A Path-following Polynomial Equations Systems Approach for Computing Nash Equilibria

Hélène Fargier
Université de Toulouse, IRIT
F-31062, Toulouse, France
helene.fargier@irit.fr

Paul Jourdan
Université de Toulouse, INRAE-MIAT
F-31320, Castanet-Tolosan, France
paul.jourdan@inrae.fr

Régis Sabbadin
Université de Toulouse, INRAE-MIAT
F-31320, Castanet-Tolosan, France
regis.sabbadin@inrae.fr

ABSTRACT

This paper presents a path-following combinatorial framework based on systems of polynomial equations to compute a mixed Nash equilibrium in N-person games. We provide the first detailed implementable description of Wilson's path-following method, extending Lemke-Howson's algorithm to N-person games and handling degenerate games. Our approach resembles, in some respects, support enumeration methods. We thus compare both approaches, theoretically and experimentally. Then, we show that the path-following approach allows to deal with a large family of succinctly expressed games: hypergraphical games, graphical games and polymatrix games. The described algorithms have been implemented in Python, making use of Sagemath libraries to solve systems of polynomial equations, allowing an experimental comparison of the different combinatorial approaches on a large variety of games.

KEYWORDS

Game Theory, Nash Equilibrium, Path Following Algorithm

ACM Reference Format:
Hélène Fargier, Paul Jourdan, and Régis Sabbadin. 2022. A Path-following Polynomial Equations Systems Approach for Computing Nash Equilibria. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 9 pages.

1 INTRODUCTION

The geometric path-following approach of [23] extends Lemke and Howson's 2-player games algorithm [17] and theoretically allows for the computation of exact mixed Nash equilibria of N-person games. However, while implementations of Lemke-Howson's type algorithms do exist for bimatrix and polymatrix games [14], Wilson's approach has never been implemented. Wilson only showed that computing a mixed Nash equilibrium amounts to finding a solution to a *Polynomial Complementary Problem (PCP)* and provided a high-level informal mathematical description of his approach. On different grounds, [21] provided a simple Nash equilibrium computation algorithm based on an exhaustive exploration of strategies' supports, exploiting polynomial systems and dominated strategies computation. This method is still state of the art for practical exact NE computation in N-person games. In this article, we compare both approaches theoretically. Then we extend the path-following approach in order to handle (i) degenerate games and (ii) polymatrix [24], graphical [16] and hypergraphical games [20]. Finally, we provide implementations of support enumeration and path-following approaches, allowing comparisons on different classes of games.

In Section 2, we present the PCP formulation of the Nash equilibrium search problem introduced in [23] and show how [21]'s support enumeration method can also be cast in the PCP framework. In Section 3 we present an original "combinatorial" view of [23]'s path-following algorithm and in Section 4 we extend it to handle degenerate PCP (approximately, in some cases). In Section 5, we show that Nash equilibrium computation in graphical and hypergraphical games [16, 20] can be formulated as a PCP of size polynomial in the concise expression of the games. Finally, Section 6 provides an experimental comparison of both approaches[1].

2 N-PLAYER GAMES AND POLYNOMIAL COMPLEMENTARITY PROBLEMS (PCP)

2.1 Nash equilibria as solutions of a PCP

Let us consider a N-player game $\Gamma^N = (P, \pi, a)$. $P = \{1, \ldots, N\}$ is the set of players. $\pi = S_1 \times \ldots \times S_N$ is the set of joint pure strategies of the game (S_n is the finite set of pure strategies available to player $n \in P$). a^n_ω is the strictly positive disutility[2] received by player n when the joint pure strategy is $\omega \in \pi$. $a = (a^n_\omega)_{\omega \in \pi, n \in P}$ can thus be represented as a matrix with $|\pi| = \prod_{n=1..N} |S_n|$ lines and N columns. A mixed strategy $\xi^n = (\xi^n_i)_{i \in S_n}$ is a probability distribution over the pure strategies of player n. A joint mixed strategy is a tuple $\xi = (\xi^n)_{n \in P}$ of mixed strategies. ξ^{-n} denotes the mixed strategies of all players except n. Hence, $\xi = (\xi^n, \xi^{-n})$.

A mixed Nash equilibrium of a game Γ^N is defined as:

Definition 2.1 (Mixed Nash equilibrium). Joint mixed strategy $\xi = (\xi^n_i)_{n \in P, i \in S_n}$ is a mixed Nash equilibrium of game $\Gamma^N = (P, \pi, a)$ if and only if: $\forall n \in P, Dis_n[\xi] \leq Dis_n[(\bar{\xi}^n, \xi^{-n})], \forall \bar{\xi}^n \neq \xi^n$, where

$$Dis_n[\xi] =_{def} \sum_{\omega=(\omega_1,\ldots,\omega_n) \in \pi} a^n_\omega \prod_{\nu=1}^{N} \xi^\nu_{\omega_\nu}$$

and $(\bar{\xi}^n, \xi^{-n})$ is the joint mixed strategy where the mixed strategy ξ^n has been replaced with mixed strategy $\bar{\xi}^n$.

[23] showed that a mixed Nash equilibrium can be computed from a solution of a *Polynomial Complementarity Problem (PCP)* . Let us define E^π, the Euclidian space of dimension $D = \sum_{n \in P} |S_n|$. Joint mixed strategies will be indirectly represented by lists of

[1] Python/Sagemath codes as well as tutorial Jupyter notebooks are available online: https://forgemia.inra.fr/game-theory-tools-group/gtnash-git/.
[2] The PCP formulation considers that players are disutility minimizers. This is without loss of generality since any utility maximization game can be equivalently expressed as a disutility minimization game.

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Online.

vectors $x = (x^n)_{n\in P}$ of coordinates $(x_i^n)_{i\in S_n} \in E^\pi$. The x^n's are non-negative but generally not normalized (we will build a probability distribution by normalization of x^n). Let us define multilinear polynomials A_i^n:

$$A_i^n(x^{-n}) \quad =_{def} \quad \sum_{\substack{\omega\in\pi\\ \omega_n=i}} a_\omega^n \prod_{\nu\neq n} x_{\omega_\nu}^\nu, \forall n \in P, i \in S_n \tag{1}$$

$$\text{and } A^n(x) \quad =_{def} \quad \sum_{i\in S_n} A_i^n(x^{-n}) x_i^n, \forall n \in P \tag{2}$$

Where, by definition, $x^{-n} = (x_i^\nu)_{\nu\in P\setminus\{n\}, i\in S_\nu}$.

When every x^n is normalized, i.e. $x^n = \xi^n$ are mixed strategies, $A^n(\xi)$ is the expected disutility of joint mixed strategy ξ to player n and $A_i^n(\xi^{-n})$ is the expected disutility of player n when n plays pure strategy $i \in S_n$ instead of mixed strategy ξ^n.

By definition of a mixed Nash equilibrium, ξ is a mixed NE iff:

$$\begin{cases} A^n(\xi) & \leq & A_i^n(\xi^{-n}), \forall n \in P, i \in S_n, \\ A^n(\xi) & = & A_i^n(\xi^{-n}), \forall n \in P, i \in S_n, \text{ s.t. } \xi_i^n > 0. \end{cases} \tag{3}$$

By a change of variables, system (3) can be represented by the following *Polynomial Complementarity Problem* [23]:

Definition 2.2 (Polynomial Complementarity Problem). The PCP corresponding to $\Gamma^N = (P, \pi, a)$ is a system of equations/inequations in variables $(x_i^n)_{n\in P, i\in S_n} \in E^\pi$:

$$\forall (n,i) \in I_N, \begin{cases} x_i^n & \geq & 0 \\ A_i^n(x^{-n}) & \geq & 1 \\ x_i^n \cdot \left(A_i^n(x^{-n}) - 1\right) & = & 0 \end{cases} \left(\mathcal{S}^N\right)$$

where $I_N =_{def} \{(n,i), n \in P, i \in S_n\}$.

