Computing Approximate Pure Nash Equilibria in Digraph k-Coloring Games

Raffaello Carosi
Gran Sasso Science Institute - L'Aquila, Italy
raffaello.carosi@gssi.infn.it

Michele Flammini
Gran Sasso Science Institute, and DISIM - University of L'Aquila, Italy
michele.flammini@univaq.it

Gianpiero Monaco
DISIM - University of L'Aquila, Italy
gianpiero.monaco@univaq.it

ABSTRACT

We investigate approximate pure Nash equilibria in digraph k-coloring games, where we are given an unweighted directed graph together with a set of k colors. Vertices represent agents and arcs capture their mutual unidirectional interests. The strategy set of each agent v consists of the k colors and the payoff of v in a given state or coloring is given by the number of outgoing neighbors with a color different from the one of v. Such games form some of the basic payoff structures in game theory, model lots of real-world scenarios with selfish agents and extend or are related to several fundamental class of games.

It is known that the problem of understanding whether the game admits a pure Nash equilibrium is NP-complete. Therefore we focus on designing polynomial time algorithms that return approximate Nash equilibria. Informally, we say that a coloring is a γ-Nash equilibrium (for some $\gamma \geq 1$) if no agent can strictly improve her payoff by a multiplicative factor of γ by changing color. We first propose a deterministic polynomial time algorithm that, for any $k \geq 3$, returns a k-coloring that is a $\Delta_o(G)$-Nash equilibrium, where $\Delta_o(G)$ is the maximum outdegree of the digraph.

We then provide our two main results: i) By exploiting the constructive version of the well known Lovász Local Lemma, we show a randomized algorithm with polynomial expected running time that, given any constant $k \geq 2$, computes a constant-Nash equilibrium for a broad class of digraphs, i.e., for digraphs where, for any $v \in V$, $\delta_o^v(G) = \Omega(\ln \Delta_o(G) + \ln \Delta_i(G))$ where $\Delta_o(G)$ (resp. $\Delta_i(G)$) is the maximum outgoing (resp. maximum ingoing) degree of G, and $\delta_o^v(G)$ is the outgoing degree of agent v. ii) For generic digraphs, we show a deterministic polynomial time algorithm that computes a $(1+\epsilon)$-Nash equilibrium, for any $\epsilon > 0$, by using $O(\frac{\log n}{\epsilon})$ colors.

Keywords

Noncooperative games: computation; Game theory for practical applications.

1. INTRODUCTION

In this paper, we consider anti-coordination games where agents need to mutually anti coordinate their strategies in order to maximize their payoffs. Such games form some of the basic payoff structures in game theory and indeed they can model lots of real-world scenarios. For instance, agents can be miners deciding which land to drill for resources. A miner maximizes her happiness when the number of other miners that choose the same land is minimized. Or company employees trying to learn diverse skills. If an employee learns a skill that few other employees also learn, she will have more chance of exploiting it. Moreover, suppose that companies or countries have to produce some product. Typically they are interested in maximizing their profit, that most of the time corresponds to minimize the number of their trading competitors that produce the same commodity. Another example in a social networks setting could be the one in which people have to choose an outfit so that a minimum number of friends wears the same dress. Finally in a radio setting, radio towers are agents and their goal is selecting a frequency such that neighbouring radio-towers have a different one in order to minimize the interference. In all these scenarios agents have an interest in to mutually anti coordinate their strategies.

In general, each situation in which agents have to anti-coordinate their strategies so that the "penalty" (that can be of any type) on their reward is minimized can be described by anti-coordination games. Generally an agent does not care about all other agents' decisions, but wants to anti-coordinate her strategy only with a subset of agents of interest. Moreover, this interest may not be necessarily mutual but only one-sided.

A way of modeling this kind of setting with selfish agents is the following *graph k-coloring game*. There is a graph where a) agents are vertices, b) edges between agents indicate social connections, namely whether there is a mutual or unidirectional interest (depending on whether the graph is undirected or directed, respectively) about an agent's decision, and c) the set of the possible strategies is given by a natural number $k \geq 2$ representing the number of alternative available choices. Each agent picks a color (i.e., a number from 1 to k) and the resulting strategy vector induces a k-coloring of the nodes. The objective of every agent is to select the color that maximizes her payoff, which is equal to the number of neighbours with different color from her own.

In such a setting with selfish agents, a stable solution of the game is represented by a (pure) Nash equilibrium, that is a k-coloring where no agent can improve her payoff by changing color. The Nash equilibrium is one of the most important concepts in game theory, forming the basis of much

Appears in: *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

recent work in multiagent decision making and electronic marketplaces. As such, efficiently computing Nash equilibria is one of the most important problems in computational game theory.

When the graph is undirected, the graph k-coloring game is a potential game [19] and therefore a Nash equilibrium always exists. Moreover, when the graph is unweighted, the game converges (i.e., the dynamic where at each step one agent performs an improving move converges) to a Nash equilibrium in polynomial time [15, 18].

Unfortunately, in case of directed graphs, for any constant $k \geq 2$, the problem of understanding whether the game admits a Nash equilibrium is NP-complete [18]. Therefore, like in a variety of classes of games falling in this class, where Nash equilibria do not exist or cannot be computed in polynomial time, we focus on the milder form of approximate Nash equilibria. Namely, a state is called a γ-Nash equilibrium (for some $\gamma \geq 1$) if no agent can strictly improve her payoff by a factor of γ by changing her strategy. These equilibria can suitably model the cases in which agents incur a proportional cost in changing their strategies.

