The Power of the Defender*

Marina Gelastou†, Marios Mavronicolas†, Vicky Papadopoulou†, Anna Philippou†, and Paul Spirakis‡

Abstract

We consider a network security problem involving harmful procedures, (e.g. viruses) and a defender procedure (e.g. a system security software). Harmful entities choose a network node and damage it, if it is not protected by the security software. The security software is able to scan a limited part of the network and clean it from possible harmful entities. In [7], a basic case of the problem, called the Edge model, where the defender protects a single link of the network, was modelled and studied as a non-cooperative strategic game. Here, we consider a more general case where the defender is able to scan a set of $k$ links of the network, which we call the Tuple model. It is natural to expect that this increased power of the defender should result in a better quality of protection for the network. Ideally, this would be achieved at little expense on the existence and complexity of Nash equilibria in the model, that is configurations where no entity can unilaterally improve its local objective. We derive the following results:

- We show that the existence problem of pure Nash equilibria is solvable in polynomial time.
- We provide a graph-theoretic characterization of mixed Nash equilibria.
- Inspired by a class of polynomial-time Nash equilibria, called matching, introduced in the Edge model, we introduce $k$-matching configurations that generalize matching configurations.
- We provide a polynomial-time reduction for transforming any matching Nash equilibrium of any instance of the Edge model to a $k$-matching Nash equilibrium on a corresponding instance of the Tuple model and vice versa.
- Using the polynomial-time reduction, we provide a characterization of graphs admitting $k$-matching Nash equilibria. We develop a polynomial-time algorithm for computing $k$-matching Nash equilibria on graphs satisfying the characterization. We exhibit the applicability of the algorithm for bipartite graphs.
- We establish that the increased power of the defender results in an improved quality of protection of the network. In particular, for the case of matching Nash equilibria, we obtain that the gain of the defender, which amounts to the expected number of the arrested harmful procedures, is linear to the parameter $k$, that is, the number of network links the defender is able to scan and protect.

---

*This work was partially supported by the IST Programs of the European Union under contract numbers IST-2004-001907 (DELIS).

†Department of Computer Science, University of Cyprus, P.O. Box 20537, Nicosia CY-1678, Cyprus. \{gelastoum, mavronic,viki,annap\}@ucy.ac.cy

‡Department of Computer Engineering and Informatics, University of Patras, 265 00 Patras, Greece, & Research and Academic Computer Technology Institute, 261 10 Patras, Greece. spirakis@cti.gr
1 Introduction

1.1 Motivation

The recent huge growth of public networks (e.g. the Internet) has given Network Security, an issue of great importance in computer networks, an even more critical role [10]. This work considers a dimension of this area related to the protection of a system from harmful entities (e.g. viruses, worms, trojan horses, eavesdroppers [3]), when the power of the security software is limited. We consider an information network where the nodes of the network are insecure and vulnerable to infection by attackers such as viruses, Trojan horses, eavesdroppers. In particular, at any time, a number of harmful entities is known (or an upper bound of this number) to be present in the network. Each harmful entity targets a location (i.e. a node) of the network via a probability distribution; the node is damaged unless it is cleaned by the system security software. A defender, i.e. a system security software, is available in the system but it can guarantee security only to a limited part of the network, such as a link or a set of links, which it may choose using a probability distribution. Such limitations are due to financial costs, e.g. the cost of purchasing a global security software or due to performance reasons, e.g. the reduced efficiency or usability of the protected network part.

Obviously, the network entities involved in this scenario have opposite objectives; the system security software seeks to protect the network as much as possible, while the harmful entities wish to avoid being caught by the software so as to be able to damage the network. Thus, it is reasonable to view the problem as a non-cooperative game with players of conflicting interests. In [7] a basic case of this scenario, is modelled as a non-cooperative multi-player strategic game. The game is played on a graph with two kinds of players: the vertex players, which can choose a node of the network representing the harmful entities, and the defender, called the edge player, which can choose a single edge of the network, representing the system security software. The defender player seeks to maximize the expected number of vertex players it catches, while each vertex player seeks to maximize the probability of escaping the defender. The resulting game is called the Edge model. Such a modelling captures a simple case of the problem. At the same time, its simplicity enables a relative ease for exploring the problem using Graph-Theoretic tools. However, the Edge model considers only a restricted case for the system security software, where it can clean only a single link of the network each time.

In this work, we generalize the model of [7] by giving the defender player increased power. Specifically, it may choose a set of $k$ edges instead of only one. We call the resulting game as the Tuple model. Note that the Edge model is a special case of the Tuple model for $k = 1$. We are interested in the Nash equilibria [5, 6] associated with this game, where no player can unilaterally improve its individual objective by switching to a more advantageous probability distribution. Further, we investigate the trade-offs between the gains in system protection and the characterization and efficient computation of Nash equilibria caused by this increase in the defender’s power.

1.2 Summary of Results

Our contribution in this work is summarized as follows:

- We provide a graph-theoretic characterization of pure Nash equilibria of the Tuple model (Theorem 3.1). This result implies that the existence problem of pure Nash equilibria of the Tuple model is solvable in polynomial time (Corollary 3.2). The same result implies also that for non-trivial cases of graphs, i.e when the number of vertices of the related graph is
at least $2k + 1$, the instance contains no pure Nash equilibria (Corollary 3.3). A consequence of this result is that the increase in the defender’s power results in a greater class of graphs admitting pure Nash equilibria.

- Next, we provide a graph-theoretic characterization of mixed Nash equilibria of the problem (Theorem 3.4). Interestingly, the characterization suggests similar properties as the corresponding characterization of the Edge model of [7], indicating the extensibility of the latter model.

- Inspired by a class of polynomial-time Nash equilibria introduced in the Edge model, called matching, we investigate whether their variations can be useful to the more general game studied here. In particular, we introduce $k$-matching configurations for the Tuple model, that generalize matching configurations. We provide sufficient conditions for such a configuration to be a Nash equilibrium, called $k$-matching Nash equilibrium (Lemma 4.1).

- Furthermore, we discover a strong relationship between matching Nash equilibria of the Edge model and $k$-matching Nash equilibria introduced here: From any $k$-matching mixed Nash equilibrium of the Tuple model, a matching mixed Nash equilibrium of the Edge model can be computed in polynomial time and vice versa (Theorem 4.5). This, implies that the Tuple model is polynomial-time reducible to the Edge model with respect to $k$-matching Nash equilibria. The same result answers positively a conjecture stated in [7] about the applicability of matching equilibria in various network games suggesting them as a candidate Nash equilibrium for polynomial-time computation in various settings.

- The polynomial-time reduction between $k$-matching and matching Nash equilibria provided here (Theorem 4.5) implies a characterization of graphs admitting $k$-matching Nash equilibria (Corollary 4.11). Furthermore, it enables us to develop a polynomial-time algorithm for computing $k$-matching Nash equilibria for graph instances of the Tuple model that satisfy the characterization (Theorems 4.12, 4.13). In particular, the algorithm utilizes as a subroutine an algorithm of [7] for computing a matching Nash equilibrium of a corresponding graph. Then, it transforms it, using the proof of Theorem 4.5, to a $k$-matching Nash equilibrium of the graph, in time $O(k \cdot n)$, where $n$ is the number of nodes in the network.

- Finally, our study demonstrates the impact of the power of the defender, the parameter $k$, on the security of the network. We show that the gain of the defender depends linearly on the parameter $k$ in the Nash equilibria considered.

### 1.3 Related Work

This work contributes in the area of the quite broad field of Network Security, related to the protection of a network from harmful entities (e.g. viruses, worms, malicious procedures, or eavesdroppers). However, it differentiates from previous relative works, therefore becoming innovative, in that it considers a security problem from an alternative point of view than the usual. It considers a network security problem exploring tools of a new area, Algorithmic Game Theory and a quite developed field, Graph Theory. Network security problems have been first modelled as strategic games and associated Nash equilibria were studied on them in [4, 2, 7, 8].

