Efficient Computation of Approximate Equilibria in Discrete Colonel Blotto Games

Dong Quan Vu, Patrick Loiseau, Alonso Silva

Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Efficient Computation of Approximate Equilibria in Discrete Colonel Blotto Games

Dong Quan Vu[1], **Patrick Loiseau**[2,3], **Alonso Silva**[1]
[1] Nokia Bell Labs, Nokia Paris-Saclay, 91620 Nozay, France
[2] Univ. Grenoble Alpes, CNRS, Inria, Grenoble INP, LIG, 38000 Grenoble, France
[3] Max Planck Institute for Software Systems (MPI-SWS), Saarbrücken, Germany
quan_dong.vu@nokia.com, patrick.loiseau@univ-grenoble-alpes.fr, alonso.silva@nokia-bell-labs.com

Abstract

The Colonel Blotto game is a famous game commonly used to model resource allocation problems in many domains ranging from security to advertising. Two players distribute a fixed budget of resources on multiple battlefields to maximize the aggregate value of battlefields they win, each battlefield being won by the player who allocates more resources to it. The continuous version of the game—where players can choose any fractional allocation—has been extensively studied, albeit only with partial results to date. Recently, the discrete version—where allocations can only be integers—started to gain traction and algorithms were proposed to compute the equilibrium in polynomial time; but these remain computationally impractical for large (or even moderate) numbers of battlefields. In this paper, we propose an algorithm to compute very efficiently an *approximate* equilibrium for the discrete Colonel Blotto game with many battlefields. We provide a theoretical bound on the approximation error as a function of the game's parameters, in particular number of battlefields and resource budgets. We also propose an efficient dynamic programming algorithm to compute the best-response to any strategy that allows computing for each game instance the actual value of the error. We perform numerical experiments that show that the proposed strategy provides a fast and good approximation to the equilibrium even for moderate numbers of battlefields.

1 Introduction

The past decade has seen a rising interest in using game-theoretic models for security problems, see e.g., [Tambe, 2011]. As the modern world is facing increasingly important security threats, such models are interesting because they allow deriving defenses that are optimized against adaptive adversaries for different specific applications such as patrolling or screening problems, see e.g., [Conitzer and Sandholm, 2006; Bošanský *et al.*, 2011; Yang *et al.*, 2013; Letchford and Conitzer, 2013; Fang *et al.*, 2015; Gan *et al.*, 2015; Brown *et al.*, 2016] amongst others. Several solutions of game-theoretic models have been implemented in real-world defense applications with positive results, validating the importance and relevance of game theory for security problems [Pita *et al.*, 2011; Yin *et al.*, 2012].

Recently, the community started to gain interest in the celebrated *Colonel Blotto game*. In the Colonel Blotto game, two players (often referred to as colonels) choose how to distribute a fixed budget of resources (often called troops or soldiers) on a number of battlefields. Each battlefield has a given value and is won by the player who allocates more resources to it; each player maximizes the sum of values of battlefields he wins. As a simple and elegant model for strategic resource allocation problems, the Colonel Blotto game (and in particular the characterization of its equilibrium) has important applications in many domains including security (allocation of defense/attacks resources), but also politics (allocation of campaign resources or lobbying resources), industrial operations (allocation of R&D resources) or advertisement (allocation of ad budgets). Its continuous version, where players can choose any fractional allocation, has received high attention from the economics community since its first introduction in 1921. However, only partial results are known to date; in particular, the equilibrium characterization in the general case of parameters configuration still remains as a challenging open question (see related works below).

The discrete version of the Colonel Blotto game (where allocations can only be integers), which is meaningful in applications where individual troops cannot be divided, started to gain traction much more recently in the algorithmic game theory community. Since it is a finite constant-sum game, it can in principle be solved numerically in general cases through linear programming. However, standard solutions to compute the Nash equilibria face the issue that the strategy space of the players grows exponentially with the number of battlefields and the number of troops. To tackle this problem, two algorithms were proposed in the last two years in [Ahmadinejad *et al.*, 2016] and [Behnezhad *et al.*, 2017]. Both algorithms rely on transforming the linear program formulation which significantly improves the complexity. Yet, these algorithms still become computationally impractical when the number

of battlefields and/or the number of troops is large (e.g., it takes over 1 day to solve instances with 45 battlefields and 75 troops in our simulations). In applications such as security or politics, it is frequent that these parameters are large. In that case, the question remains open: how to efficiently compute an equilibrium strategy?

In this work, we take a different approach that relaxes the equilibrium notion considered to gain dramatically in computational efficiency: we propose and analyze an algorithm to compute very efficiently an approximate Nash equilibrium for the discrete Colonel Blotto game with many battlefields and troops. Specifically, denoting by n the number of battlefields and by p the number of troops of the stronger player, we make the following contributions:

1. We propose a mixed strategy, which we term *discrete independently uniform* strategy (hereinafter DIU strategy). Inspired by partial results on the equilibrium marginals in the continuous version of the game, our strategy first generates allocations independently on each battlefield with appropriately defined distributions, then rescales them and performs a rounding process that guarantees the integer constraint while maintaining the budget constraint. This has complexity $\mathcal{O}(n)$.
2. Denoting V_n the total payoff of the game, we prove that the DIU strategy is an $\bar{\varepsilon}V_n$-equilibrium. We give a theoretical bound on $\bar{\varepsilon}$ that shows how good of an approximation to the Nash equilibrium the DIU strategy is, depending on n and p.
3. We propose a dynamic programing algorithm to compute a best-response of a player to a given set of marginals (on each battlefield) of the adversary. Our algorithm has complexity $\mathcal{O}(p^2 \cdot n)$. It allows us to efficiently compute the actual value of $\bar{\varepsilon}$ for any given set of parameters of the game.
4. We perform numerical experiments that illustrate that the proposed DIU strategy provides a good approximation to the equilibrium even for a relatively moderate number of battlefields. We also compare our solution to the exact equilibrium found by the algorithm of [Behnezhad *et al.*, 2017] both in terms of the payoff obtained and the computation time.

Our proposed algorithm computes directly a realization of the strategy, rather than computing the mixed strategy of the players (i.e., the equilibrium distribution) as in [Ahmadinejad *et al.*, 2016; Behnezhad *et al.*, 2017]. In practice, this is what a player would need to generate his allocation. Yet, even though the distribution from which our strategy is drawn is only implicitly defined, we prove that it provides an approximate equilibrium. Besides, it is possible to generate this distribution with arbitrary precision simply by generating many realizations independently using our proposed algorithm.

Note finally that our approximate equilibrium is not "universal" in the sense that it does not provide a good approximation for any set of parameters. Roughly, the approximation is good if the numbers of battlefields and troops are large. Yet, our proposed approximate equilibrium provides an important contribution because (i) it is a realistic case in several applications and (ii) it is precisely the case where exact equilibrium computation algorithms are computationally infeasible.

