Lecture 8.2: Monitoring Autonomous Agents

Robust Submodularity, the Interventionist Greedy Game, and Poison Pills — Accessible Companion Notes

CSCE 631-D: Intelligent Agents — Computational Game Solving
Alan Kuhnle, Texas A&M University

Fall 2026 (Module 8, Week 10, Lecture 2 of 2)

Using these notes. These notes expand the Lecture 8.2 slides into readable prose, state every definition and theorem in full, give complete proofs of the Interventionist Greedy Game (IGG) results that the slides only sketch, describe every diagram and table in words, and work through numerical examples. Headings follow the deck in order. Slide references use **PDF page numbers in the 34-page slide deck**, counting the title and the maroon part dividers; the references occupy pages 32–34.

Attribution. The IGG model, the results labeled C-T1 through C-T7, the evidence items C-E1 and C-E2, and the open questions C-O1 and C-O2 are taken from the *course notes* (Kuhnle, "Interventionist Greedy Game" draft, and the November 2026 course notes on IGG poisoning and strategic audit timing). They are unpublished. Where these notes say "proved," a complete proof appears below; "evidence" means a finite computation; "open" means no proof is claimed. Please cite them as "course notes" if you use them in PA5.

Contents

1 From a Budget to an Adversary **2**

2 Part I: Robust Submodular Optimization **3**

3 Part II: The Interventionist Greedy Game **4**

4 Part III: The Off-Limits Model **6**

5 Part IV: The Poison-Pill Construction **8**

6 Part V: PA5 Preview and Wrap-Up **12**

Appendix: Two Further Worked Checks **13**

References **13**

Notation used in this lecture

- V: a finite ground set of n items (actions, outputs, sensor sites). $f : 2^V \to \mathbb{R}_{\geq 0}$: a set function. $f(a \mid S) = f(S \cup \{a\}) - f(S)$: the marginal gain of a given S.
- *Submodular*: $f(a \mid S) \geq f(a \mid S')$ whenever $S \subseteq S'$ and $a \notin S'$ (diminishing returns). *Monotone*: $f(S) \leq f(S')$ whenever $S \subseteq S'$. *Modular*: $f(S) = \sum_{a \in S} w_a$.
- k: the number of rounds (equivalently the size of the final set). $r \leq k$: the monitor's override budget. ℓ: the number of poisons.
- $\mathrm{OPT}_{\leq k} = \max_{|T| \leq k} f(T)$ and $\mathrm{OPT}_{=k} = \max_{|T|=k} f(T)$. For monotone f they coincide; for nonmonotone f they can differ.
- $\mathrm{Val}(r, k, f) = \max_\pi \min_\sigma \mathbb{E} f(S_k)$: the value of the IGG with monitor strategy π and agent strategy σ.
- S: selected set; B: blocked (off-limits) set; q: remaining override budget; $E = V \setminus (S \cup B)$: eligible proposals.

1 From a Budget to an Adversary

1.1 Slide 1 – Title

The subtitle of the deck is the thesis: the $(1 - 1/e)$ greedy guarantee from Lecture 8.1 assumed a fixed objective and a passive ground set. Today the objective, and then the set itself, are chosen in part by an adversary, and we ask what guarantee survives.

1.2 Slide 3 – Recall: Lecture 8.1 in Three Lines

Lecture 8.1 established three things we use throughout.

1. **Greedy with a guarantee.** For nonnegative monotone submodular f and a cardinality budget k, greedy returns S_k with $f(S_k) \geq (1 - 1/e)\mathrm{OPT}_k$ (Nemhauser, Wolsey & Fisher 1978). PA5 Task 4 plots $(1 - 1/e) f(V)$ as a reference line.
2. **Three regimes.** Reliability monitoring (a fallible, non-strategic agent; choose a *set*); adaptive monitoring (inspections reveal information; choose a *policy*); AI control (the agent may deliberately evade; play a *game*).
3. **Where we stopped.** In Regime 3 independence breaks: a scheming adversary concentrates attacks where monitoring is weak. The two devices introduced were randomize-and-commit (Krause, Roper & Golovin 2011) and shutdown-on-detection as a trigger strategy (Greenblatt et al. 2024, Table 1).

Today's question has three parts: what happens to the guarantee when (i) the objective is chosen adversarially, (ii) the solution set is built jointly with an adversary who proposes elements, and (iii) the objective is no longer monotone.

1.3 Slide 4 – Today: Five Parts

Part I covers robust submodular optimization and SATURATE. Part II defines the IGG, solves it exactly, and proves the r/k ratio for monotone objectives. Part III is the off-limits model and its connection to the AI-control games of Griffin et al. and the audit protocol of Greenblatt et al. Part IV is the poison-pill construction, which gives r^2/k for nonmonotone objectives, and a careful

account of what is proved and what is conjectured. Part V previews PA5 and summarizes the module.

2 Part I: Robust Submodular Optimization

2.1 Slide 6 – Worst-Case Coverage: The Max–Min Problem

Krause & Golovin (2014, §3.4) motivate robustness with sensor placement in a water network under malicious contamination. There are m intrusion scenarios; in scenario i the benefit of sensors at S is a monotone submodular $f_i(S)$. The adversary chooses the scenario *after* seeing S, so the right objective is the worst case:

$$S^* = \arg\max_{|S|\le k} \min_i f_i(S). \tag{1}$$

Two negative facts follow. First, $f_{\min}(S) = \min_i f_i(S)$ is not in general submodular, and greedy applied to it can perform arbitrarily badly. Second (Krause et al. 2008), unless P=NP no efficient algorithm returns S with $f_{\min}(S) \ge \alpha(n)\mathrm{OPT}_k$ for *any* function α of the problem size, even an exponentially small one.