Motivated by the above results, we focus on the problem of computing γ-Nash equilibria for digraph k-coloring games. Specifically, we design polynomial time algorithms that return k-coloring where no agent can strictly improve her payoff by a factor of γ, for some fixed $\gamma \geq 1$.

We provide some positive results. In particular, we partially answer the following important questions: given a fixed number of available colors, for which family of digraphs we are able to compute good approximate Nash equilibria in polynomial time? Moreover, given a generic digraph, what is the minimum number of colors such that we are able to compute good approximate Nash equilibria in polynomial time?

To the best of our knowledge, this is the first paper providing polynomial time algorithms that compute approximate Nash equilibria for the digraph k-coloring game.

1.1 Our results

All our results refer to digraph k-coloring games induced by a directed graph G with n nodes, m edges and k available colors.

We first notice that, as proved in Section 2, a pure Nash equilibrium is not guaranteed to exist for any number of players n and colors $k < n$. In fact, even for $k = n - 1$, it is possible to provide an instance for which no Nash equilibrium exists, while $k = n$ clearly suffices as it can be easily achieved by assigning a different color to each node.

Moreover we remark that, for any $k \geq 2$, a Nash equilibrium (i.e., $\gamma = 1$) in bipartite digraphs and directed acyclic graphs (DAGs) exists and can be found in polynomial time (see Section 2). However, in general, even for $k = 2$ a γ-Nash equilibria might not exist for any bounded value of γ. In fact, in any 2-coloring of a directed odd cycle (i.e., a cycle with an odd number of nodes) there is always at least one node having utility zero. This property does not hold for $k > 2$. In fact, in Section 3.1 we first provide a polynomial time algorithm that, for any $k \geq 3$, returns a k-coloring that is a $\Delta_o(G)$-Nash equilibrium, where $\Delta_o(G)$ (resp. $\Delta_i(G)$) is the maximum outgoing (resp. maximum ingoing) degree of G.

Our main results are the following:

i) By exploiting the constructive version of the well known Lovász Local Lemma [20], in Section 3.2 we show a randomized algorithm with polynomial expected running time that, given any constant $k \geq 2$, computes a constant approximate Nash equilibrium[1] for a broad class of directed unweighted graphs. In particular, our algorithm works for digraphs $G = (V, E)$ where, for any $v \in V$, the outgoing degree of v is such that $\delta_o^v(G) = \Omega(\ln \Delta_o(G) + \ln \Delta_i(G))$. For instance this holds for digraphs where the minimum outgoing degree is $\Omega(\log n)$.

ii) Then, for generic digraphs, in Section 3.3 we show a deterministic polynomial time algorithm that computes $(1+\epsilon)$-Nash equilibria, for any $\epsilon > 0$, by using $O(\frac{\log n}{\epsilon})$ colors. Such a construction works also for a higher number of colors, showing that $(1+\epsilon)$-Nash equilibria exist and can be computed in polynomial time for $k = \Omega(\frac{\log n}{\epsilon})$. 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 $P = NP$.

1.2 Related Work

The graph k-coloring game has been first investigated in [15, 18], where the authors show that, when the graph is unweighted and undirected, it is possible to compute a Nash Equilibrium in polynomial time. When the graph is weighted undirected, the problem of computing an equilibrium is PLS-complete even for $k = 2$ [27] . In fact, for such a value of k, it coincides with the classical Max Cut game. Poljak et al. [25] prove that Nash equilibria can be computed in polynomial time for the cut game if the maximum degree of the graph is at most 3. Moreover, for graphs with maximum degree $d > 3$, they notice that it is easy to compute an approximate d-Nash equilibrium. [5, 9] give an algorithm that, for any $\epsilon > 0$, computes in polynomial time a $(3+\epsilon)$-equilibrium for the cut game. All the above results exploit the potential function method. However, digraph k-coloring games in general do not admit a potential function. Indeed, when the graph is unweighted directed, even the problem of understanding whether they admit a Nash equilibrium is NP-complete for any fixed $k \geq 2$ [18].

Besides cut games, digraph k-coloring games are related to many other fundamental games considered in the scientific literature.

One example is given by the graphical games introduced in [16]. In these games 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, where an arc (i, j) means that j influences i's payoff. An interesting class of graphical games is the one of the graphical congestion games [7], 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 [7] 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

[1]The constant depends on different parameters and, for the sake of readability, its value is not specified here. It can be found at the end of the analysis in Section 3.2.

(we describe the related algorithm in Section 2). 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 [29], [17]).

The game studied in this paper can also be seen as a particular hedonic game (see [4] for a nice introduction to hedonic games) with an upper bound (i.e., k) to the number of coalitions. Specifically, given a k-coloring, the agents with the same color can be seen as members of the same coalition of the hedonic game. In order to get the equivalence among the two games, the hedonic utility of an agent v can be defined as the overall number of her neighbors minus the number of agents of her neighbourhood that are in the same coalition. Nash equilibria issues in hedonic games have been largely investigated under several different assumptions [6, 8, 12, 13, 14, 24] (just to cite a few). An interesting recent study on Nash equilibria for graphical hedonic games can be found in [23]. To the best of our knowledge, no known result concerns approximate Nash equilibria.

