Journal of Artificial Intelligence Research 81 (2024) 163-202 Submitted 03/2024; published 09/2024

Digraph k-Coloring Games: New Algorithms and Experiments

Andrea D'Ascenzo ADASCENZO@LUISS.IT
Luiss University
Viale Romania 32, 00197 Rome, Italy

Mattia D'Emidio MATTIA.DEMIDIO@UNIVAQ.IT
Dept. of Information Engineering,
Computer Science, and Mathematics
University of L'Aquila
Via Vetoio snc, 67100 L'Aquila, Italy

Michele Flammini MICHELE.FLAMMINI@GSSI.IT
Dept. of Computer Science
Gran Sasso Science Institute
Viale F. Crispi 7, 67100 L'Aquila, Italy
Dept. of Mathematics and Computer Science
University of Calabria
Via P. Bucci, 87036 Arcavacata di Rende (CS)

Gianpiero Monaco GIANPIERO.MONACO@UNICH.IT
Dept. of Economic Studies
University of Chieti-Pescara
Viale Pindaro, 65127 Pescara, Italy

Abstract

We study digraph k-coloring games where strategic agents are vertices of a digraph and arcs represent agents' mutual unidirectional conflicts/idiosyncrasies. Each agent can select, as strategy, one of k different colors, and her payoff in a given state (a k-coloring) is given by the number of outgoing neighbors with a color different from her one. Such games model lots of strategic real-world scenarios and are related to several fundamental classes of anti-coordination games. Unfortunately, the problem of understanding whether an instance of the game admits a *pure* Nash equilibrium (NE), i.e., a state where no agent can improve her payoff by changing strategy, is NP-complete. Thus, in this paper, we focus on algorithms to compute an *approximate* NE: informally, a coloring is an approximate γ-NE, for some $\gamma \geq 1$, if no agent can improve her payoff, by changing strategy, by a multiplicative factor of γ.

Our contribution is manifold and of both theoretical and experimental nature. First, we characterize the hardness of finding pure and approximate equilibria in both general and special classes of digraphs. Second, we design and analyze three approximation algorithms with different theoretical guarantees on the approximation ratio, under different conditions; (i) algorithm APPROX-1 which computes, for any $k \geq 3$, a Δ_o-NE for any n vertex graph having a maximum outdegree of Δ_o, in polynomial time; (ii) algorithm LLL-SPE, a randomized algorithm that, for any constant $k \geq 2$, determines a γ-NE for some constant γ but only in digraphs whose minimum outdegree is sufficiently large, in polynomial time in expectation; (iii) algorithm APPROX-3 which, for any $\epsilon > 0$, computes a $(1+\epsilon)$-NE by using $\mathcal{O}(\frac{\log n}{\epsilon})$ colors, for any n-vertex digraph. Note that, the latter shows that a $(1 + \epsilon)$-NE exists and can be computed in polynomial time for $k = \mathcal{O}(\log n)$.

Finally, to assess how proposed algorithms behave in the typical case, we complete our study with an extensive experimental evaluation showing that, while newly introduced algorithms achieve bounded worst case behavior, they generally perform poorly in practice. Motivated by such unsatisfactory performance, we shift our attention to the best-response paradigm, successfully applied to other classes of games, and design and experimentally evaluate it a heuristic based on such paradigm. Our experiments provide strong evidences of such approach outperforming, in terms of approximation and computational time, all other methods and hence identify it as the most suited candidate for practical usage. More remarkably, it is also able to compute exact, pure NE in the great majority of cases. This suggests that, while these games are known to not always possess a pure NE, such an equilibrium often exists and can be efficiently computed, even by a distributed uncoordinated interaction of the agents.

1. Introduction

In this paper we study *digraph k-coloring games* (Kun, Powers, & Reyzin, 2013): we are given an unweighted directed graph where vertices represent selfish autonomous agents and arcs represent mutual unidirectional idiosyncrasies or conflicts. Moreover, we have a set of $k \geq 2$ colors denoting agents' available *choices* or *strategies*. A state of the game is a vertex k–coloring, induced by the choices of the agents. The objective of each agent is to maximize her own payoff, which is defined as the number of outgoing neighbors that has a color different from hers.

Digraph k-coloring games form some of the basic payoff structures in algorithmic game theory, and model many relevant real-world scenarios. A classic example is social networks where members tend to split in groups such that each member wish to maximize the number of rivals or opponents they do not end up with. Another example is represented by scenarios where agents can be miners that have to decide which land to drill for resources: a miner maximizes her happiness when the number of other competitor miners that choose the same land is minimized. Another real-world domain that is modeled well through the considered game is competitive skill-learning, e.g., when employees of a company have to decide which skill to learn in order to improve their value to the company: if an employee learns a skill that few other employees also learn, she will have more chance of exploiting it. Finally, perhaps one of the most intuitive scenario that is effectively modeled by such games is that related to markets (either digital or physical) and, in general, to marketing. Business owners, companies, shop keepers, when planning their trading/production strategies tend to decide to promote/produce some product, rather than another, on the basis of minimizing the number of neighboring competitors that exist in a given market (where neighboring can be interpreted in terms of geographical positioning of shops/production sites, or in terms of closeness in social networks of potential customers).

A standard, well-established solution notion of stable outcome, in scenarios with selfish and autonomous agents, is the *(pure) Nash equilibrium* (NE, for short), that is a state where no agent can improve her payoff by unilaterally changing her strategy. In our setting of digraph k-coloring games, a NE is a coloring where no vertex (agent/player) can improve her payoff by unilaterally changing color. Note that, the NE is widely considered one of the most important concept in the field of game theory and, as such, its efficient computation is one of the most important problems in algorithmic game theory and hence artificial intelligence (Fabrikant, Papadimitriou, & Talwar, 2004). Moreover, digraph k-coloring games can be seen as a particular form of hedonic games (see Section 1.2 for more details), which have been widely investigated in the artificial intelligence/multi-agent systems community. In the special setting of *graph k-coloring games*, when the input graph is undirected, the game admits a potential function (Monderer & Shapley, 1996), which implies that a NE always exists. In addition, when the graph is unweighted and undirected, the basic dynamics where at each step one agent performs an improving move always converges to a NE in a polynomial number of steps (Hoefer, 2007; Kun et al., 2013). However, in the more general context of directed graphs, for any $k \geq 2$, it is known that even the problem of understanding whether digraph k-coloring games admit a NE is NP-complete (Kun et al., 2013).

Therefore, in this paper, as done in the literature for other games belonging to the class of games such that Nash equilibria do not exist or their computation is NP-hard (Anshelevich & Sekar, 2014), we focus on a milder form of equilibrium that is typically referred to as *approximate (pure) Nash equilibrium.* More in detail, a state is called an *approximate γ-Nash equilibrium* (γ-NE, for short) when, for some $\gamma \geq 1$, no agent can strictly improve her payoff, by a multiplicative factor of γ, through a change in her strategy.

1.1 Our Contribution

Our contribution on digraph k-coloring games is manifold and both of theoretical and experimental nature. From a theoretical viewpoint, we first characterize existence conditions and hardness of finding pure equilibria. Specifically, we show that: i) a pure NE is not guaranteed to exist for any number of agents n and number of available colors $k < n$; ii) for any $k \geq 2$ a pure NE exists and can be found in polynomial time in bipartite digraphs and directed acyclic graphs (DAGs); iii) even for $k = 2$, a γ-NE might not exist for any bounded value of γ, in general digraphs. Then, we design and analyze a collection of approximation algorithms for the problem, exhibiting different approximation ratios, for either general or special classes of digraphs. In detail, we introduce a first deterministic algorithm, named APPROX-1, that, for any $k \geq 3$, returns in polynomial running time a k-coloring that is a Δ_o-NE, where Δ_o is the maximum outdegree of the given digraph (Δ_i will instead denote the maximum indegree of the given digraph). Then, we present a second, randomized algorithm, called LLL-SPE, which, by exploiting the constructive version of the well-known Lovász Local Lemma (Moser & Tardos, 2010), achieves in expected polynomial running time, and for any given $k \geq 2$, a constant approximate NE (a γ-NE for some constant $\gamma > 1$) in digraphs having outdegree of any vertex in $\Omega(\ln \Delta_o + \ln \Delta_i + \ln k)$. For instance, digraphs where the minimum outdegree is $\Omega(\log n)$ belong to this class of digraphs. Finally, we propose a third deterministic algorithm, named APPROX-3, that, for general digraphs and for any $\epsilon > 0$, computes a $(1 + \epsilon)$-NE by using $\mathcal{O}(\frac{\log n}{\varepsilon})$ colors, in polynomial time. Note that such a construction shows that a $(1 + \epsilon)$-NE always exists, and can be computed in polynomial time, when $k = \Omega(\frac{\log n}{\varepsilon})$. The latter contrasts with the fact that, for any $k < n$, pure Nash equilibria are not guaranteed to exist and in general cannot be computed in polynomial time, unless $P = NP$. We remark that, to the best of our knowledge, the above are the first polynomial time algorithms to compute approximate Nash equilibria for digraph coloring games.

From an experimental viewpoint, instead, in order to assess how newly presented methods behave in practical contexts, we complete our study by conducting an extensive experimental evaluation whose main aim is determining the best performing algorithm, in terms of both approximation and running time, in inputs arising from real-world application domains. Specifically we implement and test all approximation algorithms against both artificial and real-world digraphs, of heterogeneous sizes and topologies, and across several values of parameters k and ϵ, respectively. We measure, as main performance indicators, both approximation (namely obtained γ) and computational time. Plus, we assess other metrics, namely *average payoff* and *fraction of unhappy vertices*, which naturally characterize the practical effectiveness of the considered algorithms in the application domains where they are intended to be applied (e.g., for breaking ties in case of different equilibria exhibiting a same approximation). Our results highlight that, for all newly introduced algorithms, the theoretical analysis fails at accurately capturing their true performance in practical scenarios. More in detail, we observe that:

(a) on the negative side, while algorithms APPROX-1 and LLL-SPE are suitably designed to achieve bounded worst case behavior w.r.t. approximation, they generally exhibit poor performance in practice; measured approximation factors obtained through the execution of such algorithms, in fact, are often close to those one can achieve by simply assigning colors to agents/vertices uniformly at random;

(b) on the positive side, algorithm APPROX-3 practical performance is by far underestimated by the analysis we present, since our tests show it is always able to compute a $(1+\varepsilon)$-NE by using a number of colors k that is much smaller than the worst case upper bound.

Given the unsatisfactory performance of algorithms APPROX-1 and LLL-SPE, under the motivation of lack of practical approximation algorithms for the problem when k is given, we hence shift our focus on the classical best-response paradigm (Roughgarden, 2010), where at each step an agent that is not in a NE is selected at random and performs her *best improving move*. Such paradigm, has been widely considered in the literature for other games since, even thought it typically induces a dynamics that is not known to offer any upper bound on the provided approximation, and that might not always stabilize, it often provides good results in practice. In particular, we design and describe MYO-BEST-RESP, a heuristic based on such paradigm, and experimentally evaluate it, to assess its practical effectiveness in the context of digraph k-coloring games. We compare the behavior of MYO-BEST-RESP against that of all other solutions, again through an extensive experimental evaluation involving large sets of heterogeneous inputs and values of k.

Our experimental results provide strong empirical evidences of MYO-BEST-RESP being the most effective solution among the tested ones, and hence advised for practical usage. In fact, in essentially all considered combinations of inputs and values of k, MYO-BEST-RESP results to be the best performing algorithm in terms of approximation. Moreover, quite surprisingly, in the majority of the cases it converges to a pure NE (i.e., it computes exact, non-approximate solutions) in a reasonably low running time, and even for large graphs. This is a remarkable property of such method, given the hardness of computing pure NE in general. In the (few) remaining cases, MYO-BEST-RESP is comparable to other methods, in terms of approximation, but achieves better results in terms of average payoff and fraction of unhappy vertices.

Our study thus highlights that, while this class of games is known to not always possess NE, in almost all cases such equilibria exist and can be efficiently computed, even in a distributed uncoordinated way by a decentralized interaction of the agents, such as that underlying MYO-BEST-RESP (while APPROX-1, APPROX-3 and LLL-SPE are not naturally suited for distributed implementations). Hence, this triggers several new theoretical questions and demand for further investigation on digraph k coloring games. In particular, it remains unknown whether constant approximation can be achieved in the worst case for general digraphs or whether a pure NE can be computed in polynomial randomized time for broader special classes of digraphs, or even deterministically.

1.2 Related Work

k-coloring games in undirected graphs have been first investigated in (Hoefer, 2007; Kun et al., 2013), where it is shown that a NE always exists and can be computed in polynomial time if the graph is unweighted. When the graph is weighted, instead, a NE always exists but the problem of computing it is PLS-complete even for $k = 2$ (Schäffer & Yannakakis, 1991). Observe that graph 2-coloring games are exactly equivalent to max cut games. In (Poljak, 1995) it is proven that NE can be computed in polynomial time for graph 2-coloring games if the maximum degree of the graph is at most 3. A related investigation for the same class of games is presented in (Bhalgat, Chakraborty, & Khanna, 2010; Caragiannis, Fanelli, & Gravin, 2017), where the authors give an algorithm that, for any $\epsilon > 0$, computes a $(3 + \epsilon)$-NE in time polynomial in $\frac{1}{\epsilon}$ and in the instance size. All above results exploit the potential function method. Unfortunately, digraph k-coloring games (where the graph is directed) do not admit a potential function and the problem of understanding whether they admit a NE is NP-complete for any fixed $k \geq 2$, even in the unweighted case (Kun et al., 2013). The performance of NE in general graph k-coloring games has been addressed in (Feldman & Friedler, 2015), while the authors of (Carosi & Monaco, 2020) consider Nash equilibria where players also have an extra profit depending on the chosen color. Finally, the authors of (Gourvès & Monnot, 2010; Carosi, Fioravanti, Gualà, & Monaco, 2019) study the existence of strong NE, i.e., resistant to coalitional moves, again for graph k-coloring games.

Digraph k-coloring games are related to many fundamental games that have been widely studied in recent literature. One example is graphical games, first introduced in (Kearns, Littman, & Singh, 2001), where the payoff of each agent depends only on the strategies of her neighbors in a given social knowledge graph defined over the set of the agents, where an arc (i, j) means that j influences i's payoff. An interesting class of graphical games, related to digraph k-coloring games, is that of graphical congestion games (Bilò, Fanelli, Flammini, & Moscardelli, 2011), where each agent has to choose a set of resources while taking into account that each resource e has a latency function f_e depending on the number of agents using e. For the case where each agent can choose only one of the available resources, also called load balancing, and the latency function is linear (i.e., $f_e(x) = x$, where x the number of agents using e), a Nash equilibrium for the arising digraph k-coloring game corresponds to a Nash equilibrium for an equivalent instance of the graphical congestion game and vice versa. In (Bilò et al., 2011) it is shown that each graphical congestion game defined over a directed acyclic graph admits a Nash equilibrium that can also be found in polynomial time. We are not aware of papers providing polynomial time algorithms that compute approximate Nash equilibria in graphical congestion games for generic directed graphs. However, approximation algorithms are known for many graphical games (see for example (Vickrey & Koller, 2002; Kearns et al., 2001)).

Digraph k-coloring games can also be seen as a particular form of hedonic games with an upper bound (i.e., k) on the number of coalitions (see (Aziz & Savani, 2016) for an introduction to hedonic games). Specifically, given a k-coloring, agents having a same color can be seen as members of the same coalition in the corresponding hedonic game. In order to get the equivalence between the two games, the so–called hedonic utility of an agent v has to be defined as the overall number of her neighbors minus the number of agents of her neighborhood that are in the same coalition. Issues related to Nash equilibria in hedonic games have been largely investigated under several assumptions (see (Aziz, Brandt, & Seedig, 2013; Ballester, 2004; Bilò, Fanelli, Flammini, Monaco, & Moscardelli, 2018; Bogomolnaia & Jackson, 2002; Feldman, Lewin-Eytan, & Naor, 2015; Gairing & Savani, 2011; Monaco, Moscardelli, & Velaj, 2020, 2019; Peters & Elkind, 2015) and references therein).

An interesting study on Nash equilibria for graphical hedonic games can be found in (Peters, 2016). In these games, agents are arranged in an underlying graph and need to be partitioned into coalitions. Every agent only cares about which of her neighbours are in the same coalition as her. Every hedonic game can be made graphical by introducing edges whenever one agent's utility depends on the other's presence. To the best of our knowledge, no known result concerns approximate Nash equilibria.

While coloring games are the paradigmatic class of anti-coordination games, another very active stream of research has been dedicated to coordination games, where agents instead are rewarded for choosing common strategies rather than different ones. Results about coordination games can be found in (Anshelevich & Sekar, 2014; Apt, de Keijzer, Rahn, Schäfer, & Simon, 2017; Rahn & Schäfer, 2015; Simon & Wojtczak, 2016). Finally, the authors of (Panagopoulou & Spirakis, 2008) study games where Nash equilibria are proper vertex colorings in an undirected unweighted graphs setting.

