Development and Evaluation of Strategic Plans

Tristan Cazenave Régis Moneret

LIP6
Université Pierre et Marie Curie
4, Place Jussieu, 75005 Paris, France

e-mail : {Tristan.Cazenave,Regis.Moneret}@poleia.lip6.fr

Abstract

At the strategic level, a Go program has to manage uncertainty because of the difficulty to correctly evaluate middle game positions (strength of groups, battles). It has to be cautious not to rely on too many uncertain assumptions, otherwise its opponent will find a weakness in the plan.

When faced with multiple choices for achieving a given strategic goal, we provide a method for assessing the least hazardous plan (a plan is a subtree of goals that leads to the success of the root goal). We combine AND/OR tree search with probability estimations of success of the achievement of the goal. Intuitively, errors cumulate at AND nodes because different conditions have to be satisfied at the same time.

1 Introduction

In the game of Go, middle game positions involve many fights between groups with uncertain status. Evaluating and correctly managing the strength of groups is crucial to handle battles. Making uncertain assumptions is necessary, but a program has to be cautious not to rely on too many uncertain assumptions, lest its opponent will find a weakness in the plan.

When faced with multiple choices for achieving a given strategic goal, we provide a method for assessing the least hazardous plan (a plan is a subtree of goals that leads to the success of the root goal). We combine AND/OR tree search with probability estimations of success of the achievement of the goal. Intuitively, errors cumulate at AND nodes because different conditions have to be satisfied at the same time.

In the first part we acknowledge related works on strategic planning in Chess and Go. Then, we present our method to evaluate strategic plans. In the following part, we provide an example of the application of our method on a Go board. Eventually, we explore ways to develop efficiently our strategic trees without evaluating unnecessary leaves.

2 Related works

Works on strategic plans in games can be traced back to Robin [Pitrat 1977], this work has been followed by [Wilkins 1980]. Concerns about strategic planning in the game of Go appeared in [Fotland 1993], where fuzzy status of groups were used to make strategic decisions. [Bouzy 1995] developed further the strategic part involved in Go programs and managed relations between groups with fuzzy status. [Ricaud 1995] on the contrary, focused more on abstracting strategy in fuzeki than on reasoning about groups. [Cazenave 1996] handles uncertain status of groups, each group is constructed using learned rules about connections and eyes. Tactical learned rules are used to evaluate and choose the strategic

moves involving groups. [Moneret 1996]'s system enables to generate automatically the incremental rules used to learn the tactical rules, given the rules of the game of Go .

3 A method to evaluate strategic plans

The system is given a set of strategies. Each strategy is represented as an AND/OR tree containing tactical goals. For example the goal of saving a group G in the game of Go can be represented as :

OR (Make_Two_Eyes (G), AND (Neighbor_Of (G G1), Enemy (G G1), Kill (G1)), AND (Connect (G G2), OR (Two_Eyes (G2), Lot_Of_Territory (G2))))

We associate to each leaf goal (i.e. Make_Two_Eyes (G)) a success rate representing the frequency this goal succeeds (10% of the time). The problem is to assess the best branch for saving group G.

We make the assumption that each goal is independent of the others.

The previous assumption allows us to evaluate the success rate of the AND node as the product of the success rates of its children.

At each OR node, we can establish a minor bound on the success rate of the node by taking the maximum of the success rates of its children.

This enables us to assess a success rate for each branch of the root OR node, and thus to choose the plan which is more likely to succeed.

4 An example

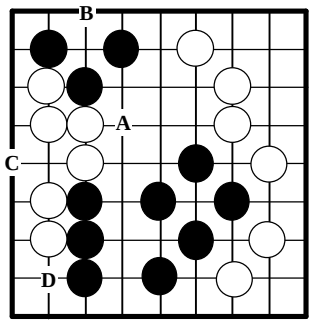

Figure 1

For instance, we apply the above goal on the particular position of Figure 1, with G the black upper left group, G1 the white group on the left and G2 the black group below.

On the example of Figure 1, we have the following success rates :

Make_Two_Eyes (G)	-> 10%
Neighbor_Of (G G1)	-> 100%
Enemy (G G1)	-> 100%
Kill (G1)	-> 30%
Connect (G G2)	-> 80%
Two_Eyes (G2)	-> 70%
Lot_Of_Territory (G2)	-> 10%

On our example, move A connects groups G and G2 with 80% chances of success. Move B makes two eyes for G with 10% chances of success. Move D kills G1 with 30% chances of success. We also know that G1 can live by playing at C.

We can then compute the success rates of each node :

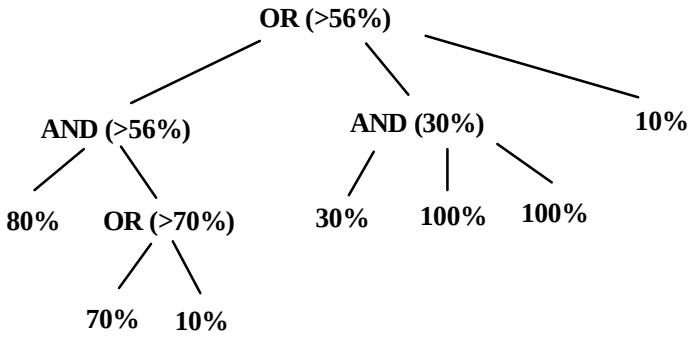

Figure 2

In this case, the algorithm chooses the rightmost branch, which seems in this position to be the most promising one.

