International Journal of Game Theory (1993) 22: 225–239

Game Theory

Computational Complexity to Verify the Unstability of Effectivity Function

MASAYOSHI MIZUTANI

Department of Business Administration, Tokyo Keizai University, 1-7-34 Minami, kokubunji, 185 Japan

YASUHIKO HIRAIDE

Advanced Technology Center, Yamatake-Honeywell Co., Ltd., 4-28-1, NishiRokugo, Ohta, 144 Japan

HISAKAZU NISHINO

Department of Administration Engineering, Faculty of Science and Technology, Keio University, 3-14-1, Hiyoshi, Kohoku, Yokohama, 223 Japan

Abstract: An n-person game is considered where each player has a preference order over a finite set A of possible alternatives and a rule for social choice is given in the form of an effectivity function E. The effectivity function is called stable if for any combination of individual preference orders there exists a subset of A called a core such that any alternative in the core cannot be 'dominated' by such individual preferences. It has been shown by Keiding (1985) that the effectivity function E is stable if and only if E does not generate any 'Cycle'. This paper is concerned with the computational complexity of the problem (CYCLE) for determining whether or not a given effectivity function has a Cycle. We show that a familiar NPC problem SATISFIABILITY can be transformed into CYCLE through a polynomial time procedure. This, combined with the fact that CYCLE is an NP problem, implies NP-completeness of CYCLE, and therefore that of verifying the unstability of the effectivity function, thereby formally proving a previously unanswered conjecture.

1 Introduction

We consider a society consisting of n members denoted by $N = \{1, 2, \ldots, n\}$. The i-th member has a preference order R^i over a set of m possible alternatives, $A = \{a_1, a_2, \ldots, a_m\}$. A preference profile is defined by $R^N = (R^1, R^2, \ldots, R^n)$. It is assumed that a rule is given for aggregating individual preferences (e.g. the majority voting rule, the Borda voting rule, etc.), through which a social choice is constructed. Some members may form a coalition, defined as a subset of N, in order to reflect their preferences on social choice. For example, under the majority voting rule, a coalition containing more than one half of the members can totally control the result of social choice.

In general, a coalition cannot yield the total dominance over a social choice. It may well be, however, that a coalition S can affect a social choice in such a way that the final result will belong to a subset B of A specified by the group. In this case, we say that a coalition S is effective for B. It should be noted that the smaller B is, the greater the potential power of S is. Even when S is effective for B, the group may not necessarily agree upon exercising the coalition power to actually realize B.

Depending on the way individuals negotiate within the group, a coalition S may have multiple effective sets. A rule for social choice may then be expressed by a function mapping a set of subsets of N into a set of subsets of a collection of subsets of A. This function is called an 'effectivity function'.

Given a preference profile, if a coalition S is effective for B and every member of S prefers any $y \in B$ to $x \notin B$, then x is blocked by S and will never be selected as a social choice. If there is a member of S who prefers $x \notin B$ to some $y \in B$, then x is not blocked by S since this member may not join S for blocking x in order to avoid one's personal loss. Given a preference profile, a set of alternatives which cannot be blocked by any coalition is called the 'core'. If the core is nonempty for every preference profile, then the effectivity function is said to be stable.

Existence of the core in the context of ordinary n-person games has been studied by Scarf (1967) and many others. For the case of games with effectivity functions described above, however, showing the existence of a core for a given preference profile is not sufficient. If there exists a preference profile under which the core is empty, the corresponding rule for aggregation of individual preference is considered to be incomplete. Accordingly, it is important to show the existence of the core for every preference profile, i.e., the stability of the effectivity function. For this problem, several approaches have been attempted in the literature. The pioneering work by Moulin and Peleg (1982) proved that an 'additive' effectivity function is always stable. Peleg (1983) showed that a 'convex' effectivity function is also stable. An interesting result due to Demange (1987) states that any core of a 'strictly stable' effectivity function, including 'convexity', is nonmanipulable in an optimistic sense. This implies that if the final social choice belongs to a core given a preference profile, then all members are convinced to accept it without complaint. A powerful necessary and sufficient condition for the stability of the effectivity function was first given by Keiding (1985) in terms of 'acyclicity'. Following this work, Lee, Nishino and Mizutani (1989) demonstrated that, for the stability, Scarf balancedness is sufficient in general, and is necessary and sufficient for a certain class of effectivity functions.

The number of all possible preference profiles increases exponentially as a function of the number of members and that of alternatives. Accordingly, in order to determine the stability, the use of any core finding algorithm is not encouraging since it involves complete enumeration of all possible preference profiles. In contrast, more promising may be the necessary and sufficient condition of Keiding (1985) that an effectivity function is stable if and only if the function has no Cycles, since the condition is independent of individual preferences of members. Although the number of Cycle candidates also increases exponentially as a function of number of members and that of alternatives, complete enumeration may be avoided by utilizing some structural properties.

This paper is concerned with the computational complexity of the problem CYCLE to determine whether or not a given effectivity function has a Cycle. In the theory of computation and algorithm, a set of problems having a nondeterministic solution algorithm which can be run in polynomial time is called $\mathcal{NP}$. A subset of $\mathcal{NP}$ consisting of 'the most intractable problems' is said to be $\mathcal{NPC}$. It is shown that a familiar NPC problem SATISFIABILITY can be transformed into CYCLE through a polynomial time procedure. We also show that CYCLE is an NP problem, thereby demonstrating the NP-completeness of CYCLE and formally proving a previously unanswered conjecture.

