On Meta-Game Approaches to Computer Go

Bruno Bouzy

LAFORIA-IBP, Université Paris VI
Tour 46 2ème étage 4, place Jussieu
75252 PARIS FRANCE
Tél.: 44 27 70 10 Fax: 44 27 70 00
e-mail : bouzy@laforia.ibp.fr

Abstract

Meta-game approaches are studied through the design of the Go playing program INDIGO. Computer Go is a challenge for AI and the machine still remains weak compared to humans in the game of Go. This difference is used to try new technics as such as the Meta-game technique discussed in this paper. A conceptual structure and a notion of Conway's game theory are described in order to show four approaches. Each approach is presented to show its method using games, its computational cost, its precision and its Meta-game's utilisation. This paper is an illustration of Meta through the game of Go.

Keywords :

Meta-game, game of Go, abstraction, conceptual structure, game, sub-game.

Introduction

Go players think not only about the whole position in front of them, but also about sub-positions on which they act. In Go game, every couple of opposite actions can be modelled as a "game" that is a sub-game of the whole game. Go players have the ability to step back from their main activity on a game and consider the state of that activity and how it is being conducted. The process by which Go players think about the playing activity itself is called Meta-level game playing.

Serious efforts to implement computational game Meta-reasoning began in the field of Computer Game. Jacques Pitrat [Pitrat 1990] presented methods to introduce Meta-reasoning in several domains and in Computer Chess programs. Meta-level reasoning has also been researched in games with Barney Pell [Pell 1992]. Given that the field of Computer game is still striving for breakthroughs in representational and software engineering, it is natural to look for Meta-game approaches.

In our past thesis, we modelled behaviour of human Go players [Bouzy 1995]. Amongst other things we tried to link past works on Meta-level reasoning with Computer Go. Particularly, we approached a definition and properties of a Meta-game concept. Parts of our ideas on Meta-game are implemented in our Go playing program INDIGO (Is Now Designed In Good Objects). Other parts are remaining at the idea's level. The aim of this paper is to show all the different approaches to the concept of Meta-game, implemented or not. We hope these approaches can be useful to other Computer Game scientists.

In the first part of the paper, we recall state of the art in Computer Go. There, we speak of static evaluations on a static decomposition of the Go board. Then we present our own utilisation on Conway's games theory, with our definition of a "game". In the second part of the paper, we show four approaches to link a "game" with its "sub-games". We classify which approaches include or not the "Meta-game" concept. Then we conclude.

Computer Go

State of the art :

Computer Go remains as a challenge [Brown & Dowsey 1979] [Bouzy & Victorri 1992] for AI. As a matter of fact, games as Othello, Checkers and Chess, are good examples of success of AI : Othello programs are clearly stronger than the human world champion, the best Checkers program gets the same level as the world champion, and Chess program gets the level of International Masters [Allis 1994]. The main reason is that positions in these games are more or less easily and quickly judged by an evaluation function. Then, tree explorations are possible and give good results in tactical positions. The second reason is that height of explored trees are computable by today's machines. In Go, these two reasons are worthless and the best program still gets the level of a fanatic beginner in a club, about 10th kyu [Boon 1991]. Nobody knows how to give an evaluation of a Go position to a machine and, even if it was so, the height of explored trees would be an other obstacle. Therefore, since the beginning of Go programming in 1960, most *programmers model human Go players to program Go.*

Human Go players use their real world representation to play Go. The aim of this paper is not to show *what* is this representation, but to show the ludic aspect of *how* it behaves. The representation is structured and permits to qualify statically the Go board.

Static qualifying of a Go board :

Conceptual structure :

Let's suppose the Go board is conceptually the following.

Firstly, 4 "groups" are on the board as shown on figure a.

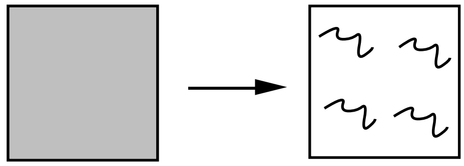

figure a

Secondly, each group gets 2 "eyes" as shown on figure b.

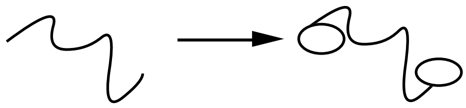

figure b

Each eye is terminal in the structure. Figure c shows the whole structure of the position.

