Global Journal of Pure and Applied Mathematics.
ISSN 0973-1768 Volume 17, Number 2 (2021), pp. 149-158
©Research India Publications
http://www.ripublication.com/gjpam.htm

Complexity of Solution of Simultaneous Multivariate Polynomial Equations

Duggirala Meher Krishna[1], **Duggirala Ravi**[2]

[1]*Gayatri Vidya Parishad College of Engineering (Autonomous)*
Madhurawada, Visakhapatnam – 530 048, Andhra Pradesh, India.

[2]*Gayatri Vidya Parishad College of Engineering (Autonomous),*
Madhurawada, Visakhapatnam – 530 048, Andhra Pradesh, India.

Abstract

In this paper, an original reduction algorithm for solving simultaneous multivariate polynomial equations is presented. The algorithm is exponential in complexity, but the well-known algorithms, such as the extended Euclidean algorithm and Buchberger's algorithm, are superexponential. The superexponential complexity of the well-known algorithms is due to their not being "minimal" in a certain sense. Buchberger's algorithm produces a Gröbner basis. The proposed original reduction algorithm achieves the required task *via* computation of determinants of parametric Sylvester matrices, and produces a Rabin basis, which is shown to be minimal, when two multivariate polynomials are reduced at a time. The minimality of Rabin basis allows us to prove exponential lower bounds for the space complexity of an algebraic proof of certification, for a specific computational problem in the computational complexity class $\mathsf{Co}\_\mathcal{NP}$, showing that the complexity classes $\mathsf{Co}\_\mathcal{NP}$ and $\mathcal{P}$ cannot be the same. It follows, from the discussions, that $\mathcal{NP} \neq \mathcal{P}$, $\mathcal{NP} \neq \mathsf{Co}\_\mathcal{NP}$, $\mathcal{PSPACE} \neq \mathcal{NP}$ and that the polynomial time hierarchy does not collapse. It is also shown that the class of languages decidable by bounded error probabilistic algorithms with (probabilistic) polynomial time proofs for the membership of input words is not the same as any one of the complexity classes $\mathcal{P}$, $\mathcal{NP}$ and $\mathsf{Co}\_\mathcal{NP}$.

Keywords: Extended Euclidean algorithm; Buchberger's algorithm; Gröbner basis; Parametric Sylvester matrices; Rabin basis; Minimality; Algebraic complexity theory.

1. Introduction

Two prominent methods for reducing simultaneous multivariate polynomial equations are the extended Euclidean algorithm and Buchberger's algorithm. The reduction can be performed eliminating one variable at a time. If two multivariate polynomials vanish simultaneously, then so does their parametric gcd , where the parametric gcd is an element in the integral domain or the field of rational functions in the remaining variables. However, the converse may not be true: for two multivariate polynomials, if the parametric gcd vanishes, for a particular interpretation of variables from the algebraic closure of the ground field without any free variables, then the same cannot be guaranteed for the two multivariate polynomials, whose gcd has vanished under the chosen interpretation. The failure of the converse to hold true in general contributes to the superexponential complexity of these two well-known algorithms (Section 2).

Now, for the two multivariate polynomials in discussion to vanish simultaneously, it is both necessary and sufficient that the determinant of the parametric Sylvester matrix, called their *resultant* with respect to the variable being eliminated, vanishes, for any or some values in the ground field, in which the zeros are being searched for, assuming that neither of the two multivariate polynomials vanishes identically, under the chosen interpretation nullifying the resultant. The equivalence of the vanishing resultant — except when at least one of the two multivariate polynomials identically vanishes, under the interpretation of values to the variables other than the variable being eliminated in the current reduction step — to the sharing of a common zero in the algebraic closure of the ground field without parameters lends us the minimality criterion, for the reduction of the two multivariate polynomials. The entries of the Sylvester matrix being multivariate polynomials themselves, *albeit* without the variable being eliminated in the current step, the Gaussian elimination procedure cannot be applied, even though the final resultant, which is the determinant of the Sylvester matrix, is the same. The reason for inapplicability of the Gaussian elimination procedure for the determinant of the parametric matrices is that the computations in the intermediate stages might become excessively large, causing superexponential space and time for the completion of its computations, as can be achieved by simplification of intermediate results. The overall performance of the Gaussian elimination procedure for computation of the parametric resultant may be comparable to the extended Euclidean algorithm or Buchberger's algorithm, if not worse than either. A step-by-step simplified computation of the resultant that does not run into the space or time explosion problem, which is experienced with the Gaussian elimination method for the parametric matrices, is also presented. The reduced multivariate polynomial basis obtained by taking the resultant for each reduction step is called a *Rabin basis*, in honour of Professor Michael Oser Rabin, for his profound contributions to computer science (Section 3).

