Digraph k-Coloring Games: From Theory to Practice

Andrea D'Ascenzo
Department of Computer Science, Information Engineering and Mathematics, University of L'Aquila, Italy

Mattia D'Emidio
Department of Computer Science, Information Engineering and Mathematics, University of L'Aquila, Italy

Michele Flammini
Gran Sasso Science Institute, L'Aquila, Italy

Gianpiero Monaco
Department of Computer Science, Information Engineering and Mathematics, University of L'Aquila, Italy

Abstract

We study digraph k-coloring games where agents are vertices of a directed unweighted graph and arcs represent agents' mutual unidirectional idiosyncrasies or conflicts. Each agent can select one of k different colors, and her payoff in a given state is given by the number of outgoing neighbors with a different color. 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 is NP-complete [33]. Therefore, in the last few years a relevant research focus has been that of designing polynomial time algorithms able to compute *approximate* Nash equilibria, i.e., states in which no agent, changing her strategy, can improve her payoff by some bounded multiplicative factor. The only two known algorithms in this respect are those in [14]. While they provide theoretical guarantees, their practical performance over real-world instances so far has not been investigated. In this paper, under the further motivation of the lack of practical approximation algorithms for the problem, we experimentally evaluate the above algorithms with the conclusion that, while they were suitably designed for achieving a bounded worst case behavior, they generally have a poor performance. Therefore, we next focus on classical best-response dynamics, and show that, despite of the fact that they might not always converge, they are very effective in practice. In particular, we provide a strong empirical evidence that they outperform existing methods, since surprisingly they quickly converge to exact Nash equilibria in almost all instances arising in practice. This also shows that, while this class of games is known to not always possess pure Nash equilibria, 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.

2012 ACM Subject Classification Theory of computation → Algorithmic game theory and mechanism design; Theory of computation → Quality of equilibria; Theory of computation → Design and analysis of algorithms; Theory of computation → Graph algorithms analysis

Keywords and phrases Algorithmic Game Theory, Coloring Games, Experimental Algorithmics, Exact vs Approximate Nash Equilibria, Decentralized Dynamics

Digital Object Identifier 10.4230/LIPIcs.SEA.2022.20

Supplementary Material *Software (Source Code)*: `https://github.com/buildfreak/sea_2022` archived at `swh:1:dir:f09c580538ebfe5b245c15fb6a24810fbb817fa2`

1 Introduction

In this paper we consider digraph k-coloring games. We are given an unweighted directed graph where vertices represent selfish autonomous agents and arcs mutual unidirectional idiosyncrasies or conflicts. Moreover, we have a set of $k \geq 2$ available colors denoting agents'

available choices or strategies. A state of the game is a vertex coloring, induced by the choices of the agents. The objective of each agent is that of maximizing her own payoff, which is defined as the number of outgoing neighbors with a color different from hers.

Digraph k-coloring games form some of the basic payoff structures in algorithmic game theory, they fall within the fundamental classes of (anti-)coordination games (see the Related Work section) and model many relevant real-world scenarios. Typical examples are wireless networks in which radio stations wish to select the transmission frequency not used by the maximum number of neighboring stations within their range in order to limit interferences, or social networks in which members must be split in groups and want to maximize the number of enemies they do not end together with, or markets in which sellers aim to locate their activities as far as possible from their direct competitors.

A classical solution concept of stable outcomes in scenarios with selfish and autonomous agents is the *(pure) Nash equilibrium* (NE, for short) 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 can improve her payoff by changing color. The NE is one of the most important concepts in game theory. As such, its efficient computation is one of the most important problems in algorithmic game theory [24]. In the specific setting when the input graph is undirected, the k-coloring game is a potential game [39], which implies that a NE always exists. In particular, when the graph is unweighted undirected, the dynamics (where at each step one agent performs an improving move) always converges to a NE in a polynomial number of steps [29, 33]. However, in the more general case 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 [33]. Therefore, like in a variety of games falling in this class, where Nash equilibria do not exist or cannot be computed in polynomial time, researchers focused on the milder form of *approximate* equilibria [14]. Namely, a state is called a γ-Nash equilibrium (γ-NE, for short), for some $\gamma \geq 1$, if no agent can strictly improve her payoff by a multiplicative factor of γ by changing her strategy.

To the best of our knowledge, the only advancements in this direction about digraph k-coloring games are those presented in [14]. In details, the authors: (i) show that a pure NE (i.e., with $\gamma = 1$) is not guaranteed to exist for any number of players n and colors $k < n$; (ii) observe that, for any $k \geq 2$, a pure NE exists and can be found in polynomial time if the graph is bipartite or directed acyclic; (iii) notice that, for the case of $k = 2$, a γ-Nash equilibria might not exist for any bounded value of γ. In fact, it is easy to see that in any 2-coloring of a directed cycle with an odd number of vertices there is always at least one vertex with zero utility (and hence no NE can have bounded γ); (iv) present a deterministic polynomial time algorithm (called AP1) that, for any $k \geq 3$, returns a k-coloring that is a Δ_o-Nash equilibrium, where Δ_o is the maximum outgoing degree of the input digraph G; (v) present a randomized algorithm (denoted as LLL-SPE in what follows) that, by exploiting the constructive version of the well-known Lovász Local Lemma (LLL, for short, from here onwards) [40], computes a constant approximate Nash equilibrium in polynomial expected running time. Specifically, the algorithm works for any constant $k \geq 2$ and for the special class of n-vertex digraphs having $\Omega(\log n)$ minimum outgoing degree.

However, notwithstanding the aforementioned progress, very little is known about the true performance of existing algorithms in practice. Specifically, it remains unclear which method performs best, in terms of approximation and running time, for inputs arising from real-world application domains. On the one hand, in fact, no average case analysis is known for AP1 and hence it is difficult to capture how much the upper bound on the approximation is pessimistic

on real inputs. On the other hand, algorithm LLL-SPE guarantees constant approximation with a value of γ that is quite large, and only for a class of digraphs that appears to be quite infrequent in the application domains of the considered game. Nothing is known about the behavior of LLL-SPE on general digraphs. Moreover, to the best our knowledge, no experimental evaluations have been conducted on the subject, nor any algorithm has been shown to perform nicely in practice.