$\mathcal{D}^N$ denotes the set of points $x \in E^\pi$ satisfying the set of inequations of the PCP: $\mathcal{D}^N = \left\{x \in E^\pi, x_i^n \geq 0,\ A_i^n(x^{-n}) \geq 1, \forall (n,i) \in I_N\right\}$. Problem $\mathcal{S}^N$ is called a polynomial complementarity problem since we look for a non-negative solution x such that for any $(n,i) \in I_N$, either $x_i^n = 0$ or $A_i^n(x^{-n}) = 1$ holds (hence a complementary solution) and $A_i^n(x^{-n})$ are multivariate polynomials in variables $\left(x_i^n\right)_{n\in P, i\in S_i}$. x^n representing an unnormalized probability distribution, the equalities in $\mathcal{S}^N$ correspond to usual best response arguments: if i is not a best response to a NE for player n (i.e. $A_i^n(x^{-n}) > 1$), its probability should be 0 (thus, also x_i^n).

In the case where there are only two players ($N = 2$), one can check that, $\forall (n,i) \in I_N, A_i^n(x^{-n}) = 1$ is a linear equation. Hence, in this case the obtained problem $\mathcal{S}^2$ is a *Linear Complementarity Problem* [17]. [23] shows the equivalence between the Nash equilibria of a game and the solutions of the corresponding PCP:

Proposition 2.3 (NE / PCP equivalence [23]). *Let Γ^N be a N-player game and $\mathcal{S}^N$ its PCP transformation. Then, the Nash equilibria of Γ^N and the solutions of $\mathcal{S}^N$ are in one-to-one correspondence, i.e.:*

(1) If x is a solution of $\mathcal{S}^N$, then ξ defined by $\xi_i^n = \frac{x_i^n}{\sum_{j\in S_n} x_j^n}$, $\forall (n,i) \in I_N$ is a Nash equilibrium of Γ^N.

(2) If ξ is a mixed Nash equilibrium of Γ^N, x defined as

$$x_i^n = \left(\frac{\prod_{\nu\neq n} A^\nu(\xi)}{A^n(\xi)^{N-2}}\right)^{\frac{-1}{N-1}} \xi_i^n, \forall (n,i) \in I_N \text{ is a solution of } \mathcal{S}^N.$$

A solution $x \in E^\pi$ of a PCP is called a *complementary point*. A point $x \in E^\pi$ is called a (n,i)-*almost complementary point* if it satisfies every constraints in $\mathcal{S}^N$, *except possibly*[3] for the equation $x_i^n \cdot \left(A_i^n(x^{-n}) - 1\right) = 0$.

In the view of this result, finding a mixed Nash equilibrium of a game Γ^N amounts to finding a solution to the PCP $\mathcal{S}^N$.

2.2 Solving PCP through support enumeration

[21] suggested a simple support enumeration method to find a Nash equilibrium of a game, which can be described very simply in terms of PCP solution. This approach consists in enumerating every set of joint supports $W \subseteq I_N$ (such that W contains at least one pair $(n,i), \forall n \in P$) and trying to solve a system of equations/inequations, which we write $\mathcal{S}^{\overline{W},W}$ for a reason which will be made clear in the following section. $\mathcal{S}^{\overline{W},W}$ is equal to the system $\mathcal{S}^N$ where the equalities are replaced with two distinct sets of equalities:

$$\begin{cases} x \in \mathcal{D}^N, \\ x_i^n = 0, \quad \forall (n,i) \in \overline{W}, \\ A_i^n(x^{-n}) = 1, \quad \forall (n,i) \in W \end{cases} \qquad (\mathcal{S}^{\overline{W},W})$$

A solution x of system $\mathcal{S}^{\overline{W},W}$, if it exists, will correspond to an equilibrium ξ. However, there are exactly $\prod_{i=1}^N \left(2^{|S_i|} - 1\right)$ possible supports $W \subseteq I_N$, so trying to solve a system of polynomial equations/inequations for every W is not a good idea. [21] suggest to enumerate supports in increasing size of W, in an order compatible with inclusion (if $W \subset W'$ W' will be explored after W). As soon as a system $\mathcal{S}^{\overline{W},W}$ with a solution is found, the algorithm exits and returns the corresponding Nash equilibrium. *Iterated Removal of Dominated Alternatives* [9] is applied before any new system $\mathcal{S}^{\overline{W},W}$ is actually solved. If any alternative is removed, this means that we can reduce support W to $W'' \subset W$. However, given the enumeration order, the algorithm has already tried to solve $\mathcal{S}^{\overline{W''},W''}$ and failed. So, W is skipped before solving $\mathcal{S}^{\overline{W},W}$. The algorithm is guaranteed to find a mixed Nash equilibrium of the game (or a continuum of solutions if a degenerate system $\mathcal{S}^{\overline{W},W}$ is encountered).

Support enumeration is quite efficient in general, especially when an equilibrium with support of small size exists. However, for games with no NE of small support size, many joint supports may be explored and many systems of equations may have to be solved. Next, we describe an alternative approach to [21] to solve polynomial complementarity problems. This approach is inspired from [23]'s method, itself extending Lemke-Howson's algorithm [17]. The idea is, instead of enumerating blindly all possible pairs $(\overline{W}, W)$ until we find a solvable system, to solve an ordered sequence of systems $\mathcal{S}^{Z,W}$, where Z may be different from $\overline{W}$ and the union $Z \cup W$ may be strictly included in I_N. This approach generally explores fewer pairs (Z,W) than [21] but cannot exploit IRDA, so a larger proportion of systems may have to be solved.

[3] Thus, a complementary point is (n,i)-almost complementary for every pair (n,i).

3 A COMBINATORIAL PATH-FOLLOWING ALGORITHM TO SOLVE PCP

Wilson [23], following the approach proposed by [17] to solve linear complementarity problems, proposed a mathematical *path following* approach to solve a *non-degenerate* PCP. Wilson's description leaves some steps of the algorithm undefined and does not deal with degenerate games. In this Section, we propose an original and operational rewriting of Wilson's approach.

The approach we propose is based on a definition of (almost-complementary) nodes, arcs and paths, in terms of sets of multilinear equations - we detail these definitions in Section 3.1. Then, we show how paths can be extended through different levels of sub-PCP (Sections 3.2 and 3.3). Section 3.4 is devoted to the arc-traversal problem and Section 3.5 describes the full PCP solution algorithm.

3.1 Almost-complementary nodes, arcs and paths

Assume that PCP $\mathcal{S}^N$ is given. For any $x \in \mathcal{D}^N$, let us write $Z(x) = \{(n,i) \in I_N, x_i^n = 0\}$ and $W(x) = \{(n,i) \in I_N, A_i^n(x^{-n}) = 1\}$. Then, by definition, $x \in \mathcal{D}^N$ is a solution of $\mathcal{S}^N$ if and only if $Z(x) \cup W(x) = I_N$. A *non-degenerate PCP at level N* is defined as:

Definition 3.1 (Non-degenerate PCP). PCP $\mathcal{S}^N$ is non-degenerate iff the following conditions hold:

(1) No point in $\mathcal{D}^N$ satisfies more than $|I_N|$ equations.
(2) No two distinct points satisfy the same set of $|I_N|$ equations.

