Chapter 5

Integer Payoff Irrational Equilibria Games

In this chapter, we present a method for computing all the Nash equilibria of an IPIE game. i.e., a game with integer payoffs and irrational equilibria. It differs from the method presented in previous chapter in two significant ways. Firstly, the polynomial algebra and Galois theory required earlier were over the field of rationals. In this chapter, we need to work with the generalized theory of ring extensions and Galois theory over rings. Secondly, instead of using Buchberger’s algorithm for identifying a sample solution, we use a numerical algorithm. We also discuss certain properties of the class of IPIE games. We conclude with an example and complexity analysis of the method.

5.1 Underlying Model

We again work with the game system $\mathcal{GS}$ of the form (2.7) as we did with RPIE games. Note that the coefficients of the $\mathcal{GS}$ now come from the ring of integers $\mathbb{Z}$. The solutions of the $\mathcal{GS}$ induce ring extensions rather than field extensions. Following the Galois theory over rings as given in Chase et al. [10], in this chapter we consider Galois ring extensions similar to those defined in 2.3.5. As was done in Algorithm 4.2.1 for RPIE games, the algorithm for IPIE games rejects unwanted solutions of the $\mathcal{GS}$ using suitable mechanisms.
5.2 Equilibria of IPIE Games

In this section, we present an algorithm for computing all Nash equilibria of IPIE games. The algorithm has two stages: in the first stage, compute a sample solution of the $GS$. Various methods for computing a sample solution are presented in [62]. In this work we use a version of the Multivariate Newton Raphson Method (MVNRM). In the second stage, apply the group action of the Galois group(s) to produce conjugate solutions of the sample solution. Finally, reject all non-equilibrium solutions from the set of solutions to obtain all the equilibria.

Recall that Nash [69] guarantees existence of at least one mixed strategy equilibrium (irrational equilibrium in our case). This implies that for each indeterminate variable, we are guaranteed to get a polynomial with an irreducible non-linear factor over the base ring $\mathbb{Z}$, since the input is an IPIE game.

Being an iterative procedure which converges to a solution, MVNRM starts with an initial guess. The mixed strategy Nash equilibria (probability tuples) form a subset of the set of solutions of $GS$. This allows choosing an initial guess of either all 0’s or 1’s or some value between $(0, 1)$. Appropriate choice of a solution tuple speeds up the convergence rate of MVNRM.

Next we have to convert the approximate solution given by MVNRM to algebraic form. Specifically, we need to confirm that the concerned exact solution has all irrational coordinates. For this purpose, we have to use the KLL algorithm, and consequently the stopping rule of the MVNRM iteration has to be tailored to meet the requirements of the KLL algorithm. The KLL algorithm constructs the minimal polynomial of an algebraic number given an approximation to a desired precision, and hence also determines whether the algebraic number is rational or irrational.

The precise calculations are as follows. Suppose that $y = (y_1, \ldots, y_{K^+})$ is the exact solution, while $\overline{y}_k = (y_{k1}, \ldots, y_{kK^+})$ is the approximate solution generated by MVNRM at the $k$-th iteration; as usual $|y|$ indicates the Euclidean norm of the
The KLL algorithm requires $O(d_i^2 + d_i \log H_i)$ bits of an approximate root for constructing the minimal polynomial of $x_i$, $i = 1, \ldots, K^+$ where $d_i$ is degree of the minimal polynomial of $x_i$ and $H_i$ is magnitude bound of the coefficient of the minimal polynomial of $y_i$. How to determine $d_i$ and $H_i$ is indicated in the following.

Let $g_i$ be the univariate polynomial of the variable $x_i$ in the reduced Gröbner basis of the $\mathcal{GS}$, obtained with a suitable lexicographic ordering. Then, since $y_i$ is a root of $g_i$, the minimal polynomial of $y_i$ is a factor of $g_i$, where

$$d_i \leq \deg g_i$$

and

$$H_i \leq M_i$$

where $M_i$ is the maximum magnitude of the coefficients of $g_i$.