In section 2, notations, some definitions and concepts are given where an effectivity function is formally introduced. A succinct summary of the necessary and sufficient condition of Keiding (1985) for the stability is given in section 3 and an illustrating example is provided. Section 4 describes basic concepts and results in theory of computation and algorithms. Using such results, we prove the main result of this paper in section 5. Some remarks are given in section 6, including the discussion of computational complexity for finding the core.

2 Notation and Definitions

Let N be a nonempty finite set of players with $|N|=n$, where $|X|$ denotes the cardinality of X. A nonempty subset S of N is called a coalition.

Let A be a finite set of alternatives with $|A|=m\geqq 2$. R^i denotes a preference relation of the player $i\in N$ which is a complete and transitive binary relation on A. We write xP^iy if xR^iy and not yR^ix for $x, y\in A$. A preference profile is a combination of individual preference relations, written by $R^N=(R^1, R^2, \dots, R^n)$.

We also use the notation $P(D)$ for the set of all subsets of a set D, and let $P^2(D)\equiv P(P(D))$. Finally, $2^D\equiv P(D)\setminus\{\emptyset\}$.

An effectivity function is defined formally as below.

Definition 2.1: An effectivity function is a mapping $E:P(N)\rightarrow P^2(A)$ such that

(1) $\emptyset\notin E(S)$ *for every* $S\in P(N)$,
(2) $B\notin E(\emptyset)$ *for every* $B\in P(A)$,
(3) *for every* $S\in 2^N$, $A\in E(S)$,
(4) *for every* $B\in 2^A$, $B\in E(N)$.

The statement (1) and (3) indicate that any coalition will yield some result, whereas empty-coalition will not affect the social choice as specified in (2). Every coalition is allowed to be not influencing the social choice as specified in (3) and the statement (4) means that the coalition of the entire members can determine any social choice.

Some key concepts concerning effectivity functions are introduced next.

Definition 2.2: Let $E:P(N)\rightarrow P^2(A)$ be an effectivity function, and let a preference profile R^N be given. Furthermore, let $B\in 2^A$ and $x\in A\setminus B$. Then we say that B dominates x via a coalition $S\in 2^N$, written by B Dom$(R^N, S)x$, if

$B\in E(S)$ *and* bP^ix *for every* $b\in B$ *and every* $i\in S$.

We also say that B dominates x, denoted by B Dom$(R^N)x$, if there exists a $S\in 2^N$ such that B Dom$(R^N, S)x$.

Definition 2.3: The Core of A with respect to E and R^N, denoted by $C(A, E, R^N)$, is the set of all undominated alternatives in A. Namely,

$$C(A, E, R^N)=\{a\in A\,|\,\text{there exists no } B\in 2^A \text{ such that } B\ \mathrm{Dom}(R^N)a\}.$$

Stability of an effectivity function is defined below.

Definition 2.4: An effectivity function E is stable if

for every preference profile R^N, $C(A, E, R^N) \neq \emptyset$.

3 Condition for the Stability of the Effectivity Function

In this section, we discuss Keiding's result (1985) on the stability of the effectivity function. The concepts of a 'Strong Cycle' and a 'Cycle' are first introduced which will play an important role throughout this paper.

Definition 3.1: A Strong Cycle in E is a family $(S_1, S_2, \ldots, S_k; B_1, B_2, \ldots, B_k)$, where $S_i \in 2^N$, $B_i \in 2^A$, $B_i \in E(S_i)$, $i = 1, 2, \ldots, k$, such that

(1) *$B_i \cap B^j = \emptyset$ for every $i, j = 1, 2, \ldots, k$, $i \neq j$,*
(2) *$\bigcap_{i=1}^{k} S_i = \emptyset$.*

Definition 3.2: A cycle in E is a family $(S_1, S_2, \ldots, S_k; B_2, \ldots, B_k)$ with associated sets $C_1, C_2, \ldots, C_k$, where $S_i \in 2^N$, $B_i \in 2^A$, $B_i \in E(S_i)$, $C_i \in 2^A$, $i = 1, 2, \ldots, k$, such that

(1) *$C_i \cap C_j = \emptyset$ for every $i, j = 1, 2, \ldots, k$, $i \neq j$ and $\bigcup_{i=1}^{k} C_i = A$,*
(2) *$C_i \cap B_i = \emptyset$ for every $i = 1, 2, \ldots, k$,*
(3) *for every $i_1, i_2, \ldots, i_r \in \{1, 2, \ldots, k\}$,*
if $C_{i_j} \cap B_{i_{j+1}} \neq \emptyset$, $j = 1, 2, \ldots, r-1$, then $\bigcap_{j=1}^{r} S_{i_j} = \emptyset$ or $C_{i_r} \cap B_{i_1} = \emptyset$.

The following theorem holds concerning a relation between a Strong Cycle and a Cycle in E.

Theorem 3.1: Let $E: P(N) \rightarrow P^2(A)$ be an effectivity function. Any Strong Cycle in E is a Cycle with some associated sets C_i's. Therefore, if E has a Strong Cycle then E has a Cycle.

The following theorem characterizes stability of an effectivity function by 'acyclicity'.

Theorem 3.2: Let $E: P(N) \rightarrow P^2(A)$ ben an effectivity function. E is stable if and only if there is no Cycle in E.

The following example illustrates the concept of Core and Cycle.

Example 1: Let a society N and alternative set A be defined by $N = \{1, 2, 3, 4\}$ and $A = \{a_1, a_2, a_3, a_4\}$ respectively. An effectivity function is given by:

$E(\{1,3\})=\{a_4\}^+$, $E(\{1,4\})=\{a_3\}^+$, $E(\{1,2,4\})=\{a_1\}^+ \cup \{a_2\}^+$, $E(\{1,3,4\})=\{a_1\}^+$, $E(\{2,3,4\})=\{a_2\}^+ \cup \{a_3\}^+ \cup \{a_4\}^+$, $E(N)=2^A$ *and, otherwise,* $E(S)=\{A\}$,

where $B^+ \equiv \{\tilde{B} \mid \tilde{B} \in P(A),\ \tilde{B} \supset B\}$.

