B,$$

which is Equation (27) with $c = c_4$; using $|K'| \le K_{\max}$ gives its $K_{\max}$ form. The exponent $4/3$ collects a factor $|K'|$ from the size of the catalog inside each epoch and a factor $|K'|^{1/3}$ from summing over at most $|K'|$ epochs. □

The proof uses only one property of the inner learner: an anytime bound on its regret within each epoch. A learner with a better rate therefore gives a better bound. If the per-epoch regret is at most $g(|C|)\sqrt{L}$, the same argument, with the Cauchy–Schwarz inequality in place of Hölder's, gives $\mathrm{Regret}(T) \le g(|K'|)\sqrt{|K'|T} + 2|K'|B$. The learner of Bernasconi et al. [6] attains $\tilde{O}(\mathrm{poly}(|K|)\sqrt{T})$ for a known type set under partial information, so if it admits an anytime version, block revelation costs only the additive delay term on top of a $\sqrt{T}$ rate. We state Theorem 4 with the learner of Balcan et al. [3] because its anytime version is the one we prove (Lemma 7).

5.3 How rarely reports are needed

The bound Equation (27) has three ingredients. The exponent $2/3$ is that of the partial-information learner of Theorem 3, not of the model; the extra factor $|K'|^{1/3}$ is the price of learning the catalog gradually, since the learner is restarted once for each new type; and the additive $2|K'|B$ is the price of hearing about new types only at the end of a block. Because Algorithm 2 never reads $K_{\max}$, the bound depends only on the types that actually appear.

Block revelation sits between the two feedback models of the earlier sections. As B grows it approaches the partial-information model of Section 4, where no algorithm learns, and at $B = 1$ it reduces to partial information with a known type set, apart from the rounds at which new types first appear.

Reports are costly in practice, since each is a piece of intelligence that someone has to gather, so it matters how rarely they can arrive. The delay is free as long as the batching term $2|K'|B$ stays below the learning term.

Corollary 3 (how rarely reports are needed)**.** *If*

$$B \;\le\; B^* \coloneqq \tfrac{1}{2}\, c\, n\, K_{\max}^{1/3}\, T^{2/3} \log^{1/3}(nK_{\max}), \tag{33}$$

then $\mathrm{Regret}(T) = O\big(nK_{\max}^{4/3} T^{2/3} \log^{1/3}(nK_{\max})\big)$*, the same rate as with a report after every round. The defender therefore needs only*

$$\lceil T/B^* \rceil \;=\; O\!\left(\frac{T^{1/3}}{n\, K_{\max}^{1/3} \log^{1/3}(nK_{\max})}\right) \tag{34}$$

reports over the whole horizon.

The threshold B^* guides whoever schedules the reports; the defender itself need not know it, and Algorithm 2 never uses it.

Proof. For $B \le B^*$ the batching term satisfies $2|K'|B \le 2K_{\max}B^*$, which is the learning term of Equation (27) evaluated at $|K'| = K_{\max}$, so the bound at most doubles. The number of reports is $\lceil T/B \rceil$, which is smallest at $B = B^*$. □

For example, with $T = 10^6$ rounds and a handful of types, $T^{2/3} = 10^4$, so, ignoring the factors in n and $K_{\max}$, a report every 10,000 rounds, about 100 reports in all, gives the same rate as a report after every round.

5.4 The additive term is necessary

The additive term of Theorem 4 is not an artifact of the analysis: no algorithm can avoid a cost of order $K_{\max}B$.

Theorem 5 (a matching lower bound in B)**.** *Let G_3 and $\mathfrak{U}$ be as in Theorem 2. For every $B \geq 1$, every $K_{\max} \geq 1$, every $T \geq K_{\max}B$ and every online algorithm for Problem 3, there is an instance with $|K| = K_{\max}$ on which*

$$\text{Regret}(T) \;\geq\; K_{\max}\,(B-2). \tag{35}$$

Proof sketch. The adversary uses the game G_3 of Section 4.1 and picks $K_{\max}$ types that one fixed coverage vector would steer onto target 2 all at once, so the best fixed strategy loses nothing and every failed round counts as regret. The types arrive one per revelation block, and each new type is drawn at random from 2^B candidates; the defender learns which one it was only when the oracle names it at the end of that block. Inside the block the defender must therefore find the new type by trial and error from the attacked targets alone, and by Lemma 5 such a search over 2^B candidates fails in more than $B-2$ of its B rounds in expectation over the draw. This cost is paid once for each of the $K_{\max}$ types. Since the bound holds on average over the random draw, some fixed draw attains it, and that fixed draw is the instance in the statement. The full proof is in Appendix B.3. □

Together with Theorem 4, the cost of batching is $\Theta(K_{\max}B)$ for $B \geq 3$. Appendix B.3 compares the two bounds and shows that, in the regime where the batching term dominates, they differ by a factor of at most $2B/(B-2)$, up to lower-order terms. At $K_{\max} = 1$ and $B = T$ the lower bound reads $T-2$, so Theorem 2 is the limiting case of Theorem 5 in which no report arrives in time. We now step back and summarize what the three settings show together.

6 Conclusions and Future Work

We studied repeated Stackelberg security games in which the defender does not know the set of attacker types, only that it is finite, and showed that learnability then hinges on what the defender observes. Seeing the type of each attacker, Grow-Catalog attains $\tilde{O}(nK_{\max}\sqrt{T})$ regret without knowing $K_{\max}$, the known-type rate up to a factor $\sqrt{K_{\max}}$. Seeing only the attacked target, no algorithm achieves sublinear regret, even against a single type, because the defender cannot identify its opponent. A periodic list of the types seen so far is enough to restore sublinear regret, $\tilde{O}(nK_{\max}^{4/3}T^{2/3} + K_{\max}B)$, with an additive cost of the delay that is unavoidable, and only $O(T^{1/3})$ such reports are needed.

Several questions remain open. Neither the factor $\sqrt{K_{\max}}$ of Theorem 1 nor the factor $K_{\max}^{1/3}$ of Theorem 4 is known to be necessary. It is also open whether the $\sqrt{T}$ learner of Bernasconi et al. [6] admits an anytime version, which would bring block revelation to a $\sqrt{T}$ rate plus the delay term, and, since a single type costs only $\Theta(\log M)$ at precision $1/M$, whether a few types already force polynomial regret in security games at polynomial payoff precision, as three do in the general games of Bollini et al. [10]. Finally, our adversary is oblivious; extending the results to adaptive attackers, or to reports that are noisy or incomplete, would bring the model closer to the intelligence that defenders actually receive.

References

[1] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 322–331, 1995.

[2] B. Awerbuch and R. Kleinberg. Online linear optimization and adaptive routing. *Journal of Computer and System Sciences*, 74(1):97–114, 2008.

[3] M.-F. Balcan, A. Blum, N. Haghtalab, and A. D. Procaccia. Commitment without regrets: online learning in Stackelberg security games. In *Proceedings of the 16th ACM Conference on Economics and Computation (EC)*, pages 61–78, 2015.

[4] M.-F. Balcan, M. Bernasconi, M. Castiglioni, A. Celli, K. Harris, and Z. S. Wu. Nearly-optimal bandit learning in Stackelberg games with side information. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2502.00204.

[5] S. Ben-David, D. Pál, and S. Shalev-Shwartz. Agnostic online learning. In *Proceedings of the 22nd Conference on Learning Theory (COLT)*, 2009.

[6] M. Bernasconi, M. Castiglioni, A. Celli, A. Marchesi, F. Trovò, and N. Gatti. Optimal rates and efficient algorithms for online Bayesian persuasion. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

[7] L. Besson and E. Kaufmann. What doubling tricks can and can't do for multi-armed bandits, 2018. arXiv:1803.06971.

[8] A. Blum and Y. Mansour. Learning, regret minimization, and equilibria. In N. Nisan, T. Roughgarden, É. Tardos, and V. Vazirani, editors, *Algorithmic Game Theory*, chapter 4. Cambridge University Press, 2007.

[9] A. Blum, N. Haghtalab, and A. D. Procaccia. Learning optimal commitment to overcome insecurity. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[10] M. Bollini, F. Bacchiocchi, S. Coutts, M. Castiglioni, and A. Marchesi. Learning in Bayesian Stackelberg games with unknown follower's types. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026. arXiv:2602.00771.

[11] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[12] N. Cesa-Bianchi, Y. Mansour, and G. Stoltz. Improved second-order bounds for prediction with expert advice. *Machine Learning*, 66(2–3):321–352, 2007.

[13] G. Chen, S. Zhang, Y. Wang, and Y. Ji. Online learning in Stackelberg security games with adaptive attacker sequences and time-varying attack intensities. arXiv:2608.01703, 2026.

[14] V. Conitzer and T. Sandholm. Computing the optimal strategy to commit to. In *Proceedings of the 7th ACM Conference on Electronic Commerce (EC)*, pages 82–90, 2006.

[15] J. Dong, A. Roth, Z. Schutzman, B. Waggoner, and Z. S. Wu. Strategic classification from revealed preferences. In *Proceedings of the 19th ACM Conference on Economics and Computation (EC)*, 2018.

[16] F. Fang, T. H. Nguyen, R. Pickles, W. Y. Lam, G. R. Clements, B. An, A. Singh, M. Tambe, and A. Lemieux. Deploying PAWS: field optimization of the protection assistant for wildlife security. In *Proceedings of the 28th Innovative Applications of Artificial Intelligence Conference (IAAI)*, pages 3966–3973, 2016.

[17] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. John Wiley & Sons, 3 edition, 1968.

[18] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[19] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 334–343, 1997.

[20] N. Haghtalab, F. Fang, T. H. Nguyen, A. Sinha, A. D. Procaccia, and M. Tambe. Three strategies to success: learning adversary models in security games. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 308–314, 2016.

[21] N. Haghtalab, T. Lykouris, S. Nietert, and A. Wei. Learning in Stackelberg games with non-myopic agents. In *Proceedings of the 23rd ACM Conference on Economics and Computation (EC)*, 2022.

[22] K. Harris, Z. S. Wu, and M.-F. Balcan. Regret minimization in Stackelberg games with side information. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[23] P. Joulani, A. Gyorgy, and C. Szepesvari. Online learning under delayed feedback. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1453–1461, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL `https://proceedings.mlr.press/v28/joulani13.html`.