The minimality criterion allows us to derive exponential lower bounds for space requirement for an algebraic proof of certification, for a specific computational and decision problem, with polynomial instances of undetermined coefficients over any finite field. The specific computational problem exists in $\mathsf{Co}\_\mathcal{NP}$. It is customarily acknowledged that, for a computational problem to be in $\mathcal{P}$, a polynomial time algorithm, together with a proof of certification — the proof being bounded in space by a polynomial in appropriate parameter values for the instance — must exist for its correctness of operation. The class of nondeterministic computational or decision problems, for which it is possible to produce machine checkable deterministic proofs, *bounded in space by a polynomial for each such specific computational problem*, denoted by $\mathcal{NPSPACE}\_\mathsf{with\_proof\_in}\_\mathcal{PSPACE}$, relative to any particular fixed system of deductive or symbolic logic, equipped with rules of inference, that might be extensible, is included in $\mathcal{NP}$, and hence the two complexity classes $\mathcal{NPSPACE}\_\mathsf{with\_proof\_in}\_\mathcal{PSPACE}$ and $\mathcal{NP}$ represent the same complexity class. The machine checkable proofs may include references to external facts, the rules of inference may be specialized to a specific computational or decision problem, and the extensibility is the system's or users' ability to add more rules perhaps adaptively and / or interactively. The nonexistence of a polynomial time deterministic verification algorithm for a computational problem can be inferred from the nonexistence of a proof of correctness for any such algorithm, for its solution, that is bounded in space by a polynomial in the acceptable parameter values for its instances. There cannot be a shorter form for the algebraic proof of certification, for the specific computational problem in $\mathsf{Co}\_\mathcal{NP}$, because an immediate reflection shows that the degree of one of the output polynomials of the computational problem under investigation is exponential. Moreover, it is easily possible to assume multivariate polynomials with undetermined coefficients as instances for the specific computational problem in $\mathsf{Co}\_\mathcal{NP}$, and recursion can be applied, to produce an algebraic proof of correctness for the specific computational problem in discussion. The occurrence of recursion effectively annihilates any little hope of finding a proof of certification bounded in space by a polynomial, for any deterministic algorithm for the specific computational problem in its most generality, even when the interpretation is restricted to small dimension extension fields. In summary, we have to become contented in accepting that $\mathsf{Co}\_\mathcal{NP}$ cannot be $\mathcal{P}$, and hence $\mathcal{NP}$ cannot be $\mathcal{P}$, as well. In fact, an almighty can be assumed to be capable of guessing the correct answer to the question posed as part of the specific computational problem, but the impossibility of producing a polynomial space proof of certification shows that $\mathsf{Co}\_\mathcal{NP}$ cannot be $\mathcal{NP}$, either. In addition to these results, it is also shown that the class of languages acceptable by bounded error probabilistic algorithms $\mathcal{BPP}$ with probabilistic polynomial time proofs for the membership of an input word is not the same as either of the complexity classes $\mathcal{P}$, $\mathcal{NP}$ and $\mathsf{Co}\_\mathcal{NP}$

(Section 4).