Monitoring reading. In PA5 Task 4 the coverage function $f(S) = 1 - \prod_{j\in S}(1 - p_j d_j)$ treats the backdoor probabilities p_j as fixed. If the adversary instead chooses *which* output to backdoor after seeing which outputs you inspect, each possible choice is a scenario and the objective becomes a minimum over scenarios. The $(1 - 1/e)$ guarantee does not apply to that objective.

2.2 Slide 7 – Saturate: Buy the Guarantee With Budget, Not Value

Despite the inapproximability, Krause et al. (2008) obtain a different kind of guarantee, bicriterion rather than multiplicative in value.

Truncation. For a guess c of the achievable worst-case value, define

$$\bar{f}_c(S) = \frac{1}{m}\sum_{i=1}^{m} \min\{f_i(S), c\}.$$

Truncating a monotone submodular function at a constant preserves monotonicity and submodularity, and an average of such functions is again monotone submodular. Moreover $\bar{f}_c(S) = c$ exactly when every scenario attains at least c, so "is c achievable with k items?" becomes a submodular *cover* question: find the smallest S with $\bar{f}_c(S) = c$.

Algorithm. Binary search over c. For each candidate, run the greedy submodular-cover algorithm on $\bar{f}_c$ until it saturates at c. Accept c if the cover used at most αk items, where $\alpha = 1 + \max_v \ln \sum_i f_i(\{v\})$; otherwise reject and lower c.

Theorem 1 (Krause, McMahan, Guestrin & Gupta 2008; Krause & Golovin §3.4)**.** SATURATE returns a set S with

$$f_{\min}(S) \ge \mathrm{OPT}_k \qquad \text{and} \qquad |S| \le \Big(1 + \max_v \ln \sum_i f_i(\{v\})\Big) k.$$

The logarithmic budget factor is optimal under standard complexity assumptions.

In words: you obtain as much worst-case value as the best k-set, but you must pay a logarithmically larger budget. This is the robust analogue of "over-budget helps" from Lecture 8.1.

2.3 Slide 8 – Two Other Ways the Adversary Can Corrupt Coverage

Table, read as prose. The slide's table has three rows, one per adversary move. Row one: the adversary picks the scenario after seeing S; the model is the max–min problem (1); what is known is SATURATE's bicriterion guarantee. Row two: the adversary deletes up to τ of the chosen items; the model is $\max_{|S|\le k} \min_{|Z|\le\tau} f(S \setminus Z)$; what is known is a family of constant-factor guarantees for small τ due to Orlin, Schulz & Udwani (2018), for monotone f only. Row three: the adversary forces additions into one shared set, while the monitor may override at most r times; this is the IGG of Parts II and IV.

The monitoring reading: an agent that evades a monitor *deletes* that monitor's contribution to coverage; an agent whose harmful action you failed to stop *adds* an element to the realized set. The first two rows keep the objective monotone and leave the defender in control of the whole set. The third row drops both assumptions, and that is why its answers look different.

Remark 1 (Do not import constants across rows)**.** The course notes (item C-K2) inspected both the SATURATE line of work and the deletion-robust line and concluded that neither is a theorem about forced additions to one shared nonmonotone set. The $(1 - 1/e)$, bicriterion, and deletion-robust constants must not be quoted as IGG guarantees.

3 Part II: The Interventionist Greedy Game

3.1 Slide 10 – The IGG: Setup

Definition 1 (Interventionist Greedy Game; course notes C-K1 and the well-posed definition)**.** Two players share a ground set V with $n = |V| \ge k + r$ and a known nonnegative submodular objective f with exact value-oracle access. The state is (S, B, q): the selected set, the blocked set, and the remaining override budget, initially $(\emptyset, \emptyset, r)$. There are k rounds. In each round:

1. The *agent* (adversary) proposes an eligible element $a \in E = V \setminus (S \cup B)$.
2. The *monitor* (controller) either *accepts*, setting $S \leftarrow S \cup \{a\}$, or, if $q > 0$, *overrides*: it chooses an eligible $i \ne a$, sets $S \leftarrow S \cup \{i\}$, $B \leftarrow B \cup \{a\}$, and $q \leftarrow q - 1$.

The payoff is $f(S_k)$, which the monitor maximizes and the agent minimizes. Both players observe the entire history and the current proposal; the monitor moves after seeing the proposal. The game value is $\mathrm{Val}(r, k, f) = \max_\pi \min_\sigma \mathbb{E} f(S_k)$.

Two conventions deserve comment. Overriding a with a itself produces the same selected set as accepting, wastes a unit of budget, and blocks an element that is already selected; it can never help the monitor, so it is excluded. Blocking is permanent: an element in B can never be proposed again. Part III shows that the whole game turns on this rule. The size condition $n \ge k + r$ ensures that at every round there is an eligible proposal, because at round $t \le k$ at most $(t - 1) + r$ elements have been selected or blocked.

3.2 Slide 11 – The Game Tree: One Round of the IGG

Figure description. The diagram shows one round as a tree. At the top is a red agent node labeled with the state (S, B, q) and the text "proposes a in V minus S union B." Three arrows labeled a_1, a_2, a_3 lead to three green monitor nodes, one per possible proposal. From the middle monitor node two arrows leave: a solid one labeled *accept* to a blue state box $(S \cup \{a_2\}, B, q)$, and

a dashed one labeled *override with $i \neq a_2$, needs $q > 0$* to a blue state box $(S \cup \{i\}, B \cup \{a_2\}, q-1)$. The outer two monitor nodes carry the gray note "same two moves." A caption below reads "repeat for k rounds; leaf payoff $f(S_k)$: monitor max, agent min."