Our Contribution. In this paper we attack such research questions through algorithmic experimentation. In particular, our contribution is twofold. First, we implement and test both AP1 and LLL-SPE[1] against both artificial and real-world graphs of heterogeneous sizes and topologies, and different values of k. We observe that the measured approximation achieved by the two solutions, on the considered combinations of inputs and values of k, is unsatisfactory. Specifically, the obtained colorings are comparable, in terms of measured γ, to those one can produce by simply assigning colors to agents/vertices uniformly at random. Then, motivated by such unsatisfactory performance, under the further motivation of the lack of practical approximation algorithms for the problem, we focus on the classical best-response paradigm, where at each step an agent having an improving move is selected uniformly at random (we refer to this approach as algorithm BEST-RESP in the following). Such paradigm induces a dynamics on the coloring that does not offer any upper bound on the provided approximation and might not always stabilize. Hence, to evaluate the practical effectiveness of BEST-RESP, we implement it and compare it against AP1 and LLL-SPE, through an extensive experimental evaluation again involving large sets of heterogeneous inputs and values of k. In our study we consider, as main performance indicators, both approximation (namely measured γ) and computational time. Plus, we assess other metrics we introduce here, namely *average payoff* and *fraction of unhappy vertices*, which naturally characterize the practical effectiveness of considered algorithms in the application domains where they are intended to be applied. Our experimental results provide strong empirical evidences of 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 $k \geq 3$, BEST-RESP results to be the best performing method in terms of approximation. Moreover, remarkably, in the majority of the cases it converges to a pure NE (i.e. computes exact, non-approximate solutions) in a reasonably low running time, even for large graphs. In the (few) remaining cases, 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 results trigger new theoretical questions and demand for new investigation on the possibility of achieving constant approximation for general digraphs or computing a pure NE in polynomial time for special classes of digraphs. Our study also 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 BEST-RESP (while AP1 and LLL-SPE are not naturally suited for distributed implementations).

Related Work. k-coloring games in undirected graphs have been first investigated in [29,33], 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, i.e. conjectured difficult, even for $k = 2$ [46] (notice

[1] More precisely, a slightly modified version of LLL-SPE that incorporates a stopping criterion to apply it on general digraphs, where the convergence is not guaranteed since the LLL might not hold.

that graph 2-coloring games are exactly max cut games). In [43] 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 [5, 12], where the authors give an algorithm that, for any $\epsilon > 0$, computes a $(3+\epsilon)$-equilibrium 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 Nash equilibrium is NP-complete for any fixed $k \geq 2$, even in the unweighted case [33]. The performance of NE in general graph k-coloring games has been addressed in [25], while the authors of [15] consider Nash equilibria where players also have an extra profit depending on the chosen color. Finally, the authors of [13, 28] 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 [32], where the payoff of each agent depends only on the strategies of her neighbours in a given social knowledge graph defined over the set of the agents. An interesting class of graphical games, related to digraph k-coloring games, is that of graphical congestion games [7]. Digraph k-coloring games can also be seen as a particular form of hedonic games with an upper bound (i.e., k) to the number of coalitions (see [4] for a brief 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 NE in hedonic games have been largely investigated under several assumptions (see [6, 26, 27, 37, 38, 42] and references therein).

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 [2, 3, 44]. Finally, the authors of [41] 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* [51]. 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 [23, 30, 31, 36] and also, more recently, with a special focus on equilibria [11, 20, 21, 44].

2 Notation and Background

Graph Notation. 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 is said to be an *outgoing arc* from vertex v (*incoming arc* to vertex w, respectively) if such arc is any arc $(v, w) \in A$.

Given a vertex $v \in V$, we denote by δ_o^v (δ_i^v, respectively) the *outgoing degree* of v (the *incoming degree* of v, respectively), that is the number of outgoing arcs from v (the number of incoming arcs to v, respectively) in G. The set of *outgoing neighbors* $N_{out}(v)$ of a vertex v is the set of vertices induced by all outgoing arcs of v, i.e. $N_{out}(v) = \{w : w \in V \wedge (v, w) \in A\}$.

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 outgoing degree of G, respectively. Similarly, we call $\overline{d_o}$ and $\overline{\overline{d_o}}$ the average and median outgoing degree, respectively, and $\Delta_i = \max_{v \in V} \delta_i^v$ the maximum incoming degree of G.

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 each c_v is the color chosen by each 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 whose color in c is different from that of v, i.e. $\mu_c(v) = |\{(v, w) \in A : c_v \neq c_w\}|$. Observe that, for a vertex v, having $N_{out}(v) = \emptyset$ or $c_v = c_w \ \forall w \in N_{out}(v)$ implies $\mu_c(v) = 0$.

A coloring c is a *pure Nash equilibrium* (a.k.a. *stable equilibrium*, denoted in what follows as pure NE for short), if no agent v 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 $v \in V$.

Unfortunately, a pure NE is not guaranteed to exist even for a large number of available colors k. In particular, while it is easy to observe that there always exists a NE with $|C| = n$ colors, by just assigning to each vertex a different color, it is also possible to prove that, for arbitrary values of $n \geq 3$, and for any fixed k such that $1 < k \leq n-1$, there exist instances of the digraph k-coloring game that do not admit any pure NE [14]. Furthermore, it is known that, for general digraphs, the problem of determining whether the digraph k-coloring game admits a pure NE is NP-complete, for all $k \geq 2$. Moreover, the cases of bipartite graphs and directed acyclic graphs are simpler and can be handled in polynomial time [14].

Therefore, from here onward we consider the 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, we have that 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:

$$\gamma \cdot \mu_c(v) \geq \mu_{(c_{-v}, c'_v)}(v).$$

Each vertex in a γ-NE is said to be *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. In this case, we define the *potential payoff* $\pi_c(v)$ of vertex v as the maximum payoff a vertex v can achieve by unilaterally changing its color to another color of the strategy, 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. $\psi_c(v) = argmax_{c'_v \in C} \mu_{(c_{-v}, c'_v)}(v)$.

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 $\pi_c(v) = \gamma \mu_c(v)$, for all vertices $v \in V$ for some $\gamma > 0$, we have a γ-NE.

3 Algorithms for Digraph k-coloring Games

In this section, we first summarize the main characteristics of known algorithms for the computation of approximate γ-NE with guarantees on the achieved γ, namely AP1 and LLL-SPE [14]. Note that we equip the latter with a stopping criterion to apply it to general digraphs. Then, we present a formal description of algorithm BEST-RESP, an iterative best-response-based approach that computes a k-coloring without any guarantee on the achieved approximation. We will show in the experimental section how this simple strategy results to be the best solution in practice.

Algorithm AP1. In this paragraph, we provide a brief description of algorithm AP1, proposed in [14]. Given a digraph G, algorithm AP1, is deterministic and, for any $k \geq 3$, returns a k-coloring such that every vertex v with $\delta_o^v \geq 1$ has payoff at least 1, in polynomial time. Clearly this corresponds to a Δ_o-NE since Δ_o is the maximum payoff any agent can achieve.