Further degree of $g_i$ is bounded by number of solutions of the $\mathcal{GS}$ for which Bernstein [4] provides an upper bound.

**Proposition 7.** The MVNRM stage of the algorithm must be iterated until the number of zero bits in the binary representation of $|\bar{y}_{k+1} - \bar{y}_k|$ is bounded above by $O(d^2 + d \log H)$, where $d = \max d_i$ and $H = \max H_i, i = 1, 2 \ldots, K^+$.

**Proof.** If number of zero bits representing $|\bar{y}_{k+1} - \bar{y}_k|$ is bounded above by $O(d^2 + d \log H)$, then clearly, $O(d^2 + d \log H)$ bits of $y_i$ in $y$ are available, since

$$|y_i - y_k| \leq |y - \bar{y}_k| \leq K|\bar{y}_{k+1} - \bar{y}_k|,$$

where $K$ is a constant that depends on the rate of convergence of MVNRM (which is quadratic).

\[\]

\[1\]From Theorem 1.11 of [44] it is clear that we require $|y - \bar{y}_k| \leq 2^{-\left(d_i^2 + 3d_i + 4d_i \log_2 H_i\right)}$ for computing a minimal polynomial. A recent result [75] show a tighter bound on the required precision. A function for computing minimal polynomial of an approximate algebraic number is available in Maple and Mathematica softwares.
5.2. Equilibria of IPIE Games

At the end of the first stage, the algorithm generates a sample solution of the $\mathcal{GS}$.

With the sample solution available (either in algebraic form or in numerical form), in the next stage of the algorithm, we apply group action by Galois groups $G$. This stage does not differ from the corresponding stage of Algorithm 4.2.1 and so we make use of the Algorithm 4.2.3 presented in Chapter 4. For IPIE games, the Galois groups are associated with ring extensions over $\mathbb{Z}$, and they generate conjugate solutions of the sample solution of the $\mathcal{GS}$. The group action is transitive and produces a single orbit for each indeterminate variable. Using all the orbits we can determine all the irrational solutions of the $\mathcal{GS}$.

Recall that all the solutions of the $\mathcal{GS}$ need not be Nash equilibria. For rejecting unwanted non-equilibrium solutions, we apply the Nash equilibrium verification algorithm.

**Algorithm 5.2.1** Computing All Nash Equilibria of an IPIE game.

**Input:** An IPIE game, Galois groups.

**Output:** All equilibria of the input IPIE game in set $X$.

1. $\beta = (\beta_1, \beta_2, \ldots, \beta_{k+})$. \{Initialize an empty tuple to store a sample solution of the $\mathcal{GS}$\}.
2. Construct the $\mathcal{GS}$ of the input game.
3. Call Algorithm 5.2.2 with $\mathcal{GS}$, $d$ and $H$ for computing a sample equilibrium of the input IPIE game. \{ $d$ and $H$ can be obtained from inequalities in (5.1). \}
4. Call the Galois group action algorithm 4.2.3 with the sample solution tuple saved in $\beta$.
5. Save output of the Algorithm 4.2.3 in $X$.
6. Reject non-equilibrium solutions of the $\mathcal{GS}$ from $X$ using verification algorithm in [30] or criteria (2.6) and (2.2).
Algorithm 5.2.2 Computation of a sample solution with MVNRM.

Input: $\mathcal{GS}$ of the input game, $d$, $H$.

Output: A sample solution $\beta = (\beta_1, \beta_2, \ldots, \beta_{K^+})$ of the input game.

