The Computational Complexity of Games and Markets: An Introduction for Economists

Andrew McLennan*
University of Queensland

December 2011

Abstract

This is an expository survey of recent results in computer science related to the computation of fixed points, with the central one being that the problem of finding an approximate Nash equilibrium of a bimatrix game is **PPAD**-complete. This means that this problem is, in a certain sense, as hard as any fixed point problem. Subsequently many other problems have been shown to be **PPAD**-complete, including finding Walrasian equilibria in certain simple exchange economies. We also comment on the scientific consequences of complexity as a barrier to equilibration, and other sorts of complexity, for our understanding of how markets operate. It is argued that trading in complex systems of markets should be analogized to games such as chess, go, bridge, and poker, in which the very best players are much better than all but a small number of competitors. These traders make positive rents, and their presence is a marker of complexity. Consequences for the efficient markets hypothesis are sketched.

Running Title: **Complexity of Games and Markets**

Journal of Economic Literature Classification Numbers G12 and G14.

Keywords: Computational complexity, two person games, Nash equilibrium, Scarf algorithm, **NP**, **TFNP**, **PPAD**, FPTAS, Lemke-Howson algorithm, Walrasian equilibrium, arbitrage, asset trading, efficient market hypothesis.

*School of Economics, Level 6 Colin Clark Building, University of Queensland, QLD 4072 Australia, a.mclennan@economics.uq.edu.au. McLennan's work was funded in part by Australian Research Council grant DP0773324. I would like to gratefully acknowledge the comments received at Games Toulouse 2011, and at seminar presentations at the Australian National University, the Institute for Social and Economic Research at Osaka University, and the Kyoto Institute for Economic Research. Tim Roughgarden provided some especially useful comments and advice. All errors are the responsibility of the author. With a project such as this it is inevitable that the some work that should be covered will be missed; I sincerely regret such oversights.

1 Introduction

With the advent of the internet, computer science has become a social science. In addition, computational issues related to the mathematics at the heart of noncooperative game theory and general equilibrium theory have become increasingly prominent. As a result of this, following the large scale research program sketched by Papadimitriou (2001), during the last decade computer science has seen an explosion of interest in game theory and other aspects of theoretical economics. This paper provides an exposition of some of the central results of this literature, and a survey of many others, that is intended for theoretical economists and other game theorists who have little or no prior background in computer science. In addition, it attempts to stake out a position concerning how these developments, and the larger phenomenon of complexity in economic affairs, should affect the scientific outlook of economists.

In a sense this can be thought of as a progress report updating McKelvey and McLennan (1996) and von Stengel (2002). However, those surveys focused on algorithms, and while there is some discussion of algorithms here, our main focus is on computational complexity. In particular, we do not attempt to provide a comprehensive overview of new algorithms for computing Nash equilibrium, its refinements, and correlated equilibrium. Datta (2010) and Herings and Peeters (2010) are recent surveys covering certain types of algorithms, and other papers in the January 2010 issue of *Economic Theory* provide perspectives on recent developments. Nor do we cover the very extensive literature in economics that is concerned with computation of equilibrium, which is largely rooted in economic theory and concrete computational experience. Judd et al. (2003) surveys a portion of this literature that is relatively close to the focus here, because it considers general equilibrium models with multiple agents.

Other references for our main topic aiming at a broader audience (at least within the computer science community) include Papadimitriou (2007), Daskalakis et al. (2009b), Daskalakis (2009), and Goldberg (2011). Longer overviews of the subject, with detailed descriptions of some of the central arguments, are given by Chen et al. (2009b) and Daskalakis et al. (2009a). In comparison with our treatment, these assume little prior background in game theory but at least some background and familiarity with the elements of computational complexity. We should also mention some very useful references that reach out to readers from economics. Roughgarden (2010b) describes the central result exposited here in the context of a broader survey of the computational complexity of equilibrium computation, and Roughgarden (2010a) is a still more expansive overview of work in computer science related to game theory and economics. On a larger scale, the essays in Nisan et al. (2007) provide many perspectives on areas in which economics and computer science are in vigorous interaction. All of these are highly recommended as next steps into the subject.

Our central topic is a result, due to Daskalakis et al. (2006a) and Chen and Deng (2006b), which states that the problem of finding a Nash equilibrium of a finite two player normal form game is "complete" for **PPAD**, which is a class of computational problems that encompasses search problems in which a fixed point of a function of correspondence is sought. In computer science it is a landmark

result resolving a well known longstanding open problem. It has been awarded distinguished prizes, and has been followed up in the research of some of the leading figures of that discipline. In my opinion this is also one of the premier theorems of game theory, and one of the major contributions of game theory to science.

In addition to its conceptual importance, the proof is quite beautiful, with several striking ideas that deserve to be widely appreciated. A distinctive feature of our exposition is that we provide an overview of the proof that is complete and hopefully compelling, in the sense that the reader will find it easy to believe that missing details can be filled in, but also quite informal and (I hope) not at all hard to read. This is possible because the main ideas have geometric descriptions that are intuitive and immediately accessible. What we omit are the nuts and bolts of certain algorithms and the verifications of supporting results that are, for the most part, intuitively plausible, even if the arguments are in some cases quite challenging.

Of course our approach requires firm foundations, so the relevant background material in computer science is introduced from the very beginning. Again, the treatment is informal and much less detailed than what one would find in texts such as Papadimitriou (1994b) and Arora and Boaz (2007), but nonetheless rigorous within the bounds it sets for itself. Indeed, quite aside from the particular subject matter, we hope to provide an introduction to computational complexity that is succinct yet precise, and which conveys some sense of the extent to which this theory is transforming our understanding of computation, and beyond that the larger landscape of mathematics.

What is the conceptual significance of the main result? Very roughly, it says that a wide variety of computational problems related to games, Walrasian equilibrium of exchange economies, and other fixed point problems, are "equally complex." In somewhat more detail, there is a method of passing from any problem in this large class to a two player normal form game whose Nash equilibria are associated with solutions of the original problem. If there was an efficient algorithm for finding a Nash equilibrium of a two player game, then for any problem in this class we would have an efficient three step method for solving the problem: a) pass to the two player game; b) find a Nash equilibrium; c) go from there to a solution of the given problem. There are good reasons to believe that some problems in this class are very hard to solve, so this result constitutes extremely compelling evidence that there will never be an algorithm that can find a Nash equilibrium of a large two player game quickly and reliably.

We should stress that this result does *not* say that computation of a Nash equilibrium of a large two player game is impossibly hard. There are computational methods that are reliable, but sometimes quite slow, and there are other methods that are quick when they work, but not reliable. You just can't have both. More crucially, the most common pattern of software development (start with something that can be proven to work in a practically finite amount of time, then make it better) is infeasible. You must begin with lower aspirations and some more sophisticated view of what you might hope to accomplish.

In addition, the central result does *not* say that all the problems in this large class are "the same" from the point of view of practical computation. Common sense and practical experience suggest that

this can hardly be the case, and that there should be some sense in which some algorithms are better than others. We will have little to say about this later, but not because the issue is uninteresting. Rather, these are issues for which robust, theoretically tractable concepts are currently unavailable. (Analysis of mean running times can provide some useful insights, but it is often mathematically challenging, and it is highly dependent on the distribution of problem instances. For these reasons there is as yet little in the way of useful general theory in this direction.)

Another way in which we can lower our computational ambitions is that we can ask for approximate, rather than exact, solutions. There is an important improvement of the main result that addresses this issue. In general, a *fully polynomial time approximation scheme* (FPTAS) is an algorithm that provides an ε-solution of a numerical problem in an amount of time that is bounded by a polynomial function of the "size" of the problem and $1/\varepsilon$. Chen et al. (2006a) show that (unless all problems in **PPAD** are simple) there is no FPTAS for Nash equilibrium of a two player game. (In this context payoffs are normalized to lie in $[0, 1]$.) On the other hand, Lipton et al. (2003) gave a method, whose running time is bounded by a subexponential function of n, for finding ε-approximate Nash equilibria when ε is on the order of $1/\log n$, where n is a measure of the size of the game. On the whole, relaxing the quality of approximation does not buy much relief from the burden of complexity.

Walrasian equilibrium is one of the premier applications of fixed point theory. For this reason, and also because it is not at all farfetched to imagine using related ideas in the design of protocols for allocating resources such as bandwidth, or in electronic commerce, there has been a large amount of work concerned with the computational complexity of Walrasian equilibrium. There are a number of interesting positive results, but the main findings are negative: finding a Walrasian equilibrium of an exchange economy is already a **PPAD**-complete problem when the given data has quite simple forms, e.g., when the agents have Leontief preferences, and also when they have additively separable utilities with the utility of each good being concave with two linear pieces.

This brings us to the second theme of this paper: how should the results studied here affect our scientific understanding of economic phenomena? In the computer science literature one commonly sees assertions that the results studied here call into question the plausibility or predictive reliability of equilibrium as a description of social or economic reality, and this is certainly correct as far as it goes. Can we go further? The next three sections present some tentative thoughts outlining a more nuanced response to the issue of complexity; at the beginning of the next section there is a broad overview of this portion of the paper.

The exposition of the central result, and our survey of related results, occupies the rest of the paper; its large scale structure is described in Section 5. Section 16 presents some thoughts concerning directions for future research.

2 Complexity and Markets: Classical Views

The central result considered here compels economics to consider the possibility that equilibration may be precluded because it would entail an intractable computation. From a scientific point of view this impetus should be unnecessary: in principle scientists stand ready to consider any possibility until someone supplies an argument showing that it is, for some reason, unrealistic or otherwise uninteresting. Although we will argue that the reasons for failure to equilibrate that the main results point to are quite distinct from others that economists have considered, much of what we will say below pertains to complexity in its much broader and more human sense. For these reasons the relationship between this and the next two sections, on the one hand, and the rest of the paper on the other, is quite tenuous. In addition, this material is discursive, attempting to paint pictures in broad strokes, and much more speculative than what one expects to see in a theoretical journal. In spite of all this, we hope the reader will find it at least somewhat relevant and interesting.

In this section we try to place the issue in historical context. We will argue that the possibility that complexity might be a barrier to equilibration was not imagined by economists until the last half century, and the handling of complexity in the literature since then has not been as sharply defined as is now possible. The next section argues that complex systems of markets can be fruitfully be analogized to complex games of strategy. In games such as chess and bridge there is a very wide range of abilities, and the upper tail of the distribution is thin. This provides a diagnostic of complexity in markets: if a system of markets is complex, we should observe a few participants making large economic rents and a somewhat larger number of people making smaller rents. Section 4 applies these ideas to financial markets, developing specific insights and hypotheses that have not appeared in earlier literature related to this subject, and describing how this perspective can illuminate the dialogue between proponents of the efficient market hypothesis and practitioners of behavioral finance.

The phrase ‘invisible hand’ already connotes a sort of organization and coordination that presupposes some ability to compute things, so it cannot be quite correct to say that economists in the 19$^{\text{th}}$ century and before were completely unaware of the issue studied here, but there is little evidence that it troubled them. Presumably the most important influence on their thinking was the observation that markets mostly seemed to do a tolerable job of adjusting to equate supply and demand.

In the twentieth century there were some prominent economists who stressed complexity as an important feature of economic life. Schumpeter (1976) in particular advanced a view of the economy as an evolving system in which innovation changed the environment, thereby bringing into existence new opportunities to innovate, in an endless cycle of “creative destruction.”

Perhaps the best known appearance of computational considerations in the first half of the 20$^{\text{th}}$ century was in the debates concerning the workability of socialism. Reacting to various theoretical frameworks offered by the proponents of socialism (e.g., Barone (1935), Lange (1938), Lerner (1944), and others cited in Marschak (1959)) von Mises (1990) and Hayek, in a number of essays, argued that socialism would be impossible, or at least severely impaired, by the problem of gathering and

processing economically relevant information. Neither they, nor there antagonists, seem ever to have considered the possibility that capitalism might face similar difficulties. For example, the following passage from Hayek (1945) exhibits no concern that the process generating prices in a capitalist society might also be fragile, or suffer from deficient bandwidth, or be overwhelmed by static, or be incapable of performing sufficiently complex calculations:

> We must look at the price system as such a mechanism for communicating information if we want to understand its real function—a function which, of course, it fulfils less perfectly as prices grow more rigid. (Even when quoted prices have become quite rigid, however, the forces which would operate through changes in price still operate to a considerable extent through changes in the other terms of the contract.) The most significant fact about this system is the economy of knowledge with which it operates, or how little the individual participants need to know in order to be able to take the right action. In abbreviated form, by a kind of symbol, only the most essential information is passed on and passed on only to those concerned. It is more than a metaphor to describe the price system as a kind of machinery for registering change, or a system of telecommunications which enables individual producers to watch merely the movement of a few pointers, as an engineer might watch the hands of a few dials, in order to adjust their activities to changes of which they may never know more than is reflected in the price movement.

This literature is not firmly based in exact models, and it is therefore sometimes difficult to say exactly what its authors had in mind. However, one idea that stands out quite clearly is what economists now describe as decentralization, and computer scientists would recognize as a type of parallel processing. Concretely, any individual agent, say a hardware store owner, might need to learn and react appropriately to dozens or hundreds of prices, but not the millions or billions that are present in the entire system, and although his problems might be rather complex, relative to human cognitive abilities, the information communicated to and from the mechanism might be a low dimensional summary. A very important point is that this sort of parallelism cannot defeat computational complexity of the sort considered later, because a number of processors that is polynomial in the size of the input cannot transform an exponential time computation into a polynomial time computation. But there also seems to be little reason to hope that the problem of finding a fixed point is susceptible to parallelization.

The light shed on the complexity of equilibration by economic theory has mostly arrived rather recently. We should remember that prior to Walras, equilibrium had not even been formulated as a system of equations and unknowns. Uzawa (1962) established that the Debreu-Gale-Kuhn-Nikaido lemma implies the Brouwer (1912) fixed point theorem, so the two results are equivalent in a mathematical sense. The Debreu-Mantel-Sonnenschein theorem (Sonnenschein (1972, 1973), Debreu (1974), Mantel (1974)) implies that any excess demand function of the sort considered by Uzawa is the aggregate excess demand function of an economy, that is, the sum of finitely many individual excess demand

functions generated by utility functions satisfying standard assumptions. Thus the problem of computing a Walrasian equilibrium is, at least in its full generality, as complicated as the problem of computing a fixed point of an arbitrary Brouwer function.

Starting with Marschak (1959) and Hurwicz (1960), economic theorists began formal modelling of the issues that arose in the debate over market socialism, leading to what is now known as mechanism design. For the most part this work can be divided into two streams, according to whether the informational burden or the incentive problems were highlighted. In the former style of work the emphasis was on the interaction between the economic agents, consumers and producers, and the mechanism, which might be a central planner, or a system of decentralized markets. Hurwicz (1960) and Hurwicz et al. (1975a,b) present concrete mechanisms for achieving Pareto optimal outcomes in a range of settings. One main result (Mount and Reiter (1974), Hurwicz (1977), Walker (1977), Osana (1978), Sato (1981), Jordan (1982), Mount and Reiter (1996)) is that the competive process is minimal for the size of the message space among processes that are nonwasteful in the sense of attaining Pareto undominated allocations. A different line of research (Saari and Simon (1978), Saari (1985), Williams (1985), Jordan (1987)) considered the informational requirements of a mechanism that stabilizes an equilibrium, in the sense of converging to a nearby equilibrium after a perturbation, finding that stabilization requires essentially all the information in the matrix of partial derivatives of the aggregate excess demand function.

In this work the amount of information transmitted was modelled using the dimension of the message space, if it was a vector space, or using more general topological notions (Mount and Reiter (1974),Walker (1977)). In spirit these concepts are close to those studied by computer science under the rubric of communication complexity (e.g., Arora and Boaz (2007), Ch. 12) but at a technical level measuring information in terms of real numbers is quite different from measuring it in terms of bits. Nisan and Segal (2001) attempted to span these two perspectives, and provided a detailed comparison.

The second stream of mechanism design, which is concerned with the problem of providing agents with incentives to truthfully reveal their private information, and to do their part in bringing about the social outcome, is by far the larger, and now reaches into many areas of economics and computer science. Here we will only mention some literature related to price formation such as Shapley-Shubik market games (Shubik (1973), Shapley (1976), Shapley and Shubik (1977)). Starting with Gabzewicz and Vial (1975), a number of authors (Novshek and Sonnenschein (1978), Hart (1979, 1980), Novshek and Sonnenschein (1980), Roberts (1980), Mas-Colell (1983), Novshek and Sonnenschein (1983)) examined circumstances under which Cournot equilibrium in economies with many firms can approximate Walrasian equilibrium. This literature is surveyed in Mas-Colell (1982). There has also been some literature (e.g., Pesendorfer and Swinkels (1997)) exploring auctions as a foundation for information aggregation and price formation, but the models are strategically quite complex even in connection with quite simple allocation problems. In particular, it seems to be quite hard to scale these models up to handle a large number of markets simultaneously.

Although explicit recognition of the issue is uncommon, it can safely be asserted that the authors mentioned above were acutely aware that the theories they were analyzing were, in a human sense, quite complex. Certain forms of computation can be seen explicitly in the mechanisms that appear in this literature, but tools for analyzing the complexity of such processes had not yet been developed. Awareness of complexity was no doubt heightened by the increasing prominence of game theory; during a first course in game theory the student quickly learns that even games with a very small number of strategies are beyond human analysis unless they are highly structured; for an instructor trying to design tractable exercises, the issue is even more vivid.

Models of price or strategy adjustment present their own difficulties. There are well known traditional stories, such as the Walrasian auctioneer, concerning the process by which prices come about, but these are often strategically naive. If prices or strategies adjust continuously and predictably, intertemporal arbitrage or strategic anticipation becomes possible, so models of this sort are systematically inconsistent with the principle of rational expectations. Even leaving this problem aside, as was mentioned above an adjustment process must necessarily be quite sophisticated if it is stable, in the sense of reliably returning to equilibrium after small perturbations. For these reasons economists tend to leave price formation as a black box that is generally assumed to happen somehow. It is easy to see how this might cloud economists' thinking about the complexity of equilibration: it is difficult to investigate the circumstances under which some mechanism fails to work as advertised if one has only the vaguest sense of how it might work at all.

Two other streams of literature deserve mention. Each studies phenomena that are quite distinct from our central topic. At the same time both are relevant to complexity in the larger and more human sense that is relevant to the discussion in the next two sections.

During the period surveyed above there was considerable interest in bounded rationality, which was brought to prominence by Herbert Simon and has more recently been studied by Daniel Kahneman, Amos Tversky, and a host of others, both theoretically and experimentally. It is important to understand that complexity of equilibration is quite distinct from bounded rationality. As many economists have pointed out over the years, it is possible that markets reliably equilibrate, and accurately measure value, even when many or most agents have very limited information and cognitive ability. The failure that this paper focuses on is that even if all the agents are fully capable of maximizing their utilities (e.g., because their utility functions are Leontief) equilibration can still be impossible if it amounts to performing a calculation that is precluded by complexity considerations.

The stream of literature on complex systems (e.g., Arthur (2006)) is a more recent elaboration of the Schumpeterian idea. In this style of research an important methodology is to generate computerized interactions of a large (and possibly evolving) collection of agents following rules that are necessarily rather simple. Among other things, this literature emphasizes that even quite simple rules governing individual behavior may lead to complex aggregate outcomes, so that the problems agents are trying to solve are, in an important sense, determined endogenously, and that the system may be characterized by

kaleidoscopic churning. Allowing the rules governing behavior to evolve only reinforces these points. This sort of complexity is quite different from what we study here because, among other things, it does not presuppose any sense of equilibrium or convergence.

3 Markets as Complex Games

Scientific theories and insights are necessarily simple (though they may not seem so to laypeople) so there is an important sense in which complex phenomena are not directly observable. We might suspect that traders are following strategies that are complex and sophisticated, but the actual strategies are not part of the available data. In a complex environment there will still be markets and prices. If the prices are not precisely in line with fundamental values, they will also probably not be ridiculous, and detecting deviations may be beyond the power of our statistical methods, simply because our methods are crude, but also because deviations that might be detected by such methods are systematically noticed, exploited, and dissipated by speculators. These thoughts suggest that complexity might have relatively little effect on our processing of economic data, and while it could linger as an ever present possibility in our thinking about the economy, the overall impact might be slight.

We will argue that the situation is not so hopeless. The first prerequisite to making complexity empirically meaningful is that there should be some observable symptom or signature. We will argue that a distinguishing feature of complex systems of markets is that some traders make significant economic rents. There are two parts to this claim, the first being that in simple markets there are no economic rents accruing to skill or deep understanding. This should be uncontroversial, or perhaps even devoid of content if it is taken as a definition of simplicity. It is, for economists, a completely familiar situation: equilibrium analysis provides an accurate picture because the world is, in its relevant aspects, as simple as our model of it. All agents have sufficient intellectual aptitude to handle the decisions they are called upon to make, so there is no such thing as superior skill that generates rents that cannot be competed away.

The other part of the hypothesis, that in complex situations some agents will make significant rents, has more substance. We conceive of a complex system of markets as a game. In other games of skill the observed range of ability is very large, but this in itself is not enough; rents accruing to ability will be competed away if there are a large number of players who are almost as good as the best in the world. But, as we will document below, in games such as chess and bridge, the best players are distinctly better than those just below them, and one must go to a much lower level before there are a large number of players between that level and the best.

The distribution of skills in chess at the highest level is documented by the Elo rating system. The conceptual underpinning of the system (which is not exactly confirmed in practice) is the assumption that a player's strength in any particular game is the sum of her underlying strength and a normally distributed random shock. For example, if B's rating is 200 points higher than A's, and they play four

games, then on average B should win three and lose one or win two with two draws. The rating is a statistic that passes from the record of play to an estimate of the underlying strength: if in fact A won two games and lost one, with one draw, then A's rating would be adjusted upward and B's would be adjusted downward.

The best chess players in the world have ratings a bit above 2800, the average rating of a tournament player is around 1500, and an average tournament player is much stronger than a novice. Currently (October 2011) there are three chess players with ratings over 2800, but only 47 with ratings over 2700 and 101 with ratings over 2650. Capablanca (the world champion from 1921 to 1927) once went eight years without losing a single tournament game. The history of go and shogi (the Japanese version of chess) exhibit similar patterns, with strong players often dominating the eras in which their strength was greatest. The evidence presented by contract bridge is similar, with eras dominated by certain teams, in spite of the fact that the most common format of competition, namely single elimination tournaments, seemingly provides ample scope for a slightly weaker team to score an upset, especially in view of the many random factors in the game.

Insofar as it is a betting game, poker seems quite analogous to asset trading, and therefore especially pertinent. However, empirical analysis of poker is more difficult for several reasons. For the other games mentioned above success is, by definition, a matter of contending successfully with other strong players. For a poker professional, on the other hand, an important part of financial success is finding her way into games with other players who are well heeled and inept. Holding one's own against other strong players may be thrilling, or intellectually stimulating, or educational, but it is not the primary means by which success is measured. It seems quite reasonable to imagine that different skill sets are involved in playing against weak players and grappling with the best, so that the concept of skill is not necessarily well defined. Poker tournaments are structured to provide entertainment for an audience, with an amount of luck that makes them an unreliable measure of skill. In spite of all this, Levitt and Miles (2011) managed to show that skill does play a significant role: in the 2010 World Series of Poker, players who were top money winners in the 2009 World Series of Poker, or appeared in published lists of top poker players, had an average return on investment of 30%, while other players had an average return on investment of -15%.

The thinness of the distribution of talent near the top is in some ways counterintuitive. If, for example, one took one million draws from the normal distribution, one would expect many draws within one standard deviation of the top order statistic. In each of the games mentioned above those near the top are all seemingly highly intelligent and hard working, and it is difficult to point to any element of unusual luck in the personal histories of the top players that singles them out for greatness. In these games skill is, to a very large extent, a matter of avoiding mistakes, and one might expect that mistakes would be most infrequent at the highest levels of competition. In this connection chess provides an interesting observation: the strongest computer chess programs now have ratings around 3100, so it seems that even the very strongest human players frequently err, if not objectively, in the

sense of making a move that brings about an objectively lost position, then at least in the sense of making a move that would allow a strong computer to force a position in which the human would be unlikely to find her way. In particular, Capablanca's aura of invincibility was probably an illusion.

The games discussed above and asset trading share two significant properties: a) they are zero sum monetarily, or in the sense that the number of wins is equal to the number of losses, but not zero sum in terms of utility; b) people can refrain from playing if they wish. These features suggest a (perhaps somewhat informal) notion of equilibrium in which "market clearing" happens through the participation decision. For the most skillful players participation brings many rewards that clearly offset any costs. As one moves down the spectrum of skills the rewards diminish and the costs increase. Below a certain level players choose not to participate, or they might stop participating after a learning period when frustration with a slow pace of improvement sets in. For gambling games the number of players participating, and their intensity of participation, determine the pool of money at risk, and the redistribution of this money is then determined by the (truncated) distribution of abilities.

4 Reflections on the Efficient Market Hypothesis

Financial markets would seem to be a natural domain in which to explore the ideas advanced above, because the phenomenon of interest occurs there in a relatively pure form. Other business activities, such as using a technical innovation to capture a greater market share, involve complex markets, but they are mixed up with other forms of complexity such as scientific and engineering expertise, and cooperation and competition in social groups. To say that asset trading is a zero sum activity pursued voluntarily by individuals acting alone is, of course, an idealization, but one that is perhaps not too far from reality.

The academic literature on financial markets is dominated by the efficient market hypothesis (EMH). According to Fama (1991) the EMH asserts that "security prices fully reflect all available information."[1] A second, weaker formulation, endorsed as more sensible by Fama, is given by Jensen (1978): "A market is efficient with respect to information set θ_t if it is impossible to make economic profits by trading on the basis of information set θ_t." One point that will emerge below is that the difference between the two formulations may be much larger than Fama seems to imagine.

As is stressed by Fama (1970), empirical tests of the EMH necessarily test jointly an asset pricing model and the hypothesis that market prices agree with those produced by the model. Here we will not be concerned with such tests, though we will accept the general finding that the EMH is not often rejected in such confrontations with the data. Instead, we are primarily concerned with the larger world view that is suggested by taking the EMH literally: assets have fundamental values, and at each time security prices are accurate estimators of those values, based on available information. Although we

[1]The EMH has weak and strong forms that differ according to whether the available information is the publicly known information or, in addition, hidden or "insider" information. The semi-strong EMH goes beyond the weak form in asserting that prices adjust instantly to reflect new public information.

will question certain aspects of this perspective, it may be possible to regard the views advanced here as consistent with the EMH if its scope is construed quite narrowly, as Fama does when he says that "the market efficiency literature should be judged on how it improves our ability to describe the time-series and cross-section behavior of security returns."

Irrational Exuberance by Shiller (2005) presents a dramatically different view of financial markets. Shiller sees asset prices as a reflection of the mood of the investing public, and thus fickle and prone to erratic swings between pessimism and optimism. The evidence for this view is of at least two sorts. Historically the value of financial assets has been much more variable than the realized discounted values of the dividends that were eventually paid out by those assets. Historical data also shows that price/earnings ratios have been negatively correlated with long run returns. To Shiller this suggests that the psychological approach to the study of financial markets utilized in behavioral finance is the most promising research strategy, and that policy and education should aim to tame market fluctuations, and to help investors protect themselves from them.

A Random Walk Down Wall Street by Malkiel (2007) presents an intermediate view. Malkiel recounts a number of investment fads and bubbles, which he regards as clear deviations of prices from fundamental values. At the same time, his recommendation is not to try to actively trade against bubbles, but simply to stay away from bubbly assets. He does not believe that investment professionals have strategies that consistently outperform buying and holding a diversified portfolio.

Here we will try to present a view of asset markets that goes some distance toward reconciling these perspectives. We will agree with Malkiel when he asserts that you and I, and the people who are managing our mutual funds, are incompetent and should use buy-and-hold strategies in order to simply sit on the sidelines. But this is not because strategies yielding excess returns do not exist. On the contrary, for several types of strategies there are some very successful practitioners, but competition between them and somewhat less skillful practitioners drives the expected excess return for amateurs to zero or below. Furthermore, this has several interesting implications, with one of them being that there may well be considerable room for the chaotic or psychologically driven phenomena that Shiller emphasizes.

First of all, it is undeniable that some individuals have had consistent success in trading over long careers, ending up with vast fortunes. Warren Buffett succeeded using the strategy of value investing, in which the investor looks for companies whose stock market values are well below what earnings, book value, and growth prospects would suggest the company is worth. In Buffett (1984) he described several other individuals who succeeded with this strategy, arguing based on detailed examination of their careers that their success cannot reasonably be attributed to luck. Naseem Nicholas Taleb made a fortune betting on tail events that he believed were mispriced by the models used by other investors. Table 1 gives a list of most highly paid Wall Street and bank traders from around the middle of 2008. Given the nature of the information and the source, any individual item in this list might be little more than rumor or unsubstantiated boasting, but the overall picture it portrays does not seem to be in dispute.

It is significant that these are salaries and bonuses, since this presumably reflects a belief on the part of the employer that prior results were due to talent rather than luck.