We can see that the above E has a Cycle $(S_1,\ldots,S_4;\ B_1,\ldots,B_4)=(\{2,3,4\},\ \{1,4\},\ \{1,3\},\ \{1,2,4\};\ \{a_3\},\ \{a_3\},\ \{a_4\},\ \{a_1\})$ *with the associated partition* $C_1=\{a_1\}$, $C_2=\{a_2\}$, $C_3=\{a_3\}$, $C_4=\{a_4\}$.

If the preference profile is

$a_1\,P^1\,a_4\,P^1\,a_3\,P^1\,a_2;$
$a_2\,P^2\,a_1\,P^2\,a_4\,P^2\,a_3;$
$a_4\,P^3\,a_3\,P^3\,a_2\,P^3\,a_1;$
$a_3\,P^4\,a_2\,P^4\,a_1\,P^4\,a_4,$

then the set $\{a_2\}(\{a_3\},\ \{a_4\},\ \{a_1\})$ *dominates the alternative* $a_1(a_2, a_3, a_4)$ *via the coalition* $\{2,3,4\}(\{1,4\},\ \{1,3\},\ \{1,2,4\})$ *and the Core is empty.*

4 Computational Complexity

This section summarizes some known results on computational complexity.

4.1 $\mathscr{P}$ and $\mathscr{N}\mathscr{P}$

A class of problems having the same solution structure is denoted by Π. An instance of problem in Π can be specified by selecting a particular set of admissible values of parameters. The total number of bits necessary for describing the problem instance is called the size of the instance. We denote an instance of size h by I_h.

In the study of computational complexity, underlying decision problems are supposed to have only two possible answers: 'yes' or 'no'. Such a decision problem can be described by specifying the following two parts;

1. INSTANCE: to determine a generic instance of the problem,
2. QUESTION: to state a yes–no question asked for the generic instance.

We abbreviate INSTANCE as I and QUESTION as Q.

If an algorithm Φ for solving the problem Π is given, the number of elementary calculations such as addition, substraction, comparison, etc., required to solve the problem instance I_h is denoted by $f_\Pi^\Phi(I_h)$. Let $\mathscr{I}_h$ be the set of all possible instances of size h, and define $F_\Pi^\Phi(h) \equiv \max_{I_h \in \mathscr{I}_h} f_\Pi^\Phi(I_h)$. Then the problem Π will be solved within $F_{\Pi}^{\Phi}(h)$ times calculations for any instances in $\mathscr{I}_h$. An algorithm Φ is called a polynomial time algorithm if there exists a polynomial p in h such that

for all h, $F_{\Pi}^{\Phi}(h) \leqq p(h)$.

Now, we define the class $\mathscr{P}$ of the decision problems.

Definition 4.1:
$\mathscr{P} \equiv \{\Pi \mid$ *there exists a polynomial time algorithm solving* $\Pi\}$.

Following Garey and Johnson (1979), we introduce the second important class of decision problems, the class $\mathscr{NP}$. We define $\mathscr{NP}$ in terms of a nondeterministic algorithm. A nondeterministic algorithm is composed of two separate stages, the first is called a guessing stage and the second a checking stage. The first stage merely guesses some structure R on a given problem instance I without care of how much time it takes to find R. Given I and R as input, the checking stage proceeds to compute and will either eventually halt with answer 'yes' or 'no', or compute forever without halting. We say an instance I satisfies the condition of the problem Π if the problem instance specified by I has the answer 'yes'. A nondeterministic algorithm is said to solve Π in polynomial time if there exists a polynomial p such that, for every instance I_h satisfying the condition of Π, there is some structure R which leads the checking stage to respond 'yes' within time $p(h)$. Now, we define the class $\mathscr{NP}$ of the decision problems.

Definition 4.2:
$\mathscr{NP} \equiv \{\Pi \mid$ *there exists a nondeterministic algorithm solving* Π *within polynomial time*$\}$.

Since a polynomial time algorithm is a polynomial time nondeterministic algorithm having polynomially solved R without regard of the answer for problem instances, $\mathscr{P}$ is included in $\mathscr{NP}$.

4.2 $\mathscr{NPC}$

In order to prove that a problem Π belongs to $\mathscr{P}$, it is enough to construct a polynomial time algorithm solving it. On the other hand, even if we can not find a polynomial time algorithm for the problem yet, we can not say that such an algorithm will never be discovered. Instead of showing that the problem considered does not belong to $\mathscr{P}$, Cook (1971) proposed an idea of defining a subclass $\mathscr{NPC}$ of $\mathscr{NP}$, with problems regarded as the most reasonable candidates not to belong to $\mathscr{P}$. Before discussing details, we define the notion of polynomial transformability.

Definition 4.3: Let D_{L_1} (D_{L_2}) *be the set of all possible instances of a problem* L_1 (L_2), *and let* Y_{L_1} (Y_{L_2}) *be the set of all instances satisfying the condition of the question of* L_1 (L_2) $(Y_{L_1} \subset D_{L_1}, Y_{L_2} \subset D_{L_2})$. *We say the problem* L_1 *is polynomially transformable* *into the problem L_2, which we denote by $L_1 \propto L_2$, if there is a polynomial time algorithm $f: D_{L_1} \to D_{L_2}$ such that*

for every $I \in D_{L_1}$, $I \in Y_{L_1} \Leftrightarrow f(I) \in Y_{L_2}$.

