Games, Complexity Classes, and Approximation Algorithms

Joan Feigenbaum

Abstract. We survey recent results about game-theoretic characterizations of computational complexity classes. We also show how these results are used to prove that certain natural optimization functions are as hard to approximate closely as they are to compute exactly.

1991 Mathematics Subject Classification: 68Q15
Keywords and Phrases: Games, Complexity, Approximation Algorithms, Perfect Information, Perfect Recall

1 Introduction

Game theory provides a framework in which to model and analyze conflict and cooperation among independent decision makers. Many areas of computer science have benefitted from this framework, including artificial intelligence, distributed computing, security and privacy, and lower bounds. Games are particularly important in computational complexity, where they are used to characterize complexity classes, to understand the power and limitations of those classes, and to interpret the complete problems for those classes.

This paper surveys three sets of results in the interplay of games and complexity. First, we present several characterizations (some old, some new) of the complexity class PSPACE that show that it is extremely robustly characterized by zero-sum, perfect-information, polynomial-depth games. Next, we explain how the more recent of these characterizations of PSPACE are used to show that certain natural maximization and minimization functions, drawn from domains such as propositional logic, graph searching, graph reliability, and stochastic optimization, are as hard to approximate closely as they are to compute exactly. Finally, we present some connections between complexity classes and imperfect information games; some tight characterizations of exponential-time classes are known, but no set of imperfect-information games is as robustly identified with any complexity class as zero-sum, perfect-information, polynomial-depth games are with PSPACE.

We assume familiarity with basic computational complexity theory, especially with the complexity classes P, NP, PSPACE, EXP, and NEXP, with the notions of reduction and completeness, and with the concept of an "approximation algorithm" for an NP-hard or PSPACE-hard optimization function. Among the books that cover this material and are accessible to all mathematically educated readers are those by Garey and Johnson [10] and Papadimitriou [16]. We also assume familiarity with elementary game theory, in particular with the notions of "perfect information" and "perfect recall." The few game-theoretic notions that we use are defined precisely in, *e.g.*, [9]

2 ALTERNATION AND RANDOMIZED PLAYERS

Chandra *et al.* [5] proved a fundamental result about the connection between games and complexity that serves as the starting point for most of the results surveyed in this paper. In the Alternating Polynomial Time computational model, there are two computationally unbounded players P_1 and P_0 and a polynomial-time referee V. There is an input string x written on a common tape readable by P_1, P_0, and V, and the goal of the computation is to determine whether x is in the language L. P_1 claims that $x \in L$, and P_0 claims that $x \notin L$. They "argue" for polynomially many rounds, and then V decides who's right. More precisely, there are two functions m and l such that, on inputs x of length n, P_1 and P_0 take turns for $m(n)$ rounds (P_1 moving in odd rounds and P_0 in even rounds), writing a string of length $l(n)$ in each round. Both $m(n)$ and $l(n)$ are polynomially bounded (abbreviated poly(n)). After the entire "game transcript" of length $m(n) \cdot l(n)$ has been written, V reads it, does a polynomial-time computation, and outputs "ACCEPT" or "REJECT," depending on whether it thinks the winner is P_1 or P_0. For (P_1, P_0, V) to be an Alternating Polynomial Time machine for the language L, it must have the property that, if $x \in L$, V always outputs ACCEPT (*i.e.,*, P_1 has a winning strategy), and, if $x \notin L$, V always outputs REJECT (*i.e.,*, P_0 has a winning strategy). The fundamental result of Chandra *et al.* [5] is that Alternating Polynomial Time is equal to PSPACE: Languages that correspond to zero-sum, perfect-information, polynomial-depth games are exactly those recognizable by Turing Machines that use polynomial space.

The fundamental correspondence between PSPACE and perfect-information games is clearly illustrated by the well-known PSPACE-complete language of true quantified Boolean formulas. Consider quantified Boolean formulas in 3CNF ("three conjunctive normal form"); that is, those of the form

$$\Phi = Q_1 x_1 Q_2 x_2 \dots Q_n x_n \phi(x_1, x_2, \dots, x_n),$$

where each $Q_i \in \{\exists, \forall\}$, each x_i is a Boolean variable, and ϕ is a formula in conjunctive normal form, each clause of which has exactly three literals. Let Q3SAT be the set of true quantified formulas in 3CNF. To obtain a perfect-information game, let the variables of the formula be chosen by P_1 and P_0, in order of quantification, where P_0 chooses the universally quantified variables and P_1 chooses the existentially quantified variables. By definition, the formula is true (*i.e.*, in Q3SAT) if and only if P_1 has a winning strategy for this game. The classical paper of Schaefer [18] provides many more examples of PSPACE-complete perfect-information games.