In mathematical terms, condition 1 is equivalent to

$$\forall x \in \mathcal{D}^N, |Z(x)| + |W(x)| \leq D = |I_N|$$

and condition 2 is equivalent to: $\forall x, y \in \mathcal{D}^N$,

$$\left.\begin{array}{rcl} Z(x) = Z(y) & = & Z \\ W(x) = W(y) & = & W \\ |Z| + |W| & = & |I_N| \end{array}\right\} \Rightarrow x = y \qquad (4)$$

Intuitively, a PCP is non-degenerate if none of its polynomial constraints are redundant (it involves $|I_N|$ independent equations in a space of dimension $|I_N|$). For the moment, we assume non-degeneracy. Considering a non-degenerate PCP allows to distinguish particular points of interest, *almost complementary nodes*:

Definition 3.2 (Almost-complementary nodes). $x \in \mathcal{D}^N$ is an almost-complementary node of $\mathcal{S}^N$, if and only if:

$$|Z(x)| + |W(x)| = |I_N| \text{ and } |Z(x) \cap W(x)| \leq 1$$

In particular, an almost-complementary node is said complementary if $Z(x) \cup W(x) = I_N$ and (n,i)-almost complementary if $(n,i) \notin Z(x) \cup W(x)$. Note that if $\mathcal{S}^N$ is non-degenerate, the almost-complementary nodes of $\mathcal{D}^N$ are in one-to-one correspondence with the pairs $Z, W \subseteq I_N$ such that $|Z| + |W| = |I_N|$, $|Z \cap W| \leq 1$ and such that the system:

$$\left\{\begin{array}{rl} & x \in \mathcal{D}^N, \\ x_i^n = 0, & \forall (n,i) \in Z, \\ A_i^n(x^{-n}) = 1, & \forall (n,i) \in W \end{array}\right. \qquad (\mathcal{S}^{Z,W})$$

has a solution. In the following we identify (n,i)-almost complementary node x (denoted $\rho(Z,W)$) with the pair (Z,W) which defines the system $\mathcal{S}^{Z,W}$ that x solves.

We also define (n,i)-almost complementary arcs.

Definition 3.3 (Almost complementary arcs). The (n,i)-almost complementary arcs of non-degenerate PCP $\mathcal{S}^N$ are the subsets $\gamma(Z,W) \subseteq \mathcal{D}^N$, for all pairs $(Z,W) \subseteq I_N$ such that $Z \cap W = \emptyset$ and $Z \cup W = I_N \setminus \{(n,i)\}$, where $\gamma(Z,W)$ is formed by the points $x \in \mathcal{D}^N$ such that $Z(x) = Z$ and $W(x) = W$.

If almost complementary arc $\gamma(Z,W)$ is non-empty and the game is non degenerate, $\gamma(Z,W)$ is included in the set of solutions of a system of $D-1$ equations over D variables. Non-degeneracy implies that this set has dimension 1 and can be parameterized by a single real-valued parameter. The *extreme* points of $\gamma(Z,W)$, belonging to the frontier of domain $\mathcal{D}^N$ are almost complementary nodes. An arc will typically have two extreme points (if bounded) or a single one if unbounded.

Wilson's approach consists in following a one-dimensional path in $\mathcal{D}^N$, by traversing arcs and nodes, until we eventually reach a complementary node. It relies on the following proposition[4]:

Proposition 3.4 (Arcs neighbouring nodes). *Let $\mathcal{S}^N$ be a non-degenerate PCP and $i \in S_N$. Two (N,i)-almost complementary arcs neighbour a (N,i)-almost-complementary node of $\mathcal{S}^N$ which is not complementary and a single (N,i)-almost complementary arc neighbours a complementary node.*

The two arcs neighbouring almost complementary node $\rho(Z,W)$ are $\gamma(Z \setminus W, W)$ and $\gamma(Z, W \setminus Z)$, as shown in the proof of Proposition 3.4. Remark that Proposition 3.4 implies the following proposition:

Proposition 3.5 (Finite path). *Let $(\mathcal{S}^N)$ be a non-degenerate PCP and $i \in S_N$. There is a unique non-directed path, made of a finite sequence of (N,i)-almost complementary nodes and arcs, which reaches any complementary node of $\mathcal{S}^N$.*

Proposition 3.5 does not tell us about the other end of the path. The path can end with another complementary node or with a (N,i)-almost complementary unbounded arc which neighbours a single (N,i)-almost complementary node. Let us see how this unique non-directed path can be extended through different "layers" of sub-PCP of the initial PCP, until it reaches an "easy-to-compute" initial node.

3.2 Defining a sequence of PCP

Let us consider a PCP $\mathcal{S}^N$, an arbitrary pure joint strategy $\omega^0 = (\omega_1^0, \ldots, \omega_N^0) \in \pi$, two integers $1 \leq n \leq k \leq N$ and a pure strategy $i \in S_n$. We write $A_i^{n,k}\left(x^{\{1,..,k\}\setminus\{n\}}\right)$ the multivariate polynomial obtained from $A_i^n(x^{-n})$ by fixing all values x_j^ν to 0 whenever $\nu > k$ and $j \neq \omega_\nu^0$ and to 1 whenever $\nu > k$ and $j = \omega_\nu^0$. Then,

$$A_i^{n,k}\left(x^{\{1,..,k\}\setminus\{n\}}\right) = \sum_{\substack{\omega \in \pi, \omega_n = i \\ \omega_m = \omega_m^0, \forall m > k}} a_\omega^n \prod_{\substack{\nu \leq k, \\ \nu \neq n}} x_{\omega_\nu}^\nu. \qquad (5)$$

$A_i^{n,k}$ is a multilinear polynomial of degree $k-1$. Note that if ξ is a joint mixed strategy, $A_i^{n,k}\left(\xi^{\{1,..,k\}\setminus\{n\}}\right)$ is the expected disutility

[4]The proofs of this Section are modern rewritings of the proofs of Wilson's original paper. The reader will find them, as well as a tutorial example and additional experiments in https://doi.org/10.5281/zenodo.5850463.

of player n playing action $i \in S_n$ when players $1, \ldots, k$ except n play their mixed strategy in ξ, while players $k+1, \ldots, N$ play their pure strategy in ω^0. For any $1 \leq k \leq N$, let us define $I_k = \{(n,i), 1 \leq n \leq k, i \in S_n\}$. We also let E_k^π denote the Euclidian space spanned by the variables x_i^n with $(n,i) \in I_k$. The dimension of E_k^π is $D_k = |I_k|$ (thus $D = D_N$). Then, from PCP $\mathcal{S}^N$, we can define the following sequence of sub-PCP $\mathcal{S}^k$, for $k = 1, \ldots, N-1$:

Definition 3.6 (Sub-PCP). Let $\mathcal{S}^N$ be a PCP and ω^0 a fixed pure joint strategy. Then, for $2 \leq k < N$, we define sub-PCP $\mathcal{S}^k$ as the following system of polynomial equations/inequations over E_k^π:

$$\forall (n,i) \in I_k, \begin{cases} x_i^n & \geq & 0 \\ A_i^{n,k}\left(x^{\{1,..,k\}\setminus\{n\}}\right) & \geq & 1 \\ x_i^n \cdot \left(A_i^{n,k}\left(x^{\{1,..,k\}\setminus\{n\}}\right) - 1\right) & = & 0 \end{cases} \quad \left(\mathcal{S}^k\right)$$