In other words, $L_1 \propto L_2$ implies that there is a procedure f to transform each I_h to $f(I_h)$ such that its computational complexity is polynomial of h and further,

$L_1(I_h)$ is 'yes' $\Leftrightarrow L_2(f(I_h))$ is 'yes'.

Since $f(D_1) \subset D_2$, if any problem belonging to $\mathscr{N}\mathscr{P}$ can be polynomially transformed to Π, then we can conclude that Π has the same computational complexity with the most intractable problems in $\mathscr{N}\mathscr{P}$. $\mathscr{N}\mathscr{P}\mathscr{C}$ is the family of such problems:

Definition 4.4:
$\mathscr{N}\mathscr{P}\mathscr{C} \equiv \{\Pi \in \mathscr{N}\mathscr{P} \mid \text{for any } L \in \mathscr{N}\mathscr{P},\ L \propto \Pi\}$.

Taking account of the fact that the polynomial transformability relation is transitive, we directly have the following statement;

$L_1 \in \mathscr{N}\mathscr{P}\mathscr{C}$, $L_2 \in \mathscr{N}\mathscr{P}$ and $L_1 \propto L_2 \Rightarrow L_2 \in \mathscr{N}\mathscr{P}\mathscr{C}$.

Thus, to prove that the problem Π belongs to $\mathscr{N}\mathscr{P}\mathscr{C}$, it is enough to show that

1. $\Pi \in \mathscr{N}\mathscr{P}$, and
2. some known problem $\Pi^* \in \mathscr{N}\mathscr{P}\mathscr{C}$ is polynomially transformed into Π.

Now, we refer to the familiar SATISFIABILITY problem (SAT, for short). The terms we shall use in describing SAT are defined as follows: Let $x_1, x_2, \ldots, x_m$ be Boolean variables and $\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_m$ be their negations, that is, x_i is *true* if and only if $\bar{x}_i$ is *false*. By a clause T, we mean a disjunction of the variables and their negations, and we say T is satisfied if there is an assignment of truth-values to $x_1, x_2, \ldots, x_m$ such that T is *true*. In the SAT problem, the conjunction of these clauses $T_1, T_2, \ldots, T_n$ is checked to be satisfiable. Formally, SAT is specified as follows;

Problem 1 (SAT):

*I: A set of Boolean variables $x_1, x_2, \ldots, x_m$, and
a collection of clauses $T_1, T_2, \ldots, T_n$ over them.*
Q: Is there a satisfying truth assignment for the collection?

Example 2: Consider the following input of SAT with $m=4$, $n=4$.

$T_1 = x_2 \vee \bar{x}_3 \vee \bar{x}_4$
$T_2 = \bar{x}_1 \vee x_3 \vee x_4$

$T_3 = x_1 \vee \bar{x}_2 \vee x_3$
$T_4 = x_4$

For example, if we set $x_1 = false$, $x_2 = false$, $x_3 = false$ and $x_4 = true$, this assignment on x_i's satisfies the above SAT.

We shall start from the following result on SAT shown by Cook (1971).

Theorem 4.1: SAT$\in \mathscr{NPC}$.

In the succeeding sections, we use SAT as the known problem Π^* in $\mathscr{NPC}$.

5 Computational Complexity of the CYCLE Problem

In this section we consider the problem (CYCLE) of verifying whether or not the effectivity function has a Cycle. First, we formulate the problem (SC) to verify whether or not the effectivity function has a Strong Cycle. Beginning with proving that SAT can be transformed into SC, we eventually show that CYCLE is in $\mathscr{NPC}$.

Since an effectivity function E is a mapping from $P(N)$ to $P^2(A)$, we formally have $I(E) = 2^n \times 2^m$, where $I(E)$ denotes the size of E. But the Strong Cycle and the Cycle can involve neither the coalition $\emptyset$ nor the alternative set A by their definition. Moreover, if the family $(S_1, S_2, \ldots, S_k; B_1, B_2, \ldots, B_k)$, $B_i \in E(S_i)$, $i = 1, 2, \ldots, k$ is a Strong Cycle (Cycle), then the family $(S_1, S_2, \ldots, S_k; B'_1, B'_2, \ldots, B'_k)$, $B'_i \in E(S_i)$, $B'_i \subset B_i$, $i = 1, 2, \ldots, k$ is also the Strong Cycle (Cycle). These imply that, in order to describe the decision problems of checking the existence of a Strong Cycle (Cycle), it is enough to provide the following restricted subfunction E^* of E;

$E^*: D \rightarrow P^2(A)$ such that
$D \equiv \{S \neq \emptyset \mid$ there exists $B \subset A$ such that $B \neq A$ and $B \in E(S)\}$

$$E^*(S) \ni B \Leftrightarrow \begin{array}{l} E(S) \ni B,\ B \neq A \text{ and} \\ \text{there exists no } B' \in E(S) \text{ such that } B' \subset B \text{ and } B' \neq B. \end{array}$$

Problem 2 (SC):

I: $N, A, E^*: D \rightarrow P^2(A)$
Q: Is there a family
$(S_1, S_2, \ldots, S_k; B_1, B_2, \ldots, B_k)$, *where* $S_i \in D$, $B_i \in 2^A$, $B_i \in E^*(S_i)$,
$i = 1, 2, \ldots, k$
such that
(1) $B_i \cap B_j = \emptyset$ *for every* $i, j = 1, 2, \ldots, k$, $i \neq j$
(2) $\bigcap_{i=1}^{k} S_i = \emptyset$?