[2] and [4] studied Interdependent Security games. In such a game, a large number of players must make individual investment decisions related to security, in which the ultimate safety of each participant may depend in a complex way on the actions of the entire population. In [2], the authors establish connections of the game considered with variants of a Graph Partition problem. Using them they provide polynomial-time Nash equilibria and prove $\mathcal{NP}$-hardness of finding best equilibria of the game studied.
[7] considers networks vulnerable to viral infection, where the system security software can guarantee safety only to a limited part of the network. In [7] the problem is modeled, for the first time, as a non-cooperative game with two kinds of players; the attackers and a defender entity. A basic case of the problem is considered, where the defender is able to guarantee safety to a single link of the network. Such problems differentiate from Interdependent Security problems in that the gain of an attacker player does not depend on the actions of any players other than the defender. [7] provides a non-existence result for pure Nash equilibria of the model. It introduces a family of structural Nash equilibria for the model, i.e., matching, it provides a characterization for their existence and a polynomial-time algorithm for their computation. Bipartite graphs are shown to possess such equilibria, thus the algorithm is applicable on such instances. In a subsequent work [8], we consider other families of structural Nash equilibria for the same problem in some practical families of graphs, such as regular graphs, graphs with perfect matchings and trees. We prove their existence and polynomial-time computation on corresponding instances. In the same work, we study also pure Nash equilibria for a generalized variation of the Edge model, where the defender is able to clean a path of the graph.

Another work which employs Game Theory on related problems is that of [3]. There, the study concerns the feasibility and computational complexity of two privacy tasks in distributed environments with mobile eavesdroppers; of distributed database maintenance and message transmission. A mobile eavesdropper is a computationally unbounded adversary that moves its bugging equipment within the system. However, [3] does not utilize Graph-Theoretic tools. In contrast, [1] employs Graph-Theoretic tools to study a two-player game on a graph. It establishes connections of the problem with the k-server problem and provides an approximate solution for the simple network design problem. However, this study does not concern network security problems.

2 The Model

We consider an undirected graph $G(V, E)$, with no isolated vertices, with $|V(G)| = n$ and $|E(G)| = m$, and an integer $1 \leq k \leq m$. When there is no confusion we omit $G$ in $V(G)$ and $E(G)$. For a set of vertices $X \subseteq V$, denote $Neigh_G(X) = \{u /: (u, v) \in E(G)\}$. Let $E^k$ be the set of all tuples of $k$ distinct edges of the graph $G$. When there is no confusion, we refer to a tuple of $k$ edges simply as a tuple. For any $t \in E^k$, let $V(t) = \{v /: (v, u) \in t\}$ be the set of the distinct endpoints of edges of the tuple $t$. Similarly, for any $t \in E^k$, let $E(t) = \{e /: e \in t\}$. Also, for any $T \subseteq E^k$, let $V(T) = \bigcup_{e \in T} V(e)$ and $E(T) = \bigcup_{e \in E} E(t)$. For any $T \subseteq E^k$, the graph obtained by $T$, denoted by $G_T$, has $V(G_T) = V(T)$ and $E(G_T) = E(T)$. Let $GCD(i, j)$ and $LCM(i, j)$ be the greatest common divisor and the least common multiple of the integers $i$ and $j$, respectively.

**Definition 2.1 (Tuple Model)** An information network is represented as an undirected, connected graph $G(V, E)$. The vertices represent the network hosts and the edges represent the communication links. We define a non-cooperative game $\Pi_k(G) = \langle N, \{S_i\}_{i \in N}, \{f_i\}_{i \in \mathcal{N}} \rangle$ on a graph $G$ where $k$ is an integer $1 \leq k \leq m$, as follows:

- **The set of players is** $N = N_{VP} \cup N_{TP}$, where $N_{VP}$ is a finite set of $\nu$ vertex players, $v_{pi}$, $1 \leq i \leq \nu$, and $N_{TP}$ is a singleton set of the tuple player, $tp$. We let $x$ range over $N$.

- **The strategy set $S_i$ of each player** $v_{pi} \in N_{VP}$, is $V$. The strategy set $S_{tp}$ of the tuple player is $E^k$, that is a strategy of $tp$ is any tuple containing $k$ edges of $G$. Thus, the strategy set of the game $S = V^\nu \times E^k$.

- **Take any strategy profile** $\mathbf{s} = (s_1, \ldots, s_\nu, s_{tp}) \in S$, called a configuration.
The Individual Profit of vertex player \( \text{vp}_i \) is a function \( \text{IP}_i : S \to \{0,1\} \) such that 
\[
\text{IP}_i(\vec{s}) = \begin{cases} 
0, & s_i \in V(s_{tp}) \\
1, & s_i \notin V(s_{tp}) 
\end{cases} ; \text{intuitively, } \text{vp}_i \text{ receives } 1 \text{ if it is not caught by the tuple player, and } 0 \text{ otherwise.}
\]

The Individual Profit of the tuple player is a function \( \text{IP}_{tp} : S \to \mathbb{N} \) such that 
\[
\text{IP}_{tp}(\vec{s}) = |\{s_i : s_i \in V(s_{tp})\}|.
\]

**Remark.** For \( k = 1 \), the Tuple model coincides with the Edge model. Thus, for any graph \( G \), \( \Pi_1(G) \) refers to an instance of the Edge model or an instance of the Tuple model for \( k = 1 \).

The configuration \( \vec{s} \) is a pure Nash equilibrium \([5, 6]\) (abbreviated as pure NE) if, for each player \( x \in \mathcal{N} \), it maximizes \( \text{IP}_x \) over all configurations \( t \) that differ from \( \vec{s} \) only with respect to the strategy of player \( x \). A mixed strategy profile or a mixed configuration for player \( x \in \mathcal{N} \) is a probability distribution over its strategy set \( S_x \); thus, a mixed configuration for a vertex player is a probability distribution over vertices of \( G \). A mixed strategy for the tuple player is a probability distribution over tuples of \( k \) edges of \( G \). A mixed configuration \( \vec{s} \) is a collection of mixed strategies, one for each player. Denote \( P_{g}(tp, t) \) the probability that the tuple player chooses the tuple of \( k \) edges \( t \in E^k \) in \( \vec{s} \); denote \( P_{g}(\text{vp}_i, v) \) the probability that player \( \text{vp}_i \) chooses vertex \( v \in V \) in \( \vec{s} \). Note that \( \sum_{v \in V} P_{g}(\text{vp}_i, v) = 1 \) for each player \( \text{vp}_i \); similarly, \( \sum_{t \in E^k} P_{g}(tp, t) = 1 \).

The support of a player \( x \in \mathcal{N} \) in a configuration \( \vec{s} \), denoted \( D_{g}(x) \), is the set of pure strategies in its strategy set to which \( x \) assigns strictly positive probability in \( \vec{s} \). Denote \( D_{g}(VP) = \bigcup_{v \in \mathcal{N}_V, p \in P} D_{g}(\text{vp}_i) \). Let also \( \text{Tuples}_{g}(v) = \{t : v \in V(t), t \in D_{g}(tp)\} \), i.e. set \( \text{Tuples}_{g}(v) \) contains all tuples \( t \) in the support of player \( tp \) such that \( v \in V(t) \). Given a configuration \( \vec{s} \), we denote \((\vec{s}_{-x}, [y])\) the configuration obtained by \( \vec{s} \), where all but player \( x \) play as in \( \vec{s} \) and player \( x \) plays the pure strategy \( y \).

Fix a mixed configuration \( \vec{s} \). For each vertex \( v \in V \), denote \( \text{Hit}(v) \) the event that the tuple player hits vertex \( v \). So, the probability of \( \text{Hit}(v) \) is \( P_{g}(\text{Hit}(v)) = \sum_{t \in \text{Tuples}_{g}(v)} P_{g}(tp, t) \). For each vertex \( v \in V \), denote \( m_{g}(v) \) the expected number of vertex players choosing \( v \). For each edge \( e = (u, v) \in E \), denote \( m_{g}(e) \) the expected number of vertex players choosing either \( u \) or \( v \); so, \( m_{g}(e) = m_{g}(u) + m_{g}(v) \). Moreover, for a tuple \( t \in E^k(G) \) denote \( m_{g}(t) \) the expected number of vertex players over the vertices of tuple \( t \); so \( m_{g}(t) = \sum_{v \in V(t)} m_{g}(v) \). It is easy to see that for each vertex \( v \in V \), \( m_{g}(v) = \sum_{v \in \mathcal{N}_V, p \in P} P_{g}(\text{vp}_i, v) \).

