Finding a simple Nash Equilibrium

Shu-Yang Sun, Da-You Liu, and Cheng-Min Sun

College of Computer Science and Technology, Jilin University, Changchun, 130012 China
Key Lab for Symbolic Computation and Knowledge Engineering of Ministry of Education, Changchun, 130012 China

Summary
In this paper we propose a search method for finding a simple Nash Equilibrium in a 2-player game and execute it on 24 classes of games. The result shows that our algorithm performs better than the classical Lemke-Howson algorithm.

Key words:
game theory, 2-player game, Nash Equilibrium computing.

Introduction

Non-cooperative game theory has been tightly combined with Computational Methods, in which the Nash Equilibrium (NE) is undoubtedly the most important solution concept.
While players face several Nash Equilibriums, their most pragmatic strategy may not be the most optimal one[1], but one they prefer for other reasons. Players may choose a sub-optimal strategy instead of a more optimal but more complex one which might be difficult to learn or to implement. In one word, players prefer strategies as simple as possible.
The problem is whether there exists a simple NE in a 2-player game, and if so, how to find it. To solve this problem, this paper proposes a search method. Comparing with finding a NE, it is much easier to compute whether a NE exists in a special support for each player. Based on this, our algorithm repeats checking the existence of NE in a limited and ordered support space until a NE is found. There are 3 important steps in our algorithm: first limiting the search space; second ordering the search space; lastly, checking in turn (according to the order) whether there exists a NE with this support for each player.
We are not the first ones who search support space to find a NE. Dickhaut and Kaplan[2] proposed an algorithm to find all Nash Equlibria. Porter, Nudelman and Shoham[3] designed an algorithm to find a sample NE. However, the innovation in this paper is on pruning the search space. Our pruning rules are (1) no conditionally dominant strategies in any NE, (2) there exists such a NE whose support is less than k+1 (k is the rank of corresponding payoff matrix). These rules ensure that our search space is smaller than that of [3], but the disadvantage is that ranks of payoff matrixes must be computed.

Manuscript received February 25, 2006.
Manuscript revised February 28 , 2006.

The rest of this paper is structured as follows: first we formulate the problem and give the relevant definitions, and then we describe the algorithm. After that, we compare the execution of our algorithm and the Lemke-Howson algorithm[4]. In the final section, we conclude our work.

2. Notations and Definitions

The necessary notations and definitions are shown in this section.
The set of players is noted as $N = \{1, 2\}$.
The sets of pure strategies of player 1 and 2 are noted as $S_1 = \{s_{11}, s_{12}, \ldots, s_{1m}\}$ and $S_2 = \{s_{21}, s_{22}, \ldots, s_{2n}\}$.
The payoff matrixes of player 1 and 2 are noted as $(U_1)_{m\times n}$ and $(U_2)_{n\times m}$ respectively.
Definition 1. The i^{th} row of matrix $U_{m\times n}$ is *conditionally dominant*, if $\exists 1 \le j \le m$, $\forall k \in \{1,\ldots,n\}$, s.t. $u_{ik} \le u_{jk}$ and $\exists 1 \le t \le m$, s.t. $u_{it} < u_{jt}$.
$S_1' = \{s_{1i} \in S_1 \mid$ the i^{th} row of matrix U_1 is conditionally dominant $\}$
$S_2' = \{s_{2i} \in S_2 \mid$ the i^{th} row of matrix U_2 is conditionally dominant $\}$
The strategies in S_1' and S_2' are called *conditionally dominant strategies*.
Definition 2. A *mixed strategy* for a player is a probability distribution over the set of his pure strategies and will be represented by a vector $x=(x_1, x_2, \ldots x_m)$, where $x_i \ge 0$ *and* $\sum_{i=1}^{m} x_i = 1$. Here x_i is the probability that the player will choose his i^{th} pure strategy. The *support* of x (Supp(x)) is the set of pure strategies that it uses.
Definition 3. A profile of mixed strategies (p, q) for a 2-player game is balanced, if $|Supp(p)| = |Supp(q)|$.
Theorem 1 : For any Nash Equilibrium (x^* , y^*), $Supp(x^*) \cap S_1' = \Phi$, $Supp(y^*) \cap S_2' = \Phi$.
Proof. Using proof by contradiction.
Theorem 2: $(U_1)_{m\times n}$, $(U_2)_{n\times m}$ are payoff matrixes respectively for player 1 and 2 in a 2-player game.