2. Extended Euclidean Algorithm, Buchberger's Algorithm and Gröbner Basis

Let $\mathbb{F}$ be a field and $\mathbb{F}[x_1, \ldots, x_n]$, for some positive integer $n \geq 2$, be the integral domain of polynomials in n independent variables $x_1, \ldots, x_n$, with coefficients in $\mathbb{F}$. Let $\mathbf{x} = (x_1, \ldots, x_n)$, $\alpha(\mathbf{x}) = \sum_{i=0}^{d} a_i(x_1, \ldots, x_{n-1})x_n^i$ and $\beta(\mathbf{x}) = \sum_{i=0}^{d} b_i(x_1, \ldots, x_{n-1})x_n^i$ be two polynomials in $\mathbb{F}[x_1, \ldots, x_n]$, both of degree $d \geq 1$. It is further assumed that the polynomials $a_i(x_1, \ldots, x_{n-1})$ and $b_i(x_1, \ldots, x_{n-1})$ in $\mathbb{F}[x_1, \ldots, x_{n-1}]$ are all nonzero, and that each requires at least $L_{\min}$ units of space, for $0 \leq i \leq d$, such that they could include more than $L_{\min} \geq 2$ terms with very diverse exponent vectors, so that their products after expansion may contain only insignificantly small number of collision terms, for applying cancellations or simplification of terms, or they may admit succinct representations requiring at least $L_{\min}$ units of space, when their products are not expanded.

The operation of the extended Euclidean algorithm for computation of the parametric gcd is explained in the sequel. Since the two input polynomials are of the same degree d in x_n, an application of two consecutive steps to eliminate the highest degree term, *i.e.*, x_n^d, results in two multivariate polynomials of degree $d-1$ each, such that their coefficients would need $2L_{\min}$ units of space. Now, by induction, an application of two consecutive steps to eliminate x_n^{d-i}, from the two multivariate polynomials obtained as the result of the last consecutive pair of steps by eliminating x_n^{d-i+1}, for $i = 1, 2, \ldots d-1$, would result in $2^i L_{\min}$ units of space, without expansion. Thus, when the products are not expanded, the overall space requirement for the elimination of x_n is at least $\mathcal{O}\big(2^d L_{\min}\big)$. One more insight is concerning the final degree of any of the variables x_i, for $1 \leq i \leq n-1$. For simplicity, let the degree of occurrence of the variable x_i, for some fixed index i, where $1 \leq i \leq n-1$, be $\delta_i \geq 2$, for each term occurring as the coefficient of x_n in either input polynomial. The elimination procedure produces coefficients as multivariate polynomials in $\mathbb{F}[x_1, \ldots, x_{n-1}]$. Assuming that the occurrence of cancellations while simplifying the computations is a rare event, the degree of occurrence of the variable x_i in the parametric gcd can be lower bounded by $\mathcal{O}\big(2^d \delta_i\big)$, for $1 \leq i \leq n-1$.

Expansion and simplification of the products formed in the intermediate steps might not produce a lot of cancellations, and would only be expected to further blow up the space requirement. To eliminate x_n^d, by a consecutive pair steps, would need at least $\mathcal{O}\big(L_{\min}^2\big)$ space, after expansion, and by induction, to eliminate x_n^{d-i+1}, by a consecutive pair steps, would need at least $\mathcal{O}\big(L_{\min}^{2^i}\big)$ space, after expansion, for $i = 1, 2, \ldots, d$, resulting in the overall space requirement of at least $\mathcal{O}\big(L_{\min}^{2^d}\big)$ space, after expansion. This is the problem that causes the space explosion when the extended Euclidean algorithm is

applied, for eliminating the variable x_n, from the two input multivariate polynomials $\alpha(\mathbf{x})$ and $\beta(\mathbf{x})$. Most of the zeros of the parametric gcd might not lead to the common zeros of $\alpha(\mathbf{x})$ and $\beta(\mathbf{x})$.

Buchberger's algorithm follows closely the operational principle of the extended Euclidean algorithm and, in effect, emulates the latter by considering the exponent vector as a whole, in the sum of terms form. The multivariate polynomials so produced are collected in the Gröbner basis [1], named after the Ph D advisor of the author, presumably connoting Hilbert's basis theorem.