1: while one sample solution of the $\mathcal{GS}$ is not constructed do
2: Apply MVNRM with a starting solution $K^+$-tuple $u_0 = (0, 0, \ldots, 0)$. 
3: while inequality in Proposition 7 holds true. do
4: Compute approximate solution $u_k = (u_{k1}, u_{k2}, \ldots, u_{kK^+})$ of $\mathcal{GS}$. 
5: end while
6: Apply KLL Algorithm on each $u_{ki}$ and compute its minimal polynomial $f_i$.
7: if $f_i$ has linear factor $(x_i - \alpha_i)$ over $\mathbb{Q}$ then
8: Obtain a new polynomial system $\mathcal{GS}'$ after factoring out $(x_i - \alpha_i)$ from $\mathcal{GS}$ and go to Step 2 with new $\mathcal{GS}'$.
9: else
10: Save the solution tuple in $\beta$ and return.
11: end if
12: end while

The following result presents details of Step 8 in the Algorithm 5.2.2.

Proposition 8. After the factorization of linear factor in Step 8 of Algorithm 5.2.2, the new polynomial system $\mathcal{GS}'$ retains subset of solutions of the $\mathcal{GS}$ which has equilibrium solutions with all irrational coordinates.

Proof. Let $\mathcal{GS} = \{f_i(x_1, \ldots, x_{K^+}) \in \mathbb{Z}[x_1, \ldots, x_{K^+}] \mid i \in \{1, \ldots, K^+\}\}$. Let the variety $\mathcal{V}$ of the $\mathcal{GS}$ be $\{a_1, \ldots, a_m\}$. For some $a_t = (\alpha_t1, \ldots, \alpha_{ti}, \ldots, \alpha_{tK^+}) \in \mathcal{V}$ and $\alpha_{ti} \in \mathbb{Q}$.

We are interested in irrational roots only and so we must find a way to factor $p_i = (x_i - \alpha_{ti})$ from $\mathcal{GS}$. We perform the required factorization in following ways.

Construct $\mathcal{I} = \langle f_1, \ldots, f_{K^+}\rangle$, then our aim is to compute quotient $\mathcal{I} : \langle p_i \rangle$. The operation requires computation of a Gröbner basis $\{h_1, \ldots, h_k\}$ of $\mathcal{I} \cap \langle p_i \rangle$; then $\mathcal{GS}' = \{\frac{h_1}{p_i}, \ldots, \frac{h_k}{p_i}\}$.

Alternatively, compute a Gröbner basis, with a suitable lexicographic order, such that we get a univariate polynomial $g_k(x_i)$. Then we follow the usual univariate polynomial factorization $g'_k = \frac{g_k(x_i)}{p_i}$ and replace $g_k$ by $g'_k$ in the Gröbner basis to
5.2. Equilibria of IPIE Games

Consider new system of equations.

If \( \mathcal{I} \) and \( \mathcal{J} \) are any ideals, then computing intersection of ideals does the following to their variety (Ref. Th. 15, Ch. 4, [16])

\[
\mathcal{V}(\mathcal{I} \cap \mathcal{J}) = \mathcal{V}(\mathcal{I}) \cup \mathcal{V}(\mathcal{J}).
\]

In other words the variety of \( \mathcal{I} \cap \langle p_i \rangle \) remains unchanged. Gröbner basis computation does not change \( \mathcal{V} \). The only factorization operation is that of \( p_i \) that eliminates \( x_i - \alpha t_i = 0 \) and so the ideal of the new system has a variety that is a subset of the original \( \mathcal{V} \). Factorization of solution tuples with \( \alpha t_i \) which include other irrational/rational coordinates are not affecting the desired set of solutions of the GS that has solutions with all irrational coordinates.

The procedure of factoring out a root from \( \mathcal{I} \) of GS must be repeated for all \( \alpha t_i \in \mathbb{Q} \). Due to existence of a mixed Nash equilibrium and the fact that all equilibria are irrational for the input game, we are guaranteed to get one solution of GS in \( \beta \) and so Algorithm 5.2.2 reaches Step 12 every time. Due to the finiteness of the group and the variety on which it acts, as discussed in Chapter 4 the Algorithm 4.2.3 terminates and so does the Algorithm 5.2.1.

