Memory-Based Approach in Go-program "KATSUNARI"

Shinichi Sei ssei@ssl.fujitsu.co.jp
Toshiaki Kawashima kawkaw@ssl.fujitsu.co.jp
Fujitsu Social Science Laboratory Ltd.

Abstract

To make a strong Go-program, programmers analyze the thinking process of expert players and devise some algorithms to make good moves based on the result. However, such work is very difficult and there isn't a strong Go-program. We had proposed a new method which makes good move by using a large quantity of pattern knowledge extracted from professional players' games. And we had developed a Go-program "KATSUNARI"[1] which included the method. Because our method can make good moves by using pattern knowledge directly, we don't need difficult work of usual way. To make KATSUNARI stronger, we are improving mainly its database which has pattern knowledge. We classified pattern knowledge into two categories: the knowledge which show basic skill and the knowledge difficult to understand validity. Examples of former knowledge are pattern of JOSEKI and pattern of local competition and examples of latter knowledge are professional players' moves. We made two types of databases which included each knowledge. By using these databases properly, KATSUNARI can make good moves under any board situation than before. In this paper, we describe about new databases of KATSUNARI and how KATSUNARI uses them.

1 Introduction

The chess program "DeepBlue" defeated the human world champion Kasparov in 1997. However, the strongest Go-program has only strength of intermediate player of amateur[2]. Chess program became strong mainly by using search method, but the same method isn't suitable to Go-program. Reasons of it are the very large search space of Go-game compared with Chess and the difficulty of the evaluation of the board situation. Therefore, the new method to make strong Go-game program is necessary.

Most of Go-programs adopt the approach of imitating human strong player's thinking process. Programmers of these Go-programs analyze how human expert players recognize board situation and how they make moves. And they devise original algorithms to make move like expert players' and implemented those algorithms in their programs [Fotland, 1993][Chen, 1990][Sanechika, 1988][Fost, 1997][Fost, 1998]. However, this approach has a problem that all knowledge to make good move cannot be represented because the analysis of expert players' thinking process is too difficult. Although moves created by their original algorithms are sometimes good in typical situations, they aren't good in complicated situation like actual games. From these reasons, Go-programs have only strength of intermediate player of amateur.

Recently, there are some programs that adopt approach using learning functions[Brügmann, 1994] [Cazenave, 1996][Enderton, 1991][Stoutamire, 1991], but these programs are not strong yet.

We had proposed a new method which makes good move by using a large quantity of pattern knowledge extracted from professional players' games[Sei, 1994]. And we had developed a Go-program KATSUNARI which included the method[Sei, 1996]. A pattern knowledge consists of the professional players' move and local arrangement of stones around the move. We collected a large quantity of patterns automatically from professional players' games and make database retained them. KATSUNARI's process to make move is (1) compare stone arrangement on board with each pattern (2) propose written move in similar pattern as candidates (3) evaluate each candidate (4) select the best candidate by result of evaluation. Because KATSUNARI makes good move by using directly pattern knowledge extracted from professional players' games, we don't need analysis and devising original algorithms of usual way.

KATSUNARI participated in 1996 and 1997 World Open Computer Go Championship, FOST-CUP, to evaluate our method. However, we couldn't leave good

[1] This name is 勝也 in Japanese.

[2] The 1997's champion program "HandTalk" and 1998's champion program "Silver IGO" were given a 3-kyu diploma by Nihon-Kiin.

records: ranking were 13th in 19 programs (4 wins and 5 loses) in 1996, and 20th in 38 programs (5 wins and 5 loses) in 1997. As a result of our investigation about why records were not good, we found that our method has some defects[Sei, 1998]. To make KATSUNARI stronger, we are improving mainly database which has pattern knowledge. In this paper, we describe about new databases of KATSUNARI and how KATSUNARI use them.

2 Memory-Based Approach

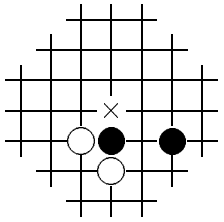
To make a strong Go-program, programmers analyze the thinking process of expert players and devise some algorithms to make good moves based on the result. However, such work is very difficult and there isn't a strong Go-program. It is easy to devise only some algorithms to accomplish single-purpose such as to surround territory or to capture stones. However, those algorithms can't frequently make a suitable move at complicated situation like actual game. Moreover, it is difficult to devise some algorithms to accomplish multi-purpose at same time. Therefore, the new method to make a strong Go-game program is necessary.