For $n = 1$, PCP $\mathcal{S}^1$ is slightly different, although deduced from best responses principles as well:

$$\begin{cases} x_i^1 & \geq & 0, \forall i \in S_1 \\ x_i^1 \cdot \left(\frac{a^1_{(i,\omega^0_{-1})}}{\min_{j\in S_1} a^1_{(j,\omega^0_{-1})}} - 1\right) & = & 0, \forall i \in S_1 \\ \sum_{i\in S_1} x_i^1 & = & 1 \end{cases} \quad \left(\mathcal{S}^1\right)$$

Sub-PCP $\mathcal{S}^k$ is constructed from $\Gamma^k(\omega^0)$, the game played by the first k players of Γ^N when the remaining players play following ω^0, in the same way as $\mathcal{S}^N$ was constructed from Γ^N. As for the initial PCP, we assume that all sub-PCP $\mathcal{S}^k$ are non-degenerate. In particular, assuming that $\mathcal{S}^1$ is non-degenerate implies that the minimum $\min_{j\in S_1} a^1_{(j,\omega^0_{-1})}$ is attained for a single index j^*. Thus, we easily get the complementary point at level 1, characterized by $Z^1 = S_1 \setminus \{j^*\}$ and $W^1 = \{j^*\}$. Then, we can define the polynomial systems $\mathcal{S}_k^{Z,W}$ for any pair $Z, W \subseteq I_k$:

$$\begin{cases} & x \in \mathcal{D}^k, \\ x_i^n = 0, & \forall (n,i) \in Z, \\ A_i^{n,k}(x^{\{1..k\}\setminus\{n\}}) = 1, & \forall (n,i) \in W, \end{cases} \quad \left(\mathcal{S}_k^{Z,W}\right)$$

where $\mathcal{D}^k$ is defined by the inequations of Definition 3.6.

3.3 Complementary nodes and initial nodes

We can now exploit the combinatorial point of view of the above sequence of systems of equations in order to design an algorithm computing a sequence of almost-complementary nodes until a complementary node at level N is reached.

Proposition 3.7 (Complementary node lifting). *Let $\mathcal{S}^k$ be a sub-PCP of $\mathcal{S}^N$ at level $1 \leq k < N$ and arbitrary pure joint strategy ω^0. Assume that, for $Z, W \subseteq I_k$, $\mathcal{S}_k^{Z,W}$ defines a complementary node of $\mathcal{S}^k$ in E_k^π. Then, $\mathcal{S}_{k+1}^{Z',W}$ defines a $(k+1, \omega^0_{k+1})$-almost complementary arc of sub-PCP $\mathcal{S}^{k+1}$, where $Z' = Z \cup \left\{(k+1, j), j \neq \omega^0_{k+1}\right\}$. Furthermore, this arc neighbours a single $(k+1, \omega^0_{k+1})$-almost complementary node at level $k+1$ (it is an unbounded arc).*

The $(k+1, \omega^0_{k+1})$-almost complementary node at level $k+1$ can be computed by trying to solve all systems $\mathcal{S}_{k+1}^{Z'\cup\{(\nu,j)\},W}$ with $(\nu, j) \in I_k \setminus Z'$ and $\mathcal{S}_{k+1}^{Z',W\cup\{(\nu,j)\}}$ with $(\nu, j) \in I_{k+1} \setminus W$ until we find a solution. Such a system with a solution exists (this is a consequence of Lemma 2 in [23]). Furthermore, it is unique, due to non-degeneracy. This solution is called *initial node* at level $k+1$.

Definition 3.8 (Initial node). An initial node at level $k+1$ is a $(k+1, \omega^0_{k+1})$-almost complementary node, solution to $\mathcal{S}_{k+1}^{Z,W}$, with $Z, W \subseteq I_{k+1}$, such that only one of its neighbouring arcs is bounded. It also satisfies: $(k+1, \omega^0_{k+1}) \notin Z$ and $(k+1, j) \in Z, \forall j \neq \omega^0_{k+1}$.

With this characterization in mind, Proposition 3.5 can be reinterpreted. It states that from any complementary node at level N there is a unique path, leading either to another complementary node, or to an *initial node* at level N. This is obviously true at any level $k \in \{2, \ldots, N\}$, since sub-PCP are derived from subgames. Complementary nodes are at one end of a path of (k, ω^0_k)-almost complementary points which other end is either another complementary node or an initial node. The set of almost-complementary points in E_k^π which satisfy $\mathcal{S}^k$ form disjoint paths (see Figure 1).

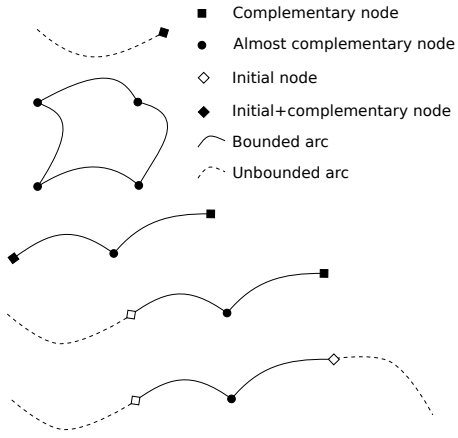

Figure 1: Example almost-complementary paths at level k.

Then, we can define a *descent* procedure from an initial node at level k to a complementary node at level $k-1$, reciprocal to the previous lifting procedure.

Proposition 3.9 (Initial node descent). *Let $\mathcal{S}_k^{Z,W}$ define an initial node at level $k > 1$. Let $Z' = Z \cap I_{k-1}$ and $W' = W \cap I_{k-1}$. Then, either $Z' \cap W' = \{(\nu, j)\}$ or $Z' \cap W' = \emptyset$.*

In the first case, either $\mathcal{S}_{k-1}^{Z'\setminus\{(\nu,j)\},W'}$ or $\mathcal{S}_{k-1}^{Z',W'\setminus\{(\nu,j)\}}$ defines a complementary node at level $k-1$. In the second case, $\mathcal{S}_{k-1}^{Z',W'}$ defines a complementary node.

3.4 Algebraic arc traversal

Our path-following algorithm heavily relies on the problem of *traversing* a (k, ω^0_k)-almost complementary arc[5] $\gamma^k(Z, W)$, at level k ($Z, W \subseteq I_k$), that leaves an almost-complementary node $\rho^k(Z', W')$ with either (i) $Z = Z' \setminus W'$ and $W = W'$ or (ii) $Z = Z'$ and $W = W' \setminus Z'$. This arc traversal problem can be stated in algebraic terms. First, remark that, by definition:

[5]The exponent k in γ^k or ρ^k indicates that we are at level k.