3. Parametric Sylvester Matrix, Parametric Resultant and Rabin Basis

Let $\alpha(\mathbf{x})$ and $\beta(\mathbf{x})$ be multivariate polynomials in $\mathbb{F}[x_1, \ldots, x_n]$, of degrees $d_\alpha \geq 1$ and $d_\beta \geq 1$. The Sylvester matrix corresponding to the polynomials $\alpha(\mathbf{x})$ and $\beta(\mathbf{x})$, for elimination of the variable x_n, is a $(D \times D)$ matrix, where $D = d_\alpha + d_\beta$, with entries either 0 or any of the multivariate polynomials $a_i(x_1, \ldots, x_{n-1})$, for $0 \leq i \leq d_\alpha$, and $b_j(x_1, \ldots, x_{n-1})$, for $0 \leq j \leq d_\beta$. It is assumed that the number of terms in the sum of terms form of expansion of the entries is at most $L_{\max}$ each for these polynomials. The resultant, denoted by $\mathsf{Res}\big(\alpha(\mathbf{x}), \beta(\mathbf{x})\big)$, with respect to the variable x_n, is the determinant of the $D \times D$ Sylvester matrix. The expansion of the determinant form as the sum of $D!$ many product terms shows that the number of terms in the resultant can be at most $D! L_{\max}^D < \big(DL_{\max}\big)^D$.

Now, comparing with the lower bounds for the number of terms obtained in the case of the extended Euclidean algorithms as in Section 2, it may be found that $\big(DL_{\max}\big)^D = L_{\min}^{D \log_{L_{\min}}(DL_{\max})} = L_{\min}^{\left(\frac{D \log_2(DL_{\max})}{\log_2(L_{\min})}\right)}$. With $d_\alpha = d_\beta = d$ and $D = 2d$, if $\big(\frac{2d \log_2(2dL_{\max})}{\log_2(L_{\min})}\big)$ is much smaller than 2^d, then, clearly, $\mathsf{Res}\big(\alpha(\mathbf{x}), \beta(\mathbf{x})\big)$ is much better. The observation holds, even when $d_\alpha \neq d_\beta$.

On the other hand, if the degree of occurrence of a variable x_i is at most Δ_i, for a fixed index i, where $1 \leq i \leq n-1$, in the multivariate polynomial coefficients of the input polynomials, then the degree of occurrence of the variable x_i in $\mathsf{Res}\big(\alpha(\mathbf{x}), \beta(\mathbf{x})\big)$, with respect to the variable x_n, is at most $D\Delta_i$. With $D = 2d$, if $2d\Delta_i$ is much smaller than $2^d \delta_i$, as found in the previous section, then, again clearly, computation of the parametric resultant yields superior performance compared to the extended Euclidean algorithm.

However, the straightforward expansion of the determinant form results in exponential time complexity, owing to the $D!$ many terms in the sum. Similarly, the Gaussian elimination method could deliver a worse performance than the extended Euclidean algorithm, because the intermediate results may not collapse into a small number of terms until the final result.

