3C3-IOS-2-7

An Efficient Index Structure for Go

Yu-Jie Ho[1]
[1]Dept. of Computer Science and Information Engineering, National Dong Hwa University, Taiwan.

Shun-Chin Hsu[2]
[2]Department of Information Management, Chang Jung Christian University, Taiwan.

Shi-Jim Yen[1]
[1]Dept. of Computer Science and Information Engineering, National Dong Hwa University, Taiwan.

1. Introduction

Pattern matching is an important problem in computer Go. Many pattern matching methods are proposed in [1][11][12]. Most of them only discuss about how to find a pattern of pattern database in a game board. This thesis discusses how to find a query pattern in a lot of Go game records and how to build a Go game records information retrieval system. This system can select and return to the user desired pattern from a large set of Go game records in accordance with criteria specified by the user. In this system, inputs are query patterns and outputs are the game record information, such as number of game records, move sequence, players' names and final results. Information retrieval concepts and techniques will be applied in this system.

Retrieval a query pattern from a lot of game records is useful for Go player and computer Go programmers. A Go player may be interesting on how to play in a special situation. For example, it is difficult to most Go players for life-and-dead problems. If he has the Go game records information retrieval system, he may find this life-and-dead problem occurred in the game record. Some Go players may be interesting to know what kind of opening is often used by Go professors. The winning rate of each opening style is also interesting. The system will be useful.

For computer Go development, the Go game records information retrieval system can help the programmers to maintain the pattern database and find more useful patterns. When the pattern database get large, pattern inconsistence problem will occur. Retrieve those inconsistence patterns from the history record game record of the program will help to solve this problem. Many researches try to apply matching learning technique to find useful patterns based on Go game records. Researches regards game records as text strings by suitable encoding from each move to a character, and uses approaches of natural language processes and statistics to acquire sequence patterns are proposed in [4][5][6]. In [7][8], the authors proposed methods of learning to predict Life and Death and score final positions in the game of Go from game records by well designed data structure and good classifiers. Game records are used as the training data of neural networks [9]. A rule based expert system could be build up by knowledge acquisition from game records [10].

Contact: Shi-Jim Yen, Dept. of CSIE, National Dong Hwa Uni., Taiwan, Address: No. 1, Sec. 2, Da Hsueh Rd., Shoufeng, Hualien 97401, Taiwan. Tel: +886-3-8634031, FAX: +886-3-8634010, E-mail address: sjyen@mail.ndhu.edu.tw

2. Proposed approaches

2.1 3.1 Apply KMP algorithm on Go game record

KMP algorithm helps to skip unnecessary matching, we apply 1–dimensional string matching algorithm on 2–dimensional pattern and board by regarding a 2-D pattern as a set of 1-D pattern. The steps are as follows.

Step1. Choose the row which is most suitable for KMP algorithm as the target row pattern. Row patterns in the set will be ranked according to their

i) Fixed state: Choose the row pattern with fixed state.

ii) Length

iii) If the row pattern has the same length, then choose the one has highest degree of repetition. (i.e. Frequency of the same substring appears in the pattern. The prefix function has contained this information)

Step2. Search the target pattern decided in step 1 on the board (row by row) by using KMP algorithm.

Step3. If the target pattern appear at somewhere on the board, then check every element in the query pattern with points around by bit mapping sequentially:

P: Query pattern

S: Board situation around the matched pattern.

R := P AND S

If R := S, then success Else fail

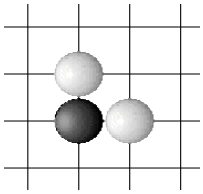
Repeat 1~3 until all the rows are scanned.

2.2 Feature Patterns

In order to construct the index structure, we have to extract the feature of data, and use them as the indexing key. In our approach, we use 400 "featured patterns". Featured patterns are some patterns that appear frequently in GO game. These patterns are part of pattern databases of the computer program "Jimmy", which is developed from 1994[1]. Almost all significant moves can be recognized by the pattern database system. Each of them has its own meaning in Go game, they could be considered as "jump", "knight`s move", or "diagonal" in Go game, but most of them are more complicated.

