Running Head: SIMPLY STABLE EQUILIBRIA

Author's address: Stanford Business School, Stanford CA 94305-5015 USA.

Computing Simply Stable Equilibria

Robert Wilson[1]

ABSTRACT: For each two-player game, a linear-programming algorithm finds a component of the Nash equilibria and a subset of its perfect equilibria that are simply stable: there are nearby equilibria for each nearby game that perturbs one strategy's probability or payoff more than others.

KEYWORDS: Equilibrium, stability, algorithm, linear complementarity.

Kohlberg and Mertens (1986) propose a refinement of Nash equilibria called stability. Basically, a set of equilibria is stable if every game nearby has equilibria nearby. They study several specifications of the neighborhood of a game and select the smallest for their definition; Mertens (1989, 1991) and Hillas (1990) study essentially stronger refinements. In this article we use a different, essentially smaller, neighborhood and therefore a weaker refinement called simple stability. A practical advantage of this refinement is that it enables an elementary procedure for computing a simply-stable set of equilibria of a two-player game.

A set of perfect equilibria within a single connected component is simply stable if each game obtained by perturbing some strategy's payoff or minimal probability has equilibria near this set. This criterion is weaker in that it considers only perturbations of pure strategies, rather than mixed strategies as in Kohlberg and Mertens. It is also mildly stronger in that it confines the set to perfect equilibria in a single component, and it allows perturbations of both probabilities and payoffs.[2] These features reflect partially

[1] NSF grants SES 8908269 and 9207850 provided financial support and Faruk Gul provided intellectual support. An STSC APL II version of the computer program is available from the author, and a faster C version has been prepared for the game solver Gambit by McKelvey (1990).

[2] As in Kohlberg and Mertens, perturbing a strategy's minimal probability perturbs

the motivations for Mertens' and Hillas' refinements.

We describe a numerical algorithm that constructs a simply-stable set comprising at most $2n$ extreme points of some component, where n is the number of pure strategies. Jansen, Jurg, and Borm (1990) for two-player games, and Mertens (1989) for N-player games, demonstrate that the task is finite in principle using a form of enumeration, even for Mertens' stronger definition (Hillas' seems unsuited to finite computation). Our contribution is an efficient algorithm based on standard methods and software of linear programming, as adapted to the calculation of equilibria of two-player games and other linear complementarity problems by Lemke and Howson (1964).

Kohlberg and Mertens (1986) and Kohlberg (1990) elaborate the economic significance of stability in terms of properties motivated by axiomatic considerations. They provide many examples and other authors have studied the application of stability criteria or its derivative properties to various economic contexts.[3] Studies of more realistic problems require an efficient algorithm. An algorithm is also essential for empirical studies of the many game-theoretic models developed to study imperfectly competitive markets (these models typically invoke refinements, although often weaker ones such as perfection). For instance, maximum-likelihood estimation of the parameters of a strategic model, and related hypothesis testing, require calculation of the equilibrium outcome for each parameter specification: Bresnahan and Reiss (1991) develop the statistical methodology of this approach to empirical studies.

After a review of the topic in §1, Part I explains the algorithm in geometric terms, emphasizing the main ideas. Part II presents an algebraic construction and a combinatorial proof that the algorithm works.

---

other players' payoffs from all their strategies, whereas perturbing its payoff gives its player a bonus for using that strategy. Thus, like stability, simple stability weakens hyperstability, which considers all payoff perturbations, by considering a restricted set of perturbations. If a game has only pure strategies (e.g., all mixed strategies are represented explicitly as pure strategies) then simple stability implies stability.

[3] Van Damme (1989) provides a critique of the 'forward induction' property of stable sets. Economic applications are studied by Bagwell and Ramey (1990), Banks and Sobel (1987), Cho and Kreps (1987), Cho and Sobel (1990), Glazer and Weiss (1990), Osborne (1990), and Ponssard (1991) — among many others. Stability is used also to refine Walrasian equilibria of economic models with features such as signaling and screening; cf. Gale (1992).

1. Introduction

A subset of a game's equilibria is stable only if each nearby game has equilibria near this subset. Kohlberg and Mertens' (1986) definition actually reserves 'stable' for a minimal closed subset with this property. Moreover, to reduce the size of stable sets, they restrict the neighborhood of nearby games to only those payoff perturbations induced by perturbing the lower bounds on players' mixed strategies. They show that stable sets exist and at least one lies within a single component. They also verify other desirable properties:

- Invariance: a stable set depends only on the minimal nonredundant description of the game, called the reduced normal form.
- Admissibility: no equilibrium in a minimal stable set uses a weakly-dominated strategy.
- Forward Induction: a stable set contains a stable set of the game obtained by deleting a strategy that is weakly dominated or strictly inferior at all equilibria in the set.

Unfortunately, their definition allows a stable set to span multiple components. Identifying a stable set within a single component is desirable in any case because all equilibria in the same component of a generic extensive game have the same equilibrium path and therefore the same outcomes; i.e., these equilibria differ only *off* the unique equilibrium path. For a generic game, therefore, a single-component stable set's equilibrium path and its outcome are 'stable' too. Uniqueness of this sort is essential for empirical studies — though more than one component can be stable.

Mertens (1989, 1991) adopts a revised definition that enforces connectedness and, partly as a result, gets the further property of Backward Induction: each stable set contains a proper (and therefore perfect) equilibrium. His definition is couched in terms of homology theory so we do not repeat it here; but for two-player games a necessary (and nearly sufficient) condition that a component is stable is again that each perturbed game has equilibria nearby.[4]

[4] For the two-player games addressed here, all computations are linear, which allows bypassing some aspects of Mertens' definition; also, for simplicity we apply the definition to stable components, not smaller connected sets. With these provisos, Mertens' definition says essentially that a component is stable if the projection from a neighborhood of the component to a neighborhood of the game is homologically nontrivial. Mertens (1986)

For expository purposes, Parts I and II develop the algorithm in two stages. The first stage (§2 and §5) reviews the setup and the Lemke-Howson (LH) algorithm for finding an equilibrium of a generic two-player normal-form game. It also describes the amendments that enable the algorithm to find perfect equilibria of nongeneric normal-form games: this is important because nontrivial extensive games, even generic ones, have nongeneric normal forms. Some examples in §3 motivate the subsequent presentation. The second stage (§4 and §6) shows how the LH algorithm can be applied repeatedly to find a component and a subset of its vertices that are simply stable.

The gist is the following. Like the LH algorithm, ours follows paths on which at most one pure strategy is used (i.e., with positive probability) without being optimal, and another's probability or payoff is perturbed. Geometrically, each path is a one-dimensional locus connecting vertices of best-reply regions along adjacent edges. It begins at a specified starting point and terminates at a vertex of some component. Each path is generated by the plausible rule of parametrically increasing the probability of a newly optimal strategy, or decreasing the profitability of a newly unused strategy, until an equilibrium is reached; cf. Wilson (1972). This rule is implemented by numerical operations on a tableau of detached coefficients as in linear programming.

To this we add a step applied when the LH algorithm terminates at a vertex that is not a strict equilibrium, and therefore not stable as a singleton set. A new perturbation is invoked to transit onto an alternative path, enabling continuation to another vertex. If a path starts to exit prematurely from a component then this step is applied in reverse to move to a vertex that terminates a path with a previous perturbation; these reversals enable the algorithm to follow folds in the graph of equilibria.

The perturbations that guide the paths reflect the motivation for stability as a refinement; viz., to be stable an equilibrium outcome must be affected slightly by small probabilities of deviant behaviors. Although the algorithm generally allows any generic perturbations as guides, for simple stability we use only $2n$ different guides, each lying

---

shows that this definition implies the necessity of the condition stated in the text; i.e., the projection is onto, so no region is left uncovered. This setup allows a large family of definitions depending on the homology theory used and the normalizations allowed, but he shows that these definitions are essentially equivalent. Mertens (1987) shows that applying a minimality requirement would violate the desirable property that the solution depends only on the ordinal properties of players' preferences.

in a different corner of a simplex of n probability and n payoff perturbations generating a neighborhood of the original game. In particular, each corner perturbation has an equilibrium near the component's vertex that is the terminus of the path guided by this perturbation — one says that this vertex 'covers' the perturbation. Coincidentally, using these perturbations assures that the algorithm generates only perfect equilibria.

The sequence of paths is called a route. From a specified starting point, a route follows a path to a vertex in some component. A route's subsequent paths within each component proceed around a 'great circle' of its vertices and their connecting edges.[5] The route exits onto another path to a new component if the sequence of covered perturbations doubles back on itself: this event reveals that the graph of equilibria has a fold that (presumably, but not surely) leaves some perturbations uncovered. In any case, each route surely terminates with vertices forming a simply-stable set. The signal for termination is that all corners are covered by the vertices encountered in the current component. Although this stopping rule assures finite termination, the vertices need not be a stable set because coverage of only the $2n$ corners (and incidentally $2n-1$ edges traversed enroute) is verified; e.g., interior perturbations are not checked explicitly. One could use additional guides (such as perturbations of mixed strategies) to verify stronger variants of simple stability, but we do not pursue that task here.

Part I: Construction of Stable Sets

2. Review of the Lemke-Howson Algorithm

As mentioned, our algorithm uses the LH algorithm repeatedly as a 'subroutine'. Its operation can be explained in geometric terms by considering the following example, which represents a generic normal-form game.[6]

Suppose a player has three pure strategies. Then the mixed strategies can be represented as points in a triangular simplex, as in Figure 1. The three extreme points identify

[5] The algorithm is designed purposely not to examine all vertices, since their number can be enormous. The situation is like the difference between the lengths of a circular orbit around the earth, and a flight over every city.

[6] The terms 'generic normal-form' in game-theoretic lingo and 'nondegenerate' in linear-programming lingo are essentially equivalent.

the pure strategies and a point in a face or the interior identifies the corresponding mixture of pure strategies of which it is a convex combination. Each vertex is marked by a circled letter that is the name of the pure strategy. The face opposite the vertex is labeled by the same letter to indicate that all the points on the face are mixed strategies *not* using that strategy; i.e., its probability is zero there.

The example shown in Figure 2 displays two simplices, one for player 1 with pure strategies a, b, and c, and one for player 2 with pure strategies d, e, and f. Further, the interior of each simplex is divided into regions: each region is labeled by the name of the pure strategy of the *other* player that is his best reply to any mixed strategy in the region. On the boundary between two or more best-reply regions, the best replies include all the best replies in adjacent regions: consequently, imagine that each edge or point bears the labels of its adjacent best-reply regions, as well as the labels of any pure strategies unused there.

For practical purposes this is a complete description of the game. To see this, recall that an equilibrium is a pair of strategies, one for each player, such that each pure strategy is used only if it is a best reply to the other's strategy. This is the same as saying that an equilibrium is a pair of points, one in each simplex, that is completely labeled: between them they bear all labels $a, \ldots, f$. For, if the point in a pure strategy's own simplex does not have its label then it is used, so it must be a best reply to the other's strategy, and therefore the point in the other simplex must bear its label.

As indicated in Figure 2 by the two endpoints of the dashed curve between the tops of the two simplices, the LH algorithm starts by assigning one player, say player 1, a strategy that is pure, say a, and the other, player 2, the strategy d that is a best reply to a. Thus, we start at a pair of pure strategies missing only label a — meaning that only a is used without being a best reply. The algorithm then follows the locus of points missing only label a, until it terminates at a pair with all labels, which is therefore an equilibrium. This path consists of the following edges in Figure 2. We move first along edge 1: b is an unused best reply at 2's pure strategy d so we can increase the probability of using b and still be missing only label a, but we must stop at the next vertex because going further would lose label d. But now f is a best reply so in 2's simplex we next traverse edge 2 until continuing would lose label b, and c becomes a best reply. Continuing thus,

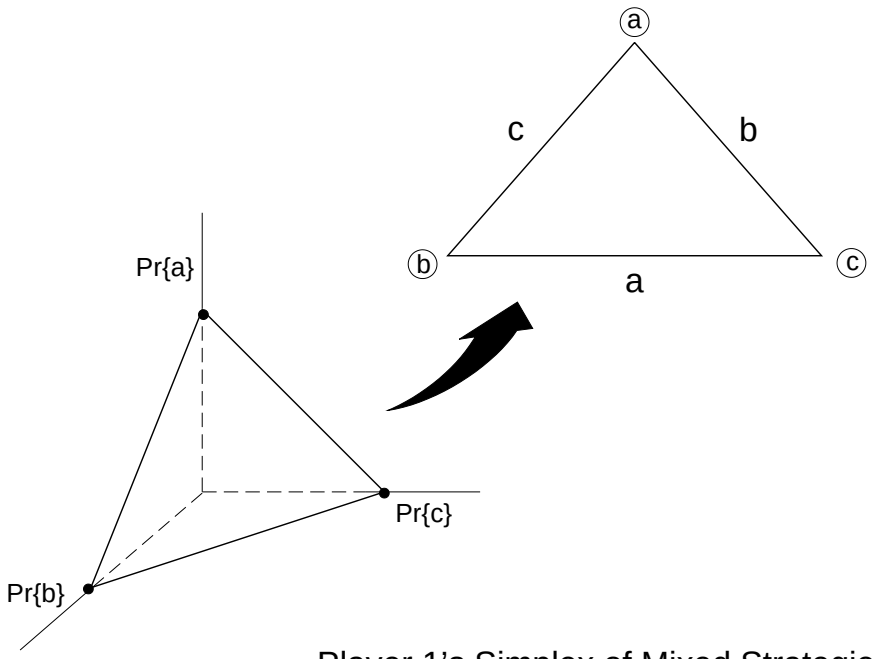
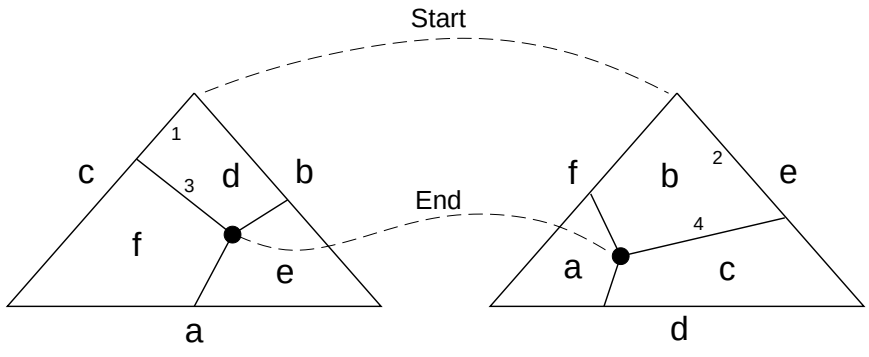


Figure 1: Geometric representation of a player's mixed strategies as a simplex with each face labeled by the strategy unused there.



we follow edge 3 and then edge 4, whereupon a becomes a best reply and a completely labeled pair is obtained, as indicated by the two endpoints of the lower dashed curve. A locus of this kind, and each point on it, is said to be almost-completely labeled; in this case, only label a is missing until the terminus is reached.