A mixed configuration \( \vec{s} \) induces an Expected Individual Profit \( \text{IP}_x \) for each player \( x \in \mathcal{N} \), which is the expectation, according to \( \vec{s} \), of its corresponding Individual Profit (defined previously for pure configurations). The mixed configuration \( \vec{s} \) is a mixed Nash equilibrium \([5, 6]\) (abbreviated as mixed NE) if for each player \( x \in \mathcal{N} \), it maximizes \( \text{IP}_x \) over all configurations \( t \) that differ from \( \vec{s} \) only with respect to the mixed strategy of player \( x \).

We proceed to calculate the Expected Individual Profit. Clearly, for the vertex player \( \text{vp}_i \in \mathcal{N}_{VP} \),
\[
\text{IP}_i(\vec{s}) = \sum_{v \in V(G)} P_{g}(\text{vp}_i, v) \cdot (1 - P_{g}(\text{Hit}(v)))
\]
\[
= \sum_{v \in V(G)} \left( P_{g}(\text{vp}_i, v) \cdot (1 - \sum_{t \in \text{Tuples}_{g}(v)} P_{g}(tp, t)) \right) 
\]
(1)
For the tuple player $tp$

$$
\text{IP}_{tp}(\vec{s}) = \sum_{t \in D_g(tp)} P_g(tp, t) \cdot m_g(t) = \sum_{t \in D_g(tp)} \left( P_g(tp, t) \cdot \sum_{v \in V(t)} m_g(v) \right)
= \sum_{t \in D_g(tp)} \left( P_g(tp, t) \cdot \sum_{v \in V(t)} \sum_{v \in N_D v, v} P_g(vp_i, v) \right)
$$

(2)

2.1 Background

A graph $G(V, E)$ is a bipartite graph if its vertex set can be partitioned as $V = V_1 \cup V_2$ such that each edge $e \in E$ has one of its vertices in $V_1$ and the other in $V_2$. Fix a set of vertices $S \subseteq V$. The graph $G$ is an $S$-expander graph if for every set $X \subseteq S$, $|X| \leq |\text{Neigh}_G(X)|$. A set $M \subseteq E$ is a matching of $G$ if no two edges in $M$ share a vertex. Given a matching $M$, say that set $S \subseteq V$ is matched in $M$ if for every vertex $v \in S$, there is an edge $(v, u) \in M$. A vertex cover of $G$ is a set $V' \subseteq V$ such that for every edge $(u, v) \in E$ either $u \in V'$ or $v \in V'$. An edge cover of $G$ is a set $E' \subseteq E$ such that for every vertex $v \in V$, there is an edge $(v, u) \in E'$. Say that an edge $(u, v) \in E$ (resp., a vertex $v \in V$) is covered by the vertex cover $V'$ (resp., the edge cover $E'$) if either $u \in V'$ or $v \in V'$ (resp., if there is an edge $(u, v) \in E'$). Otherwise, the edge (resp., the vertex) is not covered by the vertex cover (resp., the edge cover). A set $IS \subseteq V$ is an independent set of $G$ if for all vertices $u, v \in IS$, $(u, v) \notin E$. Clearly, $IS \subseteq V$ is an independent set of $G$ if and only if the set $VC = V \setminus IS$ is a vertex cover of $G$.

Our study utilizes the notion of matching NE defined for the Edge model in [7] and some related Theorems presented there. As explained in Remark 2, for any $G$, $\Pi_1(G)$ is both an instance of the Edge model and an instance of the Tuple model for $k = 1$. Thus, matching configurations of the Edge model can be defined in terms of the Tuple model as follows:

**Definition 2.2 ([7])** A matching configuration $\vec{s}$ of $\Pi_1(G)$ satisfies: (1) $D_g(vp)$ is an independent set of $G$ and (2) each vertex $v$ of $D_g(vp)$ is incident to only one edge of $D_g(tp)$.

**Lemma 2.1 ([7])** For any graph $G$, if (i) there exists a matching configuration $\vec{s}$ in $\Pi_1(G)$, (ii) $D_g(tp)$ is an edge cover of $G$ and (iii) $D_g(vp)$ is a vertex cover of the graph obtained by $D_g(tp)$, then, the strategy profile consisting of the uniform probability distribution on $D_g(vp)$ for every vertex player, and the uniform probability distribution on $D_g(tp)$ for the tuple (edge) player, gives rise to a mixed Nash equilibrium for $\Pi_1(G)$ which is called a matching NE.

**Theorem 2.2 ([7])** For any graph $G$, $\Pi_1(G)$ contains a matching mixed Nash equilibrium if and only if the vertices of the graph $G$ can be partitioned into two sets $IS$ and $VC$ ($VC = V \setminus IS$), such that $IS$ is an independent set of $G$ and $G$ is a $VC$-expander graph.

## 3 Nash Equilibria

For pure Nash equilibria we prove:

**Theorem 3.1** For any graph $G$, $\Pi_k(G)$ has a pure Nash equilibrium if and only if $G$ contains an edge cover of size $k$. 
Proof. Consider any graph $G$ that contains an edge cover $EC$ of size $k$. Consider any pure configuration $\vec{s}$ of $\Pi_k(G)$ such that $\vec{s}_{tp} = EC$ and the pure strategy of each one of the vertex players is any $v \in V$. Note that $V(\vec{s}_{tp}) = V$, thus, the Individual Profit of the tuple player $tp$ is

$$|\{(\vec{s}_i : \vec{s}_i \in V(\vec{s}_{tp}))\}| = \nu.$$ 

Obviously this value is the maximum possible of $IP_{tp}$. On the other hand, the Individual Profit of any vertex player on any $v \in V$ is 0, since $v \in V(\vec{s}_{tp})$. This implies that, for every $vp_i \in N_{vp}$ there is no $\vec{s}^* = (\vec{s}_{-vp_i}, u)$, $u \notin \vec{s}_i \in V$ such that $IP_{tp}(\vec{s}^*) > IP_{tp}(\vec{s})$. Thus, $\vec{s}$ is a pure NE.

Next, we prove that if $\Pi_k(G)$ has a pure NE, then $G$ contains an edge cover of size $k$. Assume the contrary, i.e., that any edge cover of $G$ has size at least $k + 1$. Consider any pure configuration $\vec{s}$ of $\Pi_k(G)$ that is a pure NE. Since $\vec{s}_{tp}$ consists of only $k$ edges, there is at least one vertex of $G$ that is not covered by $\vec{s}_{tp}$. Let $NC$ be the set of vertices not covered by $\vec{s}_{tp}$. Then, for all $vp_i \in N_{vp}$, $\vec{s}_i \in NC$ because these actions give the vertex players zero probability to be caught by the $tp$. But, in such a case, $IP_{tp} = 0$ since for all $vp_i \in N_{vp}$, $\vec{s}_i \notin D_{\vec{s}}(tp)$. This implies that there is at least one $\vec{s}^* = (\vec{s}_{-tp}, t')$, $t' \in E^k$ such that $IP_{tp}(\vec{s}^*) > IP_{tp}(\vec{s})$. Thus, $\vec{s}$ is not a pure NE, a contradiction to our initial assumption.

The above theorem implies that,

Corollary 3.2 For any graph $G$, the existence of pure Nash equilibria on $\Pi_k(G)$ can be solved in polynomial time.

Proof. For any graph $G$ and any integer $k$, one can check whether $G$ contains an edge cover of size $k$, and if so, compute such an edge cover, in polynomial time [11, chapter 3, pp. 115]. Thus, by Theorem 3.1, one can check if $\Pi_k(G)$ contains a pure NE and if so compute one, in polynomial time.