Top Wall Street & Bank Traders (source: Boring-est Blog)

	Age	Employer	Estimated Income
Michael Hutchins	49	UBS	$30-$40 million
Driss Ben-Brahim	40	Goldman Sachs	$25-$30 million
Ken Karl	46	UBS	$25-$30 million
Jon Wood	42	UBS	$25-$30 million
John Bertuzzi	50	Goldman Sachs	$20-$25 million
Robert Cignarella	36	Goldman Sachs Asset Mngmt.	$25-$30 million
Jeffrey Frase	37	Goldman Sachs	$20-$25 million
Geoff Grant	Early 40s	Goldman Sachs	$20-$25 million
Olaf Refvik	46	Morgan Stanley	$20-$25 million
Neal Shear	50	Morgan Stanley	$20-$25 million
Ashok Varadhan	33	Goldman Sachs	$20-$25 million
Jack DiMaio	37	CSFB	$15-$20 million
Simon Greenshields	49	Morgan Stanley	$15-$20 million
Yan Huo	41	UFJ International	$15-$20 million
Michael Nirenberg	42	Bear Stearns	$15-$20 million
John Shapiro	53	Morgan Stanley	$15-$20 million
Barry Wittlin	47	Merrill Lynch	$15-$20 million
Nasser Ahmad	37	CSFB	$10-$15 million
Charlie W.K. Chan	45	CSFB	$10-$15 million
Nicolas Dusart	32	BNP Paribus	$10-$15 million
Philippe Khuong-Huu	40	Goldman Sachs	$10-$15 million
Angie Long	30	J.P. Morgan	$10-$15 million
Rajiv Misra	Early 40s	Deutsche Bank	$10-$15 million
Aziz Nahas	33	J.P. Morgan	$10-$15 million
Sal Naro	43	UBS	$10-$15 million
Michael Phelps	Early 40s	J.P. Morgan	$10-$15 million
Eric Rosen	Early 40s	J.P. Morgan	$10-$15 million
Christopher Ryan	Late 30s	UBS	$10-$15 million
David Sabath	41	J.P. Morgan	$10-$15 million
Geoffrey Sherry	40	J.P. Morgan	$10-$15 million
Boaz Weinstein	31	Deutsche Bank	$10-$15 million

Table 1

Further evidence for the possibility of profitable speculation comes from Barber et al. (2011) who

analyzed the performance of day traders in Taiwan using a data set consisting of all stock market trades in Taiwan between 1992 and 2006. The vast majority of day traders lose money, but the most skillful earn significant excess returns. In addition, these returns are predictable: the top 1000 traders (less than 1% of all day traders) as measured by success in the preceeding year, earn an average gross (net of commissions and transaction taxes) excess return on their portfolios of 49.5 (28.1) basis points per day.

The first point to take away from this is that trading is even more dangerous than you thought. In the world of the idealized EMH, a chimpanzee who picks stocks by throwing darts does no worse than a hard working mutual fund manager. The examples and results cited above suggest that although the chimp might well be as competent as the mutual fund manager, both suffer losses at the expense of traders who actually know what they are doing. In fact the performance of the chimp may be superior if her picks are truly random. Odean (1999) and Barber and Odean (2001) found that stocks sold by individuals with brokerage account outperformed stocks purchased. Possibly these losses are created by the trading activity itself, since purchases (sales) increase (decrease) prices, thereby making the asset less (more) attractive. But it may also be that in the same way that an experienced poker player understands and exploits the behavior of less skillful players, professional investors are able to take advantage of amateurs. If successful traders make profits on average, their counterparties necessarily make losses, and the simplest way to avoid becoming one of those victims is to trade as little as possible. There is in fact quite a lot of trading that is, if perhaps not entirely irrational, then at least quite unsuccessful except to the extent that the traders derive pleasure from gambling. For example, Barber et al. (2009) find that annual private trader losses on the Taiwanese stock exchanage amount to 2.2% of GDP.

The second line of reasoning suggested by the data above concerns the EMH. The primary causal mechanism leading us to expect that the EMH is approximately correct is that gross deviations from its predictions should be undone by profitable speculation. This suggests that the accuracy of the specific testable implications of the EMH can be understood by assessing the strengths and weaknesses of the relevant trading strategies.

Roughly speaking, a trading strategy can succeed consistently in one of two ways. By making a large number of small trades, each of which has a positive expected excess return, one can get on the right side of the law of large numbers. A smaller number of larger bets can also succeed, but the overall portfolio will be quite risky if the expected excess return of each bet is small in comparison with its standard deviation. In addition, one must bear in mind that large bets compound multiplicatively, which means that a large number of large bets with positive excess return does not necessarily constitute a successful strategy[2]. On top of all these obstacles, successful speculators need to overcome competition with other speculators using similar strategies, including winner's curse effects, and the losses they incur due to the presence of other skillful traders using quite different strategies.

[2]For a concrete example consider a bet that replaces wealth W with $1.2W$ half the time and with $0.82W$ the other half of the time. The expectation of this bet is $1.01W$, but someone who makes such bets repeatedly will see the logarithm of their wealth go to $-\infty$ with probability one, due to the law of large numbers, because $\frac{1}{2}\ln 1.2 + \frac{1}{2}\ln 0.82 = \frac{1}{2}\ln 0.984 < 0$.

Trading strategies that might enforce an equality between stock prices and some notion of fundamental value necessarily involve a smaller number of larger bets. Deviations between the price and the fundamental value of an individual stock may be resolved in the course of a few months, or over a year or two, but value investment strategies that exploit such deviations will not succeed on shorter time scales. In addition, value investing requires extensive research. In order for the expected return of the bet to compensate for this effort, and to overcome the winner's curse, the deviation of price from value must be quite large. In line with the general principles described in the last section, it should not be surprising that some individuals can use such strategies successfully, but most cannot. Strategies that buy or short the market as a whole, according to measures such as aggregate price/earning ratio, require less research, but the aggregate mispricing is likely to be smaller, and to adjust more slowly, than the mispricing of individual stocks.

The converse of this is that the prices of individual stocks, and the aggregate value of the stock market, can wander across a broad range before it becomes easy and safe to exploit the mispricing. That is, value investing allows ample room for bubbles and the other sorts of psychological phenomena stressed by Shiller. To some extent this story has already been told by De Long et al. (1990), who presented a model in which irrational noise traders create volatility that deters rational arbatrageurs from betting aggressively against them, and Shleifer and Vishny (1997), who presented similar models in which the effect is amplified by a principal-agent relationship between an incompletely informed source of funding and an arbitrageur. In the cited models of limits to arbitrage skilled traders are competing with each other; here we point at the additional dangers in the form of skilled traders following quite different strategies.

Now consider a strategy that exploits deviations from an arbitrage, or near arbitrage, relationship between the values of different assets (or the same asset at different times) for example the covered interest parity relationship between the spot and forward exchange rates of two different currencies and the respective interest rates. Such deviations should disappear quickly, so the positions that exploit them will be held for short time periods. If opportunities to exploit such deviations arise frequently, it will be possible to combine many such trades in a strategy that has a high excess return and very little risk, due to the law of large numbers. Consequently one expects that deviations from the relationship should be small and fleeting. Interestingly, Mancini-Griffoli and Ranaldo (2011) document how such a seemingly simple relationship can break down during times of financial stress, along the lines suggested by De Long et al. (1990) and Shleifer and Vishny (1997), due to inability of arbitrageurs to attract financing. Duffie (2010) and Mitchell and Pulvino (2011) are two other recent papers that can serve as entry points to the more recent theoretical and empirical work on arbitrage, which cannot be covered in any systematic way here.

The perspective developed here also suggests a reconsideration of the negative assessment of technical analysis that is the received view of proponents of the EMH such as Malkiel. The trading process generates a wealth of data, and in the author's opinion it would be quite remarkable if, in a world in

which no one used strategies based on technical analysis, there were in fact no such strategies that yielded excess returns. Indeed, one may suspect that the volatility of individual asset prices is bounded more by technical trading than by the stability of information about fundamentals.

Empirical work on technical analysis shows that traditional "chartist" rules for trading do not produce excess returns. However, any successful strategy based on technical analysis that becomes publicly known is likely to be imitated and then become ineffective, so technical traders tend to be secretive about the rules they follow. One can argue that empirical research on technical analysis proves little more than that you are unlikely to be a winner at the poker table if you show everyone your cards before the betting begins. Our general perspective suggests that a small number of traders using technical strategies should enjoy considerable success, while others are somewhat less successful, and that their activities should reduce the returns to technical analysis to the point where only the most disciplined and insightful traders of this sort succeed. Indeed, the most successful day traders studied by Barber et al. (2011) do not seem to be supplying liquidity (that is, systematically exploiting others' eagerness to trade, in the manner of a dealer who posts bid and ask prices which are separated by a spread) or exploiting insider information, and, in view of their frequency of trading, they can hardly be reacting to changing fundamentals, so it seems that their strategies must be based on technical considerations.

The perspective developed here casts a new light on certain policy issues. Shiller (e.g., Shiller (2004)) has advocated the introduction of new asset classes in order to facilitate better risk management. In particular, he has advocated dividend futures associated with stock indices such as the S&P 500. As he has pointed out, historically the realized dividends of stocks have been much less volatile than the prices of the stocks, even though the stock prices are, in principle, estimators of the discounted value of the dividend flows, and should consequently have equal or lower volatility. The reasoning sketched above suggests that, in addition to their value as hedging tools, such assets could help to tame volatility. Without them, the relationship between stock prices and fundamental value is regulated by value trading, which is a weak force. There are arbitrage relationships between stock prices and the prices of dividend futures, and we should expect these relationships to be enforced rather tightly because trading against deviations from arbitrage relationships is a strong force.

In a world in which profitable speculation is impossible because all profitable opportunities have been competed away, few would spend their lives speculating when they could be employed more remuneratively. In the actual world, a certain number of professional traders thrive, many of whom are highly intelligent, disciplined, and hard working, and could presumably contribute a lot if employed in different professions. One possible policy response is a financial transaction tax. Such taxes were more common in the past than at present, but still exist in some countries: Great Britain has a 0.5% tax on stock transactions, and Taiwan has a 0.3% tax. Austria, China, Greec, Hong Kong, Luxembourg, Poland, Portugal, Singapore, Spain, and Switzerland also have such taxes. A big bet by an investment banker on a promising startup might significantly improve the allocation of capital, but it is hard to argue that taking a speculative position in some asset, then clearing it a few hours later, has any such

benefit. In this sense a financial transaction tax is well targeted, punishing the transactions of least value most severely. One might hope that a financial transaction tax would discourage foolish trading by unskilled amateurs. Finally, if a financial transaction tax did not significantly diminish the volume of undesirable trades, then it would at least raise revenue in a way that distorts incentives less than other taxes.

The last possibility seems likely to be the most pertinent. Of the 2.2% of GDP lost by private investors in Taiwan, about 30%, or 0.7% of GDP, is attributable to transaction taxes, and this is a lower bound on the revenue generated because it does not include trade between institutions, so this tax is certainly a significant source of revenue. Since there there seems to have been little or no variation in the rate, it is hard to say precisely how much it deters unwise trading activities, but insofar as most such trading would still be unwise even without such a tax, the deterence effect seems to be less powerful than models based on rationality might suggest. The welfare evaluation of the tax seems less a matter of price distortions than the desirability (moral or otherwise) of discouraging and/or exploiting the propensity of (presumably relatively well off) people to gamble.

Overall, the perspective on financial markets described here seems to work quite well. Looking in some detail at the strategies of traders suggests that arbitrage and near arbitrage relations will typically be enforced rather tightly, and at the same time, in some dimensions considerable leeway remains for prices to wander. This perspective seems broadly consistent with the general failure of the empirical research on the EMH to find deviations from its predictions, but there is still ample scope for the sorts of psychological effects stressed by Shiller. It should be stressed that in our point of view interpreting asset price fluctuations as a straightforward expression of the general public's opinion or mood is unwarranted: the amateurs who trade are a minority that may well be unrepresentative, at least insofar as they seem ill informed about the likely consequences, and in any event what we observe is not simply an opinion poll, but rather the outcome of a game. Finally, our perspective gives some new coloration to certain policy issues. The discussion above is, of course, quite preliminary and tentative, but for all these reasons it seems to be a promising direction for further theoretical and empirical research.

5 Overview

In this section we briefly describe the architecture of the remainder of the paper, which surveys results related to **PPAD**. First of all, the reader is expected to know the basic terminology from graph theory and combinatoric geometry defined in the Appendix, and should begin by at least quickly skimming it, then deciding whether to read it more carefully or consult it as the need arises.

Section 6 begins at the beginning of computer science, explaining computational problems, the Turing machine as a model of computation, the notions of polynomial and exponential time and space, and the intermediate class of problems called **NP**. Section 7 describes reduction as a technique for establishing the relative complexity of different computational problem, and provides a rather detailed

explanation of the Cook-Levin theorem, which shows that some problems in **NP** are complete for this class, which is to say that they are as hard, in this sense, as any others. Section 8 studies Boolean circuits, which provide a model of computation that is closely related to Turing machines, but distinct in certain ways.

The central issue, fixed point computation, is introduced in Section 9. Because of the guarantee of existence provided by Brouwer's fixed point theorem, problems involving computation of fixed points can fruitfully be understood as members of a computational class **TFNP**. Reviewing Sperner's lemma provides a rigorous sense in which the combinatoric version of the fixed point problem is contained in this class. Section 10 explains the Scarf algorithm, which is a path following procedure for finding an object (a "completely labelled simplex") whose existence is guaranteed by Sperner's lemma. Section 11 defines **PPAD** as a subclass of **TFNP** that abstracts the properties of the Scarf algorithm.

PPAD is defined in terms of a computational problem END OF THE LINE that asks for a leaf of a directed graph, other than a given one, whose vertices all have indegree at most one and outdegree at most one. Section 12 lists a number of other computational problems that figure in the subsequent analysis. Some of these (the problem SPERNER of finding a completely labelled simplex, the problem FIXED POINT of finding an approximate fixed point in the setting of Brouwer's fixed point theorem, and the problem NASH of finding an approximate Nash equilibrium of a finite normal form game) are computational renderings of familiar mathematical problems asking for objects that are known to exist. Another, 2D BROUWER, is similar to the two dimensional version of SPERNER, but instead of working with a triangulation of a simplex, it uses a subdivision of the square into smaller squares. Finally, GRAPHICAL GADGET GAME is a specialization of NASH to a particular type of game in which the interaction between the agents is described by a graph.

Section 13 presents a circle of reductions showing that all of these problems are equally hard, in that each of them can be reduced to any of the others. While some of these reductions will be familiar, the reduction from END OF THE LINE to 2D BROUWER, the reduction from 2D BROUWER to GRAPHICAL GADGET GAME, and the reduction from GRAPHICAL GADGET GAME to 2-NASH, are each important results with striking proofs.

The simplex algorithm for linear programming has exponential worst case complexity, but works quite well in practice. The most compelling explanation of this to date is the notion of polynomial smoothed complexity, due to Spielman and Teng (2004). Concretely, although the simplex algorithm may perform poorly on a particular linear program, they showed that if we apply it to a random perturbation of this problem it will, on average, converge rapidly. Passing from the solution of the perturbed problem to a nearby point in the feasible set of the original problem gives a fully polynomial time approximation scheme for linear programming. The Lemke-Howson algorithm for finding a Nash equilibrium of a two person game resembles the simplex algorithm in several ways, so it is natural to ask whether a similar result might obtain. But Chen et al. (2006a) have shown that this is not the case, by showing that finding an approximate Nash equilibrium, for certain error bounds, is a **PPAD**-complete

problem. This result is explained in Section 14.

After 2-NASH was shown to be **PPAD**-complete, attention turned to enlarging the the set of known instances of **PPAD**-complete problems. (The many such problems provided by Kintali et al. (2009) illustrates this phenomenon.) As a premier application of fixed point theory, general economic equilibrium was a natural target. Although in some very special cases surprising and ingenious algorithms showed that an equilibrium could be computed in polynomial time, the set of tractable problems seems to be quite small. In particular, computing a general equilibrium of an exchange economy is **PPAD**-complete even when the agents have Leontief preferences, and also when they have additively separable utility functions in which the utility of each good is concave and piecewise linear with two linear pieces. These and related results are surveyed in Section 15. Section 16 contains some final thoughts.

6 A Brief Introduction to Computation

Theoretical computer science studies what computers can and cannot do. Among the problems that computers can solve, it tries to distinguish between those whose resource requirements (time and memory) are "reasonable," and those that are intractible, in the sense that the resources required to compute a solution grow explosively with the size of the problem.

Fix a nonempty finite **alphabet** Σ. Often $\Sigma = \{0, 1\}$, and while there is not yet any advantage to assuming this, you should think of the number of elements of Σ as a small fixed number. A **computational problem** is a correspondence whose domain and range are the set of nonempty finite strings

$$\Sigma^* = \Sigma \cup \Sigma^2 \cup \Sigma^3 \cup \dots$$

of characters in Σ. That is, for any $x \in \Sigma^*$ in this set there is a nonempty set of valid outputs in Σ^*. For most problems one would expect the input to be formatted in some way, and when we specify that the domain of the correspondence is all strings, we are in effect assuming that "invalid input" is a possible output. This is not without loss of generality because it amounts to a requirement that checking the validity of the input is part of the computational problem.

How might we think about "solving" a computational problem? As with any theoretical endeavor, we need to work with useful abstractions. Perhaps the most basic of these is the Turing machine. Briefly, a **Turing machine** consists of a processing unit connected to a tape reader that holds an infinite (in both directions) paper tape that is divided into squares, each of which contains a character from the extended alphabet $\Sigma \cup \{\text{blank}\}$. An **input** to the machine is a state of the tape in which all but finitely many of the squares contain blank, none of the characters between the leftmost nonblank character and the rightmost nonblank character are blank, and the leftmost nonblank space is in the tape reader's **cursor**. Thus inputs are in one-to-one correspondence with the instances of any computational problem. (Of course various other conventions are possible, and can be found in the literature.)

The processing unit has finitely many states, one of which is a distinguished **initial state**, and some of which are **final states** at which the machine halts. In each period $t = 0, 1, 2, \dots$ the processor reads

the character in the square of the tape that is currently in the tape reader's cursor. This results in a pair consisting of the current state and the character that has been read. The behavior of the machine is characterized by functions that map this pair to:

- a character in $\Sigma \cup \{\text{blank}\}$ that is overwritten on the square of the tape currently in the cursor;
- a motion of the tape either one square to the left or one square to the right, or no motion, so that the current square stays in the cursor;
- a state of the machine in the next period.

Beginning at the initial state and the input, this cycle is repeated until one of the final states is reached, or forever if that never happens[3].

We can now define an **algorithm** for a computational problem to be a Turing machine that computes a solution for every input. By "compute" we mean that the machine eventually halts after printing out one of the valid outputs for the given input. Thus an algorithm is required to halt. A Turing machine that either halts at a solution of the problem or continues forever is called a **computational procedure**.

A computational problem is **computable** if there is an algorithm that computes it. A **decision problem** is a function whose range (for valid inputs) is $\{\text{Yes}, \text{No}\}$. Computational problems that have more complicated outputs are called **search problems**. A very important example of a decision problem that is *not* computable is the HALTING PROBLEM: given a Turing machine and an input, determine whether the machine will eventually halt. (Cf. Papadimitriou (1994b), pp. 58–60.)

As a model of computation, how good is the Turing machine? This question can be broken down into subproblems. First of all, how good is it as a model of algorithms, computational procedures, and what is computable and what is not? Historically, a number of models were proposed, and were shown to be equivalent. The **Church-Turing thesis** is the assertion that "all models of computation are equivalent to the Turing model." Although it seems natural to think of this as a conjecture, it is perhaps more precise to regard it as a definition of what we mean by computation. Thus it can never be proven, but it might be falsified (or, more precisely, rendered obsolete) by a more inclusive model of computation that was realistic, in the sense of being implementable by physical devices.

Turing machines also provide a model of the resource requirements (time and memory) of algorithms. Before asking whether the Turing model is good from this point of view, we again need to settle on theoretically meaningful and tractable abstractions. First of all, we should recognize that the resource requirements of a particular instance of a problem on a particular machine are just numbers that depend on many things in an idiosyncratic way. A more useful way of thinking is to ask how the resource requirements scale as the size of the input increases. What has proved to be most useful is the

[3]Our definition of a Turing machine is, in some ways, simpler than the sorts of definitions one will find in texts. The reason for this is that while we use the concept as a basis for other definitions, we will not describe explicit arguments involving the detailed workings of a Turing machine. If one wishes to describe concrete calculations, it quickly becomes apparent that, as with actual hardware and software, additional features can simplify many tasks.

distinction between algorithms whose resource requirements are bounded by polynomial functions of the input size and those whose resource requirements grow more rapidly than any polynomial function. Rather remarkably, this distinction is the basis of a rich theoretical endeavor, and at the same time it is quite an accurate mirror of the problems that computers are actually able to solve in practice.

We say that an algorithm runs in **polynomial time** if its **running time** (the number of cycles of the Turing machine) is bounded by a polynomial function of the size of the input. (The **size** of $x \in \Sigma^*$, denoted by $|x|$, is its length.) An algorithm requires **polynomial space** if the size of the portion of the tape that it uses during the computation is bounded by a polynomial function of the size of the input. Similarly, an algorithm runs in **exponential time** (**exponential space**) if its running time (space requirement) for input x is bounded by a function of the form $e^{p(|x|)}$ where p is a polynomial. It turns out that the Turing model is a good one in the sense that many other "reasonable" models of computation give the same classification of algorithms. That is, an algorithm runs on your laptop in polynomial (exponential) time (space) if and only if it requires polynomial (exponential) time (space) in the Turing model[4].

We can now begin to classify problems, beginning with **P**, which is the class of all decision problems that have polynomial time algorithms. (We will sometimes abuse notation, extending the meaning of **P** to encompass search problems that have polynomial time algorithms.) Graph theory is a rich source of such problems. For example, REACHABILITY is the problem of deciding whether one vertex in a graph is reachable from another. Since there are obvious polynomial time algorithms that sort the vertices into connected components, REACHABILITY is in **P**. MATCHING is the problem of deciding whether a graph has a perfect matching[5]. This problem is also in **P**, but this is far from obvious. Arithmetic, algebra, and number theory are also obvious sources of computational problems. For example, Gaussian elimination (that is, row and column operations) shows that many decision problems related to determinants and matrix inversion are in **P**. Recently Agrawal et al. (2004) showed that PRIMES, which is the problem of determining whether an integer is prime, is in **P**, answering a question that had been open for millenia.

The class of problems that have exponential time algorithms is **EXP**. Exponential time algorithms are, for the most part, thought to be impractical, except when people are interested in small problem instances. Consequently problems that are in **EXP** but not in **P** are often regarded as "intractable." Especially in connection with computation of fixed points, one should be wary of this attitude, because an algorithm can be quite practical, in the sense of delivering a valid output in a reasonable amount of time often enough to be quite useful, and still have exponential worst case running time.

[4]Quantum computation is now a major frontier of physics. The algorithm of Shor (1997) for factoring integers runs in "quantum polynomial time." It is not known whether there is a Turing machine that factors integers in polynomial time, so it is possible that quantum computation (as a theoretical construct) is more powerful than the Turing model. At the same time, large scale quantum computers have not yet been built, so it is still possible that there are physical principles that preclude them.

[5]A **perfect matching** is a collection of edges such that each vertex is an endpoint of exactly one edge in the collection.

A third important class of problems is **NP**, which is the class of decision problems for which an affirmative answer has a "witness" or "certificate" that can be verified in polynomial time. Such a problem can be specified by a relation $R \subset \Sigma^* \times \Sigma^*$, with the problem being to determine, for a valid input x, whether there is any $y \in \Sigma^*$ such that xRy. (As usual with relations, we write xRy rather than $(x, y) \in R$.) Such a y is called a **certificate** or **witness** for the fact that the decision problem has an affirmative answer. In order for the problem to be in **NP**, there should be a polynomial time algorithm for determining whether xRy, but here "polynomial time" relative to the size $|x| + |y|$ of the input (x, y) is not good enough, because what we really want is that the running time is bounded by a polynomial function of $|x|$. We say that R is **polynomially balanced** if there is a integer $k > 0$ and a constant $C > 0$ such that $|y| \leq C|x|^k$ whenever xRy. In order for R to define a problem in **NP** it must be the case that R is polynomially balanced and there is a Turing machine M that determines whether xRy whose running time is bounded by a polynomial function of $|x|$. (This is possible even though $|y|$ can be arbitrarily large because M can begin by determining whether $|y| < C|x|^k$ without necessarily looking at all of y.)

Note that it is possible that two different such relations define the same decision problem in **NP**. That is, a problem in **NP** may have more than one "system of certificates."

The problem $\text{CLIQUE}_{n,k}$ of determining whether a graph G with n vertices has a clique with k vertices is in **NP** because if such a clique exists, and one has it in hand, one can verify in polynomial time that it is in fact a clique. HAMILTONIAN CYCLE is the problem of determining whether G has a Hamiltonian cycle. Of course it is in **NP** because any Hamiltonian cycle is a suitable witness.

Among the many other complexity classes that have been defined and studied, there are two that the reader should be alerted to. A decision problem is in **coNP** if a negative answer has a certificate that can be verified in polynomial time. That is, **coNP** is precisely the set of negations of problems in **NP**. In itself **coNP** is not of independent interest; our real concern is **NP** $\cap$ **coNP**, which is the class of decision problems for which either answer has an easily verified certificate. The second class is **PSPACE**, which is the class of decision problems that have algorithms that use an amount of space that is polynomially bounded by the size of the input.

Each of the inclusions

$$\mathbf{P} \subset \mathbf{NP} \cap \mathbf{coNP} \subset \mathbf{NP} \subset \mathbf{PSPACE} \subset \mathbf{EXP}$$

is easily verified. For a problem in **P**, the run of a polynomial time algorithm is a certificate for either answer. To see that **NP** is contained in **PSPACE** we observe that for any problem in **NP**, any system of certificates, and any Turing machine processing instance-certificate pairs, applying the Turing machine to each of the possible certificates for a given input is an algorithm for the problem that uses a polynomially bounded amount of space. An algorithm that uses polynomially bounded space must have an exponentially bounded running time because the number of pairs consisting of a state of the machine and a configuration of the relevant portion of the tape is exponentially bounded. (Such a pair cannot occur more than once because if it did the computation would cycle and never halt.)

At present whether **P** = **NP** is one of the most famous open problems in mathematics, but in fact none of the inclusions above have been proven to be strict, and there are a great many other similar open questions. In most cases computer scientists strongly believe that the containment is strict, but the issue is unapproachable due to the intractability of the Turing machine as an object of mathematical analysis. Even though it is quite important in and of itself, the **P** = **NP** problem is, perhaps, best thought of as emblematic of this general state of affairs.

7 Reduction

There is a method for showing that one problem is, in a certain sense, at least as difficult as another, that will be important throughout the remainder, so we define it formally. Let $\mathcal{A}$ and $\mathcal{B}$ be two computational problems. A polynomial time **reduction** of $\mathcal{A}$ to $\mathcal{B}$ consists of two polynomial time algorithms. The first of these passes from an instance p of $\mathcal{A}$ to an instance $r(p)$ of $\mathcal{B}$, and the second passes from a pair (p, o) consisting of an instance of $\mathcal{A}$ and an output for problems in $\mathcal{B}$ to an output $\rho(p, o)$ for problems in $\mathcal{A}$. We require that if o is a valid output for $r(p)$, then $\rho(p, o)$ is a valid output for p.

If there is a reduction from $\mathcal{A}$ to $\mathcal{B}$, then an algorithm for $\mathcal{B}$ can be converted into a three step algorithm for $\mathcal{A}$: (a) pass from p to $r(p)$; (b) apply the given algorithm for $\mathcal{B}$ to $r(p)$, obtaining o; (c) pass from o to $\rho(p, o)$. If we know that $\mathcal{B}$ is in **P** then $\mathcal{A}$ is also in **P**. To see this suppose that the running time of the algorithm passing from p to $r(p)$ is bounded by $f(|p|)$, the running time of the algorithm for $\mathcal{B}$ on input x is bounded by $g(|x|)$, and the running time of the algorithm passing from (p, o) to $\rho(p, o)$ is bounded by $h(|p|, |o|)$, where f, g, and h are polynomial functions. Then the running time of the three step procedure is bounded by

$$f(|p|) + g(|r(p)|) + h(|p|, |o|) \leq f(|p|) + g(f(|p|)) + h(|p|, g(f(|p|)))$$

because the sizes of $r(p)$ and o cannot be greater than the number of machine cycles used up in their computation. Since a sum of compositions of polynomial functions is in turn a polynomial function, it follows that the running time of the three step procedure is polynomially bounded. Similar reasoning shows that if $\mathcal{B}$ is in **NP**, then $\mathcal{A}$ is in **NP**, and if $\mathcal{B}$ is in **EXP**, then $\mathcal{A}$ is in **EXP**. The converse of this is an important method of establishing that a computational problem is hard. Specifically, *if $\mathcal{A}$ is known or thought to be hard in some sense, then $\mathcal{B}$ must be at least as hard.*

This hardness ordering is transitive, because a "composition" of polynomial time reductions is a polynomial time reduction. The argument showing this is based on the same general principle used in the last paragraph: *if the output of a polynomial time procedure is the input of a second polynomial time procedure, then the combined procedure runs in polynomial time.*

A problem is **NP-hard** if it is as hard as any problem in **NP**, in the sense that any problem in **NP** can be reduced to it. A problem is **NP-complete** if it is both in **NP** and **NP**-hard. How can we prove that a computational problem is **NP**-complete? If we have some problem, say $\mathcal{A}$, that is already known

to be **NP**-complete, and there is a reduction from $\mathcal{A}$ to $\mathcal{B}$, then $\mathcal{B}$ is **NP**-hard, hence **NP**-complete if it is in **NP**. This idea can work very well once we have a single concrete **NP**-complete problem, but getting started is hard. The celebrated Cook-Levin theorem gives a first problem that is **NP**-complete. We will spend a bit of time describing the argument, since it is a prototype for much of what we do later.

A **Boolean expression** is an expression, like $((P \vee Q) \wedge \neg R) \wedge S$, that is built up from Boolean variables $P, Q, \ldots$ using negation, conjunction, disjunction, and parentheses. An input to the computational problem SAT is such an expression together with a specification of truth values for some of the Boolean variables, and the problem is to determine whether the expression can be **satisfied**, by which we mean that there are truth values for the remaining Boolean variables that make the expression true. The Cook-Levin theorem asserts that SAT is **NP**-complete.

