Games of No Chance
MSRI Publications
Volume **29**, 1996

Using Similar Positions to Search Game Trees

YASUHITO KAWANO

Abstract. We propose a method that uses information on similar positions to solve Tsume-shogi problems (mating problems in Japanese chess). Two notions, priority and simulation, are defined for this method. A given problem is solved step-by-step according to the priority. Simulation often allows us to omit searching on each step. A program made by the author and based on this method solved a celebrated problem requiring over 600 plies (half-moves) to mate.

1. Introduction

It would be useful in board games if the experience learned from one position could be applied to other, similar, positions. However, positions that seem to only slightly differ on the board can actually differ greatly in terms of strategy. This paper proposes a method that uses information on similar positions to solve *Tsume-shogi* problems (mating problems in Japanese chess, the rules of which are explained in Section 2).

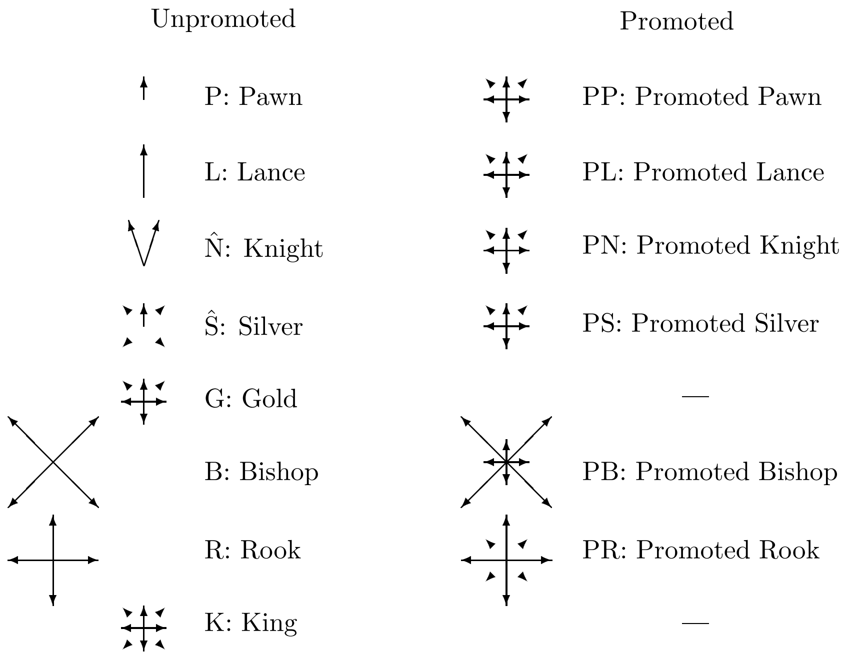
The method is based on the notions of priority and simulation. Priority means that we look at certain moves of our opponent earlier than others. Intuitively, we say that a position **P** simulates another position **Q** if we can mate on **Q** according to the mating sequence of **P**. Roughly speaking, our problem-solving strategy has two steps. First, we restrict ourselves to our opponent's moves of high priority. Second, we check whether the position **P** where the solution has already been obtained simulates a position **Q** having a lower priority. If **P** simulates **Q**, we can dispense with the search for the latter position.

We discuss a Tsume-shogi program by the author, based on this method and considered to be one of the strongest Tsume-shogi programs available today. It was the first program to solve a celebrated and long-unsolved (by computer) problem, requiring over 600 plies (half-moves) to mate.

2. Shogi and Tsume-shogi

Shogi is a board game, similar in origin and feel to chess [Fairbairn 1989; Wilkes 1950; Leggett 1966]. It was established around the fourteenth century in Japan. The biggest difference between Shogi and chess is that captured pieces in Shogi can be put back into play, on the capturer's side, so the number of pieces in play tends to remain high.