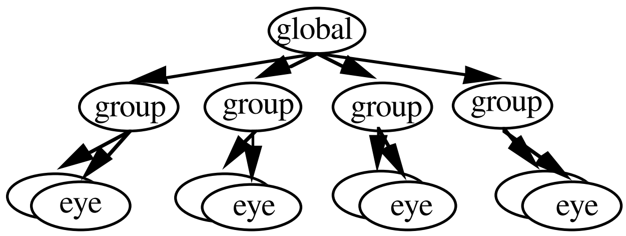

figure c

It is not the aim of the article to show what is an eye, nor a group. This is a very difficult part of Computer Go, which we do not present here. So, in this paper, *let's suppose structure of figure c is given.*

Static evaluation :

Structured position of figure c is statically evaluated with the following rules. Each node (or object) of the structure is statically evaluated with three values : WON, LOST, OTHER.

Eye 's evaluation

An eye gets a color. It is WON if three of the four diagonal intersections are with the same color. It is LOST, if the four intersections are occupied with a stone, without being in the previous case. It is OTHER, in other cases. We give an intuitive approach on figure d.

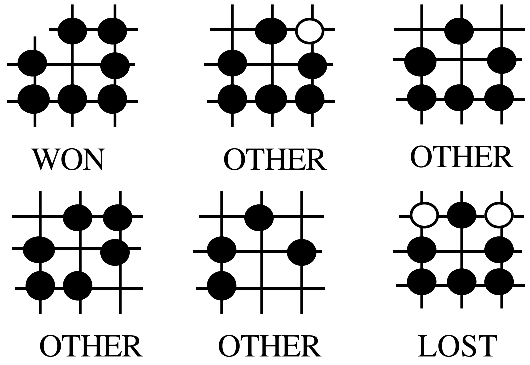

figure d

Group evaluation

Group evaluation depends on the state of its two eyes. If the group gets two eyes, its state is WON. If the group gets only one eye, its state is LOST. In other cases, group's state is OTHER.

Global evaluation

Global evaluation for a color C depends on contributions of groups. If a group is WON (resp. LOST) and with color C (resp. with other color than C) or LOST (resp. WON) and with other color than C (resp. with color C) its contribution is +1 (resp -1). Global evaluation for a color C equals the sum of all groups' contribution.

A "Game"

Most of Go playing programs use some of Conway games theory [Conway 1982] to represent dynamic[1] evaluation of positions. Our program INDIGO does the same. A "game" gets a dynamic state that can take four values according who turns first. Here is a summary of our utilisation of Conway 's theory :

> : won even if the opponent begins,
* : first player who plays wins,
< : lost even if the player begins,
0 : first player who plays loses.

This dynamic information is computed with lookahead on static information. With this instance of a conceptual static structure and the notion of "game", we get the two ingredients to go further. We can present different approaches combining differently these two ingredients.

Toward a Meta-game ?

Meta

We will say that *something* is a *Meta-something* when it applies to *somethings* of the same type. For instance, a Meta-knowledge is a knowledge applying to knowledge.

Four different dynamic approaches

Approach-zero :

This approach is the one people use to show that brute force used in Chess cannot work in Go. This approach doesn't use the static conceptual structure. There is only one "game", the global game. Its evuation function is simple : +1 (resp. -1) if the intersection is occupied by a black (resp. white) stone or empty but surrounded with black (resp. white) stones. Generated moves are the ones generated by the rules. The size of the tree of the whole game is about 361.360.359...2.1 i.e 361! on a 19-19 board. Figure e explains visually this catastroph on a 9-9 board...

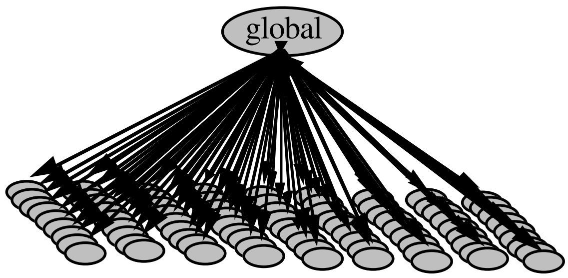

figure e

Approach-one :

This approach uses a little the static conceptual structure. There is only one "game" again, the global game. Its evaluation function uses the conceptual structure : +1 (resp. -1) if the intersection is occupied with a black (resp. white) group WON or a white (resp. black) group LOST. But the generated moves are the

[1] We use the term "dynamic" when lookahead is necessary.

ones generated by the leaves of the structure. Figure f shows this fact.