This procedure is entirely general. Start from a pair of pure strategies that is almost-completely labeled, missing at most one label (say a). If this pair is not completely labeled then following the incident locus of almost-completely labeled pairs finds a completely labeled pair that is an equilibrium. The algorithm cannot cycle because at most one pair of pure strategies is missing label a, and at the end of each edge one label becomes newly duplicated, identifying a unique edge in the other simplex along which the locus continues — or label a is obtained and an equilibrium is reached.

The path can be construed as a 'homotopy', as we now describe. We start with an artificial game that is easy to solve, namely player 1 receives a large bonus for using strategy a, in which case the pair (a, d) of pure strategies is indeed an equilibrium. Tracing the equilibria as the bonus is reduced to zero (i.e., player 1 is weaned from subsidy) amounts to following the locus of almost-completely labeled pairs until we arrive at an equilibrium of the true game. This interpretation reveals, however, that the graph of the locus may have folds, as shown for a hypothetical example in Figure 3 for the case that a is a best reply at the terminus (alternatively the terminus can occur where a's probability drops to zero while the bonus is still positive). The fact that the graph has only folds (if any) and no gaps, holes, or knots, is a general result established by Kohlberg and Mertens (1986, Theorem 1).[7]

The LH algorithm allows a significant generalization. If we start at any *other* equilibrium then again we can move along the locus of pairs missing only label a (viz., the initial segment moves away from whichever region bears label a at the equilibrium) to reach another equilibrium. Thus the number of equilibria is odd: the equilibria are connected

[7] Their theorem states that the projection from the graph of the equilibrium correspondence to the space of games is homotopic to a homeomorphism. This means that the graph can be stretched to eliminate folds without leaving holes. They use essentially the space of bonuses to model the space of games, so the graph in Figure 3 illustrates their result along a slice in which only the bonus for strategy a varies. They actually model these spaces as spheres by adding a point at infinity: in Part II we proceed similarly by including an 'extraneous vertex'.

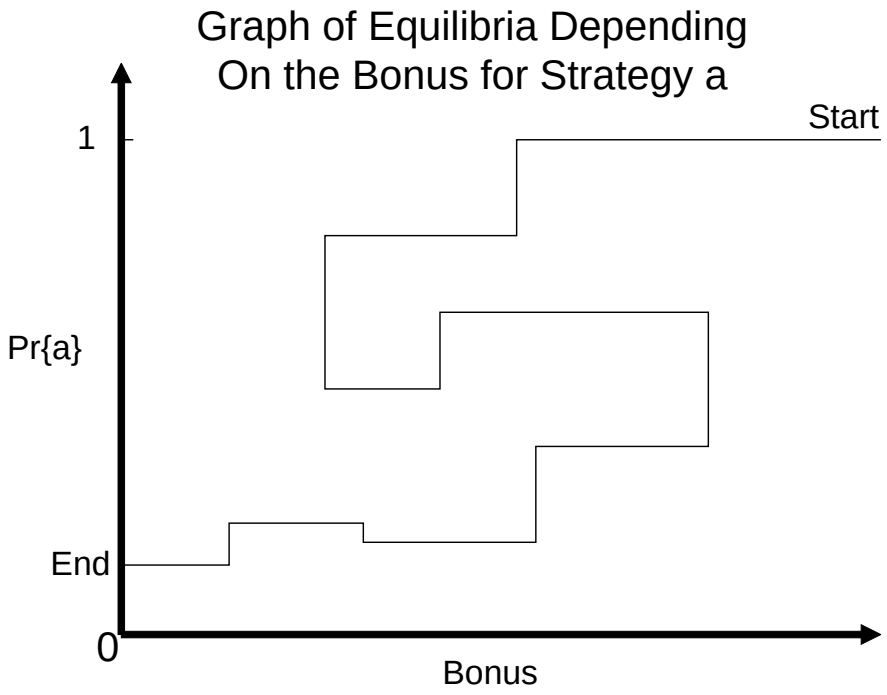


Figure 3: The path of the Lemke-Howson algorithm as a homotopy parameterized by the bonus for strategy a.

---

in pairs via paths missing only label a, except for the unique equilibrium connected by such a path to the unique pair of pure strategies missing at most label a. This can be seen in Figure 3: for each bonus specifying a generic game the number of equilibria is finite and odd.

Next we describe how the LH algorithm can be amended to handle nongeneric games. In geometric terms, a game is nongeneric if an end of an edge provides multiple choices for continuation; that is, one label is lost but two or more are acquired. This kind of degeneracy can happen along a face where a strategy of 1 is unused if one of 2's best replies is optimal only when 2 is certain that 1 will not use this strategy. A more complicated instance typical of extensive games is shown in the left panel of Figure 4, where 2 is indifferent between d and e if 1 uses only a. The middle panel shows that

this case can be handled by perturbing the probability of choosing b; i.e., this strategy is forced to be used with at least a small positive probability, which moves inward the face labeled b. This is called a primal perturbation. An alternative technique uses a dual perturbation that gives 2 a small bonus for choosing e, as shown in the right panel. In either case, the net effect is to assure that 2 has a unique best reply. In numerical computations it is unnecessary to invoke perturbations of either kind: Part II describes a unified lexicographic scheme that has the same effect.

Lastly we describe how, when the LH algorithm has reached a completely labeled pair using one perturbation, this perturbation can be replaced by another to begin a new path. We use primal perturbations to depict how this is done; dual perturbations are analogous. The left panel of Figure 5 shows a vertex of 1's mixed strategies that is part of an equilibrium $\odot$ in which strategies b and c are unused, d is a used best-reply of 2, and e is an *unused* best reply. This equilibrium is consistent with the primal perturbation in which b is perturbed more than c, as indicated by the dashed and dotted lines representing the larger and smaller perturbations. Altering this configuration by decreasing b's primal perturbation and increasing c's can be envisioned as moving the dashed line outward toward the boundary until the circled equilibrium hits the intersection between the d and e best-reply regions, and then moving inward the dotted line, carrying the equilibrium with it. The result is shown in the right panel, where now the dashed line identifies c's larger perturbation. This initiates a new path in which the missing label is b: it is used (slightly) but not optimal. Now e is an unused best reply, so the path continues by following the edge in the other player's simplex that makes e a used strategy.

In sum: Paths of the LH algorithm depend on the missing label and the perturbation or lexicographic scheme that resolves degeneracies. Each path's starting and ending points are completely labeled (and therefore equilibria), except one can be the unique pair of almost-completely labeled pure strategies. At an equilibrium where an optimal strategy is unused, one can transit onto a new path to another equilibrium by revising the perturbation.

After these preliminaries, in the next section we introduce the notion of stability via examples that illustrate the main ideas.

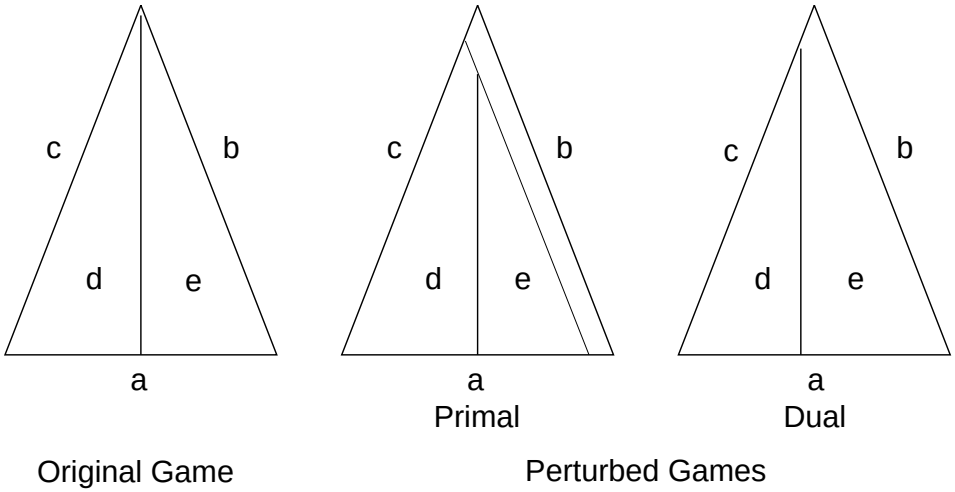
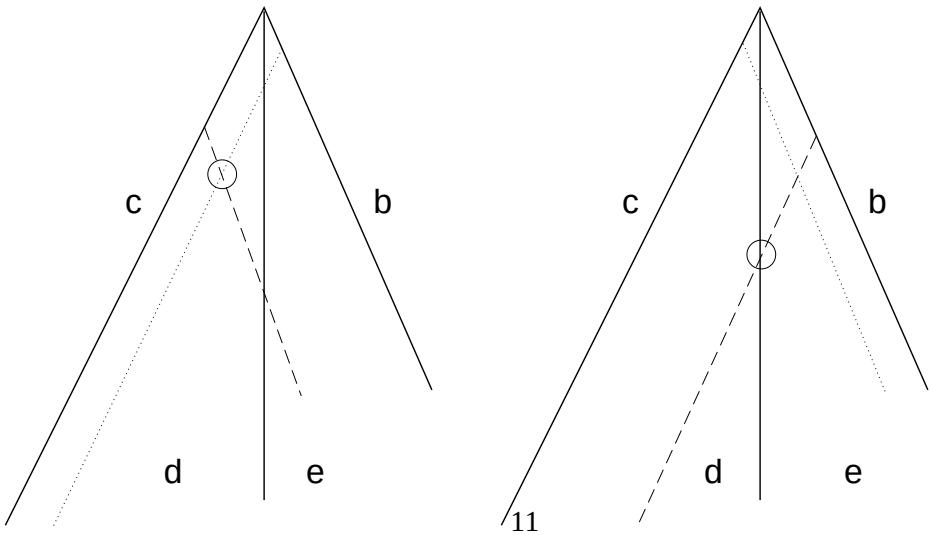


Figure 4: A primal perturbation of player 1's mixed strategies and a dual perturbation of 2's best replies.



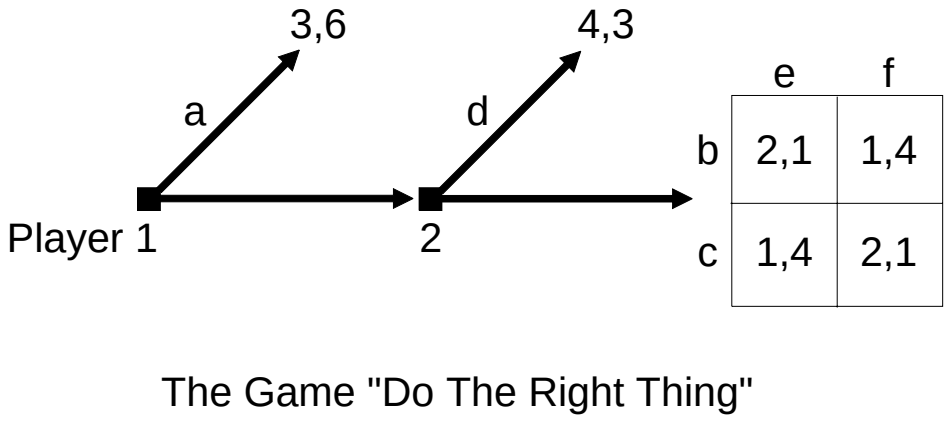


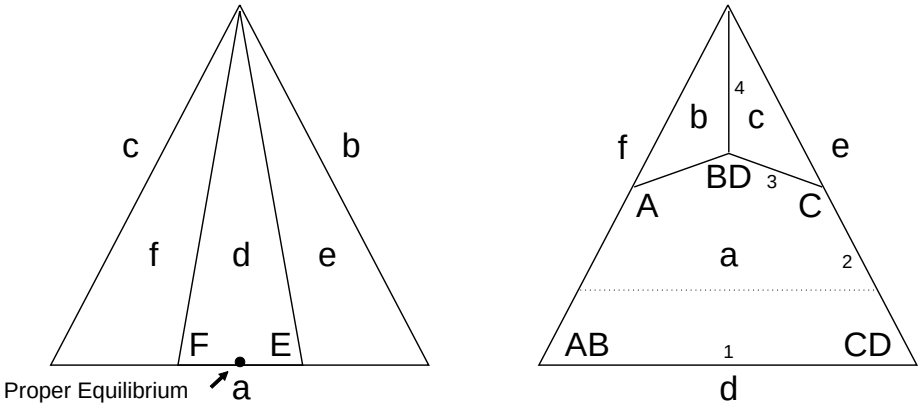
Figure 6: The extensive game 'Do The Right Thing'.