Another prominent class of games, generalizing coloring games and bearing strong connections with coalition, coordination, and anti-coordination games is the one of the so-called *polymatrix coordination games* (Yanovskaya, 1968). Here each agent v must select an action in her strategy set, and the utility is given by the preference she has for her action plus, for each neighbor w, a payoff which strictly depends on the mutual actions played by v and w. Polymatrix games have been thoroughly studied both in some classical works (Howson, 1972; Eaves, 1973; Howson & Rosenthal, 1974; Miller & Zucker, 1991) and also, more recently, with a special focus on equilibria (Rahn & Schäfer, 2015; Cai, Candogan, Daskalakis, & Papadimitriou, 2016; Deligkas, Fearnley, Savani, & Spirakis, 2017; Deligkas, Fearnley, & Savani, 2020).

1.3 Structure of the Paper

The paper is organized as follows. Section 2 introduces the notation used throughout the paper and the necessary background while Section 3 presents both negative and constructive results on the existence and computability of pure Nash equilibria, along with notions related to approximate equilibria. In Section 4 we describe approximation algorithms APPROX-1, LLL-SPE and APPROX-3, with guarantees for computing various kinds of γ-NE in both general and special digraphs, while in Section 6 we present our experimental work, that includes the definitions of our heuristic based on best response, i.e., algorithm MYO-BEST-RESP. Finally, Section 7 concludes the paper and highlights possible future research directions.

2. Preliminaries

We assume we are given an unweighted directed graph, or simply digraph, $G = (V, A)$, without self loops, having $|V| = n$ vertices and $|A| = m$ arcs connecting vertices of G. Any arc $(v, w) \in A$ is *directed* from vertex v to vertex w. An arc $(v, w) \in A$ is said to be an *outgoing arc* from vertex v and an *incoming arc* to vertex w, respectively.

Given a vertex $v \in V$, we denote by δ_o^v (δ_i^v, respectively) the *outdegree* (the *indegree* respectively) of v, that is the number of outgoing arcs from v (the number of incoming arcs to v, respectively) in G. The set of *outgoing neighbors* $N_{out}(v)$ (*incoming neighbors* $N_{in}(v)$, respectively) of a vertex v is the set of vertices induced by the outgoing arcs of v (the incoming arcs of v, respectively), i.e., $N_{out}(v) = \{w : w \in V \ \wedge \ (v, w) \in A\}$ and $N_{in}(v) = \{w : w \in V \ \wedge \ (w, v) \in A\}$, respectively. Clearly, we have $|N_{out}(v)| = \delta_o^v$ and $|N_{in}(v)| = \delta_i^v$ for any vertex $v \in V$. Moreover, we denote by $d_o = \min_{v \in V} \delta_o^v$ and $\Delta_o = \max_{v \in V} \delta_o^v$ the minimum and maximum outdegree of G, respectively. Similarly, we call $\overline{d_o}$ and $\overline{\overline{d_o}}$ the average and median outdegree, respectively, and $\Delta_i = \max_{v \in V} \delta_i^v$ the maximum indegree of G. Finally, given a subset of vertices $V' \subseteq V$, we denote with $G[V']$ the subgraph of G induced by V'.

2.1 Digraph k-Coloring Games

In a *digraph k-coloring game* we are given a digraph $G = (V, A)$, without self loops, in which each vertex $v \in V$ is a *selfish agent*, and a set C of $|C| = k$ available *colors*. Each agent has a same *set of actions* (i.e., a same *strategy set*), which is the set of the k available colors. A *state* of the game $c = \{c_1, \ldots, c_n\}$ is a *k-coloring* for graph G (simply *coloring* when k is clear from the context), where c_v is the color chosen by agent $v \in V$ (i.e., a number from 1 to k). In what follows, we will use vertex and agent interchangeably. Given a coloring c, the *payoff* $\mu_c(v)$ (often also referred to as the *utility*) of an agent v is the number of outgoing neighbors of v whose color in c is different from that of v, i.e., $\mu_c(v) = |\{w \in N_{out}(v) : c_v \neq c_w\}|$. Observe that, for a vertex v we have that either $N_{out}(v) = \emptyset$ or $c_v = c_w \ \forall w \in N_{out}(v)$ suffice to imply that $\mu_c(v) = 0$.

A coloring c is called a *pure Nash equilibrium* (sometimes also referred to, in the literature, as *stable equilibrium*, and denoted in what follows as pure NE or simply NE, for short), if no agent v in the graph can improve her payoff by unilaterally changing strategy (i.e., color). Formally, if we use (c_{-v}, c'_v) to denote the coloring obtained, from a coloring $c = \{c_1, \ldots, c_n\}$, by changing the strategy of agent v from c_v to c'_v, then a coloring c is a pure NE if $\mu_c(v) \geq \mu_{(c_{-v}, c'_v)}(v)$, for any possible color $c'_v \in C$ and for any vertex $v \in V$.

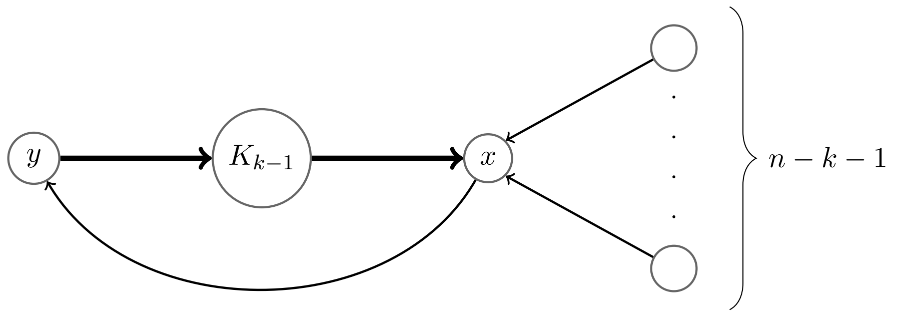


Figure 1: An instance for which there is no NE, even with a number of colors linear in the number of agents. A bold arrow means that there is complete incidence from the source subgraph to the target subgraph.

3. On the Existence of Nash Equilibria in Digraph k-Coloring Games

In this section, we show that a NE is not guaranteed to exist, for general graphs, and even for a large number of available colors k. Then, we show that for some special graph classes, instead, a NE can be computed efficiently, deterministically and in polynomial running time. In particular, on the one hand it is easy to observe that a NE can always be found with n colors, by simply assigning to each vertex a different color; on the other hand, for every $k \leq n-1$ a NE is not guaranteed to exist. In fact, it is possible to prove the following proposition.

Proposition 3.1. *For arbitrarily large values of $n \geq 3$, and any fixed k such that $1 < k \leq n-1$, there exist instances of the digraph k-coloring game with n vertices not admitting any NE.*

Proof. Consider the following instance of digraph k-coloring game: there are two vertices x, y, where x has an arc directed toward y; moreover, there is a complete, bidirectionally speaking, directed clique K_{k-1} of size $k-1$, and each vertex in the clique has an arc directed toward x; finally, y has arcs directed toward all vertices in the clique. Suppose we have k colors. Assume by contradiction that a stable coloring exists and let c be the color assigned to x. By the stability constraint, the vertices in the clique must have the remaining $k-1$ colors, one per vertex. In fact, if some vertex in the clique is using the same color as x, then it could switch to some unused color that increases its utility. Moreover, also y should have a color different from the ones in the clique, that is it must have color c. But then x would have utility 0 and would improve by switching color: a contradiction to the fact that the coloring was stable.

Notice that in the above construction $k = n-1$. In order to prove the claim for every fixed value of k such that $1 < k \leq n-1$, it suffices to add to the aforementioned graph of $k+1$ vertices $n-k-1$ additional dummy vertices with an arc directed toward x. An example of instance with this structure is shown in Figure 1. □

While in general the existence of a NE cannot be guaranteed, the following results hold for special digraph classes.

Proposition 3.2. *Given an instance of the digraph k-coloring game such that the underlying digraph is bipartite, a NE with $k \geq 2$ colors always exists and can be found in polynomial time.*

Proof. Given a digraph, it is well known that it is possible in polynomial time to test whether the graph is bipartite or not and, in the affirmative case, it is possible to return a proper 2-coloring of it, i.e., if the set of vertices v is partitioned in to the sets V_1 and V_2, then all the vertices in V_1 are colored with color 1 and all the vertices in V_2 are colored with color 2. Clearly such a coloring maximizes the utility of all the agents. $\square$

Proposition 3.3. *Given an instance of the digraph k-coloring game such that the underlying digraph is without cycles (i.e., it is a DAG), then a NE with $k \geq 2$ colors always exists and can be found in polynomial time.*

Proof. Given a digraph, it is well known that it is possible in polynomial time to test whether the graph is a DAG or not and, in the affirmative case, one can return a topological sorting of the vertices, i.e., a linear ordering of such vertices such that for every directed arc (v, w) from vertex v to vertex w, v comes before w in the ordering. Then, it is enough to consider vertices in the reverse order and to color each vertex with its best response, that is choosing the color that maximizes her payoff given the choices made by the previous agents. $\square$

Unfortunately, for general directed graphs, the problem of determining whether the digraph k-coloring game admits a NE is NP-Hard, for all $k \geq 2$ (Kun et al., 2013). Therefore, from here onward we consider the milder notion of *approximate Nash equilibrium*, defined as follows: a state or coloring c is a γ-approximate Nash equilibrium (simply γ-NE or γ-*stable* equilibrium for short), for some $\gamma \geq 1$, if no agent can strictly improve her payoff by a multiplicative factor of γ, by changing color. More formally, a coloring $c = \{c_1, \ldots, c_n\}$ is a γ-NE when, for any possible color $c'_v \in C$ and for any vertex $v \in V$ we have $\gamma \cdot \mu_c(v) \geq \mu_{(c_{-v}, c'_v)}(v)$. If such condition holds for vertex v, we say that v is γ-*happy.* Therefore, a γ-NE is a coloring in which every vertex is γ-*happy.* Viceversa, a vertex v is γ-*unhappy*, for some $\gamma \geq 1$, if and only if $\gamma \cdot \mu_c(v) < \mu_{(c_{-v}, c'_v)}(v)$ for some color $c'_v \in C$. In other words, a γ-unhappy vertex can strictly improve her payoff by a multiplicative factor of γ, by changing color. We define the *potential payoff* $\pi_c(v)$ of vertex v as the maximum payoff v can achieve by unilaterally changing its color, that is $\pi_c(v) = \max_{c'_v \in C} \mu_{(c_{-v}, c'_v)}(v)$. Consequently, we call the *potential color* of a vertex v to be the color of C inducing the potential payoff, i.e., $\psi_c(v) \in \mathit{argmax}_{c'_v \in C} \, \mu_{(c_{-v}, c'_v)}(v)$.

By analogy, we introduce the special notion of (simply) unhappy vertex, which occurs for a γ-unhappy vertex when $\gamma = 1$. Specifically, given a coloring c, we say a vertex v is *unhappy* if and only if $\mu_c(v) < \mu_{(c_{-v}, c'_v)}(v)$ for some color $c'_v \in C$, i.e., when the vertex can strictly improve her payoff by changing color unilaterally (c is not a pure NE). Clearly, if a vertex v is unhappy we have that $\mu_c(v) < \pi_c(v)$ and $c_v \neq \psi_c(v)$ while, if $\pi_c(v) \leq \gamma \mu_c(v)$, for all vertices $v \in V$ for some $\gamma > 0$, we have a γ-NE. Whenever such a value of γ can be found, we refer to it as *approximation factor* for an instance of the problem while, by analogy, we call $\gamma_v = \frac{\pi_c(v)}{\mu_c(v)}$ the *approximation factor of vertex* $v \in V$.

Observe that, in what follows, we focus on the case of $k \geq 3$. In fact, the following proposition can be easily proven.

Proposition 3.4. *When $k = 2$, we have that any 2-coloring of a direct odd cycle (i.e., a cycle with an odd number of vertices) is not a γ-NE for any $\gamma \geq 1$.*

In fact, it is easy to observe that, for any possible coloring, there exists a vertex having payoff zero and hence no bounded γ can be found.

4. Approximation Algorithms for Digraph k-Coloring Games

In this section we design and analyze algorithms to compute approximate NE for digraph k-coloring games with bounded γ. Specifically, we first introduce a deterministic algorithm, named APPROX-1 that, for general graphs and for $k \geq 3$, is able to compute, in worst case polynomial time, a Δ_o-NE. Then, we describe LLL-SPE, a randomized algorithm that computes a constant-NE, in expected polynomial running time, for graphs having sufficiently large minimum outdegree (namely the minimum degree must be $\Omega(\ln \Delta_o + \ln \Delta_i)$). Finally, we present approximation algorithm APPROX-3 that, for any digraph G and $\epsilon > 0$, is able to determine a $(1+\epsilon)$-NE by using $\mathcal{O}\left(\frac{\log n}{\epsilon}\right)$ colors in polynomial time.

4.1 Computing a Δ_o-Nash Equilibrium

In this section, we present a polynomial time algorithm that, given a digraph G returns a k-coloring, for any $k \geq 3$, where every vertex v such that $\delta_o^v(G) \geq 1$ has payoff at least 1. Clearly this corresponds to a Δ_o-NE because Δ_o is the maximum payoff that can be achieved. Notice that, the algorithm works for any digraph and uses at most three colors.

The algorithm is iterative and, at each iteration, it visits the graph induced by the uncolored vertices and detects either a cycle or a path (when the visit reaches a vertex without outgoing edge in the induced subgraph). Then it colors the vertices of the cycle or path by alternating three colors (for instance colors 1, 2 and 3) in a way that every vertex gets payoff of at least 1. In particular, if the subgraph is a cycle then the algorithm considers vertices of the cycle in clockwise order and assigns the colors in such order (starting by any vertex) by alternating the three colors. If the subgraph is a path from vertex v to vertex w, then it colors the vertex w by a different color with respect to the already colored vertex u, if the arc $(w, u) \in E$. Otherwise, it means that $\delta_o^w = 0$ and we can assign any color to w. Then, the algorithm proceeds by alternating colors (in this case two colors are enough) for the other vertices of the path considered in the reverse order starting from w. Notice that if the algorithm does not detect any odd cycle then two colors are enough. A formal description of the procedure is given in Algorithm 1. The correctness of the procedure is stated in Theorem 4.1, along with a bound on its computational complexity.

Theorem 4.1. *Algorithm* APPROX-1 *computes a Δ_o-NE in $\mathcal{O}(\Delta_o n^2)$ time.*

Proof. Observe that, in any generic iteration of APPROX-1, either a path or a cycle, emanating from a non colored vertex v is detected. In both cases, list L contains at most n vertices and, for each of such vertices, the set of neighbors is evaluated in order to decide the color to assign. This step requires $\mathcal{O}(\Delta_o)$ time per vertex and colors vertices in an alternating fashion, thus guaranteeing that any vertex that is colored gets a color such that there exists at least one neighbor being assigned a different one (and hence payoff is at least 1 and γ is

Algorithm 1: Algorithm APPROX-1

```
Input: A digraph $G = (V, A)$, a set $C$ of $|C| = k$ available colors
Output: A $k$-coloring $c$ of $G$
1  $L \leftarrow$ empty list;
2  while $\exists\ v \in V$ not colored do
3      Append $v$ at the end of $L$;
4      $x = v$;
5      while $\exists\ i \in V$ such that $(x, i) \in A$ and $i$ is not colored do
6          if $i \in L$ then
7              Let $L' \subseteq L$ be sublist of vertices in $L$ starting from vertex $i$ to the end;
8              if $|L'|$ is even then
9                  Color vertices of $L'$ by alternating two colors $c', c'' \in C, c' \neq c''$;
10                 Color $i$ with a third color different from $c'$ and $c''$;
11                 $x = i$;
12                 break;
13             else
14                 Color vertices in $L' \setminus \{i\}$ by alternating two colors $c', c'' \in C, c' \neq c''$;
15                 Color $i$ with a third color different from $c'$ and $c''$;
16                 $x = i$;
17                 break;
18         else
19             $x = i$;
20             Append $i$ to the end of $L$;
21     if $v == x$ and $v$ is not colored then
22         if $\delta_o^v > 0$ then
23             $c_v \leftarrow \psi_c(v) = argmax_{c'_v \in C}\, \mu_{(c_{-v}, c'_v)}(v)$; /* Color maximizing payoff */
24         else $c_v \leftarrow$ random color in $C$;
25     if $v \neq x$ and $x$ is not colored then
26         if $\delta_o^x > 0$ then
27             $c_x \leftarrow \psi_c(x) = argmax_{c'_x \in C}\, \mu_{(c_{-x}, c'_x)}(x)$; /* Color maximizing payoff */
28             Color vertices in $L \setminus \{x\}$ from the last vertex down to the first (i.e., $v$) by
                 alternating color $c_x$ and a color $i \in C$ such that $i \neq c_x$;
29         else
30             Color vertices of $L$ by alternating two colors $c', c'' \in C, c' \neq c''$;
```

at most Δ_o). Finally, that each iteration colors at least one vertex, hence the total number of required iterations is at most n. The claim follows. $\square$

4.2 Computing a Nash Equilibrium with Constant Approximation in Special Graph Classes

In this section we design and analyze algorithm LLL-SPE which, for any constant $k \geq 2$, computes an approximate γ-NE for constant γ, for a large class of digraphs, with high probability. In particular, the idea underlying the algorithm is to exploit the properties of the well known Lovász Local Lemma (Erdos & Lovász, 1975; Spencer, 1977). To this aim, we first show that if we color each vertex with one of the k available colors uniformly at random, there is positive probability (i.e., strictly greater than zero) that such random coloring returns a constant approximate NE for digraphs that have sufficiently large minimum outdegree. In detail, such graph G must satisfy that, for any $v \in V$, $\delta_o^v(G) = \Omega(\ln \Delta_o + \ln \Delta_i + \ln k)$; for instance, this happens for digraphs where the minimum outdegree is $\Omega(\log n)$. Observe that this implies that constant approximate Nash equilibria always exist for any constant value $k \geq 2$ in this class of digraphs. We emphasize that this result is already interesting, given that the problem of understanding whether the digraph k-coloring game admits a NE is NP-complete for any $k \geq 2$. Then, we combine the above positive probability with the results of Theorem 1.2 of (Moser & Tardos, 2010), in order to design a randomized algorithm that computes a constant approximate NE with high probability and has expected running time that is polynomial in the size of the input. We describe the algorithm we get from Theorem 1.2 of (Moser & Tardos, 2010) at the end of this section.