The algorithm is iterative and works as follows (see [14] for more details and for the pseudocode of the algorithm): at each iteration the algorithm visits the graph induced by the uncolored vertices and detects a cycle or a path (the latter happens only when the visit reaches a vertex with zero outgoing neighbors in the induced subgraph). Then, it colors the vertices of the cycle or the path by alternating three colors, say colors 1, 2 and 3, in a way that every vertex gets payoff of at least 1, as follows. If the induced subgraph is a cycle, then the algorithm considers vertices of the cycle in clockwise order and assigns the colors by following such order, starting by any vertex and alternating the three colors. Viceversa, if the subgraph is a path from vertex v to vertex w, then two cases can occur. If the arc $(w, u) \in A$ then it colors w by a different color with respect to the already colored vertex u. Otherwise, it means that $\delta_o^w = 0$ and we can assign any color to w. Then, it alternates colors (in this case two colors are enough) for the other vertices of the path considered in the reverse order starting from w. Observe that it is easy to show, by analyzing the pseudocode of AP1, given in [14], that the following holds.

▶ **Lemma 1.** *Algorithm AP1 runs in $O(\Delta_o nm)$ worst case time.*

Algorithm LLL-GEN. In this paragraph, we introduce algorithm LLL-GEN, a generalization of the randomized algorithm LLL-SPE presented in [14]. Observe that algorithm LLL-SPE is based on the *Lovász Local Lemma* (LLL, for short) [47], which can be used for proving that there is a positive probability that a random assignment of the k colors to the vertices of digraphs returns a constant approximate NE, for any value of $k \geq 2$, for any graph $G = (V, A)$ such that the outgoing degree of any $v \in V$ is $\delta_o^v = \Omega(\log \Delta_o + \log \Delta_i)$. More precisely, algorithm LLL-SPE is designed to compute such a NE, in polynomial expected running time, by exploiting the constructive version of the LLL provided by [40]. Specifically, the algorithm, starting from a random assignment of the colors, repeatedly and iteratively applies operations of random *resampling* of the colors of γ-unhappy vertices of the graph, and to vertices in their dependency sets, until the coloring converges to a γ-NE. The dependency set of a vertex v is the set of vertices whose status of being unhappy/happy is influenced by (influences, resp.) the status of v, namely v's neighbours, their incoming neighbors, and v itself (see [14]).

The convergence is guaranteed to occur, for LLL-SPE, w.h.p., if the LLL is satisfied for a given graph, i.e. when $\delta_o^v = \Omega(\log \Delta_o + \log \Delta_i)$. In details, in order to apply the LLL: (i) a "bad event" I_v is defined over each vertex v of the graph when v is not γ-happy; (ii) a bound to the maximum size of the dependency set, denoted as dep_v, of each bad event I_v is given

as $|dep_v| \leq \delta_o^v + \delta_i^v + \delta_o^v \delta_i^v$. If the LLL is satisfied, then algorithm LLL-SPE returns a γ-NE by performing a number of resampling operations of dependency sets of γ-unhappy vertices that is polynomial in expectation, for a constant value of γ, defined as follows:

$$\gamma = \max_{v \in V : \delta_o^v > 0} \frac{\text{max possible payoff}}{\text{min expected payoff}} \approx \max_{v \in V : \delta_o^v > 0} \frac{k}{k-1} + \left(\frac{k}{k-1}\right)^2 O\left((r - \frac{k}{k-1})^{-1}\right) \quad (1)$$

where $r = \delta_o^v / \log(\Delta_o \Delta_i)$. Note that vertices with zero out-degree are not considered in this formula, indeed they are always happy because they can always select a strategy that yields a non-zero payoff [14].

Now, observe that the above guarantees hold only for graphs whose structure satisfies the LLL. Thus, in order to generalize algorithm LLL-SPE and apply it to any digraph, motivated by the lack of algorithms to compute approximate NE in general digraphs, we introduce a stopping criterion to the maximum number of resampling operations that the algorithm can perform. This is a necessary change for our purpose of experimentally evaluating the behavior of such algorithm also in digraphs such that δ_o^v is not $\Omega(\log \Delta_o + \log \Delta_i)$, for some vertex v, as in this case LLL-SPE might not converge in polynomial time in expectation. In

Algorithm 1 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 ← random k–coloring of G with uniform probability 1/k;
2 Compute threshold on γ as in Eq. 1;
3 i ← 0;
4 while ∃ a γ-unhappy vertex and i < I do
5 |   Let S be the set of γ unhappy vertices; /* c is not a γ-NE, hence S ≠ ∅ */
6 |   Select randomly a vertex v ∈ S, with uniform probability 1/|S|;
7 |   Randomly, uniformly, color vertices in dependency set dep_v of v;
8 |   i ← i + 1;
9 return c;
```

more details, besides the input digraph and the number of available colors k, the modified method LLL-GEN, summarized in Algorithm 1, takes as input also an integer value I and stops when either a γ-NE is found or a maximum number of iterations I is performed. By such modification, it is easy to see that the following holds.

▶ **Lemma 2.** *Algorithm LLL-GEN runs in $O(I(n + \Delta_o + \Delta_i + \Delta_o \Delta_i))$ worst case time.*

Proof. In each iteration, lines 5 and 7 are executed, with the former requiring $O(n)$ 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. $O(\Delta_o + \Delta_i + \Delta_o \Delta_i)$. ◀

Algorithm BEST-RESP. In this paragraph we describe an approach for computing approximated NE, named BEST-RESP, that is inspired to classical best-response dynamics, since they have been shown to be effective in practice to handle similar kinds of games [16, 49, 50]. More specifically, we consider what, in the literature, is sometimes referred to as myopic best-response method [16, 50], i.e. best-response dynamics where agents decide their strategy based on knowledge of their neighbors only. Such method is universally considered one of the most appealing strategies 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 [8, 49]. 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 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 $\pi_c(v)$. The process stops if a NE is reached, i.e. if no unhappy vertices exist in the graph, or when a maximum number of iterations I, given as part of the input, is performed. The pseudocode of procedure BEST-RESP is summarized in Algorithm 2. Given the above, the following result easily follows.

Algorithm 2 Algorithm 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 ← random k–coloring of G with uniform probability 1/k;
2 i ← 0;
3 while ∃ an unhappy vertex and i < I do
4 |   Let S be the set of unhappy vertices;        /* c is not a NE, hence S ≠ ∅ */
5 |   Select randomly a vertex v ∈ S, with uniform probability 1/|S|;
6 |   c_v ← ψ_c(v);                                  // Color that maximizes payoff
7 |   i ← i + 1;
8 return c;
```