$\gamma^k(Z,W) = \mathcal{V}\left(\mathcal{S}_k^{Z,W}\right) \cap \mathcal{D}^k$ and $\rho^k(Z',W') = \mathcal{V}\left(\mathcal{S}_k^{Z',W'}\right) \cap \mathcal{D}^k$, where $\mathcal{V}(\mathcal{S})$ is the set of solutions of $(\mathcal{S})$, ignoring the domain constraint. In algebraic terms, $\mathcal{V}(\mathcal{S})$ is called an *affine variety* [5]. When the system $(\mathcal{S})$ is non-degenerate, the variety is a single point for a node and has dimension 1 for an arc. The systems can be solved approximately numerically using off-the-shelf solvers. Instead, we use an "exact" algebraic solver, based on Groebner basis computation. This kind of solver may be slower in practice and may limit the size of games which can be solved. However, it provides guaranteed "exact" finite representations of node coordinates and is especially useful in the case of degenerate games. We will discuss in the conclusion how to adapt our approach in order to use faster, approximate solvers. In our implementation, we use the functions implemented in the `Singular` toolbox, accessible from the `Sagemath` environment[6], to compute Groebner bases and varieties.

The arc traversal problem at level k consists, given the two pairs (Z',W') and (Z,W), in computing $(Z'',W'') \in I_k$ corresponding to the following (n,i)-almost complementary node $\rho^k(Z'',W'')$. Algorithm 1 computes this almost-complementary node by trying every possible constraint additions[7].

Algorithm 1: TraverseArc($(Z,W),(Z',W'),I_k$).

```
/* Computes the end node of arc γ^k(Z,W),
   given starting almost-complementary node
   ρ^k(Z',W').                                   */
/* Initialization                                */
1  Sol ← ∅;
2  for (v,j) ∈ I_k do
3      if (v,j) ∈ I_k \ Z' then
4          Z_loc ← Z ∪ {(v,j)}, W_loc ← W;
5          if dim(V(S_k^{Z_loc,W_loc})) = 0 then
6              ρ_loc ← V(S_k^{Z_loc,W_loc}) ∩ D^k;
7              if ρ_loc ≠ ∅ then
                   Sol ← Sol ∪ {(Z_loc, W_loc, ρ_loc)};
8      if (v,j) ∈ I_k \ W' then
9          Z_loc ← Z, W_loc ← W ∪ {(v,j)};
10         if dim(V(S_k^{Z_loc,W_loc})) = 0 then
11             ρ_loc ← V(S_k^{Z_loc,W_loc}) ∩ D^k;
12             if ρ_loc ≠ ∅ then
                   Sol ← Sol ∪ {(Z_loc, W_loc, ρ_loc)};
13 return Sol
```

When the game is non-degenerate, a unique constraint addition will lead to an almost-complementary node:

Proposition 3.10 (Arc traversal correctness). *When the PCP is non-degenerate, Algorithm 1 returns a single triple* $\left(Z'',W'',\mathcal{V}\left(\mathcal{S}_k^{Z'',W''}\right)\right)$ *and* $\mathcal{V}\left(\mathcal{S}_k^{Z'',W''}\right)$ *is a single point.*

[6] `https://www.sagemath.org/index.html`.

[7] In practice, as soon as a solution ρ_{loc} is found in line 9 of Algorithm 1, the main loop is exited and it is tested whether the current node is degenerate, by checking whether additional constraints indexed in Z or W are satisfied by the coordinates of ρ_{loc}.

3.5 The path-following procedure

Now, we have nearly all the elements necessary to build a path-following procedure to reach a complementary node of a PCP. We only lack an initialization procedure. This procedure uses the arbitrary pure joint strategy ω^0 and consists in solving $(\mathcal{S}^1)$, decribed in Definition 3.6. Once we have this complementary node at level 1, we compute the corresponding initial node at level 2, using Proposition 3.7 and follow a path at level 2, starting from this node. If we reach a complementary node at level 2, we climb to level 3, etc. If, at some level k, we reach an initial node, then we compute a new complementary node at level $k-1$, using node descent (Proposition 3.9), from which we go on. In the case where the game is non-degenerate, since we start from a complementary node at level 1 and since the property that every node except the initial node at level 1 and complementary node at level N has exactly two neighbours, the path followed is unique, for a given ω^0. Furthermore, it can only end in a complementary node at level N: A solution of the PCP/game. Figure 2 illustrates a few steps of the algorithm.

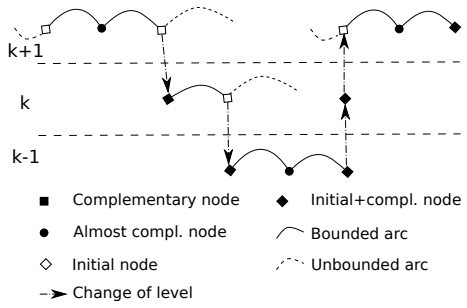

Figure 2: Portion of a path followed by the algorithm.

4 HANDLING DEGENERATE PCP

Recall (Definition 3.1) that there are two ways in which a PCP at level k can be degenerate: (i) if there are points satisfying more than $|I_k|$ equations and (ii) if some systems admit more than one solution. In the first case, which is far more frequent than the second, things may go wrong with the arc traversal algorithm (Algorithm 1). When we follow, during arc traversal or node lifting, an arc $\gamma^k(Z,W)$, we normally encounter a unique feasible almost complementary node, either $\rho^k(Z \cup \{z\},W)$ or $\rho^k(Z,W \cup \{w\})$. However, for some PCP, it may happen that an arc-traversal makes more than one new constraint binding. For example, there may exist w_1 and w_2, such that $\rho^k(Z,W \cup \{w_1\})$ and $\rho^k(Z,W \cup \{w_2\})$ define the same feasible solution (identical coordinates). Such a node is degenerate. While traversal of non-degenerate nodes is unambiguous, it is ambiguous for degenerate nodes, since they neighbour three or more almost-complementary arcs. In this case, an arbitrary choice of the next arc may lead the algorithm to cycle. Figure 3 shows an example game with some degenerate nodes, and paths linking them.

In this example, nodes 1, 2, 3 and 4 are degenerate. The corresponding graph contains cycles ($\{1,2,3\}$, $\{2,3,4\}$, $\{1,2,4,3\}$). However, this graph always contains at least one path from the initial

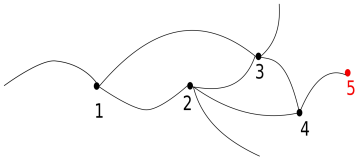

Figure 3: A PCP where four degenerate almost-complementary nodes are encountered. Node 5 is a complementary node.

node at level 0 toward a complementary node at level N[8]. Taking this fact into account, we may alter the path-following algorithm by applying depth-first search over arcs issued from degenerate nodes, as shown in Algorithm 2.

Algorithm 2: Depth-First Graph Traversal

Data: A PCP and a first degenerate almost-complementary node, $firstdegennode$
Result: A complementary node

$listofarcs \leftarrow \emptyset$;
$listofarcs.append(firstdegennode.outgoingarcs)$;
while $listofarcs \neq \emptyset$ **do**
  $currentarc \leftarrow listofarcs.pop()$;
  $nextnode \leftarrow traversepath(currentarc)$;
  if $nextnode.is\_complementary$ **then break;**
  if $nextnode = \emptyset$ **then** $currentarc \leftarrow listofarcs.pop()$;
  else $listofarcs.append(nextnode.outgoingarcs)$;
return $nextnode$

The procedure $traversepath()$ performs node traversal along a path of non-degenerate nodes, until either (i) the complementary node is reached, (ii) a new degenerate node is reached or (iii) a degenerate node which has already been explored is reached. In cases (i) and (ii) the variable $nextnode$ is set to the corresponding node, while in case (iii) it takes value $\emptyset$.