We had proposed to apply memory-based approach to make a strong Go-program. Memory-based approach means directly using a lot of knowledge which consists of problem and its solution to solve problems. Typical example which adopted this approach is Memory-Based Reasoning(MBR)[Stanfill, 1986]. The method retains a lot of previous experiences, retrieves the best similar experience from a collection and outputs it as the solution of a given problem. MBR has the feature that it can outputs good answer in a field where methods to solve problems are not established. There are some research reports using MBR in the pronunciation of word[Stanfill, 1986], machine translation[Kitano, 1993][Sato, 1993], and so on.

2.1 Pattern Knowledge

It is said that to make good stone arrangement is one of important tactics in Go-game. And we also know that strong human players make moves by considering local arrangement of stones. In Tsumego problem, there is a report that strong players use pattern knowledge that is the pair of move and local arrangement of stones around of move[Yoshikawa, 1996]. Therefore, we considered that using pattern knowledge is effective to make strong Go-program.

We show a example of KATSUNARI's pattern knowledge in Figure1. We defined a octagon shape as shape of pattern[3]. Reasons why we decided this shape are followings. Strong players usually don't consider positions of long distance from the position of candidate. Many of words(Keima, Ogeima, Ikken-Tobi, Niken-Tobi,...) which show the relation of stones in Go-terms are included in this shape. And we set center of pattern as the the position of candidate.



×: Position of candidate at Black's turn It is the position where professional player put in actual game.

Figure 1: Example of Pattern

2.2 Database Creation

Programs which adopted memory-based approach need a lot of knowledge. However, it is difficult to represent strong player's various knowledge into pattern. Moreover, we can't write down a lot of pattern knowledge by ourselves.

Before creating database, we classified pattern knowledge into two categories(Figure2): the knowledge which show basic skill and the knowledge difficult to understand validity. Examples of former knowledge are pattern of JOSEKI and pattern of local competition. Examples of latter knowledge are professional players' moves because those moves are so advanced that we can't understand their meaning and worth. We collected former knowledge from some textbooks and dictionaries about Go-game. We extracted latter knowledge from many professional players' games. We made the program to extract patterns automatically and collected about 50,000 patterns from about 400 games by professional players.

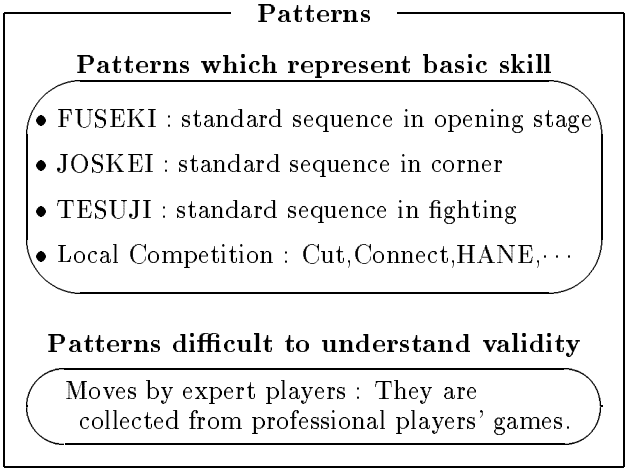


Figure 2: Classification of Pattern

3 Go-program KATSUNARI

In this section, we describe about method of KATSUNARI which adopted memory-based approach.

[3]This shape is the shape of improving KATSUNARI. The shape of old KATSUNARI is a little different from this.

3.1 Process to Make Move

We show the KATSUNARI's process to make move in Figure3. The detail of process is following.

1. *Pattern Candidate* Creation
   for each empty point on board
   (a) compare with arrangement of stones around the point and arrangement of stones of each pattern in database, and find out same pattern
   (b) propose this point as pattern candidate, if same arrangement of stones is found
2. *Capture Candidate* Creation
   for each stone on board
   (a) investigate status of stone (alive/dead/neutral)
   (b) propose move to capture/escape the stone, if status is neutral
3. Next Move Selection
   for each candidate
   (a) image a board where a candidate is temporary put, and estimate the board situation
   (b) adjust the value from considering the degree of importance of stone
   (c) select candidate with the highest value as next move