Note that the above method also computes solutions to a system of polynomial equation using its sample solution and its Galois group. After the computation of a sample solution, all other solutions computed are without factorization of the system of polynomial equations.

5.2.1 Some Properties of IPIE Games

From an algorithmic point of view, the main difference between Algorithm 4.2.1 and 5.2.1 is related to the computation of the sample solution. Algorithm 4.2.2 is a purely algebraic approach relying essentially on Buchberger’s Algorithm, whereas Algorithm 5.2.2 uses a numerical technique (MVNRM).

Apart from this, differences arise because the algebraic setting for Algorithm 5.2.1
5.2. Equilibria of IPIE Games

is that of ring extensions of \( \mathbb{Z} \). However, because linear factors over \( \mathbb{Q} \) are eliminated in Step 7 the algorithm does not compute rational solutions, i.e., those lying in rational extensions of \( \mathbb{Z} \). This is stated more precisely below.

**Proposition 9.** Algorithm [5.2.1] does not apply to (compute all equilibria of) the case of a game with integral payoffs and at least one rational (Nash) equilibrium.

**Proof.** Let \( T \) be a game with integer payoffs and one or more rational equilibria of the form \( a/b \), where \( a, b \neq 0 \in \mathbb{Z} \). This forces an extension \( \mathbb{K} = \mathbb{Z}(a/b) \) over \( \mathbb{Z} \). The group \( G \) of automorphisms of \( \mathbb{K} \) which fix \( \mathbb{Z} \) can be computed as follows.

Let \( c, d \in \mathbb{Z} \), for any \( c + (a/b)d \in \mathbb{K} \) and for any \( \sigma \neq id \in G \),

\[
\sigma(c + (a/b)d) = \sigma(c) + \sigma(a/b)\sigma(d) = c + \sigma(a/b)d,
\]

and \( \sigma(\frac{a}{b} \cdot b) = a \Rightarrow \sigma(\frac{a}{b})\sigma(b) = a \Rightarrow \sigma(\frac{a}{b}) = a/b \Rightarrow \sigma = \text{identity} \).

This means, the group of automorphisms of rational extensions of the ring of integers turns out to be a trivial identity group. And so, the group doesn’t provide necessary information for producing conjugate solutions of the \( \mathcal{GS} \).

Now, in order to prove the validity of the proposed method for IPIE games, we establish three more results, as follows:

**Proposition 10.** For any IPIE game, Galois groups representing its equilibrium solutions are non-trivial.

**Proof.** Equilibrium solutions of any IPIE game, by definition, generate irrational ring extensions over the ring of integers \( \mathbb{Z} \). Suppose, by the way of contradiction, Galois groups for some of the irrational extensions \( \mathbb{Z}(\alpha_i) \) are trivial. i.e., \( G_i = Gal(\mathbb{Z}(\alpha_i)/\mathbb{Z}) = \{e\} \). Then the minimal polynomial of each \( \alpha_i \) has all its factors linear over \( \mathbb{Z} \), and hence \( \alpha_i \in \mathbb{Z} \). This is impossible for IPIE games. And so the result follows.

The next result sets the criteria for the MVNRM to converge to a solution of a \( \mathcal{GS} \).
Proposition 11. Let \( x^j = (x_1, x_2, \ldots, x_{K^+}) \) be a \( j \)th strategy vector with each \( x_i \) denoting a probability of players for strategy \( i \) and let \( f(x) = (f_1(x), f_2(x), \ldots, f_{K^+}(x)) \), for \( f_i \in \mathcal{GS} \). Then MVNRM converges to a sample solution of \( \mathcal{GS} \) if the following condition holds: \(|f(x) \cdot J^2(f(x))| < |J(f(x))^2|\).