[24] A. Kalai and S. Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

[25] C. Kiekintveld, M. Jain, J. Tsai, J. Pita, F. Ordóñez, and M. Tambe. Computing optimal randomized resource allocations for massive security games. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 689–696, 2009.

[26] C. Kiekintveld, J. Marecki, and M. Tambe. Approximation methods for infinite Bayesian Stackelberg games: modeling distributional payoff uncertainty. In *Proceedings of the 10th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 1005–1012, 2011.

[27] C. Kiekintveld, T. Islam, and V. Kreinovich. Security games with interval uncertainty. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 231–238, 2013.

[28] R. Kleinberg, A. Niculescu-Mizil, and Y. Sharma. Regret bounds for sleeping experts and bandits. *Machine Learning*, 80(2–3):245–272, 2010.

[29] R. Klíma, V. Lisý, and C. Kiekintveld. Combining online learning and equilibrium computation in security games. In *Decision and Game Theory for Security (GameSec)*, volume 9406 of *Lecture Notes in Computer Science*. Springer, 2015.

[30] D. Korzhyk, V. Conitzer, and R. Parr. Complexity of computing optimal Stackelberg strategies in security resource allocation games. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 805–810, 2010.

[31] J. Letchford, V. Conitzer, and K. Munagala. Learning and approximating the optimal strategy to commit to. In *Proceedings of the 2nd International Symposium on Algorithmic Game Theory (SAGT)*, pages 250–262, 2009.

[32] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.

[33] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

[34] J. Marecki, G. Tesauro, and R. Segal. Playing repeated Stackelberg games with unknown opponents. In *Proceedings of the 11th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 821–828, 2012.

[35] J. Mourtada and O.-A. Maillard. Efficient tracking of a growing number of experts. In *Proceedings of the 28th International Conference on Algorithmic Learning Theory (ALT)*, 2017.

[36] T. H. Nguyen, R. Yang, A. Azaria, S. Kraus, and M. Tambe. Analyzing the effectiveness of adversary modeling in security games. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, pages 718–724, 2013.

[37] P. Paruchuri, J. P. Pearce, J. Marecki, M. Tambe, F. Ordóñez, and S. Kraus. Playing games for security: an efficient exact algorithm for solving Bayesian Stackelberg games. In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 895–902, 2008.

[38] B. Peng, W. Shen, P. Tang, and S. Zuo. Learning optimal strategies to commit to. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 2149–2156, 2019.

[39] G. Personnat, T. Lin, S. Hossain, and D. C. Parkes. Learning to play multi-follower Bayesian Stackelberg games. In *International Conference on Learning Representations (ICLR)*, 2026.

[40] J. Pita, M. Jain, J. Marecki, F. Ordóñez, C. Portway, M. Tambe, C. Western, P. Paruchuri, and S. Kraus. Deployed ARMOR protection: the application of a game theoretic model for security at the Los Angeles International Airport. In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), Industry Track*, pages 125–132, 2008.

[41] J. Pita, M. Jain, M. Tambe, F. Ordóñez, and S. Kraus. Robust solutions to Stackelberg games: addressing bounded rationality and limited observations in human cognition. *Artificial Intelligence*, 174(15):1142–1171, 2010.

[42] K. Quanrud and D. Khashabi. Online learning with adversarial delays. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[43] A. Roth, J. Ullman, and Z. S. Wu. Watch and learn: optimizing from revealed preferences feedback. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*, 2016.

[44] H. Shayestehmanesh, S. Azami, and N. A. Mehta. Dying experts: efficient algorithms with optimal regret bounds. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[45] E. Shieh, B. An, R. Yang, M. Tambe, C. Baldwin, J. DiRenzo, B. Maule, and G. Meyer. PROTECT: a deployed game theoretic system to protect the ports of the United States. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 13–20, 2012.

[46] M. Tambe. *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*. Cambridge University Press, 2012.

[47] J. Tsai, S. Rathi, C. Kiekintveld, F. Ordóñez, and M. Tambe. IRIS: a tool for strategic security allocation in transportation networks. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), Industry Track*, pages 37–44, 2009.

[48] M. J. Weinberger and E. Ordentlich. On delayed prediction of individual sequences. *IEEE Transactions on Information Theory*, 48(7):1959–1976, 2002.

[49] H. Xu, L. Tran-Thanh, and N. R. Jennings. Playing repeated security games with no prior knowledge. In *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 104–112, 2016.

[50] R. Yang, B. Ford, M. Tambe, and A. Lemieux. Adaptive resource allocation for wildlife protection against illegal poachers. In *Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 453–460, 2014.

[51] A. C. Yao. Probabilistic computations: Toward a unified measure of complexity. *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 222–227, 1977.

A Proofs for Section 4

A.1 Proof of Lemma 4 and Corollary 2

Lemma 4. Ties are broken toward the lowest-indexed target. For target 2 to be selected it must therefore be *strictly* better than target 1, since a tie is resolved in favor of target 1, but only *weakly* better than target 3, since a tie is resolved in favor of target 2. By Equation (16), these conditions are respectively

$$U_\alpha(2,\mathbf{p}) > U_\alpha(1,\mathbf{p}) \iff 0 > r_1 - p_1 \iff p_1 > r_1$$

and

$$U_\alpha(2,\mathbf{p}) \geq U_\alpha(3,\mathbf{p}) \iff 0 \geq (1-r_2) - p_3 \iff p_3 \geq 1 - r_2.$$

Conversely, if both conditions hold, then target 2 beats target 1 strictly and ties with or beats target 3, so lowest-index tie-breaking selects target 2. This proves Equation (18). For the second claim, the single-resource constraint $p_1 + p_2 + p_3 \leq 1$ together with $p_2 \geq 0$ gives

$$p_1 \leq 1 - p_3 \leq 1 - (1 - r_2) = r_2,$$

where the second inequality uses $p_3 \geq 1 - r_2$. Together with $p_1 > r_1$, this gives $p_1 \in (r_1, r_2]$. □

Corollary 2. Suppose two of the types, say $\alpha^{(j)}$ and $\alpha^{(j')}$, both strike target 2 under the same coverage vector $\mathbf{p}$. The second assertion of Lemma 4, applied to each of them, places the single number p_1 in the interval of $\alpha^{(j)}$ and in the interval of $\alpha^{(j')}$. Since the intervals are pairwise disjoint, this forces $j = j'$. □

A.2 Proof of Lemma 5

The proof rests on a counting argument: we bound, for each round t separately, how many of the M candidate indices can be steered onto target 2 for the very first time at that round. Since the events $\{\tau = t\}$ for different t are disjoint, summing those bounds gives the tail bound Equation (20), and a standard identity for the expectation of a nonnegative integer random variable then gives Equation (21). Write $\tau(j)$ for the value of τ when the hidden index is j.

Step 1: how many histories are possible. Fix a round t and consider the indices j with $\tau(j) = t$. For such an index the attacker struck target 1 or target 3 in each of the rounds $1, \ldots, t-1$; had it struck target 2, we would have $\tau(j) < t$. The defender's record of the past is the string $(i_1, \ldots, i_{t-1})$ of attacked targets, so at the start of round t it is in one of at most 2^{t-1} histories. Write H_t for this set of histories, so $|H_t| \leq 2^{t-1}$.

Step 2: each history determines one coverage vector. By hypothesis the coverage vector played in round t is a fixed function $\mathbf{p}_t(h)$ of the history h. In particular, two indices that produce the same history lead the defender to play the same coverage vector.

Step 3: each coverage vector steers at most one candidate. The intervals are pairwise disjoint, so by Corollary 2 at most one index j' has $b_{\alpha^{(j')}}(\mathbf{p}_t(h)) = 2$.

Step 4: combining the three. Let j be any index with $\tau(j) = t$. By Step 1 it leads the defender into some history $h \in H_t$; by Step 2 the defender then plays $\mathbf{p}_t(h)$; and by the definition of τ the attacker $\alpha^{(j)}$ strikes target 2 under it. By Step 3, at most one index does so. Hence distinct indices with $\tau = t$ arise from distinct histories, and

$$|\{j : \tau(j) = t\}| \;\leq\; |H_t| \;\leq\; 2^{t-1}.$$

Since j is uniform on $[M]$, dividing by M turns this count into a probability, $\Pr[\tau = t] \le 2^{t-1}/M$. The first few values are $\Pr[\tau = 1] \le 1/M$, $\Pr[\tau = 2] \le 2/M$ and $\Pr[\tau = 3] \le 4/M$.

Step 5: the tail bound. The events $\{\tau = t\}$, $t = 1, \dots, u$, are pairwise disjoint, so their probabilities add, and summing the geometric series $1 + 2 + \cdots + 2^{u-1} = 2^u - 1$ gives

$$\Pr[\tau \le u] \;=\; \sum_{t=1}^{u} \Pr[\tau = t] \;\le\; \frac{1}{M} \sum_{t=1}^{u} 2^{t-1} \;=\; \frac{2^u - 1}{M},$$

which is Equation (20).

Step 6: from the tail bound to the expectation. Put $X = \min\{\tau - 1, T\}$, a random variable with values in $\{0, 1, \dots, T\}$. For such a variable the tail-sum formula gives $\mathbb{E}[X] = \sum_{s=0}^{T-1} \Pr[X > s]$. Now $X > s$ holds exactly when $\tau > s + 1$, so writing $u = s + 1$,

$$\mathbb{E}[X] \;=\; \sum_{u=1}^{T} \Pr[\tau > u] \;=\; \sum_{u=1}^{T} \bigl(1 - \Pr[\tau \le u]\bigr) \;\ge\; T - \sum_{u=1}^{T} \frac{2^u - 1}{M},$$

where the inequality substitutes Equation (20). Finally $\sum_{u=1}^{T} 2^u = 2^{T+1} - 2$ and $\sum_{u=1}^{T} 1 = T$, so the subtracted sum equals $(2^{T+1} - 2 - T)/M$, which is Equation (21). □

A.3 Proof of Theorem 2

Set $M = 2^T$ and take the family $\mathfrak{U}_M \subset \mathfrak{U}$ of Equation (19), whose M intervals each have width 2^{-T}. For $j \in [M]$, the instance j is $K = \{\alpha^{(j)}\}$ with $a_t = \alpha^{(j)}$ in every round. It is admissible for every $K_{\max} \ge 1$, since $|K| = 1$, and the adversary is oblivious. Fix an online algorithm, write $\mathrm{Regret}_j(T)$ for its regret on instance j, and let $\overline{R} = \frac{1}{M} \sum_{j=1}^{M} \mathrm{Regret}_j(T)$ be the average over the instances. We show that $\overline{R}$ is at least the right-hand side of Equation (22); then some instance j has $\mathrm{Regret}_j(T) \ge \overline{R}$, which is the claim.

