A Technique for Reducing Normal-Form Games to Compute a Nash Equilibrium*

Vincent Conitzer
Carnegie Mellon University
Computer Science Department
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
conitzer@cs.cmu.edu

Tuomas Sandholm
Carnegie Mellon University
Computer Science Department
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
sandholm@cs.cmu.edu

ABSTRACT

We present a technique for reducing a normal-form (aka. (bi)matrix) game, O, to a smaller normal-form game, R, for the purpose of computing a Nash equilibrium. This is done by computing a Nash equilibrium for a subcomponent, G, of O for which a certain condition holds. We also show that such a subcomponent G on which to apply the technique can be found in polynomial time (if it exists), by showing that the condition that G needs to satisfy can be modeled as a Horn satisfiability problem. We show that the technique does not extend to computing Pareto-optimal or welfare-maximizing equilibria. We present a class of games, which we call ALAGIU (Any Lower Action Gives Identical Utility) games, for which recursive application of (special cases of) the technique is sufficient for finding a Nash equilibrium in linear time. Finally, we discuss using the technique to compute *approximate* Nash equilibria.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems; J.4 [**Computer Applications**]: Social and Behavioral Sciences—*Economics*; F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

General Terms

Algorithms, Economics, Theory

Keywords

Game Theory, Computing Nash Equilibria, Preprocessing

*This material is based upon work supported by the National Science Foundation under ITR grants IIS-0121678 and IIS-0427858, a Sloan Fellowship, and an IBM Ph.D. Fellowship.

AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.

1. INTRODUCTION

Nash equilibrium is the most important solution concept for games. A vector of mixed strategies for the players (that is, for each player, a probability distribution over that player's actions in the game) is a Nash equilibrium if no individual player can benefit from changing her strategy. Every finite game has at least one Nash equilibrium [20]. However, for the concept to be operational, the mere existence result is not sufficient: it needs to be accompanied by an *algorithm* for finding an equilibrium. Unfortunately, even in 2-player normal-form games, it is still unknown whether a polynomial-time algorithm exists for computing a Nash equilibrium. The *Lemke-Howson* algorithm [15] is the best-known algorithm for computing a Nash equilibrium in a 2-player normal-form game, but it can require exponential time [25]. Some special cases can be solved in polynomial time (for example, zero-sum games [18]); other related questions, such as finding a welfare-maximizing equilibrium, are NP-hard [11, 5].

Recently, there has been a renewed surge of interest in the computation of Nash equilibria. Christos Papadmitriou has called the basic problem "a most fundamental computational problem whose complexity is wide open" and "together with factoring, [...] the most important concrete open question on the boundary of P today" [22],[1] and new (exponential-time) algorithms have been suggested that search over the supports of the mixed strategies [23, 24]. Also, there has been growing interest in computing equilibria of games with special structure that allows them to be represented concisely [13, 16, 2, 12, 1].

The idea we pursue in this paper is the following. For the basic problem of computing a Nash equilibrium of a normal-form game, it would be helpful to have a recursive technique that decomposes a Nash equilibrium computation problem into one or more smaller such problems, in such a way that a solution to the original problem can easily be computed from the solutions to the smaller problems. If there were such a technique that could be applied to *any* game, and that de-

[1] A recent sequence of papers [8, 3, 9, 4] shows that the problem of finding a Nash equilibrium (even in the two-player case) is complete for a class of problems called PPAD. This makes it seem less likely that a polynomial-time algorithm for finding a Nash equilibrium exists; on the other hand, unlike (say) NP, PPAD is not known to contain many problems of interest, so that it would be less of a shock if PPAD turned out to be contained in P.

composed it into small enough subproblems, then repeatedly applying this technique would constitute a polynomial-time algorithm for computing a Nash equilibrium. However, even if the technique could not be applied to all games, it would still be of interest. It could be used as a preprocessing step in computing a Nash equilibrium, thereby reducing the load on the algorithm that is eventually called to solve the irreducible subproblems. The technique could also be applied at intermediate stages of any other algorithm that works by reducing the size of the game. Moreover, there may be special subclasses of games such that the technique can be applied on any of those games, as well as on any subproblem resulting from applying the technique on those games. In that case, the technique would constitute a polynomial-time Nash equilibrium finding algorithm for the particular subclass of games.

In this paper, we introduce such a technique for 2-player normal-form games. When possible, this technique finds a subcomponent, G, of the original game, O, for which a certain condition holds. G is then solved recursively. Based on the recursively computed equilibrium of G, the original game O is then reduced to a smaller game, R, which is also solved recursively. From the computed equilibria of G and R, an equilibrium of the original game O can then easily be constructed. To our knowledge, this is the first recursive technique for computing a Nash equilibrium (other than the iterated elimination of dominated strategies[2]).