The tree is finite and both players have perfect information, so backward induction applies. The next slide states the recursion.

3.3 Slide 12 – Exact Solution: Backward Induction (C-T1)

Theorem 2 (C-T1: exact recursion and pure optimal strategies)**.** Let $E = V \setminus (S \cup B)$ and set $V(S, B, q) = f(S)$ when $|S| = k$. Otherwise

$$V(S,B,q) = \min_{a \in E} \max\Big\{ V(S \cup \{a\}, B, q),\ \max_{i \in E \setminus \{a\}} V\big(S \cup \{i\}, B \cup \{a\}, q-1\big) \Big\},$$

with the override term omitted when $q = 0$. Then $\mathrm{Val}(r, k, f) = V(\emptyset, \emptyset, r)$, and both players have pure optimal strategies. Private randomization does not change the value.

Proof. Induct on the number of remaining rounds. At terminal states the value is the payoff. Suppose all successor values are correct. After observing a the monitor can attain the largest successor value; any distribution over replies yields an average of successor values, which is at most their maximum. The agent can choose the proposal whose best reply is smallest; a distribution over proposals cannot do better than that minimum. Hence the recursion holds and pure choices are optimal at every node. The size condition guarantees that E is nonempty at every nonterminal state, so the recursion is well defined. □

Remark 2 (When does randomization help?)**.** Here it does not, because the agent sees everything before moving and the monitor sees the proposal before replying. Contrast Lecture 8.1 and the audit game in the course notes (C-T6): when the attacker must commit *before* seeing the defender's current private audit draw, a hidden uniform choice of K audit times out of N gives safety K/N, whereas any public or deterministic rule gives 0 when $K < N$. The information order within a round, not the budget, decides whether mixing has value.

The recursion is exponential in n. It defines the game value and powers the finite-case evidence in Part IV; it is not an algorithm the monitor runs at deployment.

3.4 Slide 13 – Monotone Objectives: The Ratio Is Exactly r/k (C-T2)

Lemma 1 (Target forcing; any f)**.** Fix any $T \subseteq V$ with $|T| \leq r$. The monitor can guarantee $T \subseteq S_k$ using at most $|T|$ overrides and never blocking an element of T.

Proof. Let m be the number of target elements not yet selected and l the number of rounds remaining; initially $l = k \geq r \geq m$. Accept any target proposal (this reduces m and l together). If the proposal is outside T and $l > m$, accept it (only l falls, so $l \geq m$ still holds). If the proposal is outside T and $l = m$, override it with a missing target element (both l and m fall by one). The invariant $l \geq m$ is maintained, every override inserts a target, so at most $|T|$ overrides are spent, and no target is ever blocked because we only block non-targets. At the end $l = 0$, so $m = 0$. □

Lemma 2 (Submodular averaging)**.** Fix a set A and a disjoint M-element set U. Let a_j be the average of $f(A \cup T)$ over uniformly random j-subsets $T \subseteq U$. Then the increments $a_{j+1} - a_j$ are nonincreasing in j, hence

$$a_j \geq \Big(1 - \frac{j}{M}\Big) a_0 + \frac{j}{M}\, a_M.$$

Proof. Sample a uniform j-subset T and then two distinct uniform elements x, y of $U \setminus T$. Submodularity gives $f(x \mid A \cup T) \geq f(x \mid A \cup T \cup \{y\})$. Taking expectations, the left side is $a_{j+1} - a_j$ and the right side is $a_{j+2} - a_{j+1}$. So $j \mapsto a_j$ is concave on $\{0, \ldots, M\}$, and a concave sequence lies above the chord joining its endpoints. □

Theorem 3 (C-T2: sharp monotone ratio)**.** For every nonnegative monotone submodular f, $\mathrm{Val}(r, k, f) \geq \frac{r}{k}\mathrm{OPT}_{\leq k}$, and the ratio r/k is sharp over functions and ground sets. A polynomial-query strategy guarantees $[1 - (1 - 1/k)^r]\mathrm{OPT}_{\leq k} \geq (1 - e^{-r/k})\mathrm{OPT}_{\leq k}$.

Proof. Let O attain $\mathrm{OPT}_{\leq k}$ with $|O| = m \leq k$. If $m \leq r$, force all of O by Lemma 1; monotonicity makes any extra accepted elements harmless, so the payoff is at least $f(O)$. Otherwise apply Lemma 2 with $A = \emptyset$ and $U = O$, $j = r$: the average over r-subsets of O is at least $(1-r/m)f(\emptyset)+(r/m)f(O) \geq (r/k)f(O)$, so some r-subset $T \subseteq O$ has $f(T) \geq (r/k)\mathrm{OPT}_{\leq k}$. Force T; monotonicity again makes the remaining elements harmless.

For the efficient version, build a greedy target $G_0 = \emptyset$, $G_{j+1} = G_j \cup \{\arg\max_e f(e \mid G_j)\}$. Submodularity and monotonicity give $\mathrm{OPT}_{\leq k} - f(G_j) \leq \sum_{e \in O \setminus G_j} f(e \mid G_j) \leq k \max_e f(e \mid G_j)$, so each greedy step shrinks the gap by a factor at most $1 - 1/k$. After r steps $f(G_r) \geq [1 - (1 - 1/k)^r]\mathrm{OPT}_{\leq k}$; force G_r.

For sharpness take k unit-value goods and k zero-value elements with modular f. The agent proposes a fresh zero every round. At most r goods can enter (one per override), so the payoff is at most r while the optimum is k. Forcing any r goods attains r. □

Note where monotonicity enters: only to say that the elements the agent sneaks in cost nothing beyond the slot they occupy. The forcing lemma itself holds for every f. That observation is the seed of Part IV.