▶ **Lemma 3.** *Algorithm BEST-RESP runs in $O(n\Delta_o I)$ worst case 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 $O(\Delta_o n)$ time since, for each vertex, we need to evaluate the colors of her outgoing neighbors. Selecting at random an unhappy vertex costs $|S|$ hence $O(n)$ time, and for said unhappy vertex an additional $\delta_o^v = O(\Delta_o)$ time in necessary to determine the color that maximizes her payoff. Thus, the claim follows. ◀

4 Experimentation

In this section, we describe the experimental study we conducted to assess the performance of algorithms for digraph k-coloring games. In particular, we implemented LLL-GEN, BEST-RESP, and AP1, and designed and deployed an experimental framework to evaluate said algorithms, with respect to various metrics of interest for the context of digraph k–coloring games, and on large set of meaningful input digraphs.

Test Environment and Implementation Details. Our entire test environment is based on NetworKit [48], 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.

Input Details. As input to our experiments, inspired by other empirical studies on graph algorithms [1,10,17–19,22], we employed a large dataset of digraphs, including: (i) real-world instances, taken from publicly available repositories [34,45]; (ii) artificial digraphs, built via well-established random generators, namely *Erdős-Rényi* and *Paley* models [9]. More details on used inputs, including sizes and main characteristics are reported in Table 1, while Figure 1 shows how outgoing degree is distributed in the considered inputs.

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 outgoing degree, respectively. Finally, the 9th column highlights whether the graph is synthetic or real-world (● = true, ○ = false), while the last column specifies whether the LLL holds in the given graph (● = true, ○ = false). Inputs are sorted by $\overline{d_o}$, non-decreasing.

Dataset	Short	Type	|V|	|A|	$\overline{d_o}$	$\widetilde{d_o}$	Δ_o	S	LLL
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	●	●
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	○	○
Erdős-Rényi A	era	random	1000	12460	12.46	12	27	●	○
Blog	blg	interaction	1224	19022	15.54	7	256	○	○
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	○	○
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	●	●

Concerning parameter k, since no direct, well-established relationship is known between k itself and the approximation provided by the algorithms under study, our experimental trials consider carefully selected values of said parameter to investigate how the algorithms' behavior changes as k increases. Specifically, we consider both the reference case of $k = 3$ (representing a conjectured threshold on the computational hardness of the problem) and, as suggested by established guidelines for experimental algorithmics [35], to magnify the dependency on k in a reasonable number of tests, values of k ranging in the interval $[\max\{4, d_o\}, \Delta_o]$, evenly spaced as multiples of $\lfloor \frac{\Delta_o - d_o}{5} \rceil$. For iterative algorithms, we fix $I = n \log n$, since this leads in all cases to practical running times, even on the largest inputs.

Objectives of Experimentation. The purpose of our experimentation is twofold. First, we want to assess how effective are state-of-the-art solutions for digraph k-coloring games in practice. To this regard, we test our implementations of AP1 and LLL-GEN against all inputs and the mentioned values of k and compare, in terms of resulting γ, computed colorings

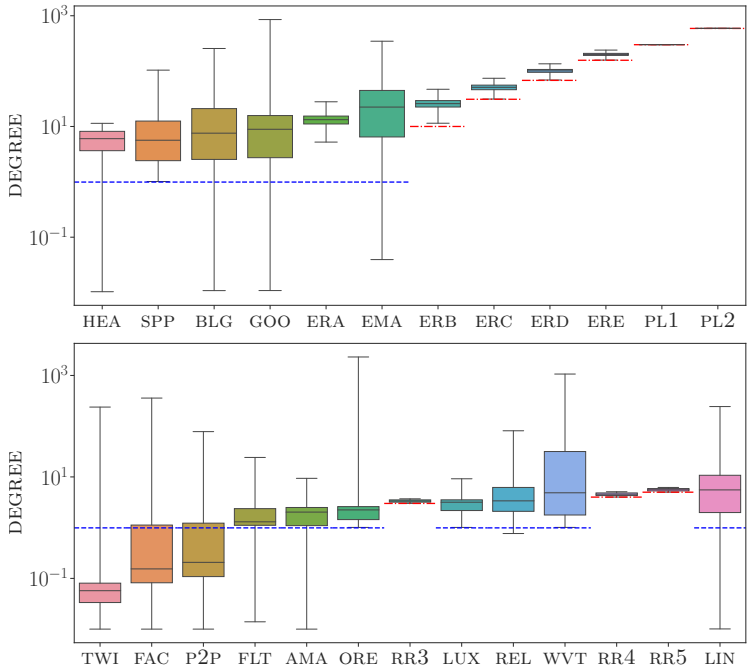

Figure 1 Distributions of outgoing vertex degrees of input graphs: each box plot shows minimum, 1st quartile, median, 3rd quartile, and maximum values. The red dotted line shows $\log \Delta_o + \log \Delta_i$ for graphs where the LLL holds. The blue line, viceversa, marks graphs where the LLL does not hold. Inputs are sorted, left to right, bottom to top, in non-decreasing order of $\overline{\overline{d_o}}$.

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). Second, we aim at establishing the practical effectiveness of algorithm BEST-RESP, hereby formalized. To this aim, we execute and compare obtained results to those achieved by AP1 and LLL-GEN for the same inputs and values of k. 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:

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

In other words, if an algorithm computes a coloring c with $\gamma(G, c) > 1$ (a sufficient condition for the latter to happen is all vertices having strictly positive payoff, cf. Sec. 1), it follows that said coloring is a γ-NE. Moreover, $\gamma(G, c) = 1$ implies the computed coloring is a pure NE. Now, since algorithms LLL-GEN and BEST-RESP are iterative and induce dynamics on the state of the game (i.e. on the coloring), this part of the study measures also further performance indicators to better characterize the behavior of the considered algorithms. More in details, 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. Notice that most of our experimental framework is written in Python, with some sub-routines in C++/Cython, with a fairly optimized code. We leave the problem open of achieving a faster version of all implementations through careful code tuning and porting to more high-performance programming languages., e.g. pure C++. In what follows, we will use LLG, BR, RND and AP1 to refer to algorithms LLL-GEN, BEST-RESP, RANDOM and AP1, respectively, for the sake of brevity.

