RJ 6439 (62557) 9/19/88
Computer Science

A Note on the Complexity of P-Matrix LCP and Computing an Equilibrium

Nimrod Megiddo*

IBM Research
Almaden Research Center
650 Harry Road
San Jose, California 95120-6099

Abstract. It is proved that if it is NP-hard to solve the linear complementarity problem with P-matrix or to compute a Nash-equilibrium point in a 2-player game, then NP = coNP.

A Note on the Complexity of P-Matrix LCP and Computing an Equilibrium

Nimrod Megiddo*

Abstract. It is proved that if it is NP-hard to solve the linear complementarity problem with P-matrix or to compute a Nash-equilibrium point in a 2-player game, then NP = coNP.

1. Introduction

In this note we are concerned with two problems which are not known to be in the polynomial time class P, but whose NP-hardness implies NP = coNP. A similar property is shared by the members of the class of polynomial-time local search (PLS) recently introduced in [1]. The problems we consider here are not known even to be in PLS. As pointed out in [2] the class PLS seems to shed some light on the complexity of a special case of the linear complementarity problem (LCP) which is the following problem:

Problem 1.1. [LCP(M, q)] Given a rational matrix $M \in R^{n\times n}$ and a rational vector $q \in R^n$, find vectors $x, y \in R^n$ such that

$$y = Mx + q \quad , \quad x, y \geq \mathbf{o} \quad , \quad x^T y = 0 \ ,$$

or else conclude that no such vectors exist.

The purpose of this note is to prove a result of the type stated in [2], not only for the LCP with a P-matrix (i.e., a matrix M with positive prinicipal minors), but in a more general setting which includes the problem of computing an equilibrium point in an n-person game. The latter does not seem to be in PLS even though it can be solved by some extensions of Lemke's method. The LCP with a P-matrix is not known to be

*IBM Almaden Research Center, 650 Harry Road, San Jose, California 95120-6099, and School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel.

in PLS [3] since it is not known whether a P-matrix can be recognized in polynomial time.

In this note we are concerned only with rational inputs, so the assumption of rationality is henceforth omitted. In Section 2 we consider the LCP with P-matrix and in Section 3 the problem of computing an equilibrium point of an n-player game.

2. The LCP with a P-matrix

It is well-known that LCP(M, q) has a unique solution for every q if and only if M is a P-matrix. Moreover, since the problem can be solved by Lemke's method, the solution is basic and hence has size bounded by a polynomial in terms of the input size. Now, consider the following:

Problem 2.1. [PLCP] Given a P-matrix M and a vector q, solve the problem LCP(M, q).

Remark 2.2. From the viewpoint of traditional complexity theory, Problem PLCP has a non-standard form in the sense that its input space is restricted. In [1], the set of instances of a PLS problem is assumed to be a polynomial-time recognizable subset of $\{0,1\}^*$ (the set of all finite 0, 1-strings). Here, an algorithm for PLCP works under the guarantee that M is a P-matrix. Nonetheless, the notion of NP-hardness is well-defined for problems without the assumption that the set of instances is polynomial-time recognizable. Precisely, a problem L is NP-hard if there exists a polynomial-time algorithm for the satisfiability problem (SAT) which uses an oracle for L, each call to the oracle taking one time unit. A call to the oracle means that a valid input is given to the oracle and the latter returns a valid output. The oracle is not assumed to recognize in polynomial time that the input is valid.

In view of Remark 2.2, it is legitimate to ask whether the problem P-LCP is NP-hard. It is conjectured in [1] that the class PLS is easier than NP since (see Lemma 4 of Section 2 in [1]) if any PLS problem is NP-hard then NP = coNP. In [2] an attempt is made to rely on this lemma and show that if PLCP is NP-hard then NP = coNP. However, it is not known whether P-matrices can be recognized in polynomial-time, which is a prerequisite for showing that PLCP is in PLS. (Obviously, the problem of recognizing a P-matrix is in the class coNP.) If this were true, then (as argued in [2]) membership in PLS could be proved from the monotonicity of the homotopy parameter in Lemke's algorithm when the latter is applied to a P-matrix. The proof in [2] involves Lemke's method, ϵ-perturbations and other details which seem necessary for proving membership in PLS.

It turns out that we can prove that NP-hardness of PLCP implies NP = coNP without establishing that PLCP is in PLS. First, consider a more general problem where the set of valid instances is the entire $\{0,1\}^*$:

Problem 2.3. [PLCP*] Given any matrix $M \in R^{n\times n}$ and a vector $q \in R^n$, either exhibit a nonpositive principal minor of M or find a solution (x, y) of LCP(M, q).