Applying Algorithm 2 to the example of Figure 3, assuming that we append degenerate nodes' outgoing arcs from top to bottom in $listofarcs$ (different heuristics may be used to decide in which order arcs should be appended), we get the following sequence of lists of arcs: $\{(1, up), (1, down)\} \rightarrow \{(1, up), (2, up), (2, middle), (2, bottom)\}$ $\rightarrow \{(1, up), (2, up), (2, middle)\} \rightarrow \{(1, up), (2, up), (4, up), (4, bottom)\}$ $\rightarrow$ Node 5 is returned.

Note that in the case of LCP (i.e. for bimatrix games or polymatrix games), a heuristic to order the arcs to append to the list is known that allows Algorithm 2 to never backtrack [22]. This heuristic, based on a lexicographic perturbation of linear systems, is useful since it avoids unnecessary arc traversals, which are costly. The same kind of approach is likely to be applicable to PCP as well. However, computing the heuristic for PCP would require to solve polynomial systems, so the overall benefit is unclear. We leave the design of a lexicographic approach for PCP and its evaluation for further research. We should also mention the second case of degeneracy, corresponding to the case where some $\mathcal{V}\left(\mathcal{S}_k^{Z,W}\right)$ encountered in the course of the algorithm is a variety of strictly positive dimension (an arc, an hypersurface...) because its defining equations are redundant. While the use of a Groebner basis approach allows to exactly represent such positive-dimensional varieties in parameterized form, it is not clear how such varieties should be traversed in the course of our procedure[9]. We defer the design of a non-zero dimensional varieties traversal method to further work. Instead, what we suggest to do in this case of degeneracy is to apply a small (non-infinitesimal) random perturbation to the coefficients of the initial PCP and solve this new perturbed PCP. The perturbed PCP will generally not be degenerate. So doing, we obtain an approximate Nash equilibrium when this form of degeneracy occurs.

[8]This is a consequence of the fact that a random infinitesimal perturbation of the coefficients of the PCP will resolve degeneracy by making some of the binding constraints non-binding. In the process, the new non-degenerate nodes will be defined by pairs (Z', W') where $Z' \subseteq Z$ and $W' \subseteq W$.

5 GRAPHICAL/HYPERGRAPHICAL GAMES

The PCP approach extends naturally to polymatrix games [24], graphical games [16] and hypergraphical games [20]. These are succinct representations of N-player games where the utilities of the players are local, i.e. depend on the strategies of subsets of P only. Hypergraphical games are defined as follows:

Definition 5.1 (Hypergraphical game). A N-player hypergraphical game, Γ^N, is defined as:

$$\Gamma^N = \left((P_g)_{g=1,\dots,G}, (S_n)_{n\in P}, \left(a^g = (a^{g,n}_{\omega_{P_g}})_{n\in P_g, \omega_{P_g}\in\pi_{P_g}}\right)_{g=1,\dots,G}\right).$$

- $P_g \subseteq P = \{1,\dots,N\}, \forall g \in 1,\dots,G$. P_g is the set of players of the g^{th} local game and $\cup_{g=1,\dots,G} P_g = P$.
- $(S_n)_{n\in P}$ is the list of players pure strategies sets.
- $a^{g,n}_{\omega_{P_g}}$ is the (positive) disutility that player n gets in local game number g (provided that n belongs to P_g), when the joint strategy of all players in game g is ω_{P_g}.

The disutility of player $n \in P$ for joint strategy ω is $a^n_\omega = \sum_{g,n\in P_g} a^{g,n}_{\omega_{P_g}}$.

Polymatrix games and graphical games are specific cases of hypergraphical games. Polymatrix games are characterized by the fact that any local game g involves exactly two players: $|P_g| = 2, \forall g = 1,\dots,G$. Graphical games are hypergraphical games where the utility of any player $n \in P$ only depends on the strategies of a subset $P_n \in P$ of players. They are characterized by the fact that $G = N$ (there is one local game attached to each player) and that $a^{g,n}_{\omega_{P_g}} = 0, \forall \omega \in \pi, \forall n \neq g$.

Since a hypergraphical game can be represented as a normal form game (potentially exponentially larger to express), NE computation in hypergraphical games admits an exponential size PCP formulation. Fortunately, we can exploit the factorisation of the disutility functions of a hypergraphical game in order to compute a corresponding PCP of "reasonable" size. First, we can show that:

Proposition 5.2 (PCP factorization).

$$A^n_i(x^{-n}) = \sum_{g,n\in g} Q_g\left(x^{P\setminus P_g}\right) R_{n,i,g}\left(x^{P_g\setminus\{n\}}\right), \text{ where}$$

[9]When such varieties correspond to sets of complementary points at level N, we can directly provide the continuous set of equilibrium strategies in parameterized form.

$$Q_g\left(x^{P\setminus P_g}\right) = \prod_{\nu\in P\setminus P_g}\left(\sum_{\omega_\nu\in S_\nu} x^\nu_{\omega_\nu}\right) \text{ and}$$
$$R_{n,i,g}\left(x^{P_g\setminus\{n\}}\right) = \sum_{\substack{\omega_{P_g}\\ \omega_n=i}} a^{g,n}_{\omega_{P_g}} \prod_{\nu\in P_g\setminus\{n\}} x^\nu_{\omega_\nu}.$$

Proof of Proposition 5.2: The disutility of player $n \in N$ for joint strategy ω is:

$$a^n_\omega = \sum_{g,n\in P_g} a^{g,n}_{\omega_{P_g}}$$

Using this expression of a^n_ω, we can rewrite $A^n_i(x^{-n})$:

$$\begin{aligned}
A^n_i(x^{-n}) &= \sum_{\substack{\omega\in\pi\\ \omega_n=i}}\left(\sum_{g,n\in P_g} a^{g,n}_{\omega_{P_g}}\right)\prod_{\nu\neq n} x^\nu_{\omega_\nu}, \forall n\in P, i\in S_n,\\
&= \sum_{g,n\in P_g}\sum_{\substack{\omega\in\pi\\ \omega_n=i}} a^{g,n}_{\omega_{P_g}}\prod_{\nu\neq n} x^\nu_{\omega_\nu},\\
&= \sum_{g,n\in P_g}\sum_{\substack{\omega_{P_g}\\ \omega_n=i}} a^{g,n}_{\omega_{P_g}}\left(\prod_{\nu\in P_g\setminus\{n\}} x^\nu_{\omega_\nu}\right)\sum_{\omega_{P\setminus P_g}}\prod_{\nu\in P\setminus P_g} x^\nu_{\omega_\nu},\\
&= \sum_{g,n\in P_g}\sum_{\substack{\omega_{P_g}\\ \omega_n=i}} a^{g,n}_{\omega_{P_g}}\left(\prod_{\nu\in P_g\setminus\{n\}} x^\nu_{\omega_\nu}\right)\left(\prod_{\nu\in P\setminus P_g}\left(\sum_{\omega_\nu\in S_\nu} x^\nu_{\omega_\nu}\right)\right).
\end{aligned}$$

Remark that equality $\sum_{\omega_{P\setminus P_g}}\prod_{\nu\in P\setminus P_g} x^\nu_{\omega_\nu} = \prod_{\nu\in P\setminus P_g}\left(\sum_{\omega_\nu\in S_\nu} x^\nu_{\omega_\nu}\right)$ results from the repeated application of the distributivity of the product over the addition. The proposition follows. □

Note that $A^n_i(x^{-n})$ is still a polynomial in variables x^m_j, $(m,j)\neq(n,i)$ with a number of terms of degree $N-1$ equal to $|\pi_{-n}|$. But it is known, for example, that polymatrix games admit a *linear complementarity problem* formulation [14]. We show that it is possible to get similar savings in representation size and solution complexity for hypergraphical games, through the addition of auxiliary variables.

