

Applying Data Mining to the Study of Joseki

Michiel Helvensteijn

Abstract Go is a strategic two player boardgame. Many studies have been done with regard to go in general, and to joseki, localized exchanges of stones that are considered fair for both players. We give an algorithm that finds and catalogues as many joseki as it can, as well as the global circumstances under which they are likely to be played, by analyzing a large number of professional go games. The method used applies several concepts, e.g., prefix trees, to extract knowledge from the vast amount of data.

1 Introduction

Go is a strategic two player game, played on a 19×19 board. For the rules we refer to [7]. Many studies have been done with regard to go in general, cf. [6, 8], and to joseki, localized exchanges of stones that are considered fair for both players. We will give an algorithm that finds and catalogues as many joseki as it can, as well as the global circumstances under which they are likely to be played, by analyzing a large number of professional go games.

The algorithm is able to acquire knowledge out of several complex examples of professional game play. As such, it can be seen as data mining [10], and more in particular sequence mining, e.g., [1]. The use of prefix trees in combination with board positions seems to have a lot of potential for the game of go.

In Section 2 we explain what joseki are and how we plan to find them. Section 3 will explain the algorithm in more detail using an example game from a well-known database [2]. In Section 4 we mention some issues concerned with symmetry. Section 5 will discuss the results of the algorithm. We try to explore the global circumstances under which a joseki is played in Section 6. Section 7 contains conclusions and discusses further research.

Michiel Helvensteijn

LIACS, Leiden University, The Netherlands, e-mail: mhelvens@liacs.nl

Please use the following format when citing this chapter:

Helvensteijn, M., 2008, in IFIP International Federation for Information Processing, Volume 276; *Artificial Intelligence and Practice II*; Max Bramer; (Boston: Springer), pp. 8796.

2 Joseki

There are several definitions of *joseki*, see, e.g., [5] and [4]. We will use a somewhat adapted definition, which uses elements from other sources:

A joseki is a localized sequence of play in the game of go, in which both players play locally optimal moves. These sequences occur often in recorded games, especially in the corners of the board and the beginning of the game.

It is an important property of a joseki that it is a local phenomenon. It takes place in a certain part of the board and moves that are played elsewhere (before, during or after the joseki) are not part of it. The players can sometimes break away from a joseki and get back to it later. This way, multiple joseki can be in progress at the same time. The move that breaks away from a joseki is called a *tenuki*. The move that continues the joseki after a *tenuki* is called a *follow-up play*.

We do not think that a joseki results in a fair outcome for both players by definition, as is often stated in other definitions. In fact, joseki do not result in a fair outcome if played under the wrong global conditions. Of course, if a sequence did not result in a fair outcome under *some* global condition, it could never have been played often enough to be noticed and given the name joseki. This is important. Professional players do not blindly play joseki in their games, not even under optimal global conditions. At most, they use their knowledge of joseki as a heuristic. They play the move that they think is best, and that is how joseki are found. This is why we have not mentioned fairness in the definition, it is implied already. It is also irrelevant to the algorithm. A computer can not calculate whether a sequence is fair. Instead we choose to rely on human intuition, in that a sequence must be a joseki under the above definition if it is played often enough by professionals.

3 The algorithm

In this section we give an algorithm to find joseki in a given database. We will use a database from the Go4Go website [2], which contains 13,325 go games played by professional go players. It is the job of the algorithm to analyze the games from this database and eventually output the joseki (plural) that were found. The algorithm is depicted in Figure 1 and Figure 2, which show phase 1 and 2 of the algorithm respectively. In this section we mention some symmetry related issues; however, the formal treatment of this subject is postponed until Section 4.

The first phase extracts all distinguishable sequences from the games in the database and stores them in a prefix tree. The second phase prunes that tree so that only the more interesting sequences remain, resulting in a tree of joseki.

3.1 Phase 1

At the end of this phase, we will have a tree of all distinct move-sequences¹ to be found in the games. Phase 1, step 1 is basically a nested loop that iterates over all moves of all games in the database. Every move is compared to all currently stored sequences that belong to the current game. Its Manhattan distance² to the stones of those sequences is used to determine whether it belongs to one of them. It is also possible for one move to belong to more than one sequence.

Because the joseki we are looking for are played mostly in the corners and in the beginning of the game, the algorithm will stop looking for stones after each corner contains at least 20 stones. This means that at the least, we will examine 80 moves. Anything more than that means we have entered mid-game. The reason we don't just look at the first 80 stones instead is that sometimes a single corner can remain empty for a large portion of the game, which means we might miss some obvious joseki.