Papadimitriou [17] considers an interesting variation on the Alternating Polynomial Time model. In a "Game Against Nature," the input x is still given to P_1, who claims that $x \in L$, to P_0, who claims that $x \notin L$, and to the polynomial-time referee V. However, P_0 is now a "random" player; in even-numbered moves of the game, P_0 just tosses fair coins to select a string of the appropriate length uniformly at random. Thus, instead of playing against a strategic opponent, P_1 is playing against "nature." If P_1 is truthful in his claim that $x \in L$, then V must accept with probability at least $1/2$. If $x \notin L$, then V must accept with probability less than $1/2$. (The probability is computed over the coin tosses of P_0.) The main result of [17] is that Games Against Nature recognize exactly the languages in PSPACE – or, at least for perfect-information, polynomial-depth games, playing against "nature" is just as hard for P_1 as playing against an evenly-matched opponent!

Babai and Moran [3] consider "Arthur-Merlin Games." These are defined in the same way as Games Against Nature, except that there must be a "gap" in acceptance probabilities: If $x \in L$, then V must accept with probability at least $2/3$, and, if $x \notin L$, then V must accept with probability at most $1/3$. One of the most highly acclaimed results in computational complexity theory, proved by Lund *et al.* [14] and Shamir [19], is that the (seemingly very stringent) requirement of this $(1/3, 2/3)$ gap does *not* change the class of languages accepted: poly(n)-round Arthur-Merlin Games also recognize exactly PSPACE.

3 PROBABILISTICALLY CHECKABLE DEBATE SYSTEMS

In the Alternating Polynomial Time, Games Against Nature, and Arthur-Merlin Game models, the referee reads the entire transcript of a played game before deciding the winner. In this Section, we consider models in which the referee reads only a randomly selected subset of the game transcript but can still decide the winner correctly, because the players encode their moves in a clever way that makes refereeing easy. The results obtained are the PSPACE analogue of the *probabilistically checkable proof system* theory developed for NP (see, *e.g.*, [1, 20]).

A *probabilistically checkable debate system* (PCDS) for a language L consists of a player P_1, who claims that the input x is in L, a player P_0, who claims that x is not in L, and a *probabilistic* polynomial-time referee V. The language L is in the complexity class PCD$(r(n), q(n))$ if V flips at most $O(r(n))$ coins on inputs x of length n and reads at most $O(q(n))$ bits of the game transcript produced by P_1 and P_0. On inputs $x \in L$, V always declares P_1 to be the winner, and on inputs $x \notin L$, V declares P_0 to be the winner with probability at least $2/3$. An RPCDS is a PCDS in which player P_0 follows a very simple strategy: In each even round of the game, P_0 simply chooses uniformly at random from the set of all legal moves. The class RPCD$(r(n), q(n))$ is defined by analogy with PCD$(r(n), q(n))$.

The characterizations of PSPACE presented in Section 2 are those in which $r(n) = 0$ and $q(n)$ is an arbitrary polynomial. Specifically, Alternating Polynomial Time is, by definition, PCD$(0, \text{poly}(n))$, and poly(n)-round Arthur-Merlin Games are RPCD$(0, \text{poly}(n))$.

Condon *et al.* [6, 7] study the potential tradeoff between random bits and query bits. If the referee V is allowed to flip coins, might it still be able to determine the winner of the game without reading the entire transcript? The results in [6, 7] show that, as in the PCP characterization of NP, the best possible tradeoff between

$r(n)$ and $q(n)$ is obtainable. Furthermore, this tradeoff is obtainable both when the opponents are two strategic players (a PCDS) and when they are a strategic player and a random player (an RPCDS). Specifically, it is shown in [6, 7] that

$$\text{PSPACE} = \text{PCD}(\log n, 1) = \text{RPCD}(\log n, 1).$$

One surprising aspect of these results is that, while the number of rounds of the game is poly(n), the number of bits of the game examined by the referee is $O(1)$. Thus, most of the moves of *both* players are never looked at, and yet the referee still decides the winner correctly. In order to encode games to permit such efficient refereeing, Condon *et al.* [6, 7] exploit and extend the probabilistically checkable coding techniques developed in the PCP characterization of NP [1, 20],

