Amateurs' Scientific Contribution to the Future of Go

Robert Jasiek

Abstract

Amateurs have contributed a lot to Go as a Science. During the last decade their valuable activity has grown considerably because of the Internet. So time is ripe for a cross-section of their work. Many detailed studies can be found in various publications. Therefore this overview is broad rather than deep. It offers a wide range of new approaches and methods suitable for scientific study of Go. The discussed topics include Go rules mathematics, game trees, calculation size in computer Go, and functional languages.

For the modern study field Go rules mathematics it is shown that Go can be well-defined as a finite game, how complex Go is, that a particular rule can nicely simplify study, and how a general result about cycles looks like. It is well known that game trees are used to represent reading in computer Go. However, they are flexible and also allow methodical analyses of endgame values, eyespace values, status types, or plays in pass fights. Computer Go suffers from exploding calculation sizes. Restrictions are possible in case of the examples ladders, capturing after a local series of threats, equal approach moves, or eye shapes. Finally, using joseki as an example functional languages formalize strategy by describing the meanings and intentions of single plays, sequences of plays, groups, joseki as a whole, and choices within joseki in contrast to playing elsewhere.

1 About the Contribution of Go Rules Mathematics

Go rules mathematics is a modern specialized field of applied mathematics. It methodically studies Go problems that are directly related to rules questions. Go is a game defined by its rules and therefore all Go study relies on them. In practice most practical Go theory used by players is on a high level of abstraction far above fundamental rules application. Only since the early 20th century rules have been studied methodically. Therefore Go rules mathematics is still in its infancy and much remains to be discovered. In the following some research examples are given. The chapter relies mainly on Go rules mathematics.

1.1 The Definition of Go

When Go is studied as a science this must rely on a well-defined description of the game. "well-defined" means that what is defined can always be applied, e. g. a turn is always possible and the score at the game end can always be determined, either regardless of the position. We show that the game of Go can be well-defined by giving an example of such rules. Mathematical precision could be assumed for definitions of the terms turn, board, intersection, colour (black or white), stone, player, initial player, position (configuration of all stones on the board), initial position, surrounded, play (allowing removal of own stones, e. g.), pass (not playing during a turn). However, the reader will only be confronted with the following rules using them:

1. The players have alternate turns.
2. On his turn the player either plays or passes.
3. Alternation of turns ends with two successive passes.
4. After any play an earlier position may not be recreated.
5. When alternation of turns has ended the winner is the player with the higher number of intersections occupied by his stones plus intersections that are without stones and surrounded by intersections occupied by his stones. The game is a tie if the players have equal numbers.

Rule 4 is called positional "superko". – These rules can be applied by Go players, for study of Go rules mathematics, for computer Go, and a lot of other formalized Go research like study of game trees. The scope of applications is so broad because the rules are well-defined and thus can handle all positions without any exception.

1.2 Finity of Alternation

Although rules for mathematical study need not have the feature of the following proposition, it is very useful for many applications like further study of Go rules mathematics. E. g. in a game tree every path ends in a leave and thus can have value.

PROPOSITION: Alternation of turns in a game using the rules in 1.1 always ends.

The proof uses the fact that only a finite number of positions exist. Due to the superko rule the player having the turn must pass finally. – Practically, if the players do not pass earlier, then a position will be reached in that the superko rule prohibits play on any intersection because removing a single own stone would recreate the position before that play.

1.3 About the Complexity of Go

While the rules in 1.1 are very simple, we all know that Go is extremely complex and use that fact for promoting the game. What is "complex"? This can be best described by the number of legal games because when trying to win one has to make a good choice among them. It is particularly hard to determine the number of legal games. Upper and lower bounds relying on the number of intersections N=361 is all we know so far:

PROPOSITION: N! is a lower bound for the number of legal games using the rules in 1.1.

PROPOSITION: N^{3^N} is an upper bound for the number of legal sequences of plays using the rules in 1.1.

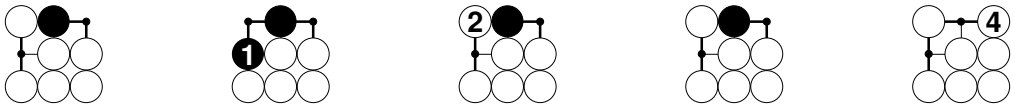
The proof for the trivial first proposition counts all possibilities of black filling the board while white always passes. The second proposition observes at most 3 states per intersection, at most 3^N positions, thus at most 3^N plays, at most N legal intersections per play, and thus at most 3^N times a choice among at most N intersections. – Now better bounds are known, which require combinatorics beyond the scope of this paper, but still comparing lower and upper bounds is like comparing apples in a basket and all particles in the universe. By Stirling [1], who showed that $n! \approx (\frac{n}{e})^n \sqrt{2\pi n}$, we know that both bounds are of exponential size. The complexity of Go is in between them and thus also exponential. This is impressive and means that Go can be solved only with

mathematics maybe together with informatics. In the author's opinion this will take a few hundreds of years. Even if Go will be solved, a slight deviation from known perfect play will leave the players at their own again.

1.4 The Fixed Ko Rule

An important example of Go rules mathematics replaces the superko rule of the rules in 1.1 by the fixed ko rule:

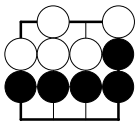
RULE: A play from a first position to a second position is prohibited if an earlier play was from the first position to the second position.



black's turn 3 = pass

The example illustrates why there are not any ko fights under the fixed ko rule. The play 2 is allowed because it is the first time that the position before it is followed by the position after it. Black may not use 3 to play at 1 because then the position before it would be the same as the position before the play 1 and the position after it would be the same as the position after the play 1. This is prohibited by the fixed ko rule. So black 3 passes. Black could already have passed at 1 achieving the same result.

In a triple ko any ko stone would be recaptured immediately. In an eternal life (chosei) after four turns the original position would be reached again. It is not possible to fight about kos or cycles in general. All kos are what Ing Chang-Ki would have called "disturbing". The reason is that the superko rule prohibits plays after an odd number of turns while the fixed ko rule prohibits plays after an even number of turns.



black's turn

In this example black cannot improve the position. Due to the fixed ko rule even a basic ko mouth is worth one eye for the player surrounding the empty intersection.