---

3. Examples of Stable Sets

The main example is the extensive game 'Do The Right Thing' shown in Figure 6. In this game, player 1 moves first, choosing either strategy a yielding payoffs $(3, 6)$ to players 1 and 2, or giving the next move to 2, who chooses either strategy d yielding payoffs $(4, 3)$, or chooses to play the simultaneous-move game with strategies b and c for 1 and e and f for 2, with the payoffs shown in the square box. The players' mixed strategies are shown in Figure 7 along with their best-reply regions. As in most extensive games there are degeneracies, at the two top vertices.

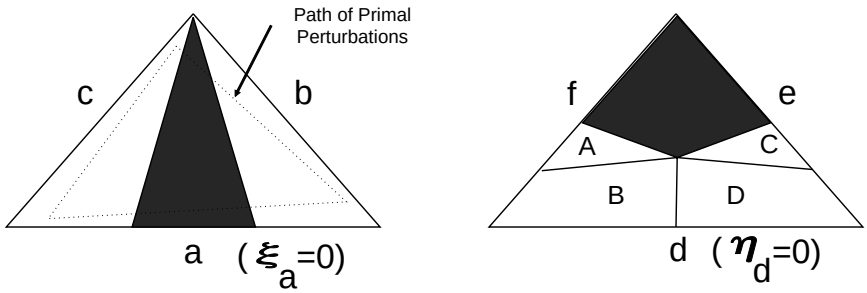
This game's equilibria lie in two components. In the first component, player 1 uses only a and 2 employs any mixed strategy in the best-reply region labeled a in the right panel of Figure 7. In the second component, 2 uses only d and 1 employs any mixed strategy in the best-reply region labeled d for which pure strategy a is unused (because



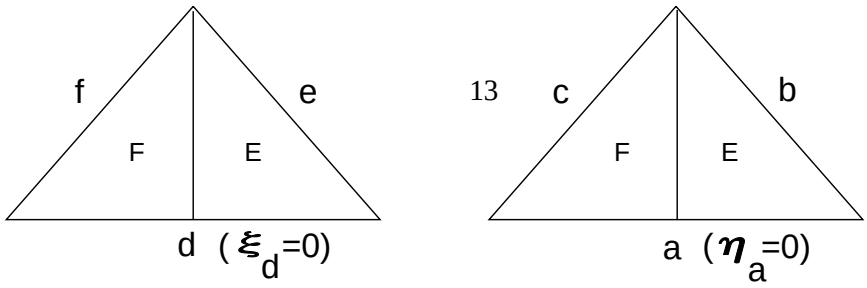
Mixed Strategies and Best Replies
For the Game "Do The Right Thing"

Figure 7: Mixed strategies and best replies for 'Do The Right Thing'. Vertices of equilibrium components are labeled by the regions of dual perturbations they cover in Figure 8. The numbered edges record part of a route of the algorithm.

First Component:



Second Component:



a is not a best reply when 2 uses only d): this is the interval between the two points E and F in the left panel. The unique proper equilibrium is d together with the midpoint of the interval between E and F.

This game illustrates the main ideas: the first component is not stable and the way it fails is indicative of the general case, whereas the second component is stable. We use the four panels in Figure 8 to describe the test for stability: the two upper panels refer to the first component, and the left and right panels pertain to primal and dual perturbations. Initially we refer only to the left panels, and then repeat the analysis in a dual form using the right panels.

Primal Stability

Although similar to the left panel of Figure 7, the upper-left panel of Figure 8 depicts the nonnegative perturbations (ξ_a, ξ_b, ξ_c) of the probabilities of 1's pure strategies a, b, and c. These are normalized so they sum to a small constant; thus, the simplex represents a cross-section of the cone of nonnegative perturbations analogous to Figure 1. The side labeled a is where ξ_a = 0, indicating that 1's pure strategy a has only the natural lower bound 0 on the probability that it is used. Recall that the first component has 1 using a and 2 adopting any mixed strategy for which a is a best reply. One vertex of this component has 2 using e (in Figure 7 this is the vertex labeled AB, as explained later): this pair obviously remains an equilibrium if the minimum probability ξ_e of using e is increased slightly, since e is already used with probability 1. This property is also true 'approximately' for perturbations ξ_d and ξ_f of 2's strategies d and f, in the sense that these perturbations move the vertex slightly and do not alter 1's best reply a. It is also true approximately for all perturbations of 1's strategies in the unshaded triangle where $\xi_b \geq 2\xi_c$ along the left side of the upper-left panel. Again, a small perturbation with $\xi_b > 0$ or $\xi_c > 0$ moves slightly the location of 1's strategy from the vertex (in the left panel of Figure 7) where a is used surely to a mixed strategy that respects the lower bounds ξ_b and ξ_c on the probabilities with which b and c must be used, and if $\xi_b \geq 2\xi_c$ then e remains 2's best reply. Similarly, the equilibrium where 2 uses f (the vertex labeled CD in Figure 7) 'covers' all perturbations with $\xi_c \geq 2\xi_b$. The equilibria labeled A and C in Figure 7 cover no additional perturbations: A covers again the unshaded triangle on the left; and C, the one on the right — and other equilibria in this component

cover lower-dimensional subsets of these.

The first component is unstable because of the existence of the shaded central triangle in Figure 8's upper-left panel. For any perturbation in this region, no equilibrium is close to any equilibrium in this component. This can be seen in Figure 7 from the fact that at the vertex where only a is used a perturbation with $\xi_b = \xi_c$ makes d the only best reply for 2, whereas a is not a best reply at the vertex where only d is used. The second component has no such deficiency: the equilibria where 2 uses d and 1 uses one of the mixed strategies E or F cover perturbations in corresponding regions labeled E and F in the lower-left panel, and these regions include all perturbations.

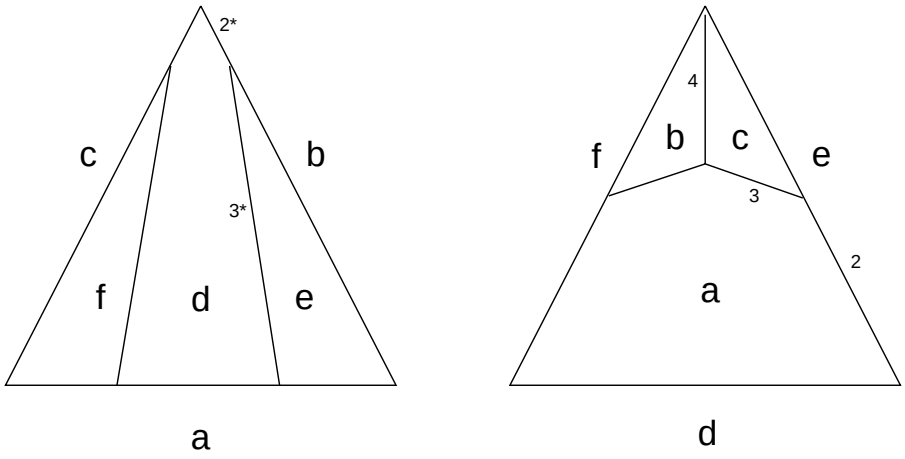
Dual Stability

Dual stability is motivated by the observation that the effect on an equilibrium of perturbing one player's strategies can be mimicked by appropriate bonuses for the other player: each strategy gets a bonus that offsets the risk represented by the other's perturbation. The upper-right panel depicts the simplex of 2's nonnegative bonuses η_d, η_e, η_f, normalized to sum to a small constant; similarly, the lower-right panel depicts 1's bonuses. Note that an equal bonus for each of 2's strategies ($\eta_d = \eta_e = \eta_f$) leaves that player's best replies unchanged: the barycenter of the dual-perturbation simplex represents a strategically equivalent game so it is in every covered region.

For the first component, the right panel of Figure 7 labels each vertex by labels from the regions shown in the upper-right panel of Figure 8. These indicate the bonuses covered by that equilibrium; e.g., for any bonus in region B there is an equilibrium near each of two vertices bearing label B. This component is not dual stable because the shaded region is not covered; i.e., dual perturbations in this region have no equilibria near the first component. One can verify this in Figure 9, which redraws the left panel of Figure 7 after the game is perturbed by a bonus for d that is larger than the bonuses for e and f: for this dual perturbation there is *no* completely-labeled pair of points for which a is a best reply for player 2. As shown in Figure 8's lower-right panel, the second component *is* stable because all bonuses are covered by equilibria E and F.

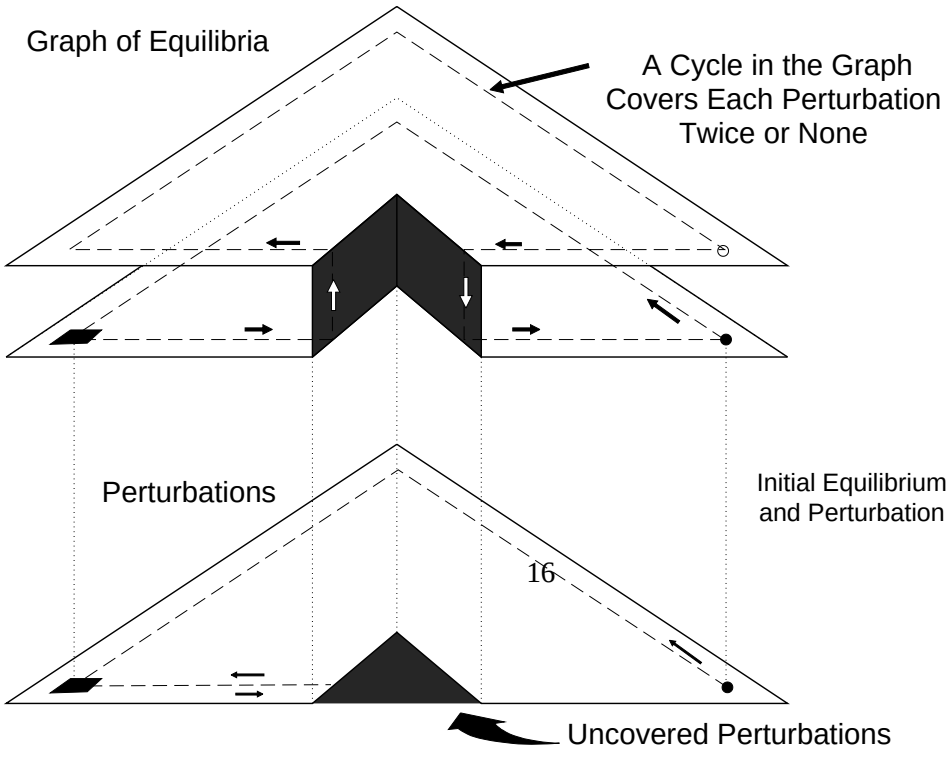
Folds in the Equilibrium Graph

It is useful to visualize the graph of the equilibrium correspondence in the two upper



A Dual Perturbation of Strategy d
That is Not Covered by the First Component

Figure 9: A dual perturbation of strategy d that is not covered by the first component. The numbered edges record a path from the first component to the second.



panels describing the first component. Above each simplex of perturbations (primal or dual) one can imagine the graph as the locus of equilibria corresponding to the perturbations. This locus may have multiple components and folds but recall that it cannot have holes. The dotted horizontal line in the right panel of Figure 7 is actually in the shadow of a fold; for, as we have seen, the perturbations in the covered regions A, B, C, and D of the upper-right panel of Figure 8 are each covered twice, once by equilibria below the dotted line and again by different equilibria above the dotted line. For instance, starting at vertex CD and circling around the boundary of the component traces out a cycle in the component, arriving back again at CD, and along the way we cover twice each covered perturbation without ever covering those in the shaded region. Similarly, in the upper-left panel there are two folds whose shadows are the left and right boundaries of the shaded region. For instance, the unshaded region on the left is covered by equilibrium AB and again by equilibrium A, and the edge connecting AB and A covers precisely the fold's shadow along the left edge of the shaded region.

Another example is depicted in Figure 10. A component of the graph is shown above the simplex of perturbations, which has an uncovered region. Each of two portions of the graph is shown schematically as a planar surface, without indicating the equilibria it comprises. The two planes cannot be connected by a tube because the graph cannot have holes; instead, they are connected by the shaded vertical segment above the boundary of the region of uncovered perturbations. Consequently, a full cycle — such as the one shown by dashed lines and arrows in the graph — passes through equilibria whose projections cover twice the perturbations below.

According to Kohlberg and Mertens (1986, Theorem 1) and Mertens (1989, 1991), this situation is typical. A component is unstable if over a neighborhood of perturbations its graph folds over so as to leave uncovered some residual region of perturbations. Each generic perturbation is covered an even number of times, including some covered zero times.[8] A stable component can cover generic perturbations an even number of times (Mertens (1986) constructs a three-player example) but at least one must produce odd

[8] The nongeneric perturbations are those in the intersections of covered regions, such as the line between regions A and B, which represent nearby nongeneric games. Typically, a nongeneric perturbation is associated with a 'vertical' segment of the graph where there is a continuum of equilibria.