Another related stream of research considers coordination games. The idea is that agents are rewarded for choosing common strategies in order to capture the influences. Apt et al. [2] propose a coordination game modeled as an undirected graph where vertices are agents and each agent has a list of allowed colors. Given a coloring, an agent has a payoff equal to the number of adjacent nodes with her same color. The authors show that the game converges to a NE in polynomial time. Apt et al. [3] consider the coordination game on directed weighted graphs. They prove that the problem of determining the existence of a Nash equilibrium on directed graphs is NP-complete. [3, 28] show that there exist some special digraph topologies for which Nash equilibria are guaranteed and can be found in polynomial time. Rahn et al. [26] extend the results of [2] by considering weighted undirected graphs and the problem of finding α-approximate k-equilibria, namely colorings where no coalition of at most k agents can deviate such that each member increases his payoff by at least a factor α. The authors show the existence of α-approximate k-equilibria for $\alpha > 2$. Anshelevich et al. [1] propose a coordination game where agents get incentive for coordinating. They show a polynomial time algorithm that computes a constant approximate Nash equilibrium with good social welfare.

Panagopoulou and Spirakis [22] study games where Nash equilibria are proper vertex coloring in undirected unweighted graphs setting. In particular, they consider the game where each agent v has to choose a color among k available ones and her payoff is equal to the number of vertices in the graph that have chosen her same color, unless some neighbour of v has chosen the same color, and in this case the payoff of v is 0. They prove that this is a potential game and that a Nash equilibrium can be found in polynomial time.

2. PRELIMINARIES

Given an unweighted directed simple graph (or simple digraph) $G = (V, E)$, we suppose that $|V| = n$ and $|E| = m$. An arc $(v, w) \in E$ is directed from node v to w; w is called the head and v is called the tail of the arc. An outgoing arc from node v is any arc $(v, w) \in E$. We denote by $\delta_o^v(G)$ the outgoing degree of node v, that is the number of outgoing arcs from node v in the graph G. An ingoing arc to node v is any arc $(w, v) \in E$. We denote by $\delta_i^v(G)$ the ingoing degree of node v. Moreover we denote with $d_o(G) = min_{v=1,\ldots,n}\delta_o^v(G)$ the minimum outgoing degree of G. Let $\Delta_o(G) = max_{v=1,\ldots,n}\delta_o^v(G)$ be the maximum outgoing degree of G. Let $\Delta_i(G)$ be the maximum ingoing degree of G. We will omit to specify (G) when clear from the context. Finally we denote with $G[V']$, where $V' \subseteq V$, the subgraph of G induced by V'.

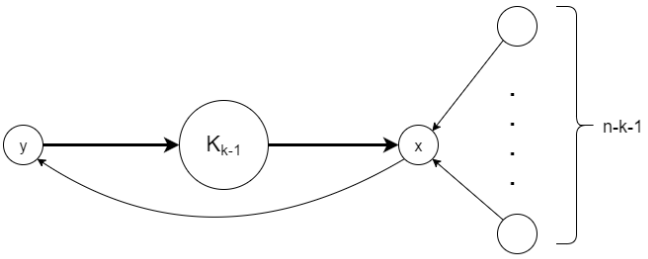

Figure 1: Instance for which there is no NE even with a number of colors linear in the number of players. A bold arrow means that there is complete incidence from the source subgraph to the target subgraph.

In the *digraph k-coloring game* we are given a directed unweighted graph $G = (V, E)$ without self loops in which each node $v \in V$ is a selfish agent (in the following we will use node and agent interchangeably), and a set of k available colors. Each agent has the same set of actions, which is the set of the k available colors. A state of the game $c = \{c_1, \ldots, c_n\}$ is a *k-coloring*, where c_v is the color (i.e., a number from 1 to k) chosen by agent v. In a certain coloring c, the payoff (or the utility) of an agent is the number of neighbors choosing colors different from her own, where the neighborhood of an agent v is the set of vertices induced by all v's outgoing edges. Formally, for a coloring c, an agent v's payoff $\mu_c(v) = \sum_{(v,w)\in E: c_v \neq c_w} 1$.

Let (c_{-v}, c'_v) denote the coloring obtained from c by changing the strategy of agent v from c_v to c'_v. A coloring c is a pure Nash or stable equilibrium, in the following NE for short, if no agent v can improve her payoff by changing strategy, i.e., color. Formally $c = \{c_1, \ldots, c_n\}$ is a NE if $\mu_c(v) \geq \mu_{(c_{-v}, c'_v)}(v)$ for any possible color c'_v and for any $v \in V$.

A NE is not guaranteed to exist even for a large number of available colors k. In particular, while there is always a NE with n colors just assigning to each node a different color, for every $k \leq n-1$ a NE is not guaranteed to exist. In fact, it is possible to prove the following proposition.

Proposition 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 nodes not admitting any NE.*

Proof. Consider the following instance of digraph k-coloring game: there are two nodes x, y, where x has an arc directed toward y; moreover, there is a complete directed clique K_{k-1} of size $k-1$, and each node in the clique has an arc directed toward x; finally, y has arcs directed toward all nodes 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 nodes in the clique must have the remaining $k-1$ colors, one per vertex. In fact, if some node 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 above graph of $k+1$ nodes $n-k-1$ additional dummy nodes with an arc directed toward x. The instance is depicted in Figure 1. □

We remark that Proposition 1 is not directly implied by the NP-completeness of determining the existence of a pure NE shown in [18], since the proof works only for $k \ll n$.

If a digraph is bipartite, a NE with $k \geq 2$ colors always exists and it can be easily found in polynomial time. In fact, 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, return a proper 2-coloring of it (i.e., if the set of nodes v is partitioned in to the sets V_1 and V_2, then all the nodes in V_1 are colored with color 1 and all the nodes in V_2 are colored with color 2. Clearly such a coloring maximizes the utility of all the agents.

