

# An Extension of the H-Search Algorithm for Artificial Hex Players

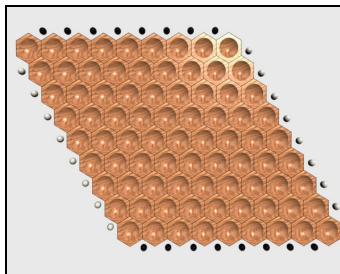
Rune Rasmussen and Frederic Maire

School of SEDC (Information Technology Faculty)  
Queensland University of Technology  
P.O. Box 2434, Brisbane QLD 4001, Australia.  
[r.rasmussen@student.qut.edu.au](mailto:r.rasmussen@student.qut.edu.au), [f.maire@qut.edu.au](mailto:f.maire@qut.edu.au)

**Abstract.** Hex is a classic board game invented in the middle of the twentieth century by Piet Hein and rediscovered later by John Nash. The best Hex artificial players analyse the board positions by deducing complex virtual connections from elementary connections using the H-Search algorithm. In this paper, we extend the H-search with a new deduction rule. This new deduction rule is capable of discovering virtual connections that the H-search cannot prove. Thanks to this new deduction rule, the horizon of the artificial Hex players should move further away.

## 1. Introduction

The Danish engineer and poet Piet Hein invented the game of Hex and presented it at the Niels Bohr Institute of Theoretical Physics in 1942. The game of Hex is a strategic two-player game on a rhombic board (see Fig. 1) The aim of the game for the player Black is to connect the North and South sides of the board with an unbroken chain of black stones. Similarly, the aim of player White is to connect the East and West sides with an unbroken chain of white stones. The first player to make such a connection wins the game. Players can never tie in a game of Hex [1]. The standard size for a Hex board is 11x11 but some players prefer to play on larger boards (like 19x19).



**Fig. 1** A 9x9 board. The player Black tries to connect the North and South edges of the board. The player White tries to connect the East and West edges of the board.

In 1948, John F. Nash rediscovered the game of Hex and presented it at Princeton University. Hex is solvable in polynomial space but is NP-hard [2]. One reason why creating an artificial player for Hex is difficult is that the branching factor of the game tree for Hex is large (much larger than Chess). In fact, it has the same branching factor as Go. In 1949, John Nash proved the existence of a winning strategy for the opening player. However, this proof does not provide a winning strategy.

In 1953, Claude Shannon and E. F. Moore of the Bell Telephone Laboratories devised the first artificial Hex player. The Shannon and Moore's Hex player considers a Hex board as an electric resistor network with a potential on the player's sides. A move by the player makes short circuit connections in the network and a move by the opponent makes open circuits in the network. Shannon and Moore used the resistance of the network as an evaluation function to evaluate board positions [3]. In other words, the connecting player tries to minimise and the opponent tries to maximise the resistance of the network.

A more recent evaluation function for the game of Hex is the Queen-Bee distance [4, 5]. The Queen-Bee distance measures the degree of connection a cell has to an edge of the board by estimating how many moves the connecting player needs to secure the connection. The Queen-Bee distance assumes that the opponent will try to prevent the connection by playing the best available cell for the connecting player. This assumption leads to a recursive definition of the Queen-Bee distance. In particular, if a cell  $x$  is empty, the Queen-Bee distance of  $x$  is equal to the  $1+2^{\text{nd}}$  best Queen-Bee distance among all the neighbours  $y$  of  $x$ . Evaluating the degree of a connection between groups of stones is a key concept in Hex. A *virtual connection* between two groups of stones is an unavoidable connection.

The H-Search algorithm deduces complex virtual connections from elementary connections. Vadim Anshelevich is the pioneer of this approach. His player, "Hexy", uses the H-Search algorithm and was the winner of the 2000 Computer Olympiads for the game of Hex [6, 7]. The strength of "Hexy" is its ability to discover virtual connections. Another artificial player that makes use of the H-Search algorithm is "Six". "Six" was the winner of the 2003 Computer Olympiads for the game of Hex [8].

In this paper, we extend the H-Search algorithm with a new deduction rule. This deduction rule can find virtual connections that the H-Search algorithm fails to find. Our deduction rule is a process that mimics the way humans play Hex. Section 2 describes the H-Search algorithm and its deduction rules. Section 3 presents our new deduction rule. Section 4 examines an edge connection template that our new deduction rule can solve, but that is beyond the deductive capability of the standard H-Search.