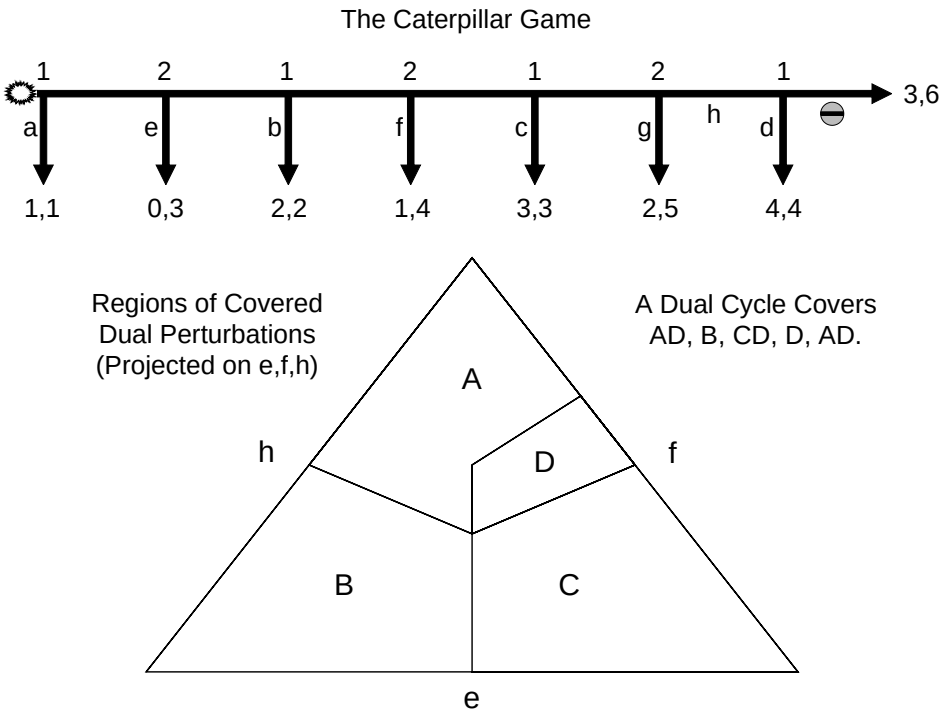


Figure 11: The Caterpillar Game and the regions of dual perturbations covered by equilibria of the unique component in which 1 uses only a. The graph folds twice over D, which is covered three times.

---

coverings.

It is important to realize, however, that folds occur frequently in components that *are* stable, so the mere existence of a fold does not imply a component is unstable. Figure 11 shows an example of the 'Caterpillar Game' and the covered regions of dual perturbations of 2's strategies e, f, and h for the single component where 1 uses only a. Equilibria in a cycle around the component cover the regions AD, B, CD, D, and again AD. Thus, there is a fold from CD to D that reverses to cover AD, and thereby D is covered three times.

4. Construction of Simply-Stable Sets

We now describe an adaptation of the LH algorithm to compute a simply-stable set

of vertices in a single component. Recall that such a set must be near an equilibrium of each game obtained by perturbing some strategy. Each strategy has two possible perturbations: the primal constrains the strategy's probability to be slightly positive and the dual awards a small bonus for choosing the strategy. If a set satisfies the weaker criterion that invokes only primal or only dual perturbations then it is called primal or dual simply stable. To describe the algorithm geometrically we use the dual version that finds a dual simply-stable set. Again, we illustrate with the game 'Do the Right Thing' in Figures 6-8.

Because stability is a relevant consideration only for games that are nongeneric in the normal form (even if they are generic in the extensive form), we implement the LH algorithm in the form adapted to handle degeneracy, as described in §2. To construct a dual simply-stable set, we guide the algorithm with perturbations from the simplex of bonuses depicted in the upper-right panel of Figure 8. Although the choice is fairly arbitrary, we use a generic perturbation close to some corner; as mentioned, this is made precise in Part II by using perturbations derived from a lexicographic order. If the guide is in the shaded region then the algorithm never enters the first component, since it has no equilibrium covering a perturbation in this corner. Suppose, however, that the initial guide is a perturbation in region B and that the LH algorithm arrives at the vertex labeled AB in Figure 7 — which it does if the missing label is e or f. Our task is to verify whether this component is dual simply stable, and if not to move on to another component.

Suppose that our intention is to find vertices covering the corners in counterclockwise order. We have covered the corner where only η_e is positive and we want next to cover the corner where only η_f is positive. To do this we alter the guide to one near that corner (i.e., in the region D) and resume the LH algorithm to arrive at the equilibrium labeled CD by traversing the path labeled 1 in Figure 7. This path can be viewed alternatively via its projection onto a path in the simplex of perturbations, connecting the first corner to the second. Having completed this step successfully, we next adopt a guide near the corner where only η_d is positive (i.e., in the shaded region). Again resuming the LH algorithm, the resulting path traverses the edge labeled 2 to the vertex C, then the edge labeled 3 to the vertex BD, and then (!) it starts to leave the component by exiting along

the edge labeled 4. This path can be seen fully in Figure 9, which shows the effect of the bonus η_d for 2's strategy d on the best-reply regions in 1's simplex of mixed strategies: the path missing label f follows the sequence $2 \to 2^* \to 3 \to 3^* \to 4$ of edges from the vertex CD in the first component to the vertex E in the second. In this example, in fact, the algorithm is allowed to exit from BD: it continues on to reach the second component's equilibrium E, and there a repetition of the procedure verifies that the second component is dual simply stable.

In general, however, one does not always let the LH algorithm exit a component, and we now describe why and the procedural rule that applies. In the above example, the LH algorithm is allowed to exit because we have arrived at an equilibrium BD that covers again the region B from which we started; indicating that region B and in particular the corner perturbation used as the initial guide are covered twice. The equilibrium correspondence folds over without covering the corner where only η_d is positive; thus, moving on to another component is a plausible tactic. Along the sequence of paths to reach such a conclusion, however, we must contend with the fact that there may be many folds, and some may be temporary ones that ultimately reverse. Thus, at intermediate steps we follow the sequence of folds until eventually either we exit at another vertex covering the initial corner (as in the example), or the folds reverse and we can proceed forward again. In general, except at a vertex covering the initial guide, whenever the LH algorithm would exit the component we revert back to the previous guide, which the current folded piece of the graph covers. From there we may need to revert further, eventually exiting if the initial guide recurs, or we may be able to continue forward again.

This reversion process is sketched in Figure 12, which depicts schematically a typical graph of the equilibrium correspondence near a single component along an edge of perturbations connecting one guide to the next. The path of the LH algorithm along this edge starts at the lower-left solid circle and proceeds in the direction of the arrows. It reverses direction at folds along the way, and eventually it starts to exit the component at the first open circle. At this point, one may be able to transit directly to another vertex that reverses the fold (this possibility is shown by the open circle at the next higher level in the figure). If not, then one continues backwards using the previous guide along the

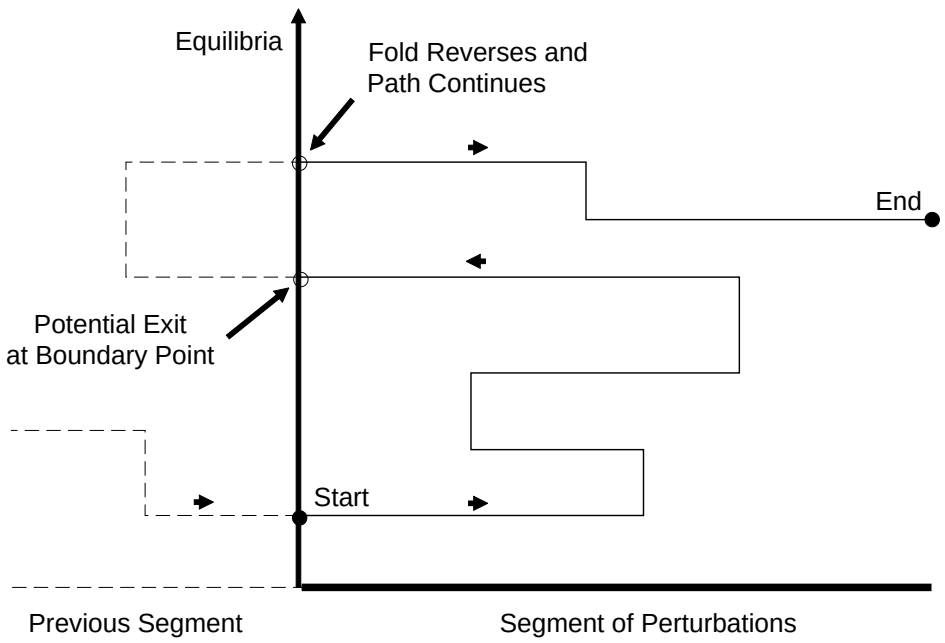


Figure 12: Where the graph folds over to cover again the first perturbation of a segment (so it would exit the component at that point), the algorithm backs up along the sequence of prior segments of perturbations until either the fold reverses — or the fold never reverses and then exit is permitted from the initial perturbation at which the component was entered.

---

corresponding edge. This reversion backwards continues until either one arrives back at the initial guide (indicating that exit is OK because the graph is folded over completely) or eventually one reaches an edge on which the fold reverses (as indicated schematically by the dashed segment).

Summary

To find a simply-stable set, the algorithm starts with an initial guide that is a generic perturbation near one corner of the simplex of primal and dual perturbations. From an almost completely-labeled pair of pure strategies it uses the LH algorithm to follow

a path to a vertex of some component. Within each component it invokes the other corners as guides around a circle of the corners and their connecting edges to generate a corresponding circle around vertices and edges of the component, though reversing each time a fold is encountered. Each edge from one corner to another generates a path of the LH algorithm in the graph above that edge. It may find all corners covered and therefore stop. Or via reversals enroute it may encounter a second vertex covering the initial guide, signaling that the graph is folded, in which case it exits onto the adjacent path that proceeds to another component, where the process repeats.

The entire route traces a unique one-dimensional locus that cannot branch or cycle: after a finite number of steps it terminates with a simply-stable set of vertices in a single component. This conclusion is derived in two parts. The first is based on (1) the previous observation that each path between or within components cannot branch or cycle, and (2) the further fact that at each vertex the rules allow precisely two incident paths: the route enters on one and exits on the other.

Part II: Algebraic Specification of the Algorithm

The geometric sketch of the algorithm leaves aspects uncovered, so we present an algebraic construction that is more precise. §5 specifies the formulation and reviews the LH algorithm as amended for nongeneric games, which require lexicographic procedures to cope with degeneracy. Details of the numerical procedures are relegated to the Appendix. §6 describes the algorithm for calculating a simply-stable set and proves that it works.

5. Formulation

The game in normal form is described by two m_1 by m_2 matrices U^1 and U^2 with elements u^p_{ij}. Player p has m_p pure strategies, and if 1 and 2 choose strategies i and j then p's utility payoff is u^p_{ij}. A mixed strategy for player p is an assignment $\bar{x}^p$ of probabilities to his pure strategies, represented as a nonnegative m_p-dimensional vector of elements summing to one.

An equilibrium is a pair $\bar{x} = (\bar{x}^1, \bar{x}^2)$ of mixed strategies for the players such that a player assigns positive probability to a pure strategy only if it is an optimal response

to the other's mixed strategy. More precisely, $\langle \bar{x}, \bar{y}, v \rangle$ solves a linear complementarity problem:[9]

$$U^1 \cdot \bar{x}^2 + \bar{y}^1 = v_1 \mathbf{1}$$

$$U^{2\prime} \cdot \bar{x}^1 + \bar{y}^2 = v_2 \mathbf{1}$$

subject to $\bar{x}^p \geq \mathbf{0}$, $\mathbf{1}' \cdot \bar{x}^p = 1$, $\bar{y}^p \geq \mathbf{0}$ and $\bar{x}^p \perp \bar{y}^p$ for $p = 1, 2$. Here, v_p is player p's equilibrium expected payoff; therefore, the i-th slack variable $\bar{y}_i^p$ must be zero if p's pure strategy i is used with positive probability $\bar{x}_i^p > 0$. This 'complementarity' condition for optimality requires that $\bar{x}^p$ and $\bar{y}^p$ are orthogonal.

This problem is modified for computational purposes as follows. One can assume without loss of generality that every payoff is strictly positive: if not, add a sufficiently large constant to each matrix. Thus, each player p's expected payoff v_p is positive. This enables consideration of a linear complementarity problem in standard form: one seeks a solution $\langle x, y \rangle$ to the linear system[10]

$$U \cdot x + I \cdot y = \mathbf{1}, \qquad x \geq \mathbf{0}, \qquad y \geq \mathbf{0},$$

satisfying the complementarity condition $x \perp y$. Here, the two players' strategies are numbered consecutively as $1, \ldots, n$, where $n = m_1 + m_2$, I is the $n \times n$ identity matrix, and

$$U = \begin{bmatrix} \mathbf{0} & U^1 \\ U^{2\prime} & \mathbf{0} \end{bmatrix}, \quad x = \begin{bmatrix} v_2^{-1}\bar{x}^1 \\ v_1^{-1}\bar{x}_2 \end{bmatrix}, \quad y = \begin{bmatrix} v_1^{-1}\bar{y}^1 \\ v_2^{-1}\bar{y}^2 \end{bmatrix}.$$

Except for the extraneous solution $\langle x, y \rangle = \langle \mathbf{0}, \mathbf{1} \rangle$, each solution of this problem yields an equilibrium by normalizing each player's elements of x to sum to 1. We sometimes refer to a solution as an equilibrium (possibly extraneous).

The set of feasible solutions to the linear system is a compact convex polyhedron whose vertices are basic solutions. Each *basis* consists of n columns from the $n \times 2n$ matrix $A \equiv [U, I]$ such that the basis matrix B comprising these columns is nonsingular

[9] We use **1** to indicate a vector of ones, **0** to indicate a vector or matrix of zeros, and a prime to indicate 'transpose'.