2. MAIN TECHNIQUE

In this section, we introduce our main technique. We are given a 2-player normal-form game O, in which the row player chooses a pure strategy from Σ_1 and the column player chooses a pure strategy from Σ_2. Suppose that the strategies in Σ_1 can be labeled as u_i, s_i, and those in Σ_2 as v_j, t_j, so that the game can be written as follows:

	v_1	v_2	$\cdots$	v_l	t_1	t_2	$\cdots$	t_n
u_1					c_{11}, b_1	c_{12}, b_1	$\cdots$	c_{1n}, b_1
u_2					c_{21}, b_2	c_{22}, b_2	$\cdots$	c_{2n}, b_2
$\vdots$		H			$\vdots$	$\vdots$		$\vdots$
u_k					c_{k1}, b_k	c_{k2}, b_k	$\cdots$	c_{kn}, b_k
s_1	a_1, d_{11}	a_2, d_{12}	$\cdots$	a_l, d_{1l}				
s_2	a_1, d_{21}	a_2, d_{22}	$\cdots$	a_l, d_{2l}				
$\vdots$	$\vdots$	$\vdots$		$\vdots$		G		
s_m	a_1, d_{m1}	a_2, d_{m2}	$\cdots$	a_l, d_{ml}				

The condition that allows for writing the game like this is the following:

CONDITION 1. *Against any fixed v_j, all the s_i give the row player the same utility (a_j); and against any fixed u_i, all the t_j give the column player the same utility (b_i).*

The intuition for why this condition is useful is the following. Suppose that the row player has decided to play one of the s_i, but has not yet decided which s_i. If the column player plays one of the v_j, then it does not matter which of the s_i the row player plays. Thus, in deciding which s_i to play, the row player only needs to consider the probabilities that the column player places on the t_j. Symmetrically, if the column player has decided to play one of the t_j (but has not yet decided which t_j), then the column player only needs to consider the probabilities that the row player places on the s_i. Hence, intuitively, the parts of the players' mixed strategies that concern the s_i and t_j should themselves be in equilibrium.

So, let us compute a Nash equilibrium of the game G (using any technique):

	t_1	t_2	t_3	$\cdots$	t_n
s_1					
s_2					
s_3			G		
$\vdots$					
s_m					

Let $p^G_{s_i}$ be the row player's probabilities in that Nash equilibrium, and let $p^G_{t_j}$ be the column player's probabilities. Also, let π^G_s be the expected utility for the row player, and let π^G_t be the expected utility for the column player. We can use these to reduce the original game O, as follows. We replace the lower left quadrant of O with a single row. The row player's payoff in the jth entry is a_j (which is the row player's payoff in the jth entry for any of the original rows). The column player's payoff in the jth entry is $\sum_{i=1}^{m} p^G_{s_i} d_{ij}$—that is, the average of the column player's payoffs in the jth entry, taken across the replaced rows, weighted by the probability with which the corresponding s_i is played in the computed equilibrium of G. We replace the upper right quadrant of O with a single column in a similar fashion. Finally, we replace the lower right quadrant (G) with a single entry π^G_s, π^G_t (the expected utilities of the players in the computed equilibrium of G). Thus, the reduced game R is the following:

	v_1	v_2	$\cdots$	v_l	t
u_1					$\sum_{j=1}^{n} p^G_{t_j} c_{1j}, b_1$
u_2					$\sum_{j=1}^{n} p^G_{t_j} c_{2j}, b_2$
$\vdots$		H			$\vdots$
u_k					$\sum_{j=1}^{n} p^G_{t_j} c_{kj}, b_k$
s	$a_1, \sum_{i=1}^{m} p^G_{s_i} d_{i1}$	$a_2, \sum_{i=1}^{m} p^G_{s_i} d_{i2}$	$\cdots$	$a_l, \sum_{i=1}^{m} p^G_{s_i} d_{il}$	π^G_s, π^G_t

We then compute a Nash equilibrium for R (using any technique), obtaining equilibrium probabilities $p^R_{u_i}, p^R_s$ for the row player and equilibrium probabilities $p^R_{v_j}, p^R_t$ for the column player. Then (as we will show later in the section), setting $p^O_{u_i} = p^R_{u_i}, p^O_{s_i} = p^R_s p^G_{s_i}, p^O_{v_j} = p^R_{v_j}$, and $p^O_{t_j} = p^R_t p^G_{t_j}$ constitutes a Nash equilibrium of O.

Before proving formally that the technique is correct, we first illustrate it with a small example. Consider the following game O:

[2]Most variants of iterated dominance can be executed in polynomial time [14, 19, 10, 6]. We have recently proposed a more general strategy eliminability criterion that can help in computing a Nash equilibrium [7], but it is not a recursive technique.

	v_1	t_1	t_2
u_1	$2,2$	$0,3$	$2,3$
s_1	$1,2$	$4,0$	$0,4$
s_2	$1,4$	$0,4$	$4,0$

Condition 1 holds for this game. Thus, we first solve the subcomponent G:

	t_1	t_2
s_1	$4,0$	$0,4$
s_2	$0,4$	$4,0$

This is a matching-pennies game where in equilibrium, each player places probability $1/2$ on each pure strategy and receives expected utility 2.[3] Thus, the reduced game R becomes:

	v_1	t
u_1	$2,2$	$1,3$
s	$1,3$	$2,2$

Again, this is a matching-pennies game where in equilibrium, each player places probability $1/2$ on each action. Thus, we have discovered an equilibrium for the original game O where u_1 and v_1 are played with probability $1/2$ each, and s_1, s_2, t_1, t_2 are played with probability $(1/2) \cdot (1/2) = 1/4$ each. We now prove the technique is correct in general.

