Everything You Always Wanted to Know About Game Theory* (* But Were Afraid to Ask)

Daniel D. Garcia (moderator)
University of California, Berkeley
ddgarcia@cs.berkeley.edu

David Ginat
Tel-Aviv University
ginat@post.tau.ac.il

Peter Henderson
Butler University
phenders@butler.edu

Categories & Subject Descriptors
K.3 [**Computers & Education**]: Computer & Information Science Education – *Computer Science Education.*

General Terms
Algorithms, Theory.

Keywords
Economic Game Theory, Prisoner's Dilemma, Ethics, Combinatorial Game Theory, Artificial Intelligence, CS1/2.

Introduction

"Every game ever invented by mankind, is a way of making things hard for the fun of it!" –John Ciardi

At recent SIGCSE conferences, there appeared to be a refreshing revival of interest in game theory as a valuable pedagogical tool for educators [4,5]. Games can serve as an excellent resource for CS0, CS1, networking, user interface and software engineering projects. Games can be utilized to motivate students, enrich their intuition, and illustrate fundamental principles of algorithm design such as exploration of regularities, modularity, correctness, and efficiency. Many CS instructors look for ways to motivate, improve intuition, and illustrate the subject matter they teach, and games are a wonderful means of doing so.

Unfortunately, not everyone in our community has taken formalized courses in the subject, and many may feel they lack the prerequisites to initiate their own "nifty" game assignment or example. This special session hopes to provide enough of a basic tutorial of game theory (and many of its variants) that attendees will be able to use it in their own pedagogy with confidence. In this panel, we will present basic theory through numerous examples, discuss didactic aspects of games, and try to identify curricular topics for illustration with mathematical games. Participants will be encouraged to present examples of successful incorporation of games into their instruction.

We have chosen to divide the broad field of game theory into three distinct categories: economic, combinatorial and computational. Briefly, economic game theory concerns matrix games with simultaneous moves, e.g., the famed Prisoner's Dilemma problem. Combinatorial game theory is founded upon the rich mathematics for finite, two-person, complete information games and their sums, e.g., Nim and Dots & Boxes. Computational game theory is our own classification of two-person games that require computation to solve and/or create a worthy opponent, such as Chess, Checkers and Go.

Peter Henderson – Economic Game Theory

The 2001 blockbuster film *A Beautiful Mind* highlighted the life of John Nash, the Nobel Prize-winning economist, and introduced the nation to the benefits of economic game theory. In fact, the field traces its roots back to von Neumann and Morgenstern's book: *The Theory of Games and Economic Behavior*. It introduced the restricted idea of a "zero-sum game", in which any benefit to one player must be balanced by a loss of the same amount by the opposition. It also classified games as either *cooperative*, in which games may have many players who may or may not collude, and *non-cooperative*, in which players are strictly competitive. John Nash became famous for relaxing the zero-sum restriction they imposed for non-cooperative games and introduced the idea of (what is now called) a Nash equilibrium describing mixed strategies [3].

One of the most famous examples of this rich theory is the Prisoner's Dilemma problem, which describes a situation in which two players (we'll name **bold** and *italic*) are accused of a crime, placed in separate cells and asked to confess. Each prisoner can independently decide to cooperate (remain silent) or defect (turn the other in), and is asked to choose simultaneously. Figure 1 illustrates the payoff matrix for the four possible outcomes described below – the numbers represent *how many years each will spend in prison for that particular outcome*. If both cooperate, both will serve only 3 years in prison. If both defect (ideal for the police), both serve 5 years in prison. If one defects and one cooperates, the one who defects is set free (0 years in prison) and the one who remained silent serves 6 years, the harshest penalty.

The behavior that emerges from this when players are asked to play some unknown consecutive number of times is, simply put, fascinating. Often students start out cooperating, then one defects, and then the other usually follows to defect quickly. However, if both continue to defect, they suffer the most cumulative punishment (5+5=10 years) than any other outcome. It is interesting to watch who is willing to cooperate given that their opponent has continued to defect on them in the recent past.