3.5 Slide 14 – Worked Example: $k = 4$, $r = 2$, Monotone

Example 1. Let $V = \{g_1, g_2, g_3, g_4, z_1, z_2, z_3, z_4\}$ and $f(S) = |S \cap \{g_1, \ldots, g_4\}|$, so $\mathrm{OPT}_{\leq 4} = 4$. The monitor chooses target $T = \{g_1, g_2\}$ and plays the forcing rule; the agent proposes zeros.

Table, read as prose. Round 1: four rounds left, two targets missing, agent proposes z_1; since $l = 4 > m = 2$ the monitor accepts; $S = \{z_1\}$. Round 2: $l = 3 > 2$, agent proposes z_2, accept; $S = \{z_1, z_2\}$. Round 3: $l = 2 = m$, agent proposes z_3; the monitor overrides, inserting g_1 and blocking z_3; $S = \{z_1, z_2, g_1\}$. Round 4: $l = 1 = m$, agent proposes z_4; override with g_2, block z_4; $S = \{z_1, z_2, g_1, g_2\}$.

The payoff is $f(S_4) = 2 = \frac{r}{k}\mathrm{OPT}$. Neither side can improve: the agent gains nothing by proposing goods, and the monitor cannot insert more than two goods with two overrides while the agent withholds them. So $\mathrm{Val}(2, 4, f) = 2$. Each override is a *replacement* of a useless action by a useful one; with a monotone objective the useless actions cost only the slot, and the entire loss is opportunity cost.

4 Part III: The Off-Limits Model

4.1 Slide 16 – Pruned Actions Stay Pruned

The rule: an overridden proposal enters B and is never eligible again.

Why it is load-bearing. Suppose blocking were *not* permanent and f has one lethal element p with $f(S) = 0$ whenever $p \in S$. The agent proposes p; the monitor must override; the agent proposes p again; the monitor must override again. After r rounds the budget is gone, the agent proposes p once more, and it is accepted. The value is 0 for every $r < k$. With permanent blocking the first override removes p for good: one intervention per *distinct* threat, not one per attempt. The course notes record this as the reason the permanent rule is "essential" (sketched obstacle after C-O1).

Looping, revisited. The syllabus asks why looping happens. Under any submodular objective $f(a \mid S) = 0$ once $a \in S$: an agent that repeats the same fix earns nothing on the second application. The off-limits rule is the monitor-side mirror of the same fact. Repeating an *attack* should cost the monitor zero marginal budget; without permanence it costs one override per repeat, and looping becomes a budget-exhaustion attack.

4.2 Slide 17 – The Agent's Side: Attempt, or Hold the Option?

Under permanence a risky proposal is a one-shot gamble for the agent. If accepted, the harmful element is in S forever (the monitor cannot delete accepted elements either). If overridden, the element is gone forever and the monitor has spent one of r coins. Two consequences are visible in the exact solution.

1. **Budget-draining is still rational, but only with distinct threats.** In the sharp example of C-T2 the agent proposes a fresh zero every round; each override costs the monitor one coin and the agent nothing. The agent's supply of distinct bad actions is the real constraint, which is exactly the role the poison count ℓ plays in Part IV.
2. **Counts, not timing.** The forcing rule waits until rounds left equal targets missing; attacking early does not make overrides cheaper. The game is decided by counts, which is why pure strategies suffice.

Remark 3 (Inspection and replacement are different resources)**.** In the one-attack audit game of the course notes, timing is everything: hidden current randomization gives safety K/N (C-T6). When the defender has K inspections and r replacements and an inspection alone cannot stop a submission, the value is $\min\{1, (K + r)/N\}$ for $r \geq 1$ and 0 for $r = 0$ (C-T7). The course notes warn (C-O2) that this formula must not be used to combine Greenblatt's audit budget with an imagined replacement budget without specifying the mechanics.

4.3 Slide 18 – The AI-Control Game Over a Sequence of Tasks (Griffin et al.)

Griffin, Thomson, Shlegeris & Abate (2024) formalize AI control as a game between Blue (the protocol) and Red (the untrusted model) over a sequence of programming problems. Blue has an audit budget and is memoryful; Red is memoryless; the solution concept is leader–follower. Sections 6–8 contain the following.

- **§6, existing protocols.** Never Audit, Trusted Model Only, Upfront Auditing, and Trusted Monitoring from Greenblatt et al. are all memoryless protocols with fixed audit and defer false-positive rates (0%, 2%, or 100% audit FPR with various defer FPRs).
- **§6, new protocols.** Raising the audit FPR from 2% to 3% gives a small Pareto improvement in the "original" setting; 5% gives a large improvement in the "weak-monitor" setting. In the original case Greenblatt's closest policy is 0.0042 from the computed frontier and deterministic policies are mostly optimal; in the weak case they are at least 0.02 away and mixed protocols make up most of the frontier.

- **§6.1, assumptions.** Safety and usefulness rise steeply in the audit budget when the budget is small; the number of problems per episode has little effect at fixed FPRs; human-auditor false negatives reduce safety almost linearly.
- **§7, dangerous actions.** A_D^R is a minimal set of actions without which an unacceptable outcome cannot occur. If harm requires four dangerous actions in series, an honest-but-imperfect model almost never produces it, whereas a schemer's dangerous actions are correlated.
- **§8, related work.** Stackelberg security games (one round), inspection games (Avenhaus et al.; one round), and recursive inspection games (memoryful follower) are the nearest relatives.