In conclusion, the results of [5, 6, 7, 14, 17, 19] demonstrate that the identification of PSPACE with zero-sum, perfection-information, polynomial-depth games is extremely robust. Numerous variations on the computational model of a game between two strategic players that is judged after it is played by a polynomial-time referee have been studied, *e.g.,* replacing one strategic player by a random player, putting a sharp threshold between yes-instances and no-instances precisely at acceptance probability $1/2$, requiring a $(1/3, 2/3)$ gap in acceptance probability between yes-instances and no-instances, and only allowing the referee to examine a constant number of bits of the played game before making a decision. All of these variations on perfect-information games (and several combinations thereof) cause the same class of languages to be accepted, namely PSPACE. As the results surveyed below in Section 5 demonstrate, there is no complexity class known to be as robustly identifiable with a class of imperfect-information games.

4 NONAPPROXIMABILITY

The game-theoretic characterizations of PSPACE presented in Sections 2 and 3 can be used to prove that many optimization functions that are PSPACE-hard to compute exactly are also PSPACE-hard to approximate closely. This use of the debate-system characterizations in Section 3 was inspired by the use of the PCP$(\log n, 1)$ characterization of NP to prove nonapproximability results for NP-hard optimization functions; see Arora and Lund [2] for an overview of these results on NP.

The basic proof structure of the nonapproximability results surveyed in this Section is as follows. First, a characterization of PSPACE is used directly to show that a particular function F is hard to approximate within a certain factor; then approximability-preserving, polynomial-time reductions are given from F to other functions of interest. Note that these reductions must be constructed with some care, because the mere fact that two optimization problems are equivalent under polynomial-time reductions does not mean that they are equivalent with respect to approximability. A canonical example of a polynomial-time reduction that does not appear to preserve approximability is the one from VERTEX COVER to INDEPENDENT SET (see Section 6.1 of Garey and Johnson [10]).

Throughout this section, we say that "algorithm A approximates the function f within ratio $\epsilon(n)$," for $0 < \epsilon(n) < 1$, if, for all x in the domain of f, $\epsilon(|x|) \leq$

$A(x)/f(x) \leq 1/\epsilon(|x|)$. If $\epsilon(n) > 1$, then "algorithm A approximates the function f within ratio $\epsilon(n)$" means that $1/\epsilon(|x|) \leq A(x)/f(x) \leq \epsilon(|x|)$.

4.1 Reductions from PCD($\log n$, 1)

From the characterization PSPACE = PCD($\log n$, 1), we obtain a nonapproximability result for one optimization version of the PSPACE-complete language Q3SAT defined in Section 2. Let the variables of the formula be chosen by P_0 and P_1, in order of quantification, where P_0 chooses the universally quantified variables and P_1 chooses the existentially quantified variables. If P_1 can guarantee that k clauses of ϕ will be satisfied by the resulting assignment, regardless of what P_0 chooses, we say that k clauses of Φ are *simultaneously satisfiable*. Let MAX Q3SAT be the function that maps a quantified 3CNF formula Φ to the maximum number of simultaneously satisfiable clauses.

Theorem: There is a constant $0 < \epsilon < 1$ such that approximating MAX Q3SAT within ratio ϵ is PSPACE-hard.

Nonapproximability results for other PSPACE-hard functions can now be obtained via approximability-preserving reductions from MAX Q3SAT. The following two are given by Condon *et al.* [6]:

MAX FA-INT: The language FA-INT consists of all sets $\{A_1, A_2, \ldots, A_m\}$ of deterministic finite-state automata having the same input alphabet Σ such that there is a string w that is accepted by all of them. FA-INT plays a key role in the field of "computer-aided verification" of devices and protocols (see, *e.g.*, Kurshan [12]) and was shown to be PSPACE-complete by Kozen [11]. The PSPACE-hard function MAX FA-INT maps each set $\{A_1, A_2, \ldots, A_m\}$ to the largest integer k such that there is a string w accepted by k of the A_i's.