THEOREM 1. *Suppose O is reduced to R using the equilibrium $p^G_{s_i}, p^G_{t_j}$ of G. If $p^R_{u_i}, p^R_s, p^R_{v_j}, p^R_t$ constitute a Nash equilibrium of R, then setting $p^O_{u_i} = p^R_{u_i}, p^O_{s_i} = p^R_s p^G_{s_i}, p^O_{v_j} = p^R_{v_j}$, and $p^O_{t_j} = p^R_t p^G_{t_j}$ constitutes a Nash equilibrium of O.*

Proof. To show that these mixed strategies constitute a Nash equilibrium for O, we first observe that the row player has no incentive to redistribute the probability placed on strategies s_i to another strategy $s_{i'}$. That is, *given* that the row player plays one of the strategies s_i (note that given this, the probability that the row player plays a given strategy s_i is $p^R_s p^G_{s_i} / p^R_s = p^G_{s_i}$), there is no incentive to switch to another strategy $s_{i'}$. The reason is as follows. Given that the column player plays a given strategy v_j, both $p^G_{s_i}$ and $s_{i'}$ will give the same utility. Given that the column player plays one of the strategies t_j (note that given this, the probability that the column player plays a given strategy t_j is $p^R_t p^G_{t_j} / p^R_t = p^G_{t_j}$), $p^G_{s_i}$ is a best response because $p^G_{s_i}, p^G_{t_j}$ constitute a Nash equilibrium for G. As a result, to show that we have an equilibrium for O, we do not need to consider deviations to an arbitrary s_i; instead, we only need to consider deviations to the mixed strategy $p^G_{s_i}$.

Next, we observe that the expected utility for the row player of playing a given u_i is the same as in the equilibrium computed for R. The reason is as follows. The probability that the column player plays a given v_j is the same in both R and O, and so is the row player's utility for the outcome (u_i, v_j). Moreover, the probability that the column player plays one of the strategies t_j in O is p^R_t, and the expected utility for the row player of playing u_i in that case is $\sum_{j=1}^{n} p^G_{t_j} c_{ij}$, which is the same as in R.

We also observe that the expected utility for the row player of playing $p^G_{s_i}$ (that is, the row player's mixed strategy given that the row player plays one of the strategies s_i) is the same as the expected utility of playing s in the equilibrium computed for R. The reason is as follows. The probability that the column player plays a given v_j is the same in both R and O, and the row player's utility in either case is a_j. Moreover, the probability that the column player plays one of the strategies t_j in O is p^R_t, and the expected utility for the row player of playing $p^G_{s_i}$ in that case is the expected utility the row player gets in the equilibrium of G, namely, π^G_s, which is the same as the row player's utility for the outcome (s, t) in R.

But from these last two observations, and the fact that we used an equilibrium for R, it follows that the row player has no incentive to deviate to any u_i or to $p^G_{s_i}$; and from the first observation, the fact that the row player has no incentive to deviate to $p^G_{s_i}$ implies that the row player has no incentive to deviate to any s_i. Hence, the row player has no incentive to deviate; by symmetry, neither does the column player. ■

3. DETECTING WHETHER THE TECHNIQUE CAN BE APPLIED

It is easy to verify whether Condition 1 holds when the pure strategies of the game are labeled u_i, s_i, v_j, t_j. However, in general, this labeling will not be given to us. As an example, suppose we are given the following game:

	β_1	β_2	β_3
α_1	$4,0$	$1,2$	$0,4$
α_2	$0,3$	$2,2$	$2,3$
α_3	$0,4$	$1,4$	$4,0$

This game is, in fact, the same game that was used as an example in Section 2, with the rows and columns permuted $(\alpha_1 = s_1, \alpha_2 = u_1, \alpha_3 = s_2, \beta_1 = t_1, \beta_2 = v_1, \beta_3 = t_2)$. But we cannot apply the technique without knowing the labeling of the strategies as u_i, s_i, v_j, and t_j.

Hence, we need an algorithm that finds a labeling of the strategies as u_i, s_i, v_j, and t_j such that Condition 1 holds. Note that it does not matter whether we label a given strategy (*e.g.*) s_1 or s_2. That is, all we need to do is partition the row player's strategies Σ_1 into two subsets, the ones labeled u_i and the ones labeled s_i; and partition the column player's strategies Σ_2 into two subsets, the ones labeled v_j and the ones labeled t_j.

There are two trivial ways of labeling the strategies so that the condition holds: 1. Label all strategies s_i or t_j. 2. Label at most one strategy s_i, and at most one strategy t_j. These trivial labelings are not useful: the first would give us $G = O$ to recurse on, and the second would not reduce the game at all $(R = O)$. Any other labeling for which Condition 1 is satisfied, though, would give us a game G that is smaller than O to recurse on, and reduce the game to a game R that is smaller than O. Moreover, the number of rows (columns) by which R is smaller than O is equal to

[3]We emphasize that there is no restriction that the game G or R should be a zero-sum game (or any other special type of game).

the number of rows (resp. columns) in G, minus 1. Hence, our objective is to find a labeling for which the condition is satisfied, other than the two trivial ones described above.