The fixed ko rule avoids all ko fights and alters life and death. The resulting game is different from Go. However, mathematics does not care for causes of life and death in a game. Go with the fixed ko rule is a nice mathematical model for Go because avoiding all ko fights makes it much simpler. So whenever theoretical studies of Go become too difficult because of ko fights, one can temporarily resort to study of Go with the fixed ko rule. E. g. ko fights are a major obstacle for applying combinatorial game theory to the endgame. Usage of the fixed ko rule allows combinatorial game theory to move further from the endgame towards the late middle game more easily. Of course, the middle game is very complex even without ko fights.

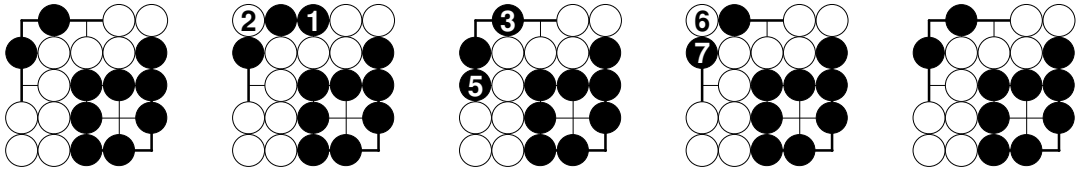
1.5 The Cycle Law

We present the following as another easy example of Go rules mathematics using, e. g., the rules in 1.1 with the fixed ko rule in 1.4 instead of the superko rule:

DEFINITION: A *situational cycle* is a sequence of alternate turns so that before and after it the position and the turn's player are the same.

PROPOSITION: In a situational cycle the difference of black passes and white passes equals the difference of removed white stones and removed black stones.

The proof is a simple transformation using the presupposition. The fascinating thing about mathematics is that a proposition can be applied in general. Here it can be applied to any situational cycle. Even Go players could apply it in practice, e. g., under a superko rule they simply respect the last move that misses to complete a cycle or they want to assess the score on the subboard involving cyclical play.



black's turn | 4 = pass | 8 = pass | black's turn

The example sequence of alternate turns from 1 to 8 is a situational cycle that includes two white passes more than black passes. Thus by the cycle law two black stones more than white stones have been removed. Furthermore, the sequence from 1 to 4 is a situational cycle that includes one white pass more than black passes. By the cycle law one black stone more than white stones has been removed. The same applies to the situational cycle of the turns 5 to 8. As can be seen, in situational cycles it is superfluous to count removed stones; it is sufficient to count passes.

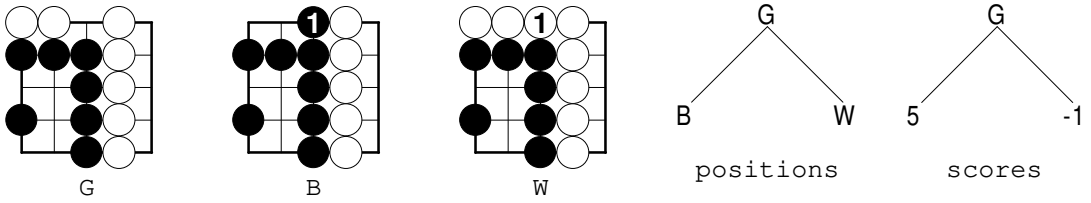
2 Applications of Game Trees

In mathematics or informatics a Go player's reading ahead of positions can be represented as game trees. They are used by amateur or professional mathematicians and by computer Go programmers. Even a reading Go player could visualize them in his mind. So this chapter relies on Go theory, mathematics, and informatics and can be applied in both scientific fields as well as by Go players.

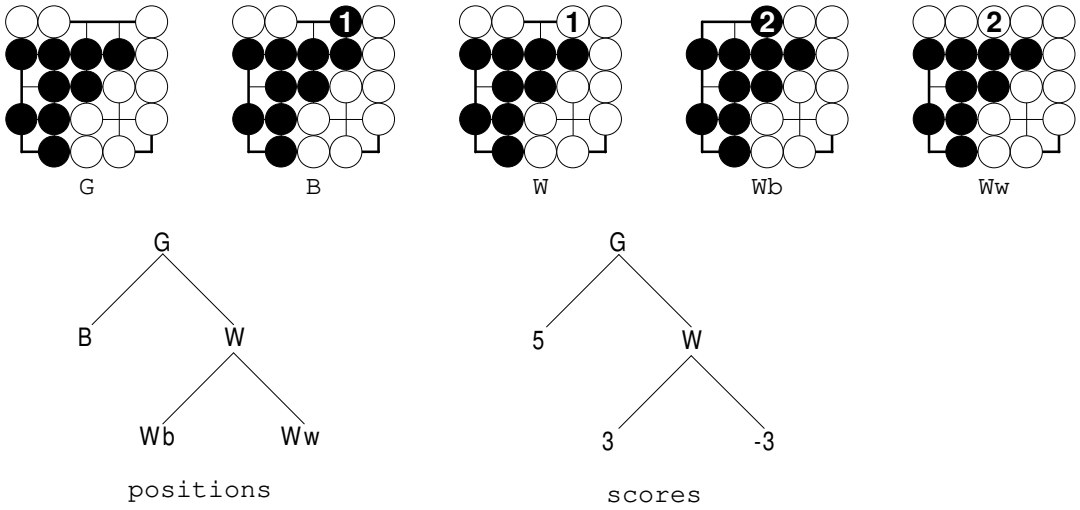
A game tree can be represented as a graph where each node represents a game position and each edge represents a move. A tree is written at read from its top to its bottom. Child nodes towards either the left ot the right represent moves by either Black or White. A tree can be abbreviated by omitted subtrees that would represent obvious continuations. Trees are very useful because they replace particular positions by general, abstract notations and values, which can then be studied by mighty, abstract means.

2.1 Endgame Values

It is well known that finite game trees can represent plays in positions by numbers. At the lowest level of a tree we interpret the numbers as the final positions' scores given by the rules like those in 1.1. This is possible because of finite alternation, which is given according to 1.2. Numbers are given from Black's point of view; negative numbers favour White.



We have the example endgame G. A black play leads to position B, a white play to position W. Their values are the scores: B = 5 and W = -1. This can be represented in a tree of positions or in a tree of their values.

