The International General Game Playing Competition

Michael Genesereth
Computer Science Department
Stanford University

Yngvi Bjornsson
School of Computer Science
Reykjavik University

1. Introduction

General game players are systems able to play strategy games based solely on formal game descriptions supplied at "runtime". (In other words, they don't know the rules until the game starts.) Unlike specialized game players, such as Deep Blue, general game players cannot rely on algorithms designed in advance for specific games; they must discover such algorithms themselves. General game playing expertise depends on intelligence on the part of the game player rather than intelligence of the programmer of the game player.

General Game Playing (GGP) is in many ways similar to AutonomousPlanning. The description of a game in the Game Description Language (GDL) is similar to that in the languages used by planners (such as PDDL); and the overall goal is the same - to achieve a state with specified properties. One obvious difference is that, in GGP, there are opponents, which complicates the process of determining an ideal course of action. Another difference is that, in GGP, there is an execution environment, making it possible for a game player to interleave planning and execution. Also, in GGP, there are time constraints, which make it essential for players to act even when they are unsure which courses of action are best.

General Game Playing is also related to Game Theory, since both are concerned with games. However, again, there are differences. In Game Theory, a game corresponds to a game tree, and there is little or no attention to how games are communicated to game players. In general game playing, the description is essential; different descriptions correspond to different games. Also, Game Theory often makes assumptions about the rationality of the players, whereas, in General Game Playing, these assumptions are less common - the opponents might not be rational at all or they may have crashed or lost connectivity to the game manager.

General Game Playing is an interesting application in its own right. It is intellectually engaging and more than a little fun. But it is much more than that. It provides a theoretical framework for modeling discrete dynamic systems and for defining rationality in a way that takes into account problem representation as well as complexities like incompleteness of information and resource bounds. It has practical applications in areas where these features are important, e.g. in enterprise management and computational law. It is also concerned with applications of AI technology in the real world, such as how to learn from experience and act autonomously in novel environments in real time. More fundamentally, it raises questions about the nature of intelligence and serves as a laboratory in which to evaluate competing approaches to artificial intelligence.

2. The International General Game Playing Competition

In order to promote progress on GGP, the AI community in 2005 established the International General Game Playing Competition, and it has run annual competitions ever since. The competitions are typically associated and co-located with either the AAAI conference or IJCAI each year.

Table 1 shows the winners of the competition over the years. Mostly different players in different years with the notable exception of CadiaPlayer, which has won three times.

Year	Game Player	Developer(s)
2005	Cluneplayer	Jim Clune
2006	Fluxplayer	Stephan Schiffel, Michael Thielscher
2007	Cadioplayer	Yngvi Bjornsson, Hilmar Finnsson
2008	Cadioplayer	Yngvi Bjornsson, Hilmar Finnsson
2009	Ary	Jean Mehat
2010	Ary	Jean Mehat
2011	TurboTurtle	Sam Schreiber
2012	CadioPlayer	Yngvi Bjornsson, Hilmar Finnsson

In recent years, the Competition has included a man-machine demonstration match pitting the competition winner against a human player. While the human player won the first of these demonstrations, the computer has won all of the matches since. In 2012, CadiaPlayer, in addition to defeating the other competitors also defeated the human race (represented by Chris Welty) in the post-competition Carbon versus Silicon match-up. See photo below. (As a consolation prize, the human was awarded two bottles of Scotch, in part to ease his disappointment at letting down the human race.)

2012 Carbon vs Silicon match: The human (right) taking counsel from another human (left)

A related development is the availability of a massive open online course (mooc) of general game playing, aimed at exposing the field to tens of thousands of students and preparing them to participate in the annual competition. The first of these moocs is scheduled to run on the Coursera platform in the Spring of 2013. See https://www.coursera.org/course/ggp.

3. Brief Overview of General Game Playing

General Game Playing is concerned with finite, synchronous games. These games take place in an environment with finitely many states, with one distinguished initial state and one or more terminal states. In addition, each game has a fixed, finite number of players; each player has finitely many possible actions in any game state, and each state has an associated goal value for each player. The dynamic model for general games is synchronous update: all players move on all steps (although some moves could be "no-ops"), and the environment updates only in response to the moves taken by the players.