Problem 3 (CYCLE):

I: $N, A, E^*: D \to P^2(A)$

Q: Is there a family
$(S_1, S_2, \ldots, S_k; B_1, B_2, \ldots, B_k)$ *with associated sets* $C_1, C_2, \ldots, C_k$, *where* $S_i \in D$, $B_i \in 2^A$, $B_i \in E^*(S_i)$, $C_i \in 2^A$, $i = 1, 2, \ldots, k$
such that
(1) $C_i \cap C_j = \emptyset$ *for every* $i, j = 1, 2, \ldots, k$, $i \neq j$ *and* $\bigcup_{i=1}^{k} C_i = A$
(2) $C_i \cap B_i = \emptyset$ *for every* $i = 1, 2, \ldots, k$
(3) *for every* $i_1, i_2, \ldots, i_r \in \{1, 2, \ldots, k\}$
if $C_{i_j} \cap B_{i_{j+1}} \neq \emptyset$, $j = 1, 2, \ldots, r-1$, *then* $\bigcap_{j=1}^{r} S_{i_j} = \emptyset$ *or* $C_{i_r} \cap B_{i_1} = \emptyset$?

Theorem 5.1: $SC \in \mathcal{N}\mathcal{P}\mathcal{C}$.

(*Proof*):

1. $SC \in \mathcal{N}\mathcal{P}$
If a guessed family $(S_1, S_2, \ldots, S_k; B_1, B_2, \ldots, B_k)$ such that $S_i \in D$, $B_j \in E^*(S_i)$, $i = 1, 2, \ldots, k$ is given, it is easy to check whether or not this satisfies the condition of SC. Indeed, to check the condition (1) in the SC (SC(1), for short) we need no more than $k^2 \times m$ times order calculations and SC(2) no more than $k \times n$. Hence $SC \in \mathcal{N}\mathcal{P}$.
2. $SAT \propto SC$
First, we present a rule f to transform SAT into SC. Adding to the original notation x_i, $\bar{x}_i$, we shall also use the notation y_i such that
$$y_i \equiv \begin{cases} x_i & \text{if } i = 1, 2, \ldots, m \\ \bar{x}_{i-m} & \text{if } i = m+1, m+2, \ldots, 2m. \end{cases}$$
Without loss of generality, we may assume there exists no y_i contained in every clauses, because SAT obviously remains $\mathcal{N}\mathcal{P}\mathcal{C}$ under this assumption.
Let N be $\{1, 2, \ldots, n\}$ and A be $\{a_1, a_2, \ldots, a_m\}$. E^* is defined as follows: Let I be $\{i \in \{1, 2, \ldots, 2m\} \mid \text{there exists } T_j \ni y_i\}$. And for every $i \in I$, set R_i to $\{j \in N \mid y_i \notin T_j\}$. Note that $R_i \neq \emptyset$, for every $i \in I$. Further, for each $S \in \{R_i \mid i \in I\}$ $(\equiv D \setminus \{N\})$, set $E^*(S)$ to $\{\{a_{i^\#}\} \mid S = R_i, i \in I\}$, where $i^\#$ denotes i if $i \leqq m$, and $i - m$ otherwise. And, set $E^*(N)$ to $\{\{a_i\} \mid i = 1, 2, \ldots, m\}$.
The above rule f is to define INSTANCE of SC from that of SAT. This f requires at most $4m^2 \times n$ times calculations, which implies f is a polynomial time algorithm.
Next, we show that a SAT problem has the answer 'yes' if and only if the corresponding SC problem has the answer 'yes'.

SAT is 'yes' $\Rightarrow$ SC is 'yes'.

Suppose that we have an assignment of y_i's by which the SAT is satisfied. Define $I_t \equiv \{i \mid y_i = true\}$ $(= \{i_1, i_2, \ldots, i_u\})$. Now, we show that $(R_{i_1}, R_{i_2}, \ldots, R_{i_u}; \{a_{i_1^\#}\}, \{a_{i_2^\#}\}, \ldots, \{a_{i_u^\#}\mathrm{N})$ is a Strong Cycle.
Since the assignment satisfies the SAT,

$$\text{for every } i, j \in I_t,\ i^\# \neq j^\#,$$

namely, the family of the one point set $\{a_{i^\#}\}$, $i \in I_t$ satisfies SC(1).
If this family does not satisfy SC(2), there must exist a clause which does not contain any *true* variable, i.e., which is not satisfied by any y_i.
Thus, if a SAT problem is 'yes', then the SC problem transformed from it is 'yes',

SC is 'yes' $\Rightarrow$ SAT is 'yes'.

Let $(S_{i_1}, S_{i_2}, \ldots, S_{i_k}; \{a_{j_1}\}, \{a_{j_2}\}, \ldots, \{a_{j_k}\})$, $\{a_{j_l}\} \in E^*(S_{i_l})$, $l = 1, 2, \ldots, k$ be a family that satisfies SC(1), (2).
Without loss of generality, we can assume that $S_{i_l} \neq N$, $l = 1, 2, \ldots, k$. Otherwise, after removing all sets S_{i_l}, $\{a_{j_l}\}$ such that $S_{i_l} = N$ from the Strong Cycle, we still have a Strong Cycle. From the transformation rule f,

$$\text{for every } l = 1, 2, \ldots, k,\ S_{i_l} = R_{j_l} \text{ or } R_{j_l + m},$$

and let $S_{i_l} = R_{j_l^*}$, where $j_l^* = j_l$ or $j_l + m$, $l = 1, 2, \ldots, k$. So, we get from SC(2),