Analysis. In Table 2 we present a summary of the results of our experimentation, for all input graphs and values of k. In details, for each considered metric, starting from the most relevant (i.e. approximation), we report the number of times each algorithm resulted to be the best performing one (2nd, 3rd, 4th best performing one, respectively) with respect to said metric. Note that, for those algorithms that resort to randomization, we executed three trials for each combination and computed average values of observed measures.

Table 2 Aggregate statistics for all tested algorithms with respect to the four considered indicators, for all combinations of inputs and values of k (for a total of 175 combinations).

metric	algorithm	best	2nd	3rd	worst
$\gamma(G, c)$	RND	4 (2.3 %)	33 (18.9 %)	81 (46.3 %)	57 (32.5 %)
	AP1	1 (0.6 %)	69 (39.4 %)	51 (29.1 %)	54 (30.9 %)
	LLG	7 (4.0 %)	62 (35.4 %)	43 (24.6 %)	63 (36.0 %)
	BR	163 (93.1 %)	11 (6.3 %)	0 (0.0 %)	1 (0.6 %)
$U(G, c)$	RND	1 (0.6 %)	42 (24.0 %)	106 (60.6 %)	26 (14.8 %)
	AP1	0 (0.0 %)	13 (7.4 %)	13 (7.4 %)	149 (85.1 %)
	LLG	6 (3.4 %)	114 (65.1 %)	55 (31.5 %)	0 (0.0 %)
	BR	168 (96.0 %)	6 (3.4 %)	1 (0.6 %)	0 (0.0 %)
$\overline{P}(G, c)$	RND	5 (2.9 %)	56 (32.0 %)	96 (54.9 %)	18 (10.2 %)
	AP1	0 (0.0 %)	7 (4.0 %)	13 (7.4 %)	155 (88.6 %)
	LLG	7 (4.0 %)	106 (60.6 %)	60 (34.3 %)	2 (1.1 %)
	BR	163 (93.2 %)	6 (3.4 %)	6 (3.4 %)	0 (0.0 %)
T (G,C)	RND	173 (98.9 %)	2 (1.1 %)	0 (0.0 %)	0 (0.0 %)
	AP1	2 (1.1 %)	152 (86.9 %)	14 (8.0 %)	7 (4.0 %)
	LLG	0 (0.0 %)	20 (11.4 %)	101 (57.7 %)	54 (30.9 %)
	BR	0 (0.0 %)	1 (0.6 %)	60 (34.3 %)	114 (65.1 %)

Our data highlight clearly that algorithm BR is the best performing one, globally, in terms of approximation. In fact, out of 175 combinations of input instances and values of k, BR computes a $\gamma(G, c)$-NE with the smallest value of $\gamma(G, c)$ in 163 of them (93.1% of the combinations). Algorithms RND, LLG and AP1, instead, behave rather badly, providing the best approximation, together, only in the remaining 6.9% of the combinations. Moreover, values of $\gamma(G, c)$ obtained by LLG and AP1 are often quite close to those of RND, as shown in Figures 2–3 (for $k = 3$) and Figure 4 (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, due to space constraints). Notice also that, in all mentioned figures, we report a default of $\gamma(G, c) = 0$ when there exist at least one vertex with zero payoff (hence γ is unbounded) while $\gamma(G, c) = 1$ and $U(G, c) = 0$ correspond to a pure NE being found. Besides BR being the best solution with respect to approximation, the most surprising outcome of our experimentation is that BR is able to compute, in almost all cases, colorings that are pure, exact NEs (see e.g. Figure 2 or 3 (middle)). This is remarkable, considering the known hardness of determining this kind of colorings in general digraphs. Specifically, BR is able to find pure Nash equilibria, often in less

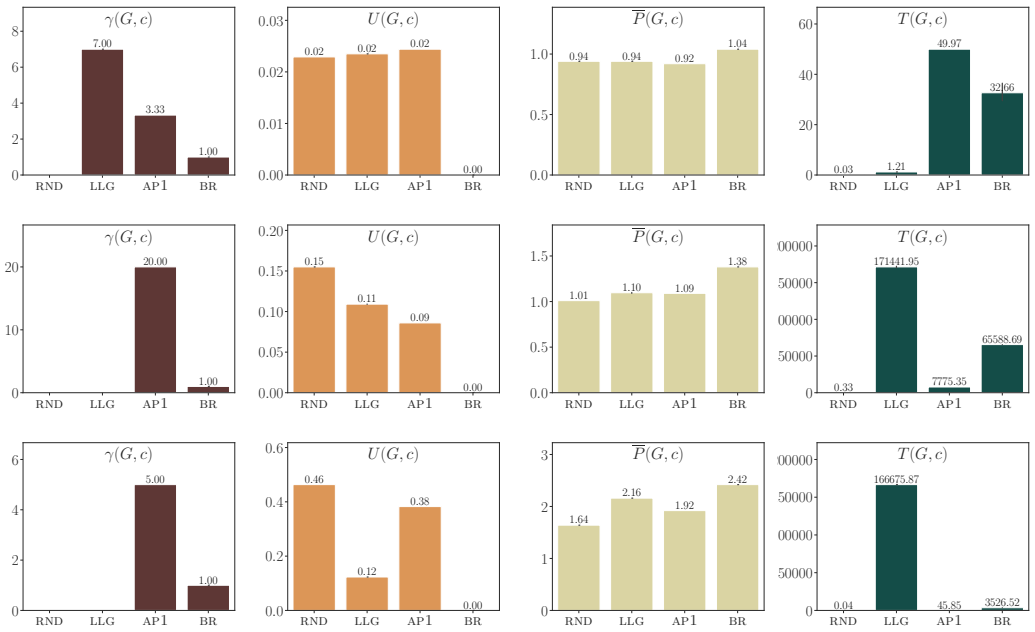

Figure 2 Performance of algorithms RND, LLG, AP1 and BR, respectively, in graphs TWI (top), FAC (middle), and LUX (bottom) with $k = 3$. Time is expressed in seconds.