Since all games in GGP are finite, it is possible, in principle, to describe such games in the form of lists of states and actions and tables or graphs to express legality, goals, termination, and update. Unfortunately, such explicit representations are not practical in all cases. Even though the numbers of states and actions are finite, they can be extremely large; and the tables relating them can be larger still. For example, in chess, there are thousands of possible moves and more than 10^30 states.

In the vast majority of games, states and actions have composite structure that allows us to define a large number of states and actions in terms of a smaller number of more fundamental entities. In chess, for example, states are not monolithic; they can be conceptualized in terms of pieces, squares, rows and columns and diagonals, and so forth. By exploiting this structure, it is possible to encode games in a form that is more compact than direct representation. The Game Description Language (GDL) supports this by relying on a conceptualization of game states as databases and by relying on logic to define the notions of legality, reward, termination, and so forth.

The process of running a game goes as follows. Upon receiving a request to run a match, a program called a Game Manager first sends a "Start" message to each player to initiate the match. The start message lists the name of the match, the role the player is to assume (e.g. white or black in chess), a formal description of the associated game (in GDL), and the "start clock" and "play clock" associated with the match. The start clock determines how much time remains before play begins. The play clock determines how much time each player has to make each move once play begins. Once game play begins, the Game Manager sends "Play" messages to each player to get their plays, and it then simulates the results. This part of the process repeats until the game is over. The Manager then sends Stop messages to each player.

Having a formal description of a game is one thing; being able to use that description to play the game effectively is something else. Since game descriptions are written in logic, game players obviously require some degree of automated reasoning. The good news is that there are powerful reasoners for GDL. The bad news is that such reasoners do not, in and of themselves, solve the real problems of general game playing, which are the same whatever representation for the game rules is used, viz. dealing with indeterminacy and size and multi-game commonalities.

4. Progress in the Field

Over the years of the competition, general game players have become more sophisticated and significantly more powerful. There is no question that today's players can easily beat players developed early on. Partly, this has been due to tuning and tweaking, but there have also been significant innovations that have dramatically improved performance. The following are notable in this regard.

Game-Independent Heuristics. The first GGP programs introduced game-independent heuristics to deal with limited search. These included things like mobility (the number of legal moves), inverse mobility (limiting the opponents' freedom, and goal proximity (similarity of intermediate states to goal states). While such heuristics are generally better than random play, they do not perform well in all games.

Learning Weights on Game Playing Heuristics. In order to deal with the deficiencies of game-independent heuristics, some early players utilized the start clock period to play games and assign weights to different general heuristics, and these weights were then used during the play clock period to differentiate moves. This helped quite a bit and led Cluneplayer to victory in the first Competition. Unfortunately, the method is error-prone. In the final game of the second competition, Cluneplayer heavily weighted inverse mobility of its opponent. Sadly, it was a variant or checkers with forced moves, and the best way Cluneplayer could find to limit its opponent's moves was to sacrifice pieces (in most cases without the opportunity to re-capture).

Monte Carlo and UCT. The most significant improvement in GGP came from the introduction of Monte Carlo Tree Search methods and UCT. Rather than using general heuristics, MC uses runtime statistics to estimate the quality of a state, dropping a number of random "depth charges" to the bottom of the game tree and averaging the results. UCT improved on this by replacing the random choice by more intelligent selections. The effect was dramatic. Suddenly, automated general game players began to perform at a high level. Using this technique Cadioplayer won the competition 3 times. Almost every general game playing program today uses some version of Monte Carlo and UCT.

Structural analysis. In many games it is possible to discern structure that can be used to decrease the combinatorics of the game. Consider, for example, the game of hodgepodge, which is a combination of in traditional games. If the player does not recognize that it is made up of independent sub-games, it is going to search a space in which the branching factor is the product of the branching factors of the individual games. If it is able to "factor" the game description, it can solve the sub-games independently and dramatically decrease search cost. In many cases, it is possible to find such factors in time proportional to the size of the game description rather than the size of the game graph. So there is substantial economy to be gained in doing such analysis. The competition has just begun emphasizing games with structure of this sort. Algorithms for funding such structure have been published in the literature but so far have not been used in competition.