To start, we summarize the main features of the Lovász Local Lemma that are useful for our purposes. Specifically, the Lovász Local Lemma (LLL) is known for being a powerful tool to be used for demonstrating that, given a large set of events with some dependencies among them, the probability that none of these events happens is strictly greater than 0 if some conditions are met. Several versions of the lemma exist, and here onward we use the following one:

Definition 4.1 (Lovász Local Lemma (LLL) (Erdos & Lovász, 1975))**.** *Let $A_1, A_2, \dots, A_n$ be a set of bad events, with $\bar{A}_1, \dots, \bar{A}_n$ denoting their complements, respectively, and let $D_i \subseteq \{A_1, A_2, \dots, A_n\}$ denote the "dependency set" of each bad event A_i, namely A_i is mutually independent of all the events that are not in D_i. If there exists a set of real numbers $x_1, \dots, x_n \in [0, 1)$ such that $Pr[A_i] \leq x_i \prod_{j \in D_i}(1 - x_j)$ for all i, then $Pr[\bigwedge_{i=1}^{n} \bar{A}_i] \geq \prod_{i=1}^{n}(1 - x_i) > 0$.*

We now show how the LLL can be used for proving that, under certain conditions on the structure of the digraph, there is positive probability that a randomly obtained k-coloring corresponds to a constant approximate NE for any $k \geq 2$. In other terms, we show the existence of constant approximate NE for a broad subclass of digraphs. In detail, to apply Definition 4.1 to our scenario, we define, for each $v \in V$, what a "bad event" I_v is for us in the context of a digraph k-coloring game. Intuitively, we would like to associate I_v to the case in which, in a given coloring c, the vertex v is not γ-happy for some constant $\gamma \geq 1$. However, this can happen only if the coloring c is not balanced for the outgoing neighbors of v, that is at least one color c' occurs much more frequently with respect to the average number $\frac{\delta_o^v}{k}$ of vertices w with $(v, w) \in A$ having the same color. Hence, we prove the stronger statement with respect to the existence of an approximate γ-NE. Namely, that there exists a balanced coloring c, i.e., such that at every vertex v there are at most $(1 + \beta)\,\dfrac{\delta_o^v}{k}$ outgoing neighbors with the same color, for some constant $\beta > 0$. As we are going to show below, if every vertex has enough outgoing neighbors, the value of β can be taken sufficiently small, so as to induce an approximate γ-NE with constant γ.

Consequently, we define:

Definition 4.2 (Bad event)**.** *A bad event I_v is associated to a vertex v when v has at least $(1 + \beta)\,\dfrac{\delta_o^v}{k}$ outgoing neighbors with the same color, for some constant $\beta > 0$.*

Once defined such a bad event, we now show that the LLL holds, under the aforementioned conditions, for all bad events of an instance of the digraph k-coloring game (i.e., there is positive probability none of these events occurs, given a random assignment of colors).

First of all we bound the maximum size of the dependency set of each bad event when each vertex is assigned a color uniformly at random, independently from the other vertices, to incorporate it in Definition 4.1. To this aim, we observe that a generic bad event I_v can be dependent from another event I_w only if v and w have some common outgoing neighbor z, that is if $N_{out}(v) \cap N_{out}(w) \neq \emptyset$. Therefore, if we denote by dep_v the dependency set of I_v, we can upper bound the number of events in such a set as $|dep_v| \leq \sum\limits_{z \in N_{out}(v)} \delta_i^z \leq \delta_o^v \Delta_i \leq \Delta_o \Delta_i$.

Observe that, since both Δ_o and Δ_i do not depend on v, the above inequality holds for every dependency set of every bad event. Hence, we can define $u_d = \Delta_o \Delta_i$ to be an upper bound on the number of dependencies of any vertex, that is $|dep_v| \leq u_d$ for any vertex $v \in V$.

In the following, for the sake of simplicity, and without loss of generality, we assume that $|dep_v| \geq 2$. Indeed, if $|dep_v| = 1$, the only possibility is that the digraph is a collection of disjoint directed paths and cycles. In this case, if there are not cycles of odd length, by alternating colors along the directed paths and cycles, a pure Nash equilibrium can be easily determined for any fixed number $k \geq 2$ colors. If instead at least one cycle has odd length, for $k = 2$ no γ-NE exists for any bounded value of γ, while if $k \geq 3$ a pure NE can be found alternating 3 colors in odd length cycles.

We can then prove the following theorem.

Theorem 4.2. *Given an instance of the digraph k-coloring game with n vertices and a constant number of colors k, there exist values $x_v \in [0, 1)$, for any vertex $v \in \{1, \dots, n\}$, such that, if $\delta_o^v = \Omega(\ln \Delta_o + \ln \Delta_i)$, we have $Pr[I_v] \leq x_v \prod_{w \in dep_v}(1 - x_w)$ for all $v \in V$, that is the LLL holds.*

Proof. In order to show that the claim holds, let us define $x_v = \frac{1}{u_d}$ $\forall v \in V$. To prove the statement, it suffices to show that the following inequality is true:

$$Pr[I_v] \leq \frac{1}{u_d}\left(1 - \frac{1}{u_d}\right)^{u_d} \qquad \forall v \in V \tag{1}$$

Since $u_d \geq |dep_v|$ for any $v \in V$, and since $\left(1 - \frac{1}{u_d}\right) < 1$, we obtain the following implication:

$$Pr[I_v] \leq \frac{1}{u_d}\left(1 - \frac{1}{u_d}\right)^{u_d} \leq \frac{1}{u_d}\left(1 - \frac{1}{u_d}\right)^{|dep_v|} \qquad \forall v \in V \tag{2}$$

which implies that the LLL holds. In order to prove the validity of Equation (1) we use the Chernoff bound (Chernoff, 2011) that gives a bound on the deviation of the sum of random variables from their expected value.

Definition 4.3 (*Chernoff bound*)**.** *Let $X_1, X_2, \dots, X_n$ be independent random variables, and let X be their sum and $\mu = E[X]$ the expected value. Then, for any $\beta > 0$:*

$$Pr[X \geq (1 + \beta)\,\mu] \leq e^{-\frac{\beta^2 \mu}{3}} \tag{3}$$

Notice that in any coloring computed by assigning colors uniformly at random, we have that $\frac{\delta_o^v}{k}$ is the expected number of vertices w having a given color c' and such that there exists a directed edge (v, w). This is true since, with k colors, the probability that a vertex has color c' is $\frac{1}{k}$ and every vertex v has outdegree δ_o^v. We remark that we assume k is a constant value. By the Chernoff bound, the probability of having at least $(1+\beta)\dfrac{\delta_o^v}{k}$ outgoing neighbors of v with color c' is at most $e^{-\frac{\beta^2\delta_o^v}{3k}}$, and by applying the union bound on the set of all the colors, we derive that the probability that v has at least $(1+\beta)\dfrac{\delta_o^v}{k}$ outgoing neighbors with the same color is at most $ke^{-\frac{\beta^2\delta_o^v}{3k}}$. Therefore, having in mind Equation (1), we want to find suitable values of β, δ_o^v and k such that

$$Pr[I_v] \le ke^{-\frac{\beta^2\delta_o^v}{3k}} \le \frac{1}{u_d}\left(1-\frac{1}{u_d}\right)^{u_d} \tag{4}$$

Since the function $(1-\frac{1}{u_d})^{u_d}$ is increasing for any value $u_d > 1$ and moreover we suppose that $u_d \ge 2$, we obtain that $(1-\frac{1}{u_d})^{u_d} \ge \frac{1}{4}$. So we get from Equation (4) that, if $ke^{-\frac{\beta^2\delta_o^v}{3k}} \le \frac{1}{4u_d}$, then Equation (4) is satisfied, and thus LLL holds. Therefore we obtain that for each vertex $v \in V$:

$$\frac{\beta^2\delta_o^v}{3k} - \ln k \ge \ln(u_d 4) = \ln[(\Delta_o\Delta_i)4] = \ln\Delta_o\Delta_i + \ln 4$$

which implies that:

$$\delta_o^v \ge \frac{3k}{\beta^2}(\ln\Delta_o\Delta_i + \ln 4 + \ln k) \tag{5}$$

Thus, since β and k are constant values, it follows that when $\delta_o^v = \Omega(\ln\Delta_o + \ln\Delta_i)$, i.e. when the outdegree of each vertex v is sufficiently large, the LLL holds. For instance we get that the LLL holds for general unweighted digraphs where $d_o = \Omega(\ln n)$. Moreover, with a very similar analysis it is possible to show that if the digraph is such that $\delta_o^v = \delta_o$ for any $v \in V$, i.e, the outdegree of all the vertices is the same (this is a broader class of digraphs with respect to regular graphs) then LLL holds when $\delta_o = \Omega(\ln\Delta_i)$. Finally, we notice that the greater d_o, the smaller β is required to be.

We now show what is the smallest value of γ such that γ-NE exists. To this purpose, we consider the minimum possible value β such that the LLL still holds, that is, according to

Equation (5), equal to $\frac{\sqrt{3k}}{\sqrt{\delta_o^v}}\sqrt{(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}$. Thus, if we write:

$$\gamma = max_{v\in V}\frac{\text{maximum possible payoff of } v}{\text{minimum expected payoff of } v} = \frac{\delta_o^v}{\delta_o^v-(1+\beta)\,\dfrac{\delta_o^v}{k}} = \frac{k}{k-(1+\beta)}$$

$$= \frac{k}{k-1-\dfrac{\sqrt{3k\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{\delta_o^v}}} = \frac{\sqrt{\delta_o^v}}{\sqrt{\delta_o^v}-\dfrac{\sqrt{\delta_o^v}}{k}-\dfrac{\sqrt{3\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{k}}}$$

$$= \frac{\sqrt{\delta_o^v}-\dfrac{k}{k-1}\dfrac{\sqrt{3\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{k}}+\dfrac{k}{k-1}\dfrac{\sqrt{3\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{k}}}{\dfrac{k-1}{k}\sqrt{\delta_o^v}-\dfrac{\sqrt{3\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{k}}} \tag{6}$$

$$= \frac{\sqrt{\delta_o^v}-\dfrac{k}{k-1}\dfrac{\sqrt{3\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{k}}+\dfrac{k}{k-1}\dfrac{\sqrt{3\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{k}}}{\dfrac{k-1}{k}\left(\sqrt{\delta_o^v}-\dfrac{k}{k-1}\dfrac{\sqrt{3\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{k}}\right)}$$

$$= \frac{k}{k-1}+\frac{\dfrac{k}{k-1}\dfrac{\sqrt{3\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{k}}}{\dfrac{k-1}{k}\sqrt{\delta_o^v}-\dfrac{\sqrt{3\,(\ln(\Delta_o\Delta_i)+\ln 4+\ln k)}}{\sqrt{k}}} \approx \frac{k}{k-1}+\left(\frac{k}{k-1}\right)^2\mathcal{O}\left(\frac{1}{r-\dfrac{k}{k-1}}\right)$$

where $r=\frac{\sqrt{\delta_o^v}}{\sqrt{\ln(\Delta_o\Delta_i)}}$, it is easy to see that when $\delta_o^v=\Omega(\ln\Delta_o\Delta_i)$ the above value of γ is constant. □

In summary, we have proved that, when $\delta_o^v=\Omega(\ln\Delta_o+\ln\Delta_i)$ for all $v\in V$, according to the LLL, a random assignment of colors produces with probability larger than zero a balanced coloring, which in turn yields a constant approximate NE.

By the above, and by Theorem 1.2 of (Moser & Tardos, 2010), it is hence possible to derive a simple randomized algorithm, which we name LLL-SPE in what follows, that returns a coloring that is a constant γ-NE with high probability, for γ as in Equation 6. The algorithm starts from a random coloring, which clearly is not necessarily a pure NE nor a γ-NE, and picks, if any, a bad event I_v associated to vertex v.

Then, the algorithm randomly assigns a new color to all the vertices that determine the bad event I_v, that is it randomly determines a new color for each vertex w such that there is an outgoing arc (v,w) from v to w. Such operation is called a *resampling.* The algorithm continues resampling bad events until it reaches a coloring that is a γ-NE for the value of γ we have highlighted in the analysis of Theorem 4.2. If the conditions of the LLL are satisfied for the graph under processing, the result in (Moser & Tardos, 2010) suffices to claim that algorithm LLL-SPE terminates in a γ-NE and the expected number of resampling operations of bad events is polynomial.

It is worth observing that the question of whether the algorithm can be derandomized is an interesting issue at this point. Unfortunately, the results of (Moser & Tardos, 2010), Theorem 1.4 specifically, allow to obtain a deterministic algorithm with a running time that depends exponentially on the maximum number of dependencies per random variable. As a

Algorithm 2: Algorithm LLL-SPE.

```
Input: A digraph $G = (V, A)$, a set $C$ of $|C| = k$ available colors
Output: A $k$-coloring $c$ of $G$
1 Let $c$ be an empty coloring;
2 foreach $v \in V$ do
3     $c_v \leftarrow$ a color in $C$ randomly selected with uniform probability $\frac{1}{k}$;
4 Compute threshold on $\gamma$ as in Equation 6;
5 while $\exists$ a $\gamma$-unhappy vertex do
6     Let $S$ be the set of $\gamma$ unhappy vertices;  /* $c$ is not a $\gamma$-NE here, hence $S \neq \emptyset$ */
7     Select randomly a vertex $v \in S$, with uniform probability $\frac{1}{|S|}$;
8     Randomly recolor vertices in the dependency set $dep_v$ of $v$ with uniform probability $\frac{1}{k}$;
9 return $c$;
```

consequence, if the input graph is such that both the in- and out-degree of each vertex are bounded by a constant (i.e., Δ_o and Δ_i are constant), then the conditions of Theorem 1.4 are satisfied and we directly obtain a deterministic polynomial time algorithm. However, for such a case, the polynomial time, deterministic construction of Algorithm 1 would already return a constant NE with $k = 3$ colors. Thus, for general graphs, understanding whether the algorithm can be derandomized remains an open question, since it is directly connected to the more general question of improving the existing construction results of LLL.

As a last observation, we remark that a simple application of the Chernoff's bound at each vertex and then of the union bound to include all the vertices, would have given a positive existential result for constant approximated NE for digraphs with minimum outdegree $\Omega(\ln n)$. However, as already observed, the finer analysis obtained by applying the Lovász Local Lemma allowed to include much more general classes of digraphs, which properly include regular digraphs with costant degree.

4.3 Computing a $(1+\epsilon)$-Nash Equilibrium with a Logarithmic Number of Colors

In this section we present a polynomial time algorithm that, for any digraph G and $\epsilon > 0$, computes a $(1+\epsilon)$-NE by using $\mathcal{O}\left(\frac{\log n}{\epsilon}\right)$ colors. The algorithm is iterative: at the beginning all the vertices are not colored. Then, during an iteration the algorithm colors a subset of vertices as described in what follows.