	Cooperate	*Defect*
Cooperate	**3**, *3*	**6**, *0*
Defect	**0**, *6*	**5**, *5*

Figure 1 – The Prisoner's Dilemma matrix.

How might one turn this problem into a nifty project? Students in CS1 or CS3 could be provided the networking module and asked to write a program to play the game many times with random opponents in a lab in succession and tabulate the scores. Networking and software engineering classes could be asked to write the client and server architecture to support the play.

David Ginat – Combinatorial Game Theory

In 1982, Elwyn Berlekamp, John Conway and Richard Guy published their landmark *Winning Ways for your Mathematical Plays*, which has since become the accepted reference to the field of combinatorial game theory [2]. Although it provides insight and analysis on a wide range of games, the book focuses mostly on "normal play" games: two-players moving alternately on a position with complete information (both players know what is going on at all times). In these games, there is no chance and the last player to move wins. They developed a rich mathematics to describe the games, and in doing so introduced hundreds of interesting and innovative games to use as illustrations.

One such game is Nim, whose rules are quite simple. The game begins with several piles of beans, often with different numbers of beans in each pile. Players take turns removing as many beans as they wish (but at least one) from a single pile. The player to remove the last bean wins. Nim is fascinating for a couple of reasons. First, it is one of the few games for which there exists an easy way to calculate whether it is better to play first (or second), and if first, what move will guarantee victory. The technique involves calculating the binary representations of the number of beans in each pile. What a sure-fire way to motivate the topic! You can guarantee they'll learn a foolproof method of winning every Nim game they'll ever play (given that they have the *choice* to move first or second). Second, Nim forms the foundation of the impartial (both players have equal moves available to them) theory of games, which has the surprising result that every impartial game is equivalent to a game of Nim with a single (possibly zero) pile of beans.

Combinatorial games serve as excellent project fodder, and help highlight that mathematics is more powerful than brute force in many cases. What makes these games so wonderful is that there can be quite interesting subtleties and strategies for games of very small size [7]. Those of us who have taught using games in our curriculum have found that they serve as excellent motivators for students to investigate the mathematics behind the play, always a rewarding result. One example of a curriculum-topic in which this has been demonstrated is that of program design and verification [6].

Daniel D. Garcia – Computational Game Theory

The field of "computational" game theory is meant to encompass a wide variety of games that lend themselves to brute force or artificially intelligent opponents. Here we often distill the combinatorial set of values for a game to three cases: win, lose or tie. With small games that can be exhaustively searched, we are able to search the game tree in its entirety and provide a perfect opponent. With larger games, it is necessary to create *static evaluators* that return how good a position is for a particular player. The system then uses this information to control a limited search of the game tree to determine the best computer's move.

Robbie Bell and Michael Cornelius have provided a great reference for board games played around the world that could be used in these systems [1]. These games serve as some of the most interesting projects and examples for courses in artificial intelligence, networking, software engineering and graph theory.

Conclusion

Games provide a wonderfully rich and fertile source of new programming projects, illustrations of fundamental CS principles, and motivational problems. The goal of this special session is to bring those attending "onto the same square", and provide them with the tools to tap this valued resource.

References

1. Bell, R and Cornelius, M. *Board Games Round the World: A Resource Book for Mathematical Investigations*. Cambridge University Press, 1988.

2. Berlekemp, E., Conway, J., and Guy, R. *Winning Ways for Your Mathematical Plays, Volume 1 & 2*. Academic Press Inc., 1982.

3. Binmore, K. *Fun and Games: A Text on Game Theory*. D. C. Heath and Company, 1992.

4. Garcia, D. Nifty Assignments – Shall We Play a Game?, *SIGCSE* (2002), Cincinnati, OH.

5. Ginat, D. Birds of a Feather – Mathematical Games as an Aid for CS Instruction, *SIGCSE* (2001), Charlotte, NC.

6. Ginat, D. Loop invariants and mathematical games, International *Journal of Mathematical Education in Science and Technology*, *32, 5,* Taylor & Francis, (2001), 635-651.

7. Nowakowski, R. *Games of No Chance*. Cambridge University Press, 1996.