Shogi is played on a 9×9 board, with, initially, twenty pieces for each player. All pieces are the same color (so they can be reused by the opponent), ownership being distinguished by the way they face. In diagrams, pieces are represented by Kanji characters, but here we use the initial letters of their English translations (with a circumflex on top of S and N, which resemble their upside-down counterparts). The three rows nearest a player are than player's domain. A piece is *promoted* when it moves into, inside, or out of the opponent's domain, and it remains promoted until it is taken. Here are the names and movements of the pieces:



The Pawn can move only one square straight ahead. It does not have the chess pawn's initial move or diagonal capturing move. The Lance moves straight ahead. The Knight moves two squares forward and one sideways (a subset of the moves of the chess knight); it is the only piece that can jump over others. The Silver can move one square forward in any direction or one square diagonally backward. The Gold can move one square forward in any direction, one square to the side, or one square straight back. When the Pawn, Lance, Knight, and Silver are promoted, their allowed moves become the same as the Gold's. The Rook and Bishop have the same moves as their chess counterparts. When they are promoted, they have the power to move one square in the other four directions. The King can move one square in any direction. The Gold and the King, however, can not be promoted.

Instead of a regular move, a player can *drop* a piece previously captured by him into a vacant square, provided this does not break one of the following rules:

DOUBLED PAWN: When a player's Pawn is already on a column, he is not allowed to drop another Pawn on the same column.
DROPPED-PAWN MATE: Mating by dropping a Pawn is prohibited. (It is legal to mate by moving a Pawn that is already on the board.)
DEADLOCK: It is illegal to move or drop a piece onto a square if it cannot move from there.

Here is the initial Shogi set-up:

L	N̂	Ŝ	G	K	G	Ŝ	N̂	L
	R						B	
P	P	P	P	P	P	P	P	P
P	P	P	P	P	P	P	P	P
	B						R	
L	N̂	Ŝ	G	K	G	Ŝ	N̂	L

Tsume-shogi is a Shogi mating problem. The data is a Shogi position; a solution consists in a sequence of legal moves satisfying several conditions:

- Each move of the mating player must be optimal, in the sense of leading to mate as soon as posbible.
- The position after each move of the mating player must be a check.
- Each response of the opponent must be optimal, in the sense of delaying mate as long as possible. (Interpositions by moving or dropping a piece must not be *muda*: this term is explained below.)

The chain of moves is called *Hontejun* (main sequence); its length, the number of plies, is called *Tsume-tesu*. The main sequence should be unique; more specifically, in the positions of the main sequence, each mating player's move except for the last one must be unique, otherwise the problem is regarded as defective.

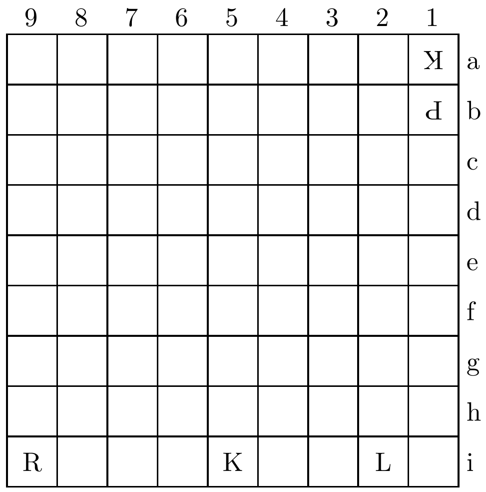
It is said that over 100,000 complete Tsume-shogi problems have already been published. For most of them, mating is completed in less than twenty moves, but there are also hundreds of problems requiring more than 100 plies. At present, the 1519-ply "Microcosmos" problem is believed to be the one requiring the largest number of moves to complete.

We call the solution of Tsume-shogi a *mating sequence* in this paper.

3. Tsume-shogi Programs

Tsume-shogi programs use one of two approaches: iterative deep search, or best-first search. At present, the best programs using iterative deep search can solve problems with 25 or less plies without fail and in a practical time. They do, however, tend to fail on problems with 27 or more plies since a lot of time is needed. On the contrary, programs using the idea of best first search often solve problems requiring large numbers of moves in a short time; but they cannot be guaranteed to solve any given problem [Ito and Noshita 1994; Matsubara 1993]. Besides this, a hardness result for generalized Tsume-shogi problems has been shown in [Adachi et al. 1987].