$$\bigcap_{l=1}^{k} S_{i_l} = \bigcap_{l=1}^{k} R_{j_l^*} = \emptyset,$$

namely,

$$\text{for every } j \in N, \text{ there exists } y_{j_l^*} \in T_j.$$

Since indices $1 \leqq j_l \leqq m$, $l = 1, 2, \ldots, k$ satisfy $j_{l_1} \neq j_{l_2}$ for every $l_1, l_2 (l_1 \neq l_2)$ from SC(1),

$$\text{for every } l_1, l_2 = 1, 2, \ldots, k,\ (j_{l_1}^*)^\# \neq (j_{l_2}^*)^\#,$$

and therefore we can set the value of $y_{j_l^*}$, $l = 1, 2, \ldots, k$ to *true* without contradiction. Such assignment on y_i's satisfies all clauses T_i, $i = 1, 2, \ldots, n$. Thus, if the SC which is transformed from a SAT is 'yes', then the SAT is also 'yes'. Q.E.D.

Example 3: Recall the SAT problem with $m = 4$, $n = 4$ in Example 2. From the above rule f in the proof of Theorem 5.1 we directly have the following input of SC.

$N=\{1,2,3,4\}$
$A=\{a_1,a_2,a_3,a_4\}$
$I=\{1,2,\dots,8\}$
$R_1=\{1,2,4\}$ $R_5=\{1,3,4\}$
$R_2=\{2,3,4\}$ $R_6=\{1,2,4\}$
$R_3=\{1,4\}$ $R_7=\{2,3,4\}$
$R_4=\{1,3\}$ $R_8=\{2,3,4\}$
$D=\{\{1,3\},\{1,4\},\{1,2,4\},\{1,3,4\},\{2,3,4\},N\}$
$E^*(\{1,3\})=\{\{a_4\}\}$
$E^*(\{1,4\})=\{\{a_3\}\}$
$E^*(\{1,2,4\})=\{\{a_1\},\{a_2\}\}$
$E^*(\{1,3,4\})=\{\{a_1\}\}$
$E^*(\{2,3,4\})=\{\{a_2\},\{a_3\},\{a_4\}\}$
$E^*(N)=\{\{a_1\},\{a_2\},\{a_3\},\{a_4\}\}$

As there is a satisfying truth assignment $x_1=$false, $x_2=$false, $x_3=$false and $x_4=$true, the SAT has the answer 'yes'. Since the corresponding family $(S_1,\dots,S_4;B_1,\dots,B_4)=(R_5,R_6,R_7,R_4;\{a_{5*}\},\{a_{6*}\},\{a_{7*}\},\{a_{4*}\})=(\{1,3,4\},\{1,2,4\},\{2,3,4\},\{1,3\};\{a_1\},\{a_2\},\{a_3\},\{a_4\})$ *satisfies all conditions of a Strong Cylce, we can see that the problem SC also has the answer 'yes'.*

On the other hand, the above E^* has another Strong Cycle $(R_7,R_5,R_4;\{a_{7*}\},\{a_{5*}\},\{a_{4*}\})=(\{2,3,4\},\{1,2,4\},\{1,3\};\{a_3\},\{a_1\},\{a_4\})$. *This leads to an assignment satisfying the SAT: $x_1=$true, $x_2=$true or false, $x_3=$false and $x_4=$true.*

In order to prove our main result which states CYCLE$\in\mathcal{NPC}$, we need two lemmas described below.

Lemma 5.2: Let $E^*:D\to P^2(A)$ be a mapping. If the family $(S_1,S_2,\dots,S_k;B_1,B_2,\dots,B_k)$ is a Cycle in E^* with associated partition $C_1,C_2,\dots,C_k$ of A, then there is a sequence of indices $i_1,i_2,\dots,i_r\in\{1,2,\dots,k\}$ such that,

(1) $C_{i_j}\cap B_{i_{j+1}}\neq\emptyset\,(j=1,2,\dots,r-1)$ *and* $C_{i_r}\cap B_{i_1}\neq\emptyset$
(2) $\bigcap_{j=1}^{r}S_{i_j}=\emptyset$
(3) *for every* $j,j'=1,2,\dots,r,\ j\neq j',\ B_{i_j}\neq B_{i_{j'}}$.

(Proof):
We can get a sequence of indices that satisfies the above condition (1) as follows;

step 1] Choose an arbitrary index $j_1\in\{1,2,\dots,k\}$, and set $l\equiv 1$, $J\equiv\{j_1\}$.
step 2] Find a $C_{j_{l+1}}$ among C_i, $i=1,2,\dots,k$ which intersects with B_{j_l}.
step 3] If $j_{l+1}\notin J$ then set $J\to J\cup\{j_{l+1}\}$, $l\to l+1$, and go to step 2], otherwise finish the procedure.