Proof. In MVNRM, an approximation of the \( n \)th strategy tuple \( x^n \) is computed using
\[
x^n = x^{n-1} - \frac{f(x^{n-1})}{Jf(x^{n-1})}.
\]
If we let
\[
\phi(x) = x - \frac{f(x)}{Jf(x)} \tag{5.3}
\]
then for overall convergence of MVNRM we need \(|\frac{d}{dx} \phi(x)| < 1\). Taking the derivative of (5.3) and simplifying it, we get \(|f(x) \cdot J^2(f(x))| < |J(f(x))^2|\). With this condition, MVNRM converges to a sample solution of the \( \mathcal{GS} \).

With the required tools in hand, we can now show the correctness of the method for computing all Nash equilibria of IPIE games.

Proposition 12. Algorithm 5.2.1 for computing all equilibria of IPIE games works. i.e., the output at termination consists of all irrational equilibria of the game, and no other solutions of the \( \mathcal{GS} \).

Proof. The input to the Algorithm 5.2.1 is an IPIE game \( T \) with \( n \) players. All the Nash equilibria of this game are characterized by a polynomial system \( \mathcal{GS} \) of the form (2.7). The polynomial system comes from the inequalities on expected payoffs and payoffs at pure strategies. These inequalities cause the system to have more solutions then just equilibria.

Algorithm 5.2.2 computes a sample solution of the \( \mathcal{GS} \) using MVNRM and saves it in \( \beta \). This is justified by the following chain of arguments: Nash [68] guarantees that an equilibrium exists with value in \((0, 1)\). Hence, MVNRM computes an approximate solution to the \( \mathcal{GS} \) within the degree of precision determined by Proposition 7. The KLL algorithm determines whether the solution is irrational. If not, then the corresponding rational or integer factor is factored out in Step 7 of Algorithm 5.2.2, and the process is repeated. Since the input game is IPIE,
eventually a sample solution is obtained. Roots in the sample solution extend the ring of integers $\mathbb{Z}$ to some Galois extension $\mathbb{K}$ of it. The Galois correspondence in Chase et al. \cite{10} and irreducible polynomials of univariate polynomials in ideal $\mathcal{I}$ of $\mathcal{GS}$ give meaningful transitive Galois groups $G = \text{Gal}(\mathbb{K}/\mathbb{Z})$ for the ring extensions. Proposition \ref{prop:5} ensures that the Algorithm \ref{alg:5.2.1} terminates with all equilibrium solutions of the input IPIE game.

Since Galois theory covers finite fields as well as arbitrary commutative rings \cite{10}, it is natural to ask whether our algorithms can also be extended to these situations. This question can be answered as follows:

**Proposition 13.** The algebra and algorithms for IPIE(RPIE) games cannot be extended to work over finite fields and their extensions.

*Proof.* If we define a finite normal form game over some finite number field, then the only polynomial algebra that we can consider is congruent-modulo algebra. i.e. polynomial system of form (2.7) will be modulo some prime or prime power. This forces the expected cost function codomain values to be restricted to the finite number field. The payoff functions in games must provide every player a choice over his strategies by suggesting an order between elements in the codomain, where the function maps strategies. It is known that, finite number fields are not ordered fields and so they fail to provide a total order amongst player strategies. Moreover, the available order over finite fields conflict with field operations and we cannot perform polynomial algebra. So, we cannot meaningfully define games, and consider polynomial algebra such as suggested in the Algorithms \ref{alg:5.2.1} and \ref{alg:4.2.1} for computing Nash equilibria of such games. \hfill \Box

Due to Proposition \ref{prop:6} in Chapter \ref{chap:4}, we know that for a subclass of RPIE games all its Nash equilibria can be computed in closed form. We now develop an analogous result for a subclass of IPIE games.

It is known that if a polynomial defined over fields has a solvable Galois group, then all its roots can be computed with radicals. If the result generalizes over rings then we can generalize the solvability by radical result, i.e. for some ring $S$
and a subring $R$ the following holds:

$$ R = \mathbb{Z} = L_0 \subset L_1 \subset \ldots \subset L_n = S, \quad (5.4) $$

and $\exists \alpha_i \in L_{i+1}$, a natural number $n_i$, such that $L_{i+1} = L_i(a_i)$ and $\alpha_i^{n_i} \in L_i$, then solvability by radicals can be extended for a subclass of IPIE games. All finite ring extensions need not be radical. With this restriction on the extension of the ring and the definition of Galois theory over rings, we have the following result.