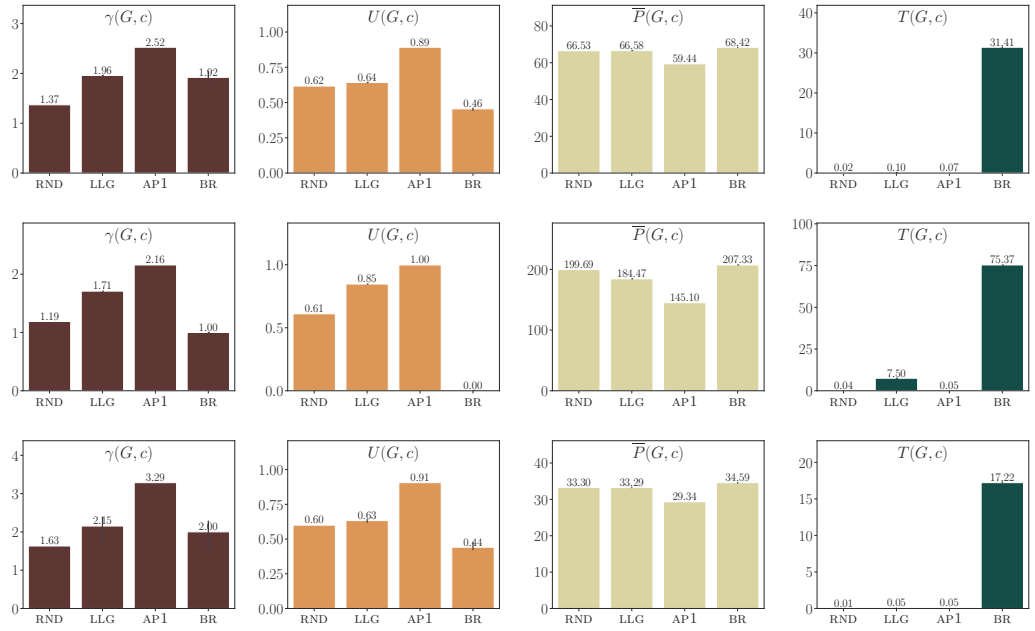

Figure 3 Performance of algorithms RND, LLG, AP1 and BR, respectively, in graphs ERD (top), PL1 (middle), and ERE (bottom), with $k = 3$. Time is expressed in seconds.

than n iterations, for all considered graphs and values of k (see, e.g., Figure 5, two left-most panels) except Erdős-Rényi instances (where however in some case still achieves the best approximation). In these latter inputs, colorings computed by BEST-RESP appears to exhibit a $\gamma(G, c)$ that becomes periodic at some point of the optimization process, around some value close to 1 (see Figure 5, two right-most panels). Note that, when BR ranks 2nd best in terms

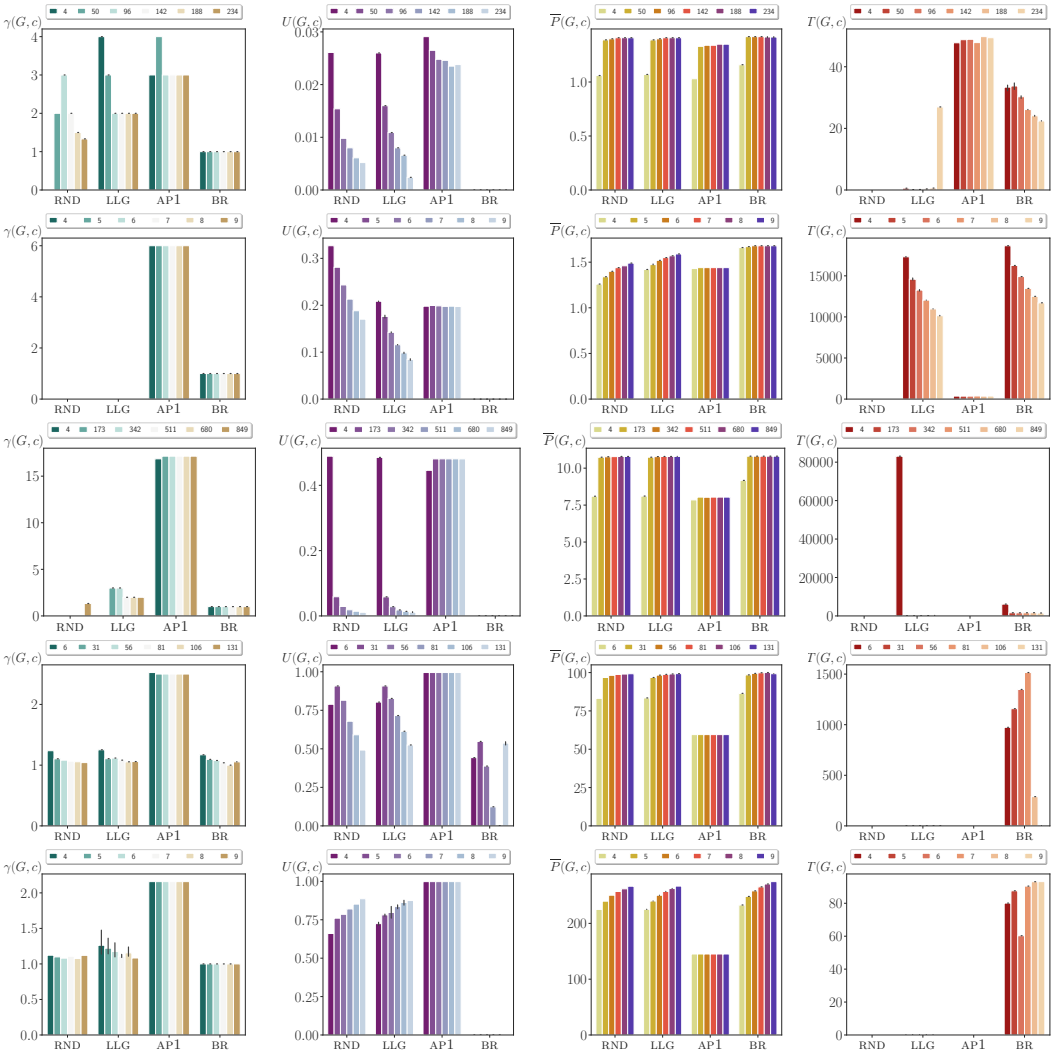

Figure 4 Performance of algorithms RND, LLG, AP1 and BR, respectively, in graphs TWI (top-most), AMA (middle-top), GOO (middle), ERD (middle-bottom) and PL1 (bottom-most), with increasing values of k. Time is expressed in seconds.

of $\gamma(G, c)$, algorithm RND results to be the best performing one (see, e.g., Figure 3, top or bottom). At the same time, BR exhibits higher average payoff and lower fraction of unhappy vertices, which suggests that random assignment might be "lucky" in picking and making happy high-degree vertices, while algorithm BR 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 BR 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 Eq. 1 with $k = 3$). In this respect, to achieve better results in practice, one might