It is also easy to see that, if a digraph is without cycles (i.e., it is a DAG), then a NE with $k \geq 2$ colors always exists and it can be easily found in polynomial time. In fact, 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, to return a topological sorting of the vertices, i.e., a linear ordering of its 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 color each node with its best response, that is choosing the color that maximizes her payoff given the choices made by the previous agents.

Unfortunately, for general directed graphs, the problem of determining whether the digraph k-coloring game admits a NE is NP-Hard, for all $k \geq 2$ [18]. For this reason, we consider the notion of approximate Nash equilibrium. A coloring c is a γ-Nash equilibrium (γ-NE or γ-stable for short), for some $\gamma \geq 1$, if no agent can strictly improve her payoff by a multiplicative factor of γ by changing color. Formally a coloring $c = \{c_1, \ldots, c_n\}$ is a γ-NE if $\gamma\mu_c(v) \geq \mu_{(c_{-v}, c'_v)}(v)$ for any possible color c'_v and for any $v \in V$.

The notion of approximate NE has been considered in many settings (see [21]). Typically, impossibility results (as in our case), together with the fact that in many real-life applications it may be the case that agents incur a proportional cost for changing their strategies, have naturally led researchers to consider this relaxed notion of Nash equilibrium.

3. APPROXIMATE-NE

In this section we show our polynomial time algorithms that compute approximate NE for digraph k-coloring games.

In order to get familiar with the model and before presenting our main results, we first show a polynomial time algorithm that for any $k \geq 3$ returns a k-coloring such that agents are in a $\Delta_o(G)$-NE. This solution is particularly suitable for graphs with small $\Delta_o(G)$.

We notice that when $k = 2$ any 2-coloring of a direct odd cycle (i.e., a cycle with an odd number of nodes) is not a γ-NE for any $\gamma \geq 1$. Indeed it is easy to see that for any possible coloring there exists a node having payoff zero.

3.1 Warm up: computing $\Delta_o(G)$-NE

We present a simple polynomial time algorithm that given a digraph G returns a k-coloring, for any $k \geq 3$ (indeed the algorithms uses three colors), where every node v such that $\delta_o^v(G) \geq 1$ has payoff at least 1. Clearly this corresponds to a $\Delta_o(G)$-NE because $\Delta_o(G)$ is the maximum payoff that any agent can achieve. The algorithm is iterative. At each iteration the algorithm visits the graph induced by the uncolored nodes and detects a cycle or a path (in the case of the visiting reaches a node without outgoing edge in the induced subgraph). Then it colors the nodes of the cycle or the path by alternating three colors (for instance colors 1,2 and 3) in a way that every node gets payoff of at least 1. In particular if the subgraph is a cycle then the algorithm considers nodes of the cycle in clockwise order and assigns the colors in such order (starting by any node) by alternating the three colors. If the subgraph is a path from node v to node w, then it colors the node w by a different color with respect to the already colored node u, if the arc $(w, u) \in E$ (otherwise it means that $\delta_o^w = 0$ and we can assign any color to w), and then alternates colors (in this case two colors are enough) for the other nodes of the path considered in the reverse order starting from w. We notice that if the algorithm does not detect any odd cycle then two colors are enough. A formal description of the algorithm is Algorithm Approx1.

The proof of the following theorem is straightforward.

Theorem 2. *Given a digraph G, algorithms Approx1 computes in polynomial time a coloring that uses at most three colors where each node v with $\delta_o^v \geq 1$ gets utility at least 1, i.e., Approx1 returns a $\Delta_o(G)$-NE.*

3.2 Constant approximate NE obtained by exploiting the Lovász Local Lemma

In this section we design an algorithm that for any constant $k \geq 2$, computes a constant approximate NE for a large class of digraph. In particular, by exploiting the well known Lovász Local Lemma [11], we first show that if we color each node 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 digraph where, for any $v \in V$, $\delta_o^v(G) = \Omega(\ln \Delta_o(G) + \ln \Delta_i(G))$. For instance this happens for digraphs where the minimum outgoing degree is $\Omega(\log n)$. It implies that constant approximate NE always exist for any constant value $k \geq 2$ in this class of digraphs. We point out 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 $k \geq 2$. Furthermore, we can use the results of Theorem 1.2 of [20] in order to get a randomized algorithm whose expected running time is polynomial, that computes such constant approximate NE (we will describe the algorithm we get from Theorem 1.2 of [20] at the end of this section).

The Lovász Local Lemma (LLL) is a powerful tool used for demonstrating that, given a large set of events with some dependencies among them, the probability that none of these

Algorithm: Algorithm Approx1

Input: a digraph $G = (V, E)$ and a set $1, \ldots, k$ of colors

$L \leftarrow$ empty list
while $\exists v \in V$ *still not colored* **do**
  $L = [v]$
  $x = v$
  while $\exists i \in V$ *such that* $\{x, i\} \in E$ *and* i *is still not colored* **do**
    if $i \in L$ **then**
      let $L' \subseteq L$ be the sublist of elements of L starting from node i up to the end
      if L' *is of even length* **then**
        color the nodes in L' by alternating two colors
        $x = i$
        break
      else
        color the nodes in $L' \backslash [i]$ by alternating two colors
        color i with a third color
        $x = i$
        break
    else
      $x = i$
      append i to the end of L
  if $v == x$ *and* v *is not colored* **then**
    if $deg^+(v) > 0$ **then**
      assign to v the color that maximizes its payoff
    else
      assign to v a random color
  if $v \neq x$ *and* x *is not colored* **then**
    if $deg^+(x) > 0$ **then**
      assign to x the color that maximizes its payoff
      color the nodes in $L \backslash \{x\}$ from the last node down to v by alternating a color different from $c(x)$ and $c(x)$ itself
    else
      color the nodes in L by alternating two colors

events happens is strictly greater than 0 if some conditions are met. We use the following version of the (LLL):