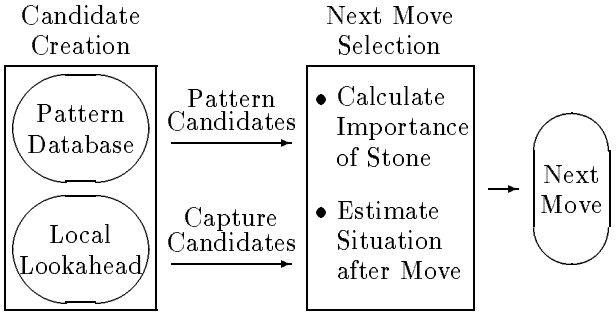


Figure 3: Process to Make Move

3.2 Candidates

KATSUNARI creates two kinds of candidates, these are *Pattern Candidate* and *Capture Candidate*. *Pattern Candidate* means candidates which created by using pattern knowledge, and it makes good arrangement of stones. *Capture Candidate* means candidates to capture enemy's stones and the candidates to escape family stones. And KATSUNARI creates some kinds of *Pattern Candidate*, these are *JOSEKI Candidate*, *FUSEKI Candidate*, *TESUJI Candidate*, *Small Pattern Candidate* and *Large Pattern Candidate*.

Capture Candidate
Capture Candidate means candidates to capture enemy's stones and the candidates to escape family stones. This candidate is created by local lookahead. Although some of candidates to capture/escape stones are created in creating *Pattern Candidates*, most of candidates to capture/escape stones aren't created. Because, KATSUNARI doesn't need to mind to make good arrangement of stones but to reduce DAME(liberty points) of stones to create *Capture Candidate*.

FUSEKI Candidate
FUSEKI is one of Go-terms and it is a standard sequence in the opening stage. FUSEKI is so analyzed in detail by human player and it is published as FUSEKI dictionary. We collected FUSEKI patterns from published FUSEKI dictionary and made FUSEKI pattern database. The bounds of FUSEKI pattern is larger than another kinds of pattern because program needs to considers wide scope(the size of the most large pattern is same as whole board). This candidate is established to overcome one of defects in old KATSUNARI. Old KATSUNARI was weak in the opening stage because it didn't have such large pattern.

JOSEKI Candidate
JOSEKI is one of Go-terms and it is a established or standard sequence at corner. It is so analyzed in detail by human player and it is published as JOSEKI dictionary as FUSEKI. We collected JOSEKI patterns from published JOSEKI dictionary and made JOSEKI pattern database.

TESUJI Candidate
TESUJI is one of Go-terms and a standard sequence to accomplish specific purpose. We collected TESUJI patterns from published TESUJI dictionary and some textbooks.

KATSUNARI estimates the worth of each candidate to select the best one. For estimation, KATSUNARI images a board where the candidate move is temporarily put and calculates how many points KATSUNARI leads. However, in this method, it is difficult to find out the move where the effect appears afterwards, such as sacrifice move. Then, to calculate the worth accurately, we added the pattern for evaluation to TESUJI pattern. We show examples in Figure4. When KATSUNARI estimates this candidate, it images the board where the pattern for evaluation is set. This candidate is established to overcome one of defects in old KATSUNARI, too.

Small Pattern Candidate
This is the candidate for local competition, e.g. Cut, Connect, HANE, NOBI, OSAE, TSUKIDASHI, FUKURAMI, etc. To make such moves, human players usually consider only arrangement of stones in small area. Then, we prepared pattern with small bounds to create these moves. We designed square(3×3) as the shape of pattern, and we set position of move at center of shape.

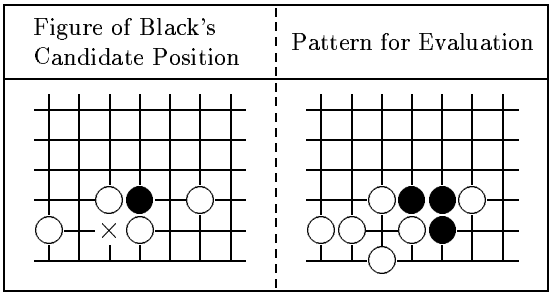
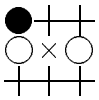


Figure 4: TESUJI Pattern

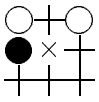
This candidate is established to overcome one of defects in old KATSUNARI, too. When there are many stones in small area, the size of old KATSUNARI's pattern is too wide to find same pattern. It often occurs especially in middle stage and end stage.

