Linear Solvers for Nonlinear Games: Using Pivoting Algorithms to Find Nash Equilibria in n-Player Games

JAMES R. WRIGHT
and
ALBERT XIN JIANG
and
KEVIN LEYTON-BROWN
Department of Computer Science
University of British Columbia

Nash equilibria of two-player games are much easier to compute in practice than those of n-player games, even though the two problems have the same asymptotic complexity. We used a recent constructive reduction to solve general games using a two-player algorithm. However, the reduction increases the game size too much to be practically usable. An open problem is to find a more compact constructive reduction, which might make this approach feasible.

Categories and Subject Descriptors: F.2.0 [**Analysis of algorithms and problem complexity**]: General

General Terms: Algorithms, Theory, Performance, Experimentation

Additional Key Words and Phrases: Nash equilibrium

It is well known that finding a Nash equilibrium in a two-player game is asymptotically no easier than finding an equilibrium of an n-player game for $n > 2$ [Chen et al. 2009]. This is surprising, since payoffs in a two-player game are linear in the mixed strategy of the opposing player, unlike in an n-player game. Performing this computation in practice using the best known solvers, this linearity makes it much easier to solve two-player games than n-player games. This presents a puzzle: if the asymptotic complexities are the same, then why are the empirical performances so different? Can we leverage fast existing algorithms for two-player games to address the n-player case?

Lemke and Howson [1964] used linear algebra pivoting operations to rapidly find a sample Nash equilibrium in two-player games. Howson [1972] extended this algorithm to polymatrix games, which also have a linear structure. For other games, a more general algorithm such as the global Newton method [Govindan and Wilson 2003] must be used. The global Newton method (GNM) is a strict generalization of Lemke-Howson, in the sense that for a given two-player game and an appropriately chosen starting point, it will follow the same path through strategy space.[1] However-

[1]The GNM operates in a strategy $\times$ game space, so strictly speaking it is the projection of the

Authors' addresses: {jrwright,jiang,kevinlb}@cs.ubc.ca

ever, the GNM does so much less efficiently, since it uses numerical path-following techniques rather than pivoting operations. This makes a big difference to practical performance; for example, Govindan and Wilson proposed to solve multiple polymatrix games just to find a good starting point for the GNM [Govindan and Wilson 2004].

Initially, it was not known how to exploit the theoretical equivalence between two-player and n-player games, because the reductions used to prove the equivalence were not constructive. However, Feige and Talgam-Cohen [2010] recently presented a direct reduction from n-player games to two-player games, via polymatrix games. This suggests a way to efficiently compute a sample equilibrium in n-player games: first "linearize" the game using the constructive reduction, then apply a linearity-exploiting algorithm to the linearized game. In this letter we describe an investigation of this approach. See the appendix for implementation details and timing results.

The Feige-Talgam-Cohen (henceforth FTC) reduction converts an arbitrary normal-form game into an approximately equivalent polymatrix game. It does this by introducing *mediator agents*, each of which plays a single strategy with approximately the same probability that a specific pure strategy profile will be played, based on the strategies of the original players. The original players' payoffs then depend only on linear combinations of their payoffs for each pure strategy profile of the other players, and so the reduced game is a polymatrix game, which can then be solved using the Howson algorithm.

Overall, we observed that this method was a very slow way of computing a sample Nash equilibrium. The slow performance was attributable to the large number of additional agents introduced by the reduction. Specifically, one mediator agent is introduced for each i-incomplete pure strategy profile (i.e., each profile of pure strategies for each agent other than i); hence $\Omega(nk^{n-1})$ mediator agents are required, where k is the number of pure strategies for each agent. The exponential increase in the number of agents means that the linear algorithm must operate on strategy profiles of length $\Omega(nk^n)$. In contrast, the GNM algorithm, although it cannot exploit linearity, need only operate in an $O(n)$-dimensional space.

One might wonder how a polynomial reduction could produce an exponential number of agents. The reduction generates $\Omega(nk^n)$ constant-sized polymatrix payoff matrices for the polymatrix version of the game. These matrices are then combined into an $\Omega(nk^n) \times \Omega(nk^n)$ two-player payoff matrix, which is polynomial in the size of the original normal-form game ($\Omega(k^n)$). Crucially, a normal-form representation for the reduced game, which would be of size $\Omega(k^{nk^n})$, is never generated.