MAX GGEOG: Instances of the game "generalized geography" consist of pairs (G, s), where G is a directed graph and s is a distinguished start node. A marker is initially placed on s, and P_0 and P_1 alternatively play by moving the marker along an arc that goes out of the node it is currently on. Each arc can be used at most once; the first player that is unable to move loses. The language GGEOG consists of all pairs (G, s) for which P_1 has a winning strategy; GGEOG is one of the many perfect-information games shown to be PSPACE-complete by Schaefer [18]. We say that (G, s) "can be played for k rounds" if P_1 has a strategy that causes the marker to be moved along k arcs, no matter what P_0 does, even if P_1 ultimately loses. The PSPACE-hard function MAX GGEOG maps pairs (G, s) to the maximum number of rounds for which they can be played.

In fact, the lower bounds for MAX FA-INT and MAX GGEOG are stronger than the one for MAX Q3SAT: In both cases, there is a constant $\epsilon > 0$ such that approximating the function to within a factor of n^ϵ is PSPACE-hard. Additional nonapproximability results from the domains of modal logic and system specification and analysis are given, respectively, by Lincoln *et al.* [13] and by Marathe *et al.* [15].

4.2 REDUCTIONS FROM RPCD(0, poly(n))

The PSPACE-complete language SSAT is defined as follows by Papadimitriou [17]. An instance is a 3CNF formula ϕ over the set of variables $\{x_1, x_2, \ldots, x_n\}$. The instance is in SSAT if there is a choice of Boolean value for x_1 such that, for a random choice (with True and False each chosen with probability 1/2) for x_2, there is a choice for x_3, etc., for which the probability that ϕ is satisfied is at least 1/2. Think of SSAT as a game between an existential player and a random player; on odd moves i, the existential player chooses an optimal value for x_i (where "optimal" means "maximizes the probability that ϕ will be satisfied") and, on even moves i, the random player chooses a random value for x_i. Yes-instances of SSAT are those in which the existential player wins with probability at least 1/2.

The function MAX-PROB SSAT maps each SSAT instance to the probability that ϕ is satisfied if the existential player plays optimally; so yes-instances of the decision problem are those on which the value of MAX-PROB SSAT is at least 1/2. The proof that PSPACE = RPCD(0, poly(n)) (see Lund *et al.* [14] and Shamir [19]) yields the following strong nonapproximability result.

THEOREM: For any language L in PSPACE and any $\epsilon < 1$, there is a polynomial-time reduction f from L to SSAT such that

$$x \in L \Rightarrow \text{MAX-PROB SSAT}(f(x)) = 1, \text{ and}$$

$$x \notin L \Rightarrow \text{MAX-PROB SSAT}(f(x)) < 2^{-n^{\epsilon}},$$

where n is the number of variables in $f(x)$.

Condon *et al.* [7] and Papadimitriou [17] give approximability-preserving reductions from MAX-PROB SSAT to the following three functions.

MIN DMP: An instance of Dynamic Markov Process (DMP) is a set S of states and an $n \times n$ stochastic matrix P, where $n = |S|$. Associated with each state s_i is a set D_i of decisions, and each $d \in D_i$ is assigned a cost $c(d)$ and a matrix R_d. Each row of R_d must sum to 0, and each entry of $P + R_d$ must be nonnegative. The result of making decision d when the process is in state s_i is that a cost of $c(d)$ is incurred, and the probability of moving to state s_j is the $(i,j)^{th}$ entry of $P + R_d$. A strategy determines which decisions are made over time; an optimal strategy is one that minimizes the expected cost of getting from state s_1 to state s_n. The language DMP, shown to be PSPACE-complete by Papadimitriou [17], consists of tuples $(S, P, \{D_i\}, c, \{R_d\}, B)$ for which there is a strategy with expected cost at most B. The optimization function MIN DMP maps $(S, P, \{D_i\}, c, \{R_d\})$ to the expected cost of an optimal strategy.

COLORING GAMES: An instance of a coloring game (see Bodlaender [4]) consists of a graph $G = (V, E)$, an ownership function o that specifies which of P_0 and P_1 owns each vertex, a linear ordering f on the vertices, and a finite set C of colors. This instance specifies a game in which the players color the vertices in the order specified by the linear ordering. When vertex i is colored, its owner chooses a color from the set of *legal* colors, *i.e.*, those in set C that are *not* colors of the colored neighbors of i. The game ends either when all vertices are colored, or when a player cannot color the next vertex in the linear ordering f because there are no legal colors. P_1 wins if and only if all vertices are colored at the end of the game. The length of the game is the number of colored vertices at the end of the game. In a stochastic coloring game (SCG), P_0 chooses a color uniformly at random from the set of legal colors at each stage. Two corresponding optimization problems are to maximize the following functions: MAX-PROB SCG(G, o, f, C), which is the maximum probability that P_1 wins the game (G, o, f, C), and MAX-LENGTH SCG(G, o, f, C), which is the maximum expected length of the game. Both maxima are computed over all strategies of P_1.