The following propagation of computations of the determinants of smaller dimension square submatrices to larger square submatrices is useful. For the computation of the determinant of a $D \times D$ matrix, for a large dimension $D > 1$, let the determinant $\mathsf{det}\big(S_k\big)$ and inverse S_k^{-1} of a $k \times k$ submatrix, S_k, be found, where S_k is a submatrix of the $k \times D$ matrix P_k, obtained by collecting the first k rows of the $D \times D$ matrix, inductively, for some $k \geq 2$, but $k \leq D - 1$. Let P_{k+1} be the $(k + 1) \times D$ matrix obtained by adjoining the next row in the $D \times D$ matrix to P_k. Assuming that the determinant of the given $D \times D$ matrix, which is $\mathsf{Res}\big(\alpha(\mathbf{x}),\, \beta(\mathbf{x})\big)$, for the Sylvester matrix, with respect to the variable x_n, does not identically vanish, as an element in the integral domain $\mathbb{F}[x_1, \ldots, x_{n-1}]$, the rows of the matrix P_{k+1} are linearly independent over the field of fractions of the integral domain $\mathbb{F}[x_1, \ldots, x_{n-1}]$. By the equality of the row rank to the column rank, there are $k + 1$ linearly independent columns of P_{k+1}. Now, of these linearly independent columns, k of the columns can be chosen to be those corresponding to the columns of S_k, for the following reason: the columns corresponding to S_k are linearly independent, by its invertibility, and if every column of P_{k+1} were a linear combination of the k columns, corresponding to those of S_k, then the column rank of P_{k+1} itself would be k. For the Sylvester matrix with respect to the variable x_n, the linear combination is taken over the field of fractions of the integral domain $\mathbb{F}[x_1, \ldots, x_{n-1}]$. Thus, at any point, if it is not possible to propagate the computation of the determinant from a $k \times k$ submatrix to $(k + 1) \times (k + 1)$ submatrix, for the reason that the column rank cannot increase, after adjoining any of the remaining $D - k$ rows to P_k, then the determinant of the given matrix itself vanishes, and, for the Sylvester matrix, with respect to the variable x_n, $\mathsf{Res}\big(\alpha(\mathbf{x}),\, \beta(\mathbf{x})\big)$ itself identically vanishes. Given S_k, $\mathsf{det}\big(S_k\big)$ and S_k^{-1}, the computations required for identifying an appropriate column in P_{k+1}, in order to form the $(k+1) \times (k+1)$ matrix S_{k+1}, and the determinant and the inverse of S_{k+1}, can be performed using standard formulas from matrix algebra. This method of computation of the parametric resultant, $\mathsf{Res}\big(\alpha(\mathbf{x}),\, \beta(\mathbf{x})\big)$, avoids needless space and time explosion, that can be observed in the Gaussian elimination method.

If $\mathsf{Res}\big(\alpha(\mathbf{x}),\, \beta(\mathbf{x})\big) = 0$, for some interpretation of the variables $x_i = \xi_i$ in the algebraic closure of $\mathbb{F}$, for $1 \leq i \leq n - 1$, and neither of $\alpha(\mathbf{x})$ and $\beta(\mathbf{x})$ vanishes identically as respective single variable polynomials in x_n, for the ground instances of $x_i = \xi_i$, for $1 \leq i \leq n - 1$, then $\alpha(\mathbf{x})$ and $\beta(\mathbf{x})$ share a common zero in the algebraic closure of $\mathbb{F}$. This property is called the *minimality* of the reduction step, for the elimination of the variable x_n, from the two participating multivariate polynomials. The collection of multivariate polynomials as obtained by computing the parametric resultant of two multivariate polynomials, at a time, with respect to any of the independent variables, is called a Rabin basis.

4. Proof Showing that $\mathsf{Co}\_\mathcal{NP} \neq \mathcal{P}$, $\mathsf{Co}\_\mathcal{NP} \neq \mathcal{NP}$ and $\mathcal{BPP} \neq \mathcal{P}$

4.1. Proof Showing that $\mathcal{NPSPACE}\_\mathsf{with\_proof\_in}\_\mathcal{PSPACE} = \mathcal{NP}$

For the definitions of the computational complexity classes denoted by $\mathcal{P}$, $\mathcal{NP}$, $\mathcal{NPSPACE}$ and $\mathcal{PSPACE}$, the readers are referred to [5], where $\mathcal{NPSPACE}$ appears as $\mathcal{NPS}$ and $\mathcal{PSPACE}$ as $\mathcal{PS}$. As an additional complexity class, let $\mathcal{NPSPACE}\_\mathsf{with\_proof\_in}\_\mathcal{PSPACE}$ be the collection of languages over an alphabet containing at least two symbols, such that for the acceptability of an input word, for each language independently, a nondeterministic requiring space bounded by a polynomial (specific to the particular language) in the string length of the input word, as with $\mathcal{NPSPACE}$, exists, but with an additional property that a proof of acceptance can be automatically generated with respect to any particular system of logic with its own rules of inference. Some of the rules of inference may be specialized to the particular language. The proof is assumed to be machine checkable, for its validity, in $\mathcal{PSPACE}$. The time required to check the validity of each step in the proof is taken to be bounded by a fixed, but sufficiently large, constant. The syntax checking of the proof is also assumed to require time bounded by a polynomial in the size of the proof. More specifically, the decision problem of proof checking is required to belong to the complexity class $\mathcal{P}$. It is easy to see that $\mathcal{NP} \subseteq \mathcal{NPSPACE}\_\mathsf{with\_proof\_in}\_\mathcal{PSPACE}$, because of the deterministic polynomial time verification condition for the languages in $\mathcal{NP}$.

