A Simple and Efficient Algorithm to Compute Epsilon-Equilibria of Discrete Colonel Blotto Games

Extended Abstract*

Dong Quan Vu
Nokia Bell Labs, Nokia Paris-Saclay
Nozay, France
quan_dong.vu@nokia.com

Patrick Loiseau†
Univ. Grenoble Alpes, CNRS, Inria, Grenoble INP, LIG
Grenoble, France
patrick.loiseau@univ-grenoble-alpes.fr

Alonso Silva
Nokia Bell Labs, Nokia Paris-Saclay
Nozay, France
alonso.silva@nokia-bell-labs.com

ABSTRACT

The Colonel Blotto game is a famous game commonly used to model resource allocation problems in 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. Recently, the discrete version of the game—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. Through numerical experiments, we show that the proposed strategy provides a fast and good approximation even for moderate numbers of battlefields.

KEYWORDS

Colonel Blotto game; epsilon-equilibrium; resource allocation

ACM Reference Format:
Dong Quan Vu, Patrick Loiseau, and Alonso Silva. 2018. A Simple and Efficient Algorithm to Compute Epsilon-Equilibria of Discrete Colonel Blotto Games. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 3 pages.

1 INTRODUCTION AND GAME MODEL

The past decade has seen a rising interest in using game-theoretic models for security problems, see e.g., [6–10, 14, 17, 20, 21]. The Colonel Blotto game is a simple and elegant model for strategic resource allocation problems. It has important applications in many domains including not only security but also politics, industrial operations or advertisement.

In the Colonel Blotto game, two players (often referred to as colonels) choose how to distribute a fixed budget of resources (often called troops) 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. Its continuous version (where players can choose any fractional allocation), has received high attention since its first introduction in 1921 [4]. However, only partial results are known to date (see [5, 11–13, 15, 16]); in particular, the general case of asymmetric players with heterogeneous battlefields remains unsolved.

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 [1] and [2], which rely on transforming the linear program formulation to significantly improves the complexity. Yet, these algorithms still become computationally impractical when the number of battlefields and/or the number of troops is large. In these cases, the problem of efficiently computing an equilibrium remains open.

In this work, we take a different approach and propose an algorithm to compute very efficiently an *approximate* equilibrium for the discrete Colonel Blotto game with many battlefields and troops.

Game model. A discrete Colonel Blotto is a one-shot game between two players denoted A and B. Each player has a fixed amount of troops (or budget), denoted $m, p \in \mathbb{N}$ for A and B respectively.[1] Let $\lambda := \frac{p}{m}$ be the ratio of players budgets. Players simultaneously allocate their troops to n battlefields ($n \geq 3$), indexed by $i \in \{1, 2, \ldots, n\} := [n]$, each has a fixed value $v_i > 0$. We assume that all values are bounded, that is $v_i \in [v_{\min}, v_{\max}], \forall i$, where $0 < v_{\min} \leq v_{\max}$. We denote by $V_n = \sum_{i=1}^n v_i$ the total value of all battlefields. A pure strategy of player A (resp., player B) is a vector $\hat{\boldsymbol{x}}^{\boldsymbol{A}} \in \mathbb{N}^n$ (resp., $\hat{\boldsymbol{x}}^{\boldsymbol{B}}$), with elements $\hat{x}_i^A$ representing the (integer) allocation to battlefield i and satisfying the constraint $\sum_{i=1}^n \hat{x}_i^A \leq m$ (resp., $\sum_{i=1}^n \hat{x}_i^B \leq p$).

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

*The full version of this work appeared in [19].
†Also with Max Planck Institute for Software Systems (MPI-SWS).

[1] Without loss of generality, we assume that A is the weak player, i.e., $m \leq p$.

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.

and player B gains $(1-\alpha)\,v_i$ for some fixed $\alpha \in [0,1]$. Each player chooses his strategy to maximize his own payoff equal to the sum of gains on all battlefields. We denote this game $\mathcal{CB}_n^{m,p}$.