IGG reading. The off-limits set B is the analogue of "actions Blue has already removed." Red's feasible proposals shrink with history, which is precisely what the memoryless-Red assumption abstracts away.

4.4 Slide 19 – Protocol Design With Scarce Trusted Labor (Greenblatt et al. §2.2)

Greenblatt et al. model scarce high-quality labor as "H points": the protocol may use unmodified GPT-4 on up to 2% of problems. Spending a point on a submission does two things: attempt to *exploit* the code (assumed to succeed exactly when the attack policy could), and *rate* its suspicion (shut down if more suspicious than any plausible normal submission).

Table, read as prose. Table 1 of the paper has three rows (successful backdoor, failed backdoor, no attempt) and three columns (audited and found bad, audited and found fine, not audited). A successful backdoor that is audited is always found bad and triggers shutdown: Blue wins. A successful backdoor not audited: Red wins. A failed backdoor that is audited and found bad triggers shutdown; otherwise the game continues. No attempt is never found bad; the game continues.

Three points for today. First, one confirmed backdoor switches the protocol to the trusted model for the rest of the episode: shutdown is the *global* off-limits rule, applied to the whole untrusted policy rather than one action. Second, H points are also spent constructing synthetic backdoored examples to calibrate the monitor. Third, auditing is imperfect but good enough in this setting; weak auditing is left open (§5.1.1). PA5 Task 2 implements the audit/run version with a single threshold τ.

5 Part IV: The Poison-Pill Construction

5.1 Slide 21 – Poison Products: A Nonmonotone Submodular Objective

Definition 2 (Poison product)**.** Let U be a set of *goods* with a nonnegative monotone submodular $h : 2^U \to \mathbb{R}_{\geq 0}$, and let P be a disjoint set of $\ell \geq 1$ *poisons*. On $V = U \cup P$ define

$$g(S) = h(S \cap U)\Big(1 - \frac{|S \cap P|}{\ell}\Big).$$

Each poison removes a $1/\ell$ fraction of whatever value the goods provide; all ℓ poisons destroy everything.

Lemma 3 (C-T3, first part)**.** g is nonnegative and submodular.

Proof. Nonnegativity: $|S \cap P| \le \ell$. For a good $u \notin S$, $g(u \mid S) = h(u \mid S \cap U) \cdot (1 - |S \cap P|/\ell)$. Both factors are nonnegative, and both are nonincreasing as S grows (the first by submodularity of h, the second because $|S \cap P|$ can only grow), so their product is nonincreasing in S. For a poison $p \notin S$, $g(p \mid S) = -h(S \cap U)/\ell$, which is nonincreasing in S because h is monotone. Both cases give diminishing returns, so g is submodular. □

Unless $h \equiv 0$, g is not monotone: adding a poison to a set with positive value strictly decreases it. (The nontriviality condition is needed; the course notes flag that the zero function is the degenerate case.)

Monitoring reading. A good is a useful action. A poison is an action whose damage scales with the useful work already done: exfiltrating weights after training them, or inserting a backdoor into a library that every later task imports. Individually each pill is cheap; together they are lethal.

5.2 Slide 22 – The Exact Game Value With Unit Goods (C-T3, C-T4)

Theorem 4 (C-T3: exact modular game and worst poison count)**.** Let $h(A) = |A|$ on at least k unit goods, let $|P| = \ell$, assume there are no other elements and $|U| + \ell \ge k + r$. Put $b_* = \max\{0, \min(k, \ell) - r\}$ and $\phi(b) = (k - b)(1 - b/\ell)$. Then

$$\mathrm{Val}(r, k, g) = \phi(b_*) = (k - b_*)\Big(1 - \frac{b_*}{\ell}\Big), \qquad \mathrm{OPT}_{\le k} = k.$$

Over all positive integers ℓ the smallest value is r^2/k, attained at $\ell = k$. Consequently the worst-case ratio over all nonnegative submodular functions is at most $(r/k)^2$.

Proof. A terminal set with b poisons has $k - b$ goods and payoff $\phi(b)$. On $0 \le b \le \min(k, \ell)$ the function ϕ is nonincreasing: its derivative is $-1 - k/\ell + 2b/\ell \le -1 - k/\ell + 2\min(k, \ell)/\ell \le 0$ when $\ell \ge k$ (then $\min(k, \ell) = k$ and the bound is $-1 + k/\ell \le 0$) and when $\ell < k$ (then $\min(k, \ell) = \ell$ and the bound is $1 - k/\ell < 0$). The same conclusion follows from consecutive differences.

Monitor's lower bound. Accept goods; override each of the first up to r poison proposals with a fresh good. If the budget is never exhausted, no poison is accepted and the payoff is k. If all r overrides are used, at most $k - r$ later slots remain and at most $\ell - r$ poisons remain unblocked, so $b \le \min(k - r, \ell - r) = b_*$, and the payoff is at least $\phi(b_*)$ by monotonicity of ϕ. A fresh good always exists because the monitor never blocks goods and $|U| \ge k$.

Agent's upper bound. Propose a fresh eligible poison while any remain. If all k proposals are poisons, every selected good required an override, so $b \ge k - r$. If poisons run out before round k, every poison has been selected or blocked; at most r are blocked, so $b \ge \ell - r$. In both cases $b \ge b_*$, including against monitors that replace a poison with another poison. Hence no monitor exceeds $\phi(b_*)$.

Optimum. Any k goods give k; any feasible set is worth at most its number of goods, at most k.