Definition 1. (LLL). Let $A_1, A_2, \ldots, A_n$ be a set of bad events, and let $D_i \subseteq \{A_1, A_2, \ldots, A_n\}$ denote the "dependency set" of A_i (namely A_i is mutually independent of all the events not in D_i). If there exists a set of real numbers $x_1, \ldots, x_n \in [0, 1)$ 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 show that the LLL can be used for proving that, under certain conditions on the structure of the digraph, there is positive probability that the Random k-coloring algorithm returns a constant approximate NE for any values of $k \geq 2$. That is, we show the existence of constant approximate NE for a broad subclass of digraphs. We define for each $v \in V$ a "bad event" I_v, that is the case in which in a coloring returned by Algorithm Random k-coloring the node v is not

Algorithm: Random k-coloring

Input: a digraph $G = (V, E)$ and a set $1, \ldots, k$ of colors

for $i = 1$ *to* n **do**
  randomly color node i by color $j = 1, \ldots, k$, with uniform probability $\frac{1}{k}$.

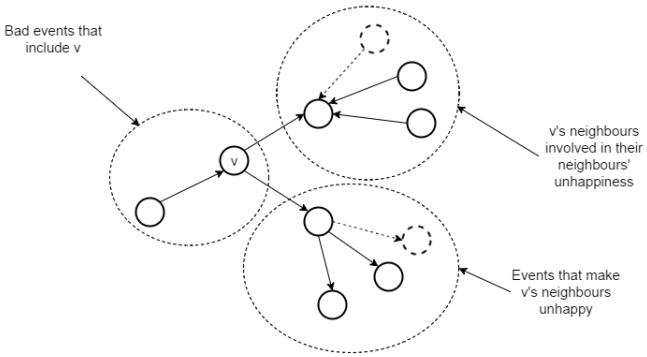

Figure 2: Graphical representation of all the types of event that can influence a node's behaviour

γ-stable for some constant γ (the value of γ will be computed at the end of the analysis):

Event I_v = *"node* v *is unhappy, i.e., it is not* γ*-stable."*

In order to apply the LLL we first need to bound the maximum size of the dependency set of each bad event. A generic event I_v is dependent on all the bad events that consider at least one node that regards I_v, namely v's neighbours and v itself. Thus, the possible event types in I_v's dependency set are shown below together with their maximum possible multiplicity in brackets:

- The events in which v's neighbours are involved in their neighbours' unhappiness ($\leq \delta_i^v \delta_o^v$);
- The events I_w, for any $w \neq v$ where v contributes to $w's$ unhappiness, namely there exists a directed edge (w, v) ($\leq \delta_i^v$)
- The events I_w, for any $w \neq v$, that make w unhappy, namely there exists a directed edge (v, w) ($\leq \delta_o^v$).

This means that a bad event I_v is dependent to at most $\delta_i^v \delta_o^v + \delta_i^v + \delta_o^v$ other bad events. The possible dependencies are shown graphically in Figure 2.

If we denote by dep_v the dependency set of I_v, then we have that:

$$|dep_v| \leq \delta_o^v + \delta_i^v + \delta_o^v \delta_i^v \quad \forall v \in V \tag{1}$$

Let $u_d = \Delta_o + \Delta_i + \Delta_o \Delta_i$ be an upper bound to the number of dependencies of any node, that is $dep_v \leq u_d$ for any $v \in V$. In the following we suppose that $dep_v \geq 2$. Indeed if $dep_v = 1$ then two cases are possible, that is either $\delta_o^v = 1$ and $\delta_i^v = 0$, or, $\delta_o^v = 0$ and $\delta_i^v = 1$. In the former case we can color node v at the end of the procedure with a color different than its unique (outgoing) neighbour. In fact its color does not affect the payoff of any other node. In the latter case, node v is always happy because it does not have

chance to get payoff greater than zero. If $d_v = 0$ then it is an isolated node and therefore we can forget about it. We thus consider that $u_d \geq 2$.

Theorem 3. *There exist values $x_v \in [0,1)$, for any $v = 1,\ldots,n$, such that, under certain conditions, $Pr[I_v] \leq x_v \prod_{w \in dep_v}(1-x_w)$ for all $v \in V$. That is the LLL holds.*

Proof. In order to show that the LLL holds, let us define $x_v = \frac{1}{u_d}\ \forall v \in V$. We want to show that if the following inequality is true:

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

Then, since $u_d \geq |dep_v|\ \forall v$, and $\left(1-\frac{1}{u_d}\right) < 1$, it implies:

$$Pr[I_v] \leq \frac{1}{u_d}\left(1-\frac{1}{u_d}\right)^{u_d} \leq \frac{1}{u_d}\left(1-\frac{1}{u_d}\right)^{|dep_v|} \quad \forall v \in V \tag{3}$$

namely the LLL holds. In order to prove equation (2) we use the Chernoff bound [10] that gives a bound on the deviation of the sum of random variables from their expected value.