[10] This form is used here for expositional purposes. For computations it is more efficient to treat the players' problems separately so that one works with two $m_1 \times m_2$ matrices. In fact, it is sufficient to carry out basis changes via pivoting on the two $m \times m$ submatrices representing the m rows and columns of the players' used strategies. In large games, m is typically small compared to m_1 and m_2.

and the solution is feasible. At the basic solution the values of the nonbasic variables are zero and the values of the basic variables are the corresponding elements of $b \equiv B^{-1} \cdot \mathbf{1}$, which must be nonnegative. One moves among vertices by numerical operations of the sort used in linear programming. Geometrically, each operation moves the solution along the edge connecting two adjacent vertices. Computationally, each operation adjoins one column to the basis matrix and deletes another selected to obtain the unique new basis that preserves feasibility. The increase (from zero) in the previously nonbasic variable that is made basic, or the decrease (to zero) in the previously basic variable that is made nonbasic, parameterizes the edge traversed in moving from one vertex to an adjacent one. As in linear programming, this operation can be executed efficiently on a tableau of detached coefficients. The Appendix describes the rules for selecting the column that leaves the basis, and for calculating the new tableau from the old.

If the basis includes one member of each complementary pair $\{x_k, y_k\}$, $k = 1, \ldots, n$, then it is complementary and therefore yields a solution. The basis is called i-almost complementary if it is complementary *or* if it includes both members of only one pair $\{x_i, y_i\}$ and therefore excludes both members of another pair, say $\{x_j, y_j\}$. Analogously, we say that the corresponding vertex is complementary or i-almost complementary (i-ac).

The Lemke-Howson Algorithm

Lemke and Howson (1964) show that if the problem is nondegenerate then, fixing some strategy i, each complementary vertex is connected to another by a locus called an i-path that consists of a sequence of adjacent i-ac vertices and their connecting edges.[11] The locus of an i-path is defined by relaxing the complementarity condition to allow $x_i y_i \geq 0$; thus, i-ac vertices occur as intermediate points. Complementary vertices are connected in pairs by i-paths, and no two pairs overlap, so their number is even. Excluding the

[11] See also Lemke (1965) and Tomlin (1978). Nondegeneracy requires that basic variables always have positive values ($b \gg \mathbf{0}$), which is satisfied by generic normal-form games but not by nontrivial generic extensive games. An extension to degenerate two-player games is by Eaves (1971). For other cases the extensions assume nondegeneracy: two-player games of incomplete information by Howson and Rosenthal (1974); N-player polymatrix games by Eaves (1973) and Howson (1972); general N-player games, by Rosenmüller (1971) and Wilson (1971); and Walrasian equilibria of piecewise-linear economies, by Wilson (1978). The latter three require nonlinear (actually, multilinear) calculations; for games with three or more players it is therefore often more efficient to use approximation methods based on simplicial subdivisions.

extraneous vertex, these yield all the equilibria (an odd number) due to nondegeneracy. To find an equilibrium, the algorithm can start from the extraneous vertex and follow some i-path to find the complementary vertex that is its other terminus.

The procedural steps of the LH algorithm are as follows. Given any complementary vertex as a starting point, the unique incident i-path is traversed by initially increasing (i.e., making basic) the one of x_i or y_i that is initially nonbasic. One arrives thereby at an initial i-ac vertex with some complementary pair $\{x_j, y_j\}$, $j \neq i$, having both members nonbasic, one of which most recently became nonbasic and the other was previously nonbasic: this is the current 'nonbasic duplicated pair'. (For instance, starting from the extraneous vertex, the initial i-ac vertex is the pair of i-ac pure strategies, as in Figure 2 for the case that $i = a$ and the nonbasic duplicated pair has $j = b$.) Thereafter the steps are mandatory: each step moves from the current i-ac vertex to an adjacent one by making basic the 'other' member of the nonbasic duplicated pair (i.e., the one that does not reverse the path), until complementarity is restored when either x_i or y_i becomes nonbasic — indicating that another complementary vertex has been reached. Nondegeneracy assures at each step that a unique basic variable becomes nonbasic, so the path always has a unique continuation until complementarity is restored. Eaves (1971) extends these results to degenerate problems by using a lexicographic order to resolve ties in the selection of the variable leaving the basis when another is introduced.

The Lexicographic Order

Our algorithm also handles degeneracy via a lexicographic order (lex-order) but in a unified setup that allows consideration of the perturbations relevant to stability. Interpret the array of coefficients in the initial tableau $[-\mathbf{1}, A]$, where $A \equiv [U, I]$ and the basis matrix is I, as representing the matrix equation

$$[-\mathbf{1}, A] + A \cdot Z = \mathbf{0}, \qquad \text{where} \qquad Z \equiv \begin{bmatrix} X \\ Y \end{bmatrix}.$$

This is an equation for the two $n \times (1 + 2n)$ matrices X and Y, each row of which provides the data for a lexicographic representation of the corresponding element of x or y. At a basis B, the induced tableau $[-b, \overline{A}] \equiv B^{-1} \cdot [-\mathbf{1}, A]$ similarly represents the corresponding matrix equation with these data. The vertex that is the basic solution

is obtained by setting to zero the nonbasic rows of Z (those corresponding to nonbasic columns), and then the basic rows are the negatives of their rows in the tableau.

Each particular lex-order is associated with a family of perturbations. For such a perturbation represented as a $(1+2n)$-dimensional vector $\delta = [\delta_0, \delta_1, \ldots, \delta_{2n}]$, the matrix equation collapses to a perturbation of the original vector equation:[12]

$$[-\mathbf{1}, A] \cdot \delta + A \cdot z = \mathbf{0},$$

where $z = Z \cdot \delta$, or after premultiplying through by B^{-1} to obtain the tableau,

$$[-b, \overline{A}] \cdot \delta + \overline{A} \cdot z = \mathbf{0}, \qquad \text{where} \qquad z \equiv \begin{bmatrix} x \\ y \end{bmatrix}.$$

This form specifies directly the dependence on the perturbation δ, whereas the matrix equation leaves open the specification of the perturbation; in either case, the tableau is a full summary.

To impose a specific lex-order, we select a permutation ℓ of the indices $1, \ldots, 2n$ of primal and dual variables with the interpretation that the implied perturbation has

$$\begin{aligned} \delta_0 &= \epsilon^0 \equiv 1, \\ \delta_{\ell(j)} &= \epsilon^j \qquad \text{if } 1 \le j \le 2n, \end{aligned}$$

where ϵ is positive and small. Thus, the permutation ℓ indicates for $\ell(j) \le n$ that strategy $\ell(j)$ is forced into play with a probability of order ϵ^j; and for $\ell(j) > n$ that strategy $\ell(j) - n$ is awarded a bonus of order ϵ^j.

Each choice of the permutation ℓ defines an associated lex-order as follows. Let M be the permutation matrix that permutes the columns of Z so that the $(1 + \ell(j))$-th column goes in the $(1 + j)$-th position. Then Z is lex-positive if in ZM each row's first nonzero element is positive. This definition assures that if Z is lex-positive then $z \equiv Z\delta$ is positive for all sufficiently small positive values of ϵ.

This construction is applied directly to a tableau $[-b, \overline{A}]$ to determine if its associated basic solution is lex-positive by using the criterion that in each row i the first nonzero

[12] The origin of this representation is the observation that the perturbed problem has the analogous form $U \cdot [x + \xi] + I \cdot [y + \eta] = \mathbf{1}$, where $\delta = [\xi, \eta]$. Although each primal perturbation ξ induces a dual perturbation $\eta = U \cdot \xi$, the converse is false and usually there are many other dual perturbations.

element in the sequence $\bar{a}_{i0}, \bar{a}_{i\ell(1)}, \ldots, \bar{a}_{i\ell(2n)}$ must be *negative*, where $\bar{a}_{i0} \equiv -b_i$. When moving from one vertex to another one brings a designated nonbasic column into the basis and makes nonbasic another that is selected to assure that the new vertex is lex-positive. As described in the Appendix, this selection is unique.

For the portion of the algorithm that uses the LH algorithm to follow an i-path we use the lex-order based on the cyclic permutation ℓ^i for which

$$\ell^i(j) = \begin{cases} i+j & \text{if } i+j \leq 2n\,, \\ i+j-2n & \text{if } i+j > 2n\,. \end{cases}$$

That is, $\ell^i = [i+1, \ldots, 2n, 1, \ldots, i]$ is the i-th forward rotation of the identity permutation $\ell^0 \equiv [1, \ldots, 2n]$, and $\ell^{2n} \equiv \ell^0$. Thus, vertices on an i-path are i-almost complementary, and feasible (lex-positive) with respect to perturbations that perturb z_i the least and z_{i+1} the most. Note that z_i is x_i if $i \leq n$ and y_{i-n} otherwise, and similarly for z_{i+1}.

A complementary vertex that is lex-positive using the i-th cyclic permutation is called an i-vertex and denoted (B, ℓ^i) if its basis matrix is B. When discussing an i-vertex we simplify by referring to its associated variable as z_i (rather than, say, x_i); but we refer directly to a complementary pair $\{x_j, y_j\}$ rather than complicated references to, say, z_j and its complement.

The previously described properties of the LH algorithm are preserved: the effect of a lex-order is implicitly to perturb the problem to make it nondegenerate. In particular, each endpoint of an i-path is an i-vertex covering the corner where only z_{i+1} is perturbed. Observe that a set including an i-vertex for each $i = 1, \ldots, 2n$ is simply stable, because each corner is covered. Our algorithm is based on the tactic of switching from an i-vertex onto an $(i+1)$-path to move to an $(i+1)$-vertex. If this is completed successfully for $i = 1, \ldots, 2n$ then the algorithm has computed a simply-stable set.

6. The Algorithm

We now specify the algorithm and derive its properties.

To recapitulate, each vertex is characterized by a pair (B, ℓ) in which B is a basis matrix yielding a basic solution that is ℓ-feasible according to the lex-order derived from the cyclic permutation ℓ of the corners. ℓ-feasibility requires that if each $\delta_{\ell(j)} = \epsilon^j$ and

ϵ is positive and sufficiently small, then the basic variables are positive. Feasibility with respect to the i-th cyclic permutation ℓ^i is called i-feasibility. A complementary vertex is one for which the basis is complementary: the basis includes one column from each complementary pair. Each complementary vertex is an equilibrium that is a vertex of some component. An i-vertex is an i-feasible complementary vertex, and it is an equilibrium that covers the perturbation constructed from ℓ^i. An i-ac vertex is a basic solution that is i-feasible and for which the basis has two columns from the i-th pair (the basic duplicated pair) and one column from all but one other pair (the nonbasic duplicated pair).[13]

As in the LH algorithm, an i-path is a connected locus of i-feasible complementary and i-almost complementary vertices consisting of its two complementary endpoints (i-vertices), intermediate i-almost complementary points (i-ac vertices), and the edges connecting them. These edges are traversed by changing the basis according to a rule, called the RP rule in the Appendix, that preserves i-feasibility.

The Appendix also describes an alternative NP rule used when transiting from a 'maximal' i-vertex to a 'boundary' $(i+1)$-ac vertex, or the reverse. An i-vertex is maximal if its basis is not $(i+1)$-feasible, and minimal if its basis is not $(i-1)$-feasible. An i-ac vertex is said to be an equilibrium i-ac vertex if the basic value of the variable z_i is zero ($b_i = 0$) — where z_i is x_i or y_i depending on which one's index is $\ell^i(2n)$. A boundary i-ac vertex is an equilibrium i-ac vertex for which the basic value of z_i would increase if the column of one of the nonbasic duplicated variables were made basic. This nonbasic duplicated variable is said below to be the one that 'interacts' with z_i.[14] The NP rule in the Appendix selects for deletion the unique basic column for which the new basic

[13] Unlike an i-vertex, an i-ac vertex's covered region need not include any corner: examples are the equilibria in Figure 7 labeled A and C, which cover the corresponding regions in the upper-right panel of Figure 8. To identify these covered regions it is easiest first to make the basis complementary by replacing one of the basic duplicated pair with one of the nonbasic duplicated pair. For a complementary basis, the inequalities defining the covered region are given by the tableau's rows for basic variables with zero values.

[14] If the nonbasic duplicated pair is $\{x_j, y_j\}$ then the member that interacts with x_i is x_j if i and j are strategies of the same player and y_j otherwise; and similarly the member that interacts with y_i is y_j in this event and x_j otherwise. A boundary i-ac vertex has the property that the column of the member of the nonbasic duplicated pair that interacts with z_i has a nonpositive element in each row with basic value zero, and a negative element in the row of z_i. The Appendix provides details.

solution will be lex-positive according to the *new* lex-order.

Adjacency Relations

For the algorithm we define a symmetric binary relation of adjacency among vertices. As a preliminary, define a symmetric relation of 'potential adjacency' as follows:

- An i-vertex (B, ℓ^i) is potentially adjacent to those among the $(i-1)$-vertex (B, ℓ^{i-1}), the $(i+1)$-vertex (B, ℓ^{i+1}), the i-ac vertex (B', ℓ^i), and the $(i+1)$-ac vertex (B'', ℓ^{i+1}) that are feasible, where:
  - B' is obtained from B by making basic the nonbasic member of the i-th complementary pair, using the RP rule.
  - If the i-vertex is maximal, then B'' is obtained from B by making z_{i+1} basic, using the NP rule.
- An i-ac vertex (B, ℓ^i) is potentially adjacent to those among the i-ac vertex (or i-vertex) (B^*, ℓ^i) and (B^{**}, ℓ^i), and the $(i-1)$-vertex $(B^\dagger, \ell^{i-1})$ that are feasible, where:
  - B^* and B^{**} are each obtained from B by making basic one of the nonbasic duplicated pair, using the RP rule.
  - If the i-ac vertex is at a boundary, then $B^\dagger$ is obtained from B by making basic the member of the nonbasic duplicated pair that interacts with z_i, using the NP rule.