Minimizing over ℓ. For $0 < r < k$: if $\ell \le r$ then $b_* = 0$ and the value is k. If $r < \ell \le k$ then $b_* = \ell - r$ and the value is $(k - \ell + r) \cdot r/\ell$, decreasing in ℓ. If $\ell \ge k$ then $b_* = k - r$ and the value is $r \cdot (\ell - k + r)/\ell$, increasing in ℓ. The two pieces meet at $\ell = k$ with value r^2/k. For $r = 0$ the minimum is 0 and for $r = k$ every value is k, consistent with r^2/k. □

Theorem 5 (C-T4: the k-poison witness survives zero padding)**.** Add any number of dummy elements that contribute nothing to h and are not poisons to the $\ell = k$ instance. The value is still exactly r^2/k.

Proof. The agent's k distinct poison proposals still force at most r goods and at least $k-r$ poisons, so the value is at most $r(1-(k-r)/k) = r^2/k$. For the lower bound the monitor accepts unit goods and overrides every other proposal (poison or dummy) with a fresh unit good until r overrides are spent. If the budget is spent, at least r goods were inserted; if not, every selected element is a good. With final counts a goods and b poisons, $a \geq r$ and $b \leq k-a$, so $g(S_k) = a(1-b/k) \geq a^2/k \geq r^2/k$. No unit good is ever blocked, so a fresh one is always available. □

5.3 Slide 23 – Worked Example: $k = 4$, $r = 2$, Four Poisons

Example 2. Let $U = \{g_1, \ldots, g_6\}$ be unit goods and $P = \{p_1, \ldots, p_4\}$, so $\ell = k = 4$ and $g(S) = |S \cap U|(1 - |S \cap P|/4)$. Then $\mathrm{OPT}_{\leq 4} = 4$ and $b_* = \min(4,4) - 2 = 2$.

Table, read as prose. Round 1: the agent proposes p_1; the monitor overrides with g_1 and blocks p_1, one override left; $S = \{g_1\}$, value 1. Round 2: agent proposes p_2; override with g_2, block p_2, no overrides left; $S = \{g_1, g_2\}$, value 2. Round 3: agent proposes p_3; the monitor must accept; $S = \{g_1, g_2, p_3\}$, value $2 \cdot \frac{3}{4} = 1.5$. Round 4: agent proposes p_4; accept; $S = \{g_1, g_2, p_3, p_4\}$, value $2 \cdot \frac{2}{4} = 1$.

The value is $\phi(2) = (4-2)(1-2/4) = 1 = r^2/k$. On the same (k, r) the monotone Example 1 had value $2 = \frac{r}{k}\mathrm{OPT}$.

Varying the poison count. With $\ell = 2$, $b_* = 0$ and the value is 4: the budget covers every poison. With $\ell = 8$, $b_* = \min(4,8) - 2 = 2$ and the value is $(4-2)(1-2/8) = 1.5$: each poison is weaker. The worst case is $\ell = k = 4$. A good way to internalize the formula: the agent wants enough distinct poisons to drain the budget, but not so many that each is diluted.

5.4 Slide 24 – The Quadratic Gap: What Monotonicity Was Buying You

Table, read as prose. The slide compares monotone and nonmonotone submodular objectives row by row. Worst-case value with r of k: exactly $\frac{r}{k}\mathrm{OPT}_{\leq k}$ versus at most $(\frac{r}{k})^2\mathrm{OPT}_{\leq k}$. Half the budget: 50% versus at most 25%. A tenth of the budget: 10% versus at most 1%. What an accepted bad action costs: one slot, versus one slot and value already earned. The monitor's winning idea: force a good r-set, versus spend overrides only on poisons.

The lesson of Module 8: monotonicity is not a technicality in the greedy theorem. It is the assumption that nothing the agent does can undo what you have already secured. Drop it and the monitor must spend its budget defending value instead of creating it; the guarantee falls from linear in the budget fraction to, at best, quadratic.

Theorem 6 (C-T5: full-control boundary)**.** At $r = k$ the game value is exactly $\mathrm{OPT}_{=k}$. For every nonnegative submodular f on $n \geq 2k$ elements, $\mathrm{OPT}_{=k} \geq \frac{n-k}{n}\mathrm{OPT}_{\leq k} \geq \frac{1}{2}\mathrm{OPT}_{\leq k}$, and the constant $\frac{1}{2}$ is asymptotically tight even for normalized functions.

Proof. Lemma 1 does not use monotonicity. With $r = k$ the monitor can force any size-k target, in particular a best one, so the value is at least $\mathrm{OPT}_{=k}$; every terminal set has size exactly k, so it is at most $\mathrm{OPT}_{=k}$. For the inequality let O attain $\mathrm{OPT}_{\leq k}$ with $|O| = o < k$ and apply Lemma 2 with $A = O$, $U = V \setminus O$ ($M = n - o$), $j = k - o$: the average over size-k supersets of O is at least $(1 - \frac{k-o}{n-o})f(O) + \frac{k-o}{n-o}f(V) \geq \frac{n-k}{n-o}f(O) \geq \frac{n-k}{n}f(O)$, using $f(V) \geq 0$. Some size-k superset attains this. For tightness take $n = 2k$, one good u, $2k-1$ poisons, and $f(S) = \mathbf{1}\{u \in S\}(2k - |S|)$, which is a positive scaling of a one-good poison product: $\mathrm{OPT}_{\leq k} = 2k-1$ at $\{u\}$ and $\mathrm{OPT}_{=k} = k$, with ratio tending to $\frac{1}{2}$. □

Consequently a universal bound of the form $\mathrm{Val} \geq (r/k)^2 \mathrm{OPT}_{\leq k}$ with coefficient one is *false* at $r = k$. This does not refute an $\Omega((r/k)^2)$ bound with a smaller constant.