Related works The Colonel Blotto game was first introduced in 1921 in its continuous version [Borel, 1921]. Since then, a number of partial solutions have been proposed. The case of symmetric players (with identical budgets) with an arbitrary number of battlefields was solved by [Borel and Ville, 1938; Gross and Wagner, 1950; Gross, 1950], who also provided a solution for the asymmetric case with two battlefields (see also [Laslier, 2002]). In 2006, a solution was found for the case of asymmetric players with an arbitrary number of battlefields but only for homogeneous battlefields (all with identical value) [Roberson, 2006]. This solution was extended in [Schwartz *et al.*, 2014] to heterogeneous battlefields but only under some restrictions. Today, the general case of asymmetric players with heterogeneous battlefields remains unsolved for the continuous Blotto game.

Many extensions and variants of the continuous Colonel Blotto game have been studied: relaxing the budget constraint [Myerson, 1993; Kovenock and Roberson, 2015], considering other objectives [Laslier, 2005] or analyzing sequential moves [Powell, 2009; Rinott *et al.*, 2012]. Many applications have also been considered, e.g., to politics [Laslier, 2002], economics [Kovenock and Roberson, 2012], security [Powell, 2009], and social networks [Masucci and Silva, 2014].

The discrete version of the Colonel Blotto game has received far less attention. Partial results for special cases are proven in [Hart, 2008; Hortala-Vallve and Llorente-Saguer, 2012]. In the last two years, the discrete Colonel Blotto game attracted interest from the algorithmic game theory community. Two algorithms were proposed to compute the Nash equilibrium for the general asymmetric and heterogeneous case. A first algorithm was proposed in [Ahmadinejad *et al.*, 2016], based on a reduction to an exponential-size linear program and a clever use of the Ellipsoid method to solve it in polynomial time. In [Behnezhad *et al.*, 2017], another algorithm was proposed that obtains a polynomial-size linear program and solves it using the Simplex method. Although both algorithms are providing polynomial-time/size solutions to find the Nash equilibrium of the discrete Colonel Blotto game, they remain computationally intractable in practice with large numbers of troops and/or battlefields.

2 Problem Formulation

Game model We consider a discrete Colonel Blotto game between two players denoted A and B. Each player has a fixed amount of troops (or budget), denoted m and p for A and B respectively, where $m, p \in \mathbb{N}$. Without loss of generality, we assume that A is the weak player, i.e., $m \leq p$. Throughout the paper, we denote by $\lambda := \frac{p}{m}$ the ratio of players budgets.

The game is a one-shot game where players simultaneously allocate their troops to n battlefields ($n \geq 3$). A pure strategy of player A is a vector $\hat{\boldsymbol{x}}^{\boldsymbol{A}} \in \mathbb{N}^n$, with integer elements $\hat{x}_i^A \geq 0$ representing the allocation to battlefield i ($i = 1, 2 \ldots, n$) and satisfying the constraint $\sum_{i=1}^n \hat{x}_i^A \leq m$. Similarly, a pure strategy of player B is a vector $\hat{\boldsymbol{x}}^{\boldsymbol{B}} \in \mathbb{N}^n$ such that the constraint $\sum_{i=1}^n \hat{x}_i^B \leq p$ holds.

Each battlefield i is commonly assessed by players with a fixed value $v_i > 0$. Values can be heterogeneous across battlefields, but we assume that each value belongs to a bounded range: $v_i \in [v_{\min}, v_{\max}]$, with $0 < v_{\min} \leq v_{\max}$. We denote by $V_n = \sum_{i=1}^n v_i$ the total value of all battlefields.

Once players have allocated their troops, the player who has the higher allocation to battlefield i wins that battlefield and gains its whole value v_i. In case of a tie, i.e., if $x_i^A = x_i^B$, player A gains αv_i and player B gains $(1-\alpha)\, v_i$ for some constant $\alpha \in [0,1]$ fixed. This is a very general tie-breaking rule that includes all previously considered rules such as sharing 50-50 ($\alpha = 0.5$) or giving the battlefield to the stronger player ($\alpha = 0$). Each player chooses his strategy to maximize his own payoff equal to the sum of gains on each battlefield. Resources not allocated to any battlefield have no outside value.

Additional notation For brevity, we denote by $\mathcal{CB}_n^{m,p}$ the discrete Colonel Blotto game with n battlefields where player A has m troops and player B has p troops. We denote by $\Pi_A(\sigma,\gamma)$ and $\Pi_B(\sigma,\gamma)$ the expected payoffs of players A and B when playing strategies σ and γ respectively. Throughout the paper, we denote by F_X the distribution function of a random variable X. We use a hat ($\hat{z}$) to denote integers.

Approximate equilibrium We use the standard definition of an approximate equilibrium as follows: given $\varepsilon \geq 0$, a strategy profile (σ^*, γ^*) is an *ε-equilibrium* of the game $\mathcal{CB}_n^{m,p}$ if $\Pi_A(\sigma,\gamma^*) \leq \Pi_A(\sigma^*,\gamma^*) + \varepsilon$ and $\Pi_B(\sigma^*,\gamma) \leq \Pi_B(\sigma^*,\gamma^*) + \varepsilon$ for any strategy σ and γ of players A and B. When it is unnecessary to emphasize the approximation error, we use the term *approximate equilibrium*.

3 Main Results

In this section, we give our proposed strategy and main theoretical results showing that it is an approximate equilibrium.

3.1 The DIU Strategy

Considering the game $\mathcal{CB}_n^{m,p}$, we propose a mixed strategy that we call *Discrete Independently Uniform* strategy (DIU strategy), which will be proven to be an approximate equilibrium of the game. Intuitively, under the DIU strategy, players first draw *independently* numbers from some particular *uniform*-type distributions; then they rescale these numbers to guarantee the budget constraints; finally, they use a specific rounding process to ensure the *discrete* requirements.