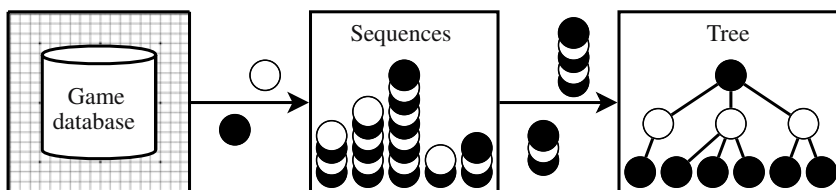


Fig. 1 The algorithm, phase 1: creating the tree

Step 2 moves these sequences to the tree, after a possible transformation (see Section 4). It is implemented as a prefix tree, a structure very popular in current data mining algorithms [3]. The root of this tree represents the empty board. Its children are the first moves of the sequences, and so on. Each node in the tree represents a sequence prefix to that point and its children represent its continuations. Each node contains a mapping of point→node (where “point” is a point on the board or a tenuki) to find its children. This provides very fast lookup and insertion of sequences. Each node also has a counter indicating how often a certain sequence has been played. An insertion increases the right counters in the tree and adds new nodes if necessary.

For efficiency, step 1 and step 2 are performed simultaneously. After every game, the sequences are added to the tree and the sequence storage is made empty.

¹ A sequence is a potential joseki. It is also called a sequence because it might eventually turn out to be irrelevant and be pruned from the tree in phase 2. Only the sequences that survive this process are called joseki.

² The Manhattan distance between two points is the absolute difference between their x -coordinates plus the absolute difference between their y -coordinates: the Manhattan distance between the points (x_1, y_1) and (x_2, y_2) is $|x_1 - x_2| + |y_1 - y_2|$.

3.2 Phase 2

Phase 2 consists of pruning and printing (in SGF format [9]) the tree that we built in phase 1. It removes the irrelevant sequences with a pruning function, that accepts or rejects a sequence based on its *pruning score*, i.e., its frequency in the prefix tree.

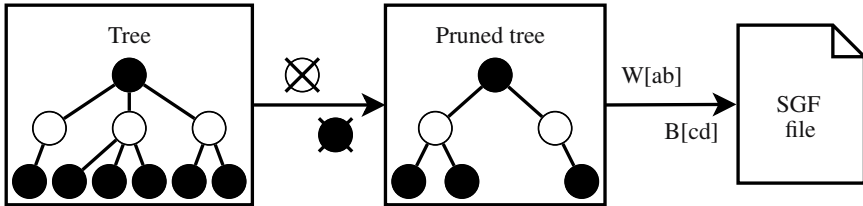


Fig. 2 The algorithm, phase 2: pruning the tree

Because of the nature of the prefix tree, the counter of any node is greater than or equal to the sum of the counters of its children, and so in particular greater than or equal to the counter of any child node. The basic pruning approach is to cut off any subtree that does not have a counter greater than or equal to a given threshold value. We have experimented with several threshold values. The optimal value appears to be around 1% of the amount of games in the database.

3.3 Example

We clarify the algorithm through an example game, a match between two strong players, Xie He (white) and Duan Rong (black), from the first round of the 18th Chinese CCTV Cup. See Diagram 1.

Both players first occupy the four corner star points. As can be seen in Diagram 1, each of the first four moves starts a new sequence. For now we will call them sequence 1, 2, 3 and 4.

Black 5 starts the first joseki of the match. It ends with black 9. A move belongs to a sequence if it is within Manhattan distance x of it, where x is a predefined threshold: the *sequence binding distance*. For this example, $x = 5$. Black 5 is clearly only within proximity of white 2, and so it is added to sequence 2. The same holds for white 6 to white 8. Black 9 also belongs to sequence 2, because it is within proximity of black 5, which was added to the sequence earlier.

White 10, black 11 and white 12 are added to sequence 3 (see Diagram 2). One will notice that black 11 is only just outside the reach of sequence 2. Black 13 starts a long joseki that ends with black 23. All of those moves are part of sequence 4.