Definition 2. (*Chernoff bound*). let $X_1, X_2, \ldots, X_n$ be independent random variables, and let X be their sum and $\mu = E[X]$ the expected value. Then for any $\beta > 0$:

$$Pr[X \geq (1+\beta)\,\mu] \leq e^{-\frac{\beta\mu}{3}} \tag{4}$$

So, in order to use the Chernoff bound we rewrite every bad event I_v as:

$$I_v = \text{“node } v \text{ has at least } (1+\beta)\,\frac{\delta_o^v}{k} \text{ neighbours colored with the same color”}$$

Notice that in the solution returned by Algorithm Random k-coloring, $\frac{\delta_o^v}{k}$ is the expected number of nodes w having v's color and such that there exists a directed edge (v,w). This is true since, with k colors, the probability that a node has the same color as v is $\frac{1}{k}$ and every node v has outdegree δ_o^v. We remark that we suppose k to be a constant value.

We use the Chernoff bound in equation (2) in the following way:

$$Pr[I_v] \leq e^{-\frac{\beta\delta_o^v}{3k}} \leq \frac{1}{u_d}\left(1-\frac{1}{u_d}\right)^{u_d} \tag{5}$$

We want to find for which values of β, δ_o^v and k the above inequality holds. Since the function $(1-\frac{1}{u_d})^{u_d}$ is increasing for any value $u_d > 1$ and moreover we suppose that $u_d \geq 2$, we obtain that $(1-\frac{1}{u_d})^{u_d} \geq \frac{1}{4}$. So we get from equation (5) that, if $e^{-\frac{\beta\delta_o^v}{3k}} \leq \frac{1}{4u_d}$, then equation 5 holds, namely the Lovász lemma holds. Therefore we obtain that for each node $v \in V$:

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

It implies that:

$$\delta_o^v \geq \frac{3k}{\beta}\left(\ln\Delta_o\Delta_i + \ln 4\right) \tag{6}$$

Therefore, since β and k are constant values, then we get that when $\delta_o^v = \Omega(\ln\Delta_o + \ln\Delta_i)$, i.e., when the outgoing degree of each node v is sufficiently large then 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 outgoing degree of all the nodes is the same (this is a broader class of digraphs than regular ones) then LLL holds when $\delta_o = \Omega(\ln\Delta_i)$. Finally we notice that the greater d_o is, the smaller β is required to be.

We now show which is the smallest value of γ such that γ-NE exists. For doing so, we consider the minimum possible value β such that the LLL still holds, that is, according to equation 6, equal to $\frac{3k}{\delta_o^v}\left((\ln\Delta_o + \ln\Delta_i) + \ln 4\right)$.

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

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

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

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

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

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

$$\approx \frac{k}{k-1} + \left(\frac{k}{k-1}\right)^2 O\left(\frac{1}{r-\frac{k}{k-1}}\right)$$

Where $r = \frac{\delta_o^v}{\ln\left(\Delta_o\Delta_i\right)}$.

It is easy to see that when $\delta_o^v = \Omega(\ln\Delta_o\Delta_i)$ the above value of γ is constant.

□

By summarizing we have proved that, under certain conditions, according to the LLL, Algorithm Random k-coloring returns a constant approximate NE with probability greater than zero.

Moreover, by Theorem 1.2 of [20] we know that there exists a simple randomized algorithm that returns a stable coloring. The algorithm is very simple: start from a random coloring (not necessarily stable) and, if exists, we arbitrarily

pick an unhappy node v (i.e. the bad event I_v holds) and we randomly assign a new color to all the nodes in the dependency set dep_v (this is called a resampling of the event I_v). We continue resampling violated events until the algorithm reaches a coloring that is a γ-NE (i.e, the values of γ we get in the above analysis). If the conditions of the LLL are satisfied, then [20] proofs that the algorithm terminates computing a γ-NE and the expected number of times an event is resampled is polynomial.

We point out that the question of whether the algorithm can be derandomized is an interesting issue. Unfortunately, the results of [20], Theorem 1.4, allow to obtain a deterministic algorithm with a running time that depends exponentially on the maximum number of dependencies per random variable. As a consequence, if the input graph is such that the in- and out-degree of each vertex are bounded by a constant, the conditions of Theorem 1.4 are satisfied and we directly obtain a deterministic polynomial time algorithm. However, for such a case, the simple polynomial time construction provided in Section 3.1 already gives a constant Nash equilibria with 3 colors. For general graphs, understanding whether the algorithm can be derandomized is a hard question, since it is directly connected to the more general question of improving the existing construction results of LLL.

3.3 Approximate NE for general digraphs 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 $O\left(\frac{\log n}{\epsilon}\right)$ colors. The algorithm is iterative. At the beginning all the nodes are not colored. In each iteration the algorithm colors a subset of nodes as described below.

Let V' be the current set of uncolored nodes (at the beginning $V' = V$). In each iteration i, we consider the subdigraph $G' = (V', E')$ of G induced by 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 then define the undirected graph $G'' = (V', E'')$ such that an undirected edge $\{v, w\}$ is in E'' if at least one between (v, w) and (w, v) is in E'. Let δ^v be the number of neighbors of v in G''.

Next, we use the algorithm described in Section 3 of [18] that computes a stable coloring for the unweighted undirected graph k'-coloring game (from now on we call this algorithm UndirectColoring). The underlying idea of the algorithm is the following: it starts from any arbitrary coloring and one node changes color if, by doing so, it strictly improves its utility (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 $V_1, V_2, \ldots, V_{k'}$ be the coloring of V' induced by UndirectColoring, namely $v \in V_j$ means that v is colored $c_{ik'-k'+j}$.