Definition 5.3 (PCP auxiliary variables). In addition to the variables $\{x^n_i\}$, let us consider an additional set of variables, $\left\{y^n_g\right\}_{g=1..G,n\in P}$.

We define these variables as:

$$y^n_g = \prod_{\nu=1}^{n}\alpha^\nu_g, \text{ where } \alpha^\nu_g = 1 \text{ if } \nu\in P_g \text{ and } \alpha^g_\nu = \sum_{\omega_\nu\in S_\nu} x^\nu_{\omega_\nu} \text{ if } \nu\in P\setminus P_g.$$

With this definition, we have $Q_g\left(x^{P\setminus P_g}\right) = y^N_g$ and

$$y^1_g = \alpha^1_g \text{ and } y^n_g = y^{n-1}_g\times\alpha^n_g, \forall n = 2,\ldots,N. \quad (6)$$

Note that there are exactly $G\times N$ additional variables y^n_g and their values are defined by equations of degree 1 when $n\in P_g$ and degree 2 when $n\in P\setminus P_g$. This gives the following expression for A^n_i:

$$A^n_i\left(x^{-n},\left\{y^N_g\right\}_{g=1,..,G}\right) = \sum_{g,n\in P_g} y^N_g\sum_{\substack{\omega_{P_g}\\ \omega_n=i}} a^{g,n}_{\omega_{P_g}}\prod_{\nu\in P_g\setminus\{n\}} x^\nu_{\omega_\nu}. \quad (7)$$

Remark that the polynomials $A^n_i\left(x^{-n},\left\{y^N_g\right\}_{g=1,..,G}\right)$ now have degree at most $max_{g=1,..,G}|P_g|$ and the size of the expression of these polynomials is comparable to that of the hypergraphical game.

Our algorithm can now be extended to deal with this new expression with additional variables. The only change in the algorithm is the definition of the polynomial subsystems which now have to consider the y^n_g variables. We extend the definition of the polynomial systems $\mathcal{S}^{Z,W}_k$ to $\overline{\mathcal{S}}^{Z,W}_k$ for pairs $Z,W\subseteq I_k$:

$$\begin{cases}
& x\in\mathcal{D}, \\
x^n_{\omega^0_n} = 1 \text{ and } x^n_i = 0, & \forall n>k, \forall i\neq\omega^0_n \\
x^n_i = 0, & \forall(n,i)\in Z, \\
y^1_g = \alpha^1_g, & \forall g = 1,\ldots,G \\
y^n_g = y^{n-1}_g\times\alpha^n_g, & \forall g = 1..G, \forall n = 2..k. \\
A^{n,k}_i\left(x^{\{1..k\}\setminus\{n\}},\left\{y^k_g\right\}_{g=1,..,G}\right) = 1, & \forall(n,i)\in W,
\end{cases}$$

With this generalization to hypergraphical games, it results that the number of terms of the polynomials of the systems is reduced and their degrees are bounded by the number of players of the largest subgame. This has an importance when it comes to computing Groebner bases. The *Ideal Membership Problem*, which is the core problem of the Groebner basis computation problem is known to be ExpSpace-complete. However, when the ideal has dimension 0 (which is the case when $\left(\overline{\mathcal{S}}^{Z,W}_k\right)$ has a finite number of solutions), it is solvable in single exponential time [7]. Furthermore, being able to upper bound the degrees of the polynomials involved in the PCP has a positive impact on the complexity of the algorithm:

Proposition 5.4 (Complexity of the path-following algorithm). *The time complexity of the path-following algorithm for a non-degenerate graphical/hypergraphical game is simply exponential in $|I_N|$ and doubly exponential in the maximal number of players of any local game.*

Proof of Proposition 5.4: Indeed, The worst-case time complexity of the best known Groebner basis computation algorithm is doubly exponential in the *maximal degree* of the involved polynomials (see e.g. [19]). The number of arc traversal steps is bounded by the number of almost complementary nodes at any level (with respect to a fixed ω_0. Thus, at level k it is bounded by $\#Nodes(k) = |\{(Z,W), |Z|+|W| = |I_k|, |Z\cap W|\leq 1\}|$. Then, remark that, for any hypergraphical game, including normal form games: $\#Nodes(k)\leq |I_k|2^{|I_k|}$. Indeed, potential nodes can be built by choosing an arbitrary $Z\subseteq I_k$ and then there exist at most $|I_k|$ W potentially leading to a (k,j)-almost complementary node. So, the total number of almost-complementary nodes for all levels is $O\left(N|I_N|2^{|I_N|}\right)$. Since each arc traversal requires at most $O(|I_N|)$ Groebner basis computations, we get the result[10]. □

6 EXPERIMENTS

The support enumeration and path-following algorithms were implemented in Python 3 using the Sagemath software for the computation of varieties. Each game was solved on a single node of a

[10]This complexity bound is theoretical. In practice the length of the paths is often small.

HPC platform. Nodes were identical Intel Xeon E5-2680 v4 2,4 Ghz bi-processors with 128 Gb of RAM, under Linux OS.

The benchmark was composed of normal-form and hypergraphical games with 3 to 7 players, 2 to 4 actions and hyperedges of size 3 (for HGG). They were generated using the GAMUT[11] suite. We generated *Covariant* and *Random* games with integer valued utilities in range 0..100. In total, 100 games were solved for each configuration, using support enumeration (SE) and path-following (PF). Hypergraphical games were converted to normal-form prior to being solved using support enumeration. Before applying SE or PF, pure equilibrium search was performed. After this step there remained from 10 to 40 unsolved games in each configuration. For these, we computed the average NE support size per player found using SE or PF, as well as the time required to solve them. A timeout of 5 minutes per run was used for SE and PF. Figure 4 shows the results obtained for random games with 5 players and 3 actions (3 local games of size 3 for hypergraphical games). All other tested configurations are shown in the supplementary material.

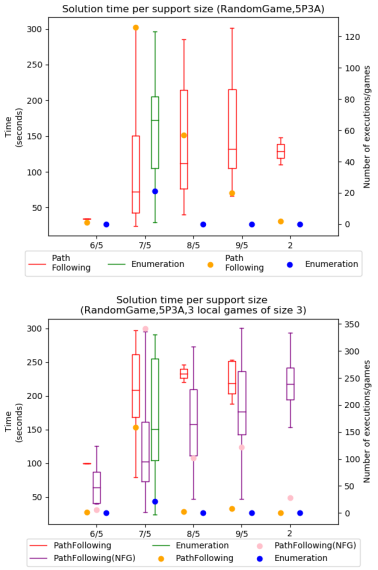

Figure 4: Experiment on 100 Random game. 66 games with PNE in NFG, 60 in HGG.

Figure 4, top, shows some features generally observed for all configurations. First, as can be expected, PF solutions will present a larger variety of support sizes than SE and larger in general. Still, it happens, rarely, that PF finds solutions of smaller support size than SE. This can (and does) happen in the case of degenerate games where the equilibrium with smallest support is at one end of an arc of solutions of dimension 1. SE will not find this solution while PF may encounter this complementary node. This is why PF finds equilibria of support size 6 when SE does not, in Figure 4.