To formalize the DIU strategy definition, we introduce, for any $i \in \{1,2,\ldots,n\}$, the uniform-type distributions:

$$F_{A_i^*}(x) := \left(1-\frac{1}{\lambda}\right) + \frac{x}{2\frac{v_i}{V_n}\lambda}\frac{1}{\lambda}, \forall x \in \left[0, 2\frac{v_i}{V_n}\lambda\right], \quad (3.1)$$

$$F_{B_i^*}(x) := \frac{x}{2\frac{v_i}{V_n}\lambda}, \forall x \in \left[0, 2\frac{v_i}{V_n}\lambda\right]. \quad (3.2)$$

We observe that $F_{B_i^*}$ is the (continuous) uniform distribution on $[0, 2v_i\lambda/V_n]$ and $F_{A_i^*}$ is the distribution where we set a probability mass $(1-1/\lambda)$ at 0 and uniformly distribute the remaining mass on $(0, 2v_i\lambda/V_n]$. We define the rounding function $r^m : \left[0, \frac{p}{m}\right] \to \left\{0, \frac{1}{m}, \frac{2}{m}, \ldots, \frac{p}{m}\right\}$, such that $\forall x, r^m(x) = \frac{\hat{x}}{m}$, where $\hat{x} \in \mathbb{N}$ is uniquely determined and satisfies $\frac{\hat{x}}{m} - \frac{1}{2m} \leq x < \frac{\hat{x}}{m} + \frac{1}{2m}$.

Algorithm 1: DIU strategy generation algorithm.

Input: $n, m, p \in \mathbb{N}$, and $v \in [v_{\min}, v_{\max}]^n$
Output: $\hat{\boldsymbol{x}}^{\boldsymbol{A}}, \hat{\boldsymbol{x}}^{\boldsymbol{B}} \in \mathbb{N}^n$

1 $\lambda = p/m$
2 **for** $i = 1, 2, \ldots, n$ **do**
3 $\quad a_i = \begin{cases} 0 & \text{with probability } 1 - \frac{1}{\lambda} \\ \sim \mathcal{U}\left(0, \frac{2v_i}{V_n}\lambda\right] & \text{otherwise} \end{cases}$
4 **if** $\sum_{j=1}^n a_j = 0$ **then** repeat line 2
5 **for** $i = 1, 2, \ldots, n$ **do**
6 $\quad b_i = \sim \mathcal{U}\left[0, \frac{2v_i}{V_n}\lambda\right]$
7 $s_0^A = s_0^B = 0$
8 **for** $i = 1, 2, \ldots, n$ **do**
9 $\quad s_i^A = \sum_{k=1}^i \frac{a_k}{\sum_{j=1}^n a_j}$; $s_i^B = \sum_{k=1}^i \frac{b_k}{\sum_{j=1}^n b_j}\frac{p}{m}$
10 $\quad \hat{x}_i^A := m\left[r^m\left(s_i^A\right) - r^m\left(s_{i-1}^A\right)\right]$
11 $\quad \hat{x}_i^B := m\left[r^m\left(s_i^B\right) - r^m\left(s_{i-1}^B\right)\right]$

We can now give the formal definition of the DIU strategy.[1]

Definition 3.1 (The DIU strategy)**.** *In the game $\mathcal{CB}_n^{m,p}$, $\mathbf{DIU_A}$ (respectively, $\mathbf{DIU_B}$) is the **mixed** strategy where player A's allocation $\hat{\boldsymbol{x}}^{\boldsymbol{A}}$ (respectively, player B's allocation $\hat{\boldsymbol{x}}^{\boldsymbol{B}}$) is randomly generated from Algorithm 1.*

Remarks Algorithm 1 guarantees that the allocations are integers and satisfy the budget constraints (with equality, i.e., without any unallocated resource). More importantly, the $\mathrm{DIU_A}$ (resp., $\mathrm{DIU_B}$) strategy is only implicitly defined via Algorithm 1, that is to say it is the **joint distribution** of all allocations $\{\hat{x}_i^A\}_i$ (resp., $\{\hat{x}_i^B\}_i$). Each pure strategy output from Algorithm 1 is only one realization of the DIU strategy.

Algorithm 1 is easy to implement and runs very fast in expected time $\mathcal{O}(n)$. Note that the *for loop* in lines 2-4 is not guaranteed to end in finite time. However, the probability that the loop runs over k times is $(1-1/\lambda)^{kn}$ and converges to zero exponentially fast in k and n. To guarantee that the algorithm ends in finite time, it is possible to put a stopping criterion and assign an arbitrary allocation to player A if it is reached. As this will happen with increasingly low probability as n grows, it can be seen from the proof of Theorem 3.2 that the result will still hold. On the other hand, the summation $\sum_{j=1}^n b_j$ equals 0 only with probability zero, therefore we do not need an additional condition for the *for loop* in lines 5-6.

When applying the DIU strategy, player A's allocation to battlefield $i = 1, 2, \ldots, n$ follows the marginal distributions $F_{A_i^D}$ while player B's allocation follows $F_{B_i^D}$ whose corresponding random variables are defined as:

$$A_i^D = m\left[r^m\left(\sum\nolimits_{k=1}^{i} A_k^n\right) - r^m\left(\sum\nolimits_{k=1}^{i-1} A_k^n\right)\right], \quad (3.3)$$

[1] We use the term DIU strategy to commonly address $\mathrm{DIU_A}$ and/or $\mathrm{DIU_B}$ when unnecessary to emphasize a particular player.

$$B_i^D = m\left[r^m\left(\sum\nolimits_{k=1}^{i} B_k^n\right) - r^m\left(\sum\nolimits_{k=1}^{i-1} B_k^n\right)\right], \quad (3.4)$$

where for any $k = 1, 2, \ldots, n$,

$$A_k^n := \frac{A_k^*}{\sum_{j=1}^n A_j^*} \text{ and } B_k^n := \frac{B_k^*}{\sum_{j=1}^n B_j^*}\frac{p}{m}, \quad (3.5)$$

and random variables A_k^*, B_k^* have distributions (3.1)-(3.2).

We end this subsection by briefly describing the intuition behind the construction of the DIU strategy in Definition 3.1. First of all, from the equilibrium analysis done in [Roberson, 2006] and [Schwartz *et al.*, 2014] for the continuous Colonel Blotto game with n battlefields, where player A's budget is 1 and player B's budget is λ, the equilibrium marginal distributions of players' allocations to battlefield i follow distributions $F_{A_i^*}$ and $F_{B_i^*}$. That is to say, a joint distribution (if it exists) satisfying the constraint $\sum_{i=1}^n x_i^A \leq m$ and yielding the marginal distributions $\left\{F_{A_i^*}\right\}_i$ will be a feasible strategy of player A which constitutes an equilibrium of this continuous game. However, the construction of such a strategy (and even its existence) remains an open question.

On the other hand, by employing the DIU strategy in the discrete Colonel Blotto game, players' marginal allocations at battlefield i follow the distributions $F_{A_i^D}$ and $F_{B_i^D}$. These distributions are r^m-rounded from terms expressed by distributions $F_{A_i^n}$ and $F_{B_i^n}$, which in turn, uniformly converge towards $F_{A_i^*}$ and $F_{B_i^*}$ when $n \to \infty$. The key idea is that, the requirement to have discrete allocations in the discrete game is less and less significant when the granularity of the game increases (i.e. $\frac{m}{n}, \frac{p}{n} \to \infty$), which makes it similar to the continuous variant. Thus, based on the optimality in the continuous variant of $F_{A_i^*}$ against $F_{B_i^*}$ (and vice versa), we expect to have "near-optimality" in playing $\mathrm{DIU_A}$ strategy against $\mathrm{DIU_B}$ strategy (and vice versa), with any arbitrary error $\bar{\varepsilon}V_n > 0$, given large parameters n, m, p.