Let V' be the current set of uncolored vertices (initially $V' = V$). In a given iteration i, we consider the subdigraph $G' = (V', A')$ of G induced by V' (i.e., any arc of A is in A' only if both endpoints are in V'), and $k' = \lceil \frac{3(1+\varepsilon)}{\varepsilon} \rceil$ new colors $c_{ik'-k'+1}, \ldots, c_{ik'}$ not used in the previous iterations (we use a set C to trace used colors during iterations). We then define an undirected graph $G_u = (V', E)$ having the same vertex set as G' and such that an undirected edge $\{v, w\}$ is in E if at least one arc between (v, w) and (w, v) is in A'. Call δ^v the number of neighbors of v in G_u. Next, we use as sub-routine the algorithm described in (Kun et al., 2013) that computes a stable coloring for the unweighted undirected graph k'-coloring game (from now on we call this algorithm UND-COLOR, pseudocode given in Algorithm 3). The underlying idea of such sub-routine is the following: it starts from any arbitrary coloring and the color of a vertex is changed only if, by doing so, it strictly improves her payoff (i.e., the number of neighbors with different color increases). The algorithm stops when it is no more possible to perform such moves. The solution is computed in polynomial time and it is a NE. Let $V_1, V_2, \dots, V_{k'}$ be the coloring of V' induced by the execution of UND-COLOR on G_u with the k' colors, namely $v \in V_j$ means that v is colored $c_{ik'-k'+j}$.

For each vertex $v \in V'$, if its outdegree in G' is at least $\lceil \frac{\delta^v}{3} \rceil$ then v is colored as in the equilibrium computed by UND-COLOR, namely v is colored $c_{ik'-k'+j}$ if $v \in V_j$. The algorithm updates the set of the uncolored vertices and iterates until all vertices are colored. A formal description of the procedure is given in Algorithm 4. Note that we use $\delta_o^v(T)$ and $\delta_i^v(T)$ to denote outdegree and indegree, respectively, of a vertex v of a generic graph T. We now

Algorithm 3: Algorithm UND-COLOR

```
    Input: A undirected graph $G = (V, E)$, a set $C$ of $|C| = k$ available colors.
    Output: A $k$-coloring of $G$.
1   $c \leftarrow$ random $k$–coloring of $G$ with uniform probability $\frac{1}{k}$;
2   while $\exists$ unhappy vertex $v$ do
3       $c_v \leftarrow \psi_c(v) = argmax_{c'_v \in C}\, \mu_{(c_{-v}, c'_v)}(v)$;   /* Color maximizing payoff of $v$ */
4   return $c$;
```

Algorithm 4: Algorithm APPROX-3

```
    Input: A digraph $G = (V, A)$, parameter $\epsilon > 0$.
    Output: A $k$-coloring of $G$ where $k \leq \frac{6(1+\varepsilon)}{\varepsilon}$.
1   $i = 1$;
2   $V' \leftarrow V$;
3   $C \leftarrow \emptyset$;
4   while $V' \neq \emptyset$ do
5       $k' = \lceil \frac{3(1+\varepsilon)}{\varepsilon} \rceil$;
6       Let $c_{ik'-k'+1}, \dots, c_{ik'}$ be $k'$ colors that are not in $C$;
7       Let $G' = (V', A')$ be a digraph such that $A' := \{(v, w) : v, w \in V'\}$;
8       Let $G_u = (V', E)$ be an undirected graph such that
          $E := \{\{v, w\} : (v, w) \in A' \vee (w, v) \in A'\}$;
9       Let $\delta^v \leftarrow |\{y \in V' : (v, y) \in E\}|$ of vertex $v \in V'$;
10      Execute UND-COLOR on $G_u$ and the selected $k'$ colors;
11      Let $V_1, V_2, \dots V_{k'}$ be the partition of $V$ induced by the $k'$ colors;
12      foreach $v \in V'$ do
13          if $\delta_o^v(G') \geq \lceil \delta^v/3 \rceil$ then
14              Let $V_j$ be the subset vertex $v$ belongs to, in the partition $V_1, V_2, \dots V_{k'}$;
15              $c_v \leftarrow c_{ik'-k'+j}$;
16              $V' = V' \setminus \{v\}$;
17              $C \leftarrow C \cup \{c_v\}$;
18      $i = i + 1$;
```

show the performance and correctness of Algorithm APPROX-3.

Theorem 4.3. *Given any digraph $G = (V, A)$ and $\epsilon > 0$, algorithm* APPROX-3 *returns in polynomial time a $(1 + \epsilon)$-NE by using at most $\frac{6(1+\varepsilon)}{\varepsilon} \log n = \mathcal{O}\left(\frac{\log n}{\epsilon}\right)$ colors.*

Proof. Consider an iteration i and let the coloring returned by UND-COLOR induce a partition $V_1, V_2, \ldots V_{k'}$ of V' in such iteration. Then, for each $v \in V'$, let $\delta^v_{V_j}$ be the number of neighbors that v has in the undirected graph induced by vertices in V_j. If we focus on a given V_j for $j = 1$ to k', we have $\delta^v_{V_j} \leq \delta^v_{V_c}$ for all $v \in V_j$, and for all V_c such that $c \neq j$ as otherwise v could improve its payoff by changing its color, thus violating the fact that the returned solution is stable, i.e., a NE. Now, consider any vertex $v \in V_j$. According to the coloring returned by UND-COLOR, its payoff is given by $\sum\limits_{V_c : j \neq c} \delta^v_{V_c}$ which clearly is larger than or equal to $(k' - 1)\, \delta^v_{V_j}$, i.e., we have:

$$\sum_{V_c : j \neq c} \delta^v_{V_c} \geq \left(k' - 1\right) \delta^v_{V_j}.$$

It follows that, since the degree of v in G_u is δ^v, the number of neighbors in the same subset of vertices as v is at most

$$\delta^v_{V_j} \leq \frac{\delta^v}{k'} = \frac{\delta^v}{\lceil \frac{3(1+\varepsilon)}{\varepsilon} \rceil} \leq \frac{\delta^v \varepsilon}{3\,(1+\varepsilon)}.$$

Let us assume that v is colored in a given coloring c obtained at the end of iteration i. Then, by the construction of APPROX-3, the outdegree of v in G' is $\delta^v_o(G') \geq \lceil \frac{\delta^v}{3} \rceil$, hence the approximation factor γ_v for vertex v is:

$$\gamma_v \leq \frac{\pi_c(v)}{\mu_c(v)} \leq \frac{\delta^v_o(G')}{\delta^v_o(G') - \dfrac{\delta^v \varepsilon}{3\,(1+\varepsilon)}} \leq \frac{\dfrac{\delta^v}{3}}{\dfrac{\delta^v}{3} - \dfrac{\delta^v \varepsilon}{3\,(1+\varepsilon)}} = \frac{1}{1 - \dfrac{\varepsilon}{(1+\varepsilon)}} = 1 + \varepsilon.$$

Thus, v finds itself to be in a $(1+\varepsilon)$-NE, since her payoff cannot be improved by more of a multiplicative factor of $\gamma_v = (1+\varepsilon)$. Therefore, as every vertex is colored during some iteration, APPROX-3 returns a $(1+\varepsilon)$-NE coloring. We now show that APPROX-3 terminates in a polynomial number of iterations by proving that we reduce the number of edges in G_u by a constant multiplicative factor during each iteration, until the algorithm ends. Consider again a generic iteration i. Let W be the set of vertices $v \in V'$ such that $\delta^v_o(G') \geq \lceil \frac{\delta^v}{3} \rceil$, namely those vertices in W that are colored. Call Z be the set of remaining vertices. If A_i is the set of arcs in G' when iteration i starts and A_{i+1} is the set of edges remaining after iteration i, then we can bound such value as follows:

$$|A_{i+1}| \leq \sum_{v \in Z} \delta^v_o(G') \leq \frac{\sum\limits_{v \in Z} \delta^v_i(G')}{2} \leq \frac{|A_i|}{2}$$

The first inequality holds by definition, since we keep only arcs between uncolored vertices. Concerning the second inequality, if by contradiction we assume it is not true, then we would obtain that there is some vertex in Z that has been colored at the end of iteration i, namely it should be in set W, which contradicts the hypothesis. Thus, the maximum number of iterations is $\log|A| \leq 2\log n$. Since we use k' new colors in each iteration, the overall number of used colors is at most:

$$2k' \log n \leq \frac{6\,(1+\varepsilon)}{\varepsilon} \log n = \mathcal{O}\left(\frac{\log n}{\epsilon}\right)$$

□

We remark that the coloring returned by the algorithm remains stable even if we increase the number of colors, so that a $(1+\varepsilon)$-approximate NE exist for any $k = \Omega\left(\frac{\log n}{\varepsilon}\right)$.

5. Heuristic Algorithms for Digraph k-Coloring Games

In this section, we introduce heuristic algorithms that compute approximate NE without guarantees on the achieved γ.

The first approach we present is named MYO-BEST-RESP and is inspired to classical best-response dynamics, which have been shown to be effective in practice to handle games similar to that considered in this work (Swenson, 2017; Swenson, Murray, & Kar, 2018; Cary, Das, Edelman, Giotis, Heimerl, Karlin, Kominers, Mathieu, & Schwarz, 2014). More specifically, we consider what, in the literature, is sometimes referred to as myopic best-response paradigm (Swenson, 2017; Cary et al., 2014), where agents decide their strategy greedily and based on knowledge of their neighbors only. Such method is universally considered one of the most appealing approaches in this domain, since its update rules depend only on local knowledge and hence they are very easy to be translated into distributed algorithms for decentralized systems of agents (Blume, 1995; Swenson et al., 2018). We remark that this is a very relevant domain for digraph k-coloring games, and no distributed solution to compute approximate NE is currently known.

In more details, the idea underlying algorithm MYO-BEST-RESP is to start from a random coloring c. Then, if c is not a pure NE, the algorithm tries to iteratively improve it by applying best response strategies to unhappy vertices. More specifically, during a generic iteration the algorithm performs the following steps: (i) an unhappy vertex, say v, is selected uniformly at random; (ii) the color c_v of the unhappy vertex is set to the color in the strategy set C that maximizes her payoff (ties are broken arbitrarily), i.e., to $\psi_c(v)$, and therefore the vertex achieves a payoff equal to $\pi_c(v)$. The process stops if a NE is reached, i.e., if no unhappy vertex exists in the graph, or when a maximum number of iterations I, given as part of the input, is performed. The pseudocode of procedure MYO-BEST-RESP is summarized in Algorithm 5. Note that we always assume I to be upper bounded by a polynomial with respect to input size. Given the above, the following result easily follows.

Lemma 5.1. *Algorithm* MYO-BEST-RESP *runs in* $\mathcal{O}(n\Delta_o I)$ *time.*

Proof. Observe that executing line 1 takes $\Theta(n)$ time. Moreover, the block of Lines 3–7 is executed at most I times, and strictly less than I times only if a pure NE is found. To this regard, testing the existence of an unhappy vertex in each iteration requires computing the payoff of all vertices in the worst case, which takes $\mathcal{O}(\Delta_o n)$ time since, for each vertex, we need to evaluate the colors of her outgoing neighbors. Selecting at random an unhappy

Algorithm 5: Algorithm MYO-BEST-RESP.

```
  Input: A digraph $G = (V, A)$, a set $C$ of $|C| = k$ available colors, a maximum
         number $I$ of iterations
  Output: A $k$-coloring $c$ of $G$
1 $c \leftarrow$ random $k$–coloring of $G$ with uniform probability $\frac{1}{k}$;
2 $i \leftarrow 0$;
3 while $\exists$ an unhappy vertex and $i < I$ do
4     Let $S$ be the set of unhappy vertices;        /* $c$ is not a NE, hence $S \neq \emptyset$ */
5     Select randomly a vertex $v \in S$, with uniform probability $\frac{1}{|S|}$;
6     $c_v \leftarrow \psi_c(v) = argmax_{c'_v \in C}\, \mu_{(c_{-v}, c'_v)}(v)$;   /* Color maximizing payoff of $v$ */
7     $i \leftarrow i + 1$;
8 return $c$;
```

vertex costs $|S|$ hence $\mathcal{O}(n)$ time, and for said unhappy vertex an additional $\delta_o^v = \mathcal{O}(\Delta_o)$ time in necessary to determine the color that maximizes her payoff. Thus, the claim follows. □

The second heuristic approach we discuss is a modification of algorithm LLL-SPE that is able to produce a k–coloring, regardless of the structure of the input digraph, by including a termination criterion. Note in fact that algorithm LLL-SPE is guaranteed to converge to a constant γ-NE, in a polynomial number of steps (i.e., resampling operations), only in graphs where δ_o^v is $\Omega(\log \Delta_o + \log \Delta_i)$ for any vertex $v \in V$. Since in this work we are interested in evaluating the behavior of such algorithm also in graphs not satisfying this constraint on the degrees, for the sake of completeness, in order to apply it and evaluate its performance on general digraphs, we incorporate in the algorithm a stopping criterion that serves the purpose of bounding the maximum number of resampling operations that the algorithm is allowed to perform. In more details, besides the input digraph and the number of available colors k, the modified method LLL-GEN, summarized in Algorithm 6, takes as input also an integer value I and stops when either a γ-NE is found (for γ as in Equation 6) or a maximum number of iterations I is performed. Note that, throughout the reminder of the paper we assume I to be a polynomial with respect to input size. By such modification, it is easy to see that the following lemma holds.

Lemma 5.2. *Algorithm* LLL-GEN *runs in* $\mathcal{O}(I(n + \Delta_o\Delta_i))$ *time.*

Proof. In each iteration, lines 5 to 8 are executed, with the former requiring $\mathcal{O}(|S|)$ time while the latter performing a number of operations that is bounded by the maximum size of the dependency set of any vertex, i.e., $\mathcal{O}(\Delta_o\Delta_i)$. Note that, through all iterations, a vertex may become unhappy several times, hence the sum of the sizes $|S|$ for all iterations is not upper bounded by n. Thus, the claim follows. □

6. Experimentation

In this section, we first present the results of a preliminary experimentation, whose main focus is to establish how APPROX-1 and LLL-SPE, i.e. the only known algorithm with worst-case guarantees on the achieved γ, perform in practice in terms of approximation and running

Algorithm 6: Algorithm lll-gen.

```
Input: A digraph $G = (V, A)$, a set $C$ of $|C| = k$ available colors, a maximum
       number of iterations $I$
Output: A $k$-coloring $c$ of $G$
1 $c \leftarrow$ random $k$–coloring of $G$ with uniform probability $\frac{1}{k}$;
2 Compute threshold on $\gamma$ as in the proof of Theorem 4.2;
3 $i \leftarrow 0$;
4 while $\exists$ a $\gamma$-unhappy vertex and $i < I$ do
5     Let $S$ be the set of $\gamma$ unhappy vertices; /* $c$ is not a $\gamma$-NE, hence $S \neq \emptyset$ */
6     Select randomly a vertex $v \in S$, with uniform probability $\frac{1}{|S|}$;
7     Randomly, uniformly, color vertices in dependency set $dep_v$ of $v$;
8     $i \leftarrow i + 1$;
9 return $c$;
```

time, since no characterization of the average case of such solutions is known with respect to these terms. Since our preliminary experiments highlight the poor practical performance of algorithms approx-1 and lll-spe, we then present the results of a second, more thorough experimental study, whose aim is to assess the performance of all algorithms considered in this paper for computing approximate NE for digraph k-coloring games, including those without guarantees on the achieved γ, when the number of colors is given as parameter. This part of our study provides strong evidences of the very good performance of the newly introduced heuristic algorithm myo-best-resp in terms of achieved approximation and running time. For the sake of completeness, we conclude the section by showing the results of an experimental study focused on algorithm approx-3, which is treated separately due to its different nature, as it takes as input a parameter $\epsilon > 0$ encoding a target approximation.

Given the above generalization, we implement and test both approx-1 and lll-gen against both artificial and real-world graphs of various sizes and classes, and different values of k. We measured the quality of approximation of the obtained colorings and noticed that in the great majority of the considered inputs such approximation is unsatisfactory for both algorithms. To show this, computed colorings are compared to randomly generated colorings, obtained by randomly, uniformly assigning a color of the strategy set C to each agent with probability $\frac{1}{|C|}$ (we denote this method by random in what follows). In all conducted tests, as a measure of quality of the computed colorings, we focus primarily on the obtained approximation. Specifically, for each graph G and value of k, and for each execution of each algorithm yielding a coloring c, we measure the *approximation ratio* with respect to a pure, exact NE, denoted as $\gamma(G, c)$, as follows:

$$\gamma(G,c) = \begin{cases} \infty & \textbf{if } \exists v \in V : c_v == c_u \ \forall \ u \in N_{out}(v) \\ \max\limits_{v \in V : \delta_o^v > 0} \frac{\pi_c(v)}{\mu_c(v)} & \textbf{otherwise.} \end{cases} \tag{7}$$

In other words, if an algorithm computes a coloring c exhibiting $\gamma(G, c) > 1$ (a sufficient condition for the latter to happen is all vertices having strictly positive payoff, cf. Section 6), it follows that the found coloring is a $\gamma(G, c)$-NE. Moreover, $\gamma(G, c) = 1$ implies c is a pure

NE. Used inputs, parameters and structure of the experimental settings are described in the following section. In what follows, we use LLG, RND and AP1 to refer to algorithms LLL-GEN, RANDOM and APPROX-1, respectively, for the sake of brevity.