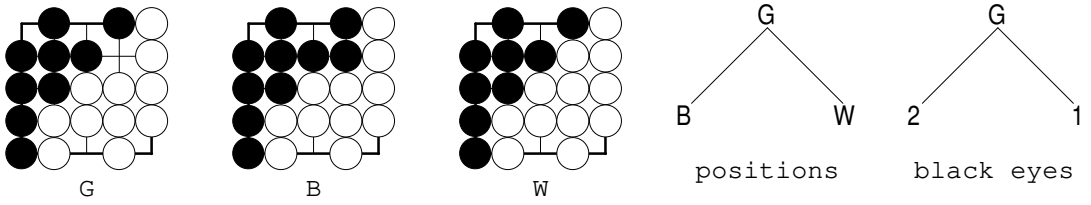


Again we call this example game G. In the subgame B Black will continue play to remove the two white stones and therefore we can easily identify the final score B = 5. The other subgame W should be notated in detail. If in it the next play is by Black, then this gives position Wb, if the next play is by White, then this gives position Ww. We know the scores of these final positions: Wb = 3 and Ww = -3. If the example game G is part of a bigger board, then this additional information will help if Black does not play Wb immediately after White W.

It is the task of combinatorial game theory to interpret game trees and their numbers, what is beyond this paper. However, it should be noted that values at the leaves of a tree can be used meaningfully to calculate nodes' values at higher tree levels.

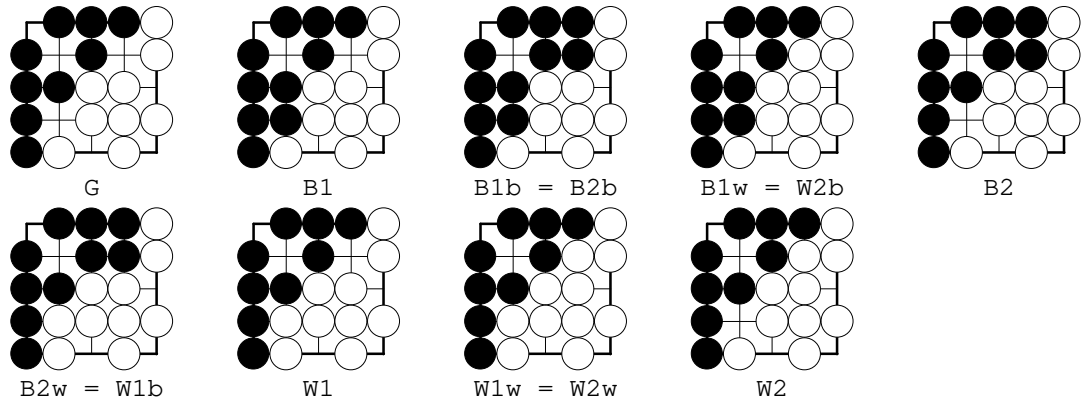
2.2 Eyespace Values

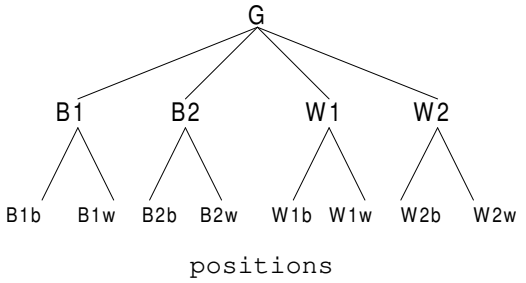
Eyespaces of a group can also be represented as numbers in game trees [2]. From a starting position G one analyses a particular group's eyespaces by counting the number of eyes after either Black or White moves first. If necessary, this process is done iteratively for each subgame until eyes can be determined readily.



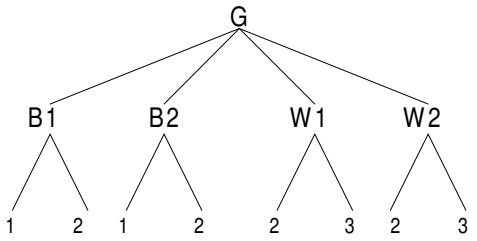
We consider the life and death game for Black's group. We have the starting position G, the position B after a black move, and position W

after a white move. Life and death depends on the number of eyes, so we count it for the black group: We get the eye values B = 2 and W = 1. Again this can be represented in a tree of values. It is meaningful to calculate the value of G as the average of B and W, which is 1.5, and this can be interpreted as the number of Black's eyes in G.





positions

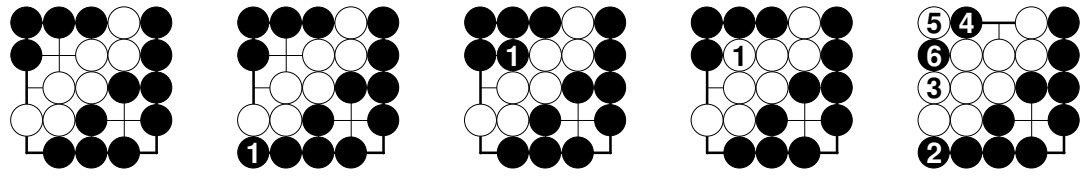


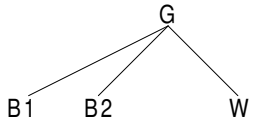
white eyes

In this example G we consider the number of eyes of the white group. For his first move Black has two options B1 or B2. Also White has two interesting options for his first move. After each possible first move we consider either Black to play next (the options marked with b) or White to play next (mark w). After two moves the interesting part of the game is settled, i. e. either White's group has at least two eyes and cannot lose them or White has fewer than two eyes and cannot gain more. Interpreting the eye tree, White is alive unless Black gets two successive moves.

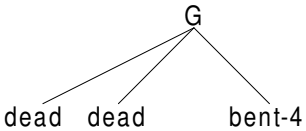
2.3 Status Types

Game trees can be used for the even more generalized application of a group's status types. These may include "alive", "dead", "coexistence", "basic ko", "double ko coexistence", "perpetual ko", "bent-4", etc. Although in theory there is great variety of status types, in practice the outcome of groups in almost all positions belongs to one of the very frequent types. A pragmatic approach would summarize the rare cases as "unknown".

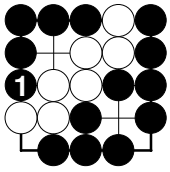




positions



white status

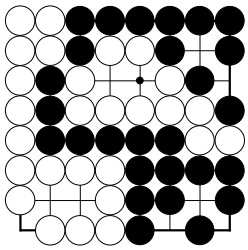


B3

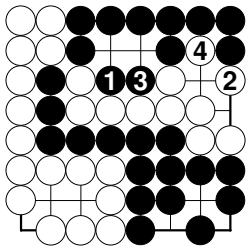
In this example position G we are interested in the status type of the white group. If Black moves first to position B1 or to position B2, then it is dead. If White moves first to position W, then it is a bent-4, as can be seen in the continuation. "bent-4" is different from "dead" because on a bigger board it might be involved in a global fight during the game, where it might turn out to be alive or dead at the game end. The first game tree shows the positions, the second one shows the status types of the white group. Note: A dull computer program would, e. g., also consider B3 as a third option of Black and then the left part of the status tree would have another child, however, with the status "alive".