For each node $v \in V'$, if its outgoing degree in G' is at least $\lceil \frac{\delta^v}{3} \rceil$ then v is colored as in the equilibrium computed by UndirectColoring, namely v is colored $c_{ik'-k'+j}$ if $v \in V_j$. The algorithm updates the set of the uncolored nodes and iterates until all nodes are colored. A formal description of the algorithm is Algorithm Approx3.

We now show the performance and correctness of Algorithm Approx3.

Theorem 4. *Given any digraph $G = (V, E)$ and $\epsilon > 0$, algorithm Approx3 returns in polynomial time a $(1+\epsilon)$-NE by using at most $\frac{6(1+\varepsilon)}{\varepsilon} \log n = O(\frac{\log n}{\epsilon})$ colors.*

Algorithm: Approx3

Input: a digraph $G = (V, E)$, $k' = \lceil \frac{3(1+\varepsilon)}{\varepsilon} \rceil$.
Output: a coloring of G such that all nodes are in a $(1+\epsilon)$-NE, for any $\epsilon > 0$.

```
1  i = 1
2  V' = V
3  while V' ≠ ∅ do
4      c_{ik'-k'+1}, ..., c_{ik'} ← k' new colors
5      G' = (V', E') ← digraph such that
       E' := {(v, w) : v, w ∈ V'}
6      G'' = (V', E'') ← undirected graph such that
       E'' := {{v, w} : (v, w) ∈ E' ∨ (w, v) ∈ E'}
7      δ^v ← |{y ∈ V' : {v, y} ∈ E''}| ∀v ∈ V'
8      apply UndirectColoring to G'' and let V_1, V_2, ... V_{k'}
       be the partition of V induced by the coloring
9      for every node v ∈ V' do
10         if δ^v_o(G') ≥ ⌈δ^v/3⌉ then
11             color v c_{ik'-k'+j}, where v ∈ V_j
12             V' = V'\{v}
13     i = i+1
```

Proof. Consider an iteration i. If the coloring returned by UndirectColoring induces a partition $V_1, V_2, \ldots V_{k'}$ of V' in iteration i, then for each $v \in V'$ let $\delta^v_{V_j}$ be the number of neighbors that v has in subset V_j. We have:

$$\delta^v_{V_j} \geq \delta^v_{V_c} \quad \text{for all } v \in V_c, \text{ and for all } V_j, j \neq c \tag{7}$$

because otherwise v could improve its utility by changing its color, thus violating the fact that the solution returned is stable. Consider then a node $v \in V_c$. According to the coloring returned by UndirectColoring, its utility is:

$$\sum_{V_j : j \neq c} \delta^v_{V_j} \geq \left(k' - 1\right) \delta^v_{V_c} \tag{8}$$

This means that, since the degree of v in G'' is δ^v, the number of neighbors in the same cluster as v is at most:

$$\delta^v_{V_c} \leq \frac{\delta^v}{k'} = \frac{\delta^v}{\lceil \frac{3(1+\varepsilon)}{\varepsilon} \rceil} \leq \frac{\delta^v \varepsilon}{3\,(1+\varepsilon)} \tag{9}$$

Let us assume that v is colored at the end of iteration i. Then, by the construction of Approx3, $\delta^v_o(G') \geq \lceil \frac{\delta^v}{3} \rceil$, so that the approximation factor γ_v of node v is

$$\gamma_v \leq \frac{\text{maximum utility of v}}{\text{utility of v}} \leq \frac{\delta^v_o(G')}{\delta^v_o(G') - \frac{\delta^v \varepsilon}{3\,(1+\varepsilon)}} \leq$$

$$\leq \frac{\frac{\delta^v}{3}}{\frac{\delta^v}{3} - \frac{\delta^v \varepsilon}{3\,(1+\varepsilon)}} = \frac{1}{1 - \frac{\varepsilon}{(1+\varepsilon)}} = 1 + \varepsilon.$$

Thus, v is $(1+\varepsilon)$-stable, and since every node is colored during some iteration, Approx3 finally returns a $(1+\varepsilon)$-stable coloring.

We now show that Approx3 ends in a polynomial number of iterations by proving that we reduce the number of edges in G'' by a constant multiplicative factor in each iteration, until the algorithm ends. Consider again a generic iteration i. Let A be the set of nodes v such that $\delta_o^v(G') \geq \lceil \frac{\delta^v}{3} \rceil$ (namely they are colored) and let B be the remaining nodes. If E_i is the set of edges when iteration i starts and E_{i+1} is the set of edges remaining after iteration i, then:

$$|E_{i+1}| \leq \sum_{v \in B} \delta_o^v(G') \leq \frac{\sum_{v \in B} \delta_i^v(G')}{2} \leq \frac{|E_i|}{2} \qquad (10)$$

This is true because the first inequality holds by definition (we keep only arcs between uncolored nodes) and if the second inequality is not true, then there is some node in B that has been colored at the end of iteration i, namely it should be in set A. Thus, the maximum number of iterations is $\log|E| \leq 2\log n$. Since we use k' new colors in each iteration, the overall number of used colors is at most $2k' \log n \leq \frac{6\,(1+\varepsilon)}{\varepsilon} \log n = O(\frac{\log n}{\epsilon})$.

□

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 $k = \Omega\left(\frac{\log n}{\varepsilon}\right)$.

4. CONCLUSIONS AND FUTURE WORK

In this paper we considered the digraph k-coloring games, that are able to model lots of real-world scenarios with selfish agents and are related to some of the most fundamental classes of games investigated in the scientific literature.