By Theorem 3.1, when $n \geq 2k + 1$ and since any edge cover of $G$ is of size at least $n/2 > k$, we get,

Corollary 3.3 If $|V(G)| \geq 2k + 1$, then $\Pi_k(G)$ has no pure Nash equilibrium.

Theorem 3.4 (Characterization) For any graph $G$, a mixed configuration $\vec{s}$ is a Nash equilibrium for any $\Pi_k(G)$ if and only if:

1. $E(D_{\vec{s}}(tp))$ is an edge cover of $G$ and $D_{\vec{s}}(VP)$ is a vertex cover of the graph obtained by $E(D_{\vec{s}}(tp))$.

2. The probability distribution of the tuple player over $E^k$, is such that, (a) $P_{\vec{s}}(Hit(v)) = P_{\vec{s}}(Hit(u)) = \min_v P_{\vec{s}}(Hit(v))$, $\forall u, v \in D_{\vec{s}}(VP)$ and (b) $\sum_{t \in D_{\vec{s}}(tp)} P_{\vec{s}}(tp, t) = 1$.

3. The probability distributions of the vertex players over $V$ are such that, (a) $m_{\vec{s}}(t_1) = m_{\vec{s}}(t_2) = \max_t m_{\vec{s}}(t)$, $\forall t_1, t_2 \in D_{\vec{s}}(tp)$ and (b) $\sum_{v \in V(D_{\vec{s}}(tp))} m_{\vec{s}}(v) = \nu$.

Proof. We proceed to prove three claims relevant to the proof of the theorem.

Claim 3.5 In any mixed NE $\vec{s}^*$ of $\Pi_k(G)$, $E(D_{\vec{s}^*}(tp))$ is an edge cover of $G$.

Proof. Assume the contrary. Let $NC$ be a set of vertices of $G$ not covered by $E(D_{\vec{s}^*}(tp))$. Then, for all $vp_i \in N_{vp}$, $D_{\vec{s}^*}(vp_i) \subseteq NC$ since these strategies give to the vertex players zero probability to be caught by the tuple player. This implies that $IP_{tp}(\vec{s}^*) = 0$ which can be improved if the tuple player selects any tuple of edges containing at least one vertex player. Thus, this configuration is not a NE, a contradiction. Henceforth, the initial assumption is false.
Claim 3.6 In any mixed NE \( s^* \) of \( \Pi_k(G) \), \( D_{s^*}(VP) \) is a vertex cover of the graph obtained by \( D_{s^*}(tp) \).

Proof. Assume the contrary, i.e. w.l.g. let \( t = \langle e_1, \ldots, e_k \rangle \in D_{s^*}(tp) \) such that \( e_k = (v, u) \), \( v, u \notin D_{s^*}(VP) \). Consider an alternative tuple \( t' = \langle e_1, \ldots, e_{k-1}, e'_k \rangle \in D_{s^*}(tp) \), such that \( e'_k = (v', u') \), \( e'_k \in E(D_{s^*}(tp)) \) and \( v' \neq D_{s^*}(VP) \), assume \( v' \). Note that such a tuple exists since \( |E(D_{s^*}(tp))| \geq k + 1 \). But then, \( m_{s^*}(t') = \sum_{u \in V} (e_1, \ldots, e_{k-1}) m_{s^*}(u) = m_{s^*}(v') > m_{s^*}(t) = \sum_{u \in V} (e_1, \ldots, e_{k-1}) m_{s^*}(u) \). Thus, \( tp \) can gain more by moving the probability of choosing tuple \( t \) to tuple \( t' \). Henceforth, \( s^* \) is not a mixed NE, a contradiction. First, note that \( |E(D_{s^*}(tp))| \geq k + 1 \), otherwise \( s^* \) is a pure configuration.

Claim 3.7 In any mixed configuration \( s^* \), of \( \Pi_k(G) \) \( \sum_{v \in V(D_{s^*}(tp))} m_{s^*}(v) = \nu \).

Proof.

\[
\sum_{v \in V(D_{s^*}(tp))} m_{s^*}(v) = \sum_{v \in V(D_{s^*}(tp))} \sum_{v_{Ip} \in \mathcal{N}_{IP}} P_{s^*}(v_{Ip}, v) = \sum_{v_{Ip} \in \mathcal{N}_{IP}} \sum_{v \in V(D_{s^*}(tp))} P_{s^*}(v_{Ip}, v) = \sum_{v_{Ip} \in \mathcal{N}_{IP}} 1 = \nu
\]

Returning to the proof of Theorem 3.4, next we prove that if \( s^* \) is a mixed NE for \( \Pi_k(G) \) then conditions 1-3 hold.
1.: By Claims 3.5 and 3.6.
2.(a): By equation (1) and since \( s^* \) is a Nash equilibrium, for any \( v_{Ip} \in \mathcal{N}_{IP} \) and \( u, v \in D_{s^*} \), we get \( IP_i(s^*) = IP_i(s_{-v_{Ip}}, v) = IP_i(s_{-v_{Ip}}, u) = 1 - P_{s^*}(Hit(v)) = 1 - P_{s^*}(Hit(u)) \). Hence, \( P_{s^*}(Hit(v)) = P_{s^*}(Hit(u)) \). Moreover, for any \( u' \in V, u' \notin D_{s^*}(VP) \), we get \( IP_i(s_{-v_{Ip}}, v) \geq IP_i(s_{-v_{Ip}}, u') \). Thus, by equation (1), \( 1 - P_{s^*}(Hit(v)) \geq 1 - P_{s^*}(Hit(u')) \) and \( P_{s^*}(Hit(v)) \leq P_{s^*}(Hit(u')) \). So \( P_{s^*}(Hit(v)) = P_{s^*}(Hit(u)) = \min_v P_{s^*}(Hit(v)), \) for all \( u, v \in D_{s^*} \).
2.(b): Obvious, since \( P_{s^*}(tp) \) is a probability distribution over tuples of edges in \( E_k(G) \).
3.(a): By equation (2) and the fact that \( s^* \) is a Nash equilibrium, we get \( IP_{tp}(s^*) = IP_{tp}(s_{-tp}, [t]) = \sum_{v_{tp} \in \mathcal{N}_{tp}} P_{s^*}(v_{tp}, [t]) = m_{s^*}(t_1) = m_{s^*}(t_2), \) for all \( t_1, t_2 \in D_{s^*}(tp) \). Moreover, for any \( t \notin D_{s^*}(tp), \) \( t_1 \in D_{s^*}(tp) \), we get \( IP_{tp}(s^*) \geq IP_{tp}(s_{-tp}, [t]) \). Thus, \( m_{s^*}(t_1) \geq m_{s^*}(t) \). So \( m_{s^*}(t_1) = m_{s^*}(t_2) = \max m_{s^*}(t) \), for any \( t_1, t_2 \in D_{s^*}(tp) \).
3.(b): By Claim 3.7.

Next, we prove that if conditions 1-3 hold for a mixed strategy profile \( s \) then \( s \) is a Nash equilibrium. Consider first the vertex players. For any player \( v_{Ip} \), by 2.(a), any \( v \in D_{s^*}(v_{Ip}) \), has minimum hitting probability in \( s \). Thus, any of these vertices is a best response ([9], chapter 3) choice for player \( v_{Ip} \). Hence, player \( v_{Ip} \) is satisfied in \( s \). Consider now the tuple player. By 3.(a), any \( t \in D_{s^*}(tp) \), has a maximum expected number of vertex players located on it. Thus, \( t \) is a best response choice for \( tp \) in \( s \) and henceforth \( tp \) is also satisfied in \( s \).