6.1 Test Environment and Implementation Details

Our entire test environment is based on NetworKit (Staudt, Sazonovs, & Meyerhenke, 2016), a widely adopted open-source toolkit for implementing graph algorithms and performing network analysis tasks at scale. All our code is written in Python, with some sub-routines in C++/Cython. All tests have been executed, through the Python 3.8 interpreter, under Linux (Kernel 5.3.0-53), on a workstation equipped with an Intel© Xeon© CPU E5-2643 3.40GHz and 128 GB of RAM, and three levels of cache (384KiB L1 cache, 1536KiB L2 cache, 20MiB L3 cache).

As input to our experiments, inspired by other empirical studies on graph algorithms (Angriman, van der Grinten, von Looz, Meyerhenke, Nöllenburg, Predari, & Tzovas, 2019; D'Emidio, 2020; D'Angelo, D'Emidio, & Frigioni, 2019, 2016; D'Andrea, D'Emidio, Frigioni, Leucci, & Proietti, 2014; Borassi & Natale, 2019), we employed a large dataset of digraphs, including: (i) real-world instances, taken from publicly available repositories (Leskovec & Krevl, 2014; Rossi & Ahmed, 2015; Peixoto, 2020); (ii) artificial digraphs, built via well-established random generators, such as, e.g., *Erdős-Rényi* and *Paley* models (Bollobás, 2011). More details on used inputs, including sizes and main characteristics are reported in Table 1. Regarding artificial graphs, generation details are as follows: graphs ERA, ERB, ERC, ERD, and ERE are random graphs generated by the *Erdős-Rényi* model $G(n, p)$, with n vertices and probabilty p for an edge between any two vertices being present in the graph (in our dataset we have selected $n = 1\,000$ and five increasing probabilities $p \in \{0.0125, 0.025, 0.05, 0.1, 0.2\}$, respectively); instances RR3, RR4, and RR5 are random regular graphs where neighbors of each vertex are sampled uniformly at random (in our dataset se have selected degrees 3, 4, and 5, respectively); finally, graphs PL1 and PL2 are random regular (expander) graphs generated by the *Paley* model: given a parameter $p \in \mathbb{Z}$ such that p is a prime power and $p \mod 4 = 1$, the generator generates a graph having vertices in $\mathbb{Z} \setminus p\mathbb{Z}$ and having an edge between vertices x, y if and only if $x - y$ is a non-zero square in $\mathbb{Z} \setminus p\mathbb{Z}$ (in our dataset, we select $p \in \{601, 1181\}$, respectively, since these are the smallest values satisfying the conditions that yielded graphs of reasonable size).

Concerning parameter k, since no direct, analytical relationship has been established between k itself and the approximation the algorithms under study are able to provide, our experimental trials consider carefully selected values of said parameter to investigate how the algorithms' behavior changes as k increases. Specifically, in our experimentation we try to explore such dependency by considering both the reference case of $k = 3$, which is conjectured to represent a threshold on the computational hardness of the problem, and, as suggested by consolidated guidelines for experimental algorithmics to magnify the dependency on parameters in a reasonable number of tests (McGeoch, 2012), values of k spanning the interval $[\max\{4, d_o\}, \Delta_o]$, evenly spaced as multiples of $\frac{\Delta_o - d_o}{5}$, rounded to the closest integer. For iterative algorithms, we fix $I = n \log n$, since for higher values LLL-GEN yields impractical running times in the largest instances, incompatible with those required for an experimental study (see Figure 9).

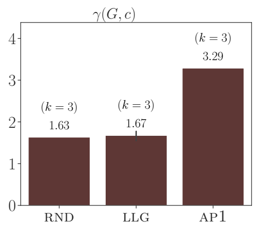
Dataset	**Short**	**Type**	**|V|**	**|A|**	$\overline{d_o}$	$\overline{\overline{d_o}}$	Δ_o	**S**
Twitter	TWI	DIGITAL SOCIAL	23370	33101	1.42	0	238	○
FACEBOOK	FAC	DIGITAL SOCIAL	309717	472792	1.53	0	358	○
AMAZON	AMA	RATINGS	80679	135336	1.68	2	9	○
Flight	FLT	INFRASTRUCTURE	1226	2613	2.13	1	24	○
Peer2Peer	P2P	INTERNET	62586	147892	2.36	0	78	○
Luxembourg	LUX	ROAD	30647	75546	2.47	3	9	○
RAND3	RR3	RANDOM	10000	30000	3	3	3	●
CHICAGO	CHI	ROAD	12978	39017	3.01	3	7	○
DBLP	DBL	CITATION	240	901	3.75	3	68	○
RAND4	RR4	RANDOM	10000	40000	4	4	4	●
Oregon-AS	ORE	AUTONOMOUS SYSTEM	10670	44004	4.12	2	2312	○
RAND5	RR5	RANDOM	10000	50000	5	5	5	●
Health	HEA	HUMAN SOCIAL	2539	12969	5.11	5	10	○
RELATIVITY	REL	COLLABORATION	5242	28968	5.53	3	81	○
Linux	LIN	COMMUNITY	30834	213424	6.92	5	243	○
Peer2PeerSm	SPP	INTERNET	10876	79988	7.35	5	103	○
Google	GOO	HYPERLINKS (LOCAL)	15763	170335	10.81	8	852	○
ARXIV	ARX	DIGITAL SOCIAL	12711	139965	11.01	7	322	○
Erdős-Rényi A	ERA	RANDOM	1000	12460	12.46	12	27	●
Blog	BLG	INTERACTION	1224	19022	15.54	7	256	○
POLITICS	POL	VOTING	793	15781	19.9	12	217	○
Erdős-Rényi B	ERB	RANDOM	1000	24943	24.94	25	45	●
WIKI-Vote	WVT	VOTING	7115	201524	28.32	4	1065	○
Email	EMA	INTERACTION	1005	32128	31.97	21	345	○
USAIR	UAT	TRAFFIC	2058	93823	45.59	10	525	○
Erdős-Rényi C	ERC	RANDOM	1000	49924	49.92	50	74	●
Erdős-Rényi D	ERD	RANDOM	1000	100025	100.03	100	134	●
Erdős-Rényi E	ERE	RANDOM	1000	199443	199.44	199	238	●
PALEY601	PL1	RANDOM	601	180300	300	300	300	●
PALEY1181	PL2	RANDOM	1181	696790	590	590	590	●

Table 1: Overview of used input digraphs. The first three columns contain dataset name, acronym, and type; the 4th and 5th columns show number of vertices and arcs of the digraph, respectively; columns from the 6th to the 8th report average, median and maximum outdegree, respectively. Finally, the 9th column highlights whether the graph is synthetic or real-world (● = true, ○ = false). Inputs are sorted by $\overline{d_o}$, non-decreasing.

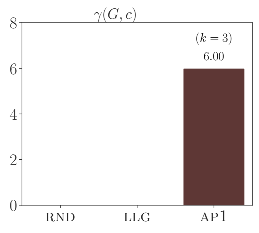
6.2 Preliminary Experimental Evaluation of Algorithms approx-1 and lll-gen

A first excerpt of the results of our experiments is shown in Figure 2 where we compare executions of lll-gen and approx-1 in terms of achieved $\gamma(G, c)$ for various graphs and values of k. Note that, for all algorithms that resort to randomization we execute five trials, for each combination of input and parameters, and computed average values of the observed measures, in order to reduce noise biases.

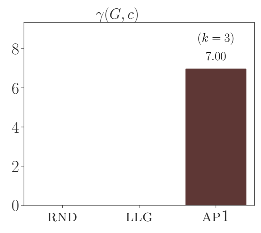
On the one hand, we can observe for instance that, while algorithm approx-1 always finds a Δ_o-NE coloring, it is not rare that random assignments yield better approximations, regardless of the chosen k, in the form of γ-NE colorings with $\gamma \ll \Delta_o$ (see, e.g., Figure 2a or 2f). On the other hand, for several other inputs algorithm approx-1 is better than random assignments and yields colorings that are γ-NE for $\gamma \ll \Delta_o$ (see, e.g., Figure 2b) while colorings produced by random are not even approximate NE, since there exist vertices having unbounded approximation factor (shown in the figure by empty columns). Similarly,



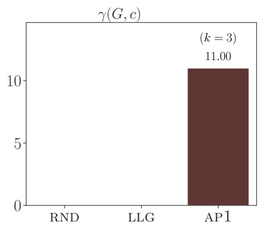
(a)



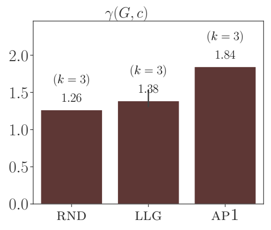
(b)



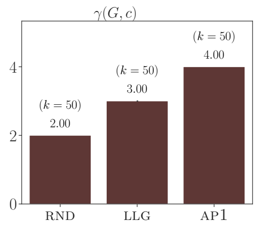
(c)



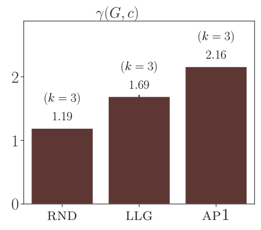
(d)



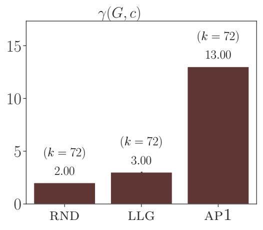
(e)



(f)



(g)



(h)

Figure 2: Empirical observations on the approximation on pure NE provided by colorings computed via algorithms APPROX-1 and LLL-GEN, when applied to a selection of eight inputs graphs, compared to colorings obtained by random uniform assignment (denoted by RND). The x-axis shows the considered algorithms while the y-axis whether the computed coloring is a γ-NE for some finite $\gamma > 0$. We report zero in each column whenever the computed coloring is not a γ-NE for any $\gamma > 0$, i.e., when there exists at least one agent with zero payoff. Considered inputs are: Erdős-Rényi C (a), Oregon-AS (b), Linux (c), Peer2Peer (d), Erdős-Rényi E (e), Twitter (f), paley601 (g), and Email (h).

algorithm LLL-GEN, which is guaranteed to converge to a γ-NE for constant γ, in polynomial time w.h.p. and for digraphs with sufficiently large minimum degree, computes a γ-NE that is of worse approximation quality with respect to random selection on such graphs (see, e.g., Figure 2g). Viceversa, algorithm LLL-GEN surprisingly produces approximate NE colorings with a finite value of γ that, for some inputs, is comparable or better to that of colorings obtained via APPROX-1, despite the graph at hand does not belong to the special class for which LLL-GEN is known to provide constant γ (see, e.g., Figure 2a or 2h).

For all the above reasons, it somehow remains unclear which, among available solutions, is the one that can be considered most suited for practical applications of the considered problem. Thus, motivated by the lack of thorough empirical studies on the subject and by the poor performance of APPROX-1 and LLL-GEN, in what follows we focus on the best-response paradigm, and experimentally evaluate it, against APPROX-1 and LLL-GEN, with the aim of establishing its performance in the considered context.

6.3 Experimental Evaluation of Algorithm myo-best-resp against Algorithms approx-1 and lll-gen

In this section we discuss the experimental work that has been conducted in order to establish the practical effectiveness of algorithm myo-best-resp, given in Section 5. To this purpose, we execute and compare obtained results to those achieved by approx-1 and lll-gen for the same inputs and values of k. Note that this part of the study measures also further performance indicators to better characterize the behavior of the considered algorithms, specifically to discriminate different equilibria exhibiting a same approximation, if any. More in detail, we measure: *average payoff* denoted as $\overline{P}(G,c) = \frac{\sum_{v \in V} \mu_c(v)}{|V|}$ and *fraction of unhappy vertices*, denoted as $U(G,c) = \frac{|\{v \in V : v \text{ is unhappy}\}|}{|V|}$. Finally, for all algorithms, we measure the running time $T(G,c)$ spent to compute c for the given digraph G. Our experimental framework is written in Python, with some sub-routines in C++/Cython, with a fairly optimized code.[1] We leave the problem open of achieving a faster version of all implementations through careful code tuning and/or porting to more high-performance programming languages, e.g., pure C++. Again, note that, for all algorithms that resort to randomization (including mbr in the initial random coloring step) we execute five trials, for each combination of input and parameters, and computed average values of the observed measures, in order to reduce noise biases.

In Table 2 we present a summary of the results of our experimentation, for all input graphs and values of k. In detail, for each considered metric, starting from the most relevant for our study (i.e., approximation factor), we report the number of trials in which each algorithm resulted to be the best (2nd best, 3rd best, worst, respectively) performing one with respect to said metric. Our data highlight that algorithm mbr is the best performing one, globally, in terms of approximation. In fact, out of 210 combinations of input instances and values of k, mbr computes a $\gamma(G,c)$-NE with the smallest value of $\gamma(G,c)$ in 175 of them (83.3% of the combinations). Algorithms rnd, llg and ap1, instead, behave rather badly, being rnd the only one providing the best $\gamma(G,c)$ in the remaining 16.7% of the combinations. Moreover, values of $\gamma(G,c)$ obtained by llg and ap1 are often close to those of rnd, as shown in Figures 3–4 (for $k=3$) and Figures 6–8 (for larger k), which represents a solid evidence of the practical ineffectiveness of the two. Notice that, in such figures we report detailed measures of $\gamma(G,c)$, and of other indicators introduced in this section, for all algorithms and for a meaningful selection of input graphs and values of k. Results for other inputs lead to similar considerations and hence are omitted. Notice also that, in all panels reporting measures of approximation factors, again we use an empty column to show that $\gamma(G,c)$ equals infinity for the computed coloring(s), according to the definition of Equation 7. This condition occurs when in a computed coloring there exists at least one vertex having zero payoff. On the other hand, $\gamma(G,c) = 1$ and $U(G,c) = 0$ correspond to a pure NE being found. On top of the above observations, besides mbr being the best solution with respect to approximation, the most surprising outcome of our experimentation is that mbr is able to compute, in almost all cases, colorings that are pure, exact Nash equilibria (see e.g., Figure 3, Figure 4.middle-top or Figure 5). This is remarkable, considering the known hardness of determining this kind of colorings in general digraphs. More specifically, mbr

1. Code is publicly available at `https://tinyurl.com/bdfsafcp`

METRIC	ALGORITHM	BEST	2ND	3RD	WORST
$\gamma(G,c)$	RND	35 (16.7 %)	84 (40.0 %)	0 (0.0 %)	91 (43.3 %)
	AP1	0 (0.0 %)	91 (43.3 %)	63 (30.0 %)	56 (26.7 %)
	LLG	0 (0.0 %)	0 (0.0 %)	147 (70.0 %)	63 (30.0 %)
	MBR	175 (83.3 %)	35 (16.7 %)	0 (0.0 %)	0 (0.0 %)
$U(G,c)$	RND	0 (0.0 %)	161 (76.7 %)	49 (23.3 %)	0 (0.0 %)
	AP1	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)	210 (100.0 %
	LLG	0 (0.0 %)	49 (23.3 %)	161 (76.7 %)	0 (0.0 %)
	MBR	210 (100.0 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)
$\overline{P}(G,c)$	RND	0 (0.0 %)	0 (0.0 %)	91 (43.3 %)	119 (56.7 %)
	AP1	0 (0.0 %)	0 (0.0 %)	119 (56.7 %)	91 (43.3 %)
	LLG	7 (3.3 %)	203 (96.7 %)	0 (0.0 %)	0 (0.0 %)
	MBR	203 (96.7 %)	7 (3.3 %)	0 (0.0 %)	0 (0.0 %)
$T(G,c)$	RND	210 (100.0 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)
	AP1	0 (0.0 %)	203 (96.7 %)	7 (3.3 %)	0 (0.0 %)
	LLG	0 (0.0 %)	0 (0.0 %)	42 (20.0 %)	168 (80.0 %)
	MBR	0 (0.0 %)	7 (3.3 %)	161 (76.7 %)	42 (20.0 %)

Table 2: Aggregate statistics for all tested algorithms with respect to the four performance indicators of interest, for all 210 considered combinations of inputs and values of k. The first column shows the indicator, the second column shows the considered algorithm while columns from the 3rd to the 6th report the number of trials (and the percentage of trials) in which the algorithm has been the best (2nd best, 3rd best, worst) performing one with respect to the corresponding indicator.

is able to find, often, pure Nash equilibria in less than n iterations, for the large majority of the considered graphs and for all values of k. An example of this behavior is shown in Figures 7a–7b, respectively, where we can observe how the value of $\gamma(G,c)$ induced by algorithm MYO-BEST-RESP, at some point quickly converges to a pure NE, by approaching and then stabilizing at 1, in much less than $n \log n$ and n iterations, respectively.