$\mathrm{rank}(U_1) \le k$, $\mathrm{rank}(U_2) \le d$. $\exists$ a NE (x, y), s.t. $|\mathrm{Support}(x)| \le k+1$, $|\mathrm{Support}(y)| \le d+1$.
Proof. See [5].

3. Algorithm

The following is a feasible program for finding a NE in a special support for each player, where U_1 and U_2 correspond payoff matrixes of 2 players, and S_1^t , S_2^t correspond the special support for each player.

Vectors $p_1 \in R^m, p_2 \in R^n$ correspond Nash Equilibrium (if they exist), and $c_i \ge 0$ are real numbers.

$p_{-1} \in R^m, p_{-2} \in R^n$ are vectors satisfying constraints 1, 2 and 4.

Feasibility Program 1

Input: $(U_1)_{m\times n}$, $(U_2)_{n\times m}$, S_1^t , S_2^t .

Output: p_i , c_i , $i = 1, 2$ s.t.

Constraints: $\forall\, i = 1, 2$

1) $\sum_{s_{ij} \in S_i^t} p_{\pm i}(s_{ij}) = 1$,

and
$$0 < p_{\pm i}(s_{ij}) \le 1, \quad if\ s_{ij} \in S_i^t$$
$$p_{\pm i}(s_{ij}) = 0, \quad if\ s_{ij} \notin S_i^t$$

2) $\exists s_{ij} \in S_i^t$, st. $p_i(s_{ij}) \ne p_{-i}(s_{ij})$

3) $p_1 \cdot U_i \cdot p_2^{\,T} = c_i$

4) $p_{(-1)^i \cdot 1} \cdot U_i \cdot p_{(-1)^{i+1} \cdot 2}^{\,T} \le c_i$

See [6] for solving feasible program 1.

Definition 4. Suppose that x and y are positive integers. G={1,2,…,x}×{1,2,…,y} , $(x_1, y_1), (x_2, y_2) \in G$. The simple and balanced order (G, ≤) is defined as follows: $(x_1, y_1) \le (x_2, y_2)$ if and only if one of following conditions are satisfied:

(1) $| x_1 - y_1 | < | x_2 - y_2 |$

(2) $| x_1 - y_1 | = | x_2 - y_2 |$ and
$$\min\{x_1, y_1\} \le \min\{x_2, y_2\}$$

The following algorithm is for finding a simple NE in a 2-player game.

Search space of algorithm 1: according to theorem 1, there are no conditionally dominant strategies in any Nash Equilibrium; according to theorem 2, there exists a Nash Equilibrium whose support of mixed strategies is less than k+1 (k is the rank of corresponding payoff matrix). The search space is based on these two theorems.

Algorithm 1

// computing a NE in a 2-player game, 1 and 2 are players
Input: $S_1$and S_2 (sets of pure strategies), $(U_1)_{m\times n}$ and $(U_2)_{n\times m}$ (payoff matrixes)
$S_1' \leftarrow \{s_{1i} \in S_1 \mid$ the i^{th} row of matrix U_1 is conditionally dominant $\}$
$S_2' \leftarrow \{s_{2i} \in S_2 \mid$ the i^{th} row of matrix U_2 is conditionally dominant$\}$
$S_A \leftarrow S_1 - S_1'$, $S_B \leftarrow S_2 - S_2'$
$k \leftarrow \mathrm{rank}(U_1)$, $d \leftarrow \mathrm{rank}(U_2)$
$x \leftarrow \min\{k+1, |S_A|\}$
$y \leftarrow \min\{d+1, |S_B|\}$
FOR all (x',y') where $x' \in \{1,2,...,x\}, y' \in \{1,2,...,y\}$, and ($(\{1,2,...,x\} \times \{1,2,...,y\})$,≤) is a simple and balanced order
    FOR all $S_1^t \subseteq S_A$, s.t. $|S_1^t| = x'$ DO
        FOR all $S_2^t \subseteq S_B$, s.t. $|S_2^t| = y'$ DO
            IF $\exists$ (p, c), which satisfies feasible program 1 for (S_1^t, S_2^t) THEN
                RETURN p // p is the found equilibrium

4. Assessment (Evaluation)

Several comparisons were committed between the performance of algorithm 1 and that of the Lemke-Howson algorithm implemented in Gambit[7] to assess our algorithm.