Because of the possibility of dropping, the number of candidate move is usually much larger than in chess. This rule causes problems not found in computer chess. One big problem is the treatment of interposed pieces. Consider the following easy problem in Tsume-shogi:



no piece in hand

The solution to this problem starts with the move 9i–9a+. (We will use $*$ to denote a dropped piece and + for promotion.) However, the opponent can drop any piece on any square between the Promoted Rook and the King, because all absent pieces are in the opponent's hand. This can be done seven times, so the mating sequence is 15-ply long. These interposed pieces are called *muda-ai*, and human players usually ignore them, because they just prolong the mating sequence without altering it in any essential way. In contrast, a computer will search through hundreds of thousands of positions if it does not understand muda-ai.

Muda-ai has no formal definition. Informally, it means any interposed piece incapable of essentially changing the mating sequence. Yoshikazu Kakinoki [1990] proposed an algorithm to determine muda-ai. He defines muda-ai recursively as follows:

(i) An interposed piece is muda if it is captured at the moment the King is mated.
(ii) Suppose the King is mated after n moves in the absence of a given interposition. The interposed piece is muda if it is immediately captured and not reused, and the King is mated in n moves, not counting the interposing move and the capture.

Although there exists no exact algorithmical expression of muda-ai, it is known that Kakinoki's algorithm determines muda-ai precisely in most cases, so his criterion is regarded as a standard. In practice, a definition that "localizes" the concept of muda-ai by basing it on "imaginary pieces" is adopted by many programmers. More precisely, suppose the opponent has an infinite supply of pieces to drop. We judge whether the interposition of such an imaginary piece at a given square is muda by Kakinoki's algorithm, and we decree that the mudaness of a real dropped interposition at that square is the same. This criterion for muda-ai sometimes leads to different results from Kakinoki's original criterion (which takes into account whether or not the opponent actually has additional pieces to drop), but it is computationally much more manageable, since it avoids an explosion of the search space.

4. Using Similar Positions

Since muda-ai reflects the essential equality of mating sequences, Kakinoki's algorithm should be recognized as a method capable of determining this essential equality and its applications. We consider the extension of this decision of muda-ai.

Priority. We introduce the notion of priority before we define the notion of simulation. Priority is the total searching order of the opponent's next moves at a position: moves of low priority will not be searched for until moves of high priority are searched for. The same priority is used with moves of corresponding positions. In Kakinoki's algorithm, moves other than interpositions have the highest priority, and interpositions near the King have higher priorities than those farther away. In addition, for example, captures by many opponent's pieces except for the King are noted as low priorities. The priority is set in the following discussion.

Though priority is defined locally, we can extend it to a partial order on a set of moves (nodes) in a game tree for a given problem. We set the priority of each mating player's move to be equal to its parent's. We also introduce the following relations:

(a) The problem (the root node) has the highest priority in the game tree,
(b) The priority of a move having the highest priority in the local sense is the same as its parent's, and
(c) The priority of a move not having the highest priority in the local sense is lower than its older siblings' descendents'.

We solve each given problem according to this extended priority. We first restrict ourselves to our opponent's moves of highest priority. Each position $\mathbf{Q}$ not having highest priority goes unnoticed until we solve the older sibling of the root node of the subtree consisting of moves having the same priority as $\mathbf{Q}$. If there exists a position $\mathbf{P}$ that simulates the position $\mathbf{Q}$, we can dispense with the search for $\mathbf{Q}$. To find such a position $\mathbf{P}$, we appoint candidates beforehand for each position $\mathbf{Q}$, where the candidates satisfy the relation with $\mathbf{Q}$ in high probability; otherwise, we use a transposition table to find candidates.