Evaluating the expression for a given vector of truth values of the Boolean variables can obviously be done in polynomial time, so if a satisfying vector of truth values exists, it is a certificate for the given instance of SAT having an affirmative answer. Thus SAT is in **NP**.

The more significant part of the Cook-Levin theorem is the assertion that SAT is **NP**-hard. Let $\mathcal{A}$ be a problem in **NP**. Our goal is to show that $\mathcal{A}$ can be reduced to SAT. Let M be a Turing machine that verifies a certificate for an instance of $\mathcal{A}$ in polynomial time. An input to M is then the instance of the problem together with a certificate. Let s denote the size of this input, and let $p(s)$ be the polynomial upper bound on the running time of M for an input of size s.

The strategy of the proof is to build a very large (but polynomially bounded) Boolean expression that encodes the run of M on inputs of size s. To this end we introduce a large collection of auxilary Boolean variables. In the following $t = 0, \ldots, p(s)$ denotes a time period, which is to say a cycle of M, ℓ denotes a character in $\Sigma \cup \{\text{blank}\}$, σ denotes a state of the machine, and $i = -p(s), \ldots, p(s)$ denotes a space on the tape[6]. For t, i, ℓ, and σ in their respective domains we have the Boolean variables:

- $T_{ti\ell}$ is true if and only if ℓ is the character in position i of the tape at the beginning of period t.
- I_{ti} is true if and only if i is the position of the cursor at the beginning of period t.
- $L_{t\ell}$ is true if and only if ℓ is the character in the cursor at the beginning of period t.
- $N_{t\ell}$ is true if and only if ℓ is written on the space in the cursor at the end of period t.
- $S_{t\sigma}$ is true if and only if σ is the state of the machine at the beginning of period t.

We now create a Boolean expression E_0 that describes what happens during a run of the Turing machine. This will be a conjunction of a large number of disjunctions of one, two, or three literals, where a **literal** is either a Boolean variable or its negation. There are the disjunctions

$$\neg S_{t\sigma} \vee \neg S_{t\sigma'}$$

[6]We assume that the tape reader starts at position 0, so it cannot move outside the indicated range of the tape during the computation. Also, to simplify the description we adopt the convention that if the machine reaches a final state, it just repeats that state without moving the tape reader until period $p(s)$.

for all t and distinct σ, σ'. These guarantee that there is at most one state of the machine in each period. Similar disjunctions guarantee that in each period there is at most one character in the reader, at most one character in each position of the tape, and at most one character written by the tape reader.

Next are the disjunctions that describe the machine's transitions. For example, if σ' is the next state when σ is the current state and ℓ is the character in the tape reader, then for each $t = 0, \ldots, p(t) - 1$ it must be the case that

$$\neg S_{t\sigma} \vee \neg L_{t\ell} \vee S_{t+1,\sigma'}.$$

There are similar disjunctions describing the new character that is written in the position under the tape reader and the motion of the tape. There are also disjunctions of the form

$$\neg T_{ti\ell} \vee I_{ti} \vee \neg T_{t+1,i\ell'}$$

where $\ell \neq \ell'$. Collectively these require that if i is not the position of the tape reader at the beginning of period t, then the character in position i at the beginning of period $t + 1$ is the same as the character in position i at the beginning of period t.

This is a huge mess, but the underlying idea is simple and obvious. Consider a specification of truth values for the Boolean variables $T_{0i\ell}$, I_{0i}, $L_{0\ell}$, $N_{0\ell}$, and $S_{0\sigma}$ for all ℓ, i, and σ that is consistent and conforms to the description of the Turing machine: each space on the tape contains a unique character, the initial state of the machine is the prespecified one, the initial position of the cursor is 0, and the character in the cursor is the character at position 0 on the tape. Then there is unique specification of the truth values of all other Boolean variables such that E_0 is satisfied, and these collectively describe the run of the machine.

We now build a Boolean expression E_1 that is true if and only if E_0 is satisfied and the final state is one in which the certificate has been verified. Formally, E_1 is the conjunction of E_0 and all literals $\neg S_{p(t),\sigma}$ where σ is a state of the machine that is *not* a final states in which the certificate has been verified. Then, for each consistent specification of the truth values of the Boolean variables describing period 0—that is, for each problem instance and certificate—there is collection of truth values for the remaining variables such that E_1 is satisfied if and only if the run of the Turing machine confirms that the certificate is indeed a witness for the problem instance having an affirmative answer.

Finally, we construct E_2 by specifying the truth values of the Boolean variables describing the instance of the problem and the initial state of the machine and position of the cursor, do *not* the truth values of the Boolean variables describing the certificate. Evidently E_2 can be satisfied if and only if there is *some* certificate that verifies that the problem instance has an affirmative answer.

The Boolean expression that we have created has a special form. It is in **conjunctive normal form**, which means that it is a conjunction of disjunctions of literals. In addition, if you fill out the details of the construction you will see that each of the disjunctions is a disjunction of one, two, or three literals. The restriction of SAT to Boolean expressions with these properties is called 3-SAT. Since the Turing machine is fixed, obvious counting arguments show that the number of Boolean variables,

and the number of disjunctions, are bounded by polynomial functions of $p(s)$ and thus by polynomial functions of s. Finally, note that we have not only shown that there is a Boolean expression with the desired properties, but have also described (rather informally) an algorithm for constructing it, and it is obvious that the running time of this algorithm is bounded by a polynomial function of s. Thus we have reduced $\mathcal{A}$, an arbitrary problem in **NP**, to 3-SAT, thereby showing that 3-SAT is **NP**-hard. Since 3-SAT is in **NP**, it is **NP**-complete.

In the wake of the Cook-Levin theorem a very large number of problems have been shown to be **NP**-complete, including CLIQUE$_{n,k}$ and HAMILTONIAN CYCLE. In particular, Gilboa and Zemel (1989) showed (Conitzer and Sandholm (2003) and McLennan and Tourky (2010) gave alternate proofs) that a large number of decision problems related to Nash equilibrium of two player normal form games are **NP**-complete. (Is there only one equilibrium? Does there exist an equilibrium attaining a certain utility for all players? Does there exist an equilibrium whose support includes/excludes a given set of pure strategies? Does there exist an equilibrium whose support has at least/most a given number of pure strategies?) However, although these results suggest that finding a single Nash equilibrium is hard, they do not bear directly on the difficulty of that computational problem, for reasons explained in greater detail in Section 9.

The theory of **NP**-completeness has enormous practical and theoretical importance. Among other things, instead of trying to come up with clever algorithms for each of the many **NP**-complete problems, we can tackle a single **NP**-complete problem that seems simple, or presents some favorable feature. On the other hand, we can just leave all the **NP**-complete problems alone because, at this point, resolving $\mathbf{P} = \mathbf{NP}$ seems hopeless, or for some specific problem we can adopt goals that are more modest than an algorithm that solves every possible instance quickly.

The notion of **NP**-completeness gives the $\mathbf{P} = \mathbf{NP}$ problem a very concrete flavor: any particular **NP**-complete problem, e.g., CLIQUE$_{n,k}$, either has a polynomial time algorithm or it doesn't. Concrete experience with algorithms for **NP**-complete problems, and the fact that $\mathbf{P} = \mathbf{NP}$ would have consequences that are quite remarkable and counterintuitive, has led almost all computer scientists to believe that it is very unlikely that there is a polynomial time algorithm for any **NP**-complete problem, so $\mathbf{P} \neq \mathbf{NP}$ is regarded as an established (which is not quite the same thing as proven) fact. The numerous applications of the theory to cryptography and other aspects of computer science have enormous importance, but are far beyond our scope. Overall, the definition and analysis of **NP** has been a huge success.

Although it is a bit aside from our main topic, there is another issue related to **NP** that I believe should be well known to game theorists. At present problems that might be in $(\mathbf{NP} \cap \mathbf{coNP}) \setminus \mathbf{P}$ are rather scarce[7]. Condon (1992) defined a class of two player zero sum stochastic games, called **simple**

[7]Factoring is a one source of problems of this sort. For example, the question of whether a given integer has a prime factor whose last digit is 3 is in $\mathbf{NP} \cap \mathbf{coNP}$ because a prime factorization is a witness, for either possible answer, that can be verified in polynomial time. (This follows from the fact that PRIMES is in **P**, but it has actually been known since Pratt (1975) that PRIMES is in **NP**, which already implies that each prime in the factorization has a witness that can be verified in polynomial

stochastic games, that are a potential source of problems in (**NP** ∩ **coNP**) \ **P**. Parity games are a particularly interesting class of simple stochastic games. In a **parity game** there is a directed graph $G = (V, A)$. There is a token that begins at a designated initial vertex v_0 and is moved around the graph. There are two players 0 and 1 who take turns moving the token, with player 0 moving first. A legal move consists of moving the token from its current position to the head of one of the arrows that has the current vertex as its tail. (In order to simplify the discussion we assume that every vertex has positive outdegree. In a slightly more general version a player loses if she has no legal move.) Thus a play of the game is a sequence $v_0, v_1, v_2, \ldots$ with $(v_t, v_{t+1}) \in A$ for all t. There is a partition $V_1, V_2, \ldots, V_r$ of V into "parity classes." Player 0 wins and player 1 loses the game if the largest i such that $v_t \in V_i$ for infinitely many t is even, and otherwise player 0 loses and player 1 wins.

In a parity game one of the two players has a stationary strategy that forces a win[8]. In addition, for a given pure strategy of player 0 it is easy (i.e., polynomial time computable) to determine whether player 1 has a winning response. Thus the question of whether player 0 can force a win is in **NP**, but it is also in **coNP** because a stationary pure strategy for player 1 that forces a win is a suitable certificate for player 0 being unable to force a win. Andersson and Miltersen (2009) show that a large number of problems, including solving simple stochastic games and solving the generalization of parity games in which the successors of some nodes are chosen randomly, are equally difficult, by virtue of polynomial time reductions. That paper provides a fairly comprehensive overview of related literature.

8 Boolean Circuits

In this section we consider a different model of computation that passes from a vector of input bits to a vector of output bits. The computation is implemented by a network of gates performing elementary logical operations.

Formally a **Boolean circuit** is an acyclic directed graph $C = (V, A)$ whose vertices of indegree zero are called **input vertices** and whose vertices of outdegree zero are called **output vertices**. Let V_i and V_o be the sets of input and output vertices. Vertices in $V \setminus V_i$ are thought of as logic gates. Different types of circuits are obtained by allowing different types of gates; here it will suffice to consider circuits

time.) There is no known polynomial time algorithm for factoring, and no known proof that there is no such algorithm, so at present we cannot say whether such problems are in **P**.

[8]One way to show this is to consider a discounted version in which there are numbers $u_1, \ldots, u_r$ with $u_1, u_3, u_5, \ldots < 0$ and $u_2, u_4, u_6, \ldots > 0$, and a discount factor $\delta \in (0, 1)$. If the sequence of vertices reached in the play is $v_0 \in V_{i_0}, v_1 \in V_{i_1}, v_2 \in V_{i_2}, \ldots$, then player 0's payoff is $\sum_{t=0}^{\infty} \delta^t u_{i_t}$ and player 1's payoff is the negation of this. Standard facts about discounted zero sum stochastic games imply that each pair consisting of a vertex and a player to move has a value that will be attained if the game starts there and both players play optimally. Moreover, any pair of stationary pure strategies in which both player always choose value maximizing moves is a Nash equilibrium. A bit of algebra shows that if the ratios $|u_2|/|u_1|, \ldots, |u_r|/|u_{r-1}|$ are sufficiently large and δ is sufficiently close to 1, then player 0's equilibrium payoff is positive (negative) if and only if any stationary Nash equilibrium pure strategy for player 0 (player 1) forces a win in the undiscounted version.

with gates for binary conjunction, binary disjunction, and negation. Thus we require that $V \setminus V_i$ has a partition $V_\wedge \cup V_\vee \cup V_\neg$ such that the indegree of each vertex in $V_\wedge \cup V_\vee$ is two and the indegree of each vertex in $V_\neg$ is one.

A **Boolean function** is a function $f : \{0,1\}^X \to \{0,1\}$ where X is a finite set. We think of the circuit C as a device that computes a $|V_o|$-tuple of Boolean functions $f^C : \{0,1\}^{V_i} \to \{0,1\}^{V_o}$ by inserting a vector $t \in \{0,1\}^{V_i}$ into the input nodes, then propogating truth values through the various gates, under the usual identification of 0 with False and 1 with True. The function $f^C : \{0,1\}^{V_i} \to \{0,1\}^{V_o}$ is the composition of a function $\tilde{f}^C : \{0,1\}^{V_i} \to \{0,1\}^V$ with the natural projection $\{0,1\}^V \to \{0,1\}^{V_o}$, where $\tilde{f}^C$ is given by the obvious rules:

- If $v \in V_i$, then $\tilde{f}^C_v(t) = t_v$.
- If $v \in V_\wedge$ and v_1, v_2 are its direct predessors, then $\tilde{f}^C_v(t) = 1$ if $\tilde{f}^C_{v_1}(t) = 1$ and $\tilde{f}^C_{v_2}(t) = 1$, and otherwise $\tilde{f}^C_v(t) = 0$.
- If $v \in V_\vee$ and v_1, v_2 are its direct predessors, then $\tilde{f}^C_v(t) = 1$ if $\tilde{f}^C_{v_1}(t) = 1$ or $\tilde{f}^C_{v_2}(t) = 1$, and otherwise $\tilde{f}^C_v(t) = 0$.
- If $v \in V_\neg$ and v' is its direct predessor, then $\tilde{f}^C_v(t) = 1 - \tilde{f}^C_{v'}(t)$.

Since C is acyclic, it is obvious and easy to prove formally that there is a unique function $\tilde{f}^C$ satisfying these conditions.

Evidently Boolean expressions and Boolean circuits with a single output node are two different ways to encode a Boolean function, and it is easy to see how to pass from one to the other. (As we have defined them, Boolean expressions allow conjunctions and disjunctions of more than two subexpressions, but this is just a matter of adding extra parenthesis to the Boolean expression.) We now wish to understand the relation between Boolean circuits and Turing machines.

Henceforth we assume that $\Sigma = \{0,1\}$. The details are a bit messy, but in the digital age it is obvious that this restriction is without loss of generality, from the point of view of issues involving complexity, because an arbitrary Turing machine can be converted into one satisfying this conditions by equipping it with a) a translator that passes back and forth between a larger alphabet and a binary encoding of it; b) modules that give binary renderings of the given operations that were defined in terms of the original alphabet.

We now reconsider the construction used in the last section to prove the Cook-Levin theorem. In that construction we passed from a Turing machine and a size of input s to a logical formula with variables T_{til}, I_{ti}, $L_{t\ell}$, $N_{t\ell}$, and $S_{t\sigma}$ that is satisfied if and only if the truth values of these variables describe the run of the machine. The details are a bit different, but the same general ideas could obviously be used to pass from the Turing machine and s to a Boolean circuit that computes the values of all these variables. That is, *for any Turing machine whose running time is polynomially bounded, and any input size s, there is a Boolean circuit C_s that describes the run of the machine on inputs of*

size s. In fact we can say a bit more: there is an algorithm that passes from the machine, s, and the polynomial bound $p(s)$, to the circuit, in time bounded by a scalar multiple of $p(s)$.

Once we restrict to the sorts of binary inputs described above, a decision problem can be identified with a sequence of Boolean functions

$$f_1 : \{0,1\} \to \{0,1\},\ f_2 : \{0,1\}^2 \to \{0,1\},\ f_3 : \{0,1\}^3 \to \{0,1\},\ \ldots$$

expressing the desired outputs for the inputs of various sizes. Any Boolean function has a (possibly exponentially large) Boolean circuit, so there is a sequence of Boolean circuits $C_1 = (V_1, A_1), C_2 = (V_2, A_2), \ldots$ that compute the functions $f_1, f_2, \ldots$. If the decision problem is in **P**, then there is a polynomial function p such that there is a sequence of circuits with $|V_s| \leq p(s)$ for all s.

We hasten to point out that the existence of a sequence $C_1, C_2, \ldots$ of Boolean circuits for $f_1, f_2, \ldots$ with particular properties usually does not imply much about the decision problem. Among other things, it does not follow that the problem is computable; after all, there is a sequence of Boolean circuits for HALTING PROBLEM. In particular, if there is a sequence $C_1, C_2, \ldots$ for $f_1, f_2, \ldots$ with the size of C_s bounded by $p(s)$, where p is a polynomial, it does *not* follow that the problem is in **P**. Roughly speaking, the difficulty is that the circuits can be arbitrarily diverse.

In order to create concepts of circuit complexity for decision problems that are closer to the Turing model, we can require that the sequence $C_1, C_2, \ldots$ be **uniform**, by which we mean that there is a Turing machine that takes s as input and outputs C_s, and that satisfies certain bounds on its resource requirements and/or various attributes of C_s. (For example, the **depth** of a Boolean circuit is the maximum length of any walk in it.) This turns out to be a fruitful method of defining interesting complexity classes, but that topic is beyond our scope.

Now suppose that the sequence $f_1, f_2, \ldots$ expresses an **NP**-complete decision problem. For each s let n_s be the minimum of $|V_s|$ over all circuits $C_s = (V_s, A_s)$ that compute f_s. If we could show that the sequence $n_1, n_2, \ldots$ is not bounded by any polynomial function, then it would follow that **NP** $\neq$ **P**. This is seemingly quite a sensible approach to the problem: in exchange for accepting the burden of proving a slightly stronger assertion, we pass from Turing machines, which seemingly defy mathematical analysis, to Boolean circuits, which are at least built up from simple elements according to simple rules. Indeed, it is difficult to imagine any attempt to prove **NP** $\neq$ **P** that does not begin with this maneuver.

This approach has provided evidence that **NP** $\neq$ **P** is, in a certain sense, "highly probably." Shannon (1949) compared the number 2^{2^s} of Boolean functions $\{0,1\}^s \to \{0,1\}$ with the number of Boolean circuits of various sizes, arriving at the conclusion that there is a constant $K > 0$ such that "almost all" such Boolean functions require a circuit of size $K2^s/s$. Remarkably, there are no known concrete instances of sequences of Boolean functions with exponential circuit complexity.

A Boolean function $f : \{0,1\}^s \to \{0,1\}$ is **monotone** if $t \leq t'$ implies that $f(t) \leq f(t')$. Less formally, flipping input bits from 0 to 1 can never result in the output changing from 1 to 0. A Boolean circuit $C = (V, A)$ is **monotone** if $V_\neg = \emptyset$, so that negation is never employed. Any monotone boolean

function can be computed with a monotone circuit; to show this we simply pass from an enumeration of all minimal inputs that are mapped to 1 to a monotone circuit that computes, in the obvious way, whether the input is "at least as large" as one of these. The problem $\text{CLIQUE}_{n,k}$ is an **NP**-complete problem that is monotone: adding an edge to the graph might create a clique, but it can never destroy one. Perhaps the closest approach to **NP** = **P** to date is a remarkable result of Razborov (1985b) which (after being strengthened by Andreev (1985) and Alon and Boppana (1987)) asserts that the minimal size of a monotone circuit for $\text{CLIQUE}_{n,n^{1/4}}$ is bounded below by an exponential function. But Razborov (1985a) himself showed that there are monotone problems in **P** whose minimal monotone circuits have superpolynomial size, so this is not quite enough.

9 Fixed Points and Total Problems in NP

We now turn to the specific computational issues that are the central concern of this paper. Suppose that D is a nonempty compact convex subset of a Euclidean space and $f : D \to D$ is a continuous function. A **fixed point** of f is an $x^* \in D$ such that $f(x^*) = x^*$. Knowing that f has a fixed point is one thing (that is what Brouwer's fixed point theorem asserts) and finding one is quite another. During the last two decades computer scientists have learned a lot about this problem.

We begin with a simple and, when you think about it, obvious point. Computing an exact fixed point may be computationally infeasible simply because the components of the fixed point are transcendental numbers that do not have finite representations. At best we can hope for an answer that is correct in some approximate sense.

We should also distinguish between various versions of the approximate problem. One might imagine that, for a given $\varepsilon > 0$, our hope is to find a point $x^* \in D$ such that the ε-ball centered at x^* contains a fixed point of f. If the only a priori theoretical restriction on f is that it is continuous, and our only source of additional information concerning f is an "oracle" that accepts a point $x \in D$ as an input and outputs $f(x)$, then this is unreasonably ambitious because there is no algorithm for this problem. To see this suppose that, on the contrary, some algorithm consulted the oracle to learn the values of f at $x_1, \ldots, x_k$ (none of which are fixed points) and then declared that the ε-ball around x^* contained a fixed point. The algorithm would behave in the same way if f was replaced with $h^{-1} \circ f \circ h$ where $h : D \to D$ is a homeomorphism with

$$h(x_1) = x_1,\ h(f(x_1)) = f(x_1),\ \ldots\ ,\ h(x_k) = x_k,\ h(f(x_k)) = f(x_k).$$

Provided that the dimension of D is greater than one and f has finitely many fixed points, it is easy to see that h can be chosen in such a way that $h^{-1} \circ f \circ h$ has no fixed points in the ε-ball centered at x^*.

If we have more a priori information about f it can become possible, in principle, to find an approximation of a fixed point. In particular, when f is semi-algebraic (that is, described by polynomial equations and inequalities) there are known algorithms for the general problem, and also for Nash equilibrium (e.g., Lipton and Markakis (2004)) but their complexity is daunting, and it seems likely that

these difficulties are inherent. (Cf. Etessami and Yannakis (2010).) In any event, methods that depend on additional a priori information necessarily represent a piecemeal approach.

Instead of looking for a point that is guaranteed to approximate a fixed point of f, we might do better to look for a point x that is approximately fixed in the sense that $\|f(x) - x\| < \varepsilon$. Actually, we will look for another type of structure that can be understood as a proxy for such a point. Specifically we recall Sperner's lemma and how it is used to prove Brouwer's fixed point theorem.

Fixing a dimension n, suppose now that D is the **standard n-dimensional simplex**:

$$D = \{\, x \in \mathbb{R}^{n+1}_{\geq} : x_0 + \cdots + x_n = 1 \,\}.$$

Let $\mathcal{K}$ be a simplicial subdivision of D, and let V be the set of vertices of $\mathcal{K}$. A **Sperner labelling** for $\mathcal{K}$ is a function $\ell : V \to \{0, \ldots, n\}$ such that $v_{\ell(v)} > 0$ for all v. That is, $\ell(v) = i$ implies that $v_i > 0$. A simplex $\sigma \in \mathcal{K}$ is **completely labelled** if the set of labels of its vertices is exactly $\{0, \ldots, n\}$. Evidently a completely labelled simplex must be n-dimensional, and the labelling must put its vertices in one-to-one correspondence with the indices $0, \ldots, n$. Sperner's lemma (which will be proved in the next section) asserts that if ℓ is a Sperner labelling, then there is a completely labelled simplex.

To prove Brouwer's fixed point theorem we take a sequence $\{\mathcal{K}^r\}$ of simplicial subdivisions of D with the mesh of $\mathcal{K}^r$ going to zero as $r \to \infty$. For each r there is a Sperner labelling ℓ^r_f defined by letting $\ell^r_f(v)$ be the smallest index i such that $v_i > f_i(v)$. (It is ironic that the possibility that v is a fixed point creates a small complication here; we can specify that in this case $\ell^r_f(v)$ is the smallest i such that $v_i > 0$.) Sperner's lemma implies that for each r there is a completely labelled simplex for ℓ^r_f. After passing to a subsequence, we may assume that the sequence of completely labelled simplicies converges to a point x^*, so for each $i = 0, \ldots, n$ there is a sequence $\{v^{i,r}\}$ of vertices with $v^{i,r}_i \geq f_i(v^{i,r})$ for all r and $v^{i,r} \to x^*$, which implies that $x^*_i \geq f_i(x^*)$. Since $\sum_i x^*_i = \sum_i f_i(x^*) = 1$ we have $f(x^*) = x^*$.

This argument gives a sense in which a completely labelled simplex can be taken as a proxy for a fixed point. There is no absolute guarantee that a completely labelled simplex is close to an actual fixed point, or even to a point that is approximately fixed, but for functions that are not too "wild" it should mostly be the case that a completely labelled simplex is, in fact, a good approximation of what we really want.

How should we think of finding a completely labelled simplex as a computational problem, and how hard is it? If the input of our algorithm included a list of all the simplices in the triangulation, the computational burden would not be very large, in proportion to the size of the input. We could simply evaluate f at each vertex, assign the corresponding label, and then search over all the n-dimensional simplices until we found one that was completely labelled. This sounds too easy, and of course what this line of thought actually points out is that it is unreasonable to assume that the input has such a verbose form.

We need to think more carefully about how the computational problem is formulated. Sperner's lemma has two ingredients, namely a triangulation and a method of assigning a valid label to each vertex of the triangulation. We consider each of these in turn.

We cannot really say that an attack on the problem is successful unless we are in a position to find a completely labelled simplex of a triangulation with arbitrarily small mesh. In the Appendix we describe how repeated barycentric subdivision can give such triangulations, but this is recursive, hence difficult to describe algorithmically, and inefficient (the simplices rapidly become extremely thin) to such an extent that its asymptotic computational burden is opaque. A description of industrial strength techniques (e.g., van der Laan and Talman (1982)) would be out of place here, but in order to support various complexity assertions later we will mention one simple method that gets the job done.

For any integer k one can first form the polytopal division of D consisting of its intersections with cubes of the form

$$[i_0/k, (i_0+1)/k] \times \cdots \times [i_n/k, (i_n+1)/k],$$

and all the faces of such intersections. Each polytope P in this subdivision is the intersection of D with one of the faces F of such a cube. When the dimension of P is positive one can take β_P to be the point where D intersects the diagonal of F going between its minimal and maximal vertices. The desired triangulation is obtained by taking the barycentric subdivision of the polytopal subdivision of D with respect to $\{\beta_P\}$. This is highly inelegant, because the intersections above come in a variety of sizes and shapes, but at least it should be evident that one could construct an algorithm that can describe all the simplices that contain a given vertex in an amount of time that is bounded by a polynomial function of n and the logarithm of k, which we think of as fixed multiple of the number of bits required to specify a vertex in some encoding of the vertices.

We now consider the assignment of labels. In the applications of interest the Sperner labelling is the labelling ℓ_f derived from a continuous $f : D \to D$, so f itself is an input. How might it be encoded?

One approach is to assume that all our information about f comes from an oracle, or black box, that accepts any point $x \in D$ as an input and outputs $f(x)$. Possibly the oracle looks at a map, or performs some physical measurement, but really the main point is that the algorithm cannot be based on any other information about f. Hirsch et al. (1989) showed that the "black box complexity" of the problem is exponential: there is a constant c such that the worst case number of calls of any algorithm to the oracle is bounded below by constant times c^{np} where p is the number of digits of accuracy. Roughly speaking, once you commit to an algorithm, the Devil can concoct a problem that hides the completely labelled simplices in places you will not look until you have looked at exponentially many other simplices.

An alternative formulation is to suppose that f is given to us in the form of a Turing machine or Boolean circuit that computes the value of f at any vertex of the triangulation. In practical computation f will, of course, typically be encoded as a formula or a more complicated computational procedure. In this formulation of the problem, in addition to sampling f at various points, we are in principle allowed to "look under the hood" of the device that computes f. For many particular formulas that might define f this can be quite helpful, obviously, but it is hard to see how the internal structure of the device might be used in a fully general algorithm. Thus there is, in principle, a slight sliver of hope that this version

of the problem might be less intractable than the oracle version, but this seems extremely unlikely. In full generality, finding fixed points seems to be quite hard.

We now settle on a precise version of the problem. An instance of the computational problem n-SPERNER is, by definition, given by a Boolean circuit that passes from a bitstring encoding a vertex of some triangulation of D to a bitstring describing the simplices in the triangulation that contain the vertex and the labels of all of their vertices. Of course the desired output is a completely labelled simplex. In this formulation the "size" of the input is the size of the Boolean circuit. Of course the amount of time required to perform the computation specified by the circuit is bounded by a polynomial (in fact linear) function of the size of the circuit.

Insofar as we are interested in how the computational burden increases as we increase precision by reducing the mesh of the triangulation, at first sight this formulation of the problem seems a bit odd. It would seemingly be more natural to have the function be given by a Turing machine that computes the values of the function with arbitrary precision. That is, the input of the Turing machine is a vector of numbers, specified with a precision that we would think of as the number of significant digits, and a tolerance for error $\varepsilon > 0$, and the output is a vector of numbers that is guaranteed to be within ε of the true value of the function at the given input.

Once we fix a triangulation, a tolerance for error ε (typically on the order of the mesh of the triangulation) and the number of digits of the input and output, we can (as we described in the last section) pass from the Turing machine to a Boolean circuit that takes a vertex as input and computes the values of the function, and thus the labels, at the neighboring vertices of the input. That is, we can pass from the function, given in the form that seems natural, to an instance of n-SPERNER. Moreover, this maneuver puts the complexity of computing the function out of sight, which is appropriate because we wish to focus exclusively on the complexity arising from the difficulty of searching for a completely labelled simplex.

How can we go further in our understanding of the complexity of n-SPERNER? At this point we observe that while finding a completely labelled simplex may be difficult, verifying that a given simplex is completely labelled is quite easy. Specifically, a consequence of our formulation of the problem is that there is a polynomial time algorithm that does this.

For any problem in **NP**, and any system of certificates for the problem, there is a related search problem: given an instance of the problem, either produce a certificate confirming that the answer is Yes or determine that the answer to the problem is No. The class of such problems is **FNP**, where the **F** stands for "function."[9] More precisely, a problem in **FNP** is defined by a polynomially balanced relation $R \subset \Sigma^* \times \Sigma^*$ for which there is a Turing machine that can determine whether a pair (x, y) is in R in polynomial time. The associated problem in **FNP** for input x is to either find a $y \in \Sigma^*$ such that xRy or to determine that there is no such y.