We now show how to reformulate this problem as a Horn satisfiability problem. (Recall that a satisfiability clause is a Horn clause if it can be written as $(\neg x_1 \vee \neg x_2 \vee \ldots \vee \neg x_n \vee y)$, or equivalently, $x_1 \wedge x_2 \wedge \ldots \wedge x_n \Rightarrow y$.) Let the variable $v(\sigma)$ be *true* if σ is labeled as one of the s_i or t_j, and *false* if σ is labeled as one of the u_i or v_j. The key observation is that to satisfy Condition 1, if row player strategies α_1 and α_2 obtain different payoffs for the row player against column player strategy β, then it cannot be the case that α_1 and α_2 are both labeled as one of the s_i and β is labeled as one of the v_j—in other words, $v(\alpha_1) \wedge v(\alpha_2) \Rightarrow v(\beta)$. Similarly, if column player strategies β_1 and β_2 obtain different payoffs for the column player against row player strategy α, then $v(\beta_1) \wedge v(\beta_2) \Rightarrow v(\alpha)$.

Using this, the game described at the beginning of this section has the following Horn clauses:[4]

1. $v(\alpha_1) \wedge v(\alpha_2) \Rightarrow v(\beta_1) \wedge v(\beta_2) \wedge v(\beta_3)$,
2. $v(\alpha_1) \wedge v(\alpha_3) \Rightarrow v(\beta_1) \wedge v(\beta_3)$,
3. $v(\alpha_2) \wedge v(\alpha_3) \Rightarrow v(\beta_2) \wedge v(\beta_3)$,
4. $v(\beta_1) \wedge v(\beta_2) \Rightarrow v(\alpha_1) \wedge v(\alpha_2)$,
5. $v(\beta_1) \wedge v(\beta_3) \Rightarrow v(\alpha_1) \wedge v(\alpha_3)$,
6. $v(\beta_2) \wedge v(\beta_3) \Rightarrow v(\alpha_1) \wedge v(\alpha_2) \wedge v(\alpha_3)$.

It is straightforward to check that the only nontrivial satisfying assignment, that is, the only assignment that

- satisfies all these clauses,
- does not set all variables to *true*, and
- sets either at least two $v(\alpha_i)$ to *true* or at least two $v(\beta_j)$ to *true*;

sets $v(\alpha_1), v(\alpha_3), v(\beta_1)$, and $v(\beta_3)$ to *true* and everything else to *false*. We note that this corresponds to the labeling that we presented. The Horn clauses can be computed efficiently:

LEMMA 1. *The set of Horn clauses for a normal-form game can be computed in time* $O(|\Sigma_1|^2 \cdot |\Sigma_2| + |\Sigma_1| \cdot |\Sigma_2|^2)$.

We now show that the Horn clauses fully capture the problem, that is, the Horn clauses being satisfied is a necessary and sufficient condition for Condition 1 to be satisfied.

THEOREM 2. *A labeling of the strategies as* u_i, s_i, v_j, *and* t_j *satisfies Condition 1 if and only if it satisfies all the Horn clauses.*

Proof. First, suppose that not all of the Horn clauses are satisfied. Then (without loss of generality) there exist $\alpha_{i_1}, \alpha_{i_2}$, and β_j such that $v(\alpha_{i_1}) \wedge v(\alpha_{i_2}) \Rightarrow v(\beta_j)$, $v(\alpha_{i_1})$ and $v(\alpha_{i_2})$ are set to *true*, and $v(\beta_j)$ is set to *false*. (Alternatively the roles of the α_i and β_j could be reversed, but by symmetry we can without loss of generality restrict our attention to the first case.) That is, α_{i_1} and α_{i_2} are among the strategies s_i, β_j is among the strategies v_j, and α_{i_1} and α_{i_2} give the row player different payoffs against β_j. But then Condition 1 is not satisfied.

Now, suppose that all of the Horn clauses are satisfied. We must show, for any v_j, that any s_{i_1} and s_{i_2} give the row player the same payoff against v_j. (We must show the same for the u_i and the t_j, but this will follow by symmetry.) Suppose they do not; then one of the Horn clauses must be $v(s_{i_1}) \wedge v(s_{i_2}) \Rightarrow v(v_j)$. But this clause would then be *false*, which is contrary to our assumption. ■

A general systematic approach to finding a nontrivial satisfying assignment is to start by setting two $v(\alpha_i)$ (or two $v(\beta_j)$) to *true*, and subsequently iteratively apply the implication clauses; if this process ends without all the variables being set to *true*, we have found an assignment with the desired properties. If we do this once for every initial pair of $v(\alpha_i)$ and every initial pair of $v(\beta_j)$, then we will find an assignment with the desired properties if it exists.

For instance, in the above example, we start by setting $v(\alpha_1)$ and $v(\alpha_2)$ to *true*; then, applying the first implication, we find that all the $v(\beta_j)$ must be set to *true* as well, and hence by the last implication so must $v(\alpha_3)$, so that all variables are set to *true*. Then, we try again, starting by setting $v(\alpha_1)$ and $v(\alpha_3)$ to *true*; by the second implication, we find that $v(\beta_1)$ and $v(\beta_3)$ must also be set to *true*. Then we apply the fifth implication, but this does not set any new variables to *true*, so the process ends here, and we have found an assignment with the desired properties.