The only exceptions to this very effective optimization by MYO-BEST-RESP are Erdős-Rényi instances (where however, in some case, MYO-BEST-RESP still achieves the best approximation). In these latter inputs, as shown in Figures 7c–7d, $n \log n$ iterations do not suffice to achieve convergence at pure NE and colorings computed by MYO-BEST-RESP exhibit a $\gamma(G,c)$ that appears to (i) become periodic at some point of the optimization process and (ii) to stabilize around some value slightly above 1 (around 2 and 1.5, in the specific cases). Nonetheless, for larger values of k (see Figure 6.bottom), we notice that MBR manages to find pure NE even in instances for which neither MYO-BEST-RESP nor other methods were able to compute a pure equilibrium for low values of k (e.g., Erdős-Rényi graphs). This is another peculiar element supporting the good performance of MYO-BEST-RESP in practice. Moreover, it might be an hint of the problem being "computationally easier" to be attacked when k is above some threshold, function of the structure/size of the graph. This aspect certainly deserves additional investigation and we leave the problem open of characterizing this relationship, if any.

Observe that, when MBR ranks 2nd best in terms of $\gamma(G,c)$, algorithm RND results to be the best performing one (see, e.g., Figures 4.top or 4.bottom). At the same time, MBR exhibits higher average payoff and lower fraction of unhappy vertices, which suggests that

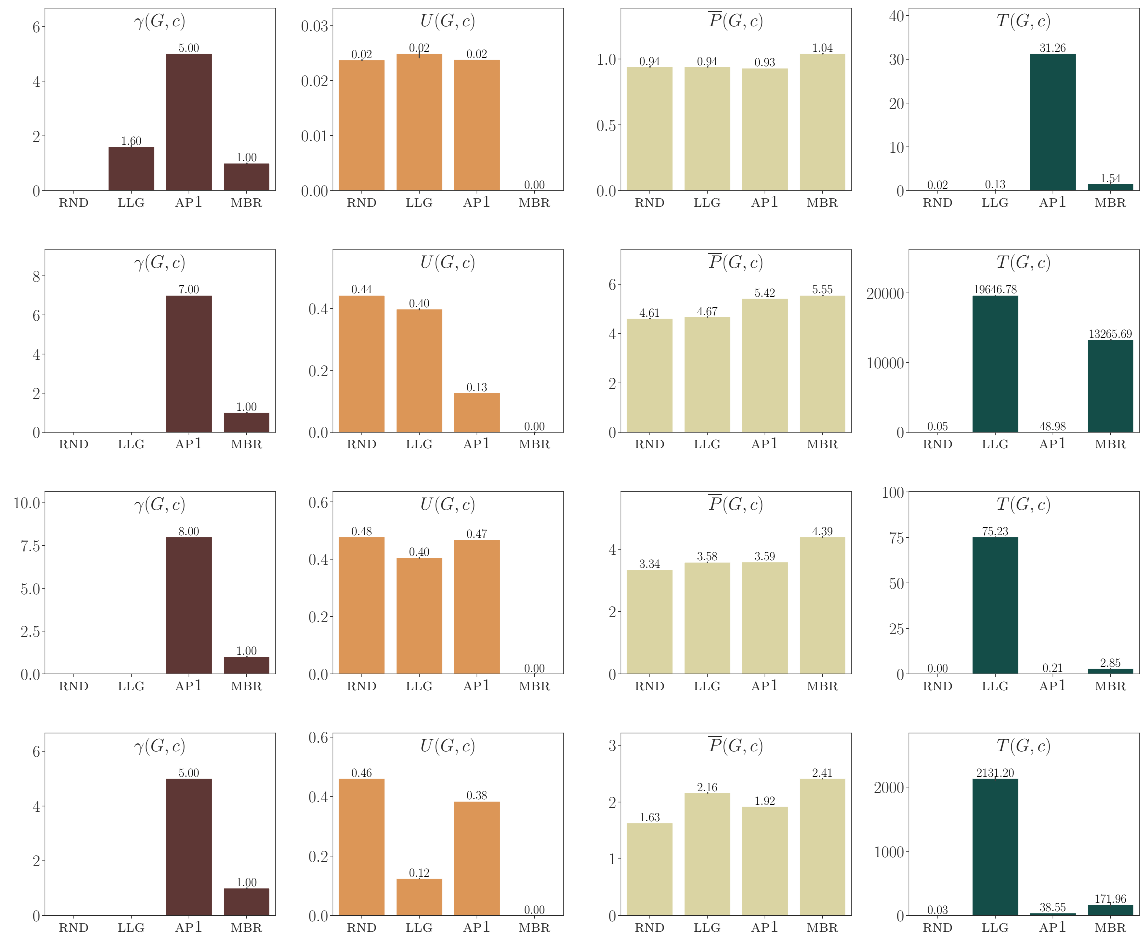


Figure 3: Performance of algorithms RND, LLG, AP1 and MBR, respectively, in graphs TWI (top), LIN (middle-top), HEA (middle-bottom), and LUX (bottom) with $k = 3$. Running time $T(G,c)$ is expressed in seconds.

random assignment might be "lucky" in picking and making happy high-degree vertices, while algorithm MBR is able to achieve maximum payoff for a large fraction of the vertex set. By the above, we conjecture that there might exist an analysis for algorithm MBR to prove a bounded approximation ratio for a broad class of graphs.

In this direction, it is worth noticing that, unexpectedly, LLG fails at achieving the best approximation even in graphs where the LLL is satisfied (i.e., where LLG finds constant approximation in expected polynomial time). This might be due the fact that the threshold value of γ, for which LLG stops, is rather high, often larger than Δ_o (e.g., 2690.36 for instance ERC, see Equation 6 with $k = 3$). In this respect, to achieve better results in practice, a possibility (without any guarantee) could be removing such stopping criterion from LLL-GEN to let the coloring being updated, via resampling, for a maximum number of iterations, as done by MYO-BEST-RESP. For the sake of fairness and completeness, we considered also such

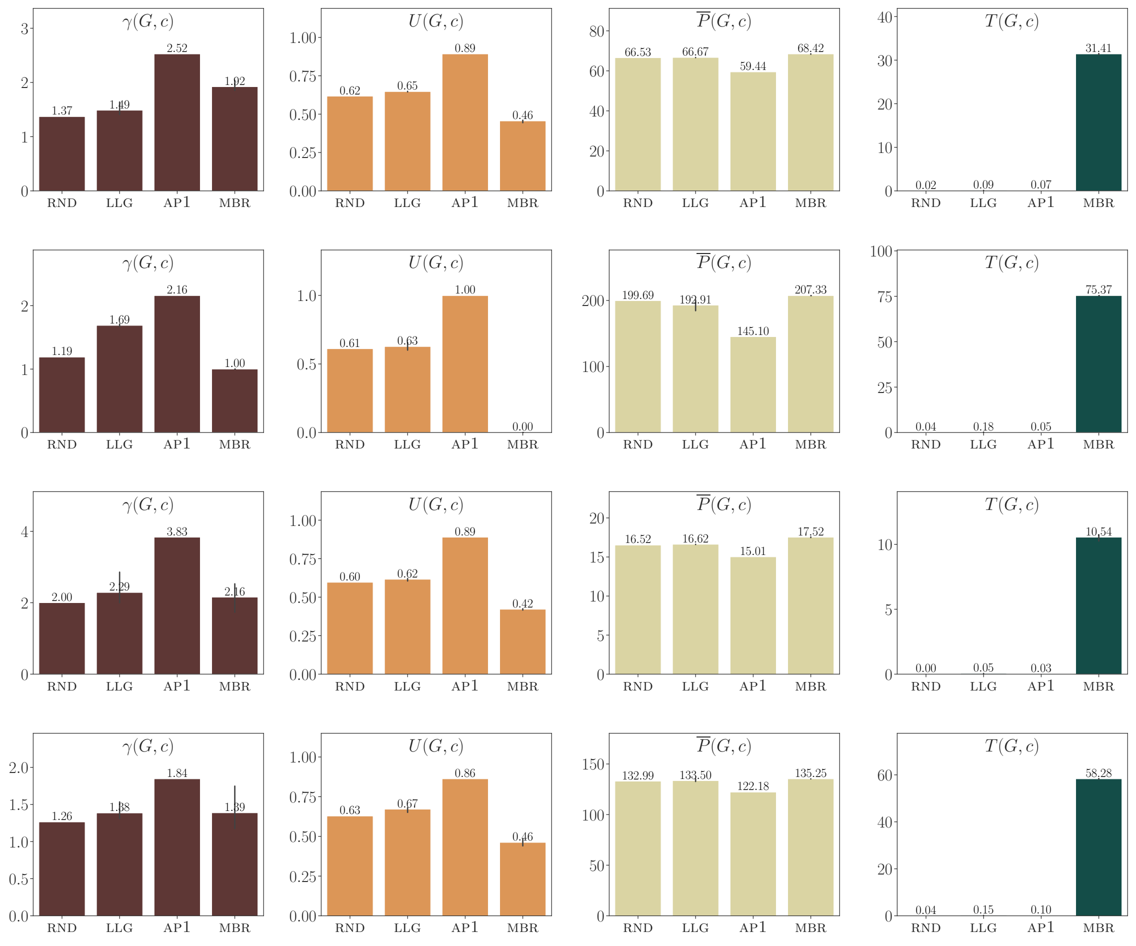


Figure 4: Performance of algorithms RND, LLG, AP1 and MBR, respectively, in graphs ERD (top), PL1 (middle-top), ERB (middle-bottom) and ERE (bottom), with $k = 3$. Running time $T(G, c)$ is expressed in seconds.

possibility and modified and tested our implementation of LLL-GEN without the stopping criterion based on the achieved γ.

Nonetheless, further experimentation, omitted for the sake of brevity, shows that this does not lead to meaningful improvements, in terms of both approximation and other indicators, and the observed behavior is very close to the one shown in this experimental section. We can hence conclude that repeated operations of resampling of the dependency set are indeed enough to obtain constant approximation w.h.p. but result to be empirically ineffective. Thus, another outcome of our study is that the use of LLG is discouraged for practical purposes, unless more effective ways of exploiting the LLL can be determined.

Regarding the impact of varying k on the performance of the considered algorithms, we observe that, while MBR remains the best performing in essentially all cases, approximation ratio and fraction of unhappy vertices (running time and average payoff, respectively) tend

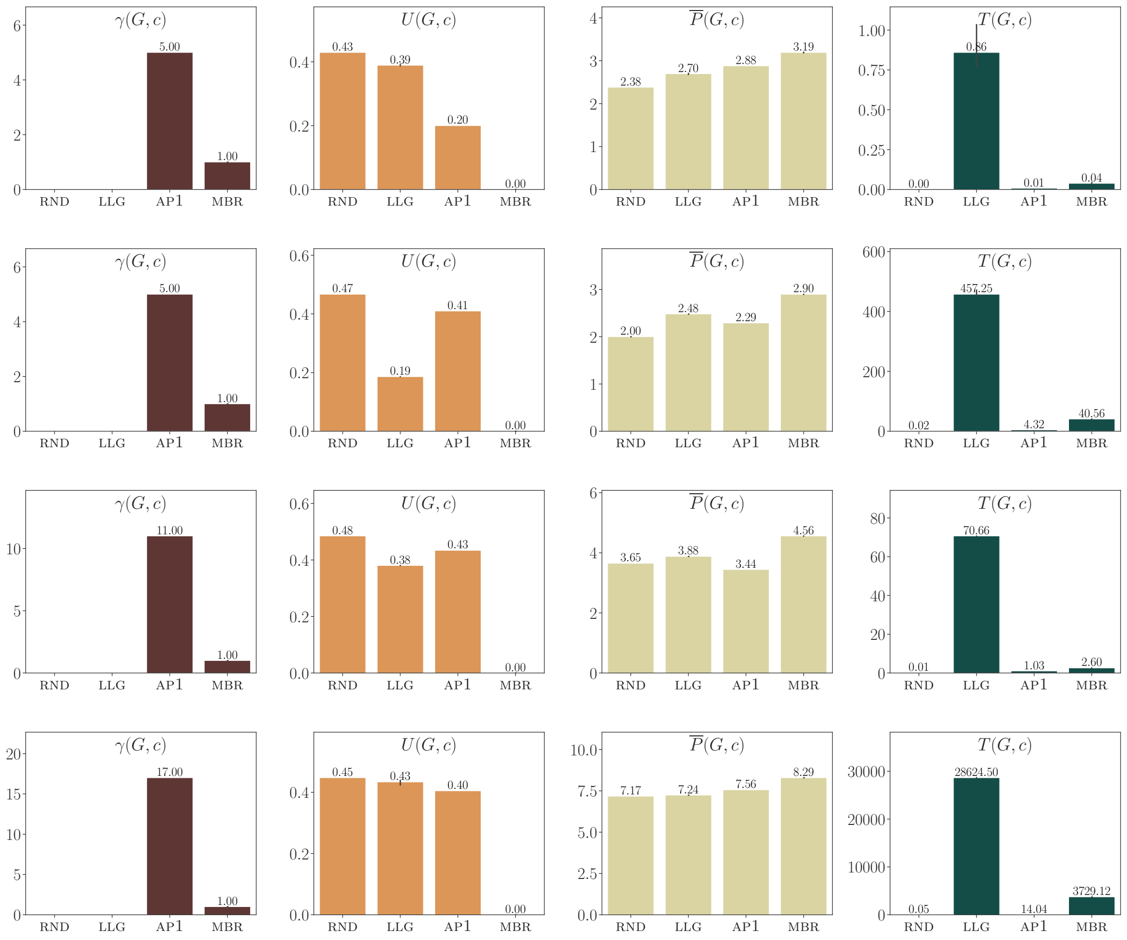


Figure 5: Performance of algorithms RND, LLG, AP1 and MBR, respectively, in graphs DBL (top), CHI (middle-top), REL (middle-bottom) and GOO (bottom), with $k = 3$. Running time $T(G, c)$ is expressed in seconds.

to decrease (increase, respectively) with k, for all algorithms. This might suggest that larger values of k could reduce the possibility of being unhappy, by increasing the choices of the agents in the strategy set. Further investigation is however necessary also here to find out whether there exists some analytical relationship between k, Δ_o, and the quality of the equilibrium that can be achieved, or some computational barrier on the computability of good NE depending on k.

As an additional remark, concerning running time, our data mostly confirm what it is expected, i.e., that: (i) RND is trivially the fastest method; (ii) in some cases MBR is the most time consuming solution (yet achieving the best approximation or values of gamma very close to the best approximation); (iii) in the remaining cases, LLG yield the largest $T(G, c)$ and does not achieve the best approximation (see Figure 9 for the largest input instance considered in this study). These latter observations represent further evidence of MBR being

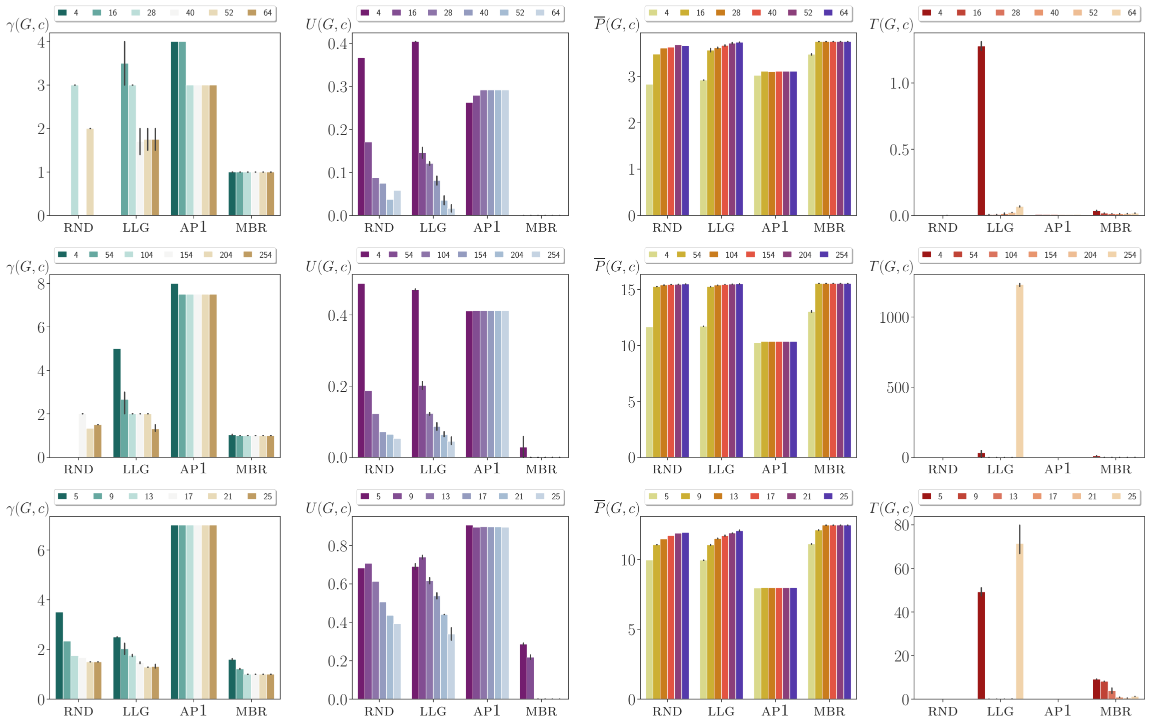


Figure 6: Performance of algorithms RND, LLG, AP1 and MBR, respectively, in graphs DBL (top) and BLG (middle) and ERA (bottom), with increasing values of k. Running time $T(G, c)$ is expressed in seconds.