[9]Of course this is a bit misleading because there can be many acceptable certificates. For economists and game theorists, at least, it would be more natural to denote this class by **CFP**, where the **C** stands for "correspondence."

In many cases it is easy to see that the complexity of the decision problem is the same as the complexity of the associated function problem. Consider, for example, the following method for passing from an algorithm for CLIQUE$_{n,k}$ to an algorithm for constructing a k-clique when one exists. First we use the decision problem algorithm to determine whether the graph contains a clique of size k. If it does, then for any node v we can use the decision problem algorithm to ask whether there is a clique of size $k-1$ contained in $\mathcal{N}(v)$. If the answer is yes, our problem is reduced to looking for such a clique, and if the answer is no we can eliminate v from the graph. Proceeding through the vertices one by one will eventually produce the desired clique. In particular, if the decision problem runs in polynomial time, then so does the derived function problem algorithm.

Somewhat surprisingly, a problem in **NP** can be trivial, because the answer is always Yes, and at the same time the associated problem in **FNP** is quite interesting. "Does a given integer have a prime factorization?" "In a parity game, does one of the two players have a stationary pure strategy that forces a win?" "Does a two person normal form game with rational payoffs have a Nash equilibrium in mixed strategies?" In each case the answer is always yes, but finding the factorization/stationary pure strategy/Nash equilibrium is still a challenging computational problem for which no polynomial time algorithm is known. The class of problems in **FNP** that are associated with decision problems in **NP** that are "trivial," because the answer is always affirmative, is denoted by **TFNP**, where the **T** stands for "total." Evidently Sperner's lemma implies that SPERNER is a member of **TFNP**.

The class **TFNP** was introduced by Megiddo and Papadimitriou (1991). Earlier Megiddo (1988) had observed that any problem in **NP** $\cap$ **coNP** gives rise to a problem in **TFNP** (find a certificate for one answer or the other) and conversely, if a problem in **NP** has a reduction to a problem in **TFNP**, then it is in **NP** $\cap$ **coNP** because the reduction provides a system of certificates for either answer. Since computer scientists believe that **NP** $\cap$ **coNP** is almost certainly a proper subset of **NP**, it is very unlikely that an **NP**-complete problem has a reduction to a problem in **TFNP**.

It turns out that **TFNP** is quite large and diverse, so the fact that a problem is a member does not in itself do much to help us understand the problem. We need to find some way to narrow the focus. The next section describes a specific algorithm for n-SPERNER, and in the section after that the structure of this algorithm provides a concrete example and motivation for the definitions of more refined concepts.

10 The Scarf Algorithm

In this section we provide a brief description of an algorithm developed by Scarf (1967, 1973) for n-SPERNER. For each $i = 0, \ldots, n$ let $\mathbf{e}_i$ be the element of $\mathbb{R}^{n+1}$ whose i^{th} component is one and whose other components are all zero. Then D is the convex hull of $\mathbf{e}_0, \ldots, \mathbf{e}_n$. Fix an instance of n-Sperner, let $\mathcal{K}$ be the associated simplicial subdivision of D, and let V be the set of vertices of this subdivision. Then the Boolean circuit that encodes the instance of n-SPERNER passes from an encoding of a $v \in V$ to a description of the simplices that contain v and the labels of the vertices of these simplices. Let

$\ell : V \to \{0, \ldots, n\}$ be the Sperner labelling given by this circuit.

For each i let D^i be the convex hull of $\mathbf{e}_0, \ldots, \mathbf{e}_i$, let $\mathcal{K}^i$ be the set of elements of $\mathcal{K}$ that are contained in D^i (of course $\mathcal{K}^i$ is a simplicial subdivision of D^i) let V^i be the set of vertices in $\mathcal{K}^i$, and let ℓ_i be the restriction of ℓ to V^i. For $v \in V^i$ we have $v_{\ell_i(v)} = v_{\ell(v)} > 0$, so $\ell_i(v) \in \{0, \ldots, i\}$, and ℓ_i is therefore a Sperner labelling for $\mathcal{K}^i$. We say that an i-simplex in $\mathcal{K}^i$ is **completely labelled** if its vertices have all the labels $0, \ldots, i$, and it is **almost completely labelled** if its vertices have all the labels $0, \ldots, i-1$. Note that a completely labelled simplex is almost completely labelled.

We define a symmetric adjacency relation on the set of all almost completely labelled simplices, of all dimensions, as follows. Two almost completely labelled i-simplices in $\mathcal{K}^i$ are **adjacent** if their intersection is a facet of both simplices whose vertices have all the labels $0, \ldots, i-1$. A completely labelled $(i-1)$-simplex in $\mathcal{K}^{i-1}$ and an almost completely labelled i-simplex in $\mathcal{K}^i$ are **adjacent** if the first simplex is a facet of the second. For example, $\mathbf{e}_0$, understood as the unique element of the simplicial subdivision of $\mathcal{K}^0$, is completely labelled, and when $n > 0$ it is an endpoint of a unique 1-simplex in $\mathcal{K}^1$, which is necessarily almost completely labelled.

Consider an almost completely labelled i-simplex that is not completely labelled. We claim that the given simplex is adjacent to exactly two other almost completely labelled simplices. First observe that it is not adjacent to an almost completely labelled $(i+1)$-simplex because it is not completely labelled. It has $i+1$ vertices which have the labels $0, \ldots, i-1$, so there is one label that occurs twice, and for each of the vertices with this label, the opposite facet is an $(i-1)$-simplex with the labels $0, \ldots, i-1$. Any simplex adjacent to the given simplex must contain one of these facets. If one of these two facets is contained in the boundary of D^i, it must be contained in D^{i-1}, because the labelling is Sperner. Therefore each of the two facets is either

- a facet of another i-simplex in $\mathcal{K}^i$, which is necessarily almost completely labelled and consequently adjacent to the given simplex, or
- contained in D^{i-1} and consequently a completely labelled $(i-1)$-simplex that is adjacent to the given simplex.

It cannot be the case that both facets are contained in D^{i-1} because then their convex hull, namely the given i-simplex, would also be contained in D^{i-1}. If the two facets are faces of two other i-simplices, these two simplices cannot be the same because if they were each of them would contain the given simplex, which is obviously impossible.

Now consider a completely labelled i-simplex. We claim that if $0 < i < n$, then the given simplex is adjacent to exactly two other almost completely labelled simplices. It is a facet of exactly one $(i+1)$-simplex in $\mathcal{K}^{i+1}$, which is necessarily almost completely labelled, so that the two simplices are adjacent. The given simplex has one vertex with the label i, and the opposite facet is an $(i-1)$-simplex in $\mathcal{K}^i$ with the labels $0, \ldots, i-1$. Any i-simplex or $(i-1)$-simplex adjacent to the given one must contain this facet. If the facet is contained in in the boundary of D, then, as above, it must be a contained in $\mathcal{K}^{i-1}$

and it is consequently a completely labelled $(i-1)$-simplex, so it and the given simplex are adjacent. Otherwise the facet is a facet of one other i-simplex in $\mathcal{K}^i$, which is almost completely labelled, hence adjacent to the given simplex.

The completely labelled 0-simplex is adjacent to an almost completely labelled 1-simplex (provided that $n > 0$) but it does not have any facets. A completely labelled n-simplex is not a face of an almost completely labelled $(n+1)$-simplex, but (provided that $n > 0$) it has one facet whose vertices have the labels $0, \ldots, n-1$, and it is either adjacent to that facet or to the other n-simplex that shares that facet. Combining the various cases considered above, we conclude that a completely labelled i-simplex is adjacent to $2 - \delta_{i0} - \delta_{in}$ almost completely labelled simplices, where δ_{ij} is the **Kronecker delta**: if $i = j$, then $\delta_{ij} = 1$, and otherwise $\delta_{ij} = 0$.

Consider the graph whose vertices are the almost completely labelled simplices, and whose edges are given by the adjacency relation. If $n = 0$, then the graph consists of a single isolated point. Suppose that $n > 0$. Then every vertex has either one of two neighbors, so the graph is a union of loops and paths. The vertices with one neighbor, which are the endpoints of the paths, are:

(a) $\mathbf{e}_0$, thought of as a completely labelled 0-simplex;

(b) the completely labelled n-simplices.

Since each path has two endpoints, there are an odd (hence nonzero, so Sperner's lemma is proved!) number of completely labelled n-simplices. In particular, the path beginning at $\mathbf{e}_0$ has a completely labelled n-simplex as its other endpoint. The **Scarf algorithm** is the process of following this path from $\mathbf{e}_0$ to this endpoint. Figure 1 shows a sample path of the Scarf algorithm.

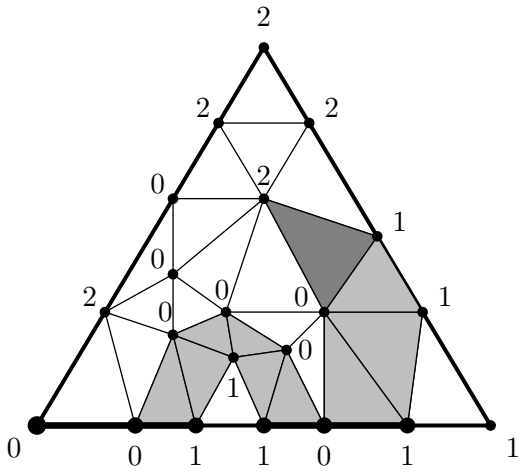

Figure 1

It turns out that the algorithm has an additional property that will be important. As one proceeds from the starting point to the final completely labelled simplex, it would certainly be natural to keep

track of where one had been, and to use this information to decide which direction to go. But it turns out that even if you have forgotten the direction you came from, you can use local information to figure out where to go next.

In order to describe this property we need to introduce the concept of orientation. A k-dimensional **oriented vector space** is a k-dimensional vector space W with a designated ordered basis $w_1, \ldots, w_k$. A second ordered basis $w'_1, \ldots, w'_k$ is **positively oriented** if the linear transformation taking each w_i to w'_i has a positive determinant, and otherwise $w'_1, \ldots, w'_k$ is **negatively oriented**. Concretely, if you look at a positively oriented basis in a mirror, its image is a negatively oriented basis, because there is a coordinate system in which the mirror image transformation is given by $\mathbf{e}_1 \mapsto -\mathbf{e}_1$ and $\mathbf{e}_2 \mapsto \mathbf{e}_2, \ldots, \mathbf{e}_k \mapsto \mathbf{e}_k$.

For each $i = 0, \ldots, n$ let W^i be the linear subspace of $\mathbb{R}^{n+1}$ that is parallel to the affine hull of D^i. We endow W^i with the orientation given by the ordered basis $\mathbf{e}_1 - \mathbf{e}_0, \ldots, \mathbf{e}_i - \mathbf{e}_0$. Suppose that we have a completely labelled simplex with vertices $v^0, \ldots, v^i$, where the indices are chosen so that $\ell(v^0) = 0, \ldots, \ell(v^i) = i$. This simplex is said to be **positively oriented** if $v^1 - v^0, \ldots, v^i - v^0$ is a positively oriented ordered basis of W^i, and otherwise it is **negatively oriented**.

Now suppose that we find ourselves at an almost completely labelled i-simplex along the path of the Scarf algorithm. The algorithm ends at a positively oriented completely labelled n-simplex, and the general idea is that the forward direction of the algorithm is the one such that the next simplex one encounters is positively oriented if it happens to be completely labelled. In order for this to work, this system of choosing the next almost completely labelled simplex has to satisfy three conditions:

(a) the first step on the path beginning at $\mathbf{e}_0$ satisfies this condition;

(b) at any almost completely labelled simplex that is adjacent to two other almost exactly labelled simplices, exactly one of the two possible steps satisfies this condition;

(c) after one takes one step satisfying this condition, the next step continues in the same direction instead to returning to the starting point.

A detailed verification would be tedious reading, but with a bit of patience, an interested reader should be able to check that these conditions hold. Figure 1 contains concrete instances of each of the situations that need to be checked.

Another important computational procedure to bear in mind is the pivoting procedure of Lemke (1965) for linear complementarity problems. (The linear complementarity problem in defined in Section 15.) This is guaranteed to find a solution under various conditions, including the less general situation in which the linear complementary problem arises from 2-NASH, which is the problem of finding a Nash equilibrium of a two person normal form game. The specialization of Lemke's procedure to this setting is the Lemke-Howson algorithm (Lemke and Howson (1964)). A Nash equilibrium is a pair of mixed strategies such that each pure strategy is either optimal, given the mixed strategy of the opponent, or is assigned no probability. The set of mixed strategy profiles satisfying this condition

for all but one designated pure strategy is (for generic payoffs) a one dimensional simplicial complex. For generic payoffs the Lemke-Howson algorithm follows the path in this complex that begins at an easily computed starting point, namely the designated pure strategy and its pure best response, to its other endpoint, which is necessarily a Nash equilibrium. For nongeneric payoffs the algorithm follows the path that would be followed for the generic payoffs resulting from an infinitesimal perturbation. Todd (1976) established an orientation for Lemke's algorithm.

11 PPAD

So far we have seen two ways to study the complexity of a computational problem. The most direct way to show that a computational problem is "easy" is to provide a provably efficient algorithm for it. To show that a problem is hard we reduce from another problem that is, in some sense, known to be hard. The way to do this for many problems in an organized manner is to define a class of problems that has a complete problem, and then use reductions to show that various other problems are complete for the class. This program was a smashing success in its application to **NP**, and we would like to apply it to the computation of fixed points.

Factoring a large integer and finding an approximate fixed point certainly seem like very different computational problems, so it would be quite remarkable if there was a "natural" problem that each of these problems could be reduced to. This observation suggests that it is unlikely that **TFNP** has any complete problems. There is also a more specific line of reasoning pointing in this direction. The information that defines a problem in **NP** is a Turing machine that can verify a certificate for an instance of the problem in polynomial time. In the proof of the Cook-Levin theorem, this Turing machine is part of the raw material of the construction. In general, if there is a complete problem, there must be a reduction to it from the data defining the class, so this data must have some useful form. The information that defines a problem in **TFNP** is a Turing machine for a problem in **NP**, with the additional proviso that for *every* problem instance there is at least one confirming certificate. There can be exponentially many problem instances satisfying any given bound on the size of the input, so it is hard to see how this last piece of information might be used in the construction of a polynomial time reduction. More generally, Papadimitriou (1994b) defines a **semantic class** to be a subclass of **TFNP** in which some syntactically given object, e.g., a Turing machine, defines a problem in the class if and only if it satisfies some property quantified over all inputs. The line of reasoning given above suggests that semantic classes will usually not have complete problems.

Papadimitriou (1994b) defined certain computational classes that are contained in **TFNP** and are large enough to contain the version of the fixed point problem in which the function is given by a Turing machine or Boolean circuit. He observed that for every known member of **TFNP**, there is a proof of membership, and these proofs potentially provide some structure that might be useful in the construction of a reduction. For example, every graph has an even number of vertices of odd degree.

Based on this, one can define a class of search problems, where such a problem is given by a Turing machine that generates local information about the structure of such a graph, and one vertex of odd degree is given, with the problem being to find a second such vertex. A more specific principle is that a graph whose vertices have maximum degree two has an even number of leaves. A still more specific principle is that in a directed graph in which the indegree and the outdegree of all vertices are either zero or one, if there is a source, then there is another vertex that is either a sink or a different source.

PPAD is an acronym for "polynomial parity argument, directed." It is a computational class defined in terms of a computational problem called END OF THE LINE. An instance of END OF THE LINE is given by two Boolean circuits S and P (for "successor" and "predecessor") each of which have m input bits and m output bits. These circuits define a directed graph in which the maximum indegree and the maximum outdegree are both one, so each vertex has a most one direct predecessor and at most one direct successor. Formally $G_{S,P} = (V_{S,P}, A_{S,P})$ where

$$V_{S,P} = \{\, v \in \{0,1\}^m : \text{either } S(v) \neq v \text{ and } P(S(v)) = v, \text{ or } P(v) \neq v \text{ and } S(P(v)) = v \,\}$$

and

$$A_{S,P} = \{\, (v,w) \in V_{S,P} \times V_{S,P} : v \neq w,\, S(v) = w,\, P(w) = v \,\}.$$

(Here we are simplifying the notation of Section 8 by writing S and P in place of f^S and f^P.) It is required that $P(0^m) = 0^m \neq S(0^m)$ and $P(S(0^m)) = 0^m$, where $0^m = (0, \ldots, 0) \in \{0,1\}^m$, so that 0^m is a source. The problem is to find either a sink or a different source.

PPAD is the class of all problems in **TFNP** that have polynomial time reductions to instances of END OF THE LINE. That is, END OF THE LINE is, by definition, a complete problem for **PPAD**. For example, although it would be quite difficult to describe in detail, it should be evident that one can pass (algorithmically, in polynomial time) from a Boolean circuit defining an instance of n-SPERNER to a pair of Boolean circuits that each take a simplex in the induced triangulation of D as input, deciding whether it is almost completely labelled, and, if so, what its predecessor and successor simplices are.

Although we will focus on **PPAD**, the reader should have some sense of why this class is more useful than possible alternatives. Of course we need a class that is large enough to encompass SPERNER, but among the various candidates satisfying this condition, we are likely to get the most out of those classes that are smallest, because the defining problem of such a class will be easier to reduce. More concretely, the additional structural information included in the definition provides additional raw material for a reduction. In addition, the definition of a solution should be as permissive as is allowed by the computational problems that motivate our definition. For example, OTHER END OF THE LINE is the computational problem with the same given data as END OF THE LINE in which the goal is to find the other end of the path leading away from 0^m. It turns out (Papadimitriou (1994a), Goldberg et al. (2011)) that OTHER END OF THE LINE is complete for **FPSPACE**, which is the class of search problems that can be solved by algorithms whose space requirement is bounded by a polynomial function of the size of the input. Note that **FNP** $\subset$ **FPSPACE** because, for any system of certificates for

a problem in **NP**, trying each of the possible certificates in turn requires polynomially bounded space. As usual, we expect that this inclusion is strict, but do not know how to prove it. In particular, OTHER END OF THE LINE seems unlikely to be in **TFNP** because there is no obvious way to quickly verify that a solution is correct, i.e., that it is actually the other end of the path commencing at 0^m.

Yannakakis (2009) surveyed **TFNP**, **PPAD** and other subclasses of **TFNP** that are related to computation of equilibria and fixed points. Another class of interest to economists is **PLS**, which is an acronym for "polynomial local search." A problem in **PLS** consists of an exponentially large search space and two polynomial time algorithms that take elements of this space as inputs. The first algorithm computes a real valued cost. For an element of the search space there is a "neighborhood," which is a (possibly exponentially large) set of points in the search space, and the second algorithm either returns a point in the neighborhood with lower cost or verifies that the given point is a local optimum, which is to say that its cost is not greater than the cost of any other point in the neighborhood. The problem is to find a local optimum. (Of course the problem of finding a global optimum may also be interesting.) Potential games (Rosenthal (1973), Monderer and Shapley (1996)) are an economically important class of games which always have pure Nash equilibria because a local maximum of the potential function is an equilibrium if we define the neighborhood of a pure strategy profile to be the set of profiles that can be reached by a deviation by a single agent. Congestion games are a particular type of potential game modelling traffic delays on networks such as highway systems or the internet. Fabrikant et al. (2004) showed that computing a pure Nash equilibrium of a congestion game is a **PLS**-complete problem.

Recall that in our discussion of the black box complexity of finding a completely labelled simplex, we allowed the Devil to choose a problem after the user had committed to an algorithm. One might imagine that an algorithm using randomization might do better, on average, because the Devil must commit to a problem before the user starts rolling the dice. This possibility was established for **PLS** by Aldous (1983). However, Chen and Teng (2007) showed that randomization cannot improve the black box complexity of finding an approximate fixed point by more than a little bit. A consequence of this is a precise sense in which local optimization in the context of **PLS** is easier than finding an approximate fixed point, which is in turn easier than global optimization in the context of **PLS**.

12 A Collection of Problems

In this section we describe a number of problems that, together with END OF THE LINE, constitute the cast of characters in our work in the next section.

12.1 n-SPERNER

Recall that an instance of n-SPERNER is given by a Boolean circuit whose input is a bitstring encoding a vertex of some triangulation of D. The output of the circuit describes all the simplices in the triangulation that contain the vertex and the labels of all of their vertices. The desired output is a completely

labelled simplex.

12.2 n-FIXED POINT

An instance of n-FIXED POINT is given by a number $\varepsilon > 0$ and a Boolean circuit that takes a bit string of a certain length as an input and outputs a bit string of a certain (possibly different) length. There is a given interpretation of these bitstrings as points in D, which could be implemented by a Boolean circuit that passes from these bitstrings to $(n+1)$-tuples of numbers, and which must in any event be computable, so that the circuit is understood to compute a function $f : D \to D$. The desired output is an $(n+1)$-tuple $v^0, \ldots, v^n$ of points in D such that $\|v^j - v^i\| < \varepsilon$ for all i and j and $v^i_i \geq f_i(v^i) - \varepsilon$ for all i. We require that D is covered by the open ε^2-balls around the points represented by these strings.

It is neither necessary to assume that f is continuous, nor really possible, given that the set of allowed inputs and outputs is a finite set. Of course the user's interpretation of the output will typically depend on such an assumption.

It is useful to generalize n-FIXED POINT. Suppose $E \subset D$ is nonempty and compact, and there is a circuit that computes a discretized version of a retraction $\rho : D \to E$. (The function ρ is a **retraction** if it is continuous and $\rho(x) = x$ for all $x \in E$.) If we are also given a circuit that computes a function $f : E \to E$, then we can apply n-FIXED POINT to the function

$$f \circ \rho : D \to D,$$

interpreting the output as an approximate fixed point of f. Even more generally, instead of assuming that E is a subset of D, we may assume that there is an embedding $\iota : E \to D$ and that $\rho : D \to \iota(E)$ is a retraction. It makes sense to regard the problem definition of FIXED POINT as encompassing this extension, and we will do so.

12.3 NASH

Let $(S_1, \ldots, S_r; u_1, \ldots, u_r)$ be an r-player game. That is, each S_p is a finite set of **pure strategies**, and each u_p is a real valued function whose domain is the set $S = S_1 \times \cdots \times S_r$ of **pure strategy profiles**. For each p let Δ^p be the set of **mixed strategies** for p, which is to say the set of probability measures on S_p. A typical element of Δ^p is denoted by σ_p with components σ_{jp} for $j \in S_p$. The space of **mixed strategy profiles** is $\Delta = \Delta^1 \times \cdots \times \Delta^r$ with typical element $\sigma = (\sigma_1, \ldots, \sigma_r)$. We extend each u_p to Δ multilinearly:

$$u_p(\sigma) = \sum_{s \in S} \Big(\prod_{i=1}^{r} \sigma_{s_p p} \Big) u_p(s).$$

For $\sigma \in \Delta$, player p, and pure strategy $j \in S_p$, let $u_{jp}(\sigma) = u_p(j, \sigma_{-p})$ where (j, σ_{-p}) denotes the element of Δ obtained from σ by replacing σ_p with the mixed strategy that assigns all probability to j. A mixed strategy profile σ^* is a (mixed) **Nash equilibrium** if $u_p(\sigma^*) \geq u_{jp}(\sigma^*)$ for all players p and all $j \in S_p$. That is, there is no player who can improve her expected payoff by deviating unilaterally.

For our purposes it turns out that approximate equilibrium is the more useful notion. Following Daskalakis et al. (2006a), we distinguish between two versions of this concept. We say that σ^* is an ε-**approximate Nash equilibrium** if

$$u_p(\sigma^*) \geq -\varepsilon + \max_{j \in S_p} u_{jp}(\sigma^*)$$

for each p. We say that σ^* is an ε-**Nash equilibrium** if, for each p and $j, j' \in S_p$, either $u_{jp}(\sigma^*) \geq -\varepsilon + u_{j'p}(\sigma^*)$ or $\sigma^*_{jp} = 0$. That is, σ^* is an ε-approximate Nash equilibrium if each agent's "average" choice of pure strategy does not forego more than ε utility, and it is a ε-Nash equilibrium if no pure strategy that is assigned positive probability foregoes more than ε utility. Thus ε-Nash equilibrium is the more demanding concept. Lemma 1 of Daskalakis et al. (2009a) gives a polynomial time procedure for passing from an ε-approximate Nash equilibrium to a $C\sqrt{\varepsilon}$-Nash equilibrium when ε is sufficiently small, where C is a constant that depends on the number of players and the range within which the utilities vary. (The idea is to shift all probability away from pure strategies that are inferior by more than a carefully chosen multiple of $\sqrt{\varepsilon}$.) For this reason the two concepts have the same effective computational complexity.

We define r-NASH to be the problem of computing an ε-Nash equilibrium of a given r-player game with rational payoffs, for a given ε. When $r = 2$ and all the payoffs are rationals, the Lemke-Howson computes an exact Nash equilibrium whose mixture probabilities are rationals. (Lipton and Markakis (2004) showed that computation of an exact equilibrium with algebraic mixture probabilities is possible when $r > 2$ and the payoffs are rational, but as explained in Etessami and Yannakis (2010), the level of complexity is much greater.) Therefore the the problem of finding such a Nash equilibrium is in **PPAD**, and is intuitively more natural than 2-NASH. Nonetheless, one of the reasons 2-NASH is more useful is precisely because it is easier: that finding an exact equilibrium of a two player game with rational payoffs is a **PPAD**-complete problem will follow once we have shown that 2-NASH is **PPAD**-complete.

12.4 2D BROUWER

The triangulations of D that appear in Sperner's lemma and the Scarf algorithm are difficult to use for constructive purposes. Papadimitriou (1994a) introduced what is, in effect, a version of Sperner's lemma for the cubical lattice, that is much easier to work with. In this subsection we describe the two dimensional version of this combinatoric computational problem, which is called 2D BROUWER.

Let $C = [0, 1]^2$ be the unit square. Fix an integer $N \geq 2$, and let

$$Q = \{0, \ldots, N-1\}^2.$$

Typical elements of this set are $i = (i_x, i_y)$, $j = (j_x, j_y)$, etc. For each $i \in Q$ there is an associated **squarelet**

$$C_i = \left[\tfrac{1}{N} i_x, \tfrac{1}{N}(i_x + 1)\right] \times \left[\tfrac{1}{N} i_y, \tfrac{1}{N}(i_y + 1)\right].$$

Of course these squarelets and their faces constitute an polyhedral decomposition of C. The vertices of this decomposition are the points $\frac{1}{N}i = (\frac{1}{N}i_x, \frac{1}{N}i_y)$ where i_x, i_y are integers in the range $0, \ldots, N$. The **interior vertices** are the $\frac{1}{N}i$ for $i \in Q^*$ where

$$Q^* = \{1, \ldots, N-1\}^2.$$

Fix a number α with $0 < \alpha < \frac{3}{4N}$. There are three **colors**:

$$\chi_0 = (-\alpha, -\alpha), \;\; \chi_x = (\alpha, 0), \;\; \chi_y = (0, \alpha).$$

We will work with an assignment of a color $\chi^i \in \{\chi_0, \chi_x, \chi_y\}$ to each $i \in Q$ that we think of as a coloring of the squarelet C_i. Such an assignment is **legal** if its assignment of colors to the "boundary" elements of Q satisfies the following restrictions:

- if $i_x = 0$, then $\chi^i = \chi_x$;
- if $i_x > 0$ and $i_y = 0$, then $\chi^i = \chi_y$;
- if $i_x, i_y > 0$ and $\max\{i_x, i_y\} = N - 1$, then $\chi^i = \chi_0$.

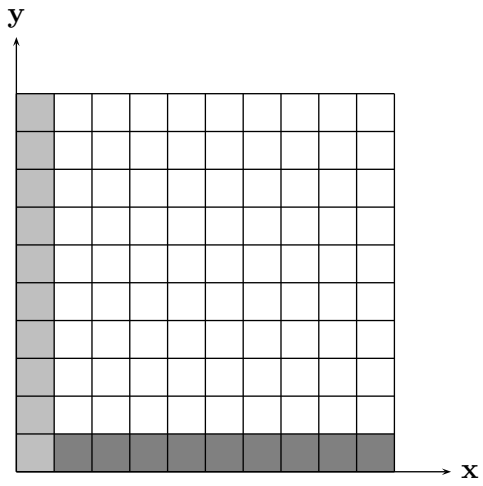

Figure 2

Figure 2 shows the region near $(0, 0, 0)$ when $\chi^i = \chi_0$ for all i such that C_i is in the interior of C. As can be seen, in our illustrations χ_0 is represented by white, while χ_x and χ_y are represented light gray and gray respectively.

We say that $j \in Q$ **touches** $i \in Q^*$ if the squarelet C_j has $\frac{1}{N}i$ as a vertex, which is to say that

$$j_x \in \{i_x - 1, i_x\} \quad \text{and} \quad j_y \in \{i_y - 1, i_y\}.$$

An $i \in Q^*$ is **panchromatic**, and $\frac{1}{N}i$ is a **panchromatic vertex**, if

$$\{\chi^j : j \text{ touches } i\} = \{\chi_0, \chi_x, \chi_y\}.$$

That is, a panchromatic i is one such that all three colors occur among the squarelets that have $\frac{1}{N}i$ as a vertex. The reader should note that $\frac{1}{N}(1,1)$ is the unique panchromatic vertex in Figure 2.

The problem 2D BROUWER is the following: given a Boolean circuit that takes $i \in Q$ as input and outputs a legal χ^i, find a panchromatic $i \in Q^*$.

In the next section we will use Brouwer's fixed point theorem to prove that a panchromatic i always exists, so that 2D BROUWER is in **TFNP**. Conversely, a continuous function $f : C \to C$ induces an assignment that is defined, for those i that are not constrained by legality, by setting $\chi^i_f = \chi_x$ if $f_x(\frac{1}{N}i) \geq \frac{1}{N}i_x$, setting $\chi^i_f = \chi^y$ if $f_x(\frac{1}{N}i) < \frac{1}{N}i_x$ and $f_y(\frac{1}{N}i) \geq \frac{1}{N}i_y$, and otherwise setting $\chi^i_f = \chi_0$. In the same way that Sperner's lemma was used to prove Brouwer's fixed point theorem, we can derived the two dimensional case of this result by taking a limit point of a sequence of panchromatic vertices for the assignments induced by f for various N.