2.4 Plays in Pass Fights

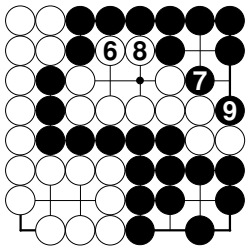
An only recently invented application of game trees is pass fights, i. e. the fight to avoid passing first. The purpose of the game trees is to denote the difference of numbers of still available plays on the board. Surprisingly, even this application could be used by Go players confronted with pass fights immediately.



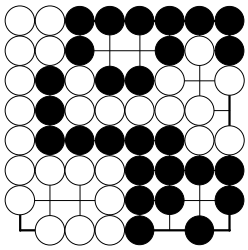
Molasses Ko
Black to move



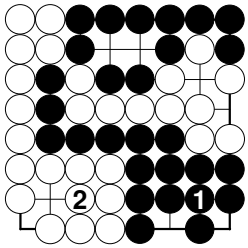
5 = pass
cycle part 1



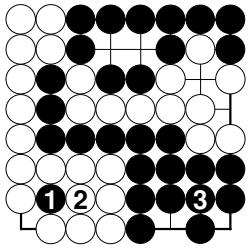
10 = pass
cycle part 2



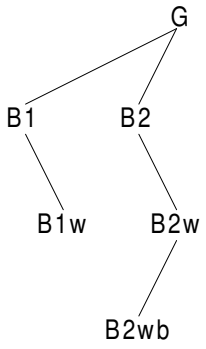
G



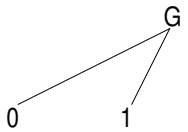
B1 subtree of ko threats



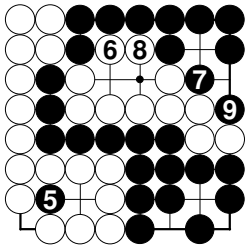
B2 subtree of ko threats



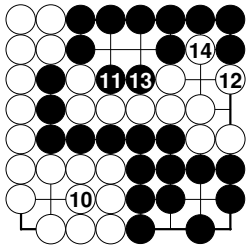
positions



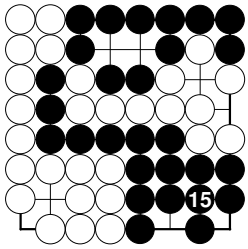
difference between the numbers of black plays and white plays



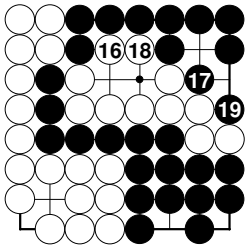
ko fight part 1



ko fight part 2

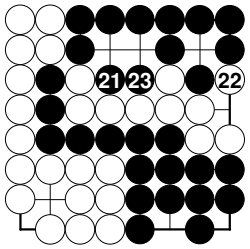


ko fight part 3

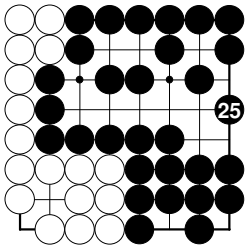


20 = pass

ko fight part 4



24 = pass
ko fight part 5



ko fight part 6

The upper right part of the example position is a Molasses Ko. To understand the nature of this ko, we first analyse a typical cycle that might occur in it if there were no restriction of repetition other than that a single stone may not immediately recapture a single stone that has just captured a single stone. Such a cycle has the rare feature that locally every fifth move is a pass that is not followed by a successive opposing pass. So after an even number of plays on the board the next player must pass or play a threat elsewhere on the board. Contrarily in a normal ko fight one plays elsewhere after an odd number of local plays in a ko. Using the rules in 1.1 with its positional superko rule, in the Molasses Ko play 9 is prohibited because it would recreate the position before play 1. Hence the move 5, which is a Black pass, is a mistake. Instead Black must play elsewhere on the board to change the position. In general, the first player that has to pass and cannot play elsewhere on the board any longer loses the Molasses Ko fight and thus the game. So the game is about avoiding to pass first. This is the pass fight. Play elsewhere on the board takes place every fifth move while the other four moves in between are plays in the Molasses Ko. This justifies its name because play elsewhere proceeds at one fifth of its usual speed. To win the pass fight, each player has to ensure to get more plays elsewhere on the board than his opponent. We shall count the number of plays elsewhere and represent them in a tree.

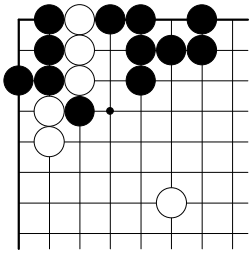
After four moves we reach position G that is the start for counting numbers of plays elsewhere (ko threats). To determine Black's best play, we first count those. Black has two major options: He can use his ko threat 1 by filling one of his eyes creating position B1 or he can use his ko threat 1 to fill a big opposing eye creating position B2. From position B1 White can play at 2 creating position B1w. From position B2 White can play at 2 creating position B2w and then Black can play at 3 creating position B2wb. The game tree of positions is shown. Its left path describes the diagram "B1 subtree of ko threats", its right path describes the diagram "B2 subtree of ko threats". For each path we count the difference between the numbers of black plays and white plays. From this we get the tree of difference numbers. Greater numbers favour Black. The number 1 is the most favourable for Black, so he chooses to use his play 1 to create position B2. With this knowledge it is now easy to fight the Molasses Ko. Every fifth move must be a play elsewhere. Black 15, which is play 3 in the B2 subtree of ko threats, is the last reasonable ko threat. Thus Black will win the ko fight. White 20 as the first pass indicates the loss. Because of the positional superko rule White 24 cannot fight in the Molasses Ko any longer and must be a pass.

3 Restriction of Calculation Size in Computer Go

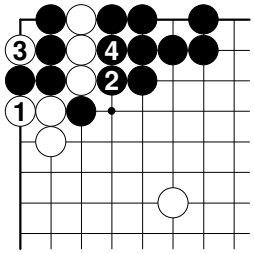
This chapter relies on Go theory and informatics and provides applications especially for the latter. The most frequent failure in computer Go is caused by too great sizes of game trees. Therefore

their pruning is essential. Game trees must become low and narrow. For this we discuss cases that allow a considerable contraction of reading subtrees. Such is often possible and good for computers when it is so for humans. We neglect details here and illustrate only examples of principle applications.