Part 1: the benchmark earns zero, so regret counts failed rounds. On instance j let $\mathbf{p}^* = (\frac{2j-1}{2M}, 0, 1 - \frac{2j-1}{2M})$, whose first coordinate is the midpoint of I_j. Writing $r_1 = (j-1)/M$ and $r_2 = j/M$ for the endpoints of I_j, we have $p_1^* > r_1$ and $p_3^* = 1 - p_1^* > 1 - r_2$, both strictly. By Equation (18), $b_{\alpha^{(j)}}(\mathbf{p}^*) = 2$ whatever the tie-breaking rule, so $\mathbf{p}^*$ earns 0 in every round. As $U_d \le 0$ everywhere by Equation (14), no coverage vector does better, so $\mathbf{p}^*$ is a maximizer in Equation (2) and the benchmark term of Equation (3) vanishes. Every other round is worth -1, so

$$\mathrm{Regret}_j(T) = \mathbb{E}\bigl[\#\{t \le T : \text{the defender fails at } t\}\bigr], \tag{36}$$

the expectation being over the algorithm's coins.

Part 2: deterministic algorithms. Suppose first that the algorithm is deterministic. Under partial information its only feedback is the attacked target, so its coverage vector in round t is a function of $i_1, \dots, i_{t-1}$, which is the setting of Lemma 5. Every round before the first success τ is a failure, so the count in Equation (36) is at least $\min\{\tau - 1, T\}$. Averaging over j uniform on $[M]$ and applying Equation (21) with $M = 2^T$,

$$\overline{R} \;\ge\; \mathbb{E}_j\bigl[\min\{\tau - 1, T\}\bigr] \;\ge\; T - \frac{2^{T+1} - T - 2}{2^T} \;=\; T - 2 + \frac{T+2}{2^T}.$$

Part 3: randomized algorithms. A randomized algorithm is a distribution over deterministic ones, and its coins are independent of j. The benchmark of instance j does not depend on the coins, so $\mathrm{Regret}_j(T)$ is the average over the coins of the regret of the deterministic algorithm they select, and so is $\overline{R}$. Each deterministic algorithm satisfies the bound of Part 2, hence so does their average. This is the easy direction of Yao's principle [51], and it completes the proof. □

A.4 Proof of Proposition 2

Draw $a_1, \dots, a_T$ i.i.d. uniformly from K, fix an online algorithm, and let $\overline{R}$ be its regret averaged over this draw. It suffices to show $\overline{R} \geq \frac{1}{2}\sqrt{\lfloor T/2 \rfloor}$, since then some fixed sequence has at least this regret. The two types have intervals $(0, \frac{1}{2}]$ and $(\frac{1}{2}, 1]$, which are disjoint, so by Corollary 2 any single coverage vector steers at most one of them onto target 2. Write $N_i = |\{t : a_t = \alpha^{(i)}\}|$ for the number of rounds in which type i attacks, so that $N_1 + N_2 = T$.

Benchmark. A fixed coverage vector steers at most one of the two types, so on the rounds of the other type it fails and earns -1. Its total payoff is therefore at most $-\min\{N_1, N_2\}$, and this is achieved: the vector $(\frac{1}{4}, 0, \frac{3}{4})$ has $p_1 = \frac{1}{4} \in (0, \frac{1}{2}]$ and $p_3 = \frac{3}{4} \geq 1 - \frac{1}{2}$, so by Equation (18) it steers $\alpha^{(1)}$ on all N_1 of its rounds, while $(\frac{3}{4}, 0, \frac{1}{4})$ steers $\alpha^{(2)}$ on all N_2 of its rounds. The better of the two attains $-\min\{N_1, N_2\}$, so the benchmark of Equation (2) has value exactly $-\min\{N_1, N_2\}$.

Algorithm. Fix a round t and condition on the history up to that round. The history is a function of $a_1, \dots, a_{t-1}$ and of the algorithm's own coins, and the sequence is drawn i.i.d., so the history carries no information about a_t; in particular the coverage vector $\mathbf{p}_t$, which is determined by the history, is independent of a_t. By Corollary 2, $\mathbf{p}_t$ steers at most one of the two types, and a_t equals that type with probability exactly $\frac{1}{2}$. Hence the probability of success in round t is at most $\frac{1}{2}$ conditionally on any history, and therefore unconditionally. By linearity of expectation over the T rounds, the algorithm's expected payoff is at most $-T/2$.

Combining. Subtracting the algorithm's expected payoff from the benchmark's gives

$$\overline{R} \;\geq\; \tfrac{T}{2} - \mathbb{E}[\min\{N_1, N_2\}].$$

The identity $\min\{a, b\} = \frac{1}{2}((a + b) - |a - b|)$ with $a = N_1$, $b = N_2$ and $N_1 + N_2 = T$ gives $\min\{N_1, N_2\} = \frac{1}{2}(T - |N_1 - N_2|)$, so

$$\overline{R} \;\geq\; \tfrac{1}{2}\mathbb{E}\big[|N_1 - N_2|\big].$$

Each a_t is an independent fair coin flip between the two types, so $N_1 - N_2$ is the position after T steps of a simple symmetric random walk on the integers, and Lemma 8 below gives $\mathbb{E}|N_1 - N_2| \geq \sqrt{\lfloor T/2 \rfloor}$, which proves Equation (23). □

The bound is tight in a strong sense: the success probability $\frac{1}{2}$ is attained by any algorithm that always plays inside one of the two intervals, so all of them, follow the leader and an algorithm simply told K alike, have regret exactly $\frac{1}{2}\mathbb{E}|N_1 - N_2| = \sqrt{T/(2\pi)}\,(1 + o(1))$ on average over the draw, the value Figure 7(b) measures.

Lemma 8 (random walk)**.** *Let $S_T = \sum_{t \leq T} \varepsilon_t$ with $\varepsilon_t \in \{-1, +1\}$ independent and uniform. Then $\mathbb{E}|S_T| \geq \sqrt{\lfloor T/2 \rfloor}$.*

Proof. Reduction to even length. Since $|\cdot|$ is convex and S_m is a martingale, $|S_m|$ is a submartingale, so $\mathbb{E}|S_m|$ is nondecreasing in m. With $\nu = \lfloor T/2 \rfloor$ we have $2\nu \leq T$, hence $\mathbb{E}|S_T| \geq \mathbb{E}|S_{2\nu}|$.

Closed form. For a walk of 2ν steps the classical identity $\mathbb{E}|S_{2\nu}| = 2\nu\binom{2\nu}{\nu}4^{-\nu}$ holds [17], so it suffices to show

$$\binom{2\nu}{\nu}4^{-\nu} \;\geq\; \frac{1}{2\sqrt{\nu}}. \tag{37}$$

Central binomial bound. We prove Equation (37) by induction on ν. At $\nu = 1$ both sides equal $1/2$. Passing from ν to $\nu + 1$ multiplies the left-hand side by $(2\nu + 1)/(2\nu + 2)$ and the right-hand side by $\sqrt{\nu/(\nu + 1)}$, so the inductive step holds provided the first ratio is at least the second. Both are positive, so squaring and cross-multiplying, this is $(2\nu + 1)^2(\nu + 1) \geq \nu(2\nu + 2)^2$, that is, $4\nu^3 + 8\nu^2 + 5\nu + 1 \geq 4\nu^3 + 8\nu^2 + 4\nu$, which reduces to $\nu + 1 \geq 0$. Hence Equation (37) holds for all $\nu \geq 1$, and $\mathbb{E}|S_T| \geq 2\nu \cdot \frac{1}{2\sqrt{\nu}} = \sqrt{\nu} = \sqrt{\lfloor T/2 \rfloor}$. □

B Proofs for Section 5

B.1 The partial-information learner of Balcan et al. [3]

The proof of Lemma 7 needs more than the statement of Theorem 3: it reopens the argument behind it. This subsection describes the learner $\mathcal{A}^{\mathrm{pi}}(C)$ of Balcan et al. [3, Algorithm 1] and the two properties of it that the argument uses. Everything here is due to Balcan et al. [3, Sec. 6]; we restate it in our notation, with the catalog C in the role of their known type set, a horizon $H \le T$ known in advance, and $\epsilon = 1/T$ as in Algorithm 2. Throughout, the attacker sequence is oblivious and takes values in C, and we write $u_t(\mathbf{p}) = U_d(b_{a_t}(\mathbf{p}), \mathbf{p}) \in [-1, 1]$ for the payoff that $\mathbf{p}$ would earn at round t.

Experts and the obstacle. Like Algorithm 1, the learner competes only against the finite set $\mathcal{E}(C;\epsilon)$ of Lemma 1, which on any $H \le T$ rounds loses at most $2\epsilon H \le 2$ against the best $\mathbf{p} \in \mathcal{P}$. Under full information one would run the expert algorithm of Proposition 1 over $\mathcal{E}(C;\epsilon)$, charging every expert $\mathbf{p}$ the loss $-u_t(\mathbf{p})$ at every round. Under partial information the defender observes only the attacked target i_t, and $u_t(\mathbf{p})$ for $\mathbf{p} \neq \mathbf{p}_t$ depends on a_t, so these losses are not available. Balcan et al. [3] instead give the expert algorithm one *estimated* loss vector per block of rounds. We call it the *background learner*.