12.5 GRAPHICAL GADGET GAME

The computational problem described in this subsection is based on a simple, elegant, and powerful idea: in a two player game in which each player has two pure strategies, the relationship between the payoffs and the ε-Nash equilibria can be used to compute elementary logical and arithmetic operations. We first describe a collection of **gadgets**, which are 2×2 games of this sort. We then describe how to hook these up in a large network using the notion of a graphical game.

Our first three gadgets perform basic logical operations.

$G_\vee$	0	1
0	$(1, \frac{1}{2})$	$(0, P+Q)$
1	$(0, \frac{1}{2})$	$(1, P+Q)$

Input: $P, Q \in \{0,1\}$
Output: 1 if $P \vee Q$ and 0 otherwise

$G_\wedge$	0	1
0	$(1, \frac{3}{2})$	$(0, P+Q)$
1	$(0, \frac{3}{2})$	$(1, P+Q)$

Input: $P, Q \in \{0,1\}$
Output: 1 if $P \wedge Q$ and 0 otherwise

$G_\neg$	0	1
0	$(0, 1-P)$	$(1, P)$
1	$(1, 1-P)$	$(0, P)$

Input: $P \in \{0,1\}$
Output: $\neg P$

In each case here and below the inputs are numbers in $[0,1]$, coming from outside this particular interaction, that affect only the column player's payoffs, and the output is the ε-Nash equilibrium probability that the row player plays 1. Restricting the inputs to lie in a subset of $[0,1]$, as we have done here, is interpreted as meaning that the gadget is not guaranteed to perform "as advertised" when the restriction

does not hold. In each of these three games the column player has a strictly dominant strategy when the restriction is satisfied, to which the row player has a unique best response. It follows that for sufficiently small ε there is a unique ε-Nash equilibrium that gives the indicated output.

In addition to logical operations, it will also be necessary to do some arithmetic. The next three gadgets compute sums, differences, and multiplication by a fixed number α.

G_+	0	1
0	$(0, p+q)$	$(1, 0)$
1	$(1, p+q)$	$(0, 1)$

Input: $p, q \in [0,1]$
Output: $r \in [\min\{p+q-\varepsilon, 1\}, \min\{p+q+\varepsilon, 1\}]$

G_-	0	1
0	$(0, p-q)$	$(1, 0)$
1	$(1, p-q)$	$(0, 1)$

Input: $p, q \in [0,1]$
Output: $r \in [\max\{p-q-\varepsilon, 0\}, \max\{p-q+\varepsilon, 0\}]$

$G_{\times\alpha}$	0	1
0	$(0, \alpha p)$	$(1, 0)$
1	$(1, \alpha p)$	$(0, 1)$

Input: $p \in [0,1]$
Output: $r \in [\min\{\alpha p-\varepsilon, 1\}, \min\{\alpha p+\varepsilon, 1\}]$

These games, and the three more below, are all like matching pennies insofar as an ε-equilibrium strategy for the row player is one that makes the column player indifferent (up to ε) between her two pure strategies. In contrast to the three logic gadgets above, which gave exact outputs if the inputs conformed to expectations, all the arithmetical gadgets produce outputs with bounded errors, and of course the analysis needs to control the accumulation of error. Also, it should be noted that, in the interest of conveying the main ideas, we have failed to take into account certain possibilities such as $p+q-\varepsilon < 0$ in the definition of G_+, and of course a fully rigorous analysis would also need to control these sources of error.

The next gadget is akin to a command in a computer language that initializes a variable to a given value.

G_α	0	1
0	$(0, \alpha)$	$(1, 0)$
1	$(1, \alpha)$	$(0, 1)$

Input: none
Output: $r \in [\alpha-\varepsilon, \alpha+\varepsilon]$

The following gadget requires that in ε-Nash equilibrium the value of one variable is within ε of being equal to the value of another. Its implementation is just the special case of $G_{\times\alpha}$ given by setting $\alpha = 1$.

$G_=$	0	1
0	$(0, p)$	$(1, 0)$
1	$(1, p)$	$(0, 1)$

Input: $p \in [0,1]$
Output: $r \in [p-\varepsilon, p+\varepsilon]$

This can be used to make approximate equality of two quantities (e.g., a point in the domain and its image under a function) a condition of equilibrium. We will also think of it as akin to an assignment operator that sets the value of one variable equal to the value of another.

Finally there is a gadget that decides which of two numbers is bigger.

$G_<$	0	1
0	$(0,p)$	$(1,q)$
1	$(1,p)$	$(0,q)$

Input: $p, q \in [0,1]$
Output: 0 if $p < q - \varepsilon$ and 1 if $p > q + \varepsilon$

This gadget gives an indeterminate output when $|p - q| \leq \varepsilon$. Dealing with this particular ambiguity will be an important consideration in the design of the reduction from 2D BROUWER to GRAPHICAL GADGET GAME.

In thinking about how these gadgets might be hooked up to perform complex calculations, the most intuitive approach is to use a **graphical game**. When it was introduced by Kearns et al. (2001) this concept was defined using a normal form game and an undirected graph whose vertex set is the set of players. An edge between two players indicates that they affect each other, and the normal form is required to satisfy the requirement that each agent's payoffs do not depend on the pure strategy choices of unrelated players.

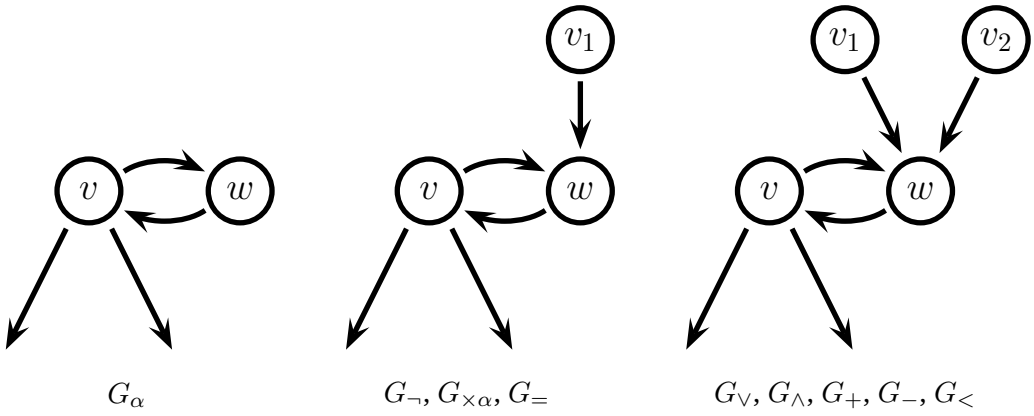

Figure 3

Goldberg and Papadimitriou (2006) and Schoenebeck and Vadhan (2006) introduced a more general model in which the graph is directed[10]. Now an arrow from one player to another indicates that the tail's behavior affects the head's payoff, but this relationship need not be symmetric.

Although the initial motivation for these concepts arose from modelling issues in computer science, it is evident that they are also very natural from the point of view of social science. In addition, if the game is "sparse," in the sense that the maximum degree of the graph is small, then these models becomes potentially attractive from the point of view of the computational complexity of computing

[10] Action-graph games, due to Bhat and Leyton-Brown (2004), are more general still. Computation and complexity of Nash equilibria for such games has been studied by Jiang and Leyton-Brown (2006) and Daskalakis et al. (2009c).

an equilibrium. Kearns et al. (2001), Littman et al. (2001), Elkind et al. (2006), and Daskalakis and Papadimitriou (2009) study issues of this sort.

In a different sense the class of models we consider is more specialized: we will only consider graphical games that are **binary** in the sense that each agent has exactly two pure strategies, denoted by 0 and 1. Some players are said to be **arithmetic**. The practical meaning of this is that the probability an arithmetic player's mixed strategy assigns to 1 is one of the variables in our computation. Other players are said to be **internal**, because they serve only to regulate the incentives of the arithmetic players.

The relevant subgraphs for the various gadgets are shown in Figure 3. Whenever one of the gadgets above is embedded in the graphical game there is a node v for the row player, who is arithmetic, and a node w for the column player, who is internal. These two players affect each other. In addition, w is affected by zero, one, or two players, who are the source of the inputs, and the output is transmitted though v's effect on up to two players, as shown. (We can restrict the nodes for arithmetic players to have outdegree no greater than two because $G_=$ can be used to reproduce a variable. This results in some greater accumulation of round off error, but for our purposes this is easily handled by choosing a smaller ε, so this restriction is in effect without loss of generality.)

An instance of the computational problem GRAPHICAL GADGET GAME is an $\varepsilon > 0$ and a graphical game with the structure described above: the nodes in the graph are grouped in pairs consisting of an arithmetic node and an internal node, the internal node is affected by up to two other arithmetic nodes, the arithmetic node affects up to two other internal nodes, and the interaction within the pair is one of the gadgets described above. The desired output is an ε-Nash equilibrium of this game.

13 A Circle of Reductions

The large scale logic of the project can be summarized using two sequences of reductions. The first is

$$\text{NASH} \longrightarrow \text{FIXED POINT} \longrightarrow \text{SPERNER} \longrightarrow \text{END OF THE LINE}.$$

The Scarf algorithm gives a reduction from SPERNER to END OF THE LINE, and below we will describe a reduction from FIXED POINT to SPERNER that is patterned on the argument passing from Sperner's lemma to the Brouwer fixed point theorem. We also give a simple reduction from NASH to FIXED POINT. The first of these reductions was the motivation for the introduction of the class **PPAD** by Papadimitriou (1994a), the second has been known since Sperner (1928), and the third has been known since Nash (1951). In addition, we give a direct reduction from 2D BROUWER to FIXED POINT that is not required by the large scale logic of the argument, but which illuminates how 2D BROUWER (and its higher dimensional analogues) are, in effect, variants of SPERNER.

The second sequence of reductions is

$$\text{END OF THE LINE} \longrightarrow \text{2D BROUWER} \longrightarrow \text{GRAPHICAL GADGET GAME} \longrightarrow \text{2-NASH}.$$

For the most part these reductions were developed during a flurry of activity in 2005 and 2006. Each of these is described rather informally in a subsection below. Collectively, the two sequences imply that each of the problems is both in **PPAD** and **PPAD**-hard, hence **PPAD**-complete.

The most striking (relative to the state of knowledge prior to 2005) implication of the second sequence of reductions, that 2-Nash is **PPAD** complete, can be strengthened in certain ways by restricting the games under consideration. Chen et al. (2006b) say that a bimatrix game is **sparse** if neither payoff matrix has a row or column with more than ten nonzero entries, and they show that the restriction of 2-NASH to sparse games in **PPAD**-complete. Abbott et al. (2005) gave a reduction from 2-NASH to its restriction to **win-lose games**, which are (not necessarily zero sum) bimatrix games with all payoffs in $\{0, 1\}$, so 2-NASH for these games is also **PPAD**-complete. These results are technically useful insofar as it is sometimes possible to reduce from a smaller class of games to a problem of interest, as in Chen et al. (2009a).

13.1 From FIXED POINT to SPERNER

We can reduce from n-FIXED POINT to n-SPERNER by choosing a triangulation of D with mesh less than ε, and a method of associating a nearby point represented by a bitstring with each vertex of the triangulation, then taking ℓ_f to be the labelling of the vertices induced by f applied to the associated points.

13.2 From NASH to FIXED POINT

Let $(S_1, \ldots, S_r; u_1, \ldots, u_r)$ be a given r-player game. The **Nash function** $f : \Delta \to \Delta$ is given by the formula

$$f_{jp}(\sigma) = \frac{\sigma_{jp} + \max\{u_{jp}(\sigma) - u_p(\sigma), 0\}}{1 + \sum_{j' \in S_p} \max\{u_{j'p}(\sigma) - u_p(\sigma), 0\}}.$$

It is well known and straightforward to verify that the Nash equilibria of the game are the fixed points of the Nash function. Although the formulas are a bit messy, it is not hard to show that the Nash function is Lipschitz for an easily computed Lipschitz constant (cf. Daskalakis et al. (2009a)) so any of the points returned when FIXED POINT is applied to the Nash function and $\varepsilon > 0$ is a δ-Nash equilibrium for an easily computed δ that can be made arbitrarily small by choosing a suitable ε. It is obvious that, in a variety of ways, Δ can be embedded in the standard n-simplex D for suitable n, with a retraction $\rho : D \to \Delta$, so, in view of our extension of FIXED POINT to more general domains, we have a rather crude reduction from NASH to FIXED POINT. (Much more elegant, from an algorithmic point of view, is the reduction of NASH to FIXED POINT given by Daskalakis et al. (2009a), which is based on the triangulation of Δ given by van der Laan and Talman (1982).)

13.3 From 2D BROUWER to FIXED POINT

We are given a probrem instance consisting of a Boolean circuit that takes $i \in Q$ as input and outputs a legal χ^i. Let $d : \mathbb{R}^2 \times \mathbb{R}^2 \to [0, \infty)$ be the metric

$$d(u, u') = \max\{|u_x - u'_x|, |u_y - u'_y|\}.$$

(That is, $d(u, u') = \|u - u'\|_\infty$.) For each $i \in Q$ let $\psi^i : \mathbb{R}^2 \to [0, 1]$ be the function

$$\psi^i(u) = \max\{0, 1 - 4N \min_{u' \in C_i} d(u, u')\}.$$

Of course ψ^i is continuous and identically one on C_i. Let $U_i = \{\, u \in \mathbb{R}^3 : \psi^i(u) > 0 \,\}$. Then

$$U_i = \left(\tfrac{1}{N}(i_x - \tfrac{1}{4}), \tfrac{1}{N}(i_x + \tfrac{5}{4})\right) \times \left(\tfrac{1}{N}(i_x - \tfrac{1}{4}), \tfrac{1}{N}(i_x + \tfrac{5}{4})\right)$$

is a slightly larger open square containing C_i. Let

$$\varphi^i = \frac{\psi_i}{\sum_{j \in Q} \psi_j} : C \to [0, 1].$$

Then $\{\varphi^i\}_{i \in Q}$ be a **partition of unity** for C subordinate to the open cover $\{U_i\}_{i \in Q}$. That is, $\sum_{i \in Q} \varphi^i \equiv 1$ where each $\varphi^i : C \to [0, 1]$ is a continuous function with $\varphi^i(x) = 0$ if $x \notin U_i$. For each i let $f : C \to \mathbb{R}^2$ be the function

$$f(u) = u + \sum_{i \in Q} \varphi^i(u) \chi^i.$$

Of course f is continuous. Concrete arguments show that the image of f is contained in C.[11]

We will apply 2-FIXED POINT to f and a suitable $\varepsilon > 0$. The idea is that a weighted average of the colors cannot be $(0, 0)$ unless all three weights are equal, and, by continuity, such a weighted sum cannot be close to $(0, 0)$ unless the weights are approximately equal. If $d(u^*, f(u^*))$ is sufficiently small, then it must be the case that there are $i, j, k \in Q$ with $\varphi^i(u^*), \varphi^j(u^*), \varphi^k(u^*) > 0$ and

$$\{\chi^i, \chi^j, \chi^k\} = \{\chi_0, \chi_x, \chi_y\}.$$

The intersection $U_i \cap U_j \cap U_k$ is nonempty (it contains u^*) and this can happen only if for both $w = x, y$, the set $\{i_w, j_w, k_w\}$ is either a singleton $\{g_w\}$ or a pair $\{g_w - 1, g_w\}$. Let $g = (g_x, g_y)$. Then $\frac{1}{N} g$ is a common vertex of C_i, C_j, and C_k. Since boundary vertices are touched by one of two squarelets, g must be interior, so it is panchromatic.

[11] Fix $u \in C$. Our first goal is to demonstrate that $f_1(u) \leq 1$, and to this end we may suppose that $f_1(u) > u_x$. The formula defining f implies that there is some i such that $\varphi^i(u) > 0$ and $\chi^i = \chi_x$. Legality implies that either $i_x = 0$ or C_i is not part of the surface layer of the square, so $i_x \leq N - 2$. Since $f_1(u) \leq u_x + \alpha$, $u_x < \frac{i_x}{N} + \frac{5}{4N}$, and $\alpha < \frac{3}{4N}$, we have $f_1(u) < 1$. The next goal is to show that $f_1(u) \geq 0$, so suppose that $f_1(u) < u_x$. Now there must be an i with $\varphi^i(u) > 0$ and $\chi^i = \chi_0$, and legality implies that $i_x \geq 1$. We have $f_1(u) \geq u_x - \alpha$, $u_x > \frac{i_x}{N} - \frac{1}{4N}$, and $\alpha < \frac{3}{4N}$, so $f_1(u) > 0$. Similarly, $f_2(u), f_3(u) \in [0, 1]$.

The remaining issue is to show how we can arrange for the output of 2-FIXED POINT (the vertices of a completely labelled simplex) to consist of approximate fixed points of f. Observe that ψ^i is Lipschitz with Lipschitz constant $4N$, relative to the metric d. Since Q has N^2 elements and $\psi^i(u) \leq 1 \leq \sum_{j \in Q} \psi^j(u)$ for all i and $u \in C$, the calculation

$$\begin{aligned}
\left|\varphi^i(u) - \varphi^i(u')\right| &= \left| \frac{\psi^i(u)}{\sum_{j \in Q} \psi^j(u)} - \frac{\psi^i(u')}{\sum_{j \in Q} \psi^j(u')} \right| \\
&\leq \frac{\sum_{j \in Q} \left(|\psi^j(u') - \psi^j(u)| \cdot \psi^i(u) + \psi^j(u) \cdot |\psi^i(u) - \psi^i(u')| \right)}{\left(\sum_{j \in Q} \psi^j(u) \right) \cdot \left(\sum_{j \in Q} \psi^j(u') \right)} \\
&\leq \sum_{j \in Q} \left(|\psi^j(u') - \psi^j(u)| + |\psi^i(u) - \psi^i(u')| \right)
\end{aligned}$$

shows that φ^i is Lipschitz with Lipschitz constant $8N^3$. In turn this implies that f is Lipschitz with Lipschitz constant $16N^5\alpha$.

The particular Lipschitz constant is unimportant. The real point is that we know what it is, so we can choose ε to be small enough, relative to it. The map $u \mapsto \frac{1}{2}(2 - u_x - u_y, u_x, u_y)$ embeds C in the standard 2-simplex, and it is easy to specify a retraction of the simplex onto the image of this embedding. If we apply 2-FIXED POINT to f and $\varepsilon > 0$, the result will be $v^0, v^1, v^2 \in C$ that are within 4ε of each other, relative to the metric d, with

$$2 - v^0_x - v^0_y \geq 1 - f_x(v^0) - f_y(v^0) - 2\varepsilon, \quad v^1_x \geq f_x(v^1) - 2\varepsilon, \quad v^2_y \geq f_y(v^2) - 2\varepsilon.$$

If ε is small enough, relative to the Lipschitz constant for f, then $f(v^0)$, $f(v^1)$, and $f(v^2)$ will be close enough to each other for these inequalities to imply that v^0, v^1, and v^2 are approximate fixed points of f, for any desired precision of approximation. In particular, we can make the precision sufficient to support the argument above. This completes the description of the reduction.

13.4 From END OF THE LINE to 2D BROUWER

As with **NP**, getting a first concrete problem that is **PPAD**-complete is one of the biggest hurdles. Papadimitriou (1994a) showed that the three dimensional version of 2D BROUWER is **PPAD**-complete, and since then that result has played a central role in the further development of the topic. The ideas in the proof extend to higher dimensions without great difficulty. On the other hand the one dimensional version of 2D BROUWER has a very fast algorithm[12] and consequently cannot be **PPAD**-complete. Thus the remaining question is whether the two dimensional analogue 2D BROUWER is **PPAD**-complete,

[12]For $i = 0, \ldots, N-1$ let $C_i = [\frac{1}{N}i, \frac{1}{N}(i+1)]$, and suppose that each of these is assigned a color $\chi^i \in \{\chi_0, \chi_x\}$, with $\chi^0 = \chi_x$ and $\chi^{N-1} = \chi_0$. Choose a C_{i_1} toward the middle. If $\chi^{i_1} = \chi_0$, set $I_1 = C_0 \cup \ldots \cup C_{i_1}$, and otherwise set $I_1 = C_{i_1} \cup \ldots \cup C_{N-1}$. Now choose C_{i_2} toward the middle of I_1, and let I_2 be whichever of the two obvious subintevals containing C_{i_2} has extreme subintervals of opposite colors. Continue in this manner finds i such that C_i and C_{i+1} have different colors after a number of iterations that is on the order of $\log N$.

and Chen and Deng (2006a) showed that in fact it is. This section presents a slight reworking of their argument.

Fix an instance of END OF THE LINE given by Boolean circuits S and P, each with m input bits and m output bits. Below, when we describe an algorithm as polynomial time, we mean that its running time is bounded by a polynomial function of the sum of the sizes of S and P. The framework of 2D BROUWER is taken from the last section: $Q = \{0, \ldots, N-1\}^2$, where N is an integer that must be large enough, relative to m, to contain all the phenomena described below. Our goal is to pass from S and P to a legal, polynomial time computable assignment $i \mapsto \chi^i$ of colors to elements of Q such that the polychromatic vertices are in one-to-one correspondence with the sink and sources, other than 0^m, of $G_{S,P}$.

For the sake of brevity, we will henceforth refer to the 1-skeleton of the polytopal subdivision of C given by the squarelets C_i, $i \in Q$, simply as **the 1-skeleton**. It can be understood as a graph whose vertices are the points $\frac{1}{N}i$ for $i \in \{0, \ldots, N\}^2$, or as a one dimensional simplicial complex whose 0-simplices are the $\frac{1}{N}i$, and we will not be particularly careful about this distinction, trusting the reader to understand everything as intended. Note that $\{\frac{1}{N}i, \frac{1}{N}j\}$ is an edge of the 1-skeleton if and only if i and j are elements of $\{0, \ldots, N\}^2$ that agree in one components and differ by one in the other.

A **directed path** in the 1-skeleton is a sequence $\frac{1}{N}i_1, \ldots, \frac{1}{N}i_h$ of distinct vertices such that for each $g = 1, \ldots, h-1$, $\{\frac{1}{N}i_g, \frac{1}{N}i_{g+1}\}$ is an edge in the 1-skeleton. We say that $\frac{1}{N}i_1$ is the **tail** of the directed path and $\frac{1}{N}i_h$ is its **head**. In our specifications of directed paths we abbreviate by including only the turning points, so that, for example, we write

$$\tfrac{1}{N}(1,1) \to \tfrac{1}{N}(3,1) \to \tfrac{1}{N}(3,3)$$

rather than

$$\tfrac{1}{N}(1,1) \to \tfrac{1}{N}(2,1) \to \tfrac{1}{N}(3,1) \to \tfrac{1}{N}(3,2) \to \tfrac{1}{N}(3,3).$$

The construction has three phases. In the first phase each element of $V_{S,P}$ and $A_{S,P}$ is represented by a directed path. In the usual way a bitstring $v \in \{0,1\}^m$ can be understood as the binary representation of a number in the range $0, \ldots, 2^m - 1$ that we denote by $\langle v \rangle$. Each $v \in V_{S,P}$ is represented by the directed path

$$\tfrac{1}{N}(2, 20\langle v \rangle + 1) \to \tfrac{1}{N}(2, 20\langle v \rangle + 11).$$

The directed path representing an arrow $(v, w) \in A_{S,P}$ with $v \neq 0^m$ is

$$\begin{aligned}\tfrac{1}{N}(2, 20\langle v \rangle + 11) &\to \tfrac{1}{N}(10(2^m\langle v \rangle + \langle w \rangle), 20\langle v \rangle + 11) \\ &\to \tfrac{1}{N}(10(2^m\langle v \rangle + \langle w \rangle), 20\langle w \rangle + 1) \to \tfrac{1}{N}(2, 20\langle w \rangle + 1).\end{aligned}$$

It consists of a horizontal portion beginning at the head $\frac{1}{N}(2, 20\langle v \rangle + 11)$ of the path representing v, a vertical portion whose horizontal component is $\frac{1}{N}10(2^m\langle v \rangle + \langle w \rangle)$, and a second horizontal component ending at the tail $\frac{1}{N}(2, 20\langle w \rangle + 1)$ of the path representing w.

Papadimitriou's original construction, and the reworking of it in Daskalakis et al. (2009a), proceed along similar lines, associating directed paths in the 1-skeleton of the induced polyhedral subdivision of the cube. Because any graph can be embedded in the cube, it is possible to do this in a way that guarantees that the various elements have no intersections other than the vertices where the tail and head of the path representing an arrow (v, w) touch the head of the path representing v and the tail of the path representing w respectively. Consequently combining all the paths for elements of $V_{S,P}$ and $A_{S,P}$ gives a large directed graph whose sources are the tails of the directed paths representing sources in $G_{S,P}$ and whose sinks are the heads of the directed paths representing sinks in $G_{S,P}$. The construction then passes from this graph to a coloring of the adjacent "cubelets" that is similar to, but more complicated than, what we will see below.

Of course not every graph can be embedded in the plane, and the paths representing the arrows in $A_{S,P}$ will have intersections. The key insight of Chen and Deng (2006a) is that after we take the union of all the directed paths, we can modify the portions of the union near each of these intersections, replacing the intersection with two nonintersecting directed paths, thereby obtaining an embedded graph. This is the second phase of the construction. The resulting graph is not a straightforward rendering of $G_{S,P}$, but the modifications do not change the sources and sinks, and for us this is enough.

It is evident that the directed paths representing the various elements of $V_{S,P}$ do not intersect each other. In addition, the only vertices where a path representing an element of $V_{S,P}$ intersects a path representing an element of $A_{S,P}$ are those given by the head-to-tail arrangement.

We now consider how the paths representing the edges might intersect. The vertical components of the vertices on the two horizontal portions of the path representing $(v, w) \in A_{S,P}$ are $\frac{1}{N}(20\langle v\rangle + 11)$ and $\frac{1}{N}(20\langle w\rangle + 1)$ respectively. Evidently the vertical component of one of these portions identifies the vertex it attaches to, and whether the vertex is the tail or head. Each vertex in $G_{S,P}$ has at most one direct predecessor and at most one direct successor, so the horizontal portions of paths representing two different elements of $A_{S,P}$ do not intersect. Now observe that one can recover (v, w) from the horizontal component $\frac{1}{N}10(2^m\langle v\rangle + \langle w\rangle)$ of the vertical portion of the path representing (v, w). Therefore the vertical portions of paths representing two different elements of $A_{S,P}$ do not intersect. In view of all this, if there is an intersection of the directed paths representing two arrows, it must be an intersection of the vertical segment of the path representing one of the arrows with one of the horizontal segments of the path representing the other.

Except for a fringe along the boundary, the square can be divided into 10×10 blocks of squarelets

$$[\tfrac{1}{N}(10(2^m r + s) + 5), \tfrac{1}{N}(10(2^m r + s) + 15)] \times [\tfrac{1}{N}(10t + 6), \tfrac{1}{N}(10t + 16)]$$

where $0 \le r, s \le 2^m - 1$ and $0 \le t \le 2^{m+1} - 1$. Of course such a block need not intersect any of the paths associated with elements of $A_{S,P}$. The intersection of such a block with the various paths may consist of part of a horizontal portion of a directed path associated with one element of $A_{S,P}$, in which case the portion goes through the block's central vertex. Or it may consist of part of the vertical portion of such a path, again going through the block's central vertex.

Finally, such a block may contain part of one of the horizontal portions of the path associated with some element of $A_{S,P}$ and part of the vertical portion of the path associated with a different element, with the two portions intersecting at the center of the block. There are four possibilities, depending on whether the horizontal portion is going from left to right or right to left and whether the vertical portion is going up or down. For one of these Figure 4 shows how one can replace the intersection with two nonintersecting paths while preserving the entries and exits from the 10×10 block. Reflecting this scheme across the horizontal and vertical centerlines of the array give methods for handling the other three cases.

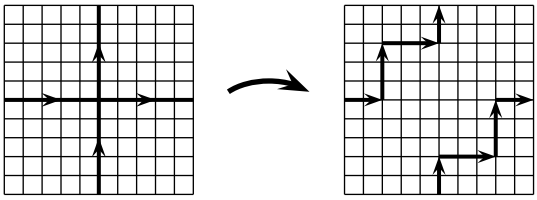

Figure 4

Let $\Gamma_{S,P}$ be the result of first taking the union of the directed paths associated with the elements of $V_{S,P}$ and $A_{S,P}$, then performing the modifications described above to eliminate all undesired intersections. This is (properly speaking, the set of arrows of) a directed graph whose vertices have maximal indegree one and maximal outdegree one. The sources of $\Gamma_{S,P}$ are the vertices $\frac{1}{N}(2, 20\langle v\rangle + 1)$ where v is a source of $G_{S,P}$, and the sinks are the vertices $\frac{1}{N}(2, 20\langle v\rangle + 11)$ where v is a sink of $G_{S,P}$.

The third phase of the construction passes from $\Gamma_{S,P}$ to a legal assignment of a color χ^i to each $i \in Q$. The assignment of colors is given by a number of rules. The first rule is that legality prescribes the assignment when $\{i_x, i_y\} \cap \{0, N-1\} \neq \emptyset$. We say that i is **interior** if this is not the case. We say that i **touches** $\Gamma_{S,P}$ if one of the vertices of C_i is a vertex of $\Gamma_{S,P}$ that is not a sink or a source. The second rule governing the assignment is that if i is interior and does *not* touch $\Gamma_{S,P}$, then $\chi^i = \chi_0$. Thus the vast majority of interior squarelets have color χ_0. The final rule is that if C_i touches $\Gamma_{S,P}$ at a vertex that is neither a source nor a sink, then it receives color χ_x if it is on the left as one proceeds along $\Gamma_{S,P}$ going past the vertex in the given direction, and it is assigned color χ_y if it is on the right.