3.2 Approximate Equilibrium of The Discrete Colonel Blotto Game

Theorem 3.2. *The* DIU *strategy is an $\bar{\varepsilon}V_n$-equilibrium of the Colonel Blotto game $\mathcal{CB}_n^{m,p}$ ($m \leq p$), where $\bar{\varepsilon} \leq \max\{\tilde{\mathcal{O}}(n^{-1/2}), \mathcal{O}(n/m)\}$ and V_n is the total value across all battlefields.*[2]

The upper bound on $\bar{\varepsilon}$ given by this theorem is important because it allows us to evaluate the approximation error in terms of the number of battlefields and amount of troops. Indeed, we can look at Theorem 3.2 from a different perspective and interpret it by an equivalent statement:

Restatement of Theorem 3.2. *Fix $\lambda \geq 1$ and $\bar{\varepsilon} > 0$; there exists $N^* = \mathcal{O}\left(\bar{\varepsilon}^{-2}\ln(\bar{\varepsilon}^{-1})\right)$ such that for $n \geq N^*$, there exists $M^* = \mathcal{O}(n/\bar{\varepsilon})$ such that for $m \geq M^*$ and $p = m\lambda \in \mathbb{N}$, in the game $\mathcal{CB}_n^{m,p}$, for any pure strategies $\hat{\boldsymbol{x}}^{\boldsymbol{A}}$ and $\hat{\boldsymbol{x}}^{\boldsymbol{B}}$ of player A and B,*

$$\Pi_A\left(\hat{\boldsymbol{x}}^{\boldsymbol{A}}, \mathrm{DIU_B}\right) \leq \Pi_A\left(\mathrm{DIU_A}, \mathrm{DIU_B}\right) + \bar{\varepsilon}V_n, \quad (3.6)$$

$$\Pi_B\left(\mathrm{DIU_A}, \hat{\boldsymbol{x}}^{\boldsymbol{B}}\right) \leq \Pi_B\left(\mathrm{DIU_A}, \mathrm{DIU_B}\right) + \bar{\varepsilon}V_n. \quad (3.7)$$

At a high level, this confirms the intuition that if the number of battlefields and the budgets are large enough, then the DIU strategy yields a near-optimal payoff against the opponent's DIU strategy. The precise result shown in this theorem, however, goes much beyond merely showing this convergence and it is interesting and non-trivial in a number of ways. First, Theorem 3.2 tells us exactly how the parameters m and n should be to reach a given level of approximation. We notice in particular that if the ratio m/n is small, then the approximation may not be good, however large n gets.

Second, Theorem 3.2 involves a double limit, with two growing parameters (n and m), and it identifies a precise *scaling regime* (i.e., ratio between the two growing parameters) under which the convergence holds. Here, Theorem 3.2 shows that the DIU strategy converges towards an equilibrium as soon as m grows at least as fast as $n^{3/2}$. This implies that, if we first make m grow to infinity, and then make n grow to infinity, the result will hold. However, the reverse is not true: if n grows first or simply if m grows too slowly compared to n, then the DIU does not converge towards an equilibrium. Intuitively, if the number of troops is low compared to the number of battlefields, then the average number of troops per battlefield at equilibrium becomes low and the DIU strategy based on a discretization of a uniform-type distribution is no longer close to optimal.

Note that due to space constraints, we limited the statement of our result to emphasize the dependence on n and m but our proof also allows extracting the dependence of $\bar{\varepsilon}$ on $v_{\min}, v_{\max}$ and λ. One then observes that the convergence is slower if $v_{\max}/v_{\min}$ is larger (i.e., the battlefields heterogeneity is higher) and if λ is larger (i.e., the players asymmetry is higher). Note that we have written the above discussion with m, but the exact same holds with p instead.

Finally, we remark that $\mathcal{CB}_n^{m,p}$ is a constant-sum game. Therefore, by using inequalities (3.6) and (3.7), we can straightforwardly prove that the DIU strategy is an approximate max-min strategy of the game. This is presented as the following corollary of Theorem 3.2.

Corollary 3.3. *$\forall\lambda \geq 1, \forall\bar{\varepsilon} > 0$, $\exists N^* = \mathcal{O}(\bar{\varepsilon}^{-2}\ln(\bar{\varepsilon}^{-1}))$: $\forall n \geq N^*$, $\exists M^* = \mathcal{O}(n/\bar{\varepsilon})$: $\forall m \geq M^*$, $p = m\lambda \in \mathbb{N}$, in the game $\mathcal{CB}_n^{m,p}$, for any strategies σ_A and σ_B of players A and B,*

$$\min_\gamma \Pi_A(\sigma_A, \gamma) \leq \min_\gamma \Pi_A(\mathrm{DIU_A}, \gamma) + \bar{\varepsilon}V_n, \quad (3.8)$$

$$\min_\sigma \Pi_B(\sigma, \sigma_B) \leq \min_\sigma \Pi_B(\sigma, \mathrm{DIU_B}) + \bar{\varepsilon}V_n. \quad (3.9)$$

This corollary ensures that the DIU strategy gives the near-optimal payoff to any player $Q \in \{A, B\}$ even in the worst-case (when the opponent $-Q$ plays the strategy that minimizes Q's payoff). This emphasizes the fact that players can "safely" use the DIU strategy in practice.

4 Numerical Evaluation

In this section, we turn to the numerical computation of quantities related to the DIU strategy, in particular to evaluate the quality of the approximation it gives depending on the game's parameters.

[2] The $\tilde{\mathcal{O}}$ notation is a variant of the big-$\mathcal{O}$ notation that "ignores" logarithmic factors.

4.1 A Dynamic-Programming Algorithm for the Best Response

First, computing the value of $\bar{\varepsilon}$ (or how close a given mixed strategy of player A is to the equilibrium) requires finding player B's optimal allocation given that player A's allocation to battlefield $i = 1, 2 \ldots, n$ follows a given marginal distribution $\{G_i\}_{i=1,2,\ldots,n}$. This itself is a non-trivial problem since there is in principle an exponential number of possible allocations to investigate. We propose an efficient algorithm based on dynamic programming [Bertsekas, 2017] to solve this problem. This is formally presented as the following proposition.

Proposition 4.1. *Algorithm 2 finds a best response strategy of player B and his optimal payoff against any set of player A's marginals with complexity* $\mathcal{O}\left(p^2 \cdot n\right)$.