2 MAIN RESULTS

2.1 The DIU strategy

In $\mathcal{CB}_n^{m,p}$, we propose a mixed strategy called *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

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

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

We define $\forall x, r^m(x) = \frac{\hat{x}}{m}$, where $\hat{x} \in \mathbb{N}$ is uniquely determined and satisfies $\frac{\hat{x}}{m} - \frac{1}{2m} \leq x < \frac{\hat{x}}{m} + \frac{1}{2m}$ and formally give the definition:

Definition 2.1 (The DIU strategy). $\mathbf{DIU_A}$ (resp., $\mathbf{DIU_B}$) is the **mixed** strategy where player A's allocation $\hat{\boldsymbol{x}}^{\boldsymbol{A}}$ (resp., player B's allocation $\hat{\boldsymbol{x}}^{\boldsymbol{B}}$) is randomly generated from Algorithm 1.

Algorithm 1: DIU strategy generation algorithm.

Input: $n, m, p \in \mathbb{N}$, and $v \in [v_{\min}, v_{\max}]^n$
Output: $\hat{\boldsymbol{x}}^A, \hat{\boldsymbol{x}}^B \in \mathbb{N}^n$
1 **for** $i = 1, 2, \ldots, n$ **do**
2 $\quad a_i \sim F_{A_i^*}$ and $b_i \sim F_{B_i^*}$
3 **if** $\sum_{j=1}^n a_j = 0$ **then** repeat generating $a_i, \forall i$
4 $s_0^A = s_0^B = 0$
5 **for** $i = 1, 2, \ldots, n$ **do**
6 $\quad s_i^A = \sum_{k=1}^i \frac{a_k}{\sum_{j=1}^n a_j}$; $s_i^B = \sum_{k=1}^i \frac{b_k}{\sum_{j=1}^n b_j}\frac{p}{m}$
7 $\quad \hat{x}_i^A := m\left[r^m\left(s_i^A\right) - r^m\left(s_{i-1}^A\right)\right]$
8 $\quad \hat{x}_i^B := m\left[r^m\left(s_i^B\right) - r^m\left(s_{i-1}^B\right)\right]$

Algorithm 1 guarantees that the allocations are integers and satisfy the budget constraints. 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 $\{x_i^A\}_i$ (resp., $\{x_i^B\}_i$). 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[2] time $O(n)$.

2.2 Approximate equilibrium of $\mathcal{CB}_n^{m,p}$

Theorem 2.2. *The DIU strategy is an $\bar{\varepsilon}V_n$-equilibrium of the game $\mathcal{CB}_n^{m,p}$, where $\bar{\varepsilon} \leq \max\{\tilde{O}(n^{-1/2}), O(n/m)\}$.*[3]

[2]The *for loop* in lines 1-3 is not guaranteed to end in a 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.

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

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. Moreover, in a different perspective, Theorem 2.2 tells us how large the parameters n and m (and p) should be to reach a given level $\bar{\varepsilon}$ of approximation; formally stated as: *Fix $\lambda \geq 1$, $\forall \bar{\varepsilon} > 0$, $\exists N^* = O\left(\bar{\varepsilon}^{-2}\ln\left(\bar{\varepsilon}^{-1}\right)\right)$: $\forall n \geq N^*$, $\exists M^* = O(n/\bar{\varepsilon})$: $\forall m \geq M^*$, $p = m\lambda \in \mathbb{N}$, the DIU strategy is an $\bar{\varepsilon}V_n$-equilibrium of the game $\mathcal{CB}_n^{m,p}$*. This result also involves an interesting double limit of two growing parameters (n and m) and identifies a precise *scaling regime* under which the convergence holds. Here, it shows that the convergence of DIU strategy towards an equilibrium requires that m grows at least as fast as $n^{3/2}$. This implies that, 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. Although the idea is natural, the proof of Theorem 2.2 is non-trivial; its main steps are the following: (i) We prove that the distribution $F_{A_i^*}$ is close to optimal against $F_{B_i^*}$ and vice versa in each battlefield. (ii) We prove the uniform convergence (and determine the convergence rate) of the marginal distributions of the DIU strategy towards $F_{A_i^*}$ and $F_{B_i^*}$. (iii) We approximate the players' DIU payoffs by $F_{A_i^*}$ and $F_{B_i^*}$ with special analysis on the tie-case. We then prove the convergences of DIU payoffs towards equilibrium payoffs.