Figure 5 shows how this works in the case $m = 1$, when $V_{S,P}$ has two elements, 0 and 1, and $(0, 1)$ is necessarily the unique element of $A_{S,P}$. Note that there is no panchromatic vertex near $(0, 0)$. In effect the panchromatic vertex $\frac{1}{N}(1, 1)$ of the assignment of colors in which χ_0 is assigned to every squarelet other than those where legality dictates another choice "cancels" the source 0^m of $G_{S,P}$, which of course is not a solution of the given instance of END OF THE LINE.

There are several things that need to be checked. As can be seen in Figure 5, the coloring scheme goes around corners without any difficulty. It must be the case that the last rule does not give rise to any ambiguity. That is, if a squarelet touches $\Gamma_{S,P}$ in multiple ways, they must all induce the same color assignment. In addition, it should not induce any undesired panchromatic vertices, as could happen if

unrelated portions of $\Gamma_{S,P}$ came too close to each other. By considering the various possibilities for the 10×10 blocks, one can easily see that such problems do not occur, and as can be seen in Figure 6, after the region near an intersection has been modified, the problem we are trying to avoid does not occur.

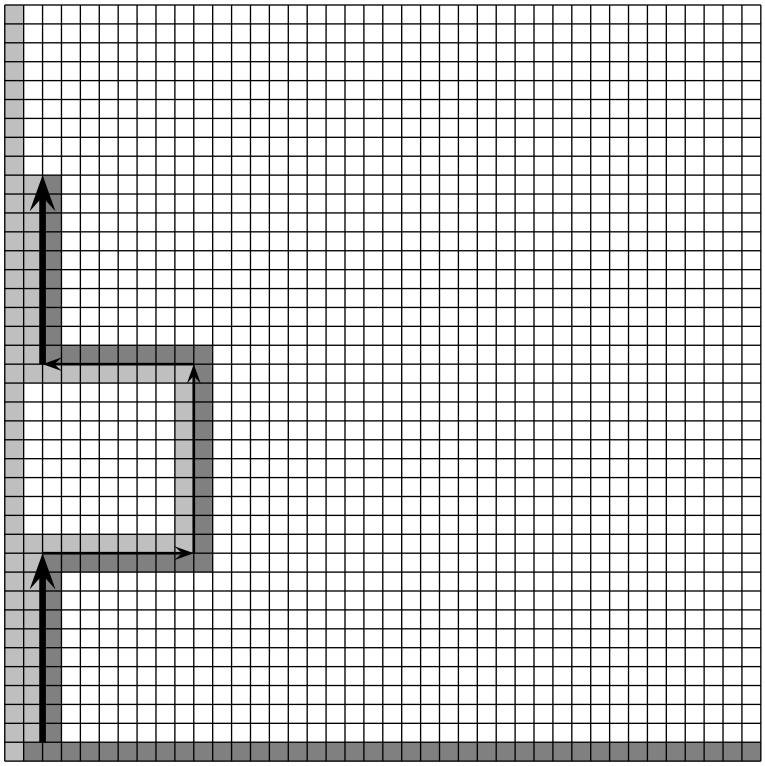

Figure 5

Finally, the sources (other than $(2, 1)$) and sinks of $\Gamma_{S,P}$ should be panchromatic. Extrapolating from the example given by Figure 5, it is easy to see the vertices $\frac{1}{N}(2, 20\langle v\rangle + 1)$ where $v \neq 0^m$ is a source of $G_{S,P}$ and the vertices $\frac{1}{N}(2, 20\langle v\rangle + 11)$ where v is a sink of $G_{S,P}$ are, in fact, panchromatic.

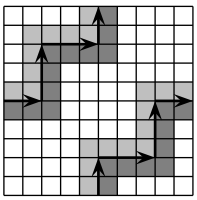

Figure 6

The description of the reduction is completed by an algorithmic explanation of how a color is assigned to $i \in Q$. We think of the square being divided into various rectangles, with the main case

being that the squarelet associated with i is contained in one of the 10×10 blocks above. We can easily pass from i_x and i_y to the integers r, s, and t, and then identify the bit strings $u, v, w \in \{0,1\}^m$ such that $r = \langle u \rangle$, $s = \langle v \rangle$, and $t \in \{2\langle w \rangle, 2\langle w \rangle + 1\}$. We can then use S and P to figure out which of u, v, and w are in $V_{S,P}$ and whether or not $(v, w) \in A_{S,P}$. This information can obviously then be used to determine the colors assigned to the squarelets in this block, including the one associated with the given i. The other types of rectangles along the boundary of the square are simpler, with details that can easily be worked out by the interested reader and embodied in acceptable algorithms. Evidently all the algorithms described here are fast enough.

13.5 From 2D BROUWER to GRAPHICAL GADGET GAME

Fix an instance of 2D BROUWER. Specifically, let B be a Boolean circuit that takes $i \in Q = \{0, \ldots, N-1\}^2$ as input and outputs a legal color for C_i, where as before the C_i give a polytopal subdivision of $C = [0,1]^2$. Our reduction of this to an instance of GRAPHICAL GADGET GAME is essentially the one given by Daskalakis et al. (2006a), except that it takes place in two dimensions instead of three.

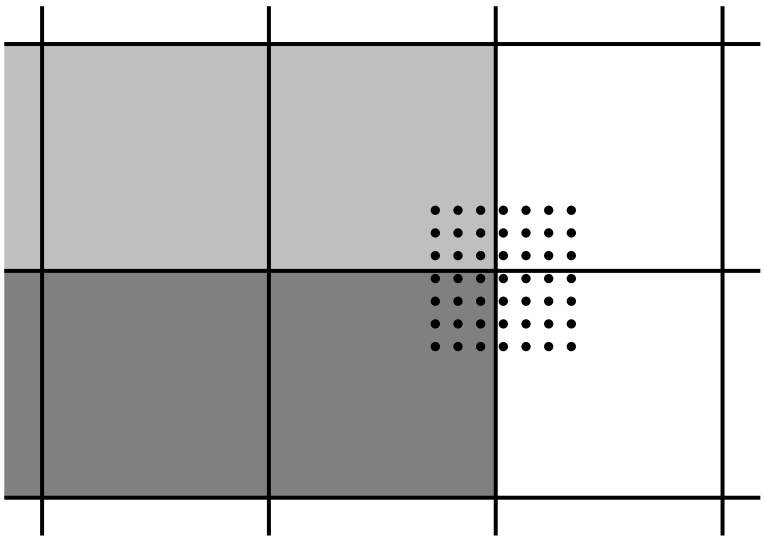

Figure 7

We assume that $N = 2^n$ for some n; this is without loss of generality because we can always embed the given square in a larger one of this sort, coloring the additional squarelets (aside from those constrained by legality) with χ_0. For the same reason we can assume that n is as large as it needs to be to support any step in the argument verifying that the construction works. Let $\alpha = 1/N^2 = 2^{-2n}$, and let $\varepsilon = \alpha^2 = 2^{-4n}$. The specific numbers are not so important here: the idea is that α is much smaller than $1/N$ and ε is much smaller than α.

The game we construct embeds a network of gadgets of the sort described in the last section, and

the computation that this network expresses amounts to a requirement that, in ε-Nash equilibrium, a certain point $v = (v_x, v_y) \in C$ is close to squarelets of all three colors, so that some nearby vertex must be panchromatic. This situation will be detected by scanning a small grid of points near v, as shown in Figure 7. Let m be a fixed positive integer, about which we will have more to say later. For

$$d = (d_x, d_y) \in \{-m, -m+1, \ldots, m-1, m\}^2$$

let $v^d = v + \alpha d$. For each d the network of gadgets has a subnetwork devoted to processing v^d.

Fix a d. Except when v is very close to the boundary of the square (about which we will say more later) we can use the various gadgets to create a $\tilde{v}^d$ that closely approximates v^d. Concretely, this is done by using $G_=$ to copy v_x and v_y, using G_α to create a variable approximating α, and then applying G_- and/or G_+ repeatedly.

Next, the n most significant bits of the binary representation of $\tilde{v}_x^d$ are extracted. Specifically, we first use $G_=$ to make x_1 approximately equal to $\tilde{v}_x^d$. We use $G_<$ to decide whether $x_1 > 1/2$. If it is we set $b_{x1}^d = 1$ and $x_2 = x_1 - 2^{-1}$, and otherwise we set $b_{x1}^d = 0$ and $x_2 = x_1$. Proceeding recursively, for $h = 2, \ldots, n$ we use $G_<$ to decide whether $x_h > 2^{-h}$. If it is we set $b_{xh}^d = 1$ and $x_{h+1} = x_h - 2^{-h}$, and otherwise we set $b_{xh}^d = 0$ and $x_{h+1} = x_h$. When v_x^d is too close to an integral multiple of 2^{-n} this procedure may fail due to the imprecision of $G_<$, and eventually we will have to deal with this, but for the time being suppose that it succeeds. We can interpret $b_{x1}^d \ldots b_{xn}^d$ as the binary representation of an integer i_x^d in the range $0, \ldots, N-1$. In the same way the bits extracted from $\tilde{v}_y^d$ defines an integer i_y^d. Thus we have computed a point $i^d = (i_x^d, i_y^d) \in Q$. We now use a copy of the circuit B, encoded as a subnetwork using the gadgets $G_\vee$, $G_\wedge$, and $G_\neg$, to find the color χ^d associated with i^d.

Having found χ^d for each of the possible values of d, we now compute $v' = v + \frac{1}{M}\sum_d \chi^d$ where $M = (2m+1)^2$ is the number of points in the grid. Finally we use $G_=$ to require that, in ε-Nash equilibrium, it is approximately the case that $v' = v$. This completes the description of the network of gadgets.

In any ε-Nash equilibrium of the game described above it will be the case that $\sum_d \chi^d$ is close to 0. The main idea is that if this is the case, then the colors associated with the squarelets containing the various v^d have each of the four colors, because if a weighted sum of the colors is close to the origin, then the weights must be nearly equal. Consequently one can easily find a panchromatic i by checking those i with i/N close to v.

Turning this intuition into a rigorous argument depends on considering what might go wrong. The Boolean circuit B uses only the gadgets $G_\vee$, $G_\wedge$, and $G_\neg$, so the behavior of the copy used in the processing of $\tilde{v}^d$ is absolutely reliable so long as its inputs are all in $\{0, 1\}$. There is the possibility that v_x^d or v_y^d is very close to an integral multiple of 2^{-n}, in which case the extracted bits need not be elements of $\{0, 1\}$, and the subsequent behavior of the copy of B is unpredictable. Since α is much smaller than 2^{-n} and ε is much smaller than α, there is at most one value of d_x in the range $-m, \ldots, m$ such that v_x^d might have this problem, even taking into account the accumulation of error in the calculation of $\tilde{v}_x^d$ and the subsequent extraction of bits. Of course the same argument pertains

to the other two variables. Therefore the number of potentially flaky grid points is bounded above by $2(2m+1)$, and the idea of the argument retains its validity so long as $2(2m+1)/M = 2/(2m+1)$ is small enough.

Another possibility is that v is very close to the boundary of the square. This can result in some of the $\tilde{v}_x^d$ or $\tilde{v}_y^d$ being set equal to 0 or 1. Although 0 and 1 are integral multiples of 2^n, it turns out that for these numbers the bit extraction process reliably returns the binary representations of 0 and $2^n - 1$. In turn this implies that all squarelets C_{i^d}, aside from those that are flaky for reasons described above, will be from (at most) two squarelets along the boundary of the square that share a common edge. Since legality does not allow any panchromatic vertices on the boundary of the square, it follows that in equilibrium v cannot be that close to the boundary of the square.

13.6 From GRAPHICAL GADGET GAME to 2-NASH

Daskalakis et al. (2006a) showed how to reduce from GRAPHICAL GADGET GAME to normal form games with three players by incentivizing each of the players to act in accord with the interests of a certain subset of the nodes in the graphical game. In this way they were able to show that 3-NASH is **PPAD**-complete. Shortly thereafter Chen and Deng (2006b) showed how to embed all of the gadgets in a large two player game, thereby showing that 2-NASH is **PPAD**-complete. Subsequently Daskalakis et al. (2009a) pointed out that a fairly simple extension of their methods could also be used to obtain this result.

Here we describe the methods of Chen and Deng. Consider a large two player zero sum game of the sort shown below. Here M is a large positive number, all blank payoffs are zero, and the number K of 2×2 blocks along the diagonal is the number of gadgets in the graphical game we wish to "simulate." We index the rows and columns by $10, 11, 20, 21, \ldots, K0, K1$. If the row player plays ib and the column player plays $i'b'$, then the row player wins M and the column player loses M if $i = i'$, and otherwise both players' payoffs are zero. Therefore a mixed strategy profile (σ, τ) is a Nash equilibrium if and only if, for each $i = 1, \ldots, K$,

$$\sigma_{i0} + \sigma_{i1} = \frac{1}{K} = \tau_{i0} + \tau_{i1}.$$

$(M,-M)\,(M,-M)$			
$(M,-M)\,(M,-M)$			
	$(M,-M)\,(M,-M)$		
	$(M,-M)\,(M,-M)$		
		$\ddots$	
			$(M,-M)\,(M,-M)$
			$(M,-M)\,(M,-M)$

The idea now is to perturb the payoffs so that for each $i = 1, \ldots, K$, the division of probability between σ_{i0} and σ_{i1}, and the division of probability between τ_{i0} and τ_{i1}, mimic the mixed strategies of the i^{th} gadget in ε-Nash equilibrium. In this way we encode variables $p_1, \ldots, p_K$, where

$$p_i = \frac{\sigma_{i1}}{\sigma_{i0} + \sigma_{i1}},$$

and induce relationships between them that approximate the relationships induced by the gadgets. For example, to enforce that p_i is approximately the sum of p_{i_1} and p_{i_2}, as per G_+, we add the following to the $i0$ and $i1$ columns of the bimatrix:

$$\begin{array}{cc} \vdots & \\ i_1 0 & (0,0)\ (0,0) \\ i_1 1 & (0,1)\ (0,0) \\ \vdots & \\ i_2 0 & (0,0)\ (0,0) \\ i_2 1 & (0,1)\ (0,0) \\ \vdots & \\ i0 & (0,0)\ (1,0) \\ i1 & (1,0)\ (0,1) \\ \vdots & \end{array}$$

Perturbing the game in this way induces small changes in the values, in ε-Nash equilibrium, of the sums $\sigma_{i0} + \sigma_{i1}$ and $\tau_{i0} + \tau_{i1}$, and these perturbations are an additional source of error. The magnitude of these errors can be controlled by choosing M to be a very large number. There are some challenging calculations, but in the end the complications are technical rather than conceptual, and with sufficient work it is possible to prove that the apparatus works satisfactorily.

14 Approximate Nash Equilibrium and Smoothed Complexity

In the face of a barrier to efficient solution of some computational problem, it is natural to look at less demanding problems. During the last few years there have been many papers that considered approximate Nash equilibrium. In formulating this problem it is natural to normalize payoffs by requiring that all payoffs lie in $[0, 1]$, with each agent having at least one payoff at each endpoint of this interval.

Perhaps the simplest problem is to find an ε-approximate Nash equilibrium of a bimatrix game for fixed ε. Daskalakis et al. (2006b) pointed out that if r_1 is any pure strategy for the row player, c is the column player's best response to r_1, and r_2 is the row player's best response to c, then $(\frac{1}{2}r_1 + \frac{1}{2}r_2, c)$ is a $\frac{1}{2}$-approximate Nash equilibrium because each player is playing a best response to the other player's pure strategy at least half the time. Feder et al. (2007) showed that this bound cannot be improved if one restricts attention to mixed strategies with fixed support size. Daskalakis et al. (2007) gave an

algorithm that for any $\varepsilon > 0$ gives a $((\sqrt{5}-1)/2+\varepsilon)$-approximate Nash equilibrium in time $n^{O(1/\varepsilon^2)}$. Bosse et al. (2007) gave a simpler polynomial time algorithm achieving this result, and also gave a polynomial time algorithm for $\varepsilon = 0.3639$. The best result in this direction to date, due to Spirakis and Tsaknakis (2007), is a polynomial time algorithm for finding a 0.3392-approximate Nash equilibrium.

Lipton et al. (2003) provided stronger approximations, at the cost of increased asymptotic running times. Suppose A and B are $n \times n$ matrices with entries between 0 and 1. They showed that for any Nash equilibrium (σ^*, τ^*) of the game (A, B), any $\varepsilon > 0$, and any $k > 12 \ln n/\varepsilon^2$, there is a pair of mixed strategies (σ, τ) that is an ε-approximate Nash equilibria whose expected payoff for each agent is within ε of the equilibrium expected payoff, and whose mixture probabilities are all integer multiples of $1/k$, so that the support size is at most k. The proof is a matter of showing that taking k independent draws for each agent distributed according to σ^* and τ^* has a positive probability of generating samples with distributions satisfying all these properties. Searching over all possible pairs of draws of k elements from the two agents sets of pure strategies is consequently an algorithm for finding an ε-Nash equilibrium. Its running time is bounded by $n^{C \ln n/\varepsilon^2}$ for some constant $C > 0$. (A **quasi-polynomial time algorithm** is one whose running time is bounded by $n^{p(\ln n)}$ where p is a polynomial.) Feder et al. (2007) gave results showing that certain smaller (but still logarithmic) support sizes do not give similar degrees of approximation.

Two papers extend these results to games with $r \geq 2$ players. Breist et al. (2008) gave a simple polynomial times algorithm that finds a $(1-1/r)$-approximate Nash equilibrium with support size two. They also showed that this bound cannot be improved using fixed support size strategies. Hémon et al. (2008) independently gave a polynomial time algorithm for a $(1-1/r)$-approximate Nash equilibrium with support size two. They also gave bounds on the support size required to achieve ε-approximate Nash equilibrium, and they refined the analysis of Lipton et al. (2003) in certain ways.

The other findings related to approximate Nash equilibrium significantly strengthen the central **PPAD**-hardness result. This topic provides an opportunity to describe another major advance in computer science in recent years. Consider the linear program

$$\text{maximize } c^T x \quad \text{subject to} \quad Ax \leq b$$

where A is an $m \times d$ matrix, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^d$. The simplex algorithm solves this problem by "walking uphill" along the edge graph of the polyhedron $\{\, x : Ax \leq b \,\}$. There are a variety of rules for choosing the next edge if more than one is improving; for example, the **shadow vertex pivot rule** follows the edges that project onto the boundary of the feasible set's image under a projection onto the plane spanned by c and another vector. Following the seminal example of Klee and Minty (1972), examples of families of problems for which the simplex algorithm takes exponentially many steps before reaching an optimum were found for most pivot rules in the literature. In practice, however, the simplex algorithm remains quite useful, even though algorithms for linear programming with polynomial worst case complexity (Khachian (1979), Karmarkar (1984)) were discovered around thirty years ago. There are a number of results that show that the simplex algorithm has a low expected running time, relative

to particular distributions on the space of problem instances. Although such results, and their proofs, can give some greater understanding of the phenomenon, they are of limited relevance insofar as the assumed distributions are unrelated to the distributions of problems that arise in various applications.

A more comprehensive and compelling result was given by Spielman and Teng (2004). They showed that for any given (suitably normalized) A and b, if $\tilde{A}$ and $\tilde{b}$ are obtained by adding independently distributed Gaussian random variables with standard deviation σ to the entries of A and b, then the expected running time of the simplex algorithm with the shadow vertex pivot rule, applied to $\tilde{A}$, $\tilde{b}$, and c, is bounded by a polynomial function of d, m, and $1/\sigma$. (They also obtained a similar result when the disturbances of the entries of A and b are independently and uniformly distributed.) Since we can average this result across a distribution over the set of pairs (A, b), this gives a sense in which the mean running time is low for any distribution of problem instances that is not concentrated near particular points. This result is described by saying that the simplex algorithm with shadow vertex pivot rule has **polynomial smoothed complexity**.

This result also gives a random algorithm for finding an approximate solution of the original problems. In general a **random algorithm** for a computational problem is an algorithm that, in addition to the portion of the input representing the given problem instance, also takes as input an auxilary object, which we imagine to be randomly chosen, from some (typically quite large) space. In order for the algorithm to be acceptable, it must be the case that for at least one half of the elements of this space the computation terminates with an acceptable output for the problem instance. (The number one half is arbitrary: one can, by repeating the calculation, achieve any desired probability of success.) The class of decision problems which have a random algorithm that runs in polynomial time is **RP**. Evidently **P** $\subset$ **RP**, and **RP** $\subset$ **NP** $\cap$ **coNP** because the definition of **RP** is a strengthening of the definition of **NP** $\cap$ **coNP**. (Instead of requiring that there is at least one certificate that verifies whichever of the two answers is correct, **RP** requires that half of them have this property.) As with other such inclusions, which of these is strict is unknown. Following Chen et al. (2009b) we abuse terminology, extending the meaning of **RP** to include search problems which have polynomial time random algorithms that succeed with high probability.

In general, a **polynomial time approximation scheme** for a quantitative search problem is an algorithm that, for any chosen value of $\varepsilon > 0$, computes an ε-solution (in a sense that is appropriate to the problem) with probability $1 - \varepsilon$ in an amount of time that is bounded by a polynomial function of the size of the input. If the running time is bounded by a polynomial function of the size of the input and $1/\varepsilon$, then it is a **fully polynomial time approximation scheme** (FPTAS). A solution of the given linear program for a nearby $\tilde{A}$ and $\tilde{b}$ can be converted into an approximate solution for the given problem by passing to a nearby point in the set of feasible solutions for the given problem.[13] Thus the Spielman-Teng result shows that adding random disturbances to the parameters of the problem, then applying the

[13]For the sake of expositional simplicity we are ignoring the possibility that the feasible set may be empty. As a general matter the simplex method detects this situation, or alternatively finds an initial feasible point, by considering a relaxed problem.

simplex algorithm with the shadow vertex pivot rule, is a random algorithm that finds an ε-approximate solution of the problem with mean running time bounded by a polynomial function of M, d, and $1/\varepsilon$. If we modify this by terminating the calculation when the number of pivots exceeds some appropriately chosen bound, so that the computation always terminates in a polynomially bounded amount of time, but fails to find a solution with an acceptably small probability, then we obtain a FPTAS for linear programming.

The Lemke-Howson algorithm, which finds a Nash equilibrium of a bimatrix game, is similar to the simplex algorithm in several ways. Nash equilibrium can be defined by the requirement that each pure strategy is either a best response or is assigned no probability. If we relax this by requiring that it hold for all but the row player's first pure strategy, then for generic given data the set of mixed strategy profiles satisfying the relaxed condition is one dimensional, and includes a path with a known starting point, where the row player is playing her first pure strategy and the column player is playing the best response to this, whose other endpoint is necessarily a Nash equilibrium. The Lemke-Howson algorithm follows this path. The path can be represented as a walk that takes alternating steps on a pair of polytopes, and its numerical implementation in terms of a tableau closely resembles the numerical implementation of the simplex algorithm. Although its worst case complexity is exponential (Savani and von Stengel (2006)) it is quite practical even for games with hundreds of pure strategies. Thus one might guess that it satisfies results similar to those we saw above.

Chen et al. (2006a, 2009b) showed that this is not the case at all. Consider a fixed two player game (A, B) where A and B are normalized $n \times n$ payoff matrices for the row and column player respectively. A Nash equilibrium (σ^*, τ^*) of a perturbation $(\tilde{A}, \tilde{B})$ of this game will necessarily be an approximate Nash equilibrium of (A, B) because for any σ and τ we have

$$\sigma A \tau^* - \sigma^* A \tau^* \leq \sigma \tilde{A} \tau^* - \sigma^* \tilde{A} \tau^* + |(\sigma - \sigma^*)(\tilde{A} - A)\tau^*|$$

and similarly for the second player. In particular, suppose that the perturbations $\tilde{a}_{ij} - a_{ij}$ and $\tilde{b}_{ij} - b_{ij}$ are independently random variables that are uniformly distributed on the interval $[-\delta, \delta]$. Then a Nash equilibrium of the game $(\tilde{A}, \tilde{B})$ is an δ-approximate Nash equilibrium of the game (A, B), and we have already described how to pass from a δ-approximate Nash equilibrium to a ε-Nash equilibrium where ε is a suitable multiple of $\sqrt{\delta}$.

Suppose that there was a method of computing a δ-approximate Nash equilibrium whose expected running time was bounded by a polynomial function of n and $1/\delta$ when applied to a random two person game $(\tilde{A}, \tilde{B})$ that is distributed as described above. Since the running time cannot exceed twice the expected running time more than half the time, this could be turned into polynomial time randomized algorithm for computing a δ-approximate Nash equilibrium of (A, B). Setting $\delta = 1/n$, we would have a polynomial time random algorithm for computing a $1/n$-approximate Nash equilibrium. But Chen et al. (2006a) showed that for any $c > 0$ the problem of computing a $1/n^c$-approximate Nash equilibrium of a two person game is **PPAD**-complete. It follows that 2-NASH does not have polynomial smoothed complexity unless **PPAD** $\subset$ **RP**. This result stands in contrast with Bárány et al. (2007), who

provided a polynomial time algorithm that finds a Nash equilibrium with high probability, relative to certain natural models of a random bimatrix game, and also with the empirical results in this direction of Porter et al. (2008).

The argument in Chen et al. (2006a) is a technical tour de force, taking the methods described earlier to a new level of sophistication. In the reduction from 2D BROUWER to GRAPHICAL GADGET GAME described in the last section ε could be as small as needed, but for their purposes it is necessary that this quantity be carefully controlled. In order to achieve this it is necessary to treat the sampling grid more carefully, and also to taylor the dimension and the grid size of the version of BROUWER as required by the analysis. The main geometric idea in the latter construction is to repeatedly embed a give instance of BROUWER in an instance of BROUWER of the next higher dimension while reducing the grid size in one direction, using a snake-like embedding. They take advantage of the fact that, by virtue of the result described in Section 13, one can begin this process with 2D BROUWER, but it turns out that this is not essential: Goldberg et al. (2011) show that it is possible to achieve the desired conclusions starting with the three dimensional version of this problem.

An even higher degree of sophistication is achieved in Daskalakis (2011). A **relative ε-Nash equilibrium** for a bimatrix game is a pair of mixed strategies such that no probability is assigned to any pure strategy such that the loss that results from deviating to it, relative to the optimal strategy, is greater than ε times the absolute value of the expected payoff resulting from playing it. This concept, and relative ε-approximate Nash equilibrium (the relative analogue of ε-approximate Nash equilibrium) are invariant under multiplication of all payoffs by a positive scalar, whereas the absolute concepts are invariant under addition of a scalar to all payoffs. Daskalakis showed that for any $\varepsilon \in [0, 1)$ the problem of finding a relative ε-Nash equilibrium of a game with payoffs in $[-1, 1]$ is **PPAD**-complete. In addition to the sorts of gadgets seen earlier, the construction involves gadgets that detect error, amplify it, and then correct for it, in order to bound the total error of the network of gadgets.

15 General Economic Equilibrium

General equilibrium theory is one of the premier applications of fixed point theory. For this reason, and perhaps also because of the hope for applications to electronic commerce, beginning with Papadimitriou (2001) and Deng et al. (2003), computer scientists have become intensely interested in the computation of economic equilibria. This section surveys work since that time in the computer science literature, which includes both polynomial time algorithms for some problems and **PPAD**-hardness results. Esteban-Bravo (2004) surveys earlier theoretical literature on linear and nonlinear optimization approaches, primarily in economics and operations research. Judd et al. (2003) surveys some of the work in economics concerning computation of dynamic general equilibrium models with multiple agents.

We begin by specifying a fairly general form of the Arrow-Debreu exchange economy with k agents

and ℓ goods. Each agent $j = 1, \ldots, k$ has an initial endowment $e_j \in \mathbb{R}^\ell_\geq$ and a continuous, concave, and monotonic increasing utility function $u_j : \mathbb{R}^\ell_\geq \to \mathbb{R}$. A **quasi-equilibrium** is a nonzero price vector $p \in \mathbb{R}^\ell_\geq$ and a k-tuple $x_1, \ldots, x_k \in \mathbb{R}^\ell_\geq$ of consumption bundles such that

(a) $p \cdot x_j \leq p \cdot e_j$ for all $j = 1, \ldots, k$,

(b) $p \cdot x'_j \geq p \cdot e_j$ for all $j = 1, \ldots, k$ and $x'_j \in \mathbb{R}^\ell_\geq$ such that $u_j(x'_j) > u_j(x_j)$, and

(c) $\sum_{j=1}^k x_j = \sum_{j=1}^k e_j$.

That is, no agent exceeds her budget, no agent can spend less while attaining a higher utility, and markets clear. For future reference we note that taking the inner product of both sides of (c) with p and comparing with (a) shows that it is actually the case that $p \cdot x_j = p \cdot e_j$ for all j.

A quasi-equilibrium is a **Walrasian equilibrium** if, in addition to (a) and (c), each agent is maximizing utility subject to her budget constraint:

(b′) $p \cdot x'_j > p \cdot e_j$ for all $j = 1, \ldots, k$ and $x'_j \in \mathbb{R}^\ell_\geq$ such that $u_j(x'_j) > u_j(x_j)$.

Quasi-equilibrium is not the conceptually correct equilibrium notion, but is technically useful: under certain standard assumptions fixed point theory can be used to show that a quasi-equilibrium exists, and consequently a Walrasian equilibrium necessarily exists if all quasi-equilibria are Walrasian equilibria, as is implied by a variety of other assumptions.