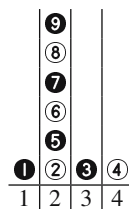
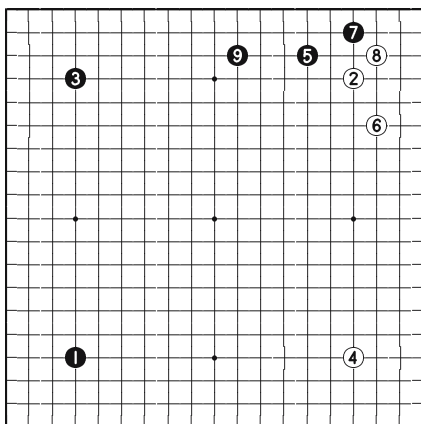


Diagram 1 Xie He vs. Duan Rong, part 1

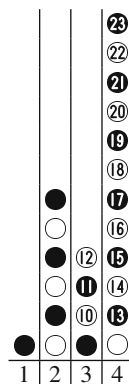
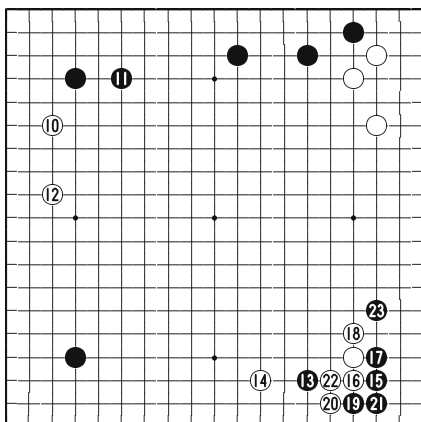


Diagram 2 Xie He vs. Duan Rong, part 2

Such long, isolated joseki are not as uncommon as one might imagine, as proved by this algorithm. That exact joseki is found 566 times in the database.

White 24 and black 25 add another small joseki to the collection, in sequence 1 (Diagram 3). But something else has also happened here. Black 25 is within range of white 12, as well as black 1, so it is also part of sequence 3, as is white 26. After two non-joseki moves, black 29 does the same thing. It is part of both sequence 1 and 4. This is bound to happen as more and more stones are played on the board. But the assumption is that either joseki are played in isolation before the sequences start interfering with each other or that only one of the sequences really “owns” the new move. It is not unthinkable that a stone with a distance of 5 from a sequence doesn’t really belong to it. For example, if black 25 were part of any joseki, it would

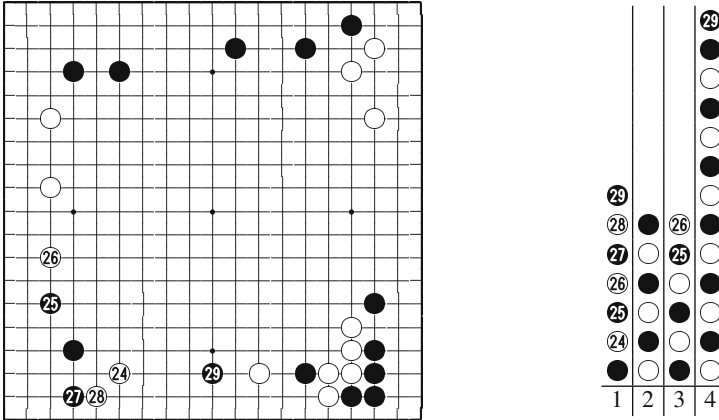


Diagram 3 Xie He vs. Duan Rong, part 3

be sequence 1. These things will be recognized in phase 2, when follow-up moves that do not belong to the joseki are pruned out of the tree.

We have now discovered and completed all joseki the algorithm is able to find in this example game. However, the algorithm will not know it at this point and will continue to build the sequences until at least 20 stones have been played in each corner.

Prefixes of all four of the sequences played so far will turn out to be important joseki in the final tree. Table 1 shows these joseki.

Table 1 Xie He vs. Duan Rong, four joseki

Joseki	Transformation	Color-swap?
1 ① ⑭ ⑮	H	No
2 ② ⑤ ⑥ ⑦ ⑧ ⑨	V	Yes
3 ③ ⑩ ⑪	D	No
4 ④ ⑬ ⑭ ⑮ ⑯ ⑰ ⑱ ⑲ ⑳ ㉑ ㉒ ㉓	H × V	Yes

The first column gives the sequence reference number. The “Joseki prefix” column gives the prefix of the sequence that forms the joseki. In other words, the stones that would be pruned in phase 2 are not shown here. The “Transformation” column shows the matrix manipulation that should be applied to each move of the sequence, so it will yield the sequence’s normal form (see Section 4). Here **H** means a reflection in the horizontal ($y = 10$) axis; **V** means a reflection in the vertical ($x = 10$) axis; **D** means a reflection in the diagonal axis on which black 3 and white 4 are played; and \times is the matrix multiplication operator. So sequence 4 has to be reflected in the horizontal and vertical axes to get to its normal form. The “Color-swap?” column

indicates if the colors of the stones need to be swapped from black to white or the other way around. This is the case for all sequences where white moves first, because by convention, black has the first move. We will adopt this convention to get our normal form.