Simulation. We define a relation $\mathbf{P} \geq \mathbf{Q}$ between positions, read "$\mathbf{Q}$ simulates $\mathbf{P}$". Let $\mathrm{pri}(m)$ and $\mathrm{pri}(\mathbf{Q})$ be the priorities of m and $\mathbf{Q}$, where m is a move and $\mathbf{Q}$ is a node of the game tree. Let $\mathbf{P}^{\mathrm{pri}(\mathbf{Q})}$ be the problem obtained from $\mathbf{P}$ by restricting ourselves to moves m of our opponent that satisfy the condition $\mathrm{pri}(m) \not\leq \mathrm{pri}(\mathbf{Q})$. We write $\mathbf{P} \geq \mathbf{Q}$ when we can mate the King of $\mathbf{P}^{\mathrm{pri}(\mathbf{Q})}$ and we can mate the King of $\mathbf{Q}$ according to one of the mating sequences for $\mathbf{P}^{\mathrm{pri}(\mathbf{Q})}$. Formally, we define $\mathbf{P} \geq \mathbf{Q}$ recursively as follows:

(i) $\mathbf{Q}$ is a position for a mating player's move.Let m be the first move of one of the mating sequences for $\mathbf{P}^{\mathrm{pri}(\mathbf{Q})}$. If m is not a checking move for $\mathbf{Q}$, then $\mathbf{P} \not\geq \mathbf{Q}$. Otherwise, let $\mathbf{P}', \mathbf{Q}'$ be the positions obtained from $\mathbf{P}, \mathbf{Q}$ by move m. By definition, $\mathbf{P} \geq \mathbf{Q}$ if and only if $\mathbf{P}' \geq \mathbf{Q}'$.
(ii) $\mathbf{Q}$ is a position for an opponent's move.If $\mathbf{Q}$ is a mate, set $\mathbf{P} \geq \mathbf{Q}$. If $\mathbf{Q}$ is not a mate, generate all of the opponent's next moves m_i, for $i \in \Lambda$. If they are not properly included in the set of next moves of $\mathbf{P}^{pri(\mathbf{Q})}$, then $\mathbf{P} \not\geq \mathbf{Q}$. Otherwise, let $\mathbf{P}'_i$ and $\mathbf{Q}'_i$ be the positions obtained from $\mathbf{P}$ and $\mathbf{Q}$ by move m_i. By definition, $\mathbf{P} \geq \mathbf{Q}$ if and only if $\mathbf{P}'_i \geq \mathbf{Q}'_i$ for all $i \in \Lambda$.

Unfortunately, the relation $\geq$ is not suitable for application to searching in Tsume-shogi, even though it is natural. For example, an interposed piece at in the first of these diagrams would be muda, although $\mathbf{P} \not\geq \mathbf{Q}$:

$\mathbf{P} =$

5	4	3	2	1	
				L	a
					b
	PR			K	c
		G		P	d
					e

no piece in hand

$\mathbf{Q} =$

5	4	3	2	1	
				L	a
					b
		PR		K	c
		G		P	d
					e

no piece in hand

The reason why $\mathbf{P} \not\geq \mathbf{Q}$ is this. In $\mathbf{Q}$, we assume that the opponent will drop a Pawn on 2c. The recorded next move of $\mathbf{P}^{\mathrm{pri}(\mathbf{Q})}$ is 4c–2c. But since 4c in $\mathbf{Q}$ is vacant, $\mathbf{Q}$ does not simulate $\mathbf{P}$.

To avoid cases such as this, in which the pieces cannot be moved, we introduce the relation $\geq_f$, which permits the mating player to choose the next move in the above definition. We introduce a choice function for the mating player $f : (m, \delta) \mapsto \Omega_{m\delta}$, where we denote the difference of two positions by δ, and the set of candidates by $\Omega_{m\delta}$. We denote by $\geq_f$ the binary relation defined by replacing case 1 in the definition of $\geq$ with the following sentence:

(i) $\mathbf{Q}$ is a position for a mating player's move. Let m be the first move of one of mating sequences of $\mathbf{P}^{pri(\mathbf{Q})}$, and let Ω be the set of checking moves in $\Omega_{m\delta}$ for $\mathbf{Q}$. If Ω is empty, then $\mathbf{P} \not\geq_f /\mathbf{Q}$. Otherwise, let $\mathbf{P}'$ and $\mathbf{Q}'$ be the positions obtained from $\mathbf{P}$ and $\mathbf{Q}$ by a move $m' \in \Omega$. By definition, $\mathbf{P} \geq_f \mathbf{Q}$ if and only if there exists $m' \in \Omega$ such that $\mathbf{P}' \geq_f \mathbf{Q}'$.