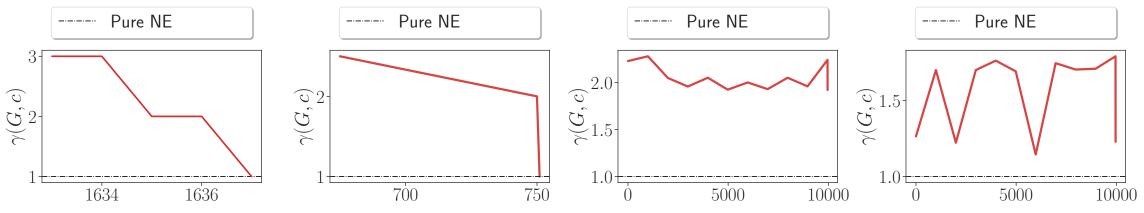

(a) (b) (c) (d)

Figure 7: Results achieved by the execution of algorithm MBR, in terms of $\gamma(G, c)$, on graphs REL (a), EMA (b), ERC (c) and ERE (d) with $k = 3$. The y–axis shows the measured value of $\gamma(G, c)$ as a function of the number of iterations performed by the algorithm, reported on the x-axis. Initial iterations where $\gamma(G, c)$ is unbounded, due, e.g., to some vertex having zero payoff are omitted for the sake of readability.

effective at converging to a pure NE, when it exists. Finally, in all cases we observe the average payoff tends to the average degree of the input graph and, in the few cases when a pure NE is not determined, the fraction of unhappy vertices is always very close to zero, showing that MYO-BEST-RESP is able to compute a coloring c that assigns $\psi_c(v)$ to the payoff of most vertices v of the instance. Note that, in this paper, all algorithms do not take into

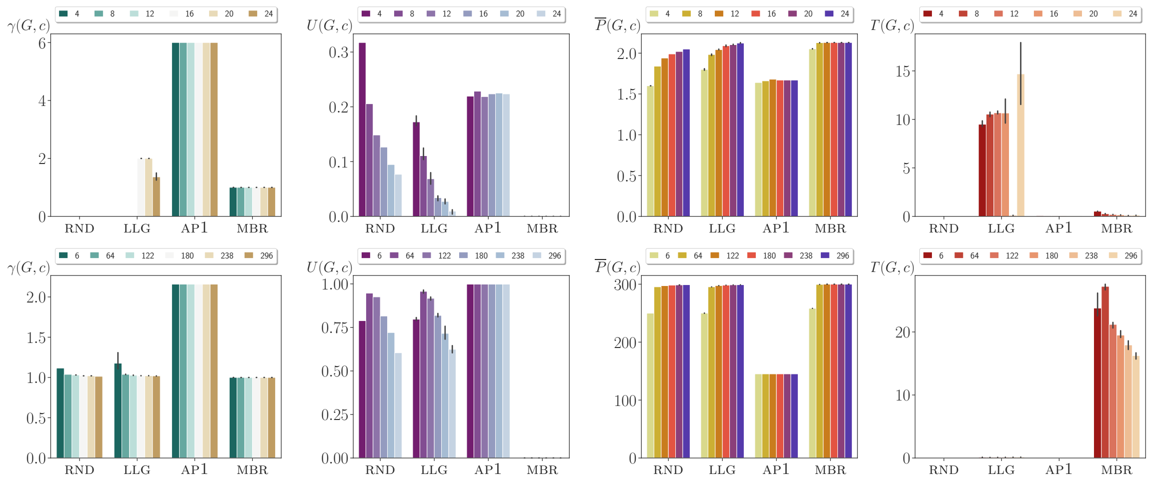


Figure 8: Performance of algorithms RND, LLG, AP1 and MBR, respectively, in graphs FLT (top) and PL1 (bottom), with increasing values of k. Running time $T(G, c)$ is expressed in seconds.

account the maximization of the achieved payoff as optimization objective, and agents only aim at finding themselves in a NE, while several equilibria, exhibiting a same γ but with different payoffs for agents, might exist (as our experimentation highlights). We leave the problem of considering optimization of payoffs in the process of achieving good equilibria open for further investigation. This could of both practical and theoretical interest, as it might lead to finding bounds on other metrics that are often considered in game theoretical contexts (e.g., price of stability).

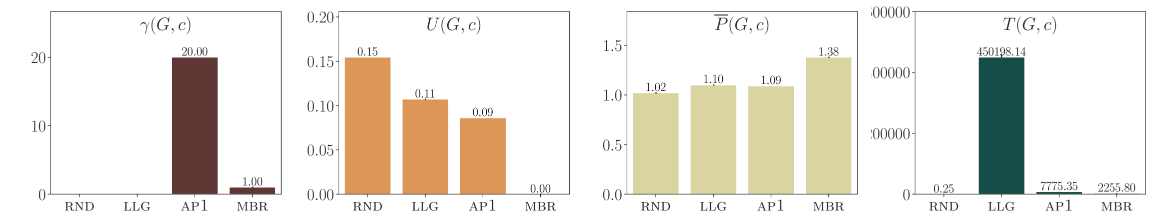


Figure 9: Performance of algorithms RND, LLG, AP1 and MBR, respectively, in the largest input graph, FAC, with $k = 3$. Running time $T(G, c)$ is expressed in seconds. LLG performs all the $n \log n$ iterations without reaching any equilibrium, and results to be the most time consuming algorithm at scale. Algorithm MYO-BEST-RESP finds a pure NE and is faster than APPROX-1, while RANDOM does not yield any bounded approximation equilibrium.

To summarize, our experimental study identifies algorithm MBR as the best performing one for digraph k-coloring games. This observation, combined with the fact that (myopic) best-response approaches are easy to implement, even in a distributed uncoordinated environment, suggests that MBR is strongly advised to find good Nash equilibria in application domains where the considered game arises, even if it does not provide theoretical guaran-

tees on the approximation ratio. On the contrary, the usage of algorithms with guarantees, considered in this work, is discouraged in practice, given their limited performance.
To this regard, it is challenging to provide unquestionable motivations of why AP1 and LLL-SPE/LLL-GEN perform poorly, also when compared to algorithm MBR, given the substantial differences among the three approaches. Concerning AP1, we remark that the algorithm is deterministic and explicitly designed only to guarantee that each vertex has an outgoing neighbor with a different color. While this yields a Δ_o approximation for the resulting equilibrium in the worst case, in practice on the one hand this strategy might correspond to a good performance for graphs with very low Δ_o; on the other hand, we might have a worse performance in dense graphs or graphs that tend to be regular w.r.t. their degree (the coloring phase of vertices in L' does not take into account the vertices' degrees and does not apply any optimization to payoffs). Regarding LLL-SPE, similarly, the algorithm is designed to guarantee a given constant approximation γ, within expected polynomial time, without minding the expected ratio and without incorporating any mechanism oriented to minimizing such ratio with local improvements. In fact, the resampling step is completely random and does not aim at optimizing the number of vertices with different colors.

The above two arguments represent tentative explanations on why both AP1 and LLL-SPE/LLL-GEN achieve low average approximation, especially if compared to the best-response-based MBR algorithm, which instead at each iteration includes a step focused on maximizing the local improvement for a vertex. However, we remark that, while MBR reveals to be a valid and practically effective heuristic, it has lower theoretical guarantees with respect to both AP1 and LLG.

Notice that, even if the above observations shed some light on the possible causes of the bad average performance of the two approaches with guarantees on the approximation, there are no data or strong evidence, however, supporting these hypotheses. It would be interesting, therefore, to provide further characterizations of the practical performance of the two algorithms, also on the basis of more extensive experimental evidences or theoretical results.

6.4 Experimental Evaluation of Algorithm APPROX-3

In this section, we complete our study by experimentally evaluating algorithm APPROX-3. Since the algorithm takes as input a parameter $\epsilon > 0$ and returns a $(1+\epsilon)$-NE by using at most $\frac{6(1+\varepsilon)}{\varepsilon}\log n = \mathcal{O}(\frac{\log n}{\epsilon})$ colors, in what follows we show the results of a set of tests performed on the same graphs of Table 1 with values of $\epsilon \in \{0.2, 0.4, 0.8, 1.6\}$. We implemented and tested the algorithm against the same inputs used for the evaluation of algorithms AP1, MBR, LLG, described above, and considered the same performance indicators.

A summary of our experiments is shown in Figures 10–11. Other results are omitted as they lead to the very same empirical conclusions. In particular, our data show that algorithm APPROX-3 performs well in practice and that the theoretical analysis provides a pessimistic characterization of the practical performance of the algorithm itself. In fact, we can observe that, in several cases, the algorithm ends up in finding, quite quickly, pure NE by using much less than $\frac{6(1+\varepsilon)}{\varepsilon}\log n$ colors (e.g., see Figures 10.bottom and 11.top and corresponding running times). Moreover, in those cases where a pure NE is not determined, the approximation found is far smaller than the $1+\epsilon$ imposed by parameter (see Figure 10.top). This suggests that either it might be possible to tighten the analysis of the approximation factor (in general or for special graph classes) or there might exists an algorithm that finds a better equilibrium, with respect to the one computed by APPROX-3, in polynomial time, for $k = \mathcal{O}(\log n)$. Finally, in all cases the average payoff tends to the average degree of the input graph and, in the few cases when a pure NE is not determined, the fraction of unhappy vertices is always very close to zero, showing that APPROX-3 is able to compute a coloring c that induces $\pi_c(v)$ to be the payoff of most vertices v of the instance.

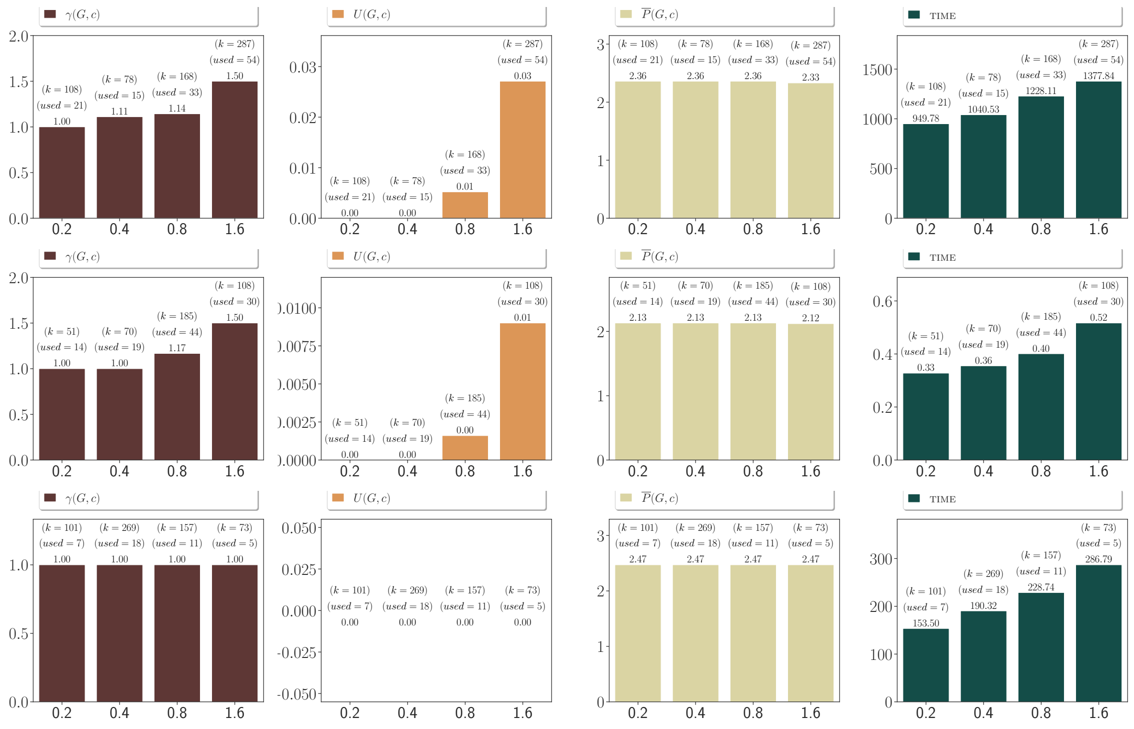


Figure 10: Performance of algorithm APPROX-3 when applied to graphs P2P (top), FLT (middle), LUX (bottom), with increasing (doubling) values of $\epsilon \in \{0.2, 0.4, 0.8, 1.6\}$.

7. Conclusion and Future Work

In this paper we advanced the state of the art on digraph k-coloring games, where selfish agents, i.e., vertices of an n-vertex input graph, aim at anti-coordination by selecting one of k possible strategies, i.e., colors, in order to maximize their payoff, given by the number of outgoing neighbors selecting a different strategy. Such games model several prominent real-world scenarios and are related to some of the most fundamental classes of games that have been investigated in the literature.

We focused on the most important notion of stability in this field, i.e., the Nash equilibrium, and, motivated by the fact that equilibria of this kind might not exist for every $k < n$ and that, in general, their determination is an NP-hard problem, we considered the milder notion of approximate Nash equilibrium. In this context, we designed and analyzed both

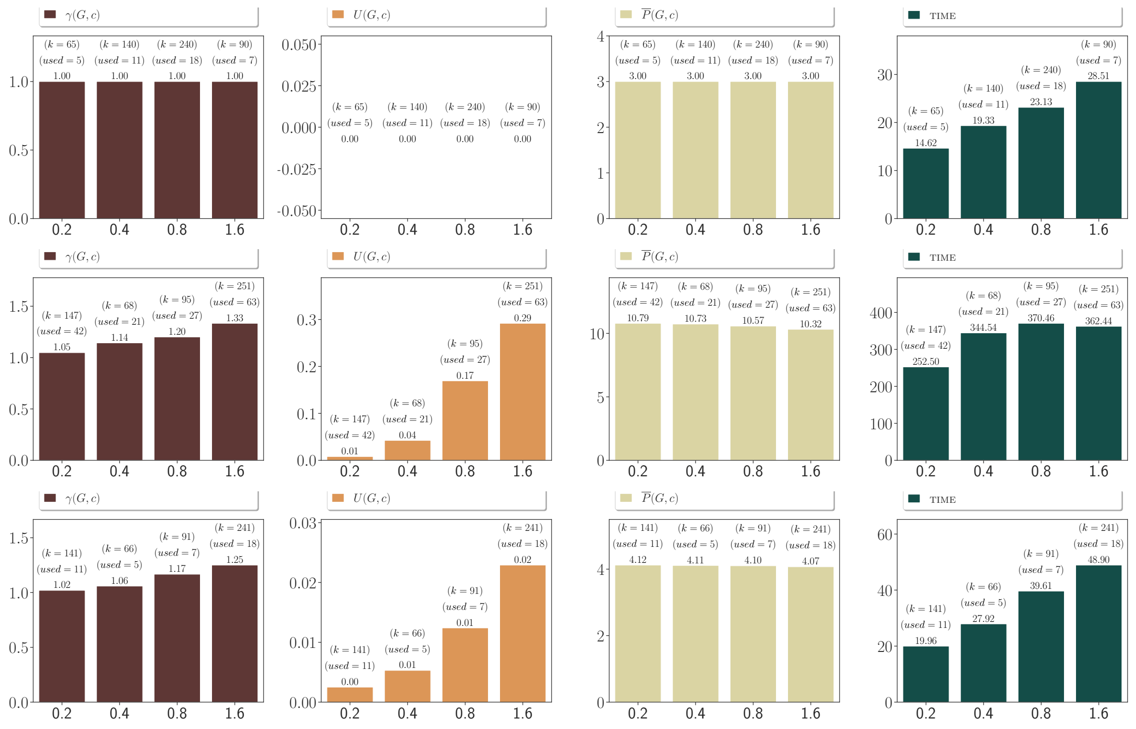


Figure 11: Performance of algorithm APPROX-3 when applied to graphs RR3 (top), GOO (middle), ORE (bottom), with increasing (doubling) values of $\epsilon \in \{0.2, 0.4, 0.8, 1.6\}$.

deterministic and randomized approximation algorithms (namely APPROX-1 and LLL-SPE) that, in polynomial time and given a number k of colors, return NE with worst-case guarantees on the approximation either in general graphs or in those having sufficiently large minimum outdegree. Furthermore, we introduced a third deterministic algorithm, named APPROX-3, that, for general digraphs and for any $\epsilon > 0$, computes a $(1+\epsilon)$-Nash equilibrium by using $O(\frac{\log n}{\varepsilon})$ colors, in polynomial time. Note that such a construction shows that $(1+\epsilon)$-Nash equilibria exist and can be computed in polynomial time for $k = \Omega(\frac{\log n}{\varepsilon})$ and this contrasts to the fact that, for any $k < n$, pure Nash equilibria are not guaranteed to exist and in general cannot be computed in polynomial time, unless $P = NP$. Clearly these results are stronger than mere existential ones, since they imply directly the existence and the fact that such equilibria can be reached in practical scenarios.