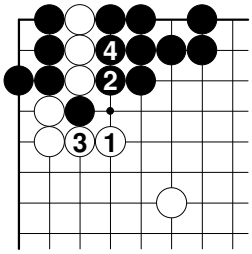
3.1 Ladders



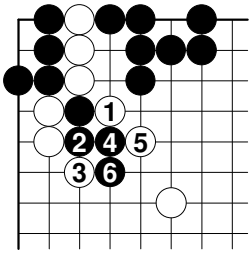
Is the black corner group alive?



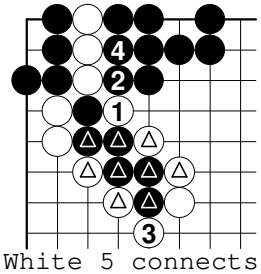
Case 1



Case 2



Case 3 / calculation depth 6

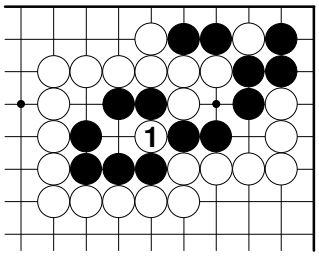


White 5 connects

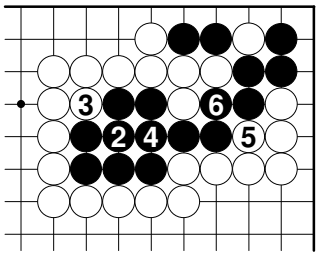
case 3 / contracted Ladder

A program wants to calculate whether the Black corner group is alive. Suppose it has a maximal calculation depth of 6. In case 1 or in case 2 it will judge that Black is alive because it can calculate deeply enough. In case 3, however, the program has to stop its calculation at move 6 and prematurely concludes again that Black is alive. Every ladder could easily outnumber a program since most ladders are very deep. On the other hand, a ladder is unbranched and therefore a program should be able to handle it. The solution is to contract the ladder and treat it as if it had depth 1. In the diagram all triangled stones are associated with White's move 1. Thereby the program is made to belief that the entire calculation in case 3 is of depth 5 only and it can conclude that Black is dead.

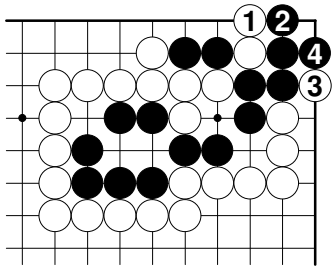
3.2 Capturing after a Local Series of Threats



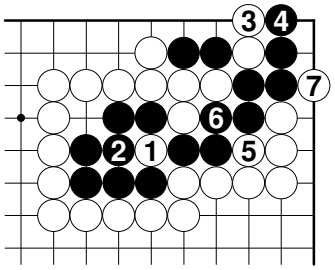
Can all black stones live?



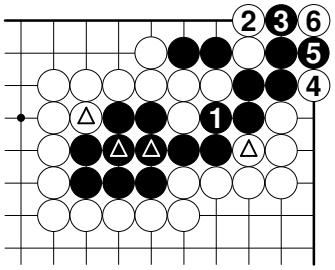
Continuation of local series of threats



Eyes in the corner



Bad reading

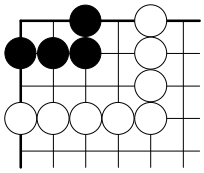


Good reading

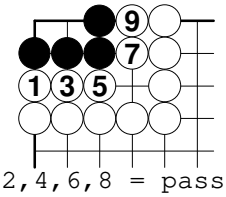
A program with a maximal reading depth 6 must decide whether Black can save all his stones after White 1 in the first diagram. The second diagram shows that a local series of threats is involved. The reader knows how to count eyes in the corner (the third diagram) but the program might choose to calculate move-sequences with bad, chaotic orders of moves (the fourth diagram) and thus fail to come to a conclusion because it could already not execute move 7. The solution

is to contract the subsequence within the local series of threats. In the fifth diagram the triangles and move 1 represent it. Then the program can do the ensuing calculation and conclude that Black cannot save all his stones.

3.3 Equal Approach Moves

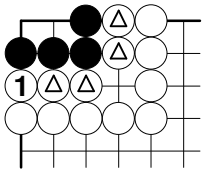


Is Black alive?



2,4,6,8 = pass

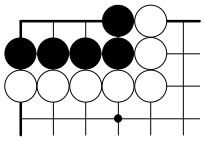
Bad reading



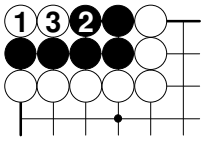
Good reading

If a program with little knowledge shall decide about Black's life and death, then it would indifferently occupy one liberty after another as in the second diagram. Even worse, it could do so in any order. Very easily the size of calculation would explode. Good reading teaches the program to treat equal approach moves equally and to contract them to a single move as in the third diagram.

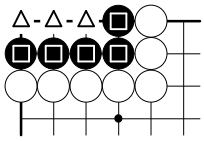
3.4 Eye Shapes



Is Black alive?



Bad reading



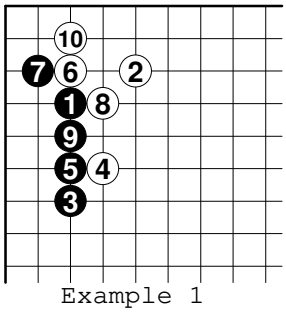
Good: no reading

Again we have the standard task for our program. It would be bad to read at all. The shape is nakade. Although here a program can easily calculate it, if it did so in every life and death position, then reading would become too complex because nakade shapes can be bigger than here and, more importantly, nakade occur frequently at terminal places of game trees that a program calculates. Therefore it is essential to contract the reading associated with a nakade shape and let the program look up a database. E. g. in the third diagram it would recognize the triangles as a particular nakade shape in the corner and the squares as a sufficient surrounding, so that it does not need to read at all.