This procedure is performed for all games in the database, resulting in a tree with still many irrelevant sequences. After phase 2, however, it will be a relatively small tree with assorted recognizable joseki.

4 Symmetry

The board has a symmetry group (the dihedral group D_4) with 8 elements, which can be generated by a rotation by 90° and a reflection in one of the four board axes (the horizontal, vertical and two diagonal axes). Reflecting twice is a rotation around the intersection of the axes, i.e., the board center or *tengen*. Another dimension of symmetry with regard to joseki is color, in that two spatially identical stones are still equivalent, even if one is black and the other is white.

This symmetry can be extended to a sequence of moves. When two joseki are equivalent in this way, we want the algorithm to recognize this. So when it transfers a sequence of moves to the tree, it first transforms each of them using reflections and color-swaps such that the resulting sequence will start with a black move in a designated triangular space on the board. Note that it is often necessary to consider more than just the first move in order to determine the exact transformation.

In theory, another transformation is possible: translation. Joseki that occur along the top edge of the board may be equally valid two places to the right or to the left. The algorithm does not take this into account, however, because this validity can be extremely complicated to judge. It is also very situation dependent, unlike reflections, rotations and color-swaps which are always valid.

5 Results

In this section we mention the results of experiments on the database [2], consisting of 13,325 games. We have experimented with several parameter settings. Each run took only a few seconds, which is not surprising in view of the linear nature of the algorithm. It was found that the following settings gave the best results:

pruning score: 150
sequence binding distance: 5

The resulting tree (Figure 3) contains 81 leafs, meaning 81 complete joseki. However, the algorithm does not only find joseki, but also a lot of what might more properly be called *fuseki* structures (opening game sequences). This is not surprising, since the algorithm looks primarily in the corners of the board and the opening

game, which are exactly the time and place fuseki structures are formed. The set of joseki and the set of fuseki seem to overlap when one only considers the opening game. The resulting tree shows some well-known joseki and fuseki.

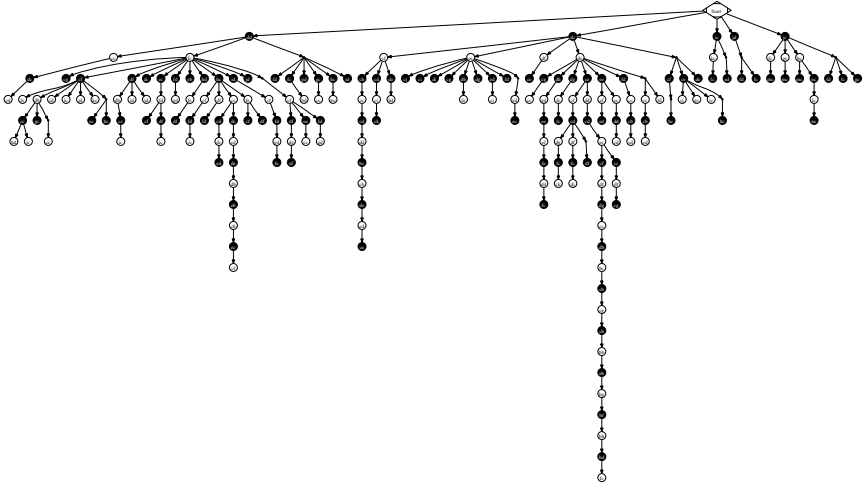



Fig. 3 Joseki tree, pruning score = 150, sequence binding distance = 5

6 Global influence

More is still to be known about these joseki. We know now which joseki (and fuseki) are played, but we still do not know *when* they are played. As explained in Section 2, this is important information.

There are many factors that could have an influence on the “fairness” or “validity” of a joseki, like ladder breakers, nearby influential stones and the overall score in points (a player who is ahead will most likely play a more balanced game). But another important factor is global influence around the board.

The algorithm calculates the global influence around a joseki. This information extends the output of the algorithm, but does not alter it. The algorithm as explained in Section 3 remains mostly unchanged, though the current state of the game is always kept in memory, so global influence can be investigated. And influence direction and color is of course transformed along with the sequence before being put into the tree.