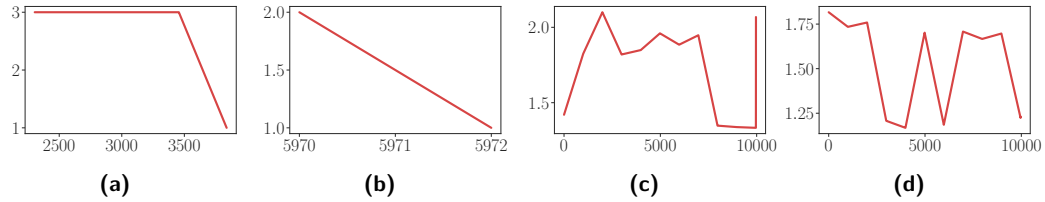

Figure 5 Results achieved by the execution of algorithm BR, in terms of $\gamma(G, c)$, on graphs BLG (a), WVT (b), ERD (c) and ERE (d) with $k = 3$, respectively. The y–axis shows the measured value of $\gamma(G, c)$ as a function of the number of iterations performed by the algorithm, reported on the x-axis (we omit iterations where $\gamma(G, c)$ is unbounded due e.g. to some vertex having zero payoff). In the two left-most panels, i.e. (a)–(b), we can observe $\gamma(G, c)$ the algorithm quickly converging to a pure NE ($\gamma(G, c)$ approaches and then stabilizes at 1 (in much less than $n \log n$ and n iterations, respectively). In the two right-most panels, , i.e. (c)–(d), instead, $n \log n$ iterations do not suffice to achieve convergence at pure NE and $\gamma(G, c)$ seems to start oscillating around a value of 1.75 and 2, respectively.

think of removing such stopping criterion from LLL-GEN to let the coloring being updated, via resampling, for a maximum number of iterations, as done by BEST-RESP. Nonetheless, as further experimentation shows (omitted due to space limitations), this does not lead to meaningful improvements, in terms of both approximation and other indicators. 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. Nonetheless, for the sake of completeness, it is worth noticing that also the latter might not be enough. In fact, our study points out that inputs arising in real-world applications satisfying the LLL are essentially non-existent (see Figure 1).

Regarding the impact of varying k on the performance of the considered algorithms, we observe that, while BR remains the best performing in essentially all cases, approximation ratio and fraction of unhappy vertices (running time and average payoff, respectively) tend 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 hence necessary also here to find out whether there exists some relationship between k, Δ_o, and the quality of the equilibrium that can be achieved. As a final 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 BR is the most time consuming solution (always achieving the best approximation in these cases); (iii) in the remaining cases, either AP1 and LLG yield the largest $T(G, c)$ but they do not achieve the best approximation. The latter represents another evidence of BR being fast at converging to a pure NE, when it exists. To summarize, our experimental study identifies algorithm BR 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 BR is strongly advised to find good Nash equilibria in application domains where the considered game arise.

5 Conclusion and Future Work

Our experimental study adds considerable insights on digraph k-coloring games w.r.t. previous theoretical work. In fact, it provides a strong empirical evidence of the fact that theoretical results probably required very rare graphs, tailored around the worst case behaviour, and that best response heuristics outperforms algorithms with guarantees. This motivates further research effort towards proving the existence of a NE in specific classes of graphs, especially the ones typically arising from social phenomena. Moreover, our results suggest that, in general, a NE with a better approximation factor could be found. In fact, for the very few cases in which a NE is not reached by BR, the returned solution is always a γ-NE with a very low approximation ratio. This renews theoretical interest on this relevant class of games. 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.

References

1 Eugenio Angriman, Alexander van der Grinten, Moritz von Looz, Henning Meyerhenke, Martin Nöllenburg, Maria Predari, and Charilaos Tzovas. Guidelines for experimental algorithmics: A case study in network analysis. *Algorithms*, 12(7):127, 2019. doi:10.3390/a12070127.

2 Elliot Anshelevich and Shreyas Sekar. Approximate equilibrium and incentivizing social coordination. In *Proceedings of the 28th Conference on Artificial Intelligence (AAAI 2014)*, pages 508–514, 2014.

3 Krzysztof R. Apt, Bart de Keijzer, Mona Rahn, Guido Schäfer, and Sunil Simon. Coordination games on graphs. *Int. J. Game Theory*, 46(3):851–877, 2017. doi:10.1007/s00182-016-0560-8.

4 Haris Aziz and Rahul Savani. Hedonic games. In Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia, editors, *Handbook of Computational Social Choice*, pages 356–376. Cambridge University Press, 2016. doi:10.1017/CBO9781107446984.016.

5 Anand Bhalgat, Tanmoy Chakraborty, and Sanjeev Khanna. Approximating pure nash equilibrium in cut, party affiliation, and satisfiability games. In *Proceedings of the 11th ACM Conference on Electronic Commerce (EC 2010)*, pages 73–82, 2010.

6 Vittorio Bilò, Angelo Fanelli, Michele Flammini, Gianpiero Monaco, and Luca Moscardelli. Nash stable outcomes in fractional hedonic games: Existence, efficiency and computation. *J. Artif. Intell. Res.*, 62:315–371, 2018. doi:10.1613/jair.1.11211.

7 Vittorio Bilò, Angelo Fanelli, Michele Flammini, and Luca Moscardelli. Graphical congestion games. *Algorithmica*, 61(2):274–297, 2011.

8 Lawrence E. Blume. The statistical mechanics of best-response strategy revision. *Games and Economic Behavior*, 11(2):111–145, 1995. doi:10.1006/game.1995.1046.

9 Béla Bollobas. *Random Graphs*. Cambridge University Press, 2001.

10 Michele Borassi and Emanuele Natale. Kadabra is an adaptive algorithm for betweenness via random approximation. *ACM J. Exp. Algorithmics*, 24, February 2019. doi:10.1145/3284359.

11 Yang Cai, Ozan Candogan, Constantinos Daskalakis, and Christos H. Papadimitriou. Zero-sum polymatrix games: A generalization of minmax. *MOR*, 41(2):648–655, 2016.

12 Ioannis Caragiannis, Angelo Fanelli, and Nick Gravin. Short sequences of improvement moves lead to approximate equilibria in constraint satisfaction games. In *Proceedings of the 7th International Symposium on Algorithmic Game Theory (SAGT 2014)*, pages 49–60, 2014.

13 Raffaello Carosi, Simone Fioravanti, Luciano Gualà, and Gianpiero Monaco. 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)*, volume 11376 of *Lecture Notes in Computer Science*, pages 94–107. Springer, 2019. doi:10.1007/978-3-030-10801-4_9.