The Debreu-Mantel-Sonnenschein theorem (cf. Section 2) implies that the problem of computing a Walrasian equilibrium is, in full generality, as hard as any other fixed point problem. Thus we should expect that when the agents' preferences attain some level of complexity, equilibrium computation will be **PPAD**-complete. Ye (2007) and Codenotti et al. (2006) provided a reduction from bimatrix games to a particular class of exchange economies with Leontief utility functions. Thus the problem of computing a Walrasian equilibrium is **PPAD**-complete even for Leontief utilities! Huang and Teng (2007) strengthened this, showing that the computation of an ε-approximate Nash equilibrium of an $n \times n$ bimatrix game can be reduced to the computation of an $(\varepsilon/n)^2$-approximate market equilibrium. It follows from this that there is no FPTAS for Walrasian equilibrium unless **PPAD** is contained in **P**. By means of somewhat more complicated reductions, Chen et al. (2009a) showed that the problem of finding a Walrasian equilibrium is **PPAD**-complete, and (assuming that **PPAD** is not contained in **P**) has no FPTAS when the agents' utilities are additively separable, with the utility of each good is concave and piecewise linear with two linear pieces.

We now turn to positive results. A great deal of work has been devoted to the case in which all agents have linear utilities: $u_j(x) = \sum_i c_{ij} x_i$. Jain et al. (2003) gave a FPTAS for Walrasian equilibrium in this case, which was improved by Devanur and Vazirani (2004) using ideas in Devanur et al. (2008). (An early version of the latter paper appeared in 2002, but a flaw delayed its publication.) Garg and Kapoor (2004) presented another algorithm for this case. A central idea in this work is to reduce the equilibrium problem to a convex program. It turned out that many of the ideas in this line

of research had been developed earlier (Primak (1973, 1984), Newman and Primak (1992), Nenakov (1999)) largely as a result of the efforts of Morduck Primak, but had attracted little attention at the time. In particular Newman and Primak (1992) showed that ellipsoid methods could be used to solve the convex program in polynomial time. A more efficient algorithm using interior point methods is given by Ye (2008). Devanur and Vazirani (2004) extends these ideas to so-called spending constraint utility functions, and Kakade et al. (2004) considers models supplemented with a graph indicating which pairs of agents are allowed to trade directly.

Garg et al. (2004), Codenotti et al. (2005b), Codenotti et al. (2005a), and Jain and Varadarajan (2006) extend these methods to more general classes of utility functions including CES for certain parameters (Eaves (1985) has already developed the Cobb-Douglas case) and to economies with production, while maintaining the feature that the problem reduces to a convex program. Devanur and Kannan (2008) managed to escape this limitation, presenting two algorithms involving exhaustive search, one of which divides the price space into small cells, while the other divides the space of vectors of marginal utilities into small cells. The first runs in polynomial time when the number of goods is fixed, and the second runs in polynomial time when the number of agents is fixed and utilities are additively separable. Positive results for a fixed number of goods were also given by Deng et al. (2003). However, Chen and Teng (2011) pointed out that slightly more complicated models with a fixed number of goods are still **PPAD**-complete.

Because it seems potentially simpler than the Arrow-Debreu model, computer scientists have been attracted to the model of exchange originated by Irving Fisher (2007) in his 1892 Ph.D. thesis. (Cf. Brainard and Scarf (2000).) In this model society's endowment of goods is a vector $c \in \mathbb{R}^\ell_>$, and each agent $i = 1, \ldots, k$ has wealth $w_i > 0$ and a utility function $u_i : \mathbb{R}^\ell_\geq \to \mathbb{R}$. A **Fisher equilibrium** consists of a price vector $p \in \mathbb{R}^\ell_\geq$ and an allocation $x_1, \ldots, x_k \in \mathbb{R}^\ell_\geq$ such that

$$x_i \in \arg\max_{p \cdot x'_i \leq w_i} u_i(x'_i) \quad (i = 1, \ldots, k) \quad \text{and} \quad \sum_{i=1}^{k} x_i = c.$$

One may conceive of the Fisher model as a special case of the Arrow-Debreu model by assuming that each agent's endowment is a scalar multiple of c. Alternatively, one may imagine that the endowment of physical commodities is initially owned by an agent who only values money (this agent is not explicit in the formalism) while all other agents are initially endowed only with money and only value physical commodities. Quite remarkably, Fisher constructed a hydraulic apparatus for computing an equilibrium.

Eisenberg and Gale (1959) showed that in the case of linear utilities Fisher equilibrium is equivalent to a convex program. Devanur et al. (2008) (which was originally circulated in 2002) gave a polynomial time primal-dual algorithm for computing an equilibrium in the linear utility case. Codenotti and Varadarajan (2004) showed that the equilibrium problem could be represented by a concave program in the Leontief utility case, and gave a polynomial time algorithm, and Codenotti et al. (2005b) extended these results to more general settings. Ye (2007) considers utility functions of the

form $u_i(x_i) = \min_j u_i^j(x_{ij})$ where u_i^j is a piecewise linear concave function (Leontief utilities are a special case of this) and showed the problem reduces to a concave program which can be solved in polynomial time. Chen et al. (2009c) provided a polynomial time algorithm for hybrid linear-Leontief utility functions that are linear within certain groups of goods and Leontief across these groups. Jain et al. (2005) extended the concave programming characterization to homothetic quasi-concave utilities, and to production economies with increasing returns to scale, where the notion of equilibrium is given by marginal cost pricing.

As should be expected, because the Fisher model is a special case of the Arrow-Debreu model, polynomial time algorithms cover a somewhat broader range of cases, but maximal complexity emerges with the introduction of a little more complexity. Chen and Teng (2009) and Vazirani and Yannakakis (2011) independently showed that the problem of finding a Fisher equilibrium is **PPAD**-complete when the agents have additively separable utility with piecewise linear pieces.

This literature has many elaborate and ingenious constructions. Of these, the reduction of Codenotti et al. (2006) from bimatrix games to Leontief exchange economies is central. It is nontrivial, but not overwhelming, and it touches on a variety of important ideas and methods, so it seems appropriate to present it in the remainder of this section.

The reduction passes through the linear complementarity problem, which is itself of great interest. Let F be an $\ell \times \ell$ matrix. The **linear complementarity problem** (LCP) is

$$F\beta + \alpha = \mathbf{1}_\ell, \quad \beta^T\alpha = 0, \quad \alpha, \beta \geq \mathbf{0}_\ell.$$

The **trivial solution** is $(\alpha, \beta) = (\mathbf{1}_\ell, \mathbf{0}_\ell)$, and the "problem" is understood to be to find a different solution. This problem occurs in many application domains, and has been very extensively studied. (Cf. Murty (1988) and Cottle et al. (1992).)

We first explain how a Nash equilibrium of a bimatrix game can be understood as a solution of a special type of LCP. Let A and B be $m \times n$ payoff matrices for the row and column player respectively. We assume that the entries of these matrices are positive; of course this is without loss of generality. Let $\ell = m + n$, and set

$$F = \begin{pmatrix} 0 & A \\ B^T & 0 \end{pmatrix}. \qquad (*)$$

Suppose that (α, β) is a nontrivial solution of the LCP. Let $\alpha = (\alpha^m, \alpha^n)$ where $\alpha^m = (\alpha_1, \ldots, \alpha_m)$ and $\alpha^n = (\alpha_{m+1}, \ldots, \alpha_\ell)$, and in the same way let $\beta = (\beta^m, \beta^n)$. Since $F\beta = (A\beta^n, B^T\beta^m)$, if $\beta^m = \mathbf{0}_m$, then $\alpha^n = \mathbf{1}_n$, which implies that $\beta^n = 0$. Symmetrically, if $\beta^n = 0$, then $\beta^m = 0$. That is, β^m and β^n are both nonzero whenever (α, β) is a nontrivial solution, so we can divide each by the sum of its components, obtaining $\sigma = \beta^m/\|\beta^m\|_1$ and $\tau = \beta^n/\|\beta^n\|_1$. These constitute a Nash equilibrium of the bimatrix game (A, B) because the complementarity condition $\beta^T(\mathbf{1}_\ell - F\beta) = 0$ asserts, in effect, that each pure strategy either maximizes expected payoff, given the mixed strategy of the other agent, or is assigned no probability.

This process can be reversed: given a Nash equilibrium (σ, τ) of the bimatrix game (A, B), we can pass from σ to β^m by dividing by the second agent's equilibrium expected utility, pass from τ to β^n by dividing by the first agent's equilibrium expected utility, and then set $\beta = (\beta^m, \beta^n)$ and $\alpha = \mathbf{1}_\ell - F\beta$. Because (σ, τ) is a Nash equilibrium, (α, β) is a nontrivial solution of the LCP. Thus the Nash equilibria of the associated two person game are in an easily computed one-to-one correspondence with the nontrivial solutions of the LCP.

We now assume only that all the entries of F are nonnegative, and that for each j there is some i with $f_{ij} > 0$. The **pairing Arrow-Debreu model** is the special case of the exchange economy defined above in which there are the same number of goods and agents and each agent j is endowed with one unit of good j and none of the other goods. We interpret F as encoding a pairing Arrow-Debreu model in which each agent j has a Leontief utility function in which f_{ij} is the amount of good i that agent j requires in order to produce one unit of utility:

$$u_j(x) = \min_{i:f_{ij}>0} \frac{x_i}{f_{ij}}.$$

Consider a quasi-equilibrium of such an economy with price vector p. Suppose, for some agent j, that there is some i with $p_i > 0$ and $f_{ij} > 0$. Then none of agent j's spending is wasteful, in the sense that spending a bit less on some good with a positive price would not decrease her utility, because otherwise there would be a violation of (b): expenditure could be reduced while actually increasing utility. In addition, increasing agent j's utility requires increased spending, so she is actually maximizing utility. If, on the other hand, $p_i = 0$ for all i such that $f_{ij} > 0$, then of course agent j cannot be maximizing utility. If her expenditure was positive, it could be decreased while increasing utility, contrary to (b), so her expenditure must be zero. In this case there is no wasteful expenditure because there is no expenditure. Since her income is equal to her expenditure, her income is zero, which is to say that $p_j = 0$. We have shown that there is no wasteful spending in a quasi-equilibrium, and a quasi-equilibrium fails to be a Walrasian equilibrium if and only if there is some j with $p_j = 0$ and $p_i = 0$ for all i such that $f_{ij} > 0$.

Codenotti et al. (2006) showed that the problem of deciding whether a pairing Arrow-Debreu economy with Leontief utilities has a Walrasian equilibrium is **NP**-complete. They gave the following simple example with no Walrasian equilibria:

$$F = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix}.$$

Suppose that p is a quasi-equilibrium price vector. If $p_3 > 0$, then agent 3 would be maximizing utility, so she would be consuming twice as much of good 2 as good 3. But in the last paragraph we saw that all expenditure is due to agents with positive income who are spending efficiently, so agent 3 would be the only agent consuming good three, and her consumption bundle would be infeasible. Therefore

$p_3 = 0$. Similarly, it cannot be the case that p_1 and p_2 are both positive, because then both agent 1 and agent 2 would be maximizing utility, and the total demand for good 1 would exceed the total supply. Thus p is proportional to either $(1, 0, 0)$ or $(0, 1, 0)$, and in either case there is an agent with no income who values only goods that are free.

We now develop a useful characterization of the vectors of utilities resulting from quasi-equilibrium allocations. Suppose that p and $x_1, \ldots, x_\ell$ is a quasi-equilibrium of our economy. For each j let $\beta_j = u_j(x_j)$, let $\beta = (\beta_1, \ldots, \beta_\ell)$, and let B be the $\ell \times \ell$ diagonal matrix whose diagonal entries are the components of β. Note that $\beta \in \mathbb{R}^\ell_\geq$, and that β has some positive components because some prices are positive, and the corresponding agents have positive incomes. We claim that

$$F\beta \leq \mathbf{1}_\ell, \quad p^T(\mathbf{1}_\ell - F\beta) = 0, \quad (FB)^T p = p. \tag{$**$}$$

Here the i^{th} entry of $F\beta$ is the total amount of good i used in the minimal allocation supporting the utility vector β. The first condition states that the aggregate bundle required to produce the agents' utilities does not exceed supply. The second states that if the price of a good is positive, then its supply is exhausted by the production of the utilities, which is the case because, as we saw above, no expenditure in the quasi-equilibrium allocation (which does exhaust the supply) is wasteful. The third condition states that each agent j's income p_j is equal to the expenditure $\sum_i f_{ij}\beta_j p_i$ on the minimal bundle that attains utility level β_j, and this also follows from the discussion above.

Conversely, suppose that $\beta, p \in \mathbb{R}^\ell_\geq$ are nonzero vectors, B is the $\ell \times \ell$ diagonal matrix whose diagonal entries are the components of β, and $(**)$ holds. Form an allocation $x_1, \ldots, x_\ell$ by first assigning each agent j the minimal bundle attaining utility β_j, then distributing the remainder of the goods in excess supply in any way that does not create positive utility for any agent with zero income, to avoid creating violations of (b). Then p and $x_1, \ldots, x_\ell$ are a quasi-equilibrium: the second condition in $(**)$ implies that the price of any good in excess supply is zero, so (a) follows from the third condition, (b) follows because a bundle giving agent j a higher utility than x_j has at least as much of all goods of positive value as x_j, and (c) is true by construction. We have shown that a nonzero $\beta \in \mathbb{R}^\ell_\geq$ is the vector of utilities resulting from a quasi-equilibrium allocation if and only if there is a nonzero $p \in \mathbb{R}^\ell_\geq$ such that $(**)$ holds.

We will now show that if every quasi-equilibrium of the pairing Arrow-Debreu model given by F is a Walrasian equilibrium, then for a nonzero $\beta \in \mathbb{R}^\ell_\geq$ the following are equivalent:

- $(\mathbf{1}_\ell - F\beta, \beta)$ is a nontrivial solution of the LCP;
- There is a nonzero $p \in \mathbb{R}^\ell_\geq$ such that $(**)$ holds.

Fix a nonzero $\beta \in \mathbb{R}^\ell_\geq$ and let $\alpha = \mathbf{1}_\ell - F\beta$.

First suppose that $(**)$ holds. Then (α, β) is a nontrivial solution of the LCP if and only if $\beta^T\alpha = 0$. This is the case when (β, p) corresponds to a Walrasian equilibrium because if $\alpha_j > 0$, then $0 = p_j = \sum_i f_{ij}\beta_j p_i$, and $\beta_j = 0$ follows from this because there is some i with $f_{ij} > 0$ and $p_i > 0$.

Now suppose that (α, β) is a nontrivial solution of the LCP. Let $P = \{\, j : \beta_j > 0 \,\}$. Since the solution is nontrivial, $P \neq \emptyset$. We adopt the following notational conventions. If v is an ℓ-vector, then v_P is the element of $\mathbb{R}^P$ whose components are the v_i for $i \in P$. For a vector $v_P \in \mathbb{R}^P$, $(v_P, 0)$ denotes the element of $\mathbb{R}^\ell$ whose components corresponding to indices in P are given by v_P and whose other components are zero. If M is an $\ell \times \ell$ matrix, then M_P is the submatrix obtained by eliminating rows and columns corresponding to indices outside of P. Note that passage to M_P commutes with taking the transpose: $(M_P)^T = (M^T)_P$.

As above, let B be the $\ell \times \ell$ diagonal matrix whose diagonal entries are given by β. Note that the (i, j)-entry of FB is $f_{ij}\beta_j$, and if $i, j \in P$, then this is also the (i, j)-entry of $F_P B_P$, so $(FB)_P = F_P B_P$. For all i we have $\sum_{j=0}^{\ell} f_{ij}\beta_j = \sum_{j \in P} f_{ij}\beta_j$, and if $i \in P$, then this is the sum of the entries of the row of $(FB)_P = F_P B_P$ corresponding to i, which is one because $\beta_i > 0$ and consequently $\alpha_i = 0$. Thus $(F_P B_P)^T$ is column-stochastic, and can be interpreted as the transition matrix of a finite state Markov process. Let p_P be an invariant measure of this process: $p_P \in \mathbb{R}^P$ is a nonnegative vector whose components sum to one with $(F_P B_P)^T p_P = p_P$. (The existence of such a p_P follows from Brouwer's fixed point theorem applied to the function $q \mapsto (F_P B_P)^T q$.) Let $p = (p_P, 0)$.

For any matrix M and vector v, if the i^{th} row of M vanishes whenever $i \notin P$, then $Mv = ((Mv)_P, 0)$, and if $v = (v_P, 0)$, then $(Mv)_P = M_P v_P$. If $j \notin P$, then the j^{th} column of FB vanishes, so

$$(FB)^T p = (((FB)^T p)_P, 0) = ((FB)_P^T p_P, 0) = ((F_P B_P)^T p_P, 0) = (p_P, 0) = p.$$

In addition we have

$$p^T(\mathbf{1}_\ell - F\beta) = p_P^T(\mathbf{1}_\ell - F\beta)_P = p_P^T \alpha_P = 0$$

because $\beta^T \alpha = 0$ implies that $\alpha_P = 0$. Thus $(**)$ holds.

A variety of conditions on F imply that all quasi-equilibria are Walrasian equilibria. In the one of interest to us, called the **two group case**, the agents are partitioned into two nonempty sets, with m and n elements respectively, so that $m + n = \ell$, Each agent needs all the goods provided by the other group and none of the goods provided by her own group, so there are $m \times n$ matrices A and B with positive entries such that (after reindexing of rows and columns) $(*)$ holds. If this is the case, (β, p) satisfies $(**)$, and the prices of all the goods in one group vanish, then the expenditures of all the agents in the other group are zero, which implies that their incomes are all zero, i.e., the prices of all goods in the other group are zero. In this way we see that in the two group case a quasi-equilibrium must have positive prices for goods in both groups, so for each j there is some i with $f_{ij} >$ and $p_i > 0$, and consequently the quasi-equilibrium is a Walrasian equilibrium.

We have shown that there are one-to-one correspondences between the set of Nash equilibria of the game (A, B), the set of nontrivial solutions of the LCP, the set of $\beta \in \mathbb{R}^\ell_{\geq}$ for which there is some $p \in \mathbb{R}^\ell_{\geq}$ such that $(**)$ holds, and the set of Walrasian equilibrium utility vectors of the pairing Arrow-Debreu model in the two group case given by A and B. To complete the description of the reduction from 2-NASH to the problem of finding a Walrasian equilibrium in the two group case we point out

that the passage from an instance of 2-NASH to the associated economy is trivial, and that obvious fast algorithms pass from a Walrasian equilibrium to the resulting vector of utilities, then the solution of the LCP, and finally the associated Nash equilibrium of the given game.

16 Concluding Remarks

The theory of **PPAD** is already a tremendous success, unifying our understanding of the computational complexity of a host of problems. Both for games and economies, **PPAD**-completeness emerges quite quickly as one passes beyond the simplest and least general models. This guides our computational aspirations and strategies in many ways. The area continues to be quite active, and there will likely be many more interesting findings.

Many mysteries remain. The literature on computation of Walrasian equilibrium suggests a strong relationship between polynomial time computability and the ability to formulate the problem as a matter of convex optimization. This relationship is less well substantiated in connection with games, and in any event is rather difficult to formulate in a precise manner.

Goldberg et al. (2011) point out that several homotopy algorithms in the literature can actually be induced to solve OTHER END OF THE LINE, which is an **FPSPACE**-complete problem, when they are applied to 3D BROUWER(the three dimensional version of 2D BROUWER). (In this connection it is interesting to note that unlike the reduction to 3D BROUWER given by Papadimitriou (1994a) in order to show that this problem is **PPAD**-complete, the reduction of END OF THE LINE to 2D BROUWER in Section 13 is *not* a reduction of OTHER END OF THE LINE.) Such algorithms include the tracing procedure of Harsanyi (1975) and Harsanyi and Selten (1988), the Herings and van den Elzen (2002) and Herings and Peeters (2001) algorithms (which use the same homotopy) and the van den Elzen and Talman (1991, 1999) algorithm. There seems to be every reason to expect that this principle will extend to smooth homotopies and other such algorithms. On the other hand, alternatives to homotopy such as grid search are neither plentiful nor attractive.

At the same time the situation is certainly not as gloomy as a naive reading of the main results here might suggest. Large scale instances of **NP**-complete problems seem to be completely intractable, but computation of fixed points of highly nonlinear functions in high dimensions, and the more general application of homotopy methods to large systems of equations, is a practical reality, even if we now know that algorithms that do this cannot come with a guarantee of both speed and reliability. The central concepts of computational complexity are in some ways quite crude, which is why they are so powerful and far reaching, but by the same token they are blind to more delicate distinctions when they categorize many things as "the same," relative to some equivalence relation. A natural geometric intuition is that computation of a fixed point should be rather straightforward for functions that are "close to linear." For functions whose graphs are highly irregular, on the other hand, it should be easier to hide a fixed point from an algorithm, but one may also imagine that there is some sense in which

fixed points for such functions are typically plentiful. These thoughts resonate with the success of many algorithms on "typical" instances of the problems they are designed to solve, but we are at present very far from having definitions that give precise expressions of such intuitions.

In addition, many problems in social science give rise to restricted classes of games. For each such class the associated equilibrium computation may be **PPAD**-complete, but there are also types of games for which the problem is not intractable, with zero sum games being the most obvious example. As we mentioned in Section 12, there is work of this sort related to graphical games. Here we also call attention to Daskalakis and Papadimitriou (2007), which studies anonymous games, and Brandt et al. (2009) and Goldberg et al. (2010), which initiate the study of the computational complexity of equilibrium in games such as tournaments, for which there is a vast literature in economics.

Clearly the theory of **PPAD**-completeness has important consequences for research methodologies in economics that depend on the computation of fixed points, and on the computation of Walrasian equilibria in particular. In economic modelling it is common to consider models with a small number of agents, say a capitalist and a worker, who are thought of as proxies representing a large number of identical agents. In such models the number of goods might be very large, especially in a dynamic model with incremental resolution of uncertainty represented by a large set of states of nature. Thus the theory of **PPAD**-completeness when the number of agents is fixed is quite important, but very few of the results described in the last section address this issue, and those that do point to algorithms that seem unattractive, even if they run in polynomial time.

Economic science has hardly begun the process of assimilating the main findings described here, and the consequences of complexity in its larger sense. We have suggested how, in the particular setting of financial markets, an appreciation of complexity leads one to expect certain phenomena, and the resulting point of view goes some distance toward reconciling the world views of the proponents of the efficient market hypothesis and the practitioners of behavioral finance. There seem to be numerous opportunities for empirical research in this direction. There is an obvious desire for a theoretical framework that captures the salient features of complex systems of markets, but this is of course a major modelling challenge.

Appendix: Graphs and Polytopal Complexes

This appendix presents some technical background that is assumed throughout. The concepts defined below, from graph theory and combinatoric geometry, are standard and basic in mathematics and computer science. For readers with sufficient background it will suffice to skim the boldfaced terms, returning later to seek clarification if the need arises.

The cardinality of a finite set S is denoted by $|S|$.

A (simple undirected) **graph** is a pair $G = (V, E)$ consisting of a finite set V of **vertices** and a set E of **edges** whose elements are two element subsets of V. The qualifier 'simple' refers to two

properties that are implicit in this definition: a) there is at most one edge between any pair of vertices; b) E does not contain any "self loops" connecting a vertex to itself. The elements of an edge are its **endpoints**. The set of **neighbors** of $v \in V$ is $\mathcal{N}(v) = \{\, v' \in V : \{v, v'\} \in E \,\}$, and the **degree** of v is $|\mathcal{N}(v)|$. A vertex of degree zero is an **isolated vertex** and a vertex of degree one is a **leaf**.

A **walk** in G is a sequence $v_0, \ldots, v_k$ of vertices such that $\{v_{i-1}, v_i\} \in E$ for all $i = 1, \ldots, k$. When there is such a walk we say that v_k is **reachable** from v_0. We regard a sequence v_0 with a single element as a walk, so every vertex is reachable from itself.

We say that $G' = (V', E')$ is a **subgraph** of the graph $G = (V, E)$ if $V' \subset V$, $E' \subset E$, and the endpoints of the elements of E' are contained in V'. For example, if V' is the set of all vertices that are reachable from v_0, and E' is the set of all edges in E whose endpoints are in V', then G' is the **connected component** of v_0 in G. We say that G is **connected** if it has a single connected component. The subgraph G' is a **clique** if any two vertices in V' are the endpoints of an edge in E', so that E' is the set of all two element subsets of V'. The subgraph is a **cycle** of G if it is connected and each element of V' is an endpoint of exactly two elements of E'. A **Hamiltonian cycle** is a cycle $G' = (V', E')$ with $V' = V$. That is, starting at any vertex in V and following the edges in the cycle results in visiting each other vertex in V once and then returning to the starting point.

A (simple) **directed graph** is a pair $G = (V, A)$ consisting of a finite set V of **vertices** and a finite set A of ordered pairs of distinct elements of V. An element of A is called an **arrow**, its first component is its **tail**, and its second component is its **head**. As above, our definition requires that the tail and head of an arrow are distinct, and that for any distinct v, v' there is at most one arrow whose tail is v and whose head is v', but it can happen that (v, v') and (v', v) are both elements of A. If $(v, v') \in A$, then v' is a **direct successor** of v and v is a **direct predecessor** of v'. The **indegree** of a vertex is the number of direct predecessors it has, and the **outdegree** is the number of direct sucessors it has. A vertex is a **source** if it has indegree zero, and it is a **sink** if it has outdegree zero.

A **walk** in a directed graph $G = (V, A)$ is a sequence $v_0, \ldots, v_k$ in V such that v_i is a direct successor of v_{i-1} for all $i = 1, \ldots, k$; in this circumstance we say that v_k is a **successor** of v_0 and v_0 is a **predecessor** of v_k. The graph is **acyclic** if no vertex is a predecessor of itself.

Turning to geometric concepts, fix a Euclidean space $\mathbb{R}^d$. The **affine hull** of a set $S \subset \mathbb{R}^d$ is the set of all **affine combinations** $\alpha_1 x_1 + \cdots + \alpha_k x_k$ where $x_1, \ldots, x_k \in S$ and $\alpha_1 + \cdots + \alpha_k = 1$. This is always a set of the form $x + L$ where L is a linear subspace, and the **dimension** of the affine hull is the dimension of L. The set S is **affinely independent** if the affine hull of any proper subset of S is a proper subset of the affine hull of S.

A **polytope** in $\mathbb{R}^d$ is a set P that is the convex hull of a finite set $S \subset \mathbb{R}^d$. If S is affinely independent, then P is a **simplex**. The **dimension** of P is the dimension of its affine hull; if this dimension is i, then we say that P is an i**-polytope** or an i**-simplex**. If $\nu \in \mathbb{R}^d$, $\alpha \in \mathbb{R}$, and $P \subset \{\, x \in \mathbb{R}^d : \langle \nu, x \rangle \geq \alpha \,\}$, then $\{\, x \in \mathbb{R}^d : \langle \nu, x \rangle = \alpha \,\}$ is a **bounding hyperplane** of P and $\{\, x \in P : \langle \nu, x \rangle = \alpha \,\}$ is a **face** of P. The empty set is always a face of P, and since we allow $\nu = 0$,

P is always a face of itself. A **facet** of P is a face whose dimension is one less than the dimension of P. (We will not define the dimension of the empty set, but in any event the empty set is not considered to be a facet of a zero dimensional polytope.) It is well known that P is a polytope if and only if P is a compact set that is the intersection of finitely many half spaces; it follows that the faces of P are themselves polytopes. The **diameter** of a polytope is the maximum distance between any two of its points.

A **polytopal complex** is a finite set $\mathcal{K}$ of polytopes such that:

- $F \in \mathcal{K}$ whenever $P \in \mathcal{K}$ and F is a face of P;
- For all $P, P' \in \mathcal{K}$, $P \cap P'$ is a face of both P and P';
- $\emptyset \in \mathcal{K}$.

(The last condition merely insures that $\mathcal{K} \neq \emptyset$.) If all of the elements of $\mathcal{K}$ are simplices, then $\mathcal{K}$ is a **simplicial complex**. Let $|\mathcal{K}| = \bigcup_{P \in \mathcal{K}} P$. We say that $\mathcal{K}$ is a **polytopal subdivision** of $|\mathcal{K}|$, and if $\mathcal{K}$ is a simplicial complex, then $\mathcal{K}$ is a **simplicial subdivision** or **triangulation** of $|\mathcal{K}|$. The **dimension** of $\mathcal{K}$ is the maximal dimension of its elements, and its **mesh** is the maximal diameter of its elements. For $i = 0, \ldots, d$, the i**-skeleton** of $\mathcal{K}$ is the set of all elements of $\mathcal{K}$ of dimension $\leq i$. The elements of the 0-skeleton of $\mathcal{K}$ are its **vertices**; technically these are singleton subsets of $\mathbb{R}^d$, but usually we treat them as points.

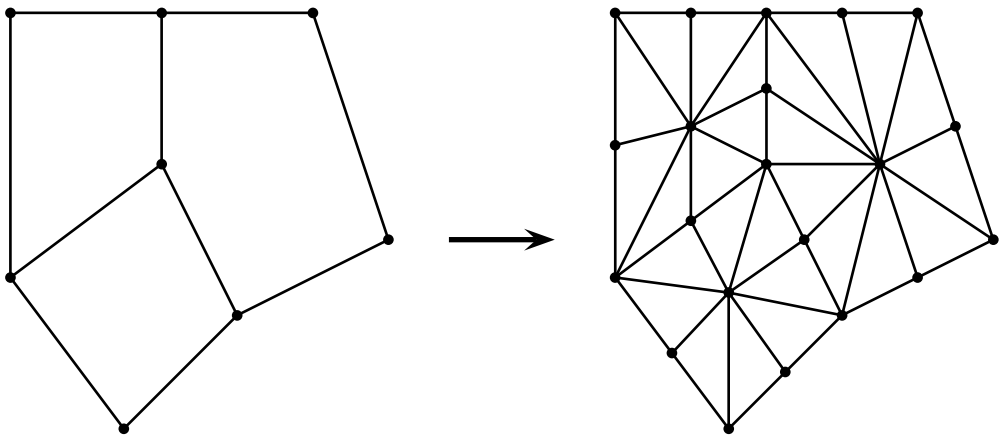

Figure 8

For each $P \in \mathcal{K}$ of positive dimension let β_P be an element of the relative interior of P. (That is, β_P is not an element of any proper face of P.) We define the **barycentric subdivision** of $\mathcal{K}$ induced by $\{\beta_P\}$ using induction on dimension. The barycentric subdivision of the zero dimensional skeleton of $\mathcal{K}$ is just the zero dimensional skeleton of $\mathcal{K}$ itself. Suppose that the barycentric subdivision of the

$(i-1)$-skeleton of $\mathcal{K}$ has already been defined. Then the barycentric subdivision of the i-skeleton of $\mathcal{K}$ is the set of all polytopes that are either in the barycentric subdivision of the $(i-1)$-skeleton or are formed by taking the convex hull of some β_P, where P is i-dimensional, and an element of the barycentric subdivision of the $(i-1)$-skeleton that is contained in a face of P. As Figure 8 suggests, easy induction arguments prove that the barycentric subdivision of $\mathcal{K}$ is a simplicial complex, and that it is a subdivision of $|\mathcal{K}|$.