## 2. The H-Search Algorithm

The H-Search algorithm deduces virtual connections. However, there are trivial connections that do not require deduction. If two stones of the same colour sit on adjacent cells, then these stones are *virtually connected*. By definition, A *group* is a single empty cell or is a mono-coloured connected component (all the stones are of the same colour) [6, 7]. A group provides a waypoint for making connections. By convention, the four sides of the board constitute four distinct groups where the colour of the side is the colour of its group. A player wins when two side-groups become connected.

The H-Search algorithm solves sub-games. A sub-game is the restriction of the game to a region of the board. In this region, one player, the *connecting player*, tries to connect two distinguished groups while the other player tries to prevent that connection.

Formally, a *sub-game* is a triple  $(x, C, y)$  where  $x$  and  $y$  are disjoint groups and  $C$  is a set of cells such that  $x \cap C = y \cap C = x \cap y = \emptyset$ . If  $x$  or  $y$  are groups of stones then the stones have the connecting player's colour. The groups  $x$  and  $y$  are called the *targets* and the set  $C$  is called the *carrier*. The H-Search algorithm involves two types of connections. A sub-game is a virtual connection or a *strong sub-game* if the connecting player can win the sub-game even when playing second. In addition, a sub-game is a *weak sub-game* if the connecting player can win when playing first. The H-Search algorithm involves two deduction rules, the AND rule and the OR rule.

### Theorem 1: The AND Deduction Rule

Let sub-games  $(x, A, u)$  and  $(u, B, y)$  be strong sub-games with a common target  $u$  and targets  $x \cap y = \emptyset$ . In addition assume  $\{x, u, y\} \cap (A \cup B) = \emptyset$  and  $A \cap B = \emptyset$ . If  $u$  is a group of stones then the sub-game  $(x, A \cup B, y)$  is a strong sub-game. If  $u$  is an empty cell then the sub-game  $(x, A \cup u \cup B, y)$  is a weak sub-game.

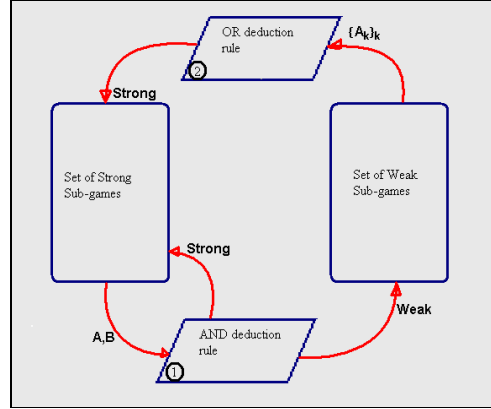
### Theorem 2: The OR Deduction Rule

Let  $(x, A_k, y)$  be of weak sub-games for  $1 \leq k \leq m$  with common targets  $x$  and  $y$ . If

$\bigcap_{k=1}^m A_k = \emptyset$  then the sub-game  $(x, A, y)$  where  $A = \bigcup_{k=1}^m A_k$  is a strong sub-game.

The H-Search algorithm derives complex connections from elementary connections. The H-Search algorithm applies the AND deduction rule first and applies it to pairs of strong sub-games. In Fig. 2, 'A' and 'B' denote strong sub-games. When the H-Search algorithm deduces a new strong sub-game, it appends it to the strong sub-game list and applies the AND deduction on another pair of strong sub-games. When the H-Search algorithm deduces a weak sub-game, it appends it to the weak sub-game list and applies the OR deduction rule on the subset of weak sub-games with the same targets. In Fig. 2,  $\{A_k\}_k$  denotes that set. At the end of one OR deduction the H-

Search performs AND deduction. The algorithm terminates when it exhausts the sub-game sets.



**Fig. 2.** The H-Search applies the AND rule first. Whenever, H-Search deduces a weak sub-game, it immediately applies the OR deduction.

### 2.1. Sub-Game Decomposition

The sub-game sets provide a connection database for the player. The player may use this database to enhance its evaluation function. A player can also use the hierarchical AND-OR decomposition of a connection into more elementary connections to select the sequence of right moves to secure a virtual connection. Sub-game decomposition provides a policy for making connections. A *policy* is a mapping from state (a board position) to action (a move) [9]. A *tactic* is a policy for securing a connection between two targets in a sub-game [10, 11]. For example, suppose that a strong sub-game was derived by applying the AND deduction rule on the strong sub-games  $(x, A, u)$  and  $(u, B, y)$ . If the opponent plays in the carrier  $A$ , then the connecting player will query the sub-game  $(x, A, u)$  for a reply move that secures the sub-game  $(x, A, u)$ . Similarly, suppose that a strong sub-game was derived by applying the OR deduction rule on the weak sub-games  $(x, A_k, y)$  for  $1 \leq k \leq m$ . If the opponent plays at  $z \in \bigcup_{k=1}^m A_k$ , then the connecting player will find a sub-game with carrier  $A_i$ , such that  $z \notin A_i$  and secure the weak sub-game  $(x, A_i, y)$ . That is the connecting player transforms  $(x, A_i, y)$  into a strong sub-game after querying  $(x, A_i, y)$ .