We added *the degree of emergency* to each *Small Pattern*. The degree is used to prune unnecessary candidates. KATSUNARI saves *Small Pattern Candidates* with high degree and abandons another. To calculate *the degree of emergency* accurately, KATSUNARI checks several items, these are arrangement of stones, amount of liberty of each stone, status of each stone(alive/dead/neutral) and each stone's distance from edge. We show examples in Figure5.

The technique of these moves is basic in Go-game. Even amateur player knows their meaning and worth. We could write all patterns by ourselves because the amount is small (about 1,000).



```
if( Left white's liberty ≤ 2 ){
    Emergency = 80;
}else if( Top left black's liberty ≤ 2 ){
    Emergency = 0;
}else{
    Emergency = 60;
```



```
if( Bottom line is edge ){
    Emergency = 80;
}else{
    Emergency = 60;
```

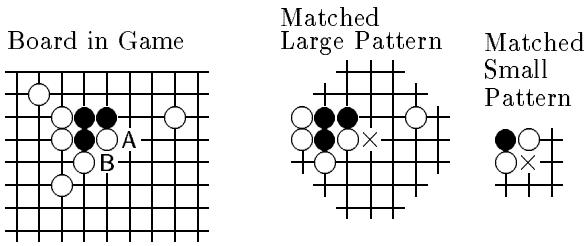
Figure 5: Small Pattern

Large Pattern Candidate
This patterns are extracted from professional players' games. We described about this candidate in previous section and showed a example pattern in Figure1.

Pattern Candidate Pruning
After KATSUNARI creates many *Pattern Candidate*, it prunes unnecessary candidates before evaluating each candidate.

The bounds of pattern is used to decide whether to prune or not. We explain about it in Figure6. When the left figure's board situation is given, KATSUNARI creates two kinds of different *Pattern Candidate*, there are *Large Pattern Candidate* at position A and *Small Pattern Candidate* at position B. However, if KATSUNARI don't create *Large Pattern Candidate* at B, *Small Pattern Candidate* of B is pruned. We can expect that the move created by considering wide scope is better than the move created by considering small scope. KATSUNARI considers that the candidate of B is not better than the candidate of A, because the bounds of *Large Pattern Candidate* of A covers *Small Pattern's* of B completely in this case.

KATSUNARI does this pruning for another kinds of candidate, too.



A: Position of *Large Pattern Candidate*
B: Position of *Small Pattern Candidate*
KATSUNARI prunes candidate of B

Figure 6: Pattern Candidate Pruning

4 Evaluation of KATSUNARI

4.1 FOST CUP

Because we are in the middle of improvement of KATSUNARI, we can't evaluate our method at the present. Indeed, a part of *Small Pattern Candidate* was implemented on 1997, and *FUSEKI Candidate* and *TESUJI Candidate* were implemented on 1998. Pruning function hasn't been implemented yet. We are still collecting pattern knowledge, there are only about 400 FUSEKI patterns and about 200 TESUJI patterns in KATSUNARI.

However, KATSUNARI participated in 1998 World Open Computer Go Championship, FOST-CUP, to evaluate the strength at that time. Although the record of present KATSUNARI isn't good, the record becomes better every year(Table1). We can expect that KATSUNARI become stronger after improvement.

Table 1: Results in FOST-CUP

Rank	wins - loses	year
1996	13th in 19 programs	4 - 5
1997	20th in 38 programs	5 - 5
1998	15th in 38 programs	4 - 2

4.2 Versus "The Strongest Game of Go"

"The Strongest Game of Go" is the name of commercial version of "Go4++". "Go4++" is one of top class Go-programs, 2nd in 1996 and 3rd in 1997 World Open Computer Go Championship. We did test matches with it in several times. Although KATSUNARI could win a few times and the average of score was about 20 points behind. But, the strength of advanced player commented that the difference of 20 points is small difference. We show one of scores in Figure7.