4 \( k \)-Matching Nash Equilibria

In [7] a special class of polynomial-time Nash equilibria is introduced for the Edge model, called \textit{matching} Nash equilibrium, based on the notion of matching configurations. In this section, we extend this class of equilibria to the Tuple model. We begin by defining \textit{k-matching} configurations which generalize \textit{matching} configurations and we provide sufficient conditions for such configurations to also form Nash equilibria, which we call \textit{k-matching} Nash equilibria. Furthermore, we
develop a polynomial-time reduction that, given a matching mixed Nash equilibrium of an instance of the Edge model, computes a $k$-matching mixed Nash equilibrium of a corresponding instance of the Tuple model, and vice versa. The reduction implies a characterization of graphs admitting $k$-matching Nash equilibria. We utilize this reduction to develop a polynomial-time algorithm for computing $k$-matching Nash equilibria for graph instances that satisfy the characterization. We proceed with the definition of $k$-matching configurations.

**Definition 4.1** A $k$-matching configuration $\bar{s}$ of $\Pi_k(G)$ satisfies: (1) $D_\bar{s}(VP)$ is an independent set of $G$, (2) each vertex $v$ of $D_\bar{s}(VP)$ is incident to only one edge of the edge set $E(D_\bar{s}(tp))$ and (3) each edge $e \in E(D_\bar{s}(tp))$ belongs to an equal number of distinct tuples in $D_\bar{s}(tp)$.

**Observation 4.1** For $k = 1$, $k$-matching configurations on $\Pi_k(G)$ coincide with matching configurations of the Edge model on $\Pi_1(G)$.

**Proof.** Obviously, an 1-matching configuration $\bar{s}$ of $\Pi_1(G)$ is also a matching configuration of $\Pi_1(G)$. We show that if $\bar{s}$ is a matching configuration of $\Pi_1(G)$, then $\bar{s}$ is also an 1-matching configuration of $\Pi_1(G)$. Conditions (1) and (2) of a matching configuration are identical with conditions (1) and (2) of an 1-matching configuration, thus they are both satisfied. Lastly, an edge $e \in E(D_\bar{s}(tp))$ may belong to exactly one pure strategy, since in the edge model strategies are tuples of size one (i.e. edges), and Condition (3) of an 1-matching configuration is also satisfied.

**Lemma 4.1** For any graph $G$, if in $\Pi_k(G)$ there exists a $k$-matching configuration $\bar{s}$ which additionally satisfies condition 1 of Theorem 3.4, then there exist polynomial-time computable probability distributions for the players on $\bar{s}$, such that the resulting configuration is a mixed Nash equilibrium for $\Pi_k(G)$.

**Proof.** Consider any configuration $\bar{s}$ as stated by the Lemma, assuming that there exists one. Let the following probability distributions of the vertex player and the tuple player:

$$\forall t \in D_\bar{s}(tp), \ P_{\bar{s}}(tp, t) := 1/|D_\bar{s}(tp)| \quad \text{and} \quad \forall t' \in E^k, \ t' \notin D_\bar{s}(tp), \ P_{\bar{s}}(tp, t') := 0 \quad (3)$$

For any $vp_i \in NV_P: \forall v \in D_\bar{s}(VP), \ P_{\bar{s}}(vp_i, v) := \frac{1}{|D_\bar{s}(vp_i)|} \quad \text{and} \quad \forall u \in V, \ u \notin D_\bar{s}(VP), \ P_{\bar{s}}(vp_i, u) := 0 \quad (4)$

We proceed with a sequence of claims utilized next in the proof.

**Claim 4.2**

$$\forall v \in D_\bar{s}(VP), \ m_{\bar{s}}(v) = \frac{\nu}{|D_\bar{s}(VP)|} \quad \text{and} \quad \forall u \in V, \ u \notin D_\bar{s}(VP), \ m_{\bar{s}}(u) = 0$$

**Proof.**

$$\forall v \in D_\bar{s}(VP), \ m_{\bar{s}}(v) = \sum_{vp_i \in NV_P} P_{\bar{s}}(vp_i, v) = \sum_{vp_i \in NV_P} \frac{1}{|D_\bar{s}(VP)|} = \frac{\nu}{|D_\bar{s}(VP)|}$$

by equation (4). In contrast, for any other vertex $u \in V, \ u \notin D_\bar{s}(VP)$, by the same equation, $m_{\bar{s}}(u) = 0$.

**Claim 4.3** $P_{\bar{s}}(\text{Hit}(v)) = \frac{k}{|E(D_\bar{s}(tp))|}, \ \forall \ v \in D_\bar{s}(VP)$. 

8
Proof.

∀ v ∈ Dg(VP), Pg(Hit(v)) = \sum_{t ∈ Tuplesg(v)} \frac{1}{|Dg(tp)|} \cdot |Tuplesg(v)| \quad (5)

by equation (3).

Next, we compute Tuplesg(v). According to condition (2) of the definition of a k-matching configuration, each vertex v ∈ Dg(VP) is incident to only one edge of E(Dg(tp)), i.e., there exists exactly one edge e = (v, u) ∈ E(Dg(tp)). Thus, |Tuplesg(v)| is equal to the number of tuples t ∈ Dg(tp) containing edge e. Let α be the number of the tuples t ∈ Dg(tp) containing edge e. Then, by equation (5)

Pg(\text{Hit}(v)) = \frac{\alpha}{|Dg(tp)|} \quad (6)

According to condition (3) of the definition of a k-matching configuration, each edge e ∈ E(Dg(tp)) belongs to an equal number of tuples in Dg(tp). Thus, this number is equal to α. Henceforth, Pg(\text{Hit}(v)) = Pg(\text{Hit}(u)), for any v, u ∈ Dg(VP). Moreover, α · |E(Dg(tp))| is the size of the multiset of edges contained in Dg(tp) which is also equal to |Dg(tp)| · k. This implies that |Dg(tp)| = \frac{\alpha}{k} |E(Dg(tp))|. Hence, equation (6) gives,

Pg(\text{Hit}(v)) = \frac{\alpha}{|E(Dg(tp))|} = \frac{k}{|E(Dg(tp))|}, \quad \forall \ v ∈ Dg(VP).

Claim 4.4 \hspace{1cm} Pg(\text{Hit}(u)) \geq \frac{k}{|E(Dg(tp))|} = Pg(\text{Hit}(v)), \forall \ u \notin Dg(VP) \text{ and } v \in Dg(VP).

Proof. As in the previous claim,

∀ u \notin Dg(VP), Pg(\text{Hit}(u)) = \sum_{t ∈ Tuplesg(u)} \frac{1}{|Dg(tp)|} \cdot \frac{1}{|Dg(tp)|} \quad (7)

by equation (3).

Consider such a vertex u \notin Dg(VP). Consider v ∈ Dg(VP) such that (v, u) ∈ t, t ∈ Dg(tp). According to condition (2) of the definition of a k-matching configuration, vertex v is incident to only one edge of the edge set E(Dg(tp)), assume edge (v, u). Thus, each t ∈ Tuplesg(v) is of the form \{(v, u), \ldots , (v', u')\}, since all tuples containing vertex v, contain actually edge (v, u). So, for all t ∈ Tuplesg(v) it holds that t ∈ Tuplesg(u). Henceforth, Tuplesg(v) ⊆ Tuplesg(u) and |Tuplesg(v)| ≤ |Tuplesg(u)|. From equation (7) and Claim 4.3, we get

Pg(\text{Hit}(u)) \geq Pg(\text{Hit}(v)) \Leftrightarrow Pg(\text{Hit}(u)) \geq \frac{|Tuplesg(v)|}{|E(Dg(tp))|} = \frac{k}{|E(Dg(tp))|} = Pg(\text{Hit}(v)) \quad (8)

Returning to the proof of the lemma, we show that ̄s satisfies all conditions of Theorem 3.4, thus it is a mixed NE.