THEOREM 3. *The algorithm described above requires* $O((|\Sigma_1| + |\Sigma_2|)^4)$ *applications of a Horn clause.*

Proof. There are $O((|\Sigma_1| + |\Sigma_2|)^2)$ iterations of the outer loop (choosing which pair of variables to start with). Moreover, within each iteration, we never need to apply the same clause twice, and there are only $O((|\Sigma_1| + |\Sigma_2|)^2)$ clauses. ■

4. LIMITATIONS OF THE TECHNIQUE

One may wonder whether the technique can be extended to find Nash equilibria with special properties. For example, if we compute a Pareto optimal or social-welfare maximizing Nash equilibrium in each recursive call, will this give us a Pareto optimal or social-welfare maximizing Nash equilibrium for the original game? The following game O shows that this is not the case.

	v_1	t_1	t_2
u_1	$1,1$	$4,0$	$0,0$
s_1	$0,4$	$3,3$	$0,0$
s_2	$0,0$	$0,0$	$2,2$

The subcomponent G is as follows:

	t_1	t_2
s_1	$3,3$	$0,0$
s_2	$0,0$	$2,2$

It has multiple equilibria, but the only Pareto optimal Nash equilibrium is (s_1, t_1). Thus the reduced game R becomes:

[4]When we have Horn clauses $v(\alpha_1) \wedge v(\alpha_2) \Rightarrow v(\beta_1)$ and $v(\alpha_1) \wedge v(\alpha_2) \Rightarrow v(\beta_2)$, we aggregate them into $v(\alpha_1) \wedge v(\alpha_2) \Rightarrow v(\beta_1) \wedge v(\beta_2)$ to improve readability.

	v_1	t
u_1	$1,1$	$4,0$
s	$0,4$	$3,3$

This is a Prisoner's Dilemma game where the only equilibrium is (u_1, v_1), and thus we will compute the equilibrium (u_1, v_1) for the original game O. Unfortunately, this is not a Pareto optimal equilibrium for O, because (s_2, t_2) is also a Nash equilibrium for O. Of course, in practice, choosing a Nash equilibrium with high social welfare in each recursive call may still be a useful heuristic for finding good equilibria.

5. ALAGIU GAMES: A CLASS OF GAMES THAT CAN BE SOLVED BY APPLYING THE TECHNIQUE RECURSIVELY

In general, we may only be able to apply the technique a limited number of times, after which we need to resort to other algorithms to obtain Nash equilibria for games in the recursive calls on which the technique cannot be applied. However, in this section, we present a subclass of normal-form games such that the technique works on any of these games, and any game resulting from a recursive call is itself in the subclass as well. Thus, the technique is sufficient for finding a Nash equilibrium.

DEFINITION 1. *We say that a 2-player normal-form game is an* ALAGIU (Any Lower Action Gives Identical Utility) *game if each player's strategy set Σ_i is a subset of $\mathbb{R}$, and the game has the property that for any pure opponent strategy $\sigma_{-i} \in \Sigma_{-i}$, for any $\sigma_i^1, \sigma_i^2 \in \{\sigma_i \in \Sigma_i : \sigma_i < \sigma_{-i}\}$, we have $\pi_i(\sigma_i^1, \sigma_{-i}) = \pi_i(\sigma_i^2, \sigma_{-i})$. That is, given that a player chooses a lower strategy than the opponent, it does not matter to that player which of the lower strategies she chooses.*

Before showing how the technique can be applied to (finite) ALAGIU games, we first give some examples. The first one is very simple:

EXAMPLE 1 (MATCHING n-SIDED PENNIES). *In matching n-sided pennies, both players choose an integer in $\{1, 2, \ldots, n\}$; player 1 wins if the players' integers are the same, and player 2 wins if they are different.*

Incidentally, this example shows that ALAGIU games can have nontrivial equilibria:

PROPOSITION 1. *There exist ALAGIU games (of arbitrary size) that have only one equilibrium in which each player randomizes uniformly over all strategies.*

PROOF. We claim that matching n-sided pennies has the property. This is a zero-sum game, and thus any equilibrium strategy is a minimax strategy. For player 1, the only minimax strategy is to place probability $1/n$ on each integer, because if player 1 plays some integer with lower probability, the other player will be better off playing that integer. Similarly, for player 2, the only minimax strategy is to place probability $1/n$ on each integer, because if player 2 plays some integer with higher probability, the other player will be better off playing that integer. ■

We note that state-of-the-art equilibrium finding algorithms tend to have difficulty computing equilibria with large supports (*i.e.* in which the players randomize over many strategies). For example, the PNS algorithm [23] for finding a Nash equilibrium explicitly searches over all (exponentially many) possible supports, searching the smaller supports first. Other algorithms similarly tend to require more time on games that only have equilibria with large supports.

Matching n-sided pennies does not illustrate the full generality of ALAGIU games. The next (somewhat playful) example allows for richer structure:

EXAMPLE 2 (VENDORS ON A ONE-WAY STREET). *Say we have a hot dog vendor and an ice cream vendor that must each choose a location on a one-way street. Assume that, given that a vendor chooses a location earlier on the street than the other vendor, it does not matter to the earlier vendor where exactly she locates herself (all traffic must pass her anyway). However, it may matter to the later vendor where he locates himself. (For example, it may be bad for the ice cream vendor to immediately follow the hot dog vendor, because potential customers will have their hands full with the hot dog. In contrast, it may be good for the ice cream vendor to locate himself significantly later on the street than the hot dog vendor, to provide some dessert.) Then, this is an ALAGIU game.*