Compilation. Game descriptions are written in logic and automated reasoning techniques can be used in generating game trees. However, GDL descriptions are very simple (essentially pure Prolog) and can be compiled into more efficient programs. Compilation does not change the asymptotic behavior of the players, but it can improve performance by orders of magnitude. Moreover, since games are finite and completely described, game descriptions are equivalent to boolean circuits. The upshot is that they could, in principle, be compiled into hardware using filed programmable gate array for even more performance improvement. While this has not yet been tried in competition, it remains a powerful idea.

5. Conclusion

Unfortunately, while some interesting technology has emerged from work on GGP, it has not yet found application outside of GGP. It is still early in the field, and this is likely to change as games are chosen that more closely resemble real-world problems.

Perhaps the biggest problem with GGP at the moment is the name. It suggests that the task is frivolous, when in point of fact many real world problems can be cast as games. The organizers have repeatedly toyed with the idea of renaming the competition General Problem Solving, but it seems that the name GGP is too entrenched to allow this.

Meanwhile, work goes on. There is already a well-established research community working on GGPm resulting in numerous publications (including several doctoral theses). Also, in addition to the International competition, several other GGP events are regularly hosted, including various national GGP competitions and the biennial GIGA workshop. Even with the current formulation of the field there is still room for progress. Once this subsides, there are several variations waiting in the wings.

General Game Playing with incomplete knowledge. In current GGP, players do not know the moves of their opponents (in advance), but they know the full details of the game world. In GGP with incomplete knowledge they do not even have complete information about the game world. For example, they may not know the initial state (as in Battleship). Or there may be probabilistic elements, as in card games. Already two languages have been developed for such games (IGDL and GDL-II), and there are some rudimentary players capable of playing games described in these languages.

Inductive General Game Playing (IGGP). The main innovation in IGGP is that the players are not provided with rules but only instances of games and they are left to induce the rules for themselves.

Really General Game playing (RGGP). Really General Game Playing takes this progression one step farther. In RGGP, the players are given a characterization of sensors and effectors and a utility meter. Their goal is to function in the world in such a way as to maximize their utility, knowing *nothing* else about the world. Not likely to be worked on soon, though some students have experimented with various approaches that could be applied.

For more details on General Game Playing and the International GGP Competition, visit the competition website (http://logic.stanford.edu/games). Other valuable GGP resources include http://www.general-game-playing.de and http://www.ggp.org.

References

Clune, J. 2007. Heuristic evaluation functions for General Game Playing. In Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, 1134-1139.

H. Finsson, Y. Bjornsson. Simulation-Based Approach to General Game Playing.

M. Genesereth, N. Love: Overview of the AAAI General Game Playing Competition. *AI Magazine* 26(2):62-72. (http://logic.stanford.edu/classes/cs227/2012/readings/aaai.pdf)

Kuhlmann, G.; Dresner, K.; and Stone, P. 2006. Automatic heuristic construction in a complete general game player. In Proceedings of the Twenty-First National Conference on Artificial Intelligence, 1457-62.

Love, N.; Hinrichs, T.; and Genesereth, M. 2006. General Game Playing: Game description language specification. Technical Report April 4 2006, Stanford University.

Schiffel, S., and Thielscher, M. 2007. Fluxplayer: A successful general game player. In Proc. of the Twenty-Second AAAI Conference on Artificial Intelligence, 1191-1196.

Michael Genesereth is an associate professor in the Computer Science Department at Stanford University. He received his Sc.B. in Physics from M.I.T. and his Ph.D. in Applied Mathematics from Harvard University. Prof. Genesereth is most known for his work on computational logic and applications of that work in enterprise computing, computational law, and general game playing. He is the current director of the Logic Group at Stanford and founder and research director of CodeX (The Stanford Center for Legal Informatics). He initiated the International General Game Playing Competition in 2005.

Yngvi Bjornsson is is an associate professor in the School of Computer Science at Reykjavik University. He received his Ph.D. in Computer Science from University of Alberta, Canada. He has been active in the computer games research community for many years is for example the co-author of the CadiaPlayer GGP agent. He is a co-founder and the current director of the CADIA research lab at Reykjavik University.