**Proposition 14.** If the ring extension associated with an IPIE game is radical, then all the equilibria of the game can be computed in closed form.

**Proof.** Follows immediately from the discussion above. \qed

Note that the numerical approach for computing a sample solution in the Algorithm 5.2.2 can be used to replace the Buchberger’s Algorithm in the Chapter 4 for the class of RPIE games. The use of numerical method is not novel but independently developed for the games that we consider. Our method in Algorithm 5.2.2 is general, except a condition in Step 7, and can be considered for other classes of games with necessary modifications.

## 5.3 Computational Complexity

The characterization of equilibria as solutions to a system of polynomial equation is a polynomial time operation in the size of input payoff matrix, where the size of the matrix is $K^*$. The while loop in Algorithm 5.2.2 of Steps (1-12) runs until a sample solution of the GS is computed. For $i \in \{1, \ldots, K^+\}$ and for each indeterminate variable $x_i$, let $d_i$ denote the degree of its univariate polynomial in $I$ of the GS. An irrational root of some indeterminate $x_i$ will be available in at most $d_i$ factorization of its univariate polynomial. This implies that the while loop of steps (1-12) runs for at most $d = \max_i d_i$ times. Average case running time analysis of the Newton’s method – for computing approximate roots of a univariate polynomial – is studied by Smale [81, 82]. A sufficient number of the steps for the Newton’s method to obtain an approximate zero of a polynomial $f$, are polynomially bounded by the degree $d_i$ of the polynomial and $1/\rho$, where $\rho \in (0, 1)$
5.3. Computational Complexity

is the probability that the method fails. Kuhn’s algorithm improves efficiency by a polynomial factor and provides global convergence. On the other hand Renegar [76] studies the problem of computing approximate solutions of multivariate system of equations using homotopy method and presents an efficient algorithm. Note that these results on the complexity analysis assumes that the numerical method converges.

In the Algorithm 5.2.2, number of operations for constructing a minimal polynomial and checking its irreducibility over \( \mathbb{Q} \) are bounded by a polynomial in the size of degree \( d \) and maximum norm \( H \) of the minimal polynomial [31]. The operation of factoring a solution tuple from multivariate polynomial system in Step 8 of the Algorithm 5.2.2 require computation of ideal quotient, which in turn require computation of Gröbner basis. Step 8 executes only when there is a rational root of some univariate polynomial. The operation takes doubly exponential time in \( K^+ \).

Keeping aside this time, with these details, we present the following complexity bound for computing a sample solution with the Algorithm 5.2.2.

**Proposition 15.** Keeping aside time for operation in Step 8, Algorithm 5.2.2 runs in \( O(K^+d(1/\rho + H + dH)) \).

**Proof.** The while loop of (1-12) runs for at most \( d \) times. Considering the complexity of computing an approximate root of each univariate polynomial with Newton-Raphson’s method, the MVNRM with Proposition 11 runs polynomial in \( O(K^+d \cdot 1/\rho) \). The KLL Algorithm runs in \( O(dH) \), requiring at most \( K^+ \) repetition in worst case. The operation of checking irreducibility of a minimal polynomial, in worst case, is required for each indeterminate variable and for every factor of the univariate polynomials. The irreducibility check runs in \( O(dH) \). Summing up all these times and rearranging terms we get the result.