Estimation blocks. The horizon $[1, H]$ is split into Z consecutive *estimation blocks* $B_1, \dots, B_Z$ of equal length $G = H/Z$; these are unrelated to the revelation blocks $\mathcal{B}_\nu$. At the start of B_τ the background learner holds a distribution q_τ over $\mathcal{E}(C;\epsilon)$, with q_1 uniform. Inside B_τ, $d \le |C|$ rounds chosen uniformly at random are *exploration* rounds, on which fixed probing strategies, described below, are played. On every other round, an *exploitation* round, the learner draws $\mathbf{p}_t \sim q_\tau$ and plays it. At the end of the block the exploration rounds yield, for every $\mathbf{p} \in \mathcal{E}(C;\epsilon)$, an estimate $\hat{c}_\tau(\mathbf{p})$ of the block's average payoff

$$c_\tau(\mathbf{p}) \;=\; \frac{1}{G} \sum_{t \in B_\tau} u_t(\mathbf{p}),$$

and the background learner receives the loss vector $-\hat{c}_\tau$ as its τ-th round and returns $q_{\tau+1}$. The number of blocks is [3, Algorithm 1]

$$Z \;=\; n\big(H^2 \log(n|C|)\big)^{1/3}, \tag{38}$$

rounded to an integer. It is chosen to balance the two error terms of Equation (42) below, and it is the only place where the learner uses H.

The estimates. Few probes suffice because, over a block, the payoff of every expert depends on the attackers only through how often each type attacked. Let $m_\tau \in \mathbb{N}^C$ count the rounds of B_τ at which each type of C attacked. Take $\mathbf{p} \in \mathcal{P}_\sigma(C)$ (Equation (4)), so that under $\mathbf{p}$ each type α attacks $\sigma(\alpha)$, and let $w_{\sigma,i} \in \{0,1\}^C$ mark the types α with $\sigma(\alpha) = i$. Then $w_{\sigma,i} \cdot m_\tau$ is the number of rounds of B_τ at which $\mathbf{p}$ would be answered by an attack on i, so

$$c_\tau(\mathbf{p}) \;=\; \frac{1}{G} \sum_{i \in N} U_d(i, \mathbf{p})\; w_{\sigma,i} \cdot m_\tau. \tag{39}$$

A single probe estimates a single term $w_{\sigma,i} \cdot m_\tau / G$: play a fixed $\mathbf{p}_w \in \mathcal{P}_\sigma(C)$ at a uniformly random round of B_τ and record $\hat{p}_\tau(w) = 1$ if target i is attacked and 0 otherwise. Under $\mathbf{p}_w$ the attacker

strikes i exactly when it is marked by $w_{\sigma,i}$, and since the sequence is fixed in advance and the round is uniform, $\mathbb{E}[\hat{p}_\tau(w)] = w_{\sigma,i} \cdot m_\tau / G$ [3, Lemma 6.3]. The map $w \mapsto w \cdot m_\tau$ is linear, so it suffices to probe a basis of the span of the finite set $\mathcal{W} = \{w_{\sigma,i}\}$, whose dimension is $d \le |C|$. To keep the estimates bounded, Balcan et al. [3] probe a *barycentric spanner* [2]: a basis $b_1, \dots, b_d \in \mathcal{W}$ such that every $w \in \mathcal{W}$ equals $\sum_j \lambda_j(w)\, b_j$ with all $\lambda_j(w) \in [-1, 1]$. Substituting into Equation (39) gives the estimate [3, Eq. 2]

$$\hat{c}_\tau(\mathbf{p}) \;=\; \sum_{i \in N} U_d(i, \mathbf{p}) \sum_{j=1}^{d} \lambda_j(w_{\sigma,i})\, \hat{p}_\tau(b_j), \tag{40}$$

which has the two properties the argument uses [3, Lemma 6.4]:

$$\mathbb{E}[\hat{c}_\tau(\mathbf{p})] = c_\tau(\mathbf{p}) \qquad \text{and} \qquad |\hat{c}_\tau(\mathbf{p})| \le \kappa \coloneqq n|C|. \tag{41}$$

The first follows from linearity, the second from $|U_d| \le 1$, $|\lambda_j| \le 1$, $\hat{p}_\tau \in \{0, 1\}$ and $d \le |C|$. The exploration rounds of B_τ are drawn afresh, independently of earlier blocks, so the first property also holds conditionally on q_τ.

B.2 Proof of Lemma 7

By Appendix B.1, $\mathcal{A}^{\mathrm{pi}}(C)$ uses its horizon H only through the number of estimation blocks Z of Equation (38), chosen to balance exploration against exploitation at that particular H. Inside an epoch the horizon is the epoch length, which is unknown when the learner starts, so Z cannot be set. This difficulty differs from the one addressed in full information feedback: there the only horizon-dependent quantity was a learning rate, which can be set without knowing the horizon. Here the horizon fixes how often the algorithm explores, and no choice of background learner makes that horizon-free, so we keep $\mathcal{A}^{\mathrm{pi}}(C)$ and wrap it. Two ingredients suffice: (i) the bound of Theorem 3, though stated for a run tuned to a horizon H, in fact bounds the regret at every prefix of that run, not only at its end; (ii) a doubling wrapper removes the tuning.

(i) The horizon-tuned bound is also a prefix bound. Fix $H \le T$ and run $\mathcal{A}^{\mathrm{pi}}(C)$ tuned to it. Fix $\tau_0 \in \{1, \dots, Z\}$; the first τ_0 blocks are the rounds $t \le \tau_0 G$. We follow the chain of inequalities in the proof of Balcan et al. [3, Lemma 6.2], but stop it after τ_0 blocks instead of Z. It has three links.

Exploration. On an exploitation round of B_τ the expected payoff given q_τ is $q_\tau \cdot u_t$, where $q_\tau \cdot u_t = \sum_{\mathbf{p}} q_\tau(\mathbf{p}) u_t(\mathbf{p})$. The d exploration rounds earn some other payoff, and since all payoffs lie in $[-1, 1]$, replacing them by $q_\tau \cdot u_t$ changes the block's total by at most $2d$. Hence the expected payoff on B_τ is at least $G\,\mathbb{E}[q_\tau \cdot c_\tau] - 2d$.

Background learner. By Equation (41), conditionally on q_τ, $\mathbb{E}[q_\tau \cdot \hat{c}_\tau] = q_\tau \cdot c_\tau$. After τ_0 blocks the background learner has received the loss vectors $-\hat{c}_1, \dots, -\hat{c}_{\tau_0}$, with entries in $[-\kappa, \kappa]$, so by Proposition 1, which holds at every round without knowing the number of rounds in advance, it trails the best expert on the estimates by at most $R(\tau_0) = O\big(\sqrt{\tau_0\, \kappa \log |\mathcal{E}(C; \epsilon)|}\big)$.

Back to true payoffs. The expectation of a maximum is at least the maximum of the expectations, and by Equation (41) the latter is the maximum of the true block averages.

Chaining the three links, multiplying by the block length G, and finally passing from $\mathcal{E}(C; \epsilon)$ to

$\mathcal{P}$ by Lemma 1:

$$\begin{aligned}
\mathbb{E}\Big[\sum_{t\le\tau_0 G} u_t(\mathbf{p}_t)\Big] &\ge G\sum_{\tau\le\tau_0}\mathbb{E}[q_\tau\cdot c_\tau] - 2d\tau_0\\
&= G\,\mathbb{E}\Big[\sum_{\tau\le\tau_0} q_\tau\cdot\hat{c}_\tau\Big] - 2d\tau_0\\
&\ge G\,\mathbb{E}\Big[\max_{\mathbf{p}\in\mathcal{E}(C;\epsilon)}\sum_{\tau\le\tau_0}\hat{c}_\tau(\mathbf{p})\Big] - G\,R(\tau_0) - 2d\tau_0\\
&\ge \max_{\mathbf{p}\in\mathcal{E}(C;\epsilon)}\sum_{t\le\tau_0 G} u_t(\mathbf{p}) - G\,R(\tau_0) - 2d\tau_0\\
&\ge \max_{\mathbf{p}\in\mathcal{P}}\sum_{t\le\tau_0 G} u_t(\mathbf{p}) - 2 - \underbrace{G\,R(\tau_0)}_{\text{background learner}} - \underbrace{2d\tau_0}_{\text{exploration}}. \qquad (42)
\end{aligned}$$

So the regret over the first τ_0 blocks is at most 2 plus the two error terms marked in Equation (42). Both are nondecreasing in τ_0, so for every $\tau_0 \le Z$ they are at most their values at $\tau_0 = Z$, which is where Balcan et al. [3] evaluate the chain. There, with $d \le |C|$, $\kappa = n|C|$, $\log|\mathcal{E}(C;\epsilon)| = O(n^2|C|\log(n|C|))$ by Lemma 2, and Z as in Equation (38),

$$G\,R(Z) = O\Big(H\sqrt{\kappa\log|\mathcal{E}(C;\epsilon)|/Z}\Big) \quad\text{and}\quad 2dZ \le 2|C|Z$$

are both $O(H^{2/3}n|C|\log^{1/3}(n|C|))$: this is the bound of Theorem 3 with horizon H and type set C.

Two cases remain. A prefix $L \le H$ that ends inside a block adds at most G rounds after the last complete block, each contributing at most 2 to the regret, and $2G = 2H^{1/3}/(n\log^{1/3}(n|C|))$ is of lower order. And if H is so small that a block cannot hold its d exploration rounds, that is, $G \le |C|$, then $H^{1/3} \le n|C|\log^{1/3}(n|C|)$, so the trivial bound $2H$ on the regret is already $O(H^{2/3}n|C|\log^{1/3}(n|C|))$. We conclude that there is an absolute constant c_0 with

$$\text{regret at prefix } L \;\le\; c_0\,H^{2/3}\,n|C|\log^{1/3}(n|C|) \qquad (43)$$

for every $L \le H$. The right-hand side depends on H, not L: at short prefixes we still pay for the horizon we tuned to. Converting $H^{2/3}$ into $L^{2/3}$ is the task of part (ii).