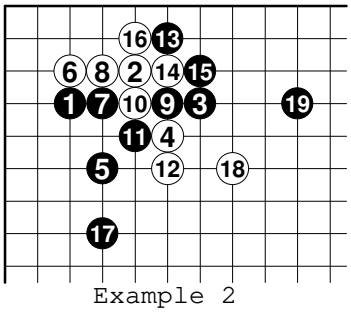
4 Functional Languages as a New Approach to Go Theory Exemplified for Joseki

For a change, besides Go theory this chapter relies on linguistics that studies Go strategy. Functional languages are invented that then can be applied to strategy. Traditional Go theory combines reading ahead with shape aspects. The former is necessary while the latter can be replaced or expanded by functional languages. The result is a new approach to study of Go theory. Since the approach is formalized, it promises to provide the Go playing community with new insights into Go theory. This will be shown for the following example joseki and languages will refer to an increasing scale – from single plays via subsequences of plays and groups to entire joseki. Rote memorization of move-sequences easily fails; instead joseki study succeeds if meanings, reasons, and methods are understood in general. Functional

languages serve this purpose. In this chapter terms of functional languages presume perfect or at least reasonable play.



Example 1



Example 2

4.1 Meanings of Plays

While merely reading ahead sequences of plays can be called tactics, evaluating meanings of plays is already elementary strategy. For centuries systematic study of Go theory has concentrated on tactics; on the other hand, strategy has been neglected a little and now functional languages could help to change this situation.

Each single play in a joseki must have one or several functional meanings. Therefore a language should be used for them. First we discuss the major meanings of each single play in the examples, however, because of space restrictions not all meanings are stated and for example 2 we list only a few of those new meanings that do not also occur in example 1:

PLAY	**FUNCTIONAL MEANING**
Example 1	
1	installation of a group
2 2	installation of a group restrict extension potential of an opposing group
3 3 3	extend a group towards a direction improve eyespace make territory
4 4	weaken connection between opposing stones force opponent to assist an own group
5 5	maintain connection of a group assist an own group by defending against a forcing play
6 6 6 6	extend a group towards a direction improve eyespace make territory restrict extension potential of an opposing group
7 7 7 7 7	extend a group towards a direction improve eyespace make territory restrict extension potential of an opposing group weaken connection between opposing stones
8 8 8	maintain connection of a group weaken connection between opposing stones improve eyespace
9 9 9	maintain connection of a group improve eyespace make territory
10 10	improve eyespace make territory
Example 2	
9	set up a crosscut
10	set up a crosscut

11	set up a crosscut
12	run away
18	run away

Obviously, functional meanings occur again and again. Therefore we can create a functional language of meanings of plays. The vocabulary of this language will be the functional meanings of single plays. The grammar will be their classification. For an illustration we use those funcational meanings already seen in the examples:

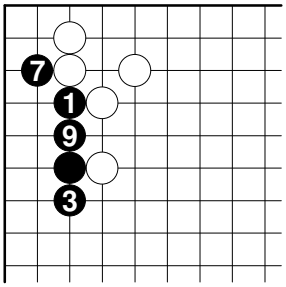
CLASS	**FUNCTIONAL MEANING OF A PLAY**
construction of own stones	installation of a group
construction of own stones	extend a group towards a direction
construction of own stones	improve eyespace
construction of own stones	make territory
construction of own stones	maintain connection of a group
destruction of opposing stones	restrict extension potential of an opposing group
destruction of opposing stones	weaken connection between opposing stones
temporary action	force opponent to assist an own group
temporary action	assist an own group by defending against a forcing play
temporary action	set up a crosscut
fight	run away

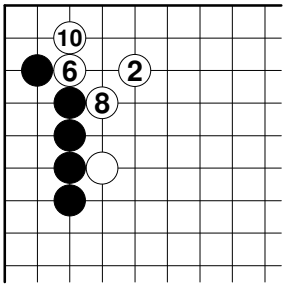
It should be noted that each functional meaning and each class has a fixed term so that it is recognized immediately whenever it it is used. Furthermore one can observe that a comparatively small language of terms of meanings is sufficient to describe almost all plays and an even much smaller number of classes is needed. Thus it requires only little effort to create a powerful functional language. In fact, it is so useful that players use it every day. The new feature is to formalize it like mankind has formalized verbal languages by creating written languages. A written functional language can ease study of strategy and is also a preparation for computer Go programming. Formalization of languages has always greatly expanded their scope of application and this should be expected for the languages in this chapter in the future.

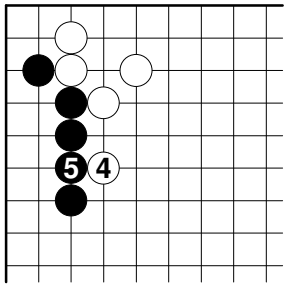
4.2 Strategic Lines

What is done with meanings of single plays can also be done for several plays together occurring in a joseki. Whenever several plays contribute to the same meaning we unify this and call it a strategic line. In a game tree of all possible moves a strategic line follows a single path but it is often interrupted by moves that do not have the same meaning.

In example 1 the following strategic lines occur:





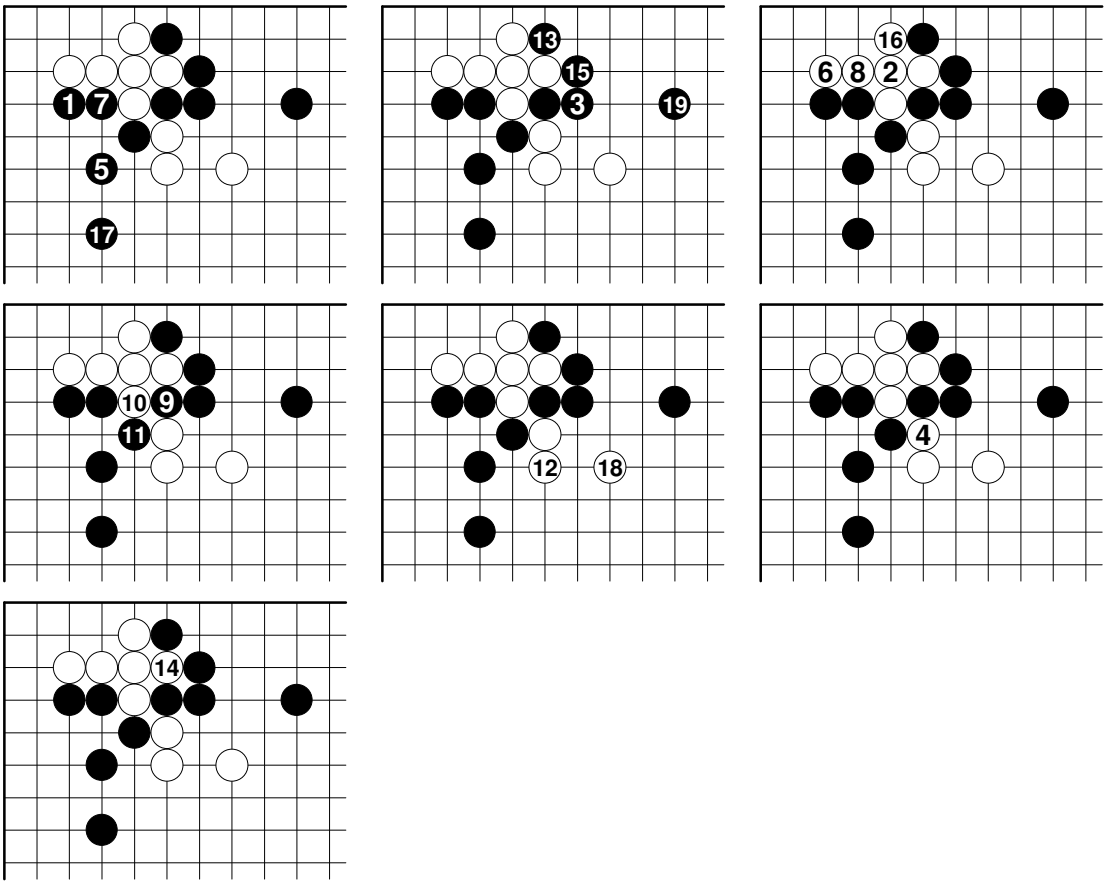


PLAYERS	**PLAYS**	**STRATEGIC LINE**

Black	1,3,7,9	construct a group
White	2,6,8,10	construct a group
both	4,5	forcing play

Each player consistently follows a constructive strategic line to install a group. Both strategic lines are interrupted by a mutual strategic line of forcing with 4 and answering the forcing play with 5. This strategic line would not be necessary for constructing the white group, however, it is efficient to use it as well. The white stone 4 does not really belong to the constructed white group but it will assist White in defending his group later, if necessary. We see that each strategic line has its meaning and that more strategic lines provide more strategic meaning.

In example 2 the following strategic lines occur:



PLAYERS	PLAYS	STRATEGIC LINE
Black	1,5,7,17	construct a group
Black	3,13,15,19	construct a group
White	2,6,8,16	construct a group
both	9,10,11	set up a crosscut
White	12,18	run away
White	4	offer strategic choices to the opponent
White	14	create possibilities

Some strategic lines can consist of just one play. Others can be performed by both players, however, most include plays of just one player. – Strategic lines can be classified, too:

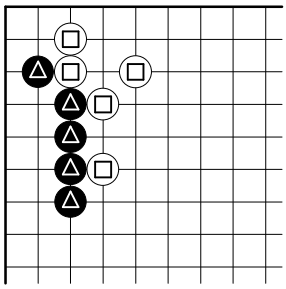
CLASS	STRATEGIC LINE

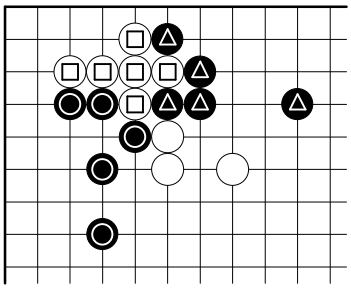
construction	construct a group
fight	set up a crosscut
fight	run away
temporary action	forcing play
temporary action	offer strategic choices to the opponent
temporary action	create possibilities

The above table can be extended and another small functional language is born. Its fixed terms are strategic lines, its grammar classifies them. Again applications can be formalized study of strategy or preparation for computer Go programming. Furthermore, one can draw relations between the functional language for plays and the functional language for strategic lines. However, this is beyond the scope of this paper.

4.3 Meanings of Groups

The scale of strategic considerations can be increased to describe the strategic meanings of groups. Simply speaking, one might identify two groups at the end of play in example 1 and four groups at the end of play in example 2. We list their major meanings:





GROUP	**MEANING**
example 1 black triangle	make territory
example 1 black triangle	develop towards a side
example 1 white square	make eye-shape
example 1 white square	develop towards a side
example 2 black circle	make sphere of influence
example 2 black circle	develop towards a side
example 2 white square	make territory
example 2 white square	make eye-shape
example 2 black triangle	make sphere of influence
example 2 black triangle	develop towards a side
example 2 white unmarked	deny opposing territory
example 2 white unmarked	cut two opposing groups

The meanings can be classified:

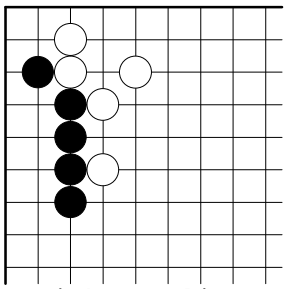
CLASS	**MEANING OF A GROUP**
construction	make territory
construction	develop towards a side
construction	make eye-shape
construction	make sphere of influence
destruction	deny opposing territory
fight	cut two opposing groups

This classification is a start for a third and small functional language. This time its terms are meanings of a group and its grammar

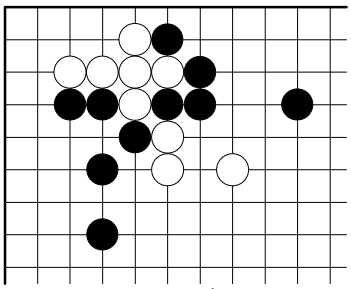
is their classification. Applications are as usual and relations to other functional languages can be drawn.

4.4 Joseki Types

Another aspect of joseki is the function of a joseki as a whole. We call this "joseki type". Joseki are classified by assigning each of them to a particular type. The joseki types of the examples are as follows:



quick settling



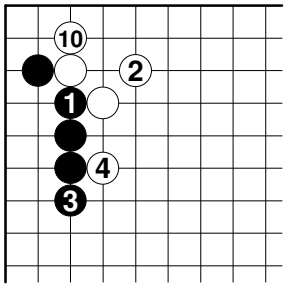
cut and fight

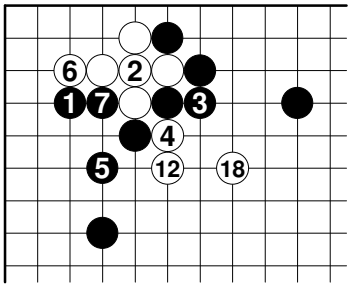
The following short overview on important joseki types shall be given but not be discussed in detail, though it is interesting to note that most joseki can be classified by a small number of types.