For the converse inclusion, let Σ be an alphabet of at least two distinct symbols, and let $\mathcal{L} \subseteq \Sigma^\star$ be a language in $\mathcal{NPSPACE}\_\mathsf{with\_proof\_in}\_\mathcal{PSPACE}$. By assumption, there is a polynomial $p_{\mathcal{L}}(|\omega|)$, for every word $\omega \in \Sigma^\star$, where $|\omega|$ denotes the string length of ω, such that whenever $\omega \in \mathcal{L}$, there is a proof attesting to this fact, of at most $p_{\mathcal{L}}(|\omega|)$ bits of information, relative to a particular fixed system of logic, together with the rules of inference, perhaps specialized for the the language $\mathcal{L}$. Now, for every $\omega \in \mathcal{L}$, the proof that ω indeed belongs to $\mathcal{L}$ can be guessed, checked for syntactic correctness of the proof, and finally checked for the validity of the proof itself, in overall time bounded by some polynomial in $|\omega|$. For an external user, $p_{\mathcal{L}}(|\omega|)$ may still remain oblivious, as the technicality of asserting the membership of $\mathcal{L}$ to $\mathcal{NPSPACE}\_\mathsf{with\_proof\_in}\_\mathcal{PSPACE}$ assumes only its existence.

4.2. A Specific Computational Problem in $\mathsf{Co}\_\mathcal{NP}$ Belonging to Neither $\mathcal{NP}$ Nor $\mathcal{P}$

Let $\mathtt{p}$ be a large prime number, and $\mathbb{Z}_{\mathtt{p}}$ be the finite field of integers with arithmetic operations $\mathsf{mod}\ \mathtt{p}$. Let $m,\ n \geq 5$ be positive integers and $f(t,\ x) = \sum_{i=0}^{n-1} a_i(t)x^i +$

$x^n \in \mathbb{Z}_{\mathtt{p}}\,[t,\,x]$, where $a_i(t) \in \mathbb{Z}_{\mathtt{p}}\,[t]$, for $0 \leq i \leq n-1$, are undetermined coefficient polynomials in t, of degree at most m each.

The computational decision question is whether or not the following holds: $f(t,\,x) \neq 0$, for all $t,\,x \in \mathbb{Z}_{\mathtt{p}}$? If $f(t,\,x) = 0$, for some $t,\,x \in \mathbb{Z}_{\mathtt{p}}$, then the problem is in $\mathcal{NP}$, and if $f(t,\,x) \neq 0$, for every $t,\,x \in \mathbb{Z}_{\mathtt{p}}$, then the problem is in $\mathsf{Co}\_\mathcal{NP}$. The condition that $f(t,\,x) \neq 0$, for every $t,\,x \in \mathbb{Z}_{\mathtt{p}}$, is equivalent to the condition that gcd of the three polynomials $f(t,\,x)$, $(x^{\mathtt{p}} - x)$ and $(t^{\mathtt{p}} - t)$ is 1. A proof of certification could be a derivation that the gcd is indeed 1.

For a single variable polynomial $\phi(x) \in \mathbb{Z}_{\mathtt{p}}\,[x]$, by repeated squaring method, $x^{2^i}\ \mathsf{mod}\ \phi(x)$, for $i = 1,\,2,\,3,\,...,\,\lfloor \log_2(\,\mathtt{p}\,) \rfloor$, can be computed, from which $(x^{\mathtt{p}} - x)\ \mathsf{mod}\ \phi(x)$ can also be computed, and eventually, gcd of $(x^{\mathtt{p}} - x)\ \mathsf{mod}\ \phi(x)$ and $\phi(x)$ can be shown in a derivation sequence.