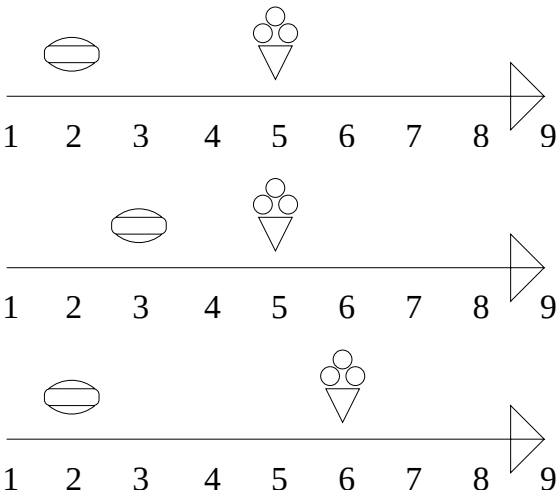

Figure 1: Three outcomes for the vendor game. Under the ALAGIU restriction, the hot dog vendor's utility must be the same for the top two outcomes. However, the ice cream vendor's utility need not be the same in any of the three outcomes.

More practically, our final example shows that ALAGIU games also encompass typical auctions.

EXAMPLE 3 (HIGHER-BIDDER-WINS AUCTIONS). *Any two-player single-item auction in which a lower bidder never wins and never pays anything is an ALAGIU game. Note that arbitrary allocation rules are allowed in case of ties. Also, arbitrary payment rules for the winning bidder are allowed, including first price and second price (but also bizarre,* e.g. *nonmonotonic, payment rules).*

Perhaps surprisingly, in contrast to auctions, *bargaining* games typically do not have the structure of ALAGIU games.[5]

[5]We thank Piotr Gmytrasiewicz for asking the question whether bargaining games are ALAGIU games.

We discuss this in more detail in the appendix, but we now move on to showing how ALAGIU games can be solved using the technique described in this paper. It turns out that any finite ALAGIU game takes one of three special forms described below, each of which satisfies Condition 1. Consider the highest strategy in the game, that is, $\sigma_M = \max\{\sigma : \sigma \in \Sigma_1 \cup \Sigma_2\}$. If only the row player has σ_M in her strategy set ($\sigma_M \in \Sigma_1 - \Sigma_2$), then the game must have the following form:

	t_1	t_2	t_3	$\cdots$	t_n
$u_1 = \sigma_M$	c_{11}, b_1	c_{12}, b_1	c_{13}, b_1	$\cdots$	c_{1n}, b_1
s_1					
s_2					
s_3			G		
$\vdots$					
s_m					

On the other hand, if only the column player has σ_M in his strategy set ($\sigma_M \in \Sigma_2 - \Sigma_1$), then the game must have the following form:

	$v_1 = \sigma_M$	t_1	t_2	t_3	$\cdots$	t_n
s_1	a_1, d_{11}					
s_2	a_1, d_{21}					
s_3	a_1, d_{31}			G		
$\vdots$	$\vdots$					
s_m	a_1, d_{m1}					

Finally, if both the row and the column player have σ_M in their strategy sets ($\sigma_M \in \Sigma_1 \cap \Sigma_2$), then the game must have the following form:

	$v_1 = \sigma_M$	t_1	t_2	t_3	$\cdots$	t_n
$u_1 = \sigma_M$	h_1, h_2	c_{11}, b_1	c_{12}, b_1	c_{13}, b_1	$\cdots$	c_{1n}, b_1
s_1	a_1, d_{11}					
s_2	a_1, d_{21}					
s_3	a_1, d_{31}			G		
$\vdots$	$\vdots$					
s_m	a_1, d_{m1}					

Our technique can be applied to every one of these three forms. Moreover, in each case the subcomponent G is still an ALAGIU game, and thus we can apply the technique recursively. The reduced game R is a 2×1, 1×2, or 2×2 game (for the three cases, respectively), so it can easily be solved.

THEOREM 4. *Using the technique described in this paper, a Nash equilibrium of a finite ALAGIU game can be computed in time $O(|\Sigma_1| \cdot |\Sigma_2|)$ (that is, in time linear in the size of the game).*

Proof. The subcomponent G that the algorithm recurses on is one row and/or column smaller than the current game. Whenever G is one row smaller (which happens $O(|\Sigma_1|)$ times), after the recursion we need to compute a weighted average of the c_{ij}, which requires $O(|\Sigma_2|)$ time. Similarly, whenever G is one column smaller (which happens $O(|\Sigma_2|)$ times), after the recursion we need to compute a weighted average of the d_{ij}, which requires $O(|\Sigma_1|)$ time. ■

6. USING THE TECHNIQUE TO COMPUTE APPROXIMATE EQUILIBRIA

It is known that *approximate* Nash equilibria can be computed faster than (we know how to compute) exact Nash equilibria [17]. In this section, we study how the technique can be used to compute approximate equilibria. Specifically, if we have found that there is no subcomponent G that satisfies the condition, we can *modify* the game so that in the modified game O', there does exist a subcomponent G' that satisfies the condition. Of course, the equilibria of O and O' are, in general, not the same, so that the equilibrium that we compute for O' using the technique is not actually an equilibrium for O. However, as we will show, if O' is close to O, then the equilibrium that we compute for O' is an approximate equilibrium for O. As our notion of distance between two games (of equal dimensions), we will use the maximum difference between two payoffs that occur in the same position in the matrices: $d(O, O') = \max_{i \in \{r,c\}} \max_{\alpha,\beta} |\pi_i^O(\alpha, \beta) - \pi_i^{O'}(\alpha, \beta)|$ (where π_r is the row player's utility function and π_c is the column player's utility function). We will use *ϵ-equilibrium* as our notion of approximate equilibrium. An ϵ-equilibrium is a pair of strategies for the agents such that neither can improve its expected payoff by more than ϵ by deviating.