Since C_i, $i=1,2,\dots,k$ provide a partition of A, we can always find $C_{j_{l+1}}$ in step 2]. Since $k\leq m$, this procedure must terminate. When the procedure has finished, there exists an index $j_{l'}\in J$ such that $j_{l+1}=j_{l'}$ which implies $B_{j_l}\cap C_{j_{l'}}\neq\emptyset$. If we set i_p to j_{l-p+1}, $p=1,2,\dots,l-l'+1(\equiv r)$ then we have $C_{i_r}\cap B_{i_1}\neq\emptyset$, and from the mechanism of this procedure we further have $C_{i_p}\cap B_{i_{p+1}}\neq\emptyset$ $(p=1,2,\dots,r-1)$. Therefore these $i_1, i_2, \ldots, i_r$ satisfy the condition (1). Moreover, the indices also satisfy the condition (2), because $(S_1, S_2, \ldots, S_k; B_1, B_2, \ldots, B_k)$ satisfies CYCLE(3) with the partition $C_1, C_2, \ldots, C_k$. Now, we show there exists a sequence of indices that satisfies the condition (3) adding to (1) and (2). Assume that any sequence of indices satisfying (1), (2) does not satisfy (3), and let $i_1, i_2, \ldots, i_r$ be a minimum sequence with respect to the number of indices that satisfies (1) and (2). From the assumption, there are indices l, l' such that $1 \leqq l, l' \leqq r (l < l')$, $B_{i_l} = B_{i_{l'}}$. Since $C_{i_l} \cap B_{i_l} = \emptyset$, we have $C_{i_l} \cap B_{i_{l'}} = \emptyset$, and consequently $l \leqq l' - 2$. If we take a subsequence $i_l, i_{l+1}, \ldots, i_{l'-1}$ of $i_1, i_2, \ldots, i_r$, we have, from the condition (1), $C_{i_j} \cap B_{i_{j+1}} \neq \emptyset$ $(j = l, l+1, \ldots, l'-2)$ and $C_{i_{l'-1}} \cap B_{i_{l'}} = C_{i_{l'-1}} \cap B_{i_l} \neq \emptyset$. Since the sequence $i_l, i_{l+1}, \ldots, i_{l'-1}$ satisfies CYCLE(3), $C_{i_{l'-1}} \cap B_{i_l} \neq \emptyset$ implies $\bigcap_{j=l}^{l'-1} S_{i_j} = \emptyset$. Thus $i_l, i_{l+1}, \ldots, i_{l'-1}$ is a sequence of indices satisfying (1), (2). Since $|\{i_l, i_{l+1}, \ldots, i_{l'-1}\}| < r$, we have a contradiction with the assumption of the minimality of r. Q.E.D.

Lemma 5.3: Let $E^*: D \to P^2(A)$ be a mapping such that $B \in E^*(S) \Rightarrow |B| = 1$. Then E^* has a Strong Cycle if and only if E^* has a Cycle.

(*Proof*):
E^* has a Strong Cycle $\Rightarrow$ E^* has a Cycle.

The proof of *Theorem 3.1* by Keiding (1985) includes the proof of this claim.

E^* has a Cycle $\Rightarrow$ E^* has a Strong Cycle.

Suppose the family $(S_1, S_2, \ldots, S_k; B_1, B_2, \ldots, B_k)$ is a Cycle in E^* with associated partition $C_1, C_2, \ldots, C_k$. From *Lemma 5.2*, there is a sequence of indices $i_1, i_2, \ldots, i_r \in \{1, 2, \ldots, k\}$ such that $\bigcap_{j=1}^{r} S_{i_j} = \emptyset$ (the condition (2) of the Strong Cycle), and for every $j, j' = 1, 2, \ldots, r$, $j \neq j'$, $B_{i_j} \neq B_{i_{j'}}$. Since $|B_{i_j}| = 1$, $j = 1, 2, \ldots, r$, the latter inequalities imply that for every $j, j' = 1, 2, \ldots, r$, $j \neq j'$, $B_{i_j} \cap B_{i_{j'}} = \emptyset$ (the condition (1) of the Strong Cycle). Hence the family $(S_{i_1}, S_{i_2}, \ldots, S_{i_r}; B_{i_1}, B_{i_2}, \ldots, B_{i_r})$ is a Strong Cycle in E^* Q.E.D.

Theorem 5.4: CYCLE $\in \mathcal{NPC}$

(*Proof*):

1. CYCLE $\in \mathcal{NP}$
   If a guessed family $(S_1, S_2, \ldots, S_k; B_1, B_2, \ldots, B_k)$ with associated sets $C_1, C_2, \ldots, C_k$ such that $S_i \in D$, $B_i \in E^*(S_i)$, $C_i \in 2^A$, $i = 1, 2, \ldots, k$ is given, it is obvious to check whether or not this satisfies the condition of CYCLE(1),(2) within polynomial time. Now, we need a method to verify whether or not the given family satisfies CYCLE(3).
   Define $I_p \equiv \{i \in \{1, 2, \ldots, k\} | p \in S_i\}$ for every $p = 1, 2, \ldots, n$. Consider a directed bipartite graph $G_p = (V_p, W_p, E_p)$ for each $p = 1, 2, \ldots, n$. Where each the vertex set V_p, W_p represents the family of sets B_i, $C_i (i \in I_p)$, namely, $V_p \equiv \{b_i | i \in I_p\}$, $W_p \equiv \{c_i | i \in I_p\}$. $E_p \subset V_p \times W_p \cup W_p \times V_p$ is the set of directed edges satisfying one of the following conditions:

(1) $(b_i, c_j) \in E_p \Leftrightarrow i=j$,
(2) $(c_i, b_j) \in E_p \Leftrightarrow C_i \cap B_j \neq \emptyset$.

It is easy to see that there exists a G_p which has a cycle for some p if and only if the corresponding sets guessed in the first stage of a nondeterministic algorithm violate the condition CYCLE(3) with respect to some nonempty subset of I_p. Since Tarjan (1972) constructed a polynomial algorithm for finding cycles in a given directed graph $G=(V, E)$, the computational complexity of checking the condition CYCLE(3) is also performed within polynomial time.

Hence CYCLE$\in \mathcal{NP}$.

2. SAT $\propto$ CYCLE
   The proof of *Theorem 5.1* has provided a polynomial time deterministic algorithm f which transforms SAT into SC with an E^* such that $B \in E^*(S) \Rightarrow |B| = 1$. Besides, *Lemma 5.3* states that the existence of a Strong Cycle in such an E^* is equivalent to the existence of a Cycle in the E^*. This implies that f transforms SAT into CYCLE in E^*.