Diagram 4 shows how this influence is calculated, using a new example. The stone marked  has just been played and added to the bottom-right sequence. The influence reaching this sequence has to be calculated for that move. From the left edge of the sequence’s bounding-box, a search to the left is done for each row. Each

row is scored (17 minus the distance from the bounding-box to the first stone, capped at 0)³. Certain threshold values can determine if the total score (all row-scores added together) is white, black or neutral. The net score for this particular search turns out to be 15 for black. This means black has the most influence in that direction, which is quite clearly the case. This procedure can be repeated for the other three sides, though two of them almost always have zero influence, since most joseki are played in a corner.

This influence information in the tree is stored in aggregate, and so it is determined what the most likely global circumstances are for each stage of each joseki. Manual inspection of the tree indicates that for most joseki, influence is most definitely a factor, as expected.

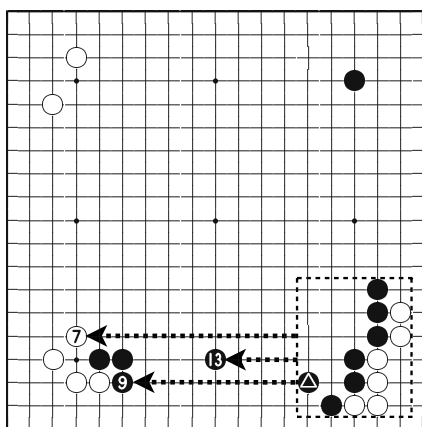


Diagram 4 Calculation of left-side influence

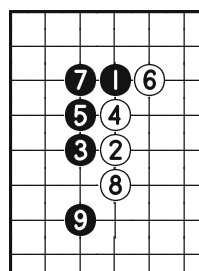


Diagram 5 The *nadare*; most variations branch off after white 6

7 Conclusions and future research

The technique of finding joseki in a database as described in this paper certainly has merit. It finds some well-known joseki and fuseki sequences and none of them seem out of place. The search for global influence also produced promising results.

For example, the algorithm finds the joseki shown in Diagram 5. This joseki is known as the *nadare*. This joseki, and some common variations on it, are described in the Kosugi/Davies book [5]. In the tree of Figure 3 it is the joseki of 9 moves deep, closest to the really long one (which seems to be a variation on the *nadare*

³ If a stone is closer to the sequence, it has more influence on it, making it more likely that a player will deviate from the joseki that would have been played without this influence. Even if a stone is on the other side of the board, though, it can have influence. The number 17 was chosen experimentally as the furthest distance from which a stone could still have *any* influence.

not described by Kosugi and Davies). By far most of the occurrences of this joseki have a consistent black influence from below throughout the sequence. To a lesser degree, white seems to have more influence to the right, which would play well with white's new wall. The fact that verifiable joseki such as this one can be found like this is very encouraging.

Because of the high ranking of the players in the database, not a big subset of the known joseki is found (there are thousands). It might be interesting to try this algorithm on a database of weaker players.

In the algorithm, the decision whether a move belongs to a sequence or not is decided by Manhattan distance. Other distance-measures could be used instead, and might be more appropriate. And the tree is now pruned with a single strict pruning-score. It may be advisable, in future research, to explore other possibilities.

Acknowledgements The author would like to thank Hendrik Blockeel, Walter Kusters and Jan Ramon for their help with this project.

References

1. Gouda, K., Hassaan, M. and Zaki, M. J.: PRISM: A Prime-Encoding Approach for Frequent Sequence Mining, Proceedings of the 7th IEEE International Conference on Data Mining, pp. 487–492 (2007)
2. Go4Go.net [online] <http://www.go4go.net/v2>
3. Han, J., Pei, J. and Yin, Y.: Mining Frequent Patterns without Candidate Generation, Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 1–12 (2000)
4. Joseki, Wikipedia [online] <http://en.wikipedia.org/wiki/Joseki>
5. Kosugi, K. and Davies, J.: Elementary Go Series, Volume 2, 38 Basic Josekis, Kiseido Publishing Company, Eighth Printing, 2007
6. Ramon, J. and Blockeel, H.: A Survey of the Application of Machine Learning to the Game of Go, Proceedings of the First International Conference on Baduk (Sang-Dae Hahn, ed.), pp. 1–10 (2001)
7. Sensei's Library, The Collaborative Go Website [online] <http://senseis.xmp.net>
8. Silver, D., Sutton, R. and Müller, M.: Reinforcement Learning of Local Shape in the Game of Go, Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI), pp. 1053–1058 (2007)
9. Smart Game Format Specifications [online] <http://www.red-bean.com/sgf>
10. Tan, P. N., Steinbach, M. and Kumar, V.: Introduction to Data Mining, Addison-Wesley, 2006