(ii) Doubling wrapper. Run $\mathcal{A}^{\mathrm{pi}}(C)$ in *stages* of exponentially growing length: stage $s = 0, 1, 2, \ldots$ has length 2^s and is a fresh copy of $\mathcal{A}^{\mathrm{pi}}(C)$ tuned to horizon 2^s; at the end of stage s we discard its state and start stage $s+1$. This variant needs no horizon in advance, and the sequence restricted to any stage is still oblivious. This is the doubling trick of Auer et al. [1] [see also 11, Sec. 2.3], and the computation below is that of Besson and Kaufmann [7, Theorem 4] at $\gamma = 2/3$; what it requires from the base learner is a bound on a stage that is cut off early, which part (i) provides. Fix any $L \le T$ and let S be the stage containing round L; stages $0, \ldots, S-1$ occupy $2^S - 1$ rounds, all before L, so $2^S \le L$. Stage S is in general cut off at L before its end, and this is why (i) is needed: Equation (43) bounds the regret of a run at every prefix, not only at the horizon it was tuned to.

Regret is subadditive across stages: for any $\mathbf{p} \in \mathcal{P}$ the payoff over $[1, L]$ decomposes as a sum of per-stage payoffs, and the maximum over $\mathcal{P}$ of a sum is at most the sum of the per-stage maxima. Subtracting the algorithm's payoff stage by stage,

$$\text{regret over } [1,L] \;\le\; \sum_{s=0}^{S}\text{regret of stage } s \;\le\; c_0\,\phi\sum_{s=0}^{S} 2^{2s/3},$$

where $\phi \coloneqq n|C|\log^{1/3}(n|C|)$ and each stage's regret comes from Equation (43) with $H = 2^s$. The remaining sum is a geometric series with ratio $2^{2/3} > 1$, so summing it from the top downwards,

$$\sum_{s=0}^{S} 2^{2s/3} \;\le\; \frac{2^{2/3}}{2^{2/3}-1}\cdot 2^{2S/3} \;\le\; \frac{2^{2/3}}{2^{2/3}-1}\cdot L^{2/3},$$

the last step using $2^S \le L$. Setting $c_4 \coloneqq 2^{2/3}c_0/(2^{2/3}-1) < 2.71\,c_0$ yields Equation (26). □

B.3 Proof of Theorem 5

The proof sketch after Theorem 5 rests on two devices.

Nested intervals, so that one strategy suits every type. By Equation (17), a coverage vector $(p, 0, 1-p)$ steers the type with interval J precisely when $p \in J$. So if the adversary chooses $K_{\max}$ types whose intervals are strictly nested, $J_1 \supset J_2 \supset \cdots \supset J_{K_{\max}}$, then any p in the innermost interval lies in all of them and therefore steers every type at once. The benchmark thus has value 0 and, as in Theorem 2, regret counts failed rounds.

Fresh randomness, so that each block is a new search. The type $\alpha^{[j]}$ attacks only during the j-th block, so the oracle names it only at that block's closing checkpoint. Given J_{j-1}, the next interval J_j is a uniformly chosen one of the $M = 2^B$ equal sub-intervals of J_{j-1}. When block j begins, the defender knows J_{j-1} from the previous checkpoint, but nothing about where inside J_{j-1} the interval J_j lies. It must locate J_j using the B rounds of that block, which is the search problem of Lemma 5 with $M = 2^B$ candidates and horizon B.

Construction. Put $M = 2^B$, draw $k_1, \dots, k_{K_{\max}}$ independently and uniformly from $[M]$, and define nested intervals $J_0 = (0,1] \supset J_1 \supset \cdots$ by letting J_j be the k_j-th of the M equal sub-intervals of J_{j-1}:

$$J_j = \Big(\ell_{j-1} + \tfrac{k_j-1}{M}|J_{j-1}|,\;\; \ell_{j-1} + \tfrac{k_j}{M}|J_{j-1}|\Big], \tag{44}$$

ℓ_{j-1} and $|J_{j-1}|$ being the left endpoint and length of J_{j-1}. Let $\alpha^{[j]}$ be the threshold type with interval J_j, put $K = \{\alpha^{[1]}, \dots, \alpha^{[K_{\max}]}\}$, and let $a_t = \alpha^{[j]}$ for $t \in \mathcal{B}_j$ and $a_t = \alpha^{[K_{\max}]}$ for $t > K_{\max}B$. The sequence is fixed once the k_j are drawn, so the adversary is oblivious, and each J_j is a *proper* sub-interval of J_{j-1}, so the $K_{\max}$ types are distinct. Fix an online algorithm and let $\overline{R}$ be its regret averaged over the draw of $k_1, \dots, k_{K_{\max}}$. As in the proof of Theorem 2, we first bound $\overline{R}$ for deterministic algorithms and then average over the coins of a randomized one.

The benchmark is 0. Let μ be the midpoint of the innermost interval $J_{K_{\max}}$ and $\mathbf{p}^* = (\mu, 0, 1-\mu)$. Write $J_j = (r_1^{[j]}, r_2^{[j]}]$. Because the intervals are nested, each contains $J_{K_{\max}}$, so their left endpoints increase and their right endpoints decrease with j. Hence $r_1^{[j]} \le r_1^{[K_{\max}]} < \mu$ and $1 - r_2^{[j]} \le 1 - r_2^{[K_{\max}]} < 1 - \mu$ for every j, the strict inequalities holding because μ lies strictly inside $J_{K_{\max}}$. Both conditions of Equation (18) are therefore met for every j, and strictly, so $\mathbf{p}^*$ steers every type regardless of the tie-breaking rule, and regret counts failed rounds.

Each block is a fresh search. Let the algorithm be deterministic. When $\mathcal{B}_j$ begins, the checkpoint closing $\mathcal{B}_{j-1}$ has revealed $\alpha^{[1]}, \dots, \alpha^{[j-1]}$, hence J_{j-1}, and nothing about $\alpha^{[j]}$, which has not yet attacked. Moreover the whole transcript of $\mathcal{B}_1, \dots, \mathcal{B}_{j-1}$ is a function of $k_1, \dots, k_{j-1}$ alone: this follows by induction on the rounds, since each play is determined by earlier observations, each observation by that play and the interval then attacking, and the intervals active in those blocks by $k_1, \dots, k_{j-1}$. As k_j is drawn independently of $k_1, \dots, k_{j-1}$, it remains uniform on $[M]$ conditionally on everything the defender has seen and been told before block j starts.

Condition on that transcript. Within $\mathcal{B}_j$ the defender's plays are functions of the targets attacked inside $\mathcal{B}_j$, the only attacker is $\alpha^{[j]}$, and its interval is one of M disjoint sub-intervals of J_{j-1}. The

oracle says nothing during $\mathcal{B}_j$, so Lemma 5 applies with these M candidates and horizon B, and gives $\mathbb{E}[\#\text{failures in } \mathcal{B}_j] \geq B - (2^{B+1} - B - 2)/2^B > B - 2$. The $K_{\max}$ blocks are disjoint sets of rounds, so their failure counts add, and summing over $j \in [K_{\max}]$ gives $\overline{R} \geq K_{\max}(B-2)$ for every deterministic algorithm. A randomized algorithm is a distribution over deterministic ones whose coins are independent of the draw, so averaging over its coins gives the same bound. Finally, since $\overline{R}$ is an average over the draw, some fixed draw of $k_1, \ldots, k_{K_{\max}}$ has regret at least $K_{\max}(B-2)$, and that fixed draw is the instance in the statement of Theorem 5. □

Comparing the two bounds. Theorems 4 and 5 bracket the cost of batching between $K_{\max}(B-2)$ and $2K_{\max}B$, so it is $\Theta(K_{\max}B)$ for $B \geq 3$; for $B \leq 2$ the lower bound is vacuous. The comparison needs one caveat: Theorem 5 bounds the *total* regret on its instance, whereas the $2K_{\max}B$ of Theorem 4 is only the batching part of an upper bound whose other part is the learning term. Take the random instance above with $T = K_{\max}B$, so that $|K'| = K_{\max}$ and $T^{2/3} = K_{\max}^{2/3}B^{2/3}$. Theorem 4 holds on every instance, hence also on average over the draw, and together with the bound $\overline{R} \geq K_{\max}(B-2)$ just proved it gives

$$K_{\max}(B-2) \;\leq\; \overline{R} \;\leq\; \underbrace{c\,nK_{\max}^2 B^{2/3}\log^{1/3}(nK_{\max})}_{\text{learning term}} + 2K_{\max}B.$$

Fix $\delta > 0$. The learning term is at most $\delta \cdot 2K_{\max}B$ exactly when $B \geq (c/2\delta)^3\, n^3 K_{\max}^3 \log(nK_{\max})$. For such B, and $B \geq 3$, the upper bound is at most $(1+\delta)\,2K_{\max}B$, so the upper and lower bounds differ by a factor of at most

$$\frac{(1+\delta)\,2K_{\max}B}{K_{\max}(B-2)} \;=\; (1+\delta)\Big(2 + \frac{4}{B-2}\Big),$$

which equals $6(1+\delta)$ at $B = 3$ and $4(1+\delta)$ at $B = 4$, and decreases to $2(1+\delta)$ as $B \to \infty$. At the other extreme, $B = 1$, the additive term is $2K_{\max}$ and Equation (27) becomes $O(nK_{\max}^{4/3}T^{2/3}\log^{1/3}(nK_{\max}) + K_{\max})$.

C Experiments

We test Algorithms 1 and 2 against the bounds of Theorems 1 and 4, and we play the game G_3 exactly to check that the lower bounds of Theorems 2 and 5 and Proposition 2 are attained.

C.1 Setup

Besides Algorithm 1, we run the known-type learner of Balcan et al. [3, Sec. 5], which is given K before the first round, and a *uniform* baseline that plays $(R/n, \ldots, R/n)$ in every round. Unless stated otherwise, $n = 3$, $R = 1$, and the utilities are drawn independently per target with fixed signs, $u_\alpha^c(i), u_d^u(i) \sim \text{Unif}[-1, -0.1]$ and $u_\alpha^u(i), u_d^c(i) \sim \text{Unif}[0.1, 1]$, so that coverage deters every attacker. At $n = 3$ the expert set $\mathcal{E}(C;\epsilon)$ is computed by enumeration. At $n = 10$ and $n = 20$ this is infeasible, and both algorithms instead use the optimization oracle of Appendix D, which returns the best coverage vector for a given linear objective without building $\mathcal{E}(C;\epsilon)$.