Computational complexity of the group action by Galois group in the Algorithm 4.2.3 is discussed in Section 4.3 of Chapter 4.
5.4 Equilibria Computation of an IPIE Game: An Example

We show the working of Algorithm 5.2.1 by computing all equilibria of a 3-player 2-strategy IPIE game with payoff matrix as in Table 5.1. The Algorithm 3.1.1 for deciding membership to the class of IPIE games confirms the game to be a member to the class of IPIE games.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a 3, 0, 2</td>
<td>0, 2, 0</td>
</tr>
<tr>
<td>1</td>
<td>b 0, 1, 0</td>
<td>1, 0, 0</td>
</tr>
<tr>
<td>2</td>
<td>a 1, 0, 0</td>
<td>0, 1, 0</td>
</tr>
<tr>
<td>2</td>
<td>b 0, 3, 0</td>
<td>2, 0, 3</td>
</tr>
</tbody>
</table>

Table 5.1: Payoff matrix of a 3-player 2-strategy IPIE game. Player 1 and 2’s strategies are indicated by a, b and A, B respectively. Player 3’s strategies are 1 and 2. Entry in each cell of the payoff table indicates player 1, 2 and 3’s payoff for their respective strategies.

We denote the probability of choosing first strategies of players 1, 2 and 3 by $x, y$ and $z$ respectively. The probability of choosing second strategies is $1 - x, 1 - y$ and $1 - z$ respectively. First, we characterize Nash equilibria of the game, in Table 5.1, as solutions to the GS with coefficients from $\mathbb{Z}$.

\[
(-1 + x)x(-1 + y + z + yz) = 0 \\
-(-1 + x)x(-1 + y + z + yz) = 0 \\
(-1 + y)y(3 + x(-2 + z) - 4z) = 0 \\
-(-1 + y)y(3 + x(-2 + z) - 4z) = 0 \\
-(3 + x(-3 + y) - 3y)(-1 + z)z = 0 \\
(3 + x(-3 + y) - 3y)(-1 + z)z = 0
\]  
(5.5)

With $d = 2$, $H = 3$ and the initial guess of a solution tuple consisting of all 0’s or 1’s, we apply MVNRM and compute the following solution tuple.

\[
x := 0.7282202113; \quad y := 0.3588989435; \quad z := 0.4717797888
\]  
(5.6)
5.4. Equilibria Computation of an IPIE Game: An Example

The KLL algorithm over the above solution tuple produces the minimal polynomial of each of the roots.

\[ 5x^2 - 16x + 9 = 0; \quad y^2 + 8y - 3 = 0; \quad 5z^2 + 4z - 3 = 0. \]  

(5.7)

These polynomials are irreducible over \( \mathbb{Z} \) and their Galois groups are isomorphic to \( \mathbb{Z}_2 \). To obtain a solution tuple in closed form, we factorize the minimal polynomials.\(^2\) Let one such solution be,

\[ x = \frac{1}{5}(8 + \sqrt{19}); \quad y = -4 - \sqrt{19}; \quad z = \frac{1}{5}(-2 - \sqrt{19}). \]  

(5.8)

This is a sample solution of the \( \mathcal{GS} \). Next we perform Galois group action on the sample solution. The action of generating Galois orbits for the solution (5.8) is similar to that given in Example 4.4 of Chapter 4. Once all the solutions are computed, we reject non-equilibria solution of the game with the polynomial time verification algorithm \[^{30}\]. This gives us the unique irrational equilibrium of the IPIE game.

\[ x = \frac{1}{5}(8 - \sqrt{19}); \quad y = -4 + \sqrt{19}; \quad z = \frac{1}{5}(-2 + \sqrt{19}). \]  

(5.9)

\(^2\)Clearly the Galois groups are solvable.
5.5 Discussion

In this chapter we presented a method for computing all equilibria of an IPIE game. The method in its first phase uses MVNRM and the KLL Algorithm. Newton-Raphson method is an efficient method for computing a sample solution and it does not stop in local minima [62]. This supports our choice of method.

It is particularly important to note that the traditional approaches of computing Nash equilibria with numerical methods produced equilibrium points in approximation form. Algorithm 5.2.1 does not depend on the probability distributions. With the convergence condition in Proposition 11 it is a deterministic method that produces equilibrium solutions in exact form using KLL algorithm.