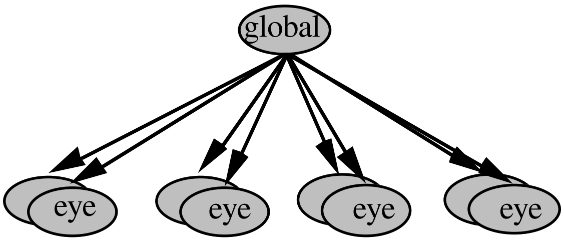

figure f

If each eye can generate four moves average, the size of the tree is about (number-of-groups.number-of-eyes-to-evaluate-per-group.number-of-moves-on-a-leaf) ! = (4.2.4) ! = 32! It is too much again.

Approach-two :

This approach uses the conceptual structure. There is a global "game", there are four group "games". For each group "game", there are two eye "games". I.e. ludic structure is the same as the conceptual static structure as shown on figure g.

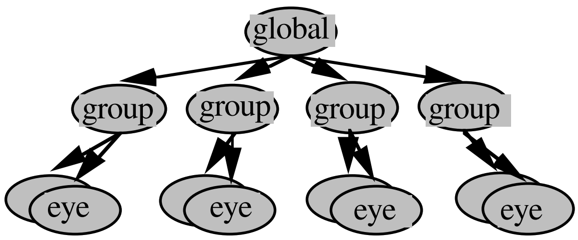

figure g

Each non terminal game is therefore constituted with sub-games. To know the state of a terminal game, one computes with lookahead techniques. To generate moves of a non terminal game, one looks at sub-games with dynamic state * and one takes moves advised by these sub-games. If there is no * sub-games, one uses dynamic recomposition rules which permit to know the dynamic state of a game knowing the dynamic states of its sub-games. For instance, at the conceptuel level of groups : if a group gets two (eye) sub-games with dynamic state >, its static evaluation is WON. If, the group gets two (eye) sub-games with dynamic state <, its static evaluation is LOST. In other cases, it is OTHER. Figure h shows instances where such rules apply.

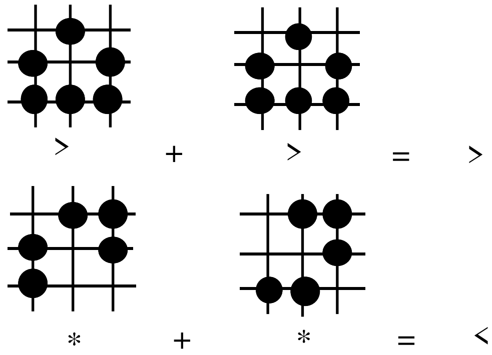

figure h

Lookahead cost is 4.2.(4!) = 8.4! = 192 (number-of-groups . number-of-eyes-per-group . (average-number-of-moves-generated) !). That is pratically right and it is implemented in our Go playing program INDIGO.

Of course, recomposition rules are false if sub-games are dependant.

Approach-three :

This approach uses the conceptual structure again. There is one global game with four "group" sub-games, with two "eye" sub-games for each. Ludic structure follows exactly the static structure again. But this approach is different from approach-two because when recomposition rules cannot apply, as on figure i, approach-three uses another method to open the sub-games in order to find a more accurate result.

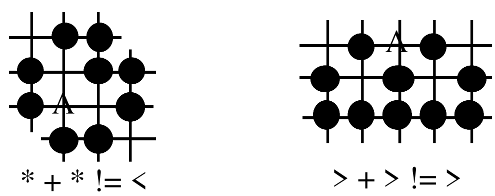

figure i

To know the dynamic state of a terminal game, one computes lookahead. To generate moves of a non terminal game, if its sub-games are in dynamic state *, one takes advised moves. If not, one uses recomposition rules of approach-two. If they are dependant, one takes double threat on each sub-games (one generates double threats taking the intersection of the sets of moves generated separately). Figure j shows this refinement with the "horizontal" double arrows between dependant sub-games.