We use two adversaries. The *staggered-uniform* adversary splits the horizon into $K_{\max}$ phases of equal length and in phase i draws the attacker uniformly from the first i types, so that new types arrive at known times while old and new types keep mixing. The *uniform-random* adversary draws every attacker uniformly from K. The staggered adversary is non-stationary, so an algorithm that

tracks the current phase can beat the best fixed coverage vector, and cumulative regret against $\mathbf{p}^*$ can be negative (at $n = 10$ it was on 14 of the 40 seeds). We therefore use it only to show the epoch structure, and the stationary uniform-random adversary to measure cumulative regret.

Algorithm 2 needs estimation blocks long enough to hold $|C|$ exploration rounds. The block length H/Z implied by Equation (38) is asymptotic and falls below one round at $n = 10$, $|C| = 4$, $T = 2000$, so we use blocks of $\max\{|C| + 1, \lceil T^{1/3} \rceil\}$ rounds, which has the same order in T. At $T = 2000$ and $K_{\max} = 4$ that is 13 rounds, of which 4 explore.

At $n = 3$ the inner learner is Hedge with rate $\eta_t = \sqrt{\ln |\mathcal{E}(C;\epsilon)|/t}$, and in Figure 6 $B = \max\{4, \text{round}(T^{2/3}/4)\}$ except in panel (c).

C.2 Full information

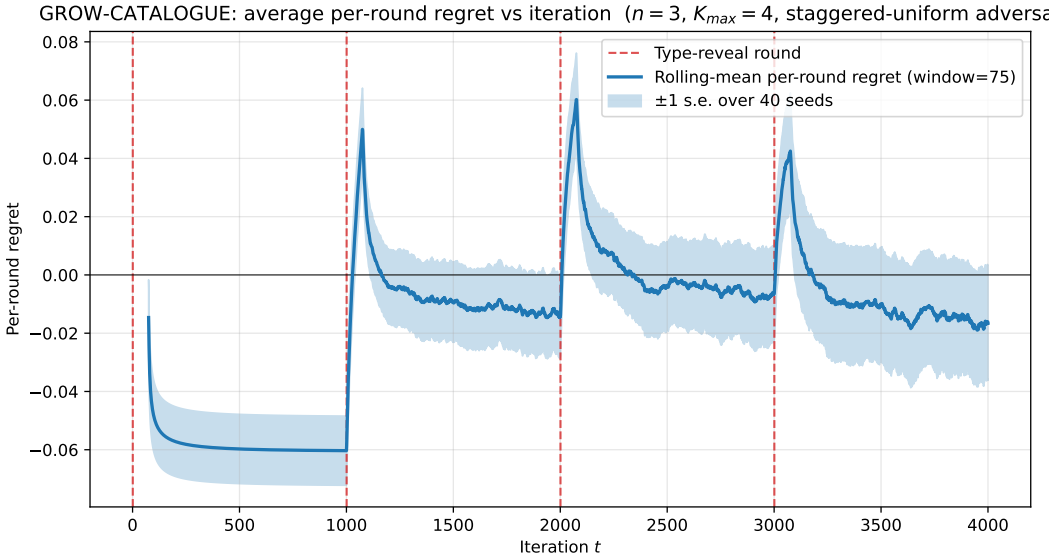


Figure 2: Rolling-window per-round regret of Grow-Catalog under the staggered-uniform adversary ($n = 3$, $K_{\max} = 4$, $T = 4000$, 40 seeds, window 75). Dashed lines mark the discovery rounds $0, 1000, 2000, 3000$.

Algorithm	Regret	Ratio
Known-type learner of Balcan et al. [3]	21.95 [16.67, 24.84]	1.00
Grow-Catalog	22.71 [18.29, 25.65]	1.03
Uniform baseline	519.31 [385.00, 716.73]	23.66

Table 2: Regret at $T = 4000$, $n = 3$, $K_{\max} = 3$, uniform-random adversary, median [interquartile range] over 40 seeds.

Figure 2 shows the epoch structure of the proof of Theorem 1: each discovery round produces a bounded spike in per-round regret, which decays as the restarted learner converges within the new epoch. Figure 3 shows the cumulative regret staying below the $C\sqrt{T}$ envelope at every horizon tested, with $\text{Regret}(T)/\sqrt{T}$ decreasing, so the stochastic adversary is far from the worst case. Table 2 shows that Grow-Catalog comes within 3% of the learner that is given K in advance, and is an order of magnitude better than the uniform baseline.

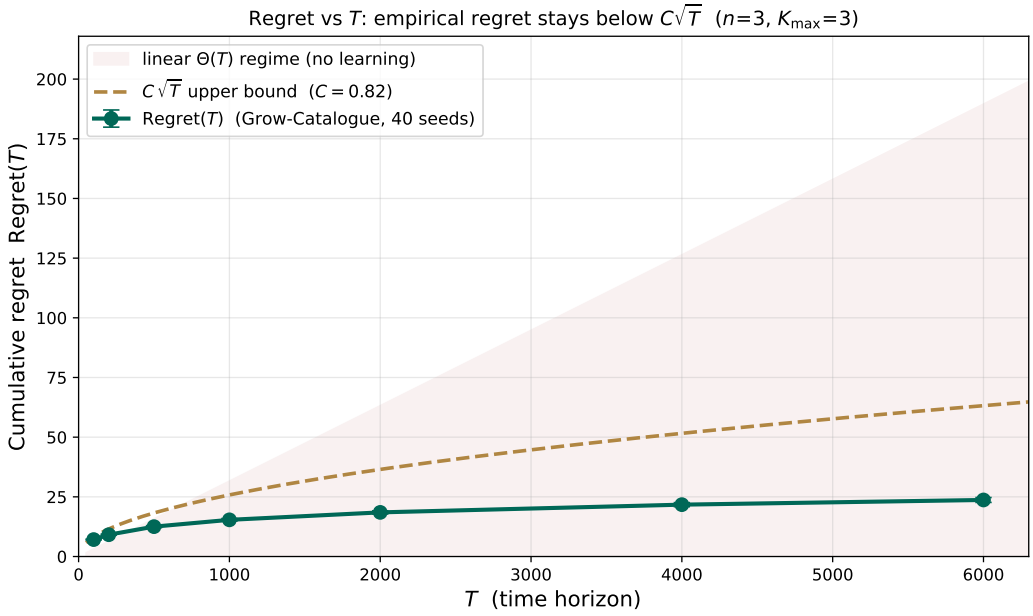


Figure 3: Cumulative regret of GROW-CATALOG against the horizon T ($n = 3$, $K_{\max} = 3$, uniform-random adversary, 40 seeds), with the $C\sqrt{T}$ envelope of Theorem 1.

Figure 5 uses the adversary on which the proof of Theorem 1 is tight: $K_{\max}$ equal epochs, each attacked only by its new type. Against the best fixed strategy over the whole horizon the regret is negative (-1032 at $K_{\max} = 4$, $T = 8000$), since no single coverage vector suits every type, so we report the sum over epochs of the regret against each epoch's own best strategy, which is what the proof bounds. It grows far more slowly than $\sqrt{K_{\max}T}$: like $T^{0.08}$ to $T^{0.21}$, and slightly faster than linearly in $K_{\max}$, as each epoch restarts Hedge on a larger expert set, reaching 57 at $K_{\max} = 6$, $T = 8000$, where $\sqrt{K_{\max}T} \approx 219$. With one type per epoch the best response never changes, so Hedge locks on within a few rounds, and the bound is not attained.

Figure 4 repeats both experiments at $n = 10$ and $n = 20$. The per-round regret again spikes after each phase boundary and decays within the following epoch, and the cumulative regret stays below the $C\sqrt{t}$ envelope, with $C = 1.20$ at $n = 10$ and $C = 1.64$ at $n = 20$, while being 14.6 and 10.1 times smaller than that of the uniform baseline.

C.3 Block revelation

In Figure 6 the simulator passes Algorithm 2 only the attacked target in each round, and at the end of each revelation block only the set of types seen so far, which the algorithm reads through membership tests, so no order or frequency information can leak. Panels (a) and (b) use the staggered-uniform adversary, which spreads the discovery blocks over the horizon and so produces the $\Theta(K_{\max})$ epochs that the proof of Theorem 4 sums over; the regret stays below the $Ct^{2/3}$ envelope, with $C = 1.38$ at $n = 10$ and $C = 1.33$ at $n = 20$. Panel (c) sweeps B over six doublings: the regret grows linearly in B, as Theorems 4 and 5 predict, with fitted slope 0.38, well below the worst case $2|K'| = 8$, which only the nested instance of Theorem 5 attains. Panel (d) compares the two feedback models on the uniform-random adversary over $T \in \{250, 500, 1000, 2000\}$: the fitted exponents are 0.695 (95% CI $[0.683, 0.706]$, bootstrapped over seeds) for block revelation, close to the 2/3 of Theorem 4, and 0.264 ($[0.230, 0.295]$) for full information, inside the 1/2 of Theorem 1.