1. By assumption.  
2.(a): By Claims 4.3 and 4.4.  
2.(b): \sum_{t ∈ Dg(tp)} Pg(tp, t) = \sum_{t ∈ Dg(tp)} \frac{1}{|Dg(tp)|} = 1.  
3.(a): m̄g(t1) = \sum_{v ∈ V(t1)} m̄g(v), for any t1 ∈ Dg(tp). By condition (2) of a k-matching configuration, t1 covers exactly k distinct vertices v1, \ldots , v_k ∈ Dg(VP) and at most k vertices...
\[ u_1, \ldots, u_k \notin D_g(VP). \] By Claim 4.2, we get that \( m_g(v_1^1) = \ldots = m_g(v_1^k) = \frac{k \nu}{|D_g(VP)|} \) and \( m_g(u_1) = \ldots = m_g(u_k) = 0. \) Thus, \( m_g(t_1) = \frac{k \nu}{|D_g(VP)|}. \) Moreover, \( m_g(t_2) = \sum v \in V(t_2) m_g(v) \) for all \( t_2 \notin D_g(tp). \) Note that \( t_2 \) covers at most \( k \) vertices \( v_1^1, \ldots, v_k^1 \in D_g(VP). \) By Claim 4.2 again, we get that \( m_g(v_1^2) = \ldots = m_g(v_k^2) = \frac{k \nu}{|D_g(VP)|}. \) Thus \( m_g(t_2) \leq \frac{k \nu}{|D_g(VP)|} + \sum u \in V(t_2), u \notin D_g(VP) m_g(u). \) Note that the latter sum equals to 0 according to Claim 4.2. Henceforth, \( m_g(t_2) \leq \frac{k \nu}{|D_g(VP)|} = m_g(t_1) \) for all \( t_1 \in D_g(tp), t_2 \notin D_g(tp). \)

3. (b): Since \( D_{tp}(\bar{s}) \) is an edge cover of \( G \) (by assumption), by Claim 4.2, we have

\[ \sum_{v \in V(D_g(tp))} m_g(v) = \sum_{v \in V} \frac{\nu}{|D_g(VP)|} = |D_g(VP)| \cdot \frac{\nu}{|D_g(VP)|} = \nu. \]

Note that the probability distributions for the vertex players and the tuple player can be computed in polynomial time.

**Definition 4.2** A \( k \)-matching configuration which additionally satisfies condition 1 of Theorem 3.4 is called a \( k \)-matching mixed NE.

**Theorem 4.5** For any \( G \), from any matching mixed Nash equilibrium \( \bar{s}' \) of \( \Pi_1(G) \) we can compute in polynomial time a \( k \)-matching mixed Nash equilibrium \( \bar{s} \) of \( \Pi_k(G) \) and vice versa. Further, for \( \bar{s} \) and \( \bar{s}' \) it holds that \( |\Pi_{tp}(\bar{s})| = k \cdot |\Pi_{tp}(\bar{s}')| \).

**Proof.**

**Lemma 4.6** For any \( G \), from any \( k \)-matching mixed NE \( \bar{s} \) of \( \Pi_k(G) \), we can compute a matching mixed NE \( \bar{s}' \) of \( \Pi_1(G) \), in polynomial time.

**Proof.** Let \( \bar{s} \) be a \( k \)-matching mixed NE of \( \Pi_k(G) \). We construct a configuration \( \bar{s}' \) of \( \Pi_1(G) \) as follows: \( D_g(VP) = D_g(VP), D_g(vp_i) := D_g(VP), \forall vp_i \in N_{\bar{s}p}, D_g(tp) := E(D_g(tp)). \) Apply the uniform probability distribution for all players. Next, we show that configuration \( \bar{s}' \) is a matching configuration of \( \Pi_1(G) \).

Observe that the definition of a \( k \)-matching configuration differs from a matching configuration (Definition 2.2) only in condition (2) and in that it was supplemented with condition (3). Thus, condition (1) of the definition of a matching configuration is fulfilled in the constructed configuration \( \bar{s}' \). In the definition of a matching configuration (Definition 2.2), condition (2) requires that each vertex \( v \in D_g(VP) \) to be incident only to one edge of the \( D_g(tp) \). In the definition of a \( k \)-matching configuration (Definition 4.1), condition (2) requires that each vertex \( v \in D_g(VP) \) is incident to only one edge of set \( E(D_g(tp)) \). However, \( D_g(tp) = E(D_g(tp)) \). Since \( \bar{s}' \) satisfies condition (2) of the definition of a \( k \)-matching configuration, we get that condition (2) of the definition of a matching configuration is also fulfilled in \( \bar{s}' \). Hence, \( \bar{s}' \) is a matching configuration of \( \Pi_1(G) \).

Next, we show that \( \bar{s}' \) satisfies also condition (ii) of Lemma 2.1, i.e., that \( D_g(tp) \) is an edge cover of \( G \) and that \( D_g(vp) \) is a vertex cover of the graph obtained by \( D_g(tp) \). Since \( D_g(tp) = E(D_g(tp)) \) and \( E(D_g(tp)) \) is an edge cover of \( G \) in instance \( \Pi_k(G) \) (recall that \( \bar{s} \) is a mixed NE), \( D_g(tp) \) is an edge cover of \( G \) in instance \( \Pi_1(G) \). Moreover, the subgraph of \( G \) obtained by \( D_g(tp) \) in \( \Pi_1(G) \) is equal to the subgraph of \( G \) obtained by \( E(D_g(tp)) \) in \( \Pi_k(G) \). \( D_g(VP) \) is a vertex cover of the graph obtained by \( E(D_g(tp)) \) and \( D_g(VP) = D_g(VP) \). Thus, \( D_g(VP) \) is a vertex cover of the graph obtained by \( D_g(tp) \). Henceforth, \( \bar{s}' \) satisfies condition (ii) of Lemma 2.1 on \( \Pi_1(G) \). Moreover, \( \bar{s}' \) uses the uniform probability distribution as required by Lemma 2.1. Thus, by the lemma, it is a matching NE of \( \Pi_1(G) \). Finally, note that, \( \bar{s}' \) is constructed in polynomial time.
Corollary 4.7 In configurations $\vec{s}$ of $\Pi_k(G)$ and $\vec{s}'$ of $\Pi_1(G)$ of Lemma 4.6 it holds that, $m_{\vec{s}}(tp) = k \cdot m_{\vec{s}'}(tp)$.

**Proof.** Consider any $k$-matching Nash equilibrium $\vec{s}$. Since $\vec{s}$ is a Nash equilibrium, for any $t \in D_{\vec{s}}(tp)$, we get $|P_{t}(\vec{s})| = |P_{t}(\vec{s}'-tp,t) = m_{\vec{s}}(t)$. By Claim 4.2, $m_{\vec{s}}(t) = |V(t)| m_{\vec{s}'}(v) = \sum_{v \in V(t)} |D_{\vec{s}}(VP)|^{-1} \nu$. Moreover, since $D_{\vec{s}}(VP)$ is an independent set of $G$ (condition (1) of a $k$-matching configuration), we get,

$$|P_{t}(\vec{s})| = \sum_{v \in V(t)} |D_{\vec{s}}(VP)|^{-1} \nu = k \cdot |D_{\vec{s}}(VP)|^{-1} \cdot \nu$$

Now, consider the matching configuration $\vec{s}'$ constructed by $\vec{s}$ according to Lemma 4.6. Since $\vec{s}'$ is a Nash equilibrium, for any $e = (v, u) \in D_{\vec{s}'}(tp), v \in D_{\vec{s}'}(VP), |P_{e}(\vec{s})| = |P_{e}(\vec{s}'-tp, e) = m_{\vec{s}'}(e)$. Moreover, since $D_{\vec{s}'}(VP)$ is an independent set of $G$ (condition (1) of a matching configuration), we have $m_{\vec{s}'}(e) = m_{\vec{s}'}(v)$. Moreover, since $\vec{s}'$ uses a uniform distribution, we get,

$$|P_{e}(\vec{s})| = m_{\vec{s}'}(v) = \sum_{i \in N_{VP}} P_{\vec{s}'}(vp_i,v) = \nu \cdot \frac{1}{|D_{\vec{s}'}(VP)|} = \frac{\nu}{|D_{\vec{s}'}(VP)|}$$