Note that, although our primary motivation is to compute a best-response of a player to the DIU strategy, the algorithm has broader applicability since it works for any mixed strategy of the adversary. We discuss here the main intuition behind Algorithm 2 and give a descriptive proof of Proposition 4.1. Note firstly that the algorithm is presented here with tie-breaking parameter $\alpha = 0$ for simplicity but could straightforwardly be adapted to any tie-breaking rule. In this algorithm, $H(j, i)$ denotes the expected payoff that player B gains from battlefield i by allocating j troops to it, which is computed via the equation in line 4. More specifically, since $\alpha = 0$, by allocating j troops to battlefield i, player B wins the value v_i if j is at least equal to player A's allocation. Since G_i is the marginal distribution of player A in this battlefield, then $G_i(j)$ is exactly the probability of this event, which implies the expected gain of player B. There are $(p + 1)\, n$ terms $H(j, i)$ to compute yielding the complexity $\mathcal{O}(p \cdot n)$ to do so.

On the other hand, we denote $\Pi(j, i)$ the optimal payoff of player B when he is allowed to spend j troops over the set $\{1, 2, \ldots, i\}$ of battlefields; thus, $\Pi(p, n)$ is exactly the best-response payoff of player B. The computation of $\Pi(j, i)$ is done by working backwards with the recursive equation given in line 5. To spend j troops over i battlefields $\{1, 2, \ldots, i\}$, player B has to choose $k \in \{0, 1, \ldots, j\}$ troops to allocate across the first $i - 1$ battlefields (whose optimal payoff is denoted $\Pi(k, i - 1)$), and put the remaining $(j - k)$ troops on $i^{th}-$battlefield (which induces the payoff $H(j - k, i)$). He then optimizes the payoff to find $\Pi(j, i)$ by selecting the number k which maximizes the summation between the payoffs gained from these two parts. There are $\mathcal{O}(p \cdot n)$ terms $\Pi(j, i)$ needed to compute, each is done by comparing between at most $(p + 1)$ terms; thus it yields the complexity $\mathcal{O}(p^2 \cdot n)$ to do so. Finally, the algorithm finds a best response strategy yielding this optimal payoff with complexity $\mathcal{O}(p \cdot n)$ as in lines 7-9. Hence, we conclude the proof of Proposition 4.1.

Reversing the roles of A and B, we can construct a similar algorithm with complexity $\mathcal{O}\left(m^2 \cdot n\right)$ to find the best response payoff of player A against any given set of player B's marginals. Note also that the algorithm is presented here with $\alpha = 0$ for simplicity but it could straightforwardly be adapted to any tie-breaking rule.

Algorithm 2: Dynamic programing algorithm searching for player B's best-response (tie-breaking rule $\alpha = 0$).

Input: $n, m, p \in \mathbb{N}$, $v \in [v_{\min}, v_{\max}]^n$ and marginals $\{G_i\}_{i=1,2,\ldots,n}$ of player A

Output: Payoff $\Pi(p, n)$ and BR strategy $\left\{\hat{x}_1^B, \cdots, \hat{x}_n^B\right\}$

1 **for** $j = 0, 1, \ldots, p$ **do**
2 $\quad \Pi(j, 0) = 0$
3 $\quad$ **for** $i = 1, 2, \ldots, n$ **do**
4 $\quad\quad H(j, i) = v_i G_i(j)$
5 $\quad\quad \Pi(j, i) = \max\limits_{k=0,\ldots,j} \{\Pi(k, i-1) + H(j-k, i)\}$
6 $j = p$
7 **for** $i = n, n-1, \ldots, 1$ **do**
8 $\quad \hat{x}_i^B = \arg\max\limits_{k=0,1,\ldots,j} \{\Pi(j-k, i-1) + H(k, i)\}$
9 $\quad j = j - \hat{x}_i^B$

4.2 Numerical Experiments

In practice, we first observe that a pure strategy instructing players to allocate their resources following the DIU strategy can be generated from Algorithm 1 in time $\mathcal{O}(n)$, which is negligible even for extremely large values of the parameters.

On the other hand, since the marginal allocations at battlefield i under the DIU strategy, $F_{A_i^D}$ and $F_{B_i^D}$, are not known in closed-form, we approximate them by their corresponding empirical CDFs denoted $\bar{F}_{A_i^D}$ and $\bar{F}_{B_i^D}$ computed by drawing "many" realizations of the DIU strategy from Algorithm 1. Indeed, it is known by the Glivenko-Cantelli theorem [Vaart, 1998] that the empirical CDF converges uniformly towards the actual CDF, with a maximum difference in $\mathcal{O}(K^{-1/2})$ where K is the number of realizations drawn. Then, to guarantee that the approximation of the DIU's CDF by its empirical CDF does not affect the computed value of $\bar{\varepsilon}$, we only need to take $K \geq \mathcal{O}(n)$ (since $\bar{\varepsilon}$ is of the order $\tilde{\mathcal{O}}(n^{-1/2})$ according to the previous section). Overall, generating a good approximation of the DIU's marginal distribution therefore takes time $\mathcal{O}(n^2)$, still negligible even for large values. Finally, to compute $\bar{\varepsilon}$, for each player $Q \in \{A, B\}$, we compare the expected payoff $\Pi_Q(\mathrm{DIU_A}, \mathrm{DIU_B})$ to player Q's best-response payoff obtained from Algorithm 2 against the set of marginal distributions $\left\{\bar{F}_{(-Q)_i^D}\right\}_i$ of player $-Q$.

We construct several numerical experiments using R to illustrate the efficiency of using the DIU strategy as an approximate equilibrium of discrete Colonel Blotto games.[3] Our experiments run on a computer with an Intel core i5-7500U 2.60GHz processor and 8GB of RAM. In all the experiments, we keep $\alpha = 0$ and $\lambda = p/m$ fixed, thus the values of m and p always have the same growth rate (up to the multiplicative constant λ); and we vary n and m. For each set of values n, m, p, we independently generate a value for each battlefield uniformly distributed in $[v_{\min}, v_{\max}]$, with $v_{\min} = 1$ and $v_{\max} = 8$. Then, for each instance of

[3] Our code for these experiments can be found at https://github.com/dongquan11/Approx_discrete_Blotto

$n, m, p, (v_1, v_2, \ldots, v_n)$ we run the simulation 3 times, each time computing the eCDFs with $K = 10 \cdot n$ to ensure not to affect the evaluation on $\bar{\varepsilon}$ and running Algorithm 2 with these eCDFs to compute $\bar{\varepsilon}$, and take the average of results.