For each of MIN DMP and MAX-PROB SGC, there is a constant $\epsilon > 0$ such that it is PSPACE-hard to approximate the function within ratio $2^{-n^{\epsilon}}$. For MAX-LENGTH SGC, the ratio within which approximation is PSPACE-hard is $n^{-\epsilon}$, for a constant $\epsilon > 0$.

4.3 REDUCTIONS FROM RPCD$(\log n,\ 1)$

The starting point for this set of nonapproximability results is the function MAX-CLAUSE SSAT, whose value on the 3CNF formula ϕ is the expected number of clauses of ϕ that are satisfied if P_1 chooses the values of the existentially quantified variables, the other variables are assigned random values, and P_1 plays optimally with the goal of maximizing the number of satisfied clauses. Using their result that PSPACE = RPCD$(\log n,\ 1)$, Condon *et al.* [7] prove the following.

THEOREM: There is a constant $0 < \epsilon < 1$ such that approximating MAX-CLAUSE SSAT within ratio ϵ is PSPACE-hard.

They then reduce MAX-CLAUSE SSAT to many other optimization functions, using reductions that preserve approximability. Two examples include:

MAX SGGEOG: Consider the variation of the game GGEOG defined in Section 4.2 in which P_0 plays randomly; that is, at every even-numbered move, P_0 simply chooses an unused arc out of the current node uniformly at random and moves the marker along that arc. The goal of P_1 is still to maximize the length of the game, and the function MAX SGGEOG maps an instance (G, s) to the expected length of the game that is achieved when P_1 follows an optimal strategy.

MAX-PROB DGR: The Graph Reliability problem is defined as follows by Valiant [21]: Given a directed, acyclic graph G, source and sink vertices s and t, and a failure probability $p(v, w)$ for each arc (v, w), what is the probability that there is a path from s to t consisting exclusively of arcs that have not failed? Papadimitriou [17] defines Dynamic Graph Reliability (DGR) as follows: The goal of a strategy is still to traverse the digraph from s to t. Now, however, for each vertex x and arc (v, w), there is a failure probability $p((v, w), x)$; the interpretation is that, if the current vertex is x, the probability that the arc (v, w) will fail before the next move is $p((v, w), x)$. The PSPACE-complete language DGR consists of those digraphs for which there exists a strategy for getting from s to t with probability at least $1/2$. A natural optimization function is MAX-PROB DGR, which maps a graph, vertices s and t, and a set $\{p((v, w), x)\}$ of failure probabilities to the probability of reaching t from s under an optimal strategy.

It is PSPACE-hard to approximate MAX SGGEOG within ratio $n^{-\epsilon}$, for any constant $0 < \epsilon < 1/2$, where n is the number of vertices in the graph. It is also PSPACE-hard to approximate MAX-PROB DGR within ratio $2^{-n^{\epsilon}}$, for some constant $\epsilon > 0$. See Condon *et al.* [7] for proofs of these results and for a related result about a stochastic version of the board game Mah-Jongg.

5 Imperfect Information Games

Feigenbaum *et al.* [9] develop a framework in which to generalize the connections between game classes and complexity classes. A *polynomially definable game system* (PDGS) for a language L consists of two arbitrarily powerful players P_0 and P_1 and a polynomial-time referee V. The referee may be probabilistic, but there are some interesting cases in which V does not need randomness. P_0 and P_1 and the referee V have a common input tape. On input x, P_1 claims that x is in L, P_0 claims that x is not in L, and V's job is to decide which of these two claims is true.

Each input x to a PDGS determines a *polynomially definable game* G_x as follows. The game is essentially run by the referee V. The moves in the game are relayed by the players to V. Neither player sees V's communication with the other, but V can transmit information about the current status of the game to one or both players. This reflects the fact that the players can have imperfect information to varying degrees. When the interaction is finished, V either accepts or rejects x. Because the referee is polynomial-time, G_x lasts for poly(n) moves, and each move can be written down in poly(n) bits, where $n = |x|$. The resulting game G_x clearly defines a two-person, zero-sum game tree in which the length of each path is polynomial. If V is probabilistic, then his coin tosses correspond to chance moves in the game tree.