We prove a claim which is stronger than the one in [2]:

Proposition 2.4. *If* PLCP* *is* NP-*hard then* NP = coNP.

> *Proof:* First note that problem PLCP* has a polynomial-time nondeterministic algorithm $\mathcal{A}$. This follows by observing that (i) if the matrix is not a P-matrix then a nonpositive principal minor can be guessed and checked in polynomial time, and (ii) if the matrix is a P-matrix then the LCP has a solution of polynomial size, which can therefore be guessed and checked in polynomial time. Suppose PLCP* is NP-hard, so there is a deterministic polynomial-time algorithm $\mathcal{B}$ for SAT which uses an $\mathcal{O}$ oracle for PLCP* . By substituting the nondeterministic $\mathcal{A}$ for $\mathcal{O}$, we obtain a polynomial-time nondeterministic algorithm for SAT which recognizes both satisfiable and unsatisfiable formulas. This means that SAT is in NP $\cap$ coNP and hence NP = coNP. ∎

Corollary 2.5. *If* PLCP *is* NP-*hard then* NP = coNP.

> *Proof:* By definition, if PLCP is NP-hard then so is PLCP* and the claim follows by Proposition 2.3. ∎

Note that the problem of recognizing whether a matrix is a P-matrix may be co-NP-complete, and also there may be easy way to prove a matrix is a P-matrix. A polynomial-time nondeterministic algorithm for PLCP* may compute a solution, but in general it would not prove that the matrix is a P-matrix. Only when the problem does have a solution the algorithm proves it is not a P-matrix.

3. Equilibrium points

A 2-player game (in normal form) can be defined as follows. The payoffs to players 1 and 2 are given, respectively, by rational matrices $\boldsymbol{A}, \boldsymbol{B} \in R^{m\times n}$. Mixed strategies for players 1 and 2 are, respectively, nonnegative vectors $\boldsymbol{x} \in R^m$ and $\boldsymbol{y} \in R^n$ such that $\boldsymbol{e}^T\boldsymbol{x} = \boldsymbol{e}^T\boldsymbol{y} = 1$ (where $\boldsymbol{e}$ denotes a vector of 1's). A (Nash)-equilibrium point is a pair

(x, y) of mixed strategies for players 1 and 2, respectively, such that for every mixed strategy z of player 1,

$$x^T A y \geq z^T A y$$

and for every mixed strategy w of player 2,

$$x^T B y \geq x^T B w \ .$$

A classic theorem says that every game has an equilibrium point. Now, denote by $M(R, C)$ a submatrix of a matrix M corresponding to a set R of row indices and a set C of column indices. Let $K_1 = \{1, \cdots, m\}$ and $K_2 = \{1, \cdots, n\}$.

Definition 3.1. An equilibrium point (x, y) is said to be *basic* if there exist subsets $M_i \subseteq L_i \subseteq K_i$ $(i = 1, 2)$ such that

(i) The columns of $A(L_1, M_2)$ are linearly independent and so are rows of $B(M_1, L_2)$.
(ii) For $i \notin M_1$, $x_i = 0$ and for $j \notin M_2$, $y_j = 0$.
(iii) $A(L_1, K_2)y = \lambda e$ and $A(K_1 \setminus L_1, K_2)y \geq \lambda e$ for some λ.
(iv) $x^T B(K_1, L_2) = \mu e^T$ and $x^T B(K_1, K_2 \setminus L_2)y \geq \mu e$, for some μ.

Once the existence of an equilibrium point has been established, standard linear programming arguments imply the existence of a basic equilibrium point. It follows that if the payoffs are rational numbers, then there exists an equilibrium point with numbers of polynomial size. This implies the following:

Proposition 3.2. *There exists a polynomial-time nondeterministic algorithm for computing an equilibrium point for any two-person game with rational payoffs.*

The following is a direct consequence:

Proposition 3.3. *If it is NP-hard to compute an equilibrium point then* NP = coNP.

Proof: The argument is essentially the same as in Proposition 2.4. If there is a polynomial-time deterministic algorithm for SAT which uses an oracle for equilibrium points, then we can substitute the oracle by a polynomial-time nondeterministic algorithm for an equilibrium point, and we obtain a polynomial-time nondeterministic algorithm for recognizing both satisfiable and unsatisfiable formulas. ∎

References

[1] D. S. Johnson, C. H. Papadimitriou and M. Yannakakis, "How easy is local search?", *J. Comp. Sci. Sys.*, to appear.

[2] D. Solow, R. Stone and C. A. Tovey, "Solving LCP on P-matrices is probably not NP-hard", unpublished note, November 1987.

[3] C. A. Tovey, personal communication, March 1988.