Figure 1(a) shows the results. We first notice that when m (and p) increases, the error $\bar{\varepsilon}$ generally decreases in consistency with Theorem 3.2. Moreover, when m is relatively small, the error $\bar{\varepsilon}$ is higher with instances having higher number of battlefields n. This is consistent with Theorem 3.2, stating that when the ratios m/n and p/n are low (they decrease when n increases), the upper bound of $\bar{\varepsilon}$ is not good. For instances with higher values of m, these ratios are sufficiently large to ensure that $\bar{\varepsilon}$ decreases when either m or n (or both) increase. This interpretation is also consistent with the results shown in Figure 1(b). When the value of n increases, at the beginning where the ratio m/n is still sufficiently large, $\bar{\varepsilon}$ decreases. However, since we keep m (and p) fixed in this experiment, the ratio m/n gradually decreases, which makes the errors to eventually get worse. Recall that, for each experiment presented here, we independently generate a value for each battlefield uniformly distributed in $[v_{\min}, v_{\max}]$; this process explains the randomness observed in the plots.

We finally compare our work with the algorithm proposed in [Behnezhad *et al.*, 2017],[4] which finds an exact equilibrium of the game (we denote it *Algorithm EQ*). Table 1 shows the computation time of evaluating the error in using DIU strategy and elapsed time of Algorithm EQ for several instances. We observe that it takes remarkably less time to compute the DIU strategy payoffs and give an upper bound of the potential error by using Algorithm 2. Note that the computation times shown here include the time to compute the empirical CDF of the DIU strategy by drawing sufficiently many realizations (recall that we take $K = 10 \cdot n$ to ensure not to affect the evaluation on $\bar{\varepsilon}$) and the elapsed time of Algorithm 2. Moreover, the last column of Table 1 shows that the DIU strategy payoffs are very close to the exact equilibrium payoffs, even for instances with small values of the parameters n, m and p.

In conclusion, it is important to note that we do not claim that our algorithm can replace more efficiently the algorithm of [Behnezhad *et al.*, 2017] (in fact, we are not computing the same thing). However, our results show that, for large values of n, m and p, the DIU strategy, which can be computed very efficiently, can be safely used by the players as it provides a good approximation to the equilibrium.

5 Sketch of The Proof of Theorem 3.2

In this section, we give the main elements of the proof of Theorem 3.2 in its restatement. We note that, although the main idea behind the DIU strategy is quite simple, the proof of this theorem is non-trivial and requires careful analysis to achieve the upper bound on $\bar{\varepsilon}$. Due to space constraints, we only give the very rough steps of the proof of inequality (3.6); and we omit less important or more technical details. A complete proof can be found in [Vu *et al.*, 2018].

[4] We use the authors' implementation from https://github.com/Soben713/ColonelBlotto

Step 1: We start by rewriting (3.6). If player A plays $\mathrm{DIU_A}$ and allocate $\hat{y}$ at battlefield i against player B's $\mathrm{DIU_B}$ strategy, he strictly wins battlefield i with probability $F_{B_i^D}(\hat{y}-1)$ and has a tie with probability $P(B_i^D = \hat{y})$. Hence, according to our general tie-breaking rule,

$$\begin{aligned}\Pi_A(\mathrm{DIU_A}, \mathrm{DIU_B}) =& \sum_{i=1}^{n}\left[v_i \sum_{\hat{y}=0}^{m} F_{B_i^D}(\hat{y}-1) P\left(A_i^D = \hat{y}\right)\right] \\ &+ \sum_{i=1}^{n}\left[\alpha v_i \sum_{\hat{y}=0}^{m} P\left(B_i^D = \hat{y}\right) P\left(A_i^D = \hat{y}\right)\right] \\ &:= \Pi_1 + \Pi_2. \end{aligned} \tag{5.1}$$

Similarly, $\Pi_A\left(\hat{\boldsymbol{x}}^{\boldsymbol{A}}, \mathrm{DIU_B}\right)$ is also expressed as the summation of the terms related to $F_{A_i^D}$ and $F_{B_i^D}$. However, the closed-form expressions of these distributions are unknown. Therefore, we need to approximate and compare these payoffs via other terms, in particular the distributions $F_{A_i^*}$ and $F_{B_i^*}$. This can be obtained by using the following lemma:

Lemma 5.1. *Fix $\lambda \geq 1$, $\forall \bar{\varepsilon} > 0$, $\exists N^* := \tilde{\mathcal{O}}\left(\bar{\varepsilon}^{-2}\ln(\bar{\varepsilon}^{-1})\right)$: $\forall n \geq N^*$, $\exists M_1 := \mathcal{O}(n/\bar{\varepsilon})$: $\forall m \geq M_1$, $\forall i = 1, 2, \ldots, n$,*

$$\sup_{\hat{x}\in\mathbb{N}} \left| F_{A_i^D}(\hat{x}) - F_{A_i^*}\left(\frac{\hat{x}}{m}\right)\right| < \bar{\varepsilon}, \tag{5.2}$$

and similarly for $F_{B_i^D}$ and $F_{B_i^*}$.

Proof. To prove (5.2), we first deduce from the special features of the rounding function r^m that $\forall \hat{x} \in \mathbb{N}$, $\forall i \in \{1, 2, \ldots, n\}$, $F_{A_i^n}(\hat{x}/m) \leq F_{A_i^D}(\hat{x}) \leq F_{A_i^n}((\hat{x}+1)/m)$.

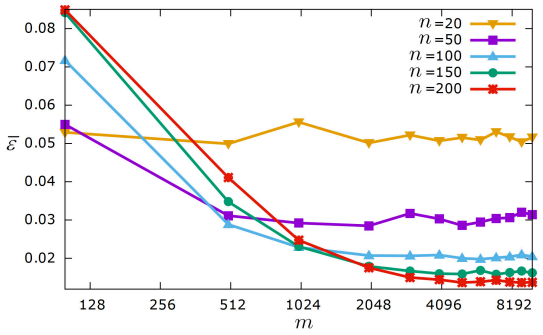
Based on Hoeffding's inequality and an intuition similar to Slutsky's theorem, we can prove that $F_{A_i^n}$ uniformly converges towards $F_{A_i^*}$ with rate $n \geq \tilde{\mathcal{O}}\left(\bar{\varepsilon}^{-2}\ln(\bar{\varepsilon}^{-1})\right)$. Then we get that, $\forall n \geq N^* = \tilde{\mathcal{O}}\left(\bar{\varepsilon}^{-2}\ln(\bar{\varepsilon}^{-1})\right)$,

$$F_{A_i^*}\left(\frac{\hat{x}}{m}\right) - \frac{\bar{\varepsilon}}{2} < F_{A_i^D}(\hat{x}) < F_{A_i^*}\left(\frac{\hat{x}}{m}\right) + \frac{1}{m}\frac{V_n}{2v_i\lambda^2} + \frac{\bar{\varepsilon}}{2}.$$