However, the same algorithm doesn't work with two variables, and in particular, the following system of simultaneous multivariate polynomial equations must be considered:

$$f(t,\,,x) = 0\,, \quad (x^{\mathtt{p}} - x) = 0 \quad \text{and} \quad (t^{\mathtt{p}} - t) = 0 \tag{1}$$

Thus, the resultant of $f(t,\,,x)$ with respect to any of the remaining two polynomials must be computed, and the gcd of the resultant and the other equation left out must be computed. This two step method is necessitated by the fact that the last two polynomial equations, *viz*, $(x^{\mathtt{p}} - x) = 0$ and $(t^{\mathtt{p}} - t) = 0$, have no variable in common. The space requirement for the resultant in the first step cannot be bounded by a polynomial in $\log(\,\mathtt{p}\,)$.

Thus, in order to testify the condition that $f(t,\,x)$, $(x^{\mathtt{p}} - x)$ and $(t^{\mathtt{p}} - t)$ have no zeros in simultaneously in satisfying them, the size of the proof cannot be bounded in space by some polynomial in $\log(\,\mathtt{p}\,)$, and hence $\mathsf{Co}\_\mathcal{NP} \neq \mathcal{P}$.

By accommodating more equations and raising the same question concerning the nonexistence of solutions to the systems of simultaneous multivariate equations, in general, the contention that $\mathsf{Co}\_\mathcal{NP} \neq \mathcal{P}$ becomes more aptly testified, because the solution space cannot be bounded by a polynomial in the acceptable parameter values, one of which is $\log(\,\mathtt{p}\,)$, in the preceding example, disallowing any claim of producing a direct polynomial time algebraic proof, for the validation of the answer produced by a nondeterministic algorithm.

Now, even if a deterministic algorithm for the problem is equipped with nondeterministic choices of size bounded by some polynomial in $\log(\,\mathtt{p}\,)$, the size of the proof remains exponential, and hence $\mathsf{Co}\_\mathcal{NP} \neq \mathcal{NP}$. The supporting evidence for the problem instance to be not in $\mathcal{NP}$ is that there is no deterministic proof of certification,

bounded in size by some polynomial in the parameter values, even if an assumed deterministic algorithm for it is equipped with the capability to make nondeterministic choices of size bounded by some polynomial in the acceptable values the parameters of the problem instance.

Some of the implications of the assertions that $\mathcal{NP} \neq \mathcal{P}$ and that $\mathsf{Co}\_\mathcal{NP} \neq \mathcal{NP}$ are that $\mathcal{PSPACE} \neq \mathcal{NP}$ and that the polynomial time hierarchy does not collapse.

4.3. Implications of the Fact that $\mathcal{PSPACE} = \mathcal{IP}$

In [6], it is shown that $\mathcal{IP} = \mathcal{PSPACE}$. By Savitch's theorem [5], $\mathcal{NPSPACE} = \mathcal{PSPACE}$. Allowing for the nondeterministic choices of space bounded by a polynomial in the string length of the input word, the class of languages that admit probabilistic polynomial time proofs, by nondeterminstic polynomial space algorithms, is exactly $\mathcal{IP}$. Let $\mathcal{BPP}$ be the class of languages acceptable by bounded error probabilistic algorithms with probabilistic polynomial time proofs attesting the membership of an input word each such language. Tentatively, if it is assumed that $\mathcal{BPP} = \mathcal{P}$, then clearly, it must be the case that $\mathcal{IP} = \mathcal{NPSPACE} = \mathcal{PSPACE} = \mathcal{NP}$. However, the discussion of the previous subsection shows that $\mathcal{PSPACE} \neq \mathcal{NP}$, implying that $\mathcal{BPP} \neq \mathcal{P}$. In fact, by equipping a probabilistic algorithm with the capability to make nondeterministic choices of size bounded by some polynomial in the string length of the input word, an algorithm in $\mathcal{NPSPACE}$ can be emulated, but a nondeterminstic polynomial time algorithm, equipped with the same additional capability, remains only a nonderministic polynomial time algorithm. Thus, $\mathcal{BPP} \neq \mathcal{NP}$ and $\mathcal{BPP} \neq \mathsf{Co}\_\mathcal{NP}$, as well.