We note that the FTC reduction was never intended to be compact in terms of the number of agents. A reduction which was improved to be both constructive and compact could make this approach a feasible way to efficiently compute a sample Nash equilibrium in n-player games. It may also be worthwhile to consider reductions from *compactly* represented games instead of from games in normal form. Daskalakis et al. [2006] showed a polynomial reduction from instances of any compact representation satisfying certain reasonable properties to two-player games. However, this reduction is indirect. If the FTC approach could be adapted to take

path onto strategy space that is the same.

compactly represented games as inputs, we would have a constructive reduction that generates a number of agents polynomial in the size of the representation.

REFERENCES

Chen, X., Deng, X., and Teng, S. 2009. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM (JACM) 56,* 3, 1–57.

Daskalakis, C., Fabrikant, A., and Papadimitriou, C. 2006. The game world is flat: The complexity of Nash equilibria in succinct games. *Automata, Languages and Programming*, 513–524.

Feige, U. and Talgam-Cohen, I. 2010. A direct reduction from k-player to 2-player approximate Nash equilibrium. *Arxiv preprint arXiv:1007.3886*.

Govindan, S. and Wilson, R. 2003. A global Newton method to compute Nash equilibria. *Journal of Economic Theory 110,* 1, 65–86.

Govindan, S. and Wilson, R. 2004. Computing Nash equilibria by iterated polymatrix approximation. *Journal of Economic Dynamics and Control 28,* 7, 1229–1241.

Howson, J. 1972. Equilibria of polymatrix games. *Management Science 18,* 5, 312–318.

Lemke, C. and Howson, J. 1964. Equilibrium points of bimatrix games. *Journal of the Society for Industrial and Applied Mathematics 12,* 2, 413–423.

McKelvey, R., McLennan, A., and Turocy, T. 2007. Gambit: Software tools for game theory, version 0.2007. 01.30.

Nudelman, E., Wortman, J., Shoham, Y., and Leyton-Brown, K. 2004. Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*. 880–887.

A. APPENDIX

We used Gamut [Nudelman et al. 2004] to generate 10 uniform random games each of $3, 4, \ldots, 18$ players. Each player had two pure strategies. We converted each game to a polymatrix game using the FTC reduction. We then applied Howson's [1972] algorithm to the converted game. We refer to this procedure as reduced-Howson.

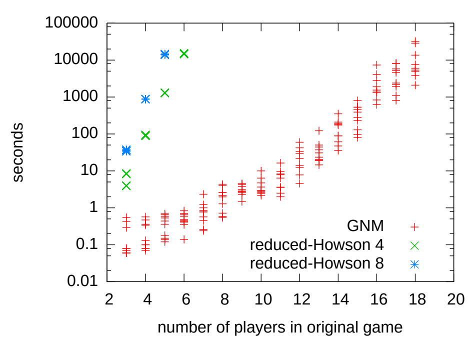

Fig. 1. Time to find a sample Nash equilibrium using GNM and reduced-Howson.

Feige and Talgam-Cohen [2010] provide reductions using a multiplication gadget that operates on binary representations of probabilities, and another, much simpler version that operates on unary representations. Both versions take an approximation precision as a parameter. At low precisions, the unary version uses fewer auxiliary agents, so we used the unary version with precisions of both 4 and 8 unary digits (equivalent to 2 or 3 binary digits, respectively). This is very imprecise! But even these approximations turned out to be quite computationally expensive. Figure 1 compares the running times of reduced-Howson (with both approximation precisions) to those of the GNM solver provided in Gambit [McKelvey et al. 2007]. The GNM solver completed for all but three of our 160 test games. The reduced-Howson solvers with 4 and 8 digits of precision solved 40 and 30 games respectively;

the remaining games either timed out or ran out of memory. The run times for reduced-Howson were orders of magnitude longer than those for GNM, illustrating the performance impact of the exponential increase in agents described above.