5.5 Slide 25 – What Is Proved, What Is Evidence, What Is Open

Proved. The monotone ratio r/k and its sharpness (C-T2); the poison product is submodular, the unit-good game value is $\phi(b_*)$, and the worst case over ℓ is r^2/k at $\ell = k$ (C-T3); zero padding does not change the $\ell = k$ value (C-T4); full control gives $\mathrm{OPT}_{=k}$ with the $\frac{1}{2}$ boundary (C-T5).

Evidence (finite computation, not proof). Exact count-state minimax over padded poison families with $k + r$ unit goods and $k + r$ zero dummies, for every $2 \leq k \leq 8$, $1 \leq r \leq k$, $1 \leq \ell \leq 2k$: the minimum over ℓ is $(r/k)^2$ in every tested case (C-E1). Exhaustive search over all 6,561 functions $2^{[3]} \to \{0, 1, 2\}$ finds 635 submodular ones, with minimum positive-optimum ratio $\frac{1}{2}$ at $(k, r) = (2, 1)$; all 4,096 directed unweighted graphs on four vertices give cut-function minima $\frac{1}{4}$ and $\frac{1}{3}$ for $(2, 1)$ and $(3, 1)$ (C-E2).

Open question 1 (C-O1: the central lower bound)**.** Does a universal constant $c > 0$ satisfy $\mathrm{Val}(r, k, f) \geq c\,(r/k)^2\,\mathrm{OPT}_{\leq k}$ for all nonnegative submodular f and all $1 \leq r \leq k$?

No proof and no asymptotic counterexample is known. The construction *suggests* the quadratic rate is the truth, and the finite evidence is consistent with it, but **the $\Omega((r/k)^2)$ lower bound is conjectured, not proved**. What is proved is the *upper* bound $(r/k)^2$ on the worst-case ratio. C-T5 caps any such c at $\frac{1}{2}$ when the comparator is $\mathrm{OPT}_{\leq k}$; using $\mathrm{OPT}_{=k}$ avoids that obstruction without settling the question. Because randomization cannot change the value (C-T1), a mixed-strategy argument alone cannot resolve it.

The obstacle. Forcing a valuable r-set, the monotone trick, no longer works because later accepted elements can destroy its value. Reserving overrides for harmful proposals sacrifices opportunities to collect goods. Random-subset lemmas do not automatically survive an adversary who observes past overrides and adapts future poison proposals. A proof would need a potential function that controls both the accepted damage and the remaining threats at once.

5.6 Slide 26 – Why Looping Happens, and Why Off-Limits Is the Natural Fix

Two kinds of repetition. An agent that loops on a fix earns $f(a \mid S) = 0$ once $a \in S$ under any submodular f; this is an agent that does not model diminishing returns (Lecture 7.1's 1,200 agents re-posting the same exploit). An agent that loops on an attack is, in the IGG, proposing a blocked element, which is illegal, so the attack has zero marginal value for the agent too. Without the rule it would have *positive* value, one override drained per repeat. The off-limits model therefore makes the monitor's world consistent with the agent's: where repeating an action is worthless, repeating an attack should be worthless as well.

What the model leaves out. Imperfect detection, proposals that change future proposals, useful-work constraints on the monitor, and several harmful submissions all require a different game. The course notes mark this as C-O2: the clean formulas above do not transfer without new mechanics, and PA5 does not test them.

6 Part V: PA5 Preview and Wrap-Up

6.1 Slide 28 – PA5 Reminder: Reproduce AI Control Monitoring

PA5 is due Saturday, December 5 (Week 14), 100 points, about 120 API calls.

1. **Environment (10 pts).** Thirty coding tasks (HumanEval 0–29 or MBPP), a subprocess sandbox with a 10-second timeout, and at least three scripted backdoor patterns that pass the provided tests but fail an edge case.
2. **Agent and monitor (25 pts).** An untrusted agent in honest and adversarial modes; a trusted monitor that sees the task and the code but not the tests, returning $s \in [0, 1]$; accept iff $s \leq \tau$.
3. **Safety–usefulness frontier (25 pts).** Sweep 21 thresholds; safety is the fraction of adversarial outputs flagged, usefulness the fraction of honest outputs accepted; 100 bootstrap resamples; baselines are no monitoring, full monitoring, and random scores; mark the equal-error point.
4. **Submodular budget analysis (20 pts).** $f(S) = 1 - \prod_{j \in S}(1 - p_j d_j)$; greedy versus random for $k = 1, \ldots, 30$; the $(1 - 1/e)f(V)$ reference line; verify diminishing gains; discuss a concentrating adversary and adaptive submodularity.
5. **Analysis (20 pts).** Monitor effectiveness; comparison with Greenblatt et al.; submodularity in practice; the PA1–PA5 arc.

Task 4's coverage function is monotone submodular by construction. Task 4.5 asks what breaks when the p_j are chosen by an adversary who has seen your S; today supplies the vocabulary for that paragraph.

6.2 Slide 29 – How Today's Ideas Show Up in PA5

Table, read as prose. The slide maps five lecture ideas to PA5 tasks. Threshold monitoring from Greenblatt §2.2 maps to Tasks 2–3: your frontier is Greenblatt's Figure 2 in miniature, and Griffin §6 says a higher audit FPR can Pareto-improve. Greedy with a monotone coverage function maps to Tasks 4.1–4.4: f is monotone submodular because each term is a product of independent miss probabilities, and you should cite the proof. The max–min view and SATURATE map to Task 4.5: if the adversary chooses p_j after seeing S, the objective is $\min_i f_i(S)$ and the $(1 - 1/e)$ line no longer applies. Poison pills map to Task 5.3: name a scenario in which an accepted output destroys earlier value, such as a shared library or credentials, and note that the r/k intuition is then wrong by a square. Off-limits and shutdown map to Task 5.2: shutdown on a caught backdoor is the global off-limits rule, your reproduction has no shutdown, and you should say what that costs.