JOSEKI TYPE
quick settling
thick settling
seeking life in the corner
seeking life quickly
cutting while moving to the center
corner versus influence
side moyo
territory versus influence
mutual running fight
cut and fight
lean and attack
pincer and playing elsewhere

4.5 Regularly Playing Elsewhere

Instead of many plays of a joseki it is often possible to play elsewhere for a regular reason like occupying another big or urgent point. Therefore we can classify all joseki moves into two types: "regularly playing elsewhere is possible" or "regularly playing elsewhere is impossible". This is analysed for the two examples, however, for example 2 only the "possible" plays elsewhere are listed:





PLAY	REGULARLY PLAYING ELSEWHERE	REASON
Example 1		
1 1	possible possible	occupy another empty corner continue a remote fight
2	possible	consider the corner not urgent enough
3	possible	consider the corner not urgent enough
4	possible	regard the prior plays 2-3 as forcing exchange
5	impossible	a local play is too big
6	impossible	the prior exchange 4-5 would waste possibilities without meaning
7	impossible	a local play is too big
8	impossible	the prior exchange 6-7 would waste possibilities without meaning
9	impossible	a local play is too big
10	possible	treat own stones lightly
Example 2		
1 1	possible possible	occupy another empty corner continue a remote fight
2	possible	consider the corner not urgent enough
3	possible	consider the corner not urgent enough
4	possible	consider the corner not urgent enough (rare)
5	possible	consider the two development directions for

		the stone 1 to be equal options
6	possible	treat the stones 2 and 4 lightly
7	possible	treat the stone 3 lightly
12	possible	rely on life in the corner + rely on a ladder for the stone 4
18	possible	treat the stones 4 and 12 lightly

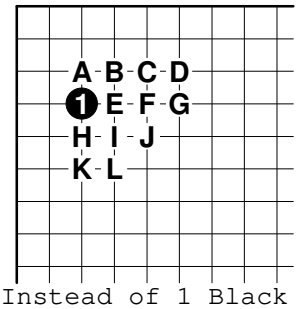
A functional language could be designed around reasons about the possibility of regularly playing elsewhere. Although quite a lot of reasons can be imagined, it should be possible to list most reasons after diligent work. This is a task for future research.

4.6 Irregularly Playing Elsewhere

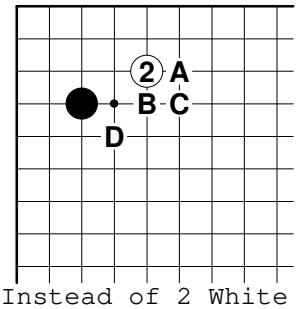
Instead of any joseki play circumstances might make irregularly playing elsewhere on the board possible. Thus we classify all joseki plays as "irregularly playing elsewhere is possible". Reasons for this include global exchanges of mirror play, ko fights, or ladder breakers. Why does it make sense to classify plays if all plays can be classified that way? In the late endgame perfect play is known. There no play could be assigned the attribute. Using it for joseki plays ackowledges our very incomplete knowledge about perfect play early in the game. Like with regularly playing elsewhere reasons could be collected, however, this time for irregularly playing elsewhere. Those reasons could then be incorporated in a more powerful functional language about playing elsewhere.

4.7 Strategic Choices

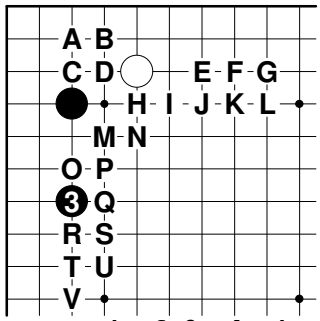
During a joseki some moves are necessary, i. e. regularly playing elsewhere is not possible. However, during some moves the player to move has a choice between two or several continuations leading to different strategic outcomes. We call this strategic choices. Strictly, there is a strategic choice at every move because there is always the option of irregularly playing elsewhere. For simplicity, considerations about playing elsewhere shall be ignored here but they could be considered together with strategic choices.



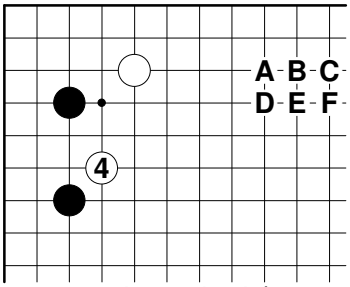
Instead of 1 Black could play at one of A, B, ..., L.



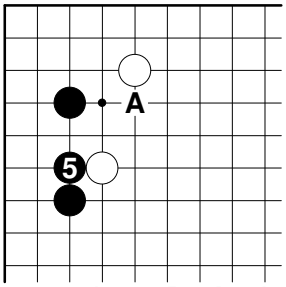
Instead of 2 White could play at A, B, C, or D.



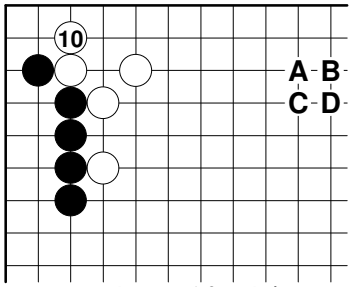
Instead of 3 Black could play at one of A, B, ..., V.



Instead of 4 White



Instead of 5 Black



Instead of 10 White

could play at one of A, B, ..., F.	could play at A.	could play at A, B, C, or D.

As shown, in example 1 instead of some of the joseki moves the player to move has a strategic choice between several reasonable moves. To create a functional language about strategic choices one can list and classify reasons for selecting one or another option. Reasons can be related to

- the decision tree of all reasonable subsequent move-sequences,
- joseki types (as discussed before),
- directions of development (like along particular a side),
- global characteristics (like choosing the joseki type "mutual running fight" because one has thick shapes elsewhere on the board).

Obviously, this is a wide study field for future research.

5 References

Most of the contents refers to original, unpublished work, discussion on a newsgroup, or discussion on mailing lists. Therefore only a few references are given:

- <jasiek@snafu.de> : email address in 2001 of Robert Jasiek
- <news:rec.games.go> : Internet newsgroup about Go
- <go-rules@usgo.org> : mailing list go-rules
- <computer-go@lists.uoregon.edu> : mailing list computer-go
- [1] "Taschenbuch der Mathematik", Bronstein / Semendjajew
- [2] "Eyespace Values in Go", Howard A. Landman