LEMMA 2. *If O and O' have equal dimensions, and $d(O, O') \leq \epsilon/2$, then any equilibrium for O' is an ϵ-equilibrium for O.*

PROOF. Let p_r be the row player's mixed strategy in the computed equilibrium for O', and let p_c be the column player's mixed strategy in the computed equilibrium for O'. Let p'_r be a best response to p_c *in the original game* O. Because we computed an equilibrium, we have $\pi_r^{O'}(p_r, p_c) \geq \pi_r^{O'}(p'_r, p_c)$. Moreover, because $d(O, O') \leq \epsilon/2$, we have $\pi_r^O(p_r, p_c) \geq \pi_r^{O'}(p_r, p_c) - \epsilon/2$ and $\pi_r^{O'}(p'_r, p_c) \geq \pi_r^O(p'_r, p_c) - \epsilon/2$. Hence, $\pi_r^O(p_r, p_c) \geq \pi_r^{O'}(p_r, p_c) - \epsilon/2 \geq \pi_r^{O'}(p'_r, p_c) - \epsilon/2 \geq \pi_r^O(p'_r, p_c) - \epsilon$, as desired. By symmetry, the same holds for the column player. ■

For sufficiently large ϵ, we can always find a game O' with $d(O, O') \leq \epsilon$ for which there is a subcomponent G satisfying the condition. For example, if ϵ is larger than all of the payoffs in the game, then O' can simply be the game in which all payoffs are 0. Of course, this weakens the guarantee on how much the agents can gain by deviating. Thus, the challenge is to find a game O' that is as close as possible to O while still having a subcomponent G satisfying the condition. One greedy way of finding such a game is the following: for a given column (row), take two utilities that the row (column) player can obtain in this column (row) that are very close, and change both of them (wherever they occur in the column (row)) to the average of the two. Then, check if the modified game has a subcomponent G satisfying the condition using the technique in Section 3; if not, repeat with the modified game.

7. CONCLUSIONS

In this paper, we presented a technique for reducing a game, O, to a smaller game, R, for the purpose of computing a Nash equilibrium. This is done by computing a Nash equilibrium for a subcomponent, G, of O for which a certain

condition holds. We also showed that such a subcomponent G on which to apply the technique can be found in polynomial time (if it exists), by showing that the condition that G needs to satisfy can be modeled as a Horn satisfiability problem. We showed that the technique does not extend to computing Pareto-optimal or welfare-maximizing equilibria. We presented a class of games, which we call ALAGIU (Any Lower Action Gives Identical Utility) games, for which recursive application of (special cases of) the technique is sufficient for finding a Nash equilibrium in linear time. Finally, we discussed using the technique to compute *approximate* Nash equilibria.

Future research includes extending the techniques presented here to games with more than two players (we note, however, that from the viewpoint of complexity theory, the two-player case is as hard as the n-player case [4]). It also includes testing experimentally whether the technique is helpful in finding equilibria in games for which we do not have a theoretical result (as we do for ALAGIU games), for instance on test suites of game generators (such as GAMUT [21]). Of course, the games generated will need to have some structure in order for the technique to have a chance—for example, if the payoffs are drawn independently from an interval of real numbers, then the probability that the technique can be applied is 0, because the probability of any two payoffs being the same is 0. (However, this does not prevent us from searching for approximate equilibria using the technique, as described in Section 6.) Yet another avenue is to find other, complementary game reduction techniques that can be applied when the techniques presented here cannot. Ideally, this would lead to a portfolio of efficient techniques of which at least one can always be applied, giving us a polynomial-time algorithm for computing a Nash equilibrium.

8. REFERENCES

[1] N. A. R. Bhat and K. Leyton-Brown. Computing Nash equilibria of action-graph games. In *Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, Banff, Canada, 2004.

[2] B. Blum, C. R. Shelton, and D. Koller. A continuation method for Nash equilibria in structured games. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, 2003.

[3] X. Chen and X. Deng. 3-Nash is PPAD-complete. *Electronic Colloquium on Computational Complexity*, Report No. 134, 2005.

[4] X. Chen and X. Deng. Settling the complexity of 2-player Nash equilibrium. *Electronic Colloquium on Computational Complexity*, Report No. 150, 2005.

[5] V. Conitzer and T. Sandholm. Complexity results about Nash equilibria. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 765–771, Acapulco, Mexico, 2003.

[6] V. Conitzer and T. Sandholm. Complexity of (iterated) dominance. In *Proceedings of the ACM Conference on Electronic Commerce (ACM-EC)*, pages 88–97, Vancouver, Canada, 2005.

[7] V. Conitzer and T. Sandholm. A generalized strategy eliminability criterion and computational methods for applying it. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 483–488, Pittsburgh, PA, USA, 2005.

[8] C. Daskalakis, P. Goldberg, and C. Papadimitriou. The complexity of computing a Nash equilibrium. *Electronic Colloquium on Computational Complexity*, Report No. 115, 2005.