We follow the idea of [3] and use GAMUT (GAMUT can generate games from a wide variety of classes of games found in the literature) to generate 24 different classes of games. Both algorithms are executed on 100 2-player 300-action games drawn from each class (i.e. 2400 games as sample games). The comparisons are made on median runtimes; percentage of samples solved, and average runtime. It has been shown that both the median runtime and the conditional average runtime of algorithm 1 are much shorter than that of Lemke-Howson on all the 24 classes. Furthermore, our algorithm finds Nash Equilibrium in more sample games on several distributions, and only the solving percentage on distribution 6 is a little lower than Lemke-Howson.

5. Conclusions

In this paper, we present a search method for finding a simple Nash Equilibrium in a 2-player game. We analyze the payoff matrix of each player to prune the support space, and order the support space to ensure the simple and balanced NE can be found as fast as possible. After testing our algorithm on 24 classes of games and comparing it with the classical Lemke-Howson algorithm, the results show our algorithm performs better.

Acknowledgments

This research benefits from the support of NSFC Major Program/China (60496321), NSFC/China (60373098, 60573073), 863 Hi-Tech Research and Development Program/China (2003AA118020), Major Program of Jilin Province Development Plan for Science and Technology/China (20020303), General Program of Jilin Province Development Plan for Science and Technology/China (20030523).

References

[1] Rubinstein, A. (1998), *Modeling Bounded Rationality*, MIT Press, Cambridge.
[2] Dickhaut, J. and Kaplan, T. (1991), A program for finding Nash equilibria, *The Mathematica Journal*, 1:4, pp. 87-93.
[3] Porter, R, Nudelman, E. and Shoham Y. (2004), Simple search methods for finding a Nash Equilibrium, *American Association for Artificial Intelligence*.
[4] Lemke, C. and Howson,J. (1964), Equilibrium points of bimatrix games, *Journal of the Society for Industrial and Applied Mathematics*, 12, pp. 413-423.
[5] Lipton, R., Markakis, E. and Mehta, A. (2003), Playing large games using simple strategies, *ACM Conference on Electronic Commerce*, 36-41.
[6] Chvatal, V. (1983), *Linear Programming*, Freeman, New York.
[7] McKelvey, R., McLennan, A., and Turocy, T. (2004), *Gambit: Software tools for game theory*, Available at http:// econweb.tamu.edu/gambit/.

Shu-yang Sun received the B.S. and M.S. degrees in Computational Mathematics from Dept. Mathematics of Jilin University in 1996 and 1998, respectively. From 1998 till now, she stays in the College. of Computer Science and technology, Jilin University. Her research interests include: Game Theory, Data Mining, Machine Learning, Rough Set, etc.

Da-You Liu Professor. received the B.S. degree in Optics from Dept. Physics of Jilin University in 1966, and the M.S. degree in Computer Technology from Dept. Computer Science of Jilin University in 1981. His research interests include: Knowledge Engineering and Expert System, Multi Agent、Mobile Agents System and Intelligent Agent, Spatial-Temporal Knowledge Representation and Reasoning, Data Structure and Computing Algorithm, Rough Set and Data Mining, Data Structure and Computing Algorithm.

Cheng-Min Sun received the B.S. and M.S. degrees in Computer Technology from College of Computer Science and Technology of Jilin University in 1998 and 2001, respectively. From 2001 till now, she stays in the College. of Computer Science and technology, Jilin University. Her research interests include: Rough Set, Data Mining, Machine Learning, etc.