It is essential to the PDGS framework that P_0 and P_1 use mixed strategies. (See [9, Section 1] for a discussion of why previous attempts to characterize complexity classes with imperfect-information games in which the players use pure strategies were unsatisfactory.) That is, for each possible input x, each player has a probability distribution over the space of his deterministic strategies. At the beginning of the game, the players examine x and independently choose a pure strategy using their respective probability distributions; those pure strategies are then played throughout the game. Since the game tree has exponential size, a pure strategy also has exponential size. An arbitrary mixed strategy could of course have size doubly exponential in n.

There are two ways to define acceptance of a language L by a PDGS (P_1, P_0, V). In the "exact model," yes-instances x correspond to games G_x of value at least $1/2$ and no-instances to games of value less than $1/2$:

- For all $x \in L$, there exists a mixed strategy μ_1 for P_1 such that, for all strategies μ_0 for P_0, V accepts with probability at least $1/2$.

- For all $x \notin L$, there exists a mixed strategy μ_0 for P_0 such that, for all strategies μ_1 for P_1, V accepts with probability less than $1/2$.

In the "approximate model," yes-instances x correspond to games G_x of value at least 2/3 and no-instances to games of value at most 1/3:

- For all $x \in L$, there exists a mixed strategy μ_1 for P_1 such that, for all strategies μ_0 for P_0, V accepts with probability at least 2/3.
- For all $x \notin L$, there exists a mixed strategy μ_0 for P_0 such that, for all strategies μ_1 for P_1, V accepts with probability at most 1/3.

In both models, the probability of acceptance is computed over the pure strategies of both players (if they use mixed strategies) and the coin tosses of V (if any).

The main question addressed in [9] is the relationship between the game-theoretic properties of P_0 and P_1 and the class of languages recognizable by PDGS's. One class of PDGS's studied are those in which at least one player has imperfect information (*i.e.*, those in which the referee V does not tell P_0 everything about its communication with P_1 and/or *vice versa*) but *perfect recall* (*i.e.*, P_0 and/or P_1 has enough memory to record everything they do and everything they receive from V and can use it in subsequent rounds of the protocol). Another class are those in which at least one player has imperfect recall: P_0 or P_1 or both cannot store everything they do and receive and may have to act in the i^{th} round of the game based on partial or no information about what happened in the first $i-1$ rounds.

In the results on PSPACE surveyed in Sections 2 and 3, the computational models are very special cases of PDGS's, in which the referee's role is trivial while the game is being played: V simply sends all information about P_1's current move and the entire history of the game to P_0 and *vice versa*. Therefore these results show that PDGS's in which both players have perfect information recognize exactly PSPACE, both in the exact model and in the approximate model. Feigenbaum *et al.* [9] obtain similarly tight results for PDGS's in which at least one player has imperfect recall. If P_1 has imperfect recall, but P_0 has either perfect information or perfect recall, then PDGS's accept exactly those languages recognizable in nondeterministic exponential time (the complexity class NEXP), in both the exact model and the approximate model. If P_0 is the one with imperfect recall, the class recognized in both models is coNEXP. An almost-tight characterization is obtained for PDGS's in which both players have imperfect recall (see [9] for details).

Feigenbaum *et al.* [9] also proved that, in the exact-value model, the languages accepted by PDGS's in which P_0 and P_1 both have perfect recall (but imperfect information) are exactly those languages recognizable in deterministic exponential time (the complexity class EXP). They left open the question of whether the approximate-value model is equivalent to the exact-value model when both players have perfect recall. This question was subsequently answered by Feige and Kilian [8]: One-round PDGS's with two perfect-recall players accept PSPACE; Polynomial-round PDGS's with two perfect-recall players accept EXP.

Thus, perfect-recall games seem to be fundamentally different perfect-information games and from games in which at least one player has imperfect recall; in particular, whether or not exact refereeing is equivalent to approximate refereeing seems to depend on the number of rounds of the game. It remains open whether there are natural explanations or generalizations of these results on imperfect information games or whether these results have applications to approximability. Also open is the question of whether there are imperfect-information analogues of the Arthur-Merlin and Games-Against-Nature characterizations of PSPACE; that is, can a random player replace one of the perfect-recall players or one of the imperfect-recall players in a class of PDGS's without changing the language-recognition power of the class.