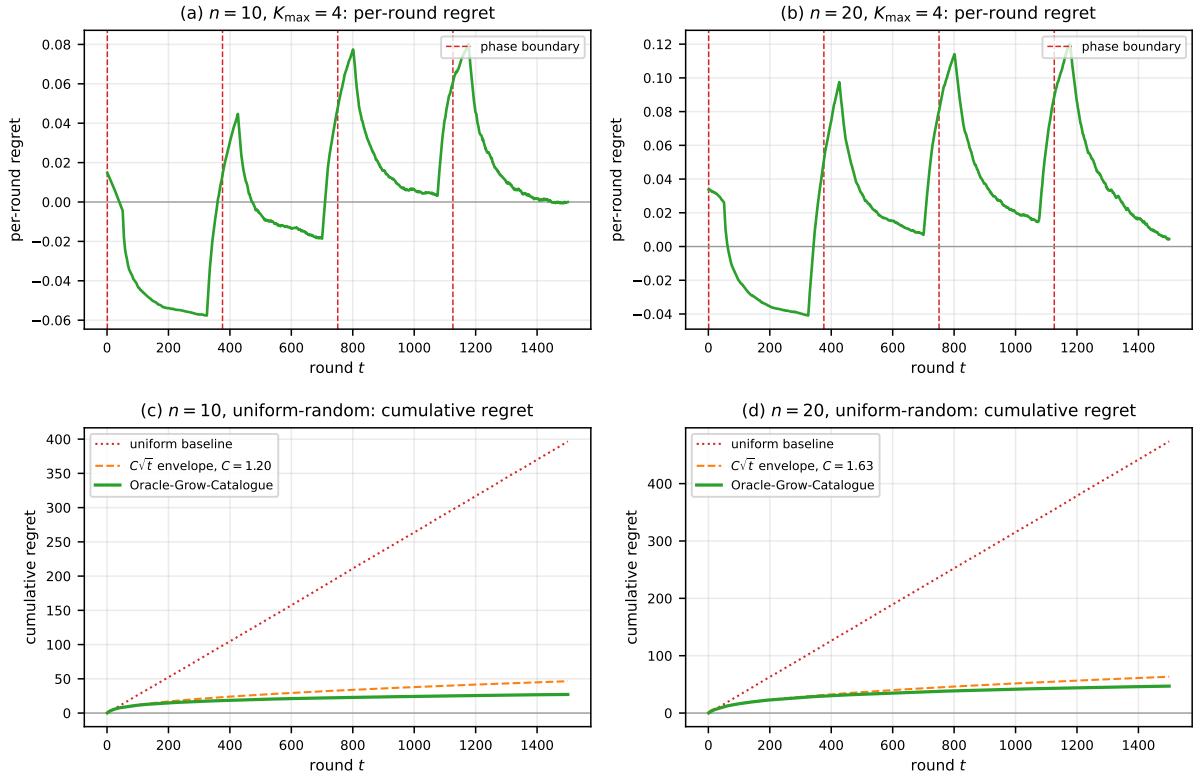


Figure 4: GROW-CATALOG with the oracle of Appendix D, $K_{\max} = 4$, $T = 1500$, 40 seeds. (a), (b) per-round regret under the staggered-uniform adversary, rolling window 101, dashed lines at the phase boundaries. (c), (d) cumulative regret under the uniform-random adversary, with the $C\sqrt{t}$ envelope of Theorem 1 and the uniform baseline.

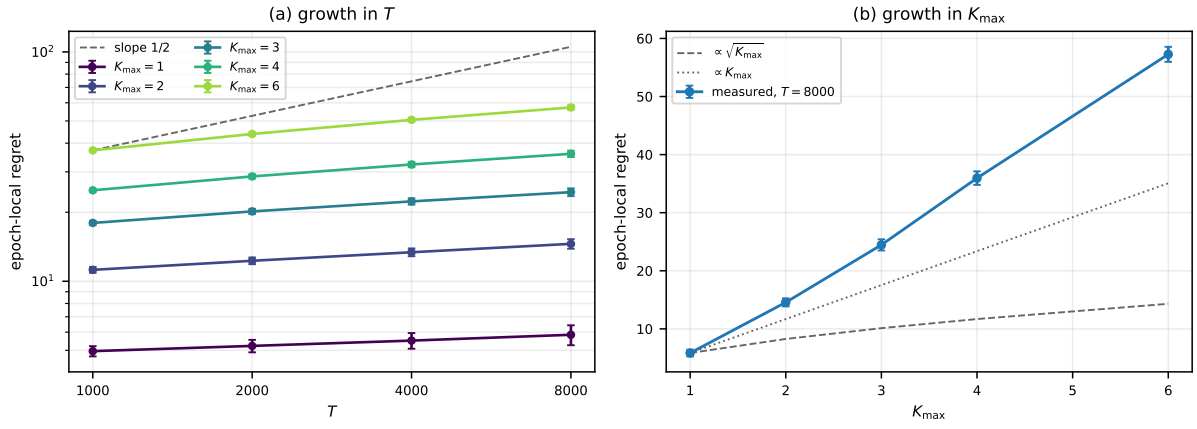


Figure 5: The hard adversary: $K_{\max}$ equal epochs, each attacked only by its new type ($n = 3$, 40 seeds). (a) epoch-local regret against T, with a slope-1/2 reference; (b) epoch-local regret against $K_{\max}$ at $T = 8000$.

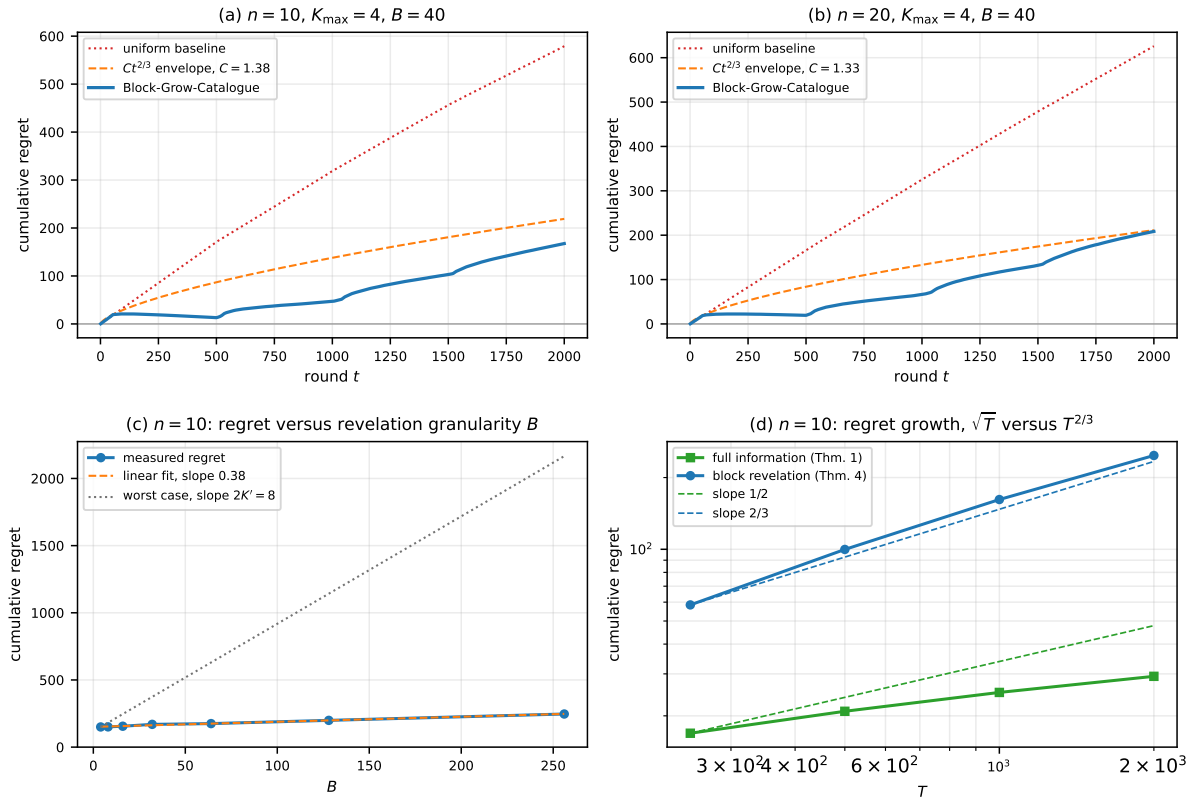


Figure 6: Block-Grow-Catalog under partial information with block revelation, $K_{\max} = 4$, $T = 2000$, 40 seeds. (a), (b) cumulative regret under the staggered-uniform adversary against the $Ct^{2/3}$ envelope of Theorem 4. (c) final regret against the revelation length B, with a least-squares fit and the worst-case slope $2|K'|$. (d) full information against block revelation as T grows, uniform-random adversary.

The difference has a simple source: Algorithm 2 spends $|C|$ exploration rounds in every estimation block whatever the sequence, about 31% of all rounds at $T = 2000$, whereas Algorithm 1 has no such floor.

C.4 The lower bounds

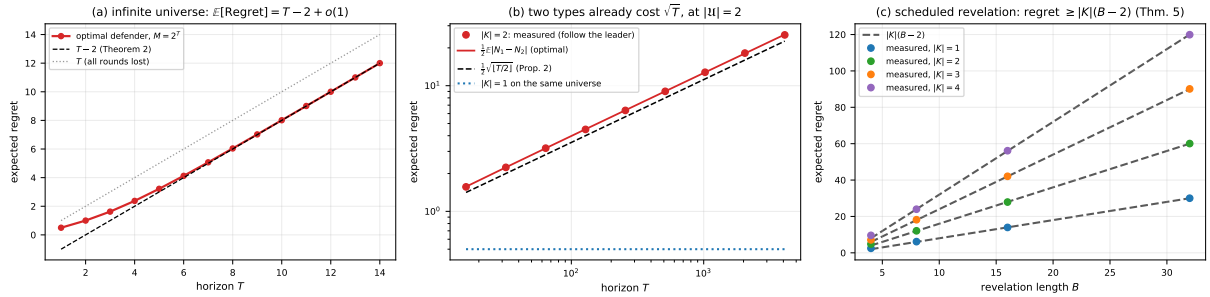


Figure 7: The game G_3 in simulation, against the optimal defender. (a) Theorem 2: with $|\mathfrak{U}_M| = 2^T$ the regret is $T - 2 + o(1)$. (b) Proposition 2: two types cost $\Theta(\sqrt{T})$, one type costs 0.5 at every horizon; logarithmic vertical axis. (c) Theorem 5: on the nested instance the regret matches $|K|(B-2)$ (dashed lines).

Figure 7 plays G_3 exactly: types are built from the utilities Equation (15), best responses are computed from them, and the expectation over the hidden type is computed by averaging over all M candidates rather than by sampling. The defender is the midpoint search of Remark 2. In panel (a), with $M = 2^T$, the regret equals $T - 2 + (T + 2)2^{-T}$ to machine precision for every $T \leq 14$, confirming that Theorem 2 is exactly attained. In panel (b), with two types, follow the leader has regret 1.57, 6.35 and 25.60 at $T = 16$, 256 and 4096, against the exact value $\frac{1}{2}\mathbb{E}|N_1 - N_2| = 1.571$, 6.377, 25.531 and the bound $\frac{1}{2}\sqrt{\lfloor T/2 \rfloor} = 1.41$, 5.66, 22.63 of Proposition 2; with one type the regret is 0.5 at every horizon. In panel (c), on the nested instance, the regret tracks $|K|(B-2)$ for $|K| \in \{1, 2, 3, 4\}$ and $B \in \{4, 8, 16, 32\}$, so the additive term of Theorem 4 is real.