14 Raffaello Carosi, Michele Flammini, and Gianpiero Monaco. Computing approximate pure nash equilibria in digraph k-coloring games. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2017)*, pages 911–919. ACM, 2017.

15 Raffaello Carosi and Gianpiero Monaco. Generalized graph k-coloring games. *Theory Comput. Syst.*, 64(6):1028–1041, 2020. doi:10.1007/s00224-019-09961-9.

16 Matthew Cary, Aparna Das, Benjamin Edelman, Ioannis Giotis, Kurtis Heimerl, Anna R. Karlin, Scott Duke Kominers, Claire Mathieu, and Michael Schwarz. Convergence of position auctions under myopic best-response dynamics. *ACM Trans. Econ. Comput.*, 2(3), 2014.

17 Annalisa D'Andrea, Mattia D'Emidio, Daniele Frigioni, Stefano Leucci, and Guido Proietti. Experimental evaluation of dynamic shortest path tree algorithms on homogeneous batches. In Joachim Gudmundsson and Jyrki Katajainen, editors, *Proceedings of the 13th International Symposium on Experimental Algorithms (SEA 2014)*, volume 8504 of *Lecture Notes in Computer Science*, pages 283–294. Springer, 2014. doi:10.1007/978-3-319-07959-2_24.

18 Gianlorenzo D'Angelo, Mattia D'Emidio, and Daniele Frigioni. Distance queries in large-scale fully dynamic complex networks. In Veli Mäkinen, Simon J. Puglisi, and Leena Salmela, editors, *Proceedings of the 27th International Workshop on Combinatorial Algorithms (IWOCA 2016)*, volume 9843 of *Lecture Notes in Computer Science*, pages 109–121. Springer, 2016. doi:10.1007/978-3-319-44543-4_9.

19 Gianlorenzo D'Angelo, Mattia D'Emidio, and Daniele Frigioni. Fully dynamic 2-hop cover labeling. *J. Exp. Algorithmics*, 24(1):1.6:1–1.6:36, 2019. doi:10.1145/3299901.

20 Argyrios Deligkas, John Fearnley, and Rahul Savani. Tree polymatrix games are ppad-hard. In *Proceedings of the 47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:14. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ICALP.2020.38.

21 Argyrios Deligkas, John Fearnley, Rahul Savani, and Paul Spirakis. Computing approximate nash equilibria in polymatrix games. *Algorithmica*, 77(2):487–514, 2017. doi:10.1007/s00453-015-0078-7.

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

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

24 Alex Fabrikant, Christos Papadimitriou, and Kunal Talwar. The complexity of pure nash equilibria. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC 2004)*, pages 604–612. ACM, 2004.

25 Michal Feldman and Ophir Friedler. 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)*, volume 9135 of *Lecture Notes in Computer Science*, pages 601–613. Springer, 2015.

26 Moran Feldman, Liane Lewin-Eytan, and Joseph (Seffi) Naor. Hedonic clustering games. *ACM Trans. Parallel Comput.*, 2(1), 2015. doi:10.1145/2742345.

27 Martin Gairing and Rahul Savani. 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)*, pages 559–566, 2011.

28 Laurent Gourvès and Jérôme Monnot. 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)*, volume 6108 of *Lecture Notes in Computer Science*, pages 234–246. Springer, 2010. doi:10.1007/978-3-642-13562-0_22.

29 Martin Hoefer. *Cost sharing and clustering under distributed competition.* PhD thesis, University of Konstanz, 2007.

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

31 Joseph T. Howson and Robert W. Rosenthal. Bayesian equilibria of finite two-person games with incomplete information. *Management Science*, 21(3):313–315, 1974.

32 Michael J. Kearns, Michael L. Littman, and Satinder P. Singh. Graphical models for game theory. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI 2001)*, pages 253–260, 2001.

33 Jeremy Kun, Brian Powers, and Lev Reyzin. Anti-coordination games and stable graph colorings. In *Proceedings of the 6th International Symposium on Algorithmic Game Theory (SAGT 2013)*, pages 122–133, 2013.

34 Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, 2014.

35 Catherine C. McGeoch. *A Guide to Experimental Algorithmics.* Cambridge University Press, 2012.

36 Douglas A. Miller and Steven W. Zucker. Copositive-plus lemke algorithm solves polymatrix games. *Operations Research Letters*, 10(5):285–290, 1991. `doi:10.1016/0167-6377(91)90015-H`.

37 Gianpiero Monaco, Luca Moscardelli, and Yllka Velaj. 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)*, pages 873–881. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

38 Gianpiero Monaco, Luca Moscardelli, and Yllka Velaj. Stable outcomes in modified fractional hedonic games. *Auton. Agents Multi Agent Syst.*, 34(1):4, 2020. `doi:10.1007/s10458-019-09431-z`.

39 Dov Monderer and Lloyd S. Shapley. Potential games. *Games and Economic Behavior*, 14(1):124–143, 1996.

40 Robin A. Moser and Gábor Tardos. A constructive proof of the general lovász local lemma. *J. ACM*, 57(2), 2010.

41 Panagiota N. Panagopoulou and Paul G. Spirakis. A game theoretic approach for efficient graph coloring. In *Proceedings of the 19th International Symposium on Algorithms and Computation (ISAAC 2008)*, pages 183–195, 2008.

42 Dominik Peters and Edith Elkind. Simple causes of complexity in hedonic games. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*, pages 617–623, 2015.

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

44 Mona Rahn and Guido Schäfer. Efficient equilibria in polymatrix coordination games. In *Proceedings of 40th International Symposium on Mathematical Foundations of Computer Science (MFCS 2015)*, pages 529–541, 2015.

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

46 Alejandro A. Schäffer and Mihalis Yannakakis. Simple local search problems that are hard to solve. *SIAM J. Comput.*, 20(1):56–87, 1991.

47 Joel Spencer. Asymptotic lower bounds for ramsey functions. *Discrete Mathematics*, 20:69–76, 1977. `doi:10.1016/0012-365X(77)90044-9`.

48 Christian L. Staudt, Aleksejs Sazonovs, and Henning Meyerhenke. Networkit: A tool suite for large-scale complex network analysis. *Network Science*, 4(4):508–530, 2016. `doi:10.1017/nws.2016.20`.

49 Brian Swenson, Ryan Murray, and Soummya Kar. On best-response dynamics in potential games. *SIAM Journal on Control and Optimization*, 56(4):2734–2767, 2018.

50 Brian Woodbury Swenson. *Myopic Best-Response Learning in Large-Scale Games*. PhD thesis, Carnegie Mellon University, 2017. doi:10.1184/R1/6720788.v1.

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