[9] C. Daskalakis and C. Papadimitriou. Three-player games are hard. *Electronic Colloquium on Computational Complexity*, Report No. 139, 2005.

[10] I. Gilboa, E. Kalai, and E. Zemel. The complexity of eliminating dominated strategies. *Mathematics of Operation Research*, 18:553–565, 1993.

[11] I. Gilboa and E. Zemel. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior*, 1989.

[12] G. Gottlob, G. Greco, and F. Scarcello. Pure Nash equilibria: hard and easy games. In *Theoretical Aspects of Rationality and Knowledge (TARK)*, pages 215–230, Bloomington, Indiana, USA, 2003.

[13] M. Kearns, M. Littman, and S. Singh. Graphical models for game theory. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2001.

[14] D. E. Knuth, C. H. Papadimitriou, and J. N. Tsitsiklis. A note on strategy elimination in bimatrix games. *Operations Research Letters*, 7(3):103–107, 1988.

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

[16] K. Leyton-Brown and M. Tennenholtz. Local-effect games. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, 2003.

[17] R. Lipton, E. Markakis, and A. Mehta. Playing large games using simple strategies. In *Proceedings of the ACM Conference on Electronic Commerce (ACM-EC)*, pages 36–41, San Diego, CA, 2003.

[18] R. D. Luce and H. Raiffa. *Games and Decisions*. John Wiley and Sons, New York, 1957. Dover republication 1989.

[19] R. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, Cambridge, 1991.

[20] J. Nash. Equilibrium points in n-person games. *Proc. of the National Academy of Sciences*, 36:48–49, 1950.

[21] E. Nudelman, J. Wortman, K. Leyton-Brown, and Y. Shoham. Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, New York, NY, USA, 2004.

[22] C. Papadimitriou. Algorithms, games and the Internet. In *Proceedings of the Annual Symposium on Theory of Computing (STOC)*, pages 749–753, 2001.

[23] R. Porter, E. Nudelman, and Y. Shoham. Simple search methods for finding a Nash equilibrium. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 664–669, San Jose, CA, USA, 2004.

[24] T. Sandholm, A. Gilpin, and V. Conitzer. Mixed-integer programming methods for finding Nash

equilibria. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 495–501, Pittsburgh, PA, USA, 2005.

[25] R. Savani and B. von Stengel. Exponentially many steps for finding a Nash equilibrium in a bimatrix game. In *Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS)*, 2004.

APPENDIX

In this appendix, we show that, in contrast to auctions, bargaining games typically do not have the structure of ALAGIU games, or even the structure required for the technique to be useful. Consider a dollar-splitting game in which each of two agents must simultaneously ask for a fraction of the dollar. If the fractions that the agents ask for sum to at most a dollar, they each receive their fraction; otherwise, neither agent receives anything. We note that this game *appears* to have somewhat similar structure to ALAGIU games, in the sense that if an agent asks for an amount that is too high, then it does not matter to the agent by how much the amount is too high. However, there is a subtle difference: informally, in (for example) auctions, bids are too low when the other bidder's bid is high, whereas in bargaining games such as this one, demands are too high if the other bargainer's demand is *also* high. We now show formally that the technique is in general not useful on bargaining games.

For simplicity, suppose that the agents can only ask for \$0.25, \$0.50, or \$0.75. Then the game is the following.

	\$0.25	\$0.50	\$0.75
\$0.25	$0.25, 0.25$	$0.25, 0.50$	$0.25, 0.75$
\$0.50	$0.50, 0.25$	$0.50, 0.50$	$0, 0$
\$0.75	$0.75, 0.25$	$0, 0$	$0, 0$

We will use the technique from Section 3 to show that there is no subcomponent G on which to apply the technique. Denoting by α_x agent 1's strategy of demanding x, and by β_x agent 2's strategy of demanding x, we obtain the following clauses:

1. $v(\alpha_{\$0.25}) \wedge v(\alpha_{\$0.50}) \Rightarrow v(\beta_{\$0.25}) \wedge v(\beta_{\$0.50}) \wedge v(\beta_{\$0.75})$,
2. $v(\alpha_{\$0.25}) \wedge v(\alpha_{\$0.75}) \Rightarrow v(\beta_{\$0.25}) \wedge v(\beta_{\$0.50}) \wedge v(\beta_{\$0.75})$,
3. $v(\alpha_{\$0.50}) \wedge v(\alpha_{\$0.75}) \Rightarrow v(\beta_{\$0.25}) \wedge v(\beta_{\$0.50})$,
4. $v(\beta_{\$0.25}) \wedge v(\beta_{\$0.50}) \Rightarrow v(\alpha_{\$0.25}) \wedge v(\alpha_{\$0.50}) \wedge v(\alpha_{\$0.75})$,
5. $v(\beta_{\$0.25}) \wedge v(\beta_{\$0.75}) \Rightarrow v(\alpha_{\$0.25}) \wedge v(\alpha_{\$0.50}) \wedge v(\alpha_{\$0.75})$,
6. $v(\beta_{\$0.50}) \wedge v(\beta_{\$0.75}) \Rightarrow v(\alpha_{\$0.25}) \wedge v(\alpha_{\$0.50})$.

It is straightforward to see that this set of clauses only has trivial solutions (all variables set to *true*, or at most one per player set to *true*). Hence the technique cannot be applied on this game (which also implies that the game is not an ALAGIU game).