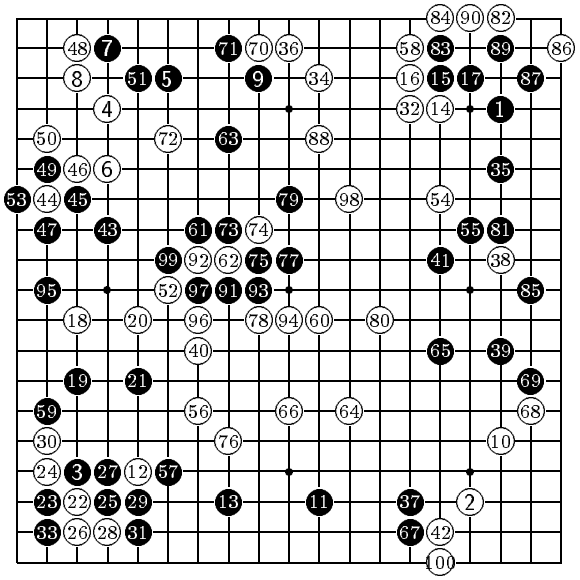


Figure 7: KATSUNARI(Black) vs The Strongest Game of Go

5 Related Go-Programs

Although most of Go-programs use a kind of original pattern knowledge, the aim of it is limited. It is generally used to create JOSEKI candidate or to create move to make/break *eyes*.

Recently, there are several programs which adopted memory-based approach like KATSUNARI. "Monkey Jump" and "KuruKuru" have a huge decision tree created from a lot of professional players' games, and they make move by using them[Fost, 1998][Fost, 1997]. To our regret, detailed reports about the method haven't been published yet.

6 Conclusion

We are improving Go-program KATSUNARI which adopted memory-based approach. KATSUNARI has various pattern databases created from different type of knowledge and makes good move by using these databases properly.

KATSUNARI's record in Computer Go Championship became better by our improvement every years. We can expect that KATSUNARI to become stronger after improvement.

Acknowledgment

We would like to thank Yukiyo Okano and Yasuhiro Ike for their helps in programming.

References

[Brügmann, 1994] Brügmann, B. Monte Carlo Go. available from Go Archive as Go/comp/mcgo.tex.Z, 1994

[Cazenave, 1996] Cazenave, T. Automatic Acquisition of Tactical Go Rules. *Proc. of the 3rd Game Programming Workshop*, pp.10-19, 1996

[Chen, 1990] Chen, K. The Move Decision Process of Go Intellect. *Computer Go*, No.14, pp.9-17, 1990

[Enderton, 1991] Enderton, H.D. The Golem Go Program. *CMU-CS-92-101*, Carnegie Mellon University, 1991

[Fost, 1997] *The Profile*, FOST-CUP, 1997

[Fost, 1998] *The Profile*, FOST-CUP, 1998

[Fotland, 1993] Fotland, D. Knowledge Representation in The Many Faces of Go. available from Go Archive as Go/comp/mfg.Z, 1993

[Kitano, 1993] Kitano, H. A Comprehensive and Practical Model of Memory-Based Machine Translation. *Proc. of 13rd International Joint Conference on Artificial Intelligence*, pp.1276-1282, 1993

[Sanechika, 1988] Sanechika, N. et al. "Go Generation" A Go Playing System. *ICOT Technical Report*, TR-545, 1990

[Sato, 1993] Sato, S. Example-Based Translation of Technical Terms. *5th Int. Conf. on Theoretical and Methodological Issues in Machine Translation*, 1993

[Sei, 1994] Sei, S. and Kawashima, T. The Experiment of Creating Move from "Local Pattern" Knowledge in Go Program. *Proc. of the 1st Game Programming Workshop*, pp.97-104, 1994 (in Japanese)

[Sei, 1996] Sei, S. and Kawashima, T. The Experiment of the Go-program "KATSUNARI" using Memory-Based Reasoning. *Proc. of the 3rd Game Programming Workshop*, pp.115-122, 1996 (in Japanese)

[Sei, 1998] Sei, S. and Kawashima, T. Evaluation of the Method to Create Candidates using Pattern Knowledge Extracted from Professional Player's Scores in Go-Game. *CGF Journal*, Vol.2, 1998 (in Japanese)

[Stoutamire, 1991] Stoutamire, D. Machine Learning Applied to Go. *MS thesis*, Case Western Reserve University, 1991

[Stanfill, 1986] Stanfill, C and Waltz, D. Toward Memory-Based Reasoning. *Communications of the ACM*, Vol.29, 1986.

[Yoshikawa, 1996] Yoshikawa, A. and Saito, Y. Can not solve Tsume-Go Problems without looking ahead? *Proc. of the 3rd Game Programming Workshop*, pp.76-83, 1996 (in Japanese)