Algorithmic Game Theory - handout1

Uriel Feige

21 March 2013

The course will have significant overlap with a course given in 2008. Its web page is accessible through the home page of Moni Naor, and includes many references, and a link to the web version of [NRTV]. Additional material may appear at a later time on my own web page.

Homework assignments are an integral part of the course and will be a significant part of the grade. Please do the reading assignments, and hand in the written assignments two weeks after they are given.

Homework. (Please keep the answers short and easy to read.)

Read chapter 1 in [NRTV].

The family of *and-or* games is a family of two-player constant-sum games given in extensive form (as a game tree). There are two players, *min* and *max*. The game tree is a full binary tree of depth n and $N = 2^n$ leaves. Every leaf has a value, which is the payoff given to *max*, which can be either 1 (win) or 0 (loose). The payoff to *min* is the complement of the payoff to *max* (when *max* loses *min* wins). The game starts at the root of the tree. Number the layers of the tree from 0 at the root to n at the leaves. At even numbered layers, *min* chooses a move (an edge leading from the current node to the next layer), and at odd layers *max* chooses a move. After n moves a leaf is reached, each player gets his respective payoff, and the game ends.

1. How would an *and-or* game be represented in standard form? In particular, how many rows and how many columns will the game matrix have?
2. Prove that in every *and-or* game one of the players has a strategy that forces a win for that player regardless of the strategy of the other player. (Hint: you may use induction.) Show an algorithm of time complexity poly(N) for computing such a strategy. (Space complexity is also important in practice. Space poly(n) suffices for computing an optimal move, though achieving this is not required in this question.)
3. What is the smallest number of leaves that can have value +1 and still *max* will have a winning strategy? Explain.
4. Prove that in every *and-or* game, at least one of the players has a dominant strategy.
5. Show an example of an *and-or* game in which *min* does not have a dominant strategy.
6. What is the largest number of leaves (as a function of n) that can have value +1 and still *max* will have a dominant strategy but no winning strategy? Explain.
7. Show an algorithm of time complexity poly(N) for computing a subgame perfect Nash equilibrium for an *and-or* game.

References

[NRTV] Noam Nisan, Tim Roughgarden, Eva Tardos and Vijay V. Vazirani (Editors), Algorithmic Game Theory, Cambridge University Press, 2007.