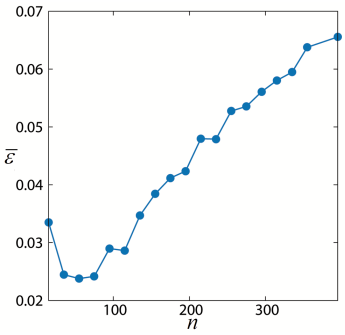
Finally, we can choose $M_1 := \frac{nv_{\max}}{\bar{\varepsilon}v_{\min}\lambda^2} = \mathcal{O}(n/\bar{\varepsilon})$ such that for $m \geq M_1$, $\frac{1}{m}\frac{V_n}{2v_i\lambda^2} \leq \frac{\bar{\varepsilon}v_{\min}\lambda^2}{nv_{\max}}\frac{V_n}{2v_i\lambda^2} \leq \frac{\bar{\varepsilon}}{2}$. □

Step 2: Using Lemma 5.1, we can approximate Π_1 via $F_{A_i^*}$ and $F_{B_i^*}$. However, we notice that this expression contains a summation of $m+1$ terms, which leads to an error in $\mathcal{O}(m\varepsilon)$ (a large number when $m \to \infty$) if we naively (approximately) replace $F_{A_i^D}$ and $F_{B_i^D}$ by $F_{A_i^*}$ and $F_{B_i^*}$. Hence, we must do a finer approximation by noticing that $P\left(A_i^D > \left\lceil 2\frac{v_i}{V_n}p\right\rceil\right)$ is upper bounded by $\mathcal{O}(\bar{\varepsilon})$ as $n \geq N^*, m \geq M_1$. Therefore, by carefully analyzing this summation, we have

$$\begin{aligned}\Pi_1 + \frac{\bar{\varepsilon}}{8}V_n \geq& \sum_{i=1}^{n}\left[v_i \sum_{\hat{y}=0}^{\left\lceil 2\frac{v_i}{V_n}p\right\rceil} F_{B_i^*}\left(\frac{\hat{y}-1}{m}\right)\left(F_{A_i^*}\left(\frac{\hat{y}}{m}\right) - F_{A_i^*}\left(\frac{\hat{y}-1}{m}\right)\right)\right] \\ \geq& \frac{V_n}{2\lambda} - \sum_{i=1}^{n}\frac{V_n}{2\lambda m} - \frac{\bar{\varepsilon}}{8}V_n. \end{aligned} \tag{5.3}$$



(a) m (and p) increases, $\lambda = \frac{p}{m} = 6/5$, log-scale x-axis.



(b) $m = 425, p = 510, \lambda = 6/5$, n increases.

Figure 1: Approximation error $\bar{\varepsilon}$ of the DIU strategy stated in Theorem 3.2 as a function of the game parameters.

Instances ($\lambda = 6/5$)	DIU error's evaluation time			Algo. EQ's elapsed time	$\frac{\lvert\text{DIU}-\text{EQ}\rvert}{V_n}$ *
	empirical CDF	Algorithm 2	Total		
$n = 20, m = 50$	0.12s	0.36s	0.49	2540.2s	0.0066
$n = 35, m = 50$	0.34s	0.67s	1.01s	10238.7s	0.0054
$n = 50, m = 100$	0.83s	1.99s	2.83s	$>$ 1.5 day	N/A
$n = 100, m = 5000$	106.46s	1396.33s	1502.79s	N/A	N/A
$n = 150, m = 8000$	380.14s	5153.11s	5533.25s	N/A	N/A
$n = 200, m = 10000$	895.36s	10991.66s	11887.02s	N/A	N/A

*The maximum rescaled difference between DIU payoffs and exact equilibrium payoffs

Table 1: Comparison between DIU error evaluation time and Algorithm EQ.

Step 3: On the other hand, for any $n \geq N^*$ and $m \geq M_2 := \mathcal{O}(n\alpha/\bar{\varepsilon})$, for any pure strategy $\hat{\boldsymbol{x}}^{\boldsymbol{A}}$, we can prove that $P\left(B_i^D = \hat{x}_i^A\right) \leq \bar{\varepsilon}/(2\alpha)$; therefore[5]

$$\Pi_2 \geq 0 \geq \sum\nolimits_{i=1}^{n} \alpha v_i P\left(B_i^D = \hat{x}_i^A\right) - \frac{\bar{\varepsilon}}{2} V_n. \qquad (5.4)$$

Step 4: Finally, using Lemma 5.1, for $n \geq N^*, m \geq M_1$, we have the approximation

$$\Pi_A\left(\hat{\boldsymbol{x}}^{\boldsymbol{A}}, \text{DIU}_\text{B}\right) = \sum_{i=1}^{n} v_i F_{B_i^D}\left(\hat{x}_i^A - 1\right) + \sum_{i=1}^{n} \alpha v_i P\left(B_i^D = \hat{x}_i^A\right)$$

$$\leq \sum_{i=1}^{n} v_i \left[F_{B_i^*}\left(\frac{\hat{x}_i^A - 1}{m}\right) + \frac{\bar{\varepsilon}}{4}\right] + \sum_{i=1}^{n} \alpha v_i P\left(B_i^D = \hat{x}_i^A\right)$$

$$\leq \left(\frac{V_n}{2\lambda} - \sum_{i=1}^{n} \frac{V_n}{2\lambda m} + \frac{\bar{\varepsilon}}{4} V_n\right) + \sum_{i=1}^{n} \alpha v_i P\left(B_i^D = \hat{x}_i^A\right). \quad (5.5)$$

By combining (5.1), (5.3), (5.4) and (5.5), we conclude (3.6) by choosing $M^* := \max\{M_1, M_2\}$.

6 Concluding Discussion

In this work, we propose the DIU strategy, defined by a simple algorithm, and prove it to be an approximate equilibrium of the discrete Colonel Blotto game. Our theoretical results also show precisely how large the number of troops and the number of battlefields of the game should be to ensure a certain level of approximation. We construct a best-response dynamic programming algorithm that efficiently computes the best-response to any marginal allocation of the opponent; and use it to evaluate the approximation error of employing the DIU strategy in practice via several numerical experiments. Our work extends the scope of applications of discrete Colonel Blotto games by trading off the accuracy with the computational efficiency, which is useful for analyzing games with large values of the parameters.

Note finally that Theorem 3.2 proves that the DIU strategy is an approximate equilibrium, i.e., no unilateral deviation can significantly improve a player's payoff. This does not directly imply that the marginals obtained under the DIU strategy are close to the marginals of the exact equilibria, although we conjecture that this is true as well (and leave its proof for future work).