Two properties of the potential adjacency relation are established in the following lemmas proved in the Appendix.

LEMMA 1: *An i-vertex is potentially adjacent to an equilibrium i-ac vertex if and only if it is minimal.*

LEMMA 2: *An i-vertex is potentially adjacent to a boundary $(i+1)$-ac vertex if and only if it is maximal.*

The implications of these lemmas are depicted schematically in the left panel of Figure 13. In the figure, an open square indicates an i-vertex, which is shaded if it is maximal; a circle indicates an equilibrium i-ac vertex, which is filled if it is at a boundary; a solid line indicates an edge generated by a basis change, in bold if the NP rule applies; and a dashed line indicates rotation of the lex-order. As shown in the top portion, the i-vertex in the center has four potential adjacencies. However, Lemma 2 indicates that only one of the bottom [an $(i+1)$-vertex] and right [a boundary $(i+1)$-ac vertex] adjacencies is

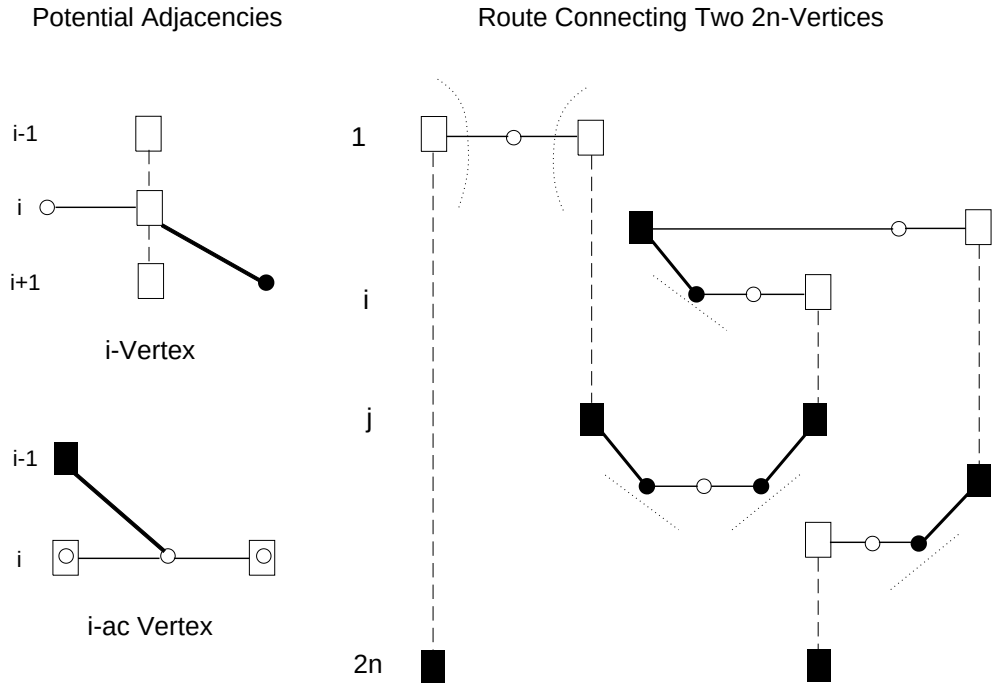


Figure 13: In the left panel, each i-vertex is potentially adjacent to four other vertices: precisely one of the left and top adjacencies is feasible, and one of the bottom and right. Similarly, an i-ac vertex has three potential adjacencies and only two are feasible. The implications of Lemmas 1 and 2 are illustrated in the right panel which displays a route connecting two $2n$-vertices in different components.

---

feasible, and Lemma 1 indicates that only one of the left [an equilibrium i-ac vertex] and top [an $(i-1)$-vertex] adjacencies is feasible. Similarly, in the bottom portion the i-ac vertex in the center has three potential adjacencies, but of the two connected to the left and right with another i-ac vertex or i-vertex, one is excluded if the one in the center is a boundary i-ac vertex and in this case it is adjacent to a maximal $(i-1)$-vertex.

To specify the algorithm we select a particular strategy, hereafter strategy 1, and then relative to this selection we define a relation of 1-adjacency, or just 'adjacency' for

brevity, that is the same as potential adjacency except for the restrictions that:[15]

- Exclude all adjacency relations involving an i-ac vertex for which $i > 1$ and the basic value of z_i is positive ($b_i > 0$).
- Exclude all adjacency relations between a $2n$-vertex and a 1-vertex or 1-ac vertex.

In keeping with these restrictions, we hereafter specify that a 1-vertex is minimal and an $2n$-vertex is maximal. This is consistent with Lemmas 1 and 2 in the following sense. For $i > 1$ the first restriction confines adjacency to an equilibrium i-ac vertex, which Lemma 1 says can be adjacent to a minimal i-vertex. In parallel, we allow that any 1-ac vertex can be adjacent to a 1-vertex, which is minimal by definition. Similarly, minimality of a 1-vertex and maximality of a $2n$-vertex is consistent with the exclusion in the second restriction of adjacencies between a 1-vertex and a $2n$-vertex. The second restriction also excludes adjacency between a $2n$-vertex and a (boundary) 1-ac vertex. As will be seen, this case never arises in the algorithm because we enforce the rule that a $2n$-vertex is a terminus of the algorithm; and in the reverse direction, allowing 1-paths to cross boundaries is the key property that enables the algorithm to move from one component to another in its search for one that is simply stable. In contrast, the first restriction assures that if $i \neq 1$ then a boundary i-ac vertex is *not* adjacent to the one of the two potentially adjacent i-ac vertices that makes basic the interacting member of the nonbasic duplicated pair, since that would make b_i positive; hence, this adjacency is replaced by adjacency to a maximal $(i-1)$-vertex within the same component. Thus, all adjacencies are within the same component except along 1-paths: it is only along a 1-path that a chain of adjacencies can enter or leave a component.

The key properties of the adjacency relation are summarized by

THEOREM 1: *A $2n$-vertex is adjacent to a unique $(2n-1)$-vertex if it is not minimal or to a unique equilibrium $2n$-ac vertex if it is minimal. If $i \neq 2n$ then an i-vertex or i-ac vertex is adjacent to zero or two other vertices.*

PROOF: The second restriction on the adjacency relation assures that for a $2n$-vertex the two listed adjacencies are the only possible ones. The first restriction requires that at an

[15] Adjacency obviously depends on both the selected corner (here, only $i = 1$ allows entry and exit from a component) and the numbering of the strategies via the specification of the cyclic permutations ℓ^i. By varying these specifications, the algorithm can find different simply-stable sets.

adjacent $2n$-ac vertex z_{2n} has the basic value b_{2n} = 0 and is therefore an equilibrium; then, Lemma 1 assures that this adjacency occurs if and only if the $2n$-vertex is minimal and therefore is not adjacent to the $(2n-1)$-vertex with the same basis. For the case i = 1, a 1-vertex is adjacent to the 1-ac vertex obtained by making basic the nonbasic member of the 1-th complementary pair. The second restriction assures that adjacency to the $2n$-vertex with the same basis is precluded, and of the other two possibilities precisely one is true according to Lemma 2: if the 1-vertex is not maximal then it is adjacent to the 2-vertex with the same basis, and otherwise it is adjacent to the boundary 2-ac vertex obtained by making z_2 basic. If $i \neq 1,\ 2n$ then Lemma 1 asserts that between the two potential adjacencies with an equilibrium i-ac vertex and the $(i-1)$-vertex with the same basis, precisely one is true (i.e., the former if the i-vertex is minimal). Similarly, Lemma 2 asserts that between the two potential adjacencies with a boundary $(i+1)$-ac vertex and the $(i+1)$-vertex with the same basis precisely one is true (i.e., the former if the i-vertex is maximal). Turning to the case of an i-ac vertex, there are no adjacencies unless i = 1 or the basic value of z_i is zero and it is an equilibrium. If i = 1, or b_i = 0 but it is not at a boundary, then it is adjacent to precisely the two i-ac vertices (or one could be an i-vertex) obtained by making basic one of the nonbasic duplicated pair. If $i \neq 1$ and it is at a boundary then one of these is replaced by adjacency to the maximal $(i-1)$-vertex obtained by making basic the nonbasic duplicated variable that interacts with z_i, using the NP rule. *Q.E.D.*

The Algorithm

In view of Theorem 1, the algorithm for simply-stable sets consists of starting at one $2n$-vertex and thereafter following the unique chain of adjacent vertices to arrive at another $2n$-vertex. The locus comprising a chain's vertices and connecting edges is called a route.

We show that a route's terminal $2n$-vertex lies in a component that is simply stable if it was entered on a 1-path, and indeed in this case the route's members of this component provide a simply-stable set (not necessarily minimal). In particular, starting from the extraneous vertex's $2n$-vertex (I, ℓ^{2n}), which is uniquely maximal in this component, the route necessarily exits the component from the 1-vertex (I, ℓ^1), which is uniquely minimal in this component, on a 1-path and terminates in a different component that is simply stable. The members of the route in the last component yield a simply-stable set

of equilibria because it enters on a 1-path and terminates at an $2n$-vertex, and includes an i-vertex for each $i = 1, \ldots, 2n$.

The statement of the properties of the algorithm is

THEOREM 2: *Each $2n$-vertex is connected via a unique route of adjacent vertices and their connecting edges to another $2n$-vertex. If the route leaves one component then it enters another on a 1-path. The last component entered is simply stable and the route's vertices in this component form a simply-stable set.*

PROOF: If $i \neq 1$ then a maximal subchain of adjacent equilibrium i-ac vertices lies within a single component and has two endpoints each of which is either a minimal i-vertex or a boundary i-ac vertex. An endpoint that is a minimal i-vertex is, if $i \neq 2n$, adjacent within the same component to either (if it is not maximal) an $(i+1)$-vertex with the same basis or (if it is maximal) a boundary $(i + 1)$-ac vertex in this component; and if $i = 2n$ then it has no other adjacent vertex and is a terminus of the route. An endpoint that is a boundary i-ac vertex is adjacent to a maximal $(i - 1)$-vertex. Letting i vary, a maximal subchain of adjacent i-vertices has a minimal and a maximal vertex as its two endpoints (or one if they are the same), say an i^*-vertex and an i°-vertex, the former adjacent to an i^*-ac vertex (in the same component if $i \neq 1$) and the latter (if $i^\circ \neq 2n$) to a boundary $(i^\circ + 1)$-ac vertex in the same component. Thus, starting from one $2n$-vertex there is a unique adjacent vertex and thereafter each vertex encountered has two adjacent vertices, one of which has just been visited and the other of which continues the route until it is terminated when another $2n$-vertex is reached. Such a terminus must be reached because the route cannot cycle (a cycle would require some vertex to be adjacent to three others) and because the number of combinations of bases and cyclic permutations is finite. By construction, components are left and entered only on 1-paths. Thus, if the last component is entered on a 1-path then from there to the terminus the route includes an i-vertex for each $i = 1, \ldots, 2n$ (possibly with repetitions) because adjacencies on each i-path are confined to the same lex-order, and between paths, each adjacency rotates the lex-order one step. *Q.E.D.*

An example of a route connecting one $2n$-vertex to another is shown schematically in the right panel of Figure 13. Starting from the lower left at the $2n$-vertex in the component of the extraneous vertex, the route moves to the 1-vertex and then exits

the component on a 1-path, passing through one or more 1-ac vertices to reach a new component at the second 1-vertex, which is minimal by definition. From there, forward rotation of the lex-order moves to a maximal j-vertex from which a basis change using the NP rule transits to a boundary (j + 1)-ac vertex and continues on the incident (j + 1)-path through an equilibrium (j + 1)-ac vertex to another boundary (j + 1)-ac vertex, from which a reverse application of the NP rule moves to another maximal j-vertex. Backwards rotation of the lex-order then arrives at a minimal i-vertex with $2 < i \leq j$, from which basis changes according to the RP rule move first to an equilibrium i-ac vertex and then a boundary i-ac vertex from which a reverse NP basis change arrives at an $(i-1)$-vertex that is both maximal and minimal, so continuation is on an $(i-1)$-path. This arrives at another minimal $(i-1)$-vertex that allows forward rotation to a maximal (j + 1)-vertex. Another NP basis change transits to a boundary (j + 2)-ac vertex, from which continuation eventually arrives at the other terminal $2n$-vertex. The second component is simply stable and the collection of vertices in this component constitute a simply-stable subset. To obtain a minimal simply-stable set one can delete redundant vertices so as still to cover all corners; e.g., delete the component's second (non-maximal) minimal vertex, leaving three forming a minimal simply-stable set.

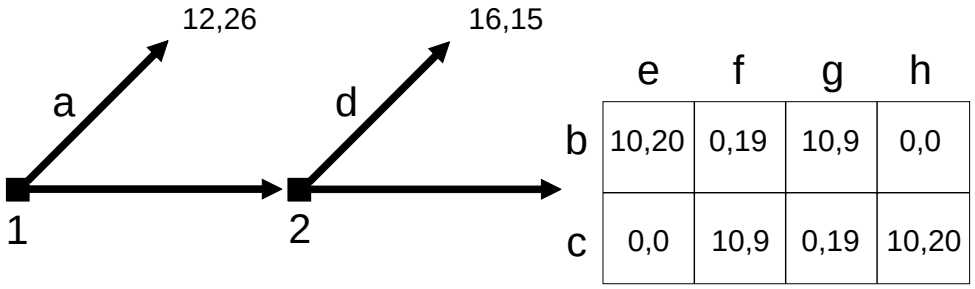
The statement of Theorem 2 suggests that a route is oriented, with a direction from a starting point to a terminus. In fact, routes are uniquely reversible: if the algorithm starts at the terminus then it proceeds along the reversed path to the starting point. Along the reversed route, rotations reverse direction; RP basis changes switch the roles of row and column variables; and, NP basis changes from a boundary vertex and from a maximal vertex switch roles. This is mostly evident in Figure 13 by tracing the route backwards.