Unfortunately, through careful experimentation conducted on both real-world and synthetic inputs, we observed that while solutions APPROX-1 and LLL-SPE are suitably designed to achieve bounded worst case behavior, they generally exhibit a poor performance in practice. Hence, we shifted our attention on best-response dynamics and introduced heuristic MYO-BEST-RESP, based on this principle. We show that, while such algorithm does not provide any upper bound on the approximation of the achieved equilibrium, it results to be very effective in practice by computing pure, non-approximate equilibria in the large majority of the considered inputs, and by widely outperforming algorithms with guarantees. Hence, it can be considered the natural candidate to be used in practice for addressing this class of games.

7.1 Future Work

Our study certainly motivates further research effort on this relevant class of games, since several interesting questions, both of theoretical and empirical type, remain open. First, our experiments suggest that, in general, it might be possible to design an algorithm computing Nash equilibria with a better guarantee on the worst case approximation factor. In fact, also for the very few cases in which a pure Nash equilibrium is not identified by myo-best-resp, the returned solution is always an equilibrium with a very low approximation ratio. Secondly, myo-best-resp being able to find pure NE, in most of the cases, suggests that the existence of a NE might be proved for special, broad classes of graphs, especially those typically arising from social phenomena. Moreover, it would be also worthwhile to prove that good approximate Nash equilibria always exist and, specifically, to understand what is the minimum value of γ such that a γ-approximate equilibria can always be found. In this direction, it is worth noticing that, a viable strategy to obtain lower bounds on the approximation factor could be that of exploiting a linear programming formulation for the problem, as done for other combinatorial problems in the past. Finally, it would be interesting to understand whether the equilibrium computed by approx-3 is the best approximation one can determine, in polynomial time, for $k = O(\log n)$.

Note that, a further step of progress toward a full characterization of digraph k-coloring games is that of considering weighted digraphs. Even if we did not treat this case explicitly, we emphasize that our result for $k = \Omega\left(\frac{\log n}{\epsilon}\right)$ colors can be generalized to show the existence of an equilibria with the same approximation ratio for $k = \Omega\left(\frac{\log W}{\epsilon}\right)$, where W is the overall sum of the edge weights. However, it remains unclear whether such an equilibria can be identified in polynomial time. In this wording, it is also worth remarking that, even when the graph is weighted and undirected (in such a case pure Nash equilibria always exists since we have a potential game, however the problem of computing pure Nash Equilibria is PLS-complete), efficient algorithms that compute constant approximate Nash equilibria only exist for the case when $k = 2$, that is for the cut game, with $\gamma = 3 + \epsilon$ (Bhalgat et al., 2010; Caragiannis et al., 2017). Moreover, even for such a particular setting, there is no evidence that the current proposed solutions are the best approximate Nash equilibria that can be computed in polynomial time, i.e., that the known results are tight.

Acknowledgments

Preliminary versions of the results presented in this paper have appeared in (Carosi, Flammini, & Monaco, 2017; D'Ascenzo, D'Emidio, Flammini, & Monaco, 2022).
Most of Andrea D'Ascenzo's work was carried out during his Ph.D. studies at University of L'Aquila.
This work was partially supported by: (i) Italian Ministry of University and Research through Project "EXPAND: scalable algorithms for EXPloratory Analyses of heterogeneous and dynamic Networked Data" (PRIN grant n. 2022TS4Y3N), funded by the European Union - Next Generation EU; (ii) Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM); (iii) Project EMERGE, innovation agreement between MiSE, Abruzzo Region, Radiolabs, Elital, Leonardo, Telespazio, University

of L'Aquila and Centre of EXcellence EX-Emerge (Grant n. 70/2017); (iv) European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - program RESTART), project MoVeOver/SCHEDULE ("Smart interseCtions witH connEcteD and aUtonomous vehicLEs", PE00000001, CUP J33C22002880001); (v) PNRR MIUR project FAIR - Future AI Research (PE00000013), Spoke 9 - Green-aware AI; (vi) PNRR MIUR project VITALITY (ECS00000041), Spoke 2 ASTRA - Advanced Space Technologies and Research Alliance.

References

Angriman, E., van der Grinten, A., von Looz, M., Meyerhenke, H., Nöllenburg, M., Predari, M., & Tzovas, C. (2019). Guidelines for experimental algorithmics: A case study in network analysis. *Algorithms*, *12*(7), 127.

Anshelevich, E., & Sekar, S. (2014). Approximate equilibrium and incentivizing social coordination. In *Proceedings of the 28th Conference on Artificial Intelligence (AAAI 2014)*, pp. 508–514. AAAI Press.

Apt, K. R., de Keijzer, B., Rahn, M., Schäfer, G., & Simon, S. (2017). Coordination games on graphs. *Int. J. Game Theory*, *46*(3), 851–877.

Aziz, H., Brandt, F., & Seedig, H. G. (2013). Computing desirable partitions in additively separable hedonic games. *Artif. Intell.*, *195*, 316–334.

Aziz, H., & Savani, R. (2016). Hedonic games. In Brandt, F., Conitzer, V., Endriss, U., Lang, J., & Procaccia, A. D. (Eds.), *Handbook of Computational Social Choice*, pp. 356–376. Cambridge University Press.

Ballester, C. (2004). Np-completeness in hedonic games. *Games Econ. Behav.*, *49*(1), 1–30.

Bhalgat, A., Chakraborty, T., & Khanna, S. (2010). Approximating pure nash equilibrium in cut, party affiliation, and satisfiability games. In *Proceedings of the 11th ACM Conference on Electronic Commerce (EC 2010)*, pp. 73–82. ACM.

Bilò, V., Fanelli, A., Flammini, M., Monaco, G., & Moscardelli, L. (2018). Nash stable outcomes in fractional hedonic games: Existence, efficiency and computation. *J. Artif. Intell. Res.*, *62*, 315–371.

Bilò, V., Fanelli, A., Flammini, M., & Moscardelli, L. (2011). Graphical congestion games. *Algorithmica*, *61*(2), 274–297.

Blume, L. E. (1995). The statistical mechanics of best-response strategy revision. *Games and Economic Behavior*, *11*(2), 111–145.

Bogomolnaia, A., & Jackson, M. O. (2002). The stability of hedonic coalition structures. *Games Econ. Behav.*, *38*(2), 201–230.

Bollobás, B. (2011). *Random Graphs, Second Edition*, Vol. 73 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press.

Borassi, M., & Natale, E. (2019). KADABRA is an adaptive algorithm for betweenness via random approximation. *ACM J. Exp. Algorithmics*, *24*(1), 1.2:1–1.2:35.

Cai, Y., Candogan, O., Daskalakis, C., & Papadimitriou, C. H. (2016). Zero-sum polymatrix games: A generalization of minmax. *MOR*, *41*(2), 648–655.

Caragiannis, I., Fanelli, A., & Gravin, N. (2017). Short sequences of improvement moves lead to approximate equilibria in constraint satisfaction games. *Algorithmica*, *77*(4), 1143–1158.

Carosi, R., Fioravanti, S., Gualà, L., & Monaco, G. (2019). Coalition resilient outcomes in max k-cut games. In *Proceedings of the 45th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2019)*, Vol. 11376 of *Lecture Notes in Computer Science*, pp. 94–107. Springer.

Carosi, R., Flammini, M., & Monaco, G. (2017). Computing approximate pure nash equilibria in digraph k-coloring games. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2017)*, pp. 911–919. ACM.

Carosi, R., & Monaco, G. (2020). Generalized graph k-coloring games. *Theory Comput. Syst.*, *64*(6), 1028–1041.

Cary, M., Das, A., Edelman, B., Giotis, I., Heimerl, K., Karlin, A. R., Kominers, S. D., Mathieu, C., & Schwarz, M. (2014). Convergence of position auctions under myopic best-response dynamics. *ACM Trans. Economics and Comput.*, *2*(3), 9:1–9:20.

Chernoff, H. (2011). Chernoff bound. In Lovric, M. (Ed.), *International Encyclopedia of Statistical Science*, pp. 242–243. Springer.

D'Andrea, A., D'Emidio, M., Frigioni, D., Leucci, S., & Proietti, G. (2014). Experimental evaluation of dynamic shortest path tree algorithms on homogeneous batches. In *Proceedings of the 13th International Symposium on Experimental Algorithms (SEA 2014)*, Vol. 8504 of *Lecture Notes in Computer Science*, pp. 283–294. Springer.

D'Angelo, G., D'Emidio, M., & Frigioni, D. (2016). Distance queries in large-scale fully dynamic complex networks. In *Proceedings of the 27th International Workshop on Combinatorial Algorithms (IWOCA 2016)*, Vol. 9843 of *Lecture Notes in Computer Science*, pp. 109–121. Springer.

D'Angelo, G., D'Emidio, M., & Frigioni, D. (2019). Fully dynamic 2-hop cover labeling. *ACM J. Exp. Algorithmics*, *24*(1), 1.6:1–1.6:36.

D'Ascenzo, A., D'Emidio, M., Flammini, M., & Monaco, G. (2022). Digraph k-coloring games: From theory to practice. In *Proceedings of the 20th International Symposium on Experimental Algorithms (SEA 2022)*, Vol. 233 of *LIPIcs*, pp. 20:1–20:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Deligkas, A., Fearnley, J., & Savani, R. (2020). Tree polymatrix games are ppad-hard. In *Proceedings of the 47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, Vol. 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 38:1–38:14. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

Deligkas, A., Fearnley, J., Savani, R., & Spirakis, P. (2017). Computing approximate nash equilibria in polymatrix games. *Algorithmica*, *77*(2), 487–514.

D'Emidio, M. (2020). Faster algorithms for mining shortest-path distances from massive time-evolving graphs. *Algorithms*, *13*(8), 191.

Eaves, B. C. (1973). Polymatrix games with joint constraints. *SIAM Journal on Applied Mathematics*, *24*(3), 418–423.

Erdos, P., & Lovász, L. (1975). Problems and results on 3-chromatic hypergraphs and some related questions. *Infinite and finite sets*, *10*(2), 609–627.

Fabrikant, A., Papadimitriou, C. H., & Talwar, K. (2004). The complexity of pure nash equilibria. In Babai, L. (Ed.), *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC 2004)*, pp. 604–612. ACM.

Feldman, M., & Friedler, O. (2015). A unified framework for strong price of anarchy in clustering games. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP 2015)*, Vol. 9135 of *Lecture Notes in Computer Science*, pp. 601–613. Springer.

Feldman, M., Lewin-Eytan, L., & Naor, J. (2015). Hedonic clustering games. *ACM Trans. Parallel Comput.*, *2*(1), 4:1–4:48.

Gairing, M., & Savani, R. (2011). Computing stable outcomes in hedonic games with voting-based deviations. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Vol. 1-3, pp. 559–566. IFAAMAS.

Gourvès, L., & Monnot, J. (2010). The max k-cut game and its strong equilibria. In *Proceedings of the 7th Annual Conference on Theory and Applications of Models of Computation (TAMC 2010)*, Vol. 6108 of *Lecture Notes in Computer Science*, pp. 234–246. Springer.

Hoefer, M. (2007). *Cost sharing and clustering under distributed competition*. Ph.D. thesis, University of Konstanz.

Howson, J. T. (1972). Equilibria of polymatrix games. *Management Science*, *18*(5), 312–318.

Howson, J. T., & Rosenthal, R. W. (1974). Bayesian equilibria of finite two-person games with incomplete information. *Management Science*, *21*(3), 313–315.

Kearns, M. J., Littman, M. L., & Singh, S. (2001). Graphical models for game theory. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI 2001)*, pp. 253–260. Morgan Kaufmann.

Kun, J., Powers, B., & Reyzin, L. (2013). Anti-coordination games and stable graph colorings. In *Proceedings of the 6th International Symposium on Algorithmic Game Theory (SAGT 2013)*, Vol. 8146 of *Lecture Notes in Computer Science*, pp. 122–133. Springer.

Leskovec, J., & Krevl, A. (2014). SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`.

McGeoch, C. C. (2012). *A Guide to Experimental Algorithmics*. Cambridge University Press.

Miller, D. A., & Zucker, S. W. (1991). Copositive-plus lemke algorithm solves polymatrix games. *Operations Research Letters*, *10*(5), 285–290.

Monaco, G., Moscardelli, L., & Velaj, Y. (2019). On the performance of stable outcomes in modified fractional hedonic games with egalitarian social welfare. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2019)*, pp. 873–881. International Foundation for Autonomous Agents and Multiagent Systems.

Monaco, G., Moscardelli, L., & Velaj, Y. (2020). Stable outcomes in modified fractional hedonic games. *Auton. Agents Multi Agent Syst.*, *34*(1), 4.

Monderer, D., & Shapley, L. S. (1996). Potential games. *Games and Economic Behavior*, *14*(1), 124 – 143.

Moser, R. A., & Tardos, G. (2010). A constructive proof of the general Lovász local lemma. *J. ACM*, *57*(2), 11:1–11:15.

Panagopoulou, P. N., & Spirakis, P. G. (2008). A game theoretic approach for efficient graph coloring. In *Proceedings of the 19th International Symposium on Algorithms and Computation (ISAAC)*, Vol. 5369 of *Lecture Notes in Computer Science*, pp. 183–195. Springer.

Peixoto, T. P. (2020). The netzschleuder network catalogue and repository. `https://networks.skewed.de`.

Peters, D. (2016). Graphical hedonic games of bounded treewidth. In *Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016)*, pp. 586–593. AAAI Press.

Peters, D., & Elkind, E. (2015). Simple causes of complexity in hedonic games. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*, pp. 617–623. AAAI Press.

Poljak, S. (1995). Integer linear programs and local search for max-cut. *SIAM J. Comput.*, *24*(4), 822–839.

Rahn, M., & Schäfer, G. (2015). Efficient equilibria in polymatrix coordination games. In *Proceedings of the 40th International Symposium on Mathematical Foundations of Computer Science (MFCS 2015)*, Vol. 9235 of *Lecture Notes in Computer Science*, pp. 529–541. Springer.

Rossi, R. A., & Ahmed, N. K. (2015). The network data repository with interactive graph analytics and visualization. In *Proceedings of the 29th Conference on Artificial Intelligence (AAAI 2015)*, pp. 4292–4293. AAAI Press.

Roughgarden, T. (2010). Algorithmic game theory. *Communications of the ACM*, *53*(7), 78–86.

Schäffer, A. A., & Yannakakis, M. (1991). Simple local search problems that are hard to solve. *SIAM J. Comput.*, *20*(1), 56–87.

Simon, S., & Wojtczak, D. (2016). Efficient local search in coordination games on graphs. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 482–488. IJCAI/AAAI Press.

Spencer, J. (1977). Asymptotic lower bounds for Ramsey functions. *Discrete Mathematics*, *20*, 69–76.

Staudt, C. L., Sazonovs, A., & Meyerhenke, H. (2016). Networkit: A tool suite for large-scale complex network analysis. *Netw. Sci.*, *4*(4), 508–530.

Swenson, B. (2017). *Myopic Best-Response Learning in Large-Scale Games*. Ph.D. thesis, Carnegie Mellon University, USA.

Swenson, B., Murray, R. W., & Kar, S. (2018). On best-response dynamics in potential games. *SIAM J. Control. Optim.*, *56*(4), 2734–2767.

Vickrey, D., & Koller, D. (2002). Multi-agent algorithms for solving graphical games. In *Proceedings of the 18th National Conference on Artificial Intelligence and 14th Conference on Innovative Applications of Artificial Intelligence*, pp. 345–351. AAAI Press / MIT Press.

Yanovskaya, E. B. (1968). Equilibrium points in polymatrix games. *Lithuanian Mathematical Journal*, *8*(2), 381–384.