A straightforward but rather tedious calculation (cf. pp. 41–2 of Dold (1980)) shows that if $\mathcal{K}$ is a simplicial complex and each β_P is the arithmetic mean of the vertices of P, then the mesh of the barycentric subdivision is not greater than $\dim \mathcal{K}/(\dim \mathcal{K}+1)$ times the mesh of $\mathcal{K}$. Thus repeated barycentric subdivision can be used to produce simplicial subdivisions of $|\mathcal{K}|$ of arbitrarily small mesh.

References

Abbott, T., Kane, D., and Valiant, P. (2005). On the complexity of two-player win-lose games. In *Proceedings of the 46th Annual IEEE Symposium on the Foundations of Computer Science*, pages 113–122.

Agrawal, M., Kayal, N., and Saxena, N. (2004). PRIMES is in p. *Annals of Mathematics*, 160:781–793.

Aldous, D. (1983). Minimization algorithms and random walk on the d-cube. *Annals of Probability*, 11:403–413.

Alon, N. and Boppana, R. (1987). The monotone circuit complexity of Boolean functions. *Combinatorica*, 7:1–22.

Andersson, D. and Miltersen, P. B. (2009). The complexity of solving stochastic games on graphs. In *Proceedings of the 20th International Symposium on Algorithms and Computation*, pages 112–121, Berlin, Heidelberg. Springer-Verlag.

Andreev, A. E. (1985). On a method for obtaining lower bounds for the complexity of individual monotone functions. *Soviet Mathematics Doklady*, 31:530–534.

Arora, S. and Boaz, B. (2007). *Computational Complexity: A Modern Approach*. Cambridge University Press, Cambridge.

Arthur, B. (2006). Out-of-equilibrium economics and agent-based modelling. In Tesfatsion, L. and Judd, K., editors, *Handbook of Computational Economics: Agent Based Computational Economics, Vol. 2*, pages 53–66. North Holland, Amsterdam.

Bárány, I., Vempala, S., and Vetta, A. (2007). Nash equilibria in random games. *Random Structures and Algorithms*, 31:391–405.

Barber, B. M., Lee, Y.-T., Liu, Y.-J., and Odean, T. (2009). Just how much do individual investors lose by trading? *Review of Financial Studies*, 22:609–632.

Barber, B. M., Lee, Y.-T., Liu, Y.-J., and Odean, T. (2011). The cross-section of speculator skill: Evidence from taiwan. Working paper available from SSRN.

Barber, B. M. and Odean, T. (2001). Boys will be boys: Gender, overconfidence, and common stock investment. *Quarterly Journal of Economics*, 116:261–292.

Barone, E. (1935). The ministry of production in the collectivist state. In von Hayek, F., editor, *Collectivist Economic Planning*, pages 245–290. George Routledge and Sons, Ltd., London.

Bhat, N. A. R. and Leyton-Brown, K. (2004). Computing Nash equilibria in action-graph games. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 35–42, Arlington, Virginia, United States. AUAI Press.

Bosse, H., Byrka, J., and Markakis, E. (2007). New algorithms for approximate Nash equilibria in bimatrix games. In *Proceedings of the 3rd International Workshop on Internet and Network Economics*, pages 17–29.

Brainard, W. and Scarf, H. (2000). How to compute equilibrium prices in 1891. Cowles Foundation Discussion Papers 1272, Cowles Foundation, Yale University.

Brandt, F., Fischer, F., Harrenstein, P., and Shoham, Y. (2009). Ranking games. *Artificial Intelligence*, pages 221–239.

Breist, P., Goldberg, P. W., and Röglin (2008). Approximate equilibria in games with few players. *CoRR*, abs/0804.4524.

Brouwer, L. E. J. (1912). Uber Abbildung von Mannigfaltikeiten. *Mathematiche Annalen*, 71:97–115.

Buffett, W. E. (1984). The superinvestors of Graham-and-Doddsville. *Hermes: the Columbia Business School Magazine*, pages 4–15.

Chen, X., Dai, D., Du, Y., and Teng, S.-H. (2009a). Settling the complexity of Arrow-Debreu equilibria in markets with additively separable utilities. In *Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 273–282.

Chen, X. and Deng, X. (2006a). On the complexity of 2D discrete fixed point problem. In *Proceedings of the 33th International Colloquium on Automata, Languages, and Programming*, pages 489–500.

Chen, X. and Deng, X. (2006b). Settling the complexity of two-player Nash equilibrium. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 261–272.

Chen, X., Deng, X., and Teng, S.-H. (2006a). Computing Nash equilibria: Approximation and smoothed complexity. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 603–612.

Chen, X., Deng, X., and Teng, S.-H. (2006b). Sparse games are hard. In *Proceedings of the 2nd Workshop on Internet and Network Economics*, pages 262–273.

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

Chen, X., Huang, L.-S., and Teng, S.-H. (2009c). Market equilibria with hybrid linear-Leontief utilities. *Theoretical Computer Science*, 410:1573–1580.

Chen, X. and Teng, S.-H. (2007). Paths beyond local search: A tight bound for randomized fixed-point computation. In *Proceedings of the 48th Annual IIIE Symposium on Foundations of Computer Science*, pages 124–134.

Chen, X. and Teng, S.-H. (2009). Spending is not easier than trading: On the computational equivalence of Fisher and Arrow-Debreu equilibria. In *Proceedings of the 20th International Symposium on Algorithms and Computation*, pages 647–656.

Chen, X. and Teng, S.-H. (2011). A complexity view of markets with social influence. In *Proceedings of Innovation in Computer Science 2011*, pages 141–154.

Codenotti, B., McCune, B., Penumatcha, S., and Varadarajan, K. (2005a). Market equilibrium for CES exchange economies: Existence, multiplicity, and computation. In *Proceedings of the 25th Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 505–516.

Codenotti, B., Pemmaraju, S., and Varadarajan, K. (2005b). On the polynomial time computation of equilibria for certain exchange economies. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 72–81.

Codenotti, B., Saberi, A., Varadarajan, K., and Yinyu, Y. (2006). Leontief economies encode nonzero sum two-player games. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 659–667. ACM Press.

Codenotti, B. and Varadarajan, K. (2004). Efficient computation of equilibrium prices for markets with Leontief utilities. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming*, pages 371–382.

Condon, A. (1992). The complexity of stochastic games. *Information and Computation*, 96:203–224.

Conitzer, V. and Sandholm, T. (2003). Complexity results about Nash equilibria. *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 765–771.

Cottle, R., Pang, J.-S., and Stone, R. (1992). *The Linear Complementarity Problem*. Academic Press, Boston.

Daskalakis, C. (2009). Nash equilibria: Complexity, symmetries, and approximation. *Computer Science Review*, 3:87–100.

Daskalakis, C. (2011). On the complexity of aproximating a Nash equilibrium. *Transactions on Algorithms*. forthcoming.

Daskalakis, C., Goldberg, P., and Papadimitriou, C. (2006a). The complexity of computing a Nash equilibrium. In *Proceedings of the 38th ACM Symposium on the Theory of Computing*.

Daskalakis, C., Goldberg, P., and Papadimitriou, C. (2009a). The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 39:195–259.

Daskalakis, C., Goldberg, P. W., and Papadimitriou, C. H. (2009b). The complexity of computing a Nash equilibrium. *Communications of the ACM*, 52:89–97.

Daskalakis, C., Mehta, A., and Papadimitriou, C. (2006b). A note on approximate Nash equilibria. In *Proceedings of the 2nd Workshop on Internet and Network Economics*.

Daskalakis, C., Mehta, A., and Papadimitriou, C. (2007). Progress in approximate Nash equilibria. In *Proceedings of the 8th ACM Conference on Electronic Commerce*.

Daskalakis, C. and Papadimitriou, C. (2007). Computing equilibria in anonymous games. In *Proceedings of the 48th Annual IEEE Symposium on the Foundations of Computer Science*, pages 83–93.

Daskalakis, C. and Papadimitriou, C. (2009). On oblivious PTAS's for Nash equilibrium. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, pages 75–84.

Daskalakis, C., Schoenebeck, G., Valiant, G., and Valiant, P. (2009c). On the complexity of Nash equilibria in action-graph games. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 719–719, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

Datta, R. (2010). Finding all Nash equilibria of a finite game using polynomial algebra. *Economic Theory*, 42(1):55–96.

De Long, J. B., Shleifer, A., Summers, L. H., and Waldmann, R. J. (1990). Noise trader risk in financial markets. *Journal of Political Economy*, 98:703–738.

Debreu, G. (1974). Excess demand functions. *Journal of Mathematical Economics*, 1:15–21.

Deng, X., Papadimtriou, C., and Safra, S. (2003). On the complexity of price equilibria. *Journal of Computer and System Sciences*, 67:311–324.

Devanur, N. and Kannan, R. (2008). Market equilibria in polynomial time for fixed number of goods or agents. In *Proceedings of the 49th Annual IEEE Symposium on the Foundations of Computer Science*, pages 45–53.

Devanur, N., Papadimtriou, C., Saberi, A., and Vazirani, V. (2008). Market equilibrium via a primal-dual algorithm for a convex program. *Journal of the ACM*, 55:22:1–22.

Devanur, N. and Vazirani, V. (2004). The spending constraint model for market equilibrium: Algorithmic, existence and uniqueness results. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 519–528.

Dold, A. (1980). *Lectures on Algebraic Topology*. Springer-Verlag, New York.

Duffie, D. (2010). Presidential address: Asset price dynamics with slow moving capital. *Journal of Finance*, 65:1238–1268.

Eaves, B. C. (1985). Finite solution of pure trade markets with Cobb-Douglas utilities. In Manne, A. S., editor, *Mathematical Programming Study 23*, pages 226–239. North-Holland.

Eisenberg, E. and Gale, D. (1959). Consensus of subjective probabilities: The pari-mutuel method. *Annals of Mathematical Statistics*, 30:165–168.

Elkind, E., Goldberg, L. A., and Goldberg, P. (2006). Nash equilibria in graphical games on trees revisited. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, pages 100–109.

Esteban-Bravo, M. (2004). Computing equilibria in general equilibrium models via interior-point methods. *Computational Economics*, 23:147–171.

Etessami, K. and Yannakis, M. (2010). On the complexity of Nash equilibria and other fixed points. *SIAM Journal on Computing*, 39:2531–2597.

Fabrikant, A., Papadimitriou, C., and Talwar, K. (2004). The complexity of pure Nash equilibria. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 604–612.

Fama, E. (1970). Efficient capital markets: A review of theory and empirical work. *Journal of Finance*, 25:383–417.

Fama, E. (1991). Efficient markets II. *Journal of Finance*, 46:1575–1617.

Feder, T., Nazerzadeh, H., and Saberi, A. (2007). Approximating Nash equilibria using small-support strategies. In *Proceedings of the 8th ACM Conference on Electronic Commerce*, pages 352–354.

Fisher, I. (2007). *Mathematical Investigations in the Theory of Value and Prices*. Cosimo, New York.

Gabzewicz, J. and Vial, J. P. (1975). Oligopoly "à la Cournot" in a general equilibrium analysis. *Journal of Economic Theory*, 4:381–400.

Garg, R. and Kapoor, S. (2004). Auction algorithms for market equilibrium. In *Proceedings of the 36th Annual Symposium on Theory of Computing*, pages 511–518.

Garg, R., Kapoor, S., and Vazirani, V. (2004). An auction-based market equilibrium algorithm for the separable gross substitutability case. In *Proceedings of the 7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 128–138.

Gilboa, I. and Zemel, E. (1989). Nash and correlated equilibria: some complexity considerations. *Games and Economic Behavior*, 1(1):80–93.

Goldberg, L. A., Goldberg, P., Krysta, P., and Ventre, C. (2010). Ranking games that have competitiveness-based strategies. In *Proceedings of the 10th ACM Conference on Electronic Commerce*, pages 335–344.

Goldberg, P. (2011). A survey of PPAD-completeness for computing Nash equilibrium. In Chapman, R., editor, *Surveys in Combinatorics 2011*, pages 51–82. Cambridge University Press, Cambridge.

Goldberg, P. and Papadimitriou, C. (2006). Reducibility among equilibrium problems. In *Proceedings of 38th Annual Conference on the Theory of Computing*, pages 61–70. ACM Press.

Goldberg, P., Papadimitriou, C., and Savani, R. (2011). The complexity of the homotopy method, equilibrium selection, and Lemke-Howson solutions. In *Proceedings of the 52nd Annual IEEE Symposium on the Foundations of Computer Science*.

Harsanyi, J. (1975). The tracing procedure: a Bayesian approach to defining a solution for n-person noncooperative games. *Internation Journal of Game Theory*, 4:61–95.

Harsanyi, J. C. and Selten, R. (1988). *A General Theory of Equilibrium Selection in Games*. MIT Press, Cambridge.

Hart, O. D. (1979). Monopolistic competition in a large economy with differentiated commodities. *Review of Economic Studies*, 46:1–30.

Hart, O. D. (1980). Perfect competition and optimal product differentiation. *Journal of Economic Theory*, 22:279–313.

Hayek, F. (1945). The use of knowledge in society. *American Economic Review*, 35:519–530.

Hémon, S., Rougement, M., and Santha, M. (2008). Approximate Nash equilibria for multi-player games. In *Proceedings of the 1st Annual Symposium on Algorithmic Game Theory*, pages 267–278.

Herings, P. J.-J. and Peeters, R. (2001). A differentiable homotopy to compute Nash equilibria of n-person games. *Economic Theory*, 18:158–185.

Herings, P. J.-J. and Peeters, R. (2010). Homotopy methods to compute equilibria in game theory. *Economic Theory*, 42(1):119–156.

Herings, P. J.-J. and van den Elzen, A. (2002). Computation of the Nash equilibrium selected by the tracing procedure in n-person games. *Games and Economic Behavior*, 38:89–117.

Hirsch, M., Papadimitriou, C., and Vavasis, S. (1989). Exponential lower bounds for finding Brouwer fixed points. *Journal of Complexity*, 5:379–416.

Huang, L.-S. and Teng, S.-H. (2007). On the approximation and smoothed complexity of Leontief market equilibrium. In *Proceedings of the 1st Annual International Conference on Frontiers in Algorithmics*, pages 96–107.

Hurwicz, L. (1960). Optimality and informational efficiency in resource allocation processes. In Karlin, S. and Suppes, P., editors, *Mathematical Methods in the Social Sciences*, pages 27–46. Stanford University Press, Stanford, CA.

Hurwicz, L. (1977). On the dimensional requirements of informationally decentralized Pareto-satisfactory processes. In Arrow, K. J. and Hurwicz, L., editors, *Studies in Resource Allocation Processes*, pages 413–424. Cambridge University Press, Cambridge.

Hurwicz, L., Radner, R., and Reiter, R. (1975a). A stochastic decentralized resource allocation process: Part I. *Econometrica*, 43:187–221.

Hurwicz, L., Radner, R., and Reiter, R. (1975b). A stochastic decentralized resource allocation process: Part II. *Econometrica*, 43:363–393.

Jain, K., Mahdian, M., and Saberi, A. (2003). Approximating market equilibria. In *Proceedings of the 6th International Workshop on Approximation Algorithms*, pages 98–108.

Jain, K. and Varadarajan, K. (2006). Equilibria for economies with production: Constant-returns technologies and production planning constraints. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 688–697.

Jain, K., Vazirani, V., and Ye, Y. (2005). Market equilibria for homothetic, quasi-concave utilities and economies of scale in production. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 63–71.

Jensen, M. C. (1978). Some anomalous evidence regarding market efficiency. *Journal of Financial Economics*, 6:95–101.

Jiang, A. X. and Leyton-Brown, K. (2006). A polynomial-time algorithm for action-graph games. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, pages 679–684. AAAI Press.

Jordan, J. S. (1982). The competitive allocation process is informationally efficient uniquely. *Journal of Economic Theory*, 28:1–18.

Jordan, J. S. (1987). The informational requirement of local stability in decentralized allocation mechanisms. In Groves, T., Radner, R., and Reiter, S., editors, *Information, Incentives, and Economic Mechanisms: Essays in Honor of Leonid Hurwicz*, pages 183–212. University of Minnesota Press, Minneapolis.

Judd, K. L., Kubler, F., and Schmedders, K. (2003). Computational methods for dynamic equilibria with heterogeneous agents. In Dewatripont, M., Hansen, L. P., and Turnovsky, S. J., editors, *Advances in Economics and Econometrics: Theory and Applications, Eighth World Congress, Volume III*, pages 243–290. Cambridge University Press, Cambridge.

Kakade, S. M., Kearns, M. J., and Ortiz, L. E. (2004). Graphical economics. In *Proceedings of 17th Annual Conference on Learning Theory*, pages 17–32. Springer.

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. In *Proceedings of the 16th ACM Symposium on Theory of Computing*, STOC '84, pages 302–311, NewYork, NY, USA. ACM.

Kearns, M. J., Littman, M. L., and Singh, S. P. (2001). Graphical models for game theory. In *Proceedings of 17th Annual Conference on Uncertainty in Artificial Intelligence*, pages 253–260. Morgan Kaufmann.

Khachian, L. (1979). A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, 20:191–194.

Kintali, S., Poplawski, L. J., Rajaraman, R., Sundaram, R., and Teng, S.-H. (2009). Reducibility among fractional stability problems. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 283–292.

Klee, V. and Minty, G. (1972). How good is the simplex algorithm? In Sisha, O., editor, *Inequalities III*. Academic Press, New York.

Lange, O. (1938). On the economic theory of socialism. In *On the Economic Theory of Socialism*, pages 57–142. University of Minnesota Press, Minneapolis.

Lemke, C. E. (1965). Bimatrix equilibrium points and mathematical programming. *Management Science*, 11:681–689.

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

Lerner, A. P. (1944). *The Economics of Control*. The Macmillan Company, New York.

Levitt, S. D. and Miles, T. J. (2011). The role of skill versus luck in poker: Evidence from the world series of poker. University of Chicago Working Paper.

Lipton, R. J. and Markakis, E. (2004). Nash equilibria via polynomial equations. In *Proceedings of the 6^{th} Latin American Symposium on Theoretical Informatics*, pages 413–422.

Lipton, R. J., Markakis, E., and Mehta, A. (2003). Playing large games using simple strategies. In *Proceedings of the 4^{th} ACM Conference on Electronic Commerce*, pages 36–41. ACM Press.

Littman, M. L., Kearns, M. J., and Singh, S. P. (2001). Graphical models for game theory. In *Advances in Neural Information Processing Systems*, volume 14.

Malkiel, B. G. (2007). *A Random Walk Down Wall Street*. W.W. Norton, New York.

Mancini-Griffoli, T. and Ranaldo, A. (2011). Limits to arbitrage during the crisis: Funding liquidity constraints and covered interest parity. *SSRN Library*.

Mantel, R. (1974). On the characterization of aggregate excess demand. *Journal of Economic Theory*, 7:348–353.

Marschak, T. (1959). Centralization and decentralization in economic organizations. *Econometrica*, 27:399–430.

Mas-Colell, A. (1982). The Cournotian foundations of Walrasian equilibrium theory. In Hildenbrand, W., editor, *Advances in Economic Theory*. Cambridge University Press, New York.

Mas-Colell, A. (1983). Walrasian equilibria as limits of noncooperative equilibria. Part I: Mixed strategies. *Journal of Economic Theory*, 30:153–170.

McKelvey, R. D. and McLennan, A. (1996). The computation of equilibria in finite games. In *Handbook of Computational Economics*. Elsevier, New York.

McLennan, A. and Tourky, R. (2010). Simple complexity from imitation games. *Games and Economic Behavior*, 68:683–688.

Megiddo, N. (1988). A note on the complexity of p-matrix LCP and computing an equilibrium. Research Report RJ6439, IBM Almaden Research Center, San Jose.

Megiddo, N. and Papadimitriou, C. H. (1991). On total functions, existence theorems and computational complexity. *Theoretical Computer Science*, 81:317–324.

Mitchell, M. and Pulvino, T. (2011). Arbitrage crashes and the speed of capital. forthcoming in the *Journal of Financial Economics*.

Monderer, D. and Shapley, L. S. (1996). Potential games. *Games and Economic Behavior*, 14:124–143.

Mount, K. R. and Reiter, S. (1974). The information size of message spaces. *Journal of Economic Theory*, 8:161–192.

Mount, K. R. and Reiter, S. (1996). A lower bound on computational complexity given by revelation mechanisms. *Economic Theory*, 7:237–266.

Murty, K. G. (1988). *Linear Complementarity, Linear and Nonlinear Programming*. Heldermann, Berlin. Sigma Series in Applied Mathematics 3.

Nash, J. (1951). Non-cooperative games. *Annals of Mathematics*, 54:286–295.

Nenakov, E. (1999). On one method for finding the equilibrium state. *Journal of Mathematical Sciences*, 97:3939–3944.

Newman, D. and Primak, M. (1992). Complexity of circumscribed and inscribed ellipsoid methods for solving equilibrium economical models. *Applied Mathematics and Computation*, 52:223–231.

Nisan, N., Roughgarden, T., Tardos, E., and Varizani, V., editors (2007). *Algorithmic Game Theory*. Cambridge University Press, New York.

Nisan, N. and Segal, I. (2001). The communication complexity of efficient allocation problems. In *DIMACS Workshop on Computational Issues in Game Theory and Mechanism Design*, pages 505–516.

Novshek, W. and Sonnenschein, H. (1978). Cournot and Walras equilibria. *Journal of Economic Theory*, 19:223–266.

Novshek, W. and Sonnenschein, H. (1980). Small efficient scale as a foundation for Walrasian equilibrium. *Journal of Economic Theory*, 22:243–256.

Novshek, W. and Sonnenschein, H. (1983). Walrasian equilibria as limites of noncooperative equilibria. Part II: Pure strategies. *Journal of Economic Theory*, 30:171–187.

Odean, T. (1999). Do investors trade too much? *American Economic Review*, 89:1279–1298.

Osana, H. (1978). On the information size of message spaces for resource allocation problems. *Journal of Economic Theory*, 17:66–78.

Papadimitriou, C. (1994a). On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Science*, 48:498–532.

Papadimitriou, C. (2007). The complexity of finding Nash equilibria. In Nisan, N., Roughgarden, T., Tardos, E., and Varizani, V., editors, *Algorithmic Game Theory*, pages 29–52. Cambridge University Press, New York.

Papadimitriou, C. H. (1994b). *Computational Complexity*. Addison Wesley Longman, New York.

Papadimitriou, C. H. (2001). Algorithms, games and the internet. In *Proceedings of the 34th Annual ACM Symposium on the Theory of Computing*, pages 749–253.

Pesendorfer, W. and Swinkels, J. M. (1997). The loser's curse and information aggregation in common value auctions. *Econometrica*, 65:1247–1281.

Porter, R. W., Nudelman, E., and Shoham, Y. (2008). Simple search methods for finding a Nash equilibrium. *Games and Economic Behavior*, 63:642–662.

Pratt, V. R. (1975). Every prime has a succinct certificate. *SIAM Journal on Computing*, 4:214–220.

Primak, M. E. (1973). A computational process of search for equilibrium points. *Cybernetics and Systems Analysis*, 9:106–113.

Primak, M. E. (1984). Algorithm for the solution of the linear exchange model and the Arrow-Debreu model. *Kibernetica*, 5.

Razborov, A. A. (1985a). A lower bound on the monotone network complexity of the logical permanent. *Russian Mathematical Notes*, 37:485–493.

Razborov, A. A. (1985b). Lower bounds on the monotone complexity of some Boolean functions. *Mathematics of the USSR, Doklady*, 31:354–357.

Roberts, K. (1980). The limit points of monopolistic competition. *Journal of Economic Theory*, 22:256–279.

Rosenthal, R. W. (1973). A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2:65–67.

Roughgarden, T. (2010a). Algorithmic game theory. *Communications of the ACM*, 53:78–86.

Roughgarden, T. (2010b). Computing equilibria: A computational complexity perspective. *Economic Theory*, 42:191–236.

Saari, D. G. (1985). Iterative price mechanisms. *Econometrica*, 53:1117–1133.

Saari, D. G. and Simon, C. P. (1978). Effective price mechanisms. *Econometrica*, 46:1097–1125.

Sato, F. (1981). On the informational size of message spaces for resource allocation processes in economies with public goods. *Journal of Economic Theory*, 24:48–69.

Savani, R. and von Stengel, B. (2006). Hard-to-solve bimatrix games. *Econometrica*, 74:397–429.

Scarf, H. (1973). *The computation of economic equilibria*. Yale University Press, New Haven, Conn. With the collaboration of Terje Hansen, Cowles Foundation Monograph, No. 24.

Scarf, H. E. (1967). The approximation of fixed points of a continuous mapping. *SIAM Journal of Applied Mathematics*, 15:1328–1343.

Schoenebeck, G. and Vadhan, S. (2006). The computational complexity of Nash equilibria in concisely represented games. In *Proceedings of 7th Annual Conference on Electronic Commerce*, pages 270–279. ACM Press.

Schumpeter, J. A. (1976). *Capitalism, Socialism and Democracy*. Routledge, London and New York, fifth edition.

Shannon, C. E. (1949). The synthesis of two-terminal switching circuits. *Bell System Technical Journal*, 28:59–98.

Shapley, L. S. (1976). Non-cooperative general exchange. In Lin, S., editor, *Theory of Measurement of Economic Externalities*. Academic Press, New York.

Shapley, L. S. and Shubik, M. (1977). Trade using one commodity as a means of payment. *Journal of Political Economy*, 85:937–968.

Shiller, R. J. (2004). Radical financial innovation. In Shishinski, E., Strom, R. J., and Baumol, W. J., editors, *Entrepreneurship, Innovation and the Growth Mechanism of the Free Market Economy, in Honor of William Baumol*, pages 306–323. Princeton University Press, Princeton.

Shiller, R. J. (2005). *Irrational Exuberance*. Princeton University Press, Princeton.

Shleifer, A. and Vishny, R. W. (1997). The limits of arbitrage. *Journal of Finance*, 52:35–55.

Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantm computer. *SIAM Journal on Computing*, 26:1484–1509.

Shubik, M. (1973). Commodity money, oligopoly, credit, and bankruptcy in a general equilibrium model. *Western Economic Journal*, 11:24–38.

Sonnenschein, H. (1973). Do Walras' identity and continuity characterize the class of community excess demand functions? *Journal of Economic Theory*, 6:345–354.

Sonnenschein, H. F. (1972). Market excess demand functions. *Econometrica*, 40:549–563.

Sperner, E. (1928). Neuer beweis für die invarianz der dimensionszahl und des gebietes. *Abh. Math. Sem. Univ. Hamburg*, 6:265–272.

Spielman, D. and Teng, S.-H. (2004). Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of Association of Computing Machinery*, 51:385–463.

Spirakis, P. G. and Tsaknakis, H. (2007). An optimization approach for approximate Nash equilibria. In *Proceedings of the 3rd Workshop on Internet and Network Economics*.

Todd, M. (1976). Orientation in complementary pivot algorithms. *Mathematics of Operations Research*, 1:54–66.

Uzawa, H. (1962). Walras's existence theorem and Brouwer's fixed point theorem. *Economic Studies Quarterly*, 13:59–62.

van den Elzen, A. and Talman, A. (1991). A procedure for finding Nash equilibria in bi-matrix games. *ZOR – Methods and Models for Operations Research*, 35:27–43.

van den Elzen, A. and Talman, A. (1999). An algorithmic approach toward the tracing procedure for bi-matrix games. *Games and Economic Behavior*, 28:130–145.

van der Laan, G. and Talman, A. (1982). On the computation of fixed points in the product space of unit simplices and an application to noncooperative n person games. *Mathematics of Operations Research*, 7:1–13.

Vazirani, V. and Yannakakis, M. (2011). Market equilibrium under separable, piecewise-linear, concave utilities. *Journal of the ACM*, 58:10:1–10:25.

von Mises, L. (1990). *Economic Calculation in the Socialist Commonwealth*. The Ludwig von Mises Institute, Auburn Alabama.

von Stengel, B. (2002). Computing equilibria for two-person games. In Aumann, R. and Hart, S., editors, *Handbook of Game Theory*, volume 3, chapter 45, pages 1723–1759. Elsevier, Amsterdam.

Walker, M. (1977). On the informational size of message spaces. *Journal of Economic Theory*, 15:366–377.

Williams, S. R. (1985). Necessary and sufficient conditions for the existence of a locally stable message process. *Journal of Economic Theory*, 35:127–154.

Yannakakis, M. (2009). Equilibria, fixed points, and complexity classes. *Computer Science Review*, 3:71–85.

Ye, Y. (2007). Exchange market equilibria with Leontief's utility: Freedom of pricing leads to rationality. *Theoretical Computer Science*, 378:134–142.

Ye, Y. (2008). A path to the Arrow-Debreu competitive market equilibrium. *Mathematical Programming*, 111:315–348.