The algorithm can be modified to obtain stronger results. For instance, after reaching a $2n$-vertex one can (for the remainder of the current component) drop the restriction on potential adjacency that excludes adjacency between a $2n$-vertex and the initial 1-vertex, and then proceed onward. In an example like Figure 10 this tactic continues the route beyond the $2n$-vertex indicated by the shaded square and continues onto the upper plane where it reverses direction and ultimately finds a second equilibrium indicated by the open circle that covers the initial perturbation, from which it exits to another component. Or, one can continue further to return to the initial equilibrium indicated by the shaded

circle, where the discovery that the initial perturbation is covered an even number of times (twice in this example) prompts an exit from the second one.

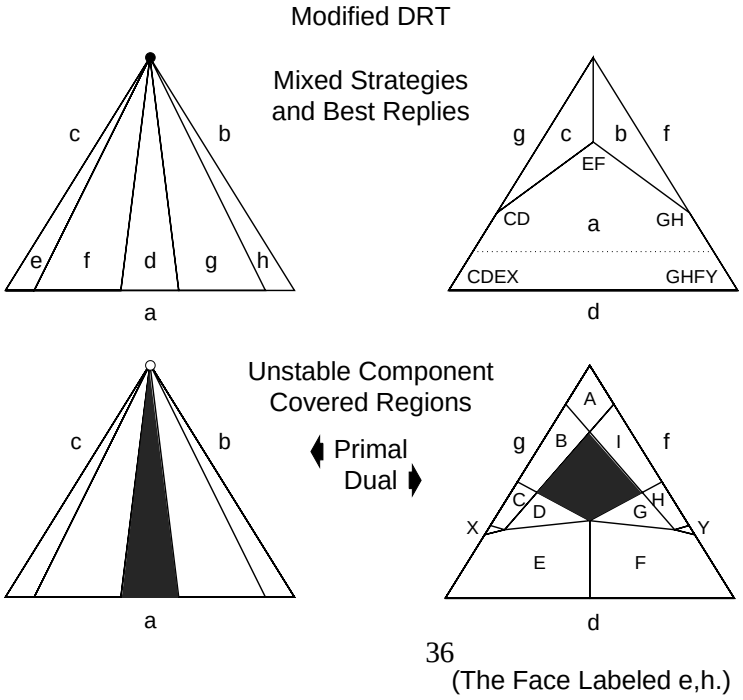
In general, the algorithm ensures only simple stability, but unfortunately I have not found examples where the algorithm terminates in an unstable component; such examples might be rare in practice but presumably they are not nongeneric. The difficulty is that if the path of perturbations intersects merely the closure of an uncovered region then the route exits from a component even if it is simply stable. This typically happens because the route's projection onto the perturbations intersects an uncovered region adjacent to a corner, such as the shaded region in the upper-left panel of Figure 8 where the dotted line shows how the path of primal perturbations intersects the uncovered region. A more elaborate example shows how this occurs even when the component is dual simply stable: Figure 14 shows a modified version of 'Do the Right Thing' and Figure 15 shows the uncovered region of dual perturbations on the face with zero perturbations of e and h (other faces are covered: the uncovered region is therefore the set of convex combinations of the barycenter and points in the shaded region shown). Dual simple stability is obtained only because region A-B-I is covered (accidentally, so to speak) by equilibria using the unperturbed strategies e and h: deleting these strategies yields a game like Figures 6-8 in which the first component is not dual simply stable.

Pictures such as Figure 15 hide the difficulty in higher dimensions of identifying unstable components. A partial but easy test takes advantage of the fact that a component is stable if any (and therefore all) generic perturbation has an odd number of equilibria near the component; cf. Mertens (1986, 1989). One therefore counts the number of coverings of each corner by the vertices generated by the algorithm. In Figure 15 for instance, the computer program noted (by checking signs of coefficients in the tableau) that regions E and F are covered twice and A (and C and H) four times, which suggests that an uncovered perturbation can be found. This test is imperfect because in an unstable component like Figure 10 the plain algorithm could generate an odd covering; moreover, the modified algorithm (mentioned above) could find an even covering when in fact the component is stable with an odd covering, as when there is a third plane that covers the perturbations not covered by the first two. In general, an even or odd covering generated by an algorithm is insufficient evidence without assurance that *all*



Modified Game "Do The Right Thing"

Figure 14: A modification of 'Do the Right Thing'.



(The Face Labeled e,h.)

Figure 15: Mixed strategies and best replies for the modified DRT game, and for the

equilibria covering some generic perturbation have been found. It appears, therefore, that the only sufficient procedures entail essentially complete enumeration as in Mertens (1989, 1991).

Stanford Business School, Stanford, CA 94305-5015, U.S.A.

APPENDIX

THE PIVOTING OPERATIONS

This Appendix describes the numerical operations used in the algorithm and proves the two lemmas.

Gaussian Elimination or 'Pivoting'

Gaussian elimination, or 'pivoting' as it is called in linear programming, is a numerical procedure for recalculating the coefficients in the tableau $\overline{A} \equiv B^{-1} \cdot A$ when the basis matrix B changes by adding one column (called the pivot column) and deleting another (called the pivot row, because in the tableau it is associated with a row). The rule for calculating the new coefficients from the old when the indices of the pivot row and column are r and s is:

$$\bar{a}_{ij} \leftarrow \begin{cases} \bar{a}_{ij}/\alpha & \text{if } i = r\,, \\ \bar{a}_{ij} - \bar{a}_{is}\bar{a}_{rj}/\alpha & \text{if } i \neq r\,, \end{cases}$$

where $\alpha \equiv \bar{a}_{rs}$ is called the pivot.

Regular Pivoting

When moving from one i-vertex or i-ac vertex to another, the regular pivoting or RP rule is used to select the column to be deleted from the basis. In this case the pivot is positive.

Let $[-b, \overline{A}]$ be the tableau for an ℓ-feasible basis. Introducing a nonbasic column into the basis while preserving ℓ-feasibility requires a unique basic column to leave; cf. Eaves (1971). The selection rule is specified as follows. Suppose s is the column of the previously nonbasic column entering the basis and let $S \equiv \{r \mid \bar{a}_{rs} > 0\}$, which must be nonempty or the set of solutions would be unbounded.

THE RP RULE: Define $S_0 = \arg\min\{b_r/\bar{a}_{rs} \mid r \in S\}$, and then recursively for $m = 1, \ldots, 2n$:

$$S_m = \arg\max\{\bar{a}_{r,\ell(m)}/\bar{a}_{rs} \mid r \in S_{m-1}\}\,,$$

until for some $m \geq 0$ sufficiently large, S_m is a singleton set. Its unique member $r(s)$ is the index of the pivot row whose column must leave the basis.

This rule is merely an algorithmic statement of the requirement that afterwards the lex-first nonzero element in each row of the new tableau must be negative. This operation is uniquely reversible. One applies the same rules at the new basis to the case that the newly nonbasic column is to be made basic again.

LEMMA 1: *An i-vertex is potentially adjacent to an equilibrium i-ac vertex if and only if it is minimal.*

PROOF: We show first that an i-vertex that is adjacent to an equilibrium i-ac vertex must be minimal. At an i-vertex there are two possibilities. If the i-th variable z_i is nonbasic then it became so via a basis change using the RP rule from the adjacent i-ac vertex where it was basic and zero, and therefore its column has a positive element in the row of a basic zero variable. If z_i is basic (and basic-zero at the adjacent i-ac vertex), then it is presently basic-zero, and again its coefficient of 1 in its own row is a positive element in the row of a basic-zero variable. In either case, backward rotation of ℓ^i renders the basis infeasible because z_i becomes first in the lex-order ℓ^{i-1}. Conversely, if the i-vertex is minimal then z_i is either basic-zero or nonbasic with a positive coefficient in a basic-zero row. In either case, making basic the nonbasic member of the i-th complementary pair yields an i-ac vertex with z_i's value being zero. Thus the adjacent i-ac vertex is an equilibrium. *Q.E.D.*

Negative Pivoting

When moving from an i-ac vertex to an $(i-1)$-vertex or the reverse, one uses the negative pivoting or NP rule to select the column to be deleted from the basis. In this case the pivot is negative. This case happens only for a boundary i-ac vertex and a maximal $(i-1)$-vertex with $i > 1$. That is, $i \neq 1$ and:

(1) The basic value of z_i is zero at the i-ac vertex, so the i-ac vertex is an equilibrium.
(2) The member of the nonbasic duplicated pair that interacts with z_i has a nonpositive element in each row with basic value zero, and a negative element in the row of z_i, so the i-ac vertex is at a boundary.
(3) The $(i-1)$-vertex basis is *not* i-feasible, so it is maximal.

We deal with the simplest case first. Given a boundary i-ac vertex (B, ℓ^i), one obtains an $(i-1)$-vertex (B'', ℓ^{i-1}) by first rotating backwards the lex-order (thus replacing ℓ^i by ℓ^{i-1}) and then altering the basis by deleting the column of the basic variable z_i and adding the column of the nonbasic duplicated pair's member z_j that interacts with z_i.

CLAIM 1: *The basis B'' is complementary and $(i-1)$-feasible, but not i-feasible; that is, it is a maximal $(i-1)$-vertex.*

PROOF: Because the pivot row's variable z_i has basic value zero, pivoting does not change any basic values. The set of basic variables with zero basic values (the 'basic-zero' rows) therefore remains unchanged except that z_i is replaced by the interacting member z_j of the nonbasic duplicated pair. At basis B, z_j's column of $\overline{A}$ has nonpositive elements in the basic-zero rows and the pivot is negative so the signs of these elements become the signs of the elements in the column of z_i in the basic-zero rows, and this column is now lex-first. The negative elements where they appeared now assure lex-positivity of the basic variable for that row; and where there are zero elements the corresponding row is unchanged, so the first nonzero element in each such row remains negative. Thus, B'' is $(i-1)$-feasible. It is not i-feasible, however, because the lex-second nonzero element of z_j's row is now positive, since it was previously negative. *Q.E.D.*

Reversing this operation is cumbersome because it parallels the RP rule in its need to identify the basic column to be deleted. Suppose that the motion is from a maximal i-vertex (B, ℓ^i) to an $(i+1)$-ac vertex (B'', ℓ^{i+1}), with $i < 2n$. Suppose s is the column of the nonbasic column entering the basis; as shown in Claim 2 this will be the column of z_{i+1} and it will have a negative element assuring that $S \equiv \{r \mid \bar{a}_{rs} < 0\}$ is nonempty.

THE NP RULE: Define $S_0 = \arg\max\{b_r/\bar{a}_{rs} \mid r \in S\}$, and then recursively for $m = 1, \ldots, 2n$:

$$S_m = \arg\min\{\bar{a}_{r,\ell(m)}/\bar{a}_{rs} \mid r \in S_{m-1}\},$$

until for some $m \geq 0$ sufficiently large, S_m is a singleton set and its unique member $r(s)$ is the index of the pivot row whose column must leave the basis.

Again, this rule is merely an algorithmic statement of the requirement that afterwards the lex-first nonzero element in each row of the tableau must be negative. If this rule were used to select the deleted column when one is at a boundary i-ac vertex then (after backward rotation to obtain the cyclic permutation ℓ^{i-1}) it selects the column of the basic

variable z_i, as described previously. That is, z_i's basic value is zero, its column has a 1 in its row and zeros elsewhere, and the pivot column has a negative element in its row, so the index of z_i's row is in S_0 and it is the only member of S_1.

CLAIM 2: (B'', ℓ^{i+1}) *is a boundary* $(i+1)$*-ac vertex.*

PROOF: We first show that B'' is $(i+1)$-feasible. At (B, ℓ^i), z_{i+1} is first in the lex-order and therefore it cannot be basic and zero. Nor can it be basic with a positive value, because if it were then B would be $(i+1)$-feasible; hence z_{i+1} is nonbasic and its column has nonpositive elements in basic rows having zero values. Similarly, some variable must be basic and zero and z_{i+1} must have a negative element in some basic row with zero values, since otherwise B would be $(i+1)$-feasible. Using the basis B and pivoting to introduce z_{i+1} into the basis, therefore, the NP rule must select a column from among those that are basic with zero values; consequently, the basic values remain unchanged after the basis change. Further, the row of the deleted variable must have its lex-second nonzero element positive. That is, if the selected row's lex-second element were negative then B would be $(i+1)$-feasible: if any other basic-zero row having a nonzero (hence negative) element in z_{i+1}'s column had its lex-second element positive then the NP rule would select it. Consequently, after the basis change this element will be negative (because the pivot is negative) and after rotating ℓ^i forward one step this element will be the lex-first nonzero element in the row of z_{i+1}. Among other basic-zero rows, any row for which the column of z_{i+1} has a zero element remains unchanged; in such rows, therefore, the lex-first nonzero element remains unchanged (hence negative) at (B'', ℓ^{i+1}). As for the remaining basic-zero rows, the NP rule assures that after the basis change the lex-second nonzero element will be negative and after rotation it will be first in the lex-order: this is because the sets S_0 and S_1 are identical and therefore the selection of the pivot row essentially begins with S_2. Thus, (B'', ℓ^{i+1}) is $(i+1)$-feasible and it is an $(i+1)$-ac vertex because the only basic duplicated pair is z_{i+1} and its complement. It is an equilibrium because the basic value of z_{i+1} is zero. Finally, it is at a boundary because the newly nonbasic variable is one of the nonbasic duplicated pair, and its column (which is a positive multiple of z_{i+1}'s column at the basis B) has a negative element in the row of z_{i+1} and nonpositive elements in all other basic-zero rows, as did z_{i+1}'s column previously. *Q.E.D.*

LEMMA 2: *An* i*-vertex is potentially adjacent to a boundary* $(i+1)$*-ac vertex if and only if*

it is maximal.