PA5's budget analysis is monotone by design. The nonmonotone case is for your discussion section, not your code.

6.3 Slide 30 – Summary: Three Regimes, and Where Each Tool Fits

Figure description. Three boxes left to right along an arrow labeled "adversary sophistication." Regime 1, reliability (green): fixed f, passive ground set, choose a set; tools are greedy with the $(1 - 1/e)$ guarantee and lazy greedy, from Lecture 1. Regime 2, adaptive (blue): inspections reveal $\phi(v)$, choose a policy; the tool is adaptive greedy with the same guarantee over policies, from Lecture 1. Regime 3, adversarial (red): f or S is chosen with an enemy; tools are SATURATE for

worst case over scenarios, randomize-and-commit for hidden draws, the IGG with r/k for monotone objectives and r^2/k for poison pills, and the off-limits rule, from Lecture 2.

Three sentences to keep. Check monotonicity before you trust a greedy bound: it is the promise that nothing can undo secured value. Against an adversary, the budget fraction is the guarantee, and nonmonotonicity squares it. Make blocking permanent, or the adversary will loop you out of budget.

6.4 Slide 31 – What Comes Next

PA5 asks you to build the monitor, sweep the threshold, run greedy on a coverage function you have verified is monotone submodular, and write the paragraph about the adversary who has read this lecture. The open problem C-O1 is a legitimate research question: is the worst-case IGG ratio $\Theta((r/k)^2)$ for all nonnegative submodular objectives? Module 9 turns to poker (Libratus, Pluribus), Cicero, and agent benchmarks: where the systems we would monitor come from.

Appendix: Two Further Worked Checks

A. The ϕ formula at every ℓ for $k = 4$, $r = 2$. Table, read as prose. For $\ell = 1$: $b_* = \max\{0, 1 - 2\} = 0$, value 4. For $\ell = 2$: $b_* = 0$, value 4. For $\ell = 3$: $b_* = 1$, value $(4 - 1)(1 - 1/3) = 2$. For $\ell = 4$: $b_* = 2$, value $(4 - 2)(1 - 2/4) = 1$. For $\ell = 5$: $b_* = 2$, value $2(1 - 2/5) = 1.2$. For $\ell = 6$: value $2(1 - 2/6) \approx 1.33$. For $\ell = 8$: value 1.5. As $\ell \to \infty$ the value tends to $2 = r$, the monotone answer, because each poison does vanishing damage.

The minimum over ℓ is $1 = r^2/k$ at $\ell = k$, as C-T3 predicts, and the sequence decreases on $r < \ell \leq k$ and increases for $\ell \geq k$.

B. Why the agent cannot do better by mixing goods and poisons. In Example 2, suppose the agent proposes g_3 in round 1 instead of a poison. The monitor accepts (a free good). The agent now has three rounds and the monitor still has two overrides. Proposing poisons in rounds 2–4 yields two overrides and one accepted poison: final set $\{g_3, g_4, g_5, p\}$ with value $3 \cdot \frac{3}{4} = 2.25 > 1$. Every good the agent volunteers raises the monitor's payoff, confirming that the all-poison proposal sequence in the proof of C-T3 is the agent's best response.

References

- Krause, A., and Golovin, D. (2014). "Submodular Function Maximization." In *Tractability: Practical Approaches to Hard Problems*, Ch. 3, Cambridge University Press. §3.4.
- Krause, A., McMahan, H. B., Guestrin, C., and Gupta, A. (2008). "Robust Submodular Observation Selection." *Journal of Machine Learning Research* 9: 2761–2801. The SATURATE algorithm.
- Krause, A., Roper, A., and Golovin, D. (2011). "Randomized Sensing in Adversarial Environments." *IJCAI 2011.*
- Orlin, J. B., Schulz, A. S., and Udwani, R. (2018). "Robust Monotone Submodular Function Maximization." *Mathematical Programming* 172: 505–537.
- Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. (1978). "An Analysis of Approximations for Maximizing Submodular Set Functions—I." *Mathematical Programming* 14(1): 265–294.
- Kuhnle, A. (2026). "Interventionist Greedy Game (IGG)." Course draft; and "IGG poisoning and strategic audit timing," course notes, November 2026. Source of C-T1–C-T7, C-E1–C-E2, C-O1–C-O2. Unpublished.

- Greenblatt, R., Shlegeris, B., Sachan, K., and Roger, F. (2024). "AI Control: Improving Safety Despite Intentional Subversion." *ICML 2024*, PMLR 235: 16295–16336; arXiv:2312.06942. §2.2, Table 1.
- Griffin, C., Thomson, L., Shlegeris, B., and Abate, A. (2024). "Games for AI Control: Models of Safety Evaluations of AI Deployment Protocols." arXiv:2409.07985. §6–8.
- Golovin, D., and Krause, A. (2011). "Adaptive Submodularity: Theory and Applications in Active Learning and Stochastic Optimization." *Journal of Artificial Intelligence Research* 42: 427–486.
- Mualem, L., Elenberg, E., Feldman, M., and Karbasi, A. (2024). "Submodular Minimax Optimization: Finding Effective Sets." *AISTATS 2024*. Related minimax formulation (course notes, C-K3).
- Avenhaus, R., von Stengel, B., and Zamir, S. (2002). "Inspection Games." *Handbook of Game Theory*, Vol. 3, Ch. 51.