Globally, we found that SE was more efficient than PF for games with no more than 6 players and 2 actions. PF is more efficient for games with at least 3 actions per player. With 7 players, 2 actions, both approaches have similar performances. It confirms that when the total number of potential supports increases, Wilson's approach becomes more efficient than blind enumeration and removal of dominated alternatives. Similar conclusions hold for random and covariant games. As far as HGG are concerned, we found that the direct PF approach did not outperform a prior translation to NFG. In the study cases, it looks like the cost of adding variables y is not compensated by the decreased polynomial degrees and number of terms. Studying whether it is possible to improve the PF approach's performance in case of HGG is left for further research.

[11] http://gamut.stanford.edu/

7 CONCLUDING REMARKS

There is an abundant literature on algorithms for approximate equilibrium search in N-player games, including *homotopy methods* [3, 10, 11, 13]. These are based on the definition of a parametrized continuum of games, joining an arbitrary game with known equilibrium, to the game of interest. An "arc" of equilibria of the parametrized games is followed and [6] have used Groebner bases to solve the specifically designed (easy) initial game. Homotopy approaches are prone to numerical errors and potential non-convergence. Uniform strategy enumeration methods [1, 2, 18], on the other hand, suggest to enumerate a space of discretized mixed strategies in order to find a ε-approximate strategy. Polynomial systems-based approaches, such as [21, 23] also rely on a form of enumeration. [21] enumerates and solves a sequence of systems formed from mixed strategies supports of increasing size, when [23] explores a deterministic sequence of systems. We provided the first implementation of [23], extended to degenerate and succinct games. Our approach explores a sequence of neighbour almost-complementary points defined as solutions of systems of equations at different levels. It differs from [21] which blindly explores every possible systems in increasing order of supports' sizes. When there exists a NE with small support, the method [21] is faster, while in other cases [23] is more efficient.

The building block of PCP methods, the polynomial system solver, can be easily adapted to follow the progresses in solvers' development. We used an exact algebraic solver. So doing, node's coordinates are *algebraic numbers* [4] that is, roots of polynomials which coefficients are either rational numbers, or algebraic numbers themselves. Thus they are finitely exactly represented. While exactness is desirable in Wilson's algorithm in order to distinguish degenerate from non-degenerate nodes, it comes at a computational cost. Wilson's arc traversal procedure can be adapted to use approximate solvers, by using depth-first search in the way we use it for dealing with degeneracy. It is enough to relax the condition that a node is degenerate whenever it corresponds to points $\rho^k(Z, W)$ and $\rho^k(Z', W')$ of identical coordinates: We may consider that (Z, W) and (Z', W') correspond to a single degenerate node whenever the coordinates of $\rho^k(Z, W)$ and $\rho^k(Z', W')$ are close enough. So doing, system solving will be faster, but the number of explored almost complementary nodes may increase, due to "false" degeneracy.

Finally, our approach may be naturally extended to other kinds of games, including Bayesian games [12] and stochastic games [8]. Indeed, it is known that two-player bayesian games can be represented as polymatrix games [15]. The same relation is likely to hold between N-player bayesian games and hypergraphical games, making a concise PCP formulation possible.

REFERENCES

[1] Y. Babichenko, S. Barman, and R. Peretz. 2014. Simple approximate equilibria in large games. In *EC'14*. ACM, Association for Computing Machinery, New York, NY, United States, 753–770.

[2] K. Berg and T. Sandholm. 2017. Exclusion method for finding Nash equilibrium in multiplayer games. In *AAAI'17*. AAAI Press, Palo Alto, CA, United States.

[3] B. Blum, D. Koller, and C.R. Shelton. 2006. A Continuation Method for Nash Equilibria in Structured Games. *Journal of Artificial Intelliegence Research* 25 (2006), 457–502.

[4] J. H. Conway and R. K. Guy. 1996. *The Book of Numbers*. Springer-Verlagr, New York, Chapter Algebraic Numbers, 189–190.

[5] D.A. Cox, J.B. Little, and D. O'Shea. 2015. *Ideals, Varieties, and Algorithms* (4 ed.). Springer, New York.

[6] R.S. Datta. 2010. Finding all Nash equilibria of a finite game using polynomial algebra. *Econonmics Theory* 42 (2010), 55–96.

[7] A. Dickenstein, N. Fitchas, M. Giusti, and C. Sessa. 1991. The membership problem for unmixed polynomial ideals is solvable in single exponential time. *Discrete Applied Mathematics* 33, 1 (1991), 73–94.

[8] J. Filar and K. Vrieze. 1997. *Competitive Markov Decision Processes*. Springer-Verlag, New York.

[9] Itzhak Gilboa, Ehud Kalai, and Eitan Zemel. 1989. The Complexity of Eliminating Dominated Strategies. *Mathematics of Operations Research* 18 (10 1989).

[10] S. Govindan and R. Wilson. 2003. A Global Newton Method to Compute Nash Equilibria. *Journal of Economic Theory* 110, 1 (2003), 65–86.

[11] S. Govindan and R. Wilson. 2004. Computing Nash Equilibria by Iterated Polymatrix Approximation. *Journal of Economic Dynamics and Control* 28, 7 (2004), 1229–1241.

[12] J.C. Harsanyi. 1967. Games with incomplete information played by "Bayesian" players, I–III Part I. The basic model. *Management science* 14, 3 (1967), 159–182.

[13] J.J. Herings and R. Peeters. 2010. Homotopy methods to compute equilibria in game theory. *Economic Theory* 42 (2010), 119–156.

[14] J.T. Howson. 1972. Equilibria of polymatrix games. *Management Science* 18, 5-part-1 (1972), 312–318.

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

[16] M. Kearns, M.L. Littman, and S. Singh. 2001. Graphical Models for Game Theory. *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI'01)* 1 (2001), 253–260.

[17] C.E. Lemke and J.T. Howson. 1964. Equilibrium points of bimatrix games. *Journal of the Society for industrial and Applied Mathematics* 12, 2 (1964), 413–423.

[18] R.J. Lipton, E. Markakis, and A. Mehta. 2003. Playing large games using simple strategies. In *EC'03*. ACM, Association for Computing Machinery, New York, NY, United States, 36–41.

[19] E. W. Mayr. 1997. Some Complexity Results for Polynomial Ideals. *Journal of Complexity* 13, 3 (1997), 303–325.

[20] C.H. Papadimitriou and T. Roughgarden. 2008. Computing correlated equilibria in multi-player games. *Journal of the ACM (JACM)* 55, 3 (2008), 14.

[21] R. Porter, E. Nudelman, and Y. Shoham. 2008. Simple search methods for finding a Nash equilibrium. *Games and Economic Behavior* 63, 2 (2008), 664–669.

[22] B. von Stengel. 2002. Computing equilibria for two-person games. In *Handbook of Game Theory with Economic Applications*, R. J. Auman and S. Hart (Eds.). Vol. 3. Elsevier, Amsterdam, Chapter 45, 1723–1759.

[23] R. Wilson. 1971. Computing equilibria of N-person games. *SIAM J. Appl. Math.* 21, 1 (1971), 80–87.

[24] E.B. Yanovskaya. 1968. Equilibrium points in polymatrix games. *Litovskii Matematicheskii Sbornik* 8 (1968), 381–384.