D An Oracle-Efficient Implementation

The experiments at $n = 10$ and $n = 20$ cannot build the expert set $\mathcal{E}(C; \epsilon)$, whose construction already takes minutes at $n = 5$ (Table 3). This appendix describes how they avoid it. The idea is that both algorithms only ever need a linear optimum over the experts, and such an optimum can be computed by one Bayesian-Stackelberg optimization.

The lift. For a catalog C and $\mathbf{p} \in \mathcal{P}$ define

$$y_{\alpha,i}(\mathbf{p}) = \mathbb{I}\{b_\alpha(\mathbf{p}) = i\}, \qquad w_{\alpha,i}(\mathbf{p}) = y_{\alpha,i}(\mathbf{p})\, p_i \tag{45}$$

for $(\alpha, i) \in C \times N$, and let $\mathcal{S}(C) = \{(y(\mathbf{p}), w(\mathbf{p})) : \mathbf{p} \in \mathcal{P}\} \subseteq \mathbb{R}^{2|C|n}$.

Lemma 9 (both algorithms are linear over the lift)**.** *Assume the types in C have pairwise distinct best-response maps, so that the set $\mathcal{W}$ of Appendix B.1 has rank $|C|$ and the matrix Λ whose rows are its barycentric spanner $b_1, \ldots, b_{|C|}$ is invertible. For any weights $\omega \in \mathbb{R}^C$,*

$$\sum_{\alpha \in C} \omega_\alpha\, U_d(b_\alpha(\mathbf{p}), \mathbf{p}) = \sum_{\alpha \in C} \sum_{i \in N} \Big[\omega_\alpha u_d^u(i)\, y_{\alpha,i}(\mathbf{p}) + \omega_\alpha \big(u_d^c(i) - u_d^u(i)\big) w_{\alpha,i}(\mathbf{p})\Big], \tag{46}$$

which is linear in the lift. Moreover:

(i) the loss used by Algorithm 1 at round t is Equation (46) *with $\omega = -\mathbb{I}_{\{a_t\}}$;*

(ii) the estimate $\hat{c}_\tau$ of Equation (40)*, used inside Algorithm 2, is Equation* (46) *with $\omega = g_\tau$, where $g_\tau = \Lambda^{-1}\hat{p}_\tau$ and $\hat{p}_\tau \in \{0,1\}^{|C|}$ collects the block's exploration outcomes; g_τ is an unbiased estimate of the per-round type frequencies in the block.*

Proof. For each α exactly one of $y_{\alpha,1}, \dots, y_{\alpha,n}$ equals 1, so $U_d(b_\alpha(\mathbf{p}), \mathbf{p}) = \sum_i y_{\alpha,i}(\mathbf{p}) U_d(i, \mathbf{p})$, and by Equation (1) $y_{\alpha,i} U_d(i, \mathbf{p}) = u_d^u(i) y_{\alpha,i} + \big(u_d^c(i) - u_d^u(i)\big) w_{\alpha,i}$. Summing against ω gives Equation (46), and (i) is the case $\omega = -\mathbb{I}_{\{a_t\}}$. For (ii), let $\lambda(w)$ be the unique vector with $\lambda(w)^\top \Lambda = w^\top$. Then $\sum_j \lambda_j(w)\hat{p}_\tau(b_j) = w^\top \Lambda^{-1}\hat{p}_\tau = w \cdot g_\tau$, and the α-th coordinate of $w_{\sigma,i}$ is $y_{\alpha,i}(\mathbf{p})$, so Equation (40) becomes $\hat{c}_\tau(\mathbf{p}) = \sum_i \big(\sum_\alpha g_{\tau,\alpha} y_{\alpha,i}(\mathbf{p})\big) U_d(i, \mathbf{p}) = \sum_\alpha g_{\tau,\alpha} U_d(b_\alpha(\mathbf{p}), \mathbf{p})$. Finally $\mathbb{E}[\hat{p}_\tau] = \Lambda m_\tau / |B_\tau|$ by Balcan et al. [3, Lemma 6.3], so $\mathbb{E}[g_\tau] = m_\tau / |B_\tau|$. □

The oracle and the learner. By Lemma 9, both algorithms are instances of online linear optimization over $\mathcal{S}(C)$, whose only primitive is

$$\textsc{Oracle}(A, \Delta) = \arg\max_{\mathbf{p} \in \mathcal{P}} \sum_{\alpha,i} \big(A_{\alpha,i}\, y_{\alpha,i}(\mathbf{p}) + \Delta_{\alpha,i}\, w_{\alpha,i}(\mathbf{p})\big). \tag{47}$$

The implementation therefore replaces the Polynomial Weights learner of the analysis by Follow the Perturbed Leader over $\mathcal{S}(C)$ [24], which calls Equation (47) once per round and not at all on the exploration rounds of Algorithm 2. This changes the learner but not the structure of either algorithm: the catalog, the restarts and the estimation blocks are as analyzed. The guarantee is of the same order in T. Follow the Perturbed Leader has regret $O(\sqrt{DRAT})$ for linear losses, where D is the ℓ_1-diameter of the decision set, R bounds the per-round loss and A the ℓ_1-norm of a loss vector [24]. On $\mathcal{S}(C)$ these are $D \le 4|C|$, $R \le 1$ and $A \le 3n$, the last because $\sum_i (|u_d^u(i)| + |u_d^c(i) - u_d^u(i)|) \le 3n$. We run its anytime perturbation schedule, since segment lengths are not known in advance. The constants of Theorems 1 and 4 are those of Polynomial Weights, and the experiments test the exponents, not the constants.

Computing the oracle. Grouping Equation (47) by the response profile $\sigma : C \to N$ makes the objective linear on each region $\mathcal{P}_\sigma(C)$ of Equation (4), so Equation (47) is the maximum of $n^{|C|}$ linear programs in n variables, the extension to a set of types of the multiple-LP method of Conitzer and Sandholm [14]. In the experiments we instead solve one mixed-integer program with $n|C|$ binary variables, in the style of the Bayesian-Stackelberg program of Paruchuri et al. [37], and use the profile enumeration as a correctness check. We do not claim polynomial time in $|C|$, since the Bayesian Stackelberg problem is NP-hard [14]; the gain is in n, the parameter that made $\mathcal{E}(C; \epsilon)$ unusable. Both methods solve over regions shrunk by a margin $\gamma = 10^{-6}$, so that the returned $\mathbf{p}$ realizes its profile under any tie-breaking rule; this plays the role of the ϵ-approximation in Lemma 1. As in the experiments, $\mathcal{P} = \{\mathbf{p} \ge 0 : \sum_i p_i = 1\}$ with one resource, and the algorithm and the benchmark optimize over the same set.

The rank condition. The rank hypothesis of Lemma 9 is a normalization of the catalog, not an assumption on the game. Two types with the same best-response map on all of $\mathcal{P}$ are indistinguishable to the defender's payoff, to the benchmark Equation (2) and to the feedback, so the catalog may keep one representative of each map. This can only shrink the catalog, so every bound of Section 5 is unaffected. The implementation does not remove duplicates but detects them: if no full-rank spanner exists, it records a rank failure and skips exploration until the catalog next changes. No rank failure occurred in the runs behind Figure 6.

n	$\lvert C\rvert$	$\binom{H}{n}$	build $\mathcal{E}$	$n^{\lvert C\rvert}$ LPs	MILP
3	4	3.3×10^{3}	242 ms ($\lvert\mathcal{E}\rvert = 444$)	30 ms	19 ms
5	4	3.5×10^{7}	infeasible	209 ms	64 ms
10	4	1.2×10^{19}	infeasible	3.2 s	145 ms
20	4	2.1×10^{45}	infeasible	65 s	0.93 s
30	4	7.7×10^{73}	infeasible	infeasible	0.89 s
40	4	9.0×10^{103}	infeasible	infeasible	1.60 s

Table 3: Cost of one step, with $H = n + 1 + 2|C|\binom{n}{2}$ the number of hyperplanes. Column 3 is the number of n-subsets that the exhaustive construction of $\mathcal{E}(C;\epsilon)$ must solve. Columns 5 and 6 are the two methods for Equation (47); wherever both ran they returned the same optimum to within 2×10^{-6}. Median of 10 calls. The MILP time is not monotone in n because it is an exact branch-and-bound solve.

The spanner and the range of the estimates. The implementation does not enumerate $\mathcal{W}$. It samples coverage vectors, records the vectors $w_{\sigma,i}$ they realize, and runs the greedy determinant maximization of Awerbuch and Kleinberg [2] on that pool. A spanner of a sub-pool need not give coefficients $\lambda \in [-1, 1]$ for every $w \in \mathcal{W}$, so the range bound $|\hat{c}_\tau| \le n|C|$ of Equation (41) is not guaranteed. Unbiasedness, which is what the estimator's correctness rests on, needs only that Λ be invertible, and holds exactly. For the same reason the implementation scales the perturbation by the range of the estimates it actually computes rather than by the worst case $n|C|$. This is a tuning choice: Theorem 4 does not rest on these constants, and Figure 6 tests the exponent.

Verification. On 60 random instances with $n \in \{3, 4, 5\}$ and $|C| \in \{1, 2, 3\}$, the two oracle methods and a brute-force maximization over the enumerated $\mathcal{E}(C;\epsilon)$ agree to within 1.4×10^{-6}. Over 4×10^4 replications of one estimation block at $n = 6$, $|C| = 3$, the mean of $|B_\tau|\, g_\tau$ matches the true type counts of the block to within 0.05 out of 40, inside the Monte Carlo error of 1.1 (three standard errors), as Lemma 9(ii) predicts.

Tractability. Table 3 times the three routines. Building $\mathcal{E}(C;\epsilon)$ takes 0.24 seconds at $n = 3$, $|C| = 4$, already 20 minutes at $n = 5$, $|C| = 2$, and is out of reach beyond that, whereas one oracle call costs 145 ms at $n = 10$, 0.9 s at $n = 20$ and 1.6 s at $n = 40$. Since neither algorithm makes more than one oracle call per round, this is what moves the experiments from $n = 3$ to $n = 20$.