**Lemma 4.8 For any matching mixed NE $\vec{s}'$ of $\Pi_1(G)$ we can compute a $k$-matching mixed NE $\vec{s}$ of $\Pi_k(G)$ in polynomial time.**

**Proof.** We compute a set of tuples of $k$ edges as follows: We label the edges in set $D_{\vec{s}'}(tp)$ with consecutive numbers, starting 0 to $E_{\text{num}} - 1$, where $E_{\text{num}} = |D_{\vec{s}}(tp)|$. Then we construct consecutive tuples $t_i, i \geq 1$ by letting

$$t_i = \langle e(i-1) \cdot k \mod(E_{\text{num}}), \ldots, e(i-k-1) \mod(E_{\text{num}}) \rangle$$

This construction allows us to move cyclically around the edges and choose consecutive $k$-tuples as we proceed. Let set $T = \{t_1, t_2, \ldots, t_k\}$ be the set of the $\delta$ first tuples we construct. Letting

$$\delta = \frac{E_{\text{num}}}{\text{GCD}(E_{\text{num}}, k)}$$

the last edge of tuple $t_\delta$ is edge

$$e(\frac{E_{\text{num}}}{\text{GCD}(E_{\text{num}}, k)} \cdot k - 1 \mod(E_{\text{num}})) = e(\text{LCM}(E_{\text{num}}, k) - 1 \mod(E_{\text{num}})) = e E_{\text{num}} - 1$$

i.e., is the last edge of set $D_{\vec{s}}(tp)$. Since we start creating tuples starting from the first edge of $D_{\vec{s}}(tp)$, we visit each edge of $D_{\vec{s}}(tp)$ and add it to $T$, the same number of times. Moreover, by our choice of $\delta$, since $\delta \cdot k = \frac{E_{\text{num}}}{\text{GCD}(E_{\text{num}}, k)} \cdot k = \text{LCM}(E_{\text{num}}, k), T$ contains the least number of tuples containing each edge an equal number of times. Furthermore, we can compute this number: 

**Claim 4.9 Each edge $e \in E(D_{\vec{s}}(tp))$ belongs to exactly $k \cdot \frac{E_{\text{num}}}{\text{GCD}(E_{\text{num}}, k)}$ tuples.**

**Proof.** The set $T$ constructed in the previous lemma, contains $\delta = \frac{E_{\text{num}}}{\text{GCD}(E_{\text{num}}, k)}$ tuples, in which each edge appears an equal number of times. The size of the multiset of edges contained in $T$ is

$$\frac{E_{\text{num}}}{\text{GCD}(E_{\text{num}}, k)} \cdot k$$

since each tuple contains $k$ edges. Hence, each edge $e \in E(D_{\vec{s}}(tp))$ belongs to exactly

$$\frac{\frac{E_{\text{num}}}{\text{GCD}(E_{\text{num}}, k)} \cdot k}{E_{\text{num}}} = k \cdot \frac{GCD(E_{\text{num}}, k)}{E_{\text{num}}}$$

tuples. 

11
Now we are ready to construct the following configuration $\bar{s}$ on $\Pi_k(G)$: Set $D_g(VP) := D_g(VP)$, $D_g(vp_i) := D_g(VP)$, $\forall vp_i \in N_{VP}$ and $D_g(tp) := T$.

We first show that $\bar{s}$ is a $k$-matching configuration of $\Pi_k(G)$. Condition (1) of a $k$-matching configuration is fulfilled because condition (1) of a matching configuration is fulfilled in $\bar{s}$ and $D_g(VP) = D_g(VP)$. Condition (2) of a $k$-matching configuration is also fulfilled in $\bar{s}$ because condition (2) of a matching configuration is fulfilled in $\bar{s}$ and $E(D_g(tp)) = D_g(tp)$. Moreover, by Claim 4.9, each edge $e \in E(D_g(tp))$ belongs to an equal number of tuples, thus, condition (3) of the definition of $k$-matching configuration is also fulfilled. Hence, $\bar{s}$ is a $k$-matching configuration of $\Pi_k(G)$.

We next show that condition 1 of Theorem 3.4 is satisfied by $\bar{s}$. We first show that $E(D_g(tp))$ is an edge cover of $G$. This is true because $E(D_g(tp)) = D_g(tp)$ and $D_g(tp)$ is an edge cover of $G$, by condition (1) of a matching configuration (Definition 2.2). Thus, $E(D_g(tp))$ is an edge cover of the graph $G$. We next show that $D_g(VP)$ is a vertex cover of the subgraph of $G$ obtained by $E(D_g(tp))$. Since $E(D_g(tp)) = D_g(tp)$, the subgraph of $G$ obtained by $E(D_g(tp))$ is equal to the subgraph of $G$ obtained by $D_g(tp)$. Moreover, $D_g(VP) = D_g(VP)$ and $D_g(VP)$ is a vertex cover of the subgraph obtained by $D_g(tp)$, by condition (1) of a matching configuration. Hence, $D_g(VP)$ is a vertex cover of the subgraph of $G$ obtained by $E(D_g(tp))$. We conclude that condition 1 of Theorem 3.4 is satisfied by $\bar{s}$. Thus, $\bar{s}$ is a $k$-matching mixed NE of $\Pi_k(G)$ according to Lemma 4.1. Moreover, note that configuration $\bar{s}$ is computed in polynomial time.

**Corollary 4.10** In configurations $\bar{s}$ of $\Pi_k(G)$ and $\bar{s}'$ of $\Pi_1(G)$ of Lemma 4.8 it holds that, $m_{g}(tp) = k \cdot m_{g}(tp)$.

**Proof.** Consider any matching NE $\bar{s}'$ of $\Pi_1(G)$. Since $\bar{s}'$ is a Nash equilibrium, for any $e = (v, u) \in D_{g}(tp)$, $v \in D_{g}(VP)$, $IP_{tp}(\bar{s}') = IP_{tp}(\bar{s}'_{-tp}, e) = m_{g}(e)$. Moreover, since $D_{g}(VP)$ is an independent set of $G$ (condition (1) of a matching configuration) and $\bar{s}'$ is a uniform distribution, we get,

$$IP_{tp}(\bar{s}') = m_{g}(v) = \sum_{i \in N_{VP}} P_{g}(vp_i, v) = \nu \cdot \frac{1}{|D_{g}(VP)|}$$

(11)

Now, consider the $k$-matching Nash equilibrium $\bar{s}$ constructed by $\bar{s}'$, according to Lemma 4.8. Since $\bar{s}$ is a Nash equilibrium, for any $t \in D_g(tp)$, we get $IP_{tp}(\bar{s}) = IP_{tp}(\bar{s}_{-tp}, t) = m_g(t)$. Moreover, since $\bar{s}$ is a uniform distribution, for any $v \in D_g(VP)$, $m_g(v) = \sum_{i \in N_{VP}} P_{g}(vp_i, v) = \nu \cdot \frac{1}{|D_{g}(VP)|}$. Thus, recalling that $D_{g}(VP)$ is an independent set of $G$ (condition (1) of a $k$-matching NE) and by the construction of $\bar{s}$, we get,

$$IP_{tp}(\bar{s}) = m_g(t) = \sum_{v \in V(t)} m_g(v) = k \cdot \nu \cdot \frac{1}{|D_{g}(VP)|}$$

(12)

Lemmas 4.6 and 4.8 prove the first statement of the Theorem and Corollaries 4.7 and 4.10, the second.

We proceed to characterize graphs that admit $k$-matching Nash equilibria.

**Corollary 4.11 (characterization of $k$-matching NE)** For any graph $G$, $\Pi_k(G)$ contains a matching mixed Nash equilibrium if and only if the vertices of the graph $G$ can be partitioned into two sets $IS, VC$ ($VC = V \setminus IS$), such that $IS$ is an independent set of $G$ and $G$ is a $VC$-expander graph.