2.3 Numerical Experiments

We constructed numerical experiments to evaluate the quality of the approximation that DIU strategy gives depending on the game's parameters, that is to evaluate $\bar{\varepsilon}$. First, computing the value of $\bar{\varepsilon}$ requires finding a player's optimal allocation given that the opponent's allocation to battlefield $i \in [n]$ follows a given marginal distribution $\{G_i\}_{i \in [n]}$. 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 (with complexity $O\left(p^2 \cdot n\right)$) based on dynamic programming (DP) [3] to solve this problem. Second, since the marginal allocations at battlefield i under the DIU strategy are not known in closed-form, we approximate them by the corresponding empirical CDFs (controlled by the Glivenko-Cantelli theorem [18]).

The experimental results support well the results given in Theorem 2.2, especially on the effect of the double limits as n and m (and p) grow. For instance, for $n = 25, m = 75, p = 90$, we find $\bar{\varepsilon} \approx 0.04$; and for $n = 150, m = 4925, p = 5910, \bar{\varepsilon} \approx 0.019$. We also compare the computation time of our algorithm (including generating empirical CDFs and running the DP algorithm) and the of exact equilibrium computation from [2]. For example, our algorithm determines a $0.02V_n$-equilibrium for games with $n = 150, m = 8000, p = 9600$ in under 2 hours, while [2]'s algorithm takes over one day to find the exact equilibrium of games with $n = 45, m = 75, p = 90$.

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).

REFERENCES

[1] AmirMahdi Ahmadinejad, Sina Dehghani, MohammadTaghi Hajiaghayi, Brendan Lucier, Hamid Mahini, and Saeed Seddighin. 2016. From Duels to Battlefields: Computing Equilibria of Blotto and Other Games.. In *AAAI*. 376–382.

[2] Soheil Behnezhad, Sina Dehghani, Mahsa Derakhshan, MohammadTaghi Haji-Aghayi, and Saeed Seddighin. 2017. Faster and Simpler Algorithm for Optimal Strategies of Blotto Game.. In *AAAI*. 369–375.

[3] Dimitri P. Bertsekas. 2017. *Dynamic Programming and Optimal Control* (4 ed.). Vol. 1. Athena Scientific.

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

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

[6] Branislav Bošanský, Viliam Lisý, Michal Jakob, and Michal Pěchouček. 2011. Computing Time-dependent Policies for Patrolling Games with Mobile Targets. In *AAMAS*. 989–996.

[7] Matthew Brown, Arunesh Sinha, Aaron Schlenker, and Milind Tambe. 2016. One Size Does Not Fit All: A Game-Theoretic Approach for Dynamically and Effectively Screening for Threats. In *AAAI*. 425–431.

[8] Vincent Conitzer and Tuomas Sandholm. 2006. Computing the Optimal Strategy to Commit to. In *EC*. 82–90.

[9] Fei Fang, Peter Stone, and Milind Tambe. 2015. When Security Games Go Green: Designing Defender Strategies to Prevent Poaching and Illegal Fishing.. In *IJCAI*. 2589–2595.

[10] Jiarui Gan, Bo An, and Yevgeniy Vorobeychik. 2015. Security Games with Protection Externalities. In *AAAI*. 914–920.

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

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

[13] Jean-François Laslier. 2002. How two-party competition treats minorities. *Review of Economic Design* 7, 3 (2002), 297–307. https://doi.org/10.1007/s100580200083

[14] Joshua Letchford and Vincent Conitzer. 2013. Solving Security Games on Graphs via Marginal Probabilities. In *AAAI*. 591–597.

[15] Brian Roberson. 2006. The Colonel Blotto game. *Economic Theory* 29, 1 (2006), 2–24. https://doi.org/10.1007/s00199-005-0071-5

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

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

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

[19] Dong Quan Vu, Patrick Loiseau, and Alonso Silva. 2018. Efficient computation of approximate equilibria in discrete Colonel Blotto games. In *IJCAI-ECAI*.

[20] Haifeng Xu. 2016. The Mysteries of Security Games: Equilibrium Computation Becomes Combinatorial Algorithm Design. In *EC*. 497–514.

[21] Rong Yang, Albert Xin Jiang, Milind Tambe, and Fernando Ordonez. 2013. Scaling-up Security Games with Boundedly Rational Adversaries: A Cutting-plane Approach.. In *IJCAI*. 404–410.