We now define muda-ai as follows.

Definition. Let $\mathbf{P}$ be a given position in which an interposed piece can forestall a check. Let $\mathbf{Q}$ be a position obtained by capturing the interposed piece immediately, checking, and eliminating the captured piece from the mating player's hand. The interposed piece is muda if and only if $\mathbf{P} \geq_f \mathbf{Q}$.

Under this definition, an interposition at 3c in the preceding example is muda, since $\mathbf{P} \geq_f \mathbf{Q}$ holds, where the difference δ is based on the positions of the PR, and we assume that f(4c–?, δ) contains 3c–?.

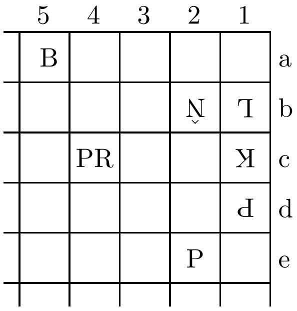
The relation $\geq_f$ depends on the function f. The safety of our method is guaranteed for any function f, since the following theorem holds independently of this choice.

Theorem. *Suppose* $\mathbf{P} \geq_f \mathbf{Q}$. *Then we can mate in* $\mathbf{Q}$, *and there exists a mating sequence in* $\mathbf{P}^{pri(\mathbf{Q})}$ *whose length is greater than or equal to the number of moves to mate the King of* $\mathbf{Q}$.

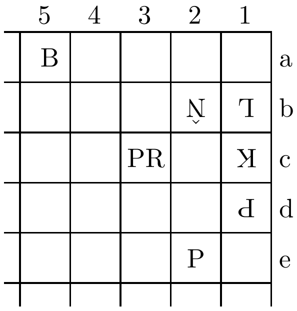
Corollary. *An interposed piece that is muda under our definition is muda under Kakinoki's definition.*

5. Examples

Another example of determining muda-ai. An interposed piece at 3c in the next diagram is non-muda, that is, $\mathbf{P} \not\geq_f \mathbf{Q}$. We can see this as follows. The recorded response to the opponent's move P*2c in **P** is 5a–2d. But in **Q**, if the opponent drops a Pawn at 2c, the Bishop cannot be moved to 2d, since the Promoted Rook is in the way.

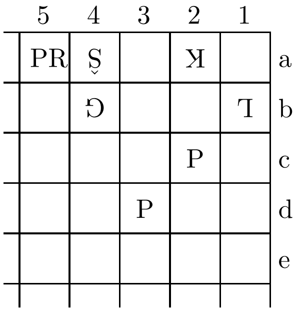
$\mathbf{P} =$ 

no piece in hand

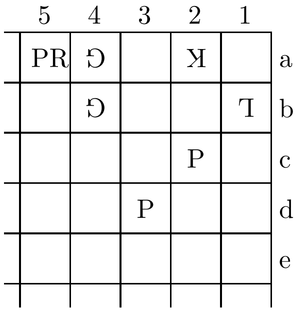
$\mathbf{Q} =$ 

no piece in hand

An example of a one-piece difference. The next diagram is an example of $\mathbf{P} \geq_f \mathbf{Q}$ and $\mathbf{Q} \not\geq_f \mathbf{P}$. We shall show that $\mathbf{Q} \not\geq_f \mathbf{P}$.