Besides, cuts can be made in the development of the strategic plans. At AND nodes, it is not necessary to calculate the success rates of all children when one has already a success rate inferior to the current success rate of the OR node above.

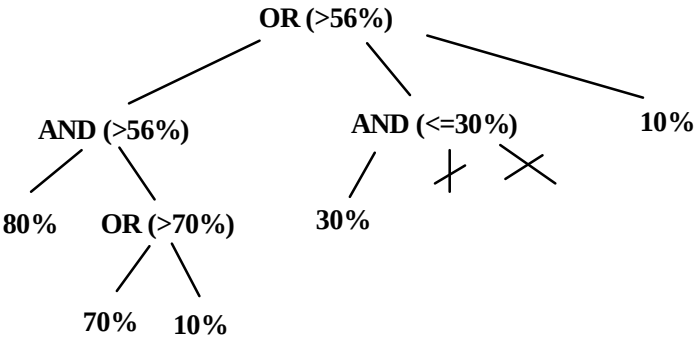

Figure 3

The program begins by calculating the leftmost leaves of the tree, and propagates up to the root the bounds of the value of the above nodes. Before each leaf evaluation, it checks that this

evaluation is not a waste of time. At AND nodes, it verifies that the already computed value associated to the AND node is higher than the value associated with the OR node above the AND node. Otherwise it is not useful to compute the leaves and the subtrees under the AND. Similarly, when a leave has a success rate of 100% under an OR node, it is not useful to compute the other leaves and subtrees under the OR node, the OR node will always be 100%.

In figure 3, this optimization enables the system to cut two branches under the second AND node.

5 Possible improvements

Moreover, a sorting algorithm can be used so as to give us the best node to develop first in an incomplete tree (i.e. the node which will lead to the maximum of cuts). This algorithm consists in developing the branch that has the least number of AND branches.

When given the following partial tree :

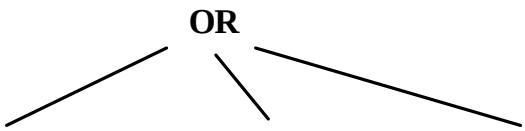

Figure 4

It begins with developing one step further the leftmost branch since the three OR branches are equivalent :

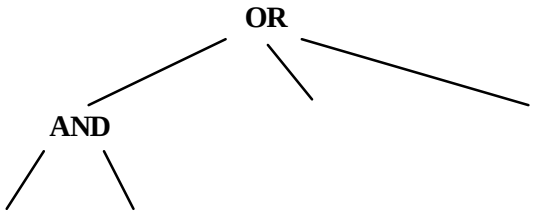

Figure 5

It finds that there are two AND nodes, versus 0 in the other branches. Therefore, it develops the other branches.

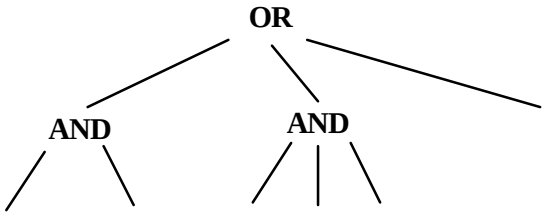

Figure 6

It finds that there are three AND nodes in the second branch, versus 0 in the last. Therefore, it develops the last one.

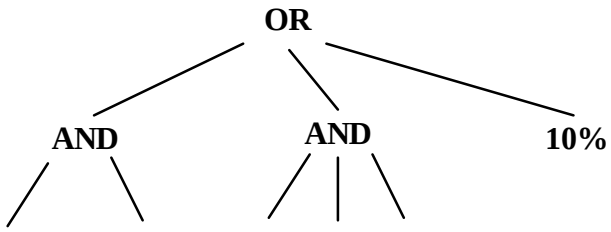

Figure 7

Now, the algorithm will choose to develop the leftmost branch first because it has only two AND branches, versus three for the second one

This algorithm ensures to make the cuts even if the leftmost branch is the branch which has three AND branches. That is not the case when we always develop the leftmost branch as in a depth-first algorithm.

6 Conclusion

In conclusion, we have formalized a method to evaluate imperfect strategic plans, and choose among these plans. Moreover, we have shown how to efficiently develop these AND/OR trees using a best-first algorithm. Efficiently and correctly managing uncertainty is crucial to strategy in the game of Go.

References

[Bouzy 1995] - Bruno Bouzy. *Modélisation cognitive du joueur de Go*. Thèse de l'université Paris 6, 1995.

[Cazenave 1996] - Tristan Cazenave. *Système d'Apprentissage par Auto-Observation. Application au Jeu de Go*. Thèse de l'Université Paris 6, Décembre 1996.

[Fotland 1993] - David Fotland. *Knowledge Representation in The Many Faces of Go*. Second Cannes/Sophia-Antipolis Go Research Day, Février 1993.

[Moneret 1996] - Régis Moneret. *Mise à jour incrémentale des concepts du jeu de Go*. Rapport de stage du D.E.A. I.A.R.F.A., 1996.

[Pitrat 1977] - Jacques Pitrat. *A Chess Combination Program which Uses Plans*. Artificial Intelligence, vol. 8, p 275-321, 1977.

[Ricaud 1995] - Patrick Ricaud. *GOBELIN : Une Approche Pragmatique de l'Abstraction Appliquée à la Modélisation de la Stratégie Elémentaire du Jeu de Go*. Thèse de l'Université Paris 6, Décembre 1995.

[Wilkins 1980] - David Wilkins. *Using Patterns and Plans in Chess*. Artificial Intelligence, vol. 14, p 165-203, 1980.