We focused on Nash stability and, motivated by the fact that NE might not exist for every number of colors $k < n$ and that in general their determination is a NP-hard problem, we focused on the design of polynomial time algorithms that return approximate Nash Equilibria for various values of k. Clearly such results are stronger than mere existential ones, since they imply directly the existence and the fact that such equilibria can be suitably reached in practical scenarios.

Our paper should be seen as opening this research direction. In fact, we provided the first non trivial results for unweighted digraphs. However there are many open problems suggested by our work.

On one hand, it is important to understand what is the best approximate NE that can be computed in polynomial time for every $k = O(\log n)$. On the other hand, even if it turns out that it is computationally hard to determine good approximate Nash equilibria, for such values of k it would be nice also to prove that good approximate Nash equilibria always exist. In particular, what is the minimum value of γ guaranteeing the existence of a γ-approximate equilibria?

Moreover, a further step 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, its determination in polynomial time remains open. In this wording, it is also worth remarking that, even when the graph is weighted undirected (we recall that 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 $\gamma = 3 + \epsilon$. 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.

In general, we believe that questions related to the computational complexity of approximate Nash equilibria for the digraph k-coloring games deserve further investigation.

REFERENCES

[1] E. Anshelevich and S. Sekar. Approximate equilibrium and incentivizing social coordination. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, 2014.*, pages 508–514, 2014.

[2] K. R. Apt, M. Rahn, G. Schäfer, and S. Simon. Coordination games on graphs (extended abstract). In *Web and Internet Economics - 10th International Conference, WINE 2014. Proceedings*, pages 441–446, 2014.

[3] K. R. Apt, S. Simon, and D. Wojtczak. Coordination games on directed graphs. In *Proceedings Fifteenth Conference on Theoretical Aspects of Rationality and Knowledge, TARK 2015.*, pages 67–80, 2015.

[4] H. Aziz and R. Savani. Hedonic games. In *Handbook of Computational Social Choice*, chapter 15. Cambridge University Press, 2016.

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

[6] V. Bilò, A. Fanelli, M. Flammini, G. Monaco, and L. Moscardelli. On the price of stability of fractional hedonic games. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015*, pages 1239–1247, 2015.

[7] V. Bilò, A. Fanelli, M. Flammini, and L. Moscardelli. Graphical congestion games. *Algorithmica*, 61(2):274–297, 2011.

[8] F. Bloch and E. Diamantoudi. Noncooperative formation of coalitions in hedonic games. *Int. J. Game Theory*, 40(2):263–280, 2011.

[9] I. Caragiannis, A. Fanelli, and N. Gravin. Short sequences of improvement moves lead to approximate equilibria in constraint satisfaction games. In *Algorithmic Game Theory - 7th International Symposium, SAGT 2014, Proceedings*, pages 49–60, 2014.

[10] H. Chernoff. Chernoff bound. In *International Encyclopedia of Statistical Science*, pages 242–243. 2011.

[11] P. Erdős and L. Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. In *A. Hajnal, R. Rado, and V. T. Sós, eds. Infinite and Finite Sets (to Paul Erdős on his 60th birthday)*, pages 609–627. 1975.

[12] M. Feldman, L. Lewin-Eytan, and J. Naor. Hedonic

clustering games. *TOPC*, 2(1):4, 2015.

[13] M. Gairing and R. Savani. Computing stable outcomes in hedonic games. In *Algorithmic Game Theory - Third International Symposium, SAGT 2010*, pages 174–185, 2010.

[14] M. Gairing and R. Savani. Computing stable outcomes in hedonic games with voting-based deviations. In *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, pages 559–566, 2011.

[15] M. Hoefer. *Cost sharing and clustering under distributed competition*. PhD thesis, University of Konstanz, 2007.

[16] M. J. Kearns, M. L. Littman, and S. P. Singh. Graphical models for game theory. In *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, 2001*, pages 253–260, 2001.

[17] M. J. Kearns, M. L. Littman, and S. P. Singh. Graphical models for game theory. In *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, University of Washington, Seattle, Washington, USA, August 2-5, 2001*, pages 253–260, 2001.

[18] J. Kun, B. Powers, and L. Reyzin. Anti-coordination games and stable graph colorings. In *Algorithmic Game Theory - 6th International Symposium, SAGT 2013, Proceedings*, pages 122–133, 2013.

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

[20] R. A. Moser and G. Tardos. A constructive proof of the general lovász local lemma. *J. ACM*, 57(2), 2010.

[21] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

[22] P. N. Panagopoulou and P. G. Spirakis. A game theoretic approach for efficient graph coloring. In *Algorithms and Computation, 19th International Symposium, ISAAC 2008, Proceedings*, pages 183–195, 2008.

[23] D. Peters. Graphical hedonic games of bounded treewidth. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, 2016*, pages 586–593, 2016.

[24] D. Peters and E. Elkind. Simple causes of complexity in hedonic games. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, pages 617–623, 2015.

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

[26] M. Rahn and G. Schäfer. Efficient equilibria in polymatrix coordination games. In *Mathematical Foundations of Computer Science 2015 - 40th International Symposium, MFCS 2015, Proceedings*, pages 529–541, 2015.

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

[28] S. Simon and D. Wojtczak. Efficient local search in coordination games on graphs. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016*, pages 482–488, 2016.

[29] D. Vickrey and D. Koller. Multi-agent algorithms for solving graphical games. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence, July 28 - August 1, 2002, Edmonton, Alberta, Canada.*, pages 345–351, 2002.