$\mathbf{P} =$ 

no piece in hand

$\mathbf{Q} =$ 

no piece in hand

(i) The recorded next move of **Q** is 5a–4a. Therefore, we move the PR on **P**.
(ii) In the resulting position, we assume that the opponent captures this PR with his Gold.
(iii) We look at the recorded next move for the moves 5a–4a and 4b–4a in **Q**. It is G*2b.
(iv) We assume that f(G*2b, δ) contains S*2b here.

(v) We cannot find the opponent's next move 2a-3b in the mating sequence of $\mathbf{Q}^{\mathrm{pri}(\mathbf{P})}$.

6. Final Remarks

The proposed best-first search algorithm was implemented by the author. It performs very well. For example, it is said that today's human experts, regardless of the time spent, can only solve about 80% of the problems posed in the well-known "Shogi Zuko" [Kanju 1755], written in the Edo period. This program, however, can solve 85% of them in a short time; other programs can solve at most 70%. The very famous last problem, "Kotobuki" (happiness), is illustrated below. It requires 611 plies, the longest mating sequence of any Tsume-shogi from the Edo period. The author's program solved it in about 8 minutes on an HP9000 workstation (model735/99MHz). Details are given in [Ito et al. 1995].

9	8	7	6	5	4	3	2	1	
G				P				PR	a
			L	P	G			P	b
	Ŝ					P	P	P	c
	L		B						d
L			B		K	P	L		e
			PP				PR	PP	f
			PP			PS		Ŝ	g
PN		PP		G	Ŝ				h
	G	PP			P		L		i

four pawns in hand

Acknowledgement

The author wishes to thank Professor Kohei Noshita, Dr. Kiyoshi Shirayanagi, Dr. Kenji Koyama, and all the members of the CSA (Computer Shogi Association) for their helpful suggestions.

Note Added in Proof

After this paper had been written, Masahiro Seo pubished new results of his research. His Tsume-shogi program can solve 99% of problems in [Kanju 1755]. See [Seo 1995] for details.

References

[Adachi et al. 1987] H. Adachi, H. Kamekawa and S. Iwata, "Shogi on $n \times n$ board is complete in exponential time" (in Japanese), *Trans. IEICE* **J70-D**, 1843–1852.

[Fairbairn 1989] J. Fairbairn, *Shogi for Beginners*, Ishi Press, Tokyo, 1989.

[Ito and Noshita 1994] T. Ito, K. Noshita, "Two fast programs for solving Tsume-shogi and their evaluation" (in Japanese), *J. Information Processing Soc. Japan*, **35** (1994), 1531–1539.

[Ito et al. 1995] T. Ito, Y. Kawano, K. Noshita, "On the algorithms for solving Tsume-shogi with extremely long solution-steps" (in Japanese), *J. Information Processing Soc. Japan* **36** (1995), 2793–2799.

[Kanju 1755] Kanju Ito, "Shogi Zuko", 1755. Included in Yoshio Kadowaki, *Tsumuya Tsumazaruya* (in Japanese), Toyo bunko 282, Heibon-sha, 1975.

[Kotani et al. 1990] Y. Kotani, T. Yoshikawa, Y. Kakinoki, K. Morita, "Computer Shogi" (in Japanese), Saiensu-sha, 1990.

[Leggett 1966] T. Leggett, *Shogi: Japan's game of strategy*, C. E. Tuttle, Rutland, VT, 1966.

[Matsubara 1993] H. Matsubara, "Shogi (Japanese chess) as the AI research target next to chess", Technical Report ETL-TR-93-23, Electrotechnical Laboratory (Ministry of International Trade and Industry), Tsukuba, Japan, 1993.

[Seo 1995] M. Seo, "Application of conspiracy numbers to a Tsume-shogi program" (in Japanese), *Proceedings of Game Programming Workshop*, Japan 1995, pp. 128–137.

[Wilkes 1950] C. F. Wilkes, *Japanese chess: Shogi* (second edition), Yamato Far Eastern Enterprises, Berkeley, CA, 1950.

Yasuhito Kawano
NTT Communication Science Laboratories
Hikaridai 2-2, Seika-cho, Soraku-gun Kyoto, Japan
kawano@progn.kecl.ntt.jp