[5]Here, we consider $\alpha \neq 0$. In case $\alpha = 0$, we have $\Pi_2 = 0$ and (3.6) is trivially implied by (5.1), (5.3) and (5.5).

Acknowledgments

This work was supported by the French National Research Agency through the "Investissements d'avenir" program (ANR-15-IDEX-02) and through grant ANR-16-TERC-0012; and by the Alexander von Humboldt Foundation. It was partly done at the Laboratory of Information, Networking and Communication Sciences (LINCS). The authors also thank Nicolas Gast and Bruno Gaujal for helpful discussions on this work.

References

[Ahmadinejad *et al.*, 2016] AmirMahdi Ahmadinejad, Sina Dehghani, MohammadTaghi Hajiaghayi, Brendan Lucier, Hamid Mahini, and Saeed Seddighin. From duels to battlefields: Computing equilibria of Blotto and other games. In *AAAI*, pages 376–382, 2016.

[Behnezhad *et al.*, 2017] Soheil Behnezhad, Sina Dehghani, Mahsa Derakhshan, MohammadTaghi HajiAghayi, and Saeed Seddighin. Faster and simpler algorithm for optimal strategies of Blotto game. In *AAAI*, pages 369–375, 2017.

[Bertsekas, 2017] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. Athena Scientific, 4 edition, 2017.

[Borel and Ville, 1938] Emile Borel and Jean Ville. *Applications de la théorie des probabilités aux jeux de hasard*. J. Gabay, 1938.

[Borel, 1921] Emile Borel. La théorie du jeu et les équations intégrales à noyau symétrique. *Comptes rendus de l'Académie des Sciences*, 173(1304-1308):58, 1921.

[Bošanský *et al.*, 2011] Branislav Bošanský, Viliam Lisý, Michal Jakob, and Michal Pěchouček. Computing time-dependent policies for patrolling games with mobile targets. In *AAMAS*, pages 989–996, 2011.

[Brown *et al.*, 2016] Matthew Brown, Arunesh Sinha, Aaron Schlenker, and Milind Tambe. One size does not fit all: A game-theoretic approach for dynamically and effectively screening for threats. In *AAAI*, pages 425–431, 2016.

[Conitzer and Sandholm, 2006] Vincent Conitzer and Tuomas Sandholm. Computing the optimal strategy to commit to. In *EC*, pages 82–90, 2006.

[Fang *et al.*, 2015] Fei Fang, Peter Stone, and Milind Tambe. When security games go green: Designing defender strategies to prevent poaching and illegal fishing. In *IJCAI*, pages 2589–2595, 2015.

[Gan *et al.*, 2015] Jiarui Gan, Bo An, and Yevgeniy Vorobeychik. Security games with protection externalities. In *AAAI*, pages 914–920, 2015.

[Gross and Wagner, 1950] Oliver Gross and Robert Wagner. A continuous Colonel Blotto game. U.S.Air Force Project RAND Research Memorandum, 1950.

[Gross, 1950] Oliver Alfred Gross. The symmetric Blotto game. U.S.Air Force Project RAND Research Memorandum, 1950.

[Hart, 2008] Sergiu Hart. Discrete Colonel Blotto and General Lotto games. *International Journal of Game Theory*, 36(3):441–460, 2008.

[Hortala-Vallve and Llorente-Saguer, 2012] Rafael Hortala-Vallve and Aniol Llorente-Saguer. Pure strategy nash equilibria in non-zero sum Colonel Blotto games. *International Journal of Game Theory*, 41(2):331–343, 2012.

[Kovenock and Roberson, 2012] Dan Kovenock and Brian Roberson. Coalitional Colonel Blotto games with application to the economics of alliances. *Journal of Public Economic Theory*, 14(4):653–676, 2012.

[Kovenock and Roberson, 2015] Dan Kovenock and Brian Roberson. Generalizations of the General Lotto and Colonel Blotto games. CESifo Working Paper No. 5291, 2015.

[Laslier, 2002] Jean-François Laslier. How two-party competition treats minorities. *Review of Economic Design*, 7(3):297–307, 2002.

[Laslier, 2005] Jean-François Laslier. Party objectives in the "divide a dollar" electoral competition. *Social Choice and Strategic Decisions*, pages 113–130, 2005.

[Letchford and Conitzer, 2013] Joshua Letchford and Vincent Conitzer. Solving security games on graphs via marginal probabilities. In *AAAI*, pages 591–597, 2013.

[Masucci and Silva, 2014] Antonia Maria Masucci and Alonso Silva. Strategic resource allocation for competitive influence in social networks. In *Allerton*, pages 951–958, 2014.

[Myerson, 1993] Roger B Myerson. Incentives to cultivate favored minorities under alternative electoral systems. *American Political Science Review*, 87(4):856–869, 1993.

[Pita *et al.*, 2011] James Pita, Milind Tambe, Christopher Kiekintveld, Shane Cullen, and Erin Steigerwald. Guards—innovative application of game theory for national airport security. In *IJCAI*, pages 2710–2715, 2011.

[Powell, 2009] Robert Powell. Sequential, nonzero-sum "Blotto": Allocating defensive resources prior to attack. *Games and Economic Behavior*, 67(2):611–615, 2009.

[Rinott *et al.*, 2012] Yosef Rinott, Marco Scarsini, and Yaming Yu. A Colonel Blotto gladiator game. *Mathematics of Operations Research*, 37(4):574–590, 2012.

[Roberson, 2006] Brian Roberson. The Colonel Blotto game. *Economic Theory*, 29(1):2–24, 2006.

[Schwartz *et al.*, 2014] Galina Schwartz, Patrick Loiseau, and Shankar S Sastry. The heterogeneous Colonel Blotto game. In *NetGCoop*, pages 232–238, 2014.

[Tambe, 2011] Milind Tambe. *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*. Cambridge University Press, 2011.

[Vaart, 1998] A. W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 1998.

[Vu *et al.*, 2018] Dong Quan Vu, Patrick Loiseau, and Alonso Silva. Efficient computation of approximation equilibria in discrete colonel blotto games. Technical report, 2018. Available at https://hal.archives-ouvertes.fr/hal-01787505.

[Yang *et al.*, 2013] Rong Yang, Albert Xin Jiang, Milind Tambe, and Fernando Ordonez. Scaling-up security games with boundedly rational adversaries: A cutting-plane approach. In *IJCAI*, pages 404–410, 2013.

[Yin *et al.*, 2012] Zhengyu Yin, Albert Xin Jiang, Milind Tambe, Christopher Kiekintveld, Kevin Leyton-Brown, Tuomas Sandholm, and John P. Sullivan. Trusts: Scheduling randomized patrols for fare inspection in transit systems using game theory. *AI Magazine*, 33(4):59–72, 2012.