PROOF: This follows from Claims 1 and 2. *Q.E.D.*

REFERENCES

BAGWELL, KYLE, AND GAREY RAMEY (1990): "Capacity, Entry, and Forward Induction," D.P. #888, Northwestern University, mimeo.

BANKS, JEFFREY, AND JOEL SOBEL (1987): "Equilibrium Selection in Signaling Games," *Econometrica*, 55, 647-661.

BRESNAHAN, TIMOTHY, AND PETER REISS (1991): "Empirical Models of Discrete Games," *Journal of Econometrics*, 48, 57-81.

CHO, IN-KOO, AND DAVID KREPS (1987): "Signaling Games and Stable Equilibria," *Quarterly Journal of Economics*, 102, 179-221.

CHO, IN-KOO, AND JOEL SOBEL (1990): "Strategic Stability and Uniqueness in Signaling Games," *Journal of Economic Theory*, 50, 381-413.

EAVES, B. CURTIS (1971): "The Linear Complementarity Problem," *Management Science*, 17, 612-634.

EAVES, B. CURTIS (1973): "Polymatrix Games with Joint Constraints," *SIAM Journal of Applied Mathematics*, 24, 418-423.

GALE, DOUGLAS (1992): "A Walrasian Theory of Markets with Adverse Selection," *Review of Economic Studies*, 59, 229-225.

GLAZER, JACOB, AND ANDREW WEISS (1990): "Pricing and Coordination: Strategically Stable Equilibria," *Games and Economic Behavior*, 2, 118-128.

HILLAS, JOHN (1990): "On the Definition of the Strategic Stability of Equilibria," *Econometrica*, 58, 1365-1391.

HOWSON, JOSEPH T. JR. (1972): "Equilibria of Polymatrix Games," *Management Science*, 18, 312-318.

HOWSON, JOSEPH T. JR., AND ROBERT W. ROSENTHAL (1974): "Bayesian Equilibria of Finite Two-Person Games with Incomplete Information," *Management Science*, 21, 313-315.

JANSEN, MATHIJS, PETER JURG, AND PETER BORM (1990): "On the Finiteness of Stable Sets," Report 9012, Department of Mathematics, Catholic University, Nijmegen, The

Netherlands.

KOHLBERG, ELON (1990): "Refinement of Nash Equilibrium: The Main Ideas" in: T. Ichiishi, A. Neyman, and Y. Tauman (eds.), *Game Theory and Applications*, pp. 3-45. San Diego: Academic Press.

KOHLBERG, ELON, AND JEAN-FRANÇOIS MERTENS (1986): "On the Strategic Stability of Equilibria," *Econometrica*, 54, 1003-1037.

LEMKE, CARLTON E. (1965): "Bimatrix Equilibrium Points and Mathematical Programming," *Management Science*, 11, 681-689.

LEMKE, CARLTON E., AND JOSEPH T. HOWSON JR. (1964): "Equilibrium Points of Bimatrix Games," *Journal of the Society of Industrial and Applied Mathematics*, 12, 413-423.

MCKELVEY, RICHARD D. (1990): "Gambit: An Interactive Extensive Form Game Program," California Institute of Technology.

MERTENS, JEAN-FRANÇOIS (1986): "Localization of the Degree on Lower-Dimensional Sets," CORE Discussion Paper 8605, Université Catholique de Louvain, Belgium.

MERTENS, JEAN-FRANÇOIS (1987): "Ordinality in Non-Cooperative Games," CORE Discussion Paper 8728, Université Catholique de Louvain, Belgium.

MERTENS, JEAN-FRANÇOIS (1989): "Stable Equilibria – A Reformulation, Part I: Definition and Basic Properties," *Mathematics of Operations Research*, 14, 575-625.

MERTENS, JEAN-FRANÇOIS (1991): Stable Equilibria – A Reformulation, Part II: Discussion of the Definition, and Further Results," *Mathematics of Operations Research*, 16, 694-753.

OSBORNE, MARTIN (1990): "Signaling, Forward Induction, and Stability in Finitely Repeated Games," *Journal of Economic Theory*, 50, 22-36.

PONSSARD, JEAN-PIERRE (1991): "Forward Induction and Sunk Costs Give Average Cost Pricing," *Games and Economic Behavior*, 3, 221-236.

ROSENMÜLLER, JOACHIM (1971): "On a Generalization of the Lemke-Howson Algorithm to Noncooperative N-Person Games," *SIAM Journal of Applied Mathematics*, 21, 73-79.

TOMLIN, JOHN (1978): "Robust Implementation of Lemke's Method for the Linear Complementarity Problem," *Mathematical Programming Studies*, 7, 55-60.

VAN DAMME, ERIC (1989): "Stable Equilibria and Forward Induction," *Journal of Economic Theory*, 48, 476-496.

WILSON, ROBERT (1971): "Computing Equilibria of N-Person Games," *SIAM Journal of Applied Mathematics*, 21, 80-87.

WILSON, ROBERT (1972): "Computing Equilibria of Two-Person Games from the Extensive Form," *Management Science*, 18, 448-460.

WILSON, ROBERT (1978): "The Bilinear Complementarity Problem and Competitive Equilibria of Piecewise-Linear Economic Models," *Econometrica*, 46, 87-103.

Footnotes

1. NSF grants SES 8908269 and 9207850 provided financial support and Faruk Gul provided intellectual support. An STSC APL II version of the computer program is available from the author, and a faster C version has been prepared for the game solver Gambit by McKelvey (1990).

2. As in Kohlberg and Mertens, perturbing a strategy's minimal probability perturbs other players' payoffs from all their strategies, whereas perturbing its payoff gives its player a bonus for using that strategy. Thus, like stability, simple stability weakens hyperstability, which considers all payoff perturbations, by considering a restricted set of perturbations. If a game has only pure strategies (e.g., all mixed strategies are represented explicitly as pure strategies) then simple stability implies stability.

3. Van Damme (1989) provides a critique of the 'forward induction' property of stable sets. Economic applications are studied by Bagwell and Ramey (1990), Banks and Sobel (1987), Cho and Kreps (1987), Cho and Sobel (1990), Glazer and Weiss (1990), Osborne (1990), and Ponssard (1991) — among many others. Stability is used also to refine Walrasian equilibria of economic models with features such as signaling and screening; cf. Gale (1992).

4. For the two-player games addressed here, all computations are linear, which allows bypassing some aspects of Mertens' definition; also, for simplicity we apply the definition to stable components, not smaller connected sets. With these provisos, Mertens' definition says essentially that a component is stable if the projection from a neighborhood of the component to a neighborhood of the game is homologically nontrivial. Mertens (1986) shows that this definition implies the necessity of the condition stated in the text; i.e., the projection is onto, so no region is left uncovered. This setup allows a large family of definitions depending on the homology theory used and the normalizations allowed, but he shows that these definitions are essentially equivalent. Mertens (1987) shows that applying a minimality requirement

would violate the desirable property that the solution depends only on the ordinal properties of players' preferences.

5. The algorithm is designed purposely not to examine all vertices, since their number can be enormous. The situation is like the difference between the lengths of a circular orbit around the earth, and a flight over every city.

6. The terms 'generic normal-form' in game-theoretic lingo and 'nondegenerate' in linear-programming lingo are essentially equivalent.

7. Their theorem states that the projection from the graph of the equilibrium correspondence to the space of games is homotopic to a homeomorphism. This means that the graph can be stretched to eliminate folds without leaving holes. They use essentially the space of bonuses to model the space of games, so the graph in Figure 3 illustrates their result along a slice in which only the bonus for strategy a varies. They actually model these spaces as spheres by adding a point at infinity: in Part II we proceed similarly by including an 'extraneous vertex'.

8. The nongeneric perturbations are those in the intersections of covered regions, such as the line between regions A and B, which represent nearby nongeneric games. Typically, a nongeneric perturbation is associated with a 'vertical' segment of the graph where there is a continuum of equilibria.

9. We use **1** to indicate a vector of ones, **0** to indicate a vector or matrix of zeros, and a prime to indicate 'transpose'.

10. This form is used here for expositional purposes. For computations it is more efficient to treat the players' problems separately so that one works with two $m_1 \times m_2$ matrices. In fact, it is sufficient to carry out basis changes via pivoting on the two $m \times m$ submatrices representing the m rows and columns of the players' used strategies. In large games, m is typically small compared to m_1 and m_2.

11. See also Lemke (1965) and Tomlin (1978). Nondegeneracy requires that basic variables always have positive values ($b \gg \mathbf{0}$), which is satisfied by generic normal-form games but not by nontrivial generic extensive games. An extension to degenerate

two-player games is by Eaves (1971). For other cases the extensions assume nondegeneracy: two-player games of incomplete information by Howson and Rosenthal (1974); N-player polymatrix games by Eaves (1973) and Howson (1972); general N-player games, by Rosenmüller (1971) and Wilson (1971); and Walrasian equilibria of piecewise-linear economies, by Wilson (1978). The latter three require nonlinear (actually, multilinear) calculations; for games with three or more players it is therefore often more efficient to use approximation methods based on simplicial subdivisions.

12. The origin of this representation is the observation that the perturbed problem has the analogous form $U \cdot [x + \xi] + I \cdot [y + \eta] = \mathbf{1}$, where $\delta = [\xi, \eta]$. Although each primal perturbation ξ induces a dual perturbation $\eta = U \cdot \xi$, the converse is false and usually there are many other dual perturbations.

13. Unlike an i-vertex, an i-ac vertex's covered region need not include any corner: examples are the equilibria in Figure 7 labeled A and C, which cover the corresponding regions in the upper-right panel of Figure 8. To identify these covered regions it is easiest first to make the basis complementary by replacing one of the basic duplicated pair with one of the nonbasic duplicated pair. For a complementary basis, the inequalities defining the covered region are given by the tableau's rows for basic variables with zero values.

14. If the nonbasic duplicated pair is $\{x_j, y_j\}$ then the member that interacts with x_i is x_j if i and j are strategies of the same player and y_j otherwise; and similarly the member that interacts with y_i is y_j in this event and x_j otherwise. A boundary i-ac vertex has the property that the column of the member of the nonbasic duplicated pair that interacts with z_i has a nonpositive element in each row with basic value zero, and a negative element in the row of z_i. The Appendix provides details.

15. Adjacency obviously depends on both the selected corner (here, only $i = 1$ allows entry and exit from a component) and the numbering of the strategies via the specification of the cyclic permutations ℓ^i. By varying these specifications, the algorithm can find different simply-stable sets.

Legends for Figures

FIGURE 1: Geometric representation of a player's mixed strategies as a simplex with each face labeled by the strategy unused there.

FIGURE 2: Mixed strategies and best replies for an example. The numbered edges record the path of the Lemke-Howson algorithm.

FIGURE 3: The path of the Lemke-Howson algorithm as a homotopy parameterized by the bonus for strategy a.

FIGURE 4: A primal perturbation of player 1's mixed strategies and a dual perturbation of 2's best replies.

FIGURE 5: The left panel shows a vertex of one player's mixed strategies where an equilibrium indicated by the circle allows a primal perturbation in which strategy b is perturbed more than c. The right panel shows the new configuration obtained from the revised perturbation in which c is perturbed more than b. This initiates a path missing label b in which the next step is to use e.

FIGURE 6: The extensive game 'Do The Right Thing'.

FIGURE 7: Mixed strategies and best replies for 'Do The Right Thing'. Vertices of equilibrium components are labeled by the regions of dual perturbations they cover in Figure 8. The numbered edges record part of a route of the algorithm.

FIGURE 8: Primal and dual perturbations covered by the two equilibrium components.

FIGURE 9: A dual perturbation of strategy d that is not covered by the first component. The numbered edges record a path from the first component to the second.

FIGURE 10: Schematic representation of how an uncovered region of perturbations induces an even number of coverings of the covered perturbations. A simply-stable set is obtained on the route from the initial vertex indicated by the black circle to the third indicated by the square. Further continuation encounters the uncovered region and returns to a second equilibrium (open circle) covering the initial perturbation and then to the initial vertex.

FIGURE 11: The Caterpillar Game and the regions of dual perturbations covered by equilibria of the unique component in which 1 uses only a. The graph folds twice over D, which is covered three times.

FIGURE 12: Where the graph folds over to cover again the first perturbation of a segment (so it would exit the component at that point), the algorithm backs up along the sequence of prior segments of perturbations until either the fold reverses — or the fold never reverses and then exit is permitted from the initial perturbation at which the component was entered.

FIGURE 13: In the left panel, each i-vertex is potentially adjacent to four other vertices: precisely one of the left and top adjacencies is feasible, and one of the bottom and right. Similarly, an i-ac vertex has three potential adjacencies and only two are feasible. The implications of Lemmas 1 and 2 are illustrated in the right panel, which displays schematically a route connecting two $2n$-vertices in different components.

FIGURE 14: A modification of the game 'Do The Right Thing'.

FIGURE 15: Mixed strategies and best replies for the modified DRT game, and for the first component, the regions of primal and dual perturbations covered.