5. Conclusions

This paper presents an original reduction method for solving simultaneous multivariate polynomial equations, by eliminating one variable, from two equations, taken at a time. The reduction method is shown to satisfy a certain minimality criterion, and hence becomes optimal in respect of the constraints stated. A mathematical problem that is in $\mathsf{Co}\_\mathcal{NP}$ but that which cannot be in either of $\mathcal{NP}$ and $\mathcal{P}$ is also presented. It follows that $\mathcal{NP} \neq \mathcal{P}$, $\mathcal{PSPACE} \neq \mathcal{NP}$, and that the polynomial time hierarchy does not collapse. As an afterthought, the trace of execution of an algorithm, even allowing for nondeterministic choices of sizes bounded by some polynomial in the appropriate values of the input parameters for the instances, must also be bounded in size by some polynomial in those parameter values. The trace may be supplied as the input to a debugger program for validation of its operation. This part may be included in the polynomial time certification for the algorithm, for each given input instance. The contentments assert that $\mathsf{Co}\_\mathcal{NP} \neq \mathcal{P}$ and $\mathsf{Co}\_\mathcal{NP} \neq \mathcal{NP}$, affirmatively. For

probabilistic algorithms, the class of languages decidable by probabilistic algorithms with probabilistic polynomial time proofs for the membership of an input word is not the same as the complexity class $\mathcal{P}$. In fact, by equipping a probabilistic algorithm with the capability to make nondeterministic choices of size bounded by some polynomial in the string length of the input word, an algorithm in $\mathcal{NPSPACE}$ can be emulated, but a nondeterminstic polynomial time algorithm, equipped with the same additional capability, remains only a nonderministic polynomial time algorithm. Thus, $\mathcal{BPP} \neq \mathcal{NP}$, as well.

Acknowledgements

The authors gratefully acknowledge fruitful discussions with Professor Michael Oser Rabin, Professor Ronald Linn Rivest and Professor Adi Shamir. The second author received introductions about machine checkable proofs and polynomial time verification certifications during a stint of postdoctoral position under the supervision of Professor Amir Pnueli, at the Weizmann Institute of Science, during the year 2002.

References

[1] Bruno Buchberger, "An Algorithm for Finding the Basis Elements of the Residue Class Ring of a Zero Dimensional Polynomial Ideal", Ph. D. Thesis, University of Innsbruck (1965), English translation by M. Abramson in *Journal of Symbolic Computation*, Special Issue on Logic, Mathematics, and Computer Science: Interactions, Vol. 41(3), 2006, pp. 475–511

[2] D. Castro, M. Giusti, J. Heintz, G. Matera, and L. M. Pardo, "The Hardness of Polynomial Equation Solving", *Foundations of Computational Mathematics*, Vol. 3(4), 2003, pp. 347–420

[3] J.-C. Faugère, "A New Efficient Algorithm for Computing Gröbner Bases (F4)", *Journal of Pure and Applied Algebra*, Vol. 139(1), 1999, pp. 61–88

[4] J.-C. Faugère, "A New Efficient Algorithm for Computing Gröbner Bases without Reduction to Zero (F5)", *Proc. International Symposium on Symbolic and Algebraic Computation*, ACM Press, 2002, pp. 75–83

[5] J. E. Hopcroft, R. Motwani, and J. D. Ullman, "Introduction to Automata Theory, Languages and Computation", Pearson Education, 2007

[6] Adi Shamir, "$\mathcal{IP} = \mathcal{PSPACE}$", *Journal of the ACM*, Vol. 39(4), 1992, pp. 869–877

[7] André Weil, "Number of Solutions of Equations in Finite Fields", *Bulletin of the American Mathematical Society*, Vol. 55(5), May 1949, pp. 497–508