Figure 1 shows the way we represent the feature patterns. We use 4 bits to express 4 possible states of every point. Bit 1 means this point could be empty or not; bit 2 means this point could be black stone or not; bit 3 means this point could be white stone or

not, and bit 4 means this point could be border or not. Every number in the array is generated from 4 bits. [3]



1111
1411
1241
1111

Bit 4	Bit 3	Bit 2	Bit 1
Border	White Stone	Black Stone	Empty
0/1	0/1	0/1	0/1

Figure 1. A representation of feature pattern

For example, a number “3” in the array means its relative binary number is “0011”, and this means this point which is marked by “3” could be either “Black Stone” or “Empty”.(The “*” point means it is a important position which has special meaning or value for player. In this paper, it will be looked as “1”)

Approach in 2.1 handles most 2-D patterns. However, we still have to generate all possible patterns if every row pattern in a query pattern contains unfixed element(s) (i.e. points marked by “3”,”5”,”6”). For example, a row pattern “3144” will be turned into “1144” and “2144”. Because every “3” means that point can be “1”(empty) or “2”(black).

These feature patterns are good indexing keys. Each of them has its own characteristics and meanings. By using them and a suitable indexing structure, we can classify game records systematically.

2.3 Index structure and searching approach

We now combine integrate our feature pattern with a popular index structure “inverted list”. We use proposed pattern searching approach to find board situations contain each feature pattern, and build an inverted file for our game records. All board situations were classified according to the feature pattern(s) they contain. This will be a good characteristic for matching processing. Almost every reasonable query pattern contains at last one feature pattern, and it is easy to perform logical operations such like “AND” “OR” under this structure.

After the inverted file was built up, pattern matching approaches proposed in [11][12][13] will be applied on every query pattern to check which feature patterns it contains. We then know which set of board situations the query pattern may appear according to the result. For example, if the pattern matcher shows that the query pattern contains feature pattern 1 and feature pattern 3, we will only search it in the set (1∩2).

In pattern searching, we have to consider all the actions (rotation, reflection, and color changing) of query patterns. All patterns (16 situations) generated from query patterns through these situations will be found.

3. Conclusion

In this paper, we propose a simple, yet efficient index structure for Go game records. This approach is based on KMP algorithm and the feature patterns extracting from the pattern database of Go program “Jimmy”. This structure could improve the speed of pattern searching to a satisfactory time.

Comparing with pattern searching in unstructured game record database using KMP algorithm, the proposed structure classifies board situations according to significant feature patterns they contain(most significant query pattern contain at least one of these feature pattern), therefore, it can help pattern matcher to skip most unnecessary matching. With proposed structure, we are able to construct a go game record information retrieval system which handles both pattern view and text view queries. This system will be useful for Go players and computer Go researchers.

References

[1] Shi-Jim Yen, Design and Implementation of a Computer GO Program JIMMY. Phd. Thesis, National Taiwan University, Taiwan. (in Chinese)

[2] Shiou-jiun Chen,“Numeric Indexing Approach for Music Database Retrieval,” Chaoyanh University, Master Thesis, 2002.

[3] Kuo Tung Hsu, ”Pattern recognization in Go game records”, National Dong Hwa University, Master Thesis, 2004.

[4] Teigo NAKAMURA, Takashi KAJIYAMA, “Feature Extraction from Encoded Texts of Moves and Categorization of Game Records,” Department of Artificial Intelligence, Kyushu Institute of Technology, Japan.

[5] Teigo NAKAMURA, “Acquisition of Move Sequence Patterns from Game Record Database Using n-gram Statistics,” Department of Artificial Intelligence, Kyushu Institute of Technology, Japan.

[6] Teigo NAKAMURA, Takashi KAJIYAMA, ”Automatic Acquisition of Move Sequence Patterns from Encoded Strings of Go Moves”, Department of Artificial Intelligence, Kyushu Institute of Technology, Japan.

[7] Erik C.D. van der Werf, Mark H.M.Winands, H. Jaap van den Herik and Jos W.H.M. Uiterwijk, “Learning to Predict Life and Death from Go Game Records,” Dept. of Computer Science, Universiteit Maastricht.

[8] E.C.D. van der Werf, H.J. van den Herik, J.W.H.M. Uiterwijk, “Learning to Score Final Positions in the Game of Go,” Department of Computer Science, Universiteit Maastricht.

[9] George Konidaris Dylan Shell Nir Oren, ”Evolving Neural Networks for the Capture Game,” School of Computer Science University of the Witwatersrand, Johannesburg.

[10] Takuya Kojima Atsushi Yoshikawa, ”Knowledge Acquisition from Game Records,” NTT Communication Science Labs, Japan.

[11] Mark Boon, “A Pattern Matcher for Goliath,” Computer Go, No.13, pp.12-24, 1990.

[12] Martin Mueller, “Pattern Matching in Explorer,” the Game Playing System Workshop, pp.1-3, Tokyo, Japan, 1991.

[13] Yu-Tang Lee, “Pattern Matching in Go based on Patricia Tree,” National Dong Hwa University, Master Thesis, 2004.