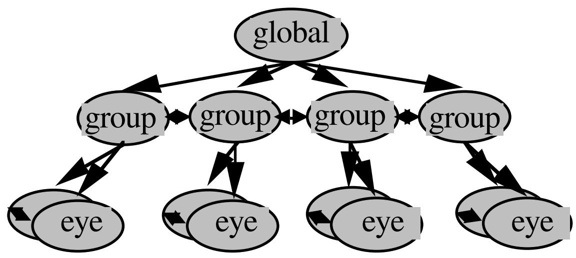

figure j

Lookahead cost is between the cost of aproach-one and approach-two. The result is more precise than in approach-two. Its difficulty is its implementation. In our program INDIGO, it stays as a perspective.

Wich games are Meta-games ?

Playing moves or playing games ?

Let's forget approach-zero which was only for reference.

The particularity of approach-one is to compute lookahead on the global game, playing moves generated with leaves of the conceptual structure and then to evaluate after. This approach is stupid but could be efficient if computers were more powerful. In the approach-one, the player plays *moves*.

Approach-two uses the abstraction mechanism. Moves of an underground level are hidden to the appearant level. In this approach, the player plays "games" (instead of real moves). In that sense, approach-two uses Meta-game.

Approach-three uses Meta-game in a stronger sense because a (Meta-)game does, not only use the result of dynamic states of its sub-games (as in approach-two), but also asks specific lookahead computations to its sub-games according to the dependency between sub-games.

Which approach to use ?

If the sub-games of a level are independant, approach-two is recommended.

If the sub-games are very dependant, approach-two is false. Approach-three needs N lookahead computations (one for each sub-game) to get the dynamic state. As sub-games are very dependant, the N computations are very similar. Then, one does not use dynamic recomposition rules; instead, one does another computation very similar to the computation one should have done in approach-one : when games are very dependant, approach-one is recommended.

In the real case, you don't know, a priori, if sub-games are independant, dependant or very dependant; so, approach-three is recommended.

Where is Meta-game ?

If we define a Meta-game as a game where the player acts in choosing a game (instead of choosing a move in a simple game), then approach-two and approach-three contain Meta-games.

If we define strongly a Meta-game as a game where the player acts in choosing a sub-game and also the appropriate solving methods for this sub-game, then approach-three only contains Meta-games.

Conclusion

In this paper, we presented how Meta-game concept can help Computer Go in refining "game" concept. We show how four different ways of applying a ludic taxonomy on a prealable conceptual taxonomy can give four different approaches. We presented each approach with relative computational costs, precision of the result and utilisation of Meta-game concept or not.

In our software INDIGO, approach-two is used. Approach-three is a perspective to better results of our software. In 1995, INDIGO is about 18th kyu. We evaluate its level playing games against strong american programs as MANY FACES OF GO from David Fotland [Fotland 1992] and POKA from Howard Landman.

Anyway, understanding of the mechanisms of different approaches to evaluate a Go position will permit to refine a definition of Meta-game in the game of Go. This illustration will be an useful application of the very general concept of Meta.

Bibliography

[Allis 1994] - L. V. Allis - Searching for Solutions in Games and Artificial Intelligence - PhThesis - Vrije Universitat Amsterdam - Maastricht - September 1994

[Boon 1991] - Mark Boon - Overzicht van de ontwikkeling van een Go spelend programma - Afstudeer scriptie informatica onder begeleiding van prof. J. Bergstra - 1991

[Bouzy & Victorri 1992] - B. Bouzy, B. Victorri - Go et Intelligence Artificielle - Revue de l'AFIA n. 10 - Juillet 1992

[Bouzy 1995] - B. Bouzy - Modélisation cognitive du joueur de Go - Thèse de doctorat de l'Université Paris 6 - Janvier 1995

[Brown & Dowsey 1979] - D.J.H. Brown S. Dowsey - The challenge of Go - New Scientist, 81, 1979

[Conway 1982] - J. Conway, E.R. Berlekamp, R. Guy - Winnings ways - tome 1 & 2 - Academic press - 1982

[Fotland 1992] - D. Fotland - Many Faces of Go, documentation and playing algorithm - 1992

[Pell 1992] - B. Pell - Metagame : A New Challenge for Games and Learning - Heuristic programming in Artificial Intelligence 3 : The Third Computer Olympiad - H.J. van den Herik & L.V. Allis, editors - Ellis Horwood - 1992

[Pitrat 1990] - J. Pitrat - Métaconnaissance - Hermès 1990