REFERENCES

[1] S. Arora, "Probabilistic Checking of Proofs and Hardness of Approximation Problems," PhD Thesis, University of California, Computer Science Division, Berkeley, 1994.

[2] S. Arora and C. Lund, "Hardness of Approximations," in APPROXIMATION ALGORITHMS FOR NP-HARD PROBLEMS, D. Hochbaum (ed.), PWS Publishing, Boston, 1997, pp. 399-446.

[3] L. Babai and S. Moran, "Arthur-Merlin Games: A Randomized Proof System and a Hierarchy of Complexity Classes," *Journal of Computer and System Sciences*, 36 (1988), pp. 254-276.

[4] H. Bodlaender, "On the Complexity of Some Coloring Games," *International Journal on Foundations of Computer Science*, 2 (1991), pp. 133-147.

[5] A. Chandra, D. Kozen, and L. Stockmeyer, "Alternation," *Journal of the Association for Computing Machinery*, 28 (1981), pp. 114-133.

[6] A. Condon, J. Feigenbaum, C. Lund, and P. Shor, "Probabilistically Checkable Debate Systems and Nonapproximability Results for PSPACE-Hard Functions," *Chicago J. Theoretical Computer Science*, vol. 1995, no. 4. `http://www.cs.uchicago.edu/publications/cjtcs/articles/-1995/4/contents.html`

[7] A. Condon, J. Feigenbaum, C. Lund, and P. Shor, "Random Debaters and the Hardness of Approximating Stochastic Functions," *SIAM Journal on Computing*, 26 (1997), pp. 369-400.

[8] U. Feige and J. Kilian, "Making Games Short," in *Proceedings of the 29th Symposium on Theory of Computing*, ACM Press, New York, 1997, pp. 506-516.

[9] J. Feigenbaum, D. Koller, and P. Shor, "A Game-Theoretic Classification of Interactive Complexity Classes," in *Proceedings of the 10th Conference on Structure in Complexity Theory*, IEEE Computer Society Press, Los Alamitos, 1995, pp. 227-237.

[10] M. Garey and D. Johnson, COMPUTERS AND INTRACTABILITY: A GUIDE TO THE THEORY OF NP-COMPLETENESS, Freeman, San Francisco, 1979.

[11] D. Kozen, "Lower Bounds for Natural Proof Systems," in *Proceedings of the 18th Symposium on Foundations of Computer Science*, IEEE Computer Society Press, Los Alamitos, 1977, pp. 254-266.

[12] R. Kurshan, Computer-Aided Verification of Coordinating Processes: The Automata-Theoretic Approach, Princeton University Press, Princeton, 1994.

[13] P. Lincoln, J. Mitchell, and A. Scedrov, "Optimization Complexity of Linear Logic Proof Games," to appear in *Theoretical Computer Science*.

[14] C. Lund, L. Fortnow, H. Karloff, and N. Nisan, "Algebraic Methods for Interactive Proof Systems," *Journal of the Association for Computing Machinery*, 39 (1992), pp. 859-868.

[15] M. Marathe, H. Hunt, R. Stearns, and V. Radhakrishnan, "Approximation Algorithms for PSPACE-Hard Hierarchically and Periodically Specified Problems," *SIAM Journal on Computing*, 27 (1998), pp. 1237-1261.

[16] C. Papadimitriou, Computational Complexity, Addison-Wesley, Reading, 1994.

[17] C. Papadimitriou, "Games Against Nature," *Journal of Computer and System Sciences*, 31 (1985), pp. 288-301.

[18] T. Schaefer, "On the Complexity of Some Two-Person, Perfect-Information Games," *Journal of Computer and System Sciences*, 16 (1978), pp. 185-225.

[19] A. Shamir, "IP = PSPACE," *Journal of the Association for Computing Machinery*, 39 (1992), pp. 869-877.

[20] M. Sudan, "Efficient Checking of Polynomials and Proofs and the Hardness of Approximation Problems," PhD Thesis, University of California, Computer Science Division, Berkeley, 1992.

[21] L. Valiant, "The Complexity of Enumeration and Reliability Problems," *SIAM Journal on Computing*, 8 (1979), pp. 410-421.

Joan Feigenbaum
AT&T Labs - Research
180 Park Avenue
Florham Park, NJ 07932-0971
USA
jf@research.att.com