**Proof.** By Theorem 4.5, for any graph $G$, $\Pi_k(G)$ contains a $k$-matching mixed Nash equilibrium if and only if $\Pi_1(G)$ contains a matching Nash equilibrium. Henceforth, by Theorem 2.2 we get the claim.
4.1 A Polynomial Time Algorithm

We utilize the proof of Theorem 4.5 to develop a polynomial-time algorithm for finding $k$-matching mixed NE for any $\Pi_k(G)$, assuming such an equilibrium exists. The algorithm, called $A_{tuple}$, uses as a subroutine, an algorithm of [7] for computing matching NE of the Edge model. This algorithm, denoted by $A(\Pi(G), IS, VC)$, assumes the existence of a partition of the vertices of $G$ into sets $IS$ and $VC$, as in Theorem 2.2. Algorithm $A_{tuple}$ is described in pseudocode in Figure 1.

Algorithm $A_{tuple}(\Pi_k(G), IS, VC)$

**INPUT:** A game $\Pi_k(G)$, a partition of $V(G)$ into sets $IS$, $VC = V \setminus IS$, such that $IS$ is an independent set of $G$ and $G$ is a $VC$-expander graph.

**OUTPUT:** A mixed NE $\vec{s}$ for $\Pi_k(G)$.

1. $\vec{s'} := A(\Pi_1(G), IS, VC)$.
2. Label the edges of set $D_{\vec{s'}}(tp)$ with consecutive integers starting from 0, i.e., $e_0, e_1, \ldots$.
3. Compute a set $T$ of tuples as follows:
   (a) **Initialization:** Set $T := \emptyset$, $CurrentEdge := 0$ (label of current edge of $D_{\vec{s'}}(tp)$) and $E_{num} = |D_{\vec{s'}}(tp)|$.
   (b) While $TRUE$ do:
      i. Set $tuple := \langle \rangle$
      ii. for ($i = 1; i \leq k; i++$)
         A. $tuple := tuple \cup \langle e_{CurrentEdge}\rangle$.
         B. $CurrentEdge = (CurrentEdge + 1)\mod(E_{num})$
      iii. $T := T \cup \{tuple\}$
      iv. If $CurrentEdge \mod(E_{num}) == 0$ then Exit While loop
4. Define a configuration $\vec{s}$ with the following support: $D_{\vec{s}}(VP) := IS$, $D_{\vec{s}}(tp) := T$.
5. Apply on $\vec{s}$ the probabilities distributions specified by equations (3) and (4) of Lemma 4.1, for respective players.

Figure 1: Algorithm $A_{tuple}$

**Theorem 4.12 (Correctness)** Algorithm $A_{tuple}$ computes a $k$-matching mixed Nash equilibrium of $\Pi_k(G)$.

**Proof.** We first show that $\Pi_k(G)$ contains a $k$-matching NE. Recall that algorithm $A_{tuple}$ takes as input sets $IS$ and $VC$ such that $IS$ is an independent set of $G$, $VC = V \setminus IS$ and $G$ is a $VC$-expander graph. When such sets are given as input for the algorithm, by Theorem 4.11, we get that $\Pi_k(G)$ contains a $k$-matching NE. We proceed to show that the algorithm computes such an equilibrium.

Recall that sets $IS$ and $VC$ given as inputs in algorithm $A_{tuple}$ satisfy the requirements of corresponding sets of Lemma 2.2 applied on instance $\Pi_1(G)$ for the existence of a matching NE. Thus, Step 1, i.e. the call of algorithm $A$ on $\Pi_1(G)$, terminates successfully computing a matching mixed NE of $\Pi_1(G)$, $\vec{s'}$. 

13
Now, recall the computation of a \( k \)-matching NE for \( \Pi_k(G) \) from a matching NE of \( \Pi_1(G) \) in the proof of Theorem 4.5. Note that the construction of set \( T \) in step 3 of the algorithm is the same as the construction of set \( T \) in the proof of the theorem. Furthermore, the support of configuration \( \tilde{s} \) constructed in the proof of the theorem is the same as that of configuration \( \tilde{s} \) computed here. Thus, using the same arguments as in the theorem, we can prove that configuration \( \tilde{s} \) constructed here, is a \( k \)-matching configuration of \( \Pi_k(G) \). Also, note that the probabilities assignment of configuration \( \tilde{s} \) in step 4 of the algorithm, is the same as the probability assignment of configuration \( \tilde{s} \) of Theorem 4.5. Henceforth, \( \tilde{s} \) is a \( k \)-matching mixed NE of \( \Pi_k(G) \). ■

Theorem 4.13 (Time Complexity) Algorithm \( A_{\text{tuple}} \) terminates in time \( O(k \cdot n) \).

Proof. Step 1 of algorithm \( A_{\text{tuple}} \) requires \( O(n) \) time [8]. Step 2 labels the edges of the set \( D_\ell(tp) \). Since configuration \( \tilde{s} \) is a matching configuration, by condition (2) of Definition 2.2, each vertex \( v \) of \( D_\ell(VP) \) is incident to only one edge of \( E(D_\ell(tp)) \). Recall that \( E(D_\ell(tp)) = D_\ell(tp) \). Thus, it holds that \( |E(D_\ell(tp))| = |D_\ell(VP)| \) and \( |E(D_\ell(tp))| \leq n \). Hence, step 2 takes \( O(n) \) time. The inner loop (for loop) of step 3(b) iterates \( k \) times in every iteration of the outer loop (while loop). The outer loop terminates after exactly \( \left\lceil \frac{E_{\text{num}}}{\gcd(E_{\text{num}},k)} \right\rceil \) iterations (Claim 4.9 of Theorem 4.5). In the worst case, \( \frac{E_{\text{num}}}{\gcd(E_{\text{num}},k)} = E_{\text{num}} \). Thus, step 3(b) takes time \( O(k \cdot E_{\text{num}}) \). Since \( k \leq n \) and \( E_{\text{num}} = |E(D_\ell(tp))| \), step 3 takes time \( O(k \cdot n) \). Steps 4 and 5 are simple assignments and take time \( O(1) \). We conclude that Algorithm \( A_{\text{tuple}} \) needs \( O(k \cdot n) \) time to be completed. ■

5 Applications

We demonstrate the applicability of \( k \)-matching NE on a quite broad family of graphs, that of bipartite graphs. We next show that bipartite graphs possess such equilibria and one can compute them in polynomial time.

Theorem 5.1 For any \( \Pi_k(G) \), for which \( G \) is a bipartite graph, a \( k \)-matching mixed Nash equilibrium of \( \Pi_k(G) \) can be computed in polynomial time, \( \max\{O(k \cdot n), O(m\sqrt{n}), O(n^{2.5}/\sqrt{\log\ n})\} \), using Algorithm \( A_{\text{tuple}} \).

Proof. We first show that any bipartite graph \( G \) posses a \( k \)-matching NE. In [8] it was proved that for any bipartite graph \( G \), \( \Pi_1(G) \) contains a matching mixed NE. Thus, by Theorem 4.5, \( \Pi_k(G) \) contains a \( k \)-matching mixed NE.

Next, we show that a \( k \)-matching NE of \( \Pi_k(G) \) can be computed in polynomial time. Apply algorithm \( A_{\text{tuple}}(\Pi_k(G), IS, VC) \), where \( VC \) is a minimum vertex cover and \( IS = V \setminus VC \) is an independent set of \( G \). In [8], it was proved that a minimum vertex cover of a bipartite graph can be computed in polynomial time, \( \max\{O(m\sqrt{n}), O(n^{2.5}/\sqrt{\log\ n})\} \). It was also proved that such sets \( IS, VC \) satisfies the requirements of the parameters of algorithm \( A \), i.e. \( A(\Pi_1(G), IS, VC) \) is applicable. Henceforth, step (1) of \( A_{\text{tuple}} \) terminates successfully. Thus, algorithm \( A_{\text{tuple}} \) can be accomplished successfully, producing a \( k \)-matching NE of \( \Pi_k(G) \) in \( O(k \cdot n) \) time (Theorem 4.13). So, in total, we used max\{\(O(k \cdot n), O(m\sqrt{n}), O(n^{2.5}/\sqrt{\log\ n})\}\) time. ■

References