Example 4: Recall the E^* in Example 3. The former Strong Cycle in Example 3 is also a Cycle, if we set $C_1=\{a_2\}$, $C_2=\{a_3\}$, $C_3=\{a_4\}$ and $C_4=\{a_1\}$, and we have that both the SAT problem and the CYCLE have the answer 'yes'.

On the other hand, the E^* has another Cycle $(S'_1, \ldots, S'_4;\ B'_1, \ldots, B'_4) = (\{2,3,4\}, \{1,4\}, \{1,3\}, \{1,2,4\};\ \{a_3\}, \{a_3\}, \{a_4\}, \{a_1\})$ with the associated partition $C'_1=\{a_1\}$, $C'_2=\{a_2\}$, $C'_3=\{a_3\}$, $C'_4=\{a_4\}$. If we set $i_1=1$, $i_2=4$ and *$i_3=3$, we can see that the sequence i_1, i_2, i_3 satisfies the conditions, described in Lemma 5.2, $C'_{i_2} \cap B'_{i_2} \neq \emptyset$, $C'_{i_2} \cap B'_{i_3} \neq \emptyset$, $C'_{i_3} \cap B'_{i_1} \neq \emptyset$ and $\bigcap_{j=1}^{3} S'_{i_j} = \emptyset$. As has been shown in Lemma 5.3, we get the latter Strong cycle in Example 3 $(S'_1, S'_4, S'_3;\ B'_1, B'_4, B'_3) = (\{2,3,4\}, \{1,2,4\}, \{1,3\};\ \{a_3\}, \{a_1\}, \{a_4\})$.*

6 Concluding Remarks

We have shown that the problem of checking whether or not an effectivity function has a Cycle belongs to $\mathcal{NPC}$.

If, it an effectivity function E, there are $B_1 \in E(S_1)$ and $B_2 \in E(S_2)$ such that $B_1 \cap B_2 = \emptyset$ and $S_1 \cap S_2 = \emptyset$, then the corresponding game is inconsistent, and E is called 'nonproper'. Since the family $(S_1, S_2;\ B_1, B_2)$ is a Strong Cycle in E, any nonproper E has a Strong Cycle with $k=2$, and vice versa. Accordingly, the properness of given E can be tested by showing whether or not a Strong Cycle exists with $k=2$, which is an easy task. Since E may neither have a Strong Cycle nor a Cycle, it is necessary to test the conditions of combinatorial number for every $k \geqq 3$.

The class of mappings E^*'s transformed from the SAT by the rule f given in the proof of *Theorem 5.1* merely provides a particular and less complex class of effectivity functions. Indeed, each E^* has only singletons of A in its range and involves at most $2n+1$ coalitions. It should be noted that E^* may be nonproper. Even for such a primitive class of effectivity functions E^* including nonproper ones, the problem SC (or CYCLE) has the same difficulty as that of SAT. This suggests the hardness of the problem to find a Strong Cycle or Cycle in an arbitrary given proper effectivity function E.

On the other hand, the problem of verifying the existence of the Core for any given preference profile is very easy to solve, and indeed belongs to $\mathscr{P}$, as we prove next.

Problem 4 (CORE):

> *I:* $A, E^*: D \rightarrow P^2(A), R^N$
> *Q: Is there a Core, i.e., is* $C(A, E, R^N)$ *nonempty?*

Theorem 6.1: $CORE \in \mathscr{P}$

(Proof):
From *Definition 2.3* the following fact is obvious: for each $a \in A$, a is not in $C(A, E, R^N)$ if and only if there exists S and B such that

(1) $S \in D$,
(2) $B \in E^*(S)$ and
(3) for every $b \in B$, $i \in S$, $bP^i a$.

Each condition can be easily checked in polynomial time with respect to $I(E^*)$ and $I(R^N)$. Therefore the problem CORE can be solved in polynomial time with respect to $|A|$, $I(E^*)$ and $I(R^N)$. Q.E.D.

Since the size of R^N is polynomial with respect to n and m, there is relatively little difference between the size of instance of the problem CORE and that of CYCLE. Nevertheless, the problem CORE belongs to $\mathscr{P}$ and CYCLE belongs to $\mathscr{N}\mathscr{P}\mathscr{C}$. These facts suggest that the problem CYCLE is much more intractable than CORE to solve. And we can conclude that the stability problem is hopeless to be solved by a polynomial time algorithm.

References

Cook SA (1971) The complexity of theorem - proving procedure. Proceedings of the third annual ACM symposium on theory of computing 151–158

Demange G (1987) Nonmanipulable cores. Econometrica 55:1057–1074

Gary MR, Johnson DS (1979) Computers and intractability - A guide to the theory of NP-completeness. Freeman, New York

Keiding H (1985) Necessary and sufficient condition for stability of effectivity functions. International Journal of Game Theory 14:93–101

Lee NC, Nishino H, Mizutami M (1989) The nonemptiness of the core - balancedness of effectivity function. Discussion Paper

Moulin H, Peleg B (1982) Cores of effectivity functions and implementation theory. Journal of Economic Theory 10:115–145

Peleg B (1983) Game theoretic analysis of voting in committees. Cambridge University Press, Cambridge

Scarf HE (1967) The core of an n-person game. Econometrica 35:50–69

Tarjan R (1972) Depth-first search and linear graph algorithms. SIAM Journal on Computing 1:146–160

Received April 1992
revised version March 1993
final version May 1993