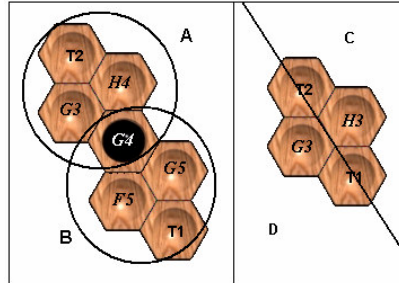


Fig. 3. The AND rule deduced the left sub-game and the OR rule deduces the right sub-game.

In Fig. 3, the AND rule is the deducing rule for the left strong sub-game and the OR rule is the deducing rule for the right sub-game. The tactic for the left sub-game is the aggregation of the tactic of strong connection ‘A’ and the tactic of strong connection ‘B’. The tactic for the right strong sub-game is the combination of the tactic of the weak connection ‘C’ and the tactic of the weak connection ‘D’. If the opponent moves on G3, the connecting player moves on H3 (and reciprocally).

### 3. Extension of the OR Deduction Rule

In Theorem 2, when the intersection of the weak sub-games is empty then the OR deduction rule deduces a strong sub-game. If the OR deduction rule fails, can some other deduction rules prove these failed OR connections? The new deduction rule that follows can prove some connections obtained from failed OR deductions. In Theorem 2, if the intersection of weak sub-games is not empty then the OR deduction fails. Hayward refers to this non-empty intersection as the *must-play (MP) region* [12]. In Fig. 4, the cell “MP” is the must-play region. If the opponent moves outside of the must-play region, then the connecting player will be able to transform the weak sub-game into a strong sub-game.

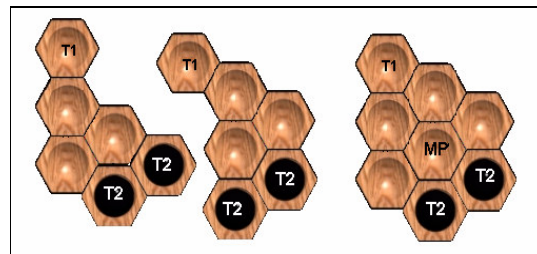


Fig. 4. Two weak sub-games between T1 and T2. The cell MP is the intersection of the carriers of the two weak connections and is the “must-play” region that prevents the connection.

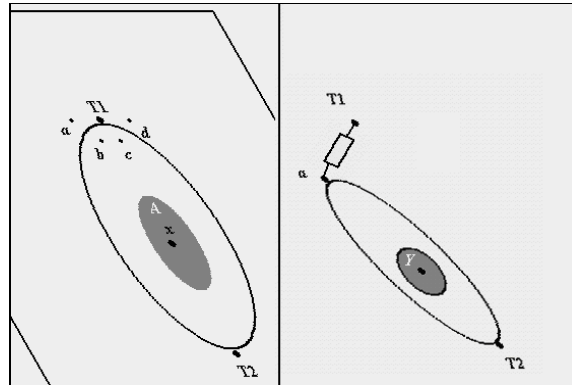
The new deduction rule that follows can deduce failed OR deductions. This deduction rule must generate the sub-game tactics explicitly as its search is more complex than the H-search. The new deduction rule deduces a sub-game and a tactic at the same

time. Here, a tactic is a decision tree where the opponent's moves are stored on the edges of the tree, and the connecting player's reply-moves are stored at the vertices of the trees. We refer to this decision tree as a *tactical decision tree*.

### 3.1. The Must-Play (MP) Deduction Rule

The Must-Play (MP) deduction rule is a guided search of the game tree of a failed OR deduction sub-game. We can assume that the opponent picks his moves in the MP region to block the connecting player; otherwise, the connecting player would immediately secure the connection. The connecting player moves to extend on the targets' groups. This deduction rule is a post H-Search rule as it relies on weak and strong connections to deduce new connections.

In Fig. 5 left, T1 and T2 are the targets of a weak sub-game where the set  $A$  is the must-play region. The cell  $x$  is a cell in the must-play region where the opponent could move and the set  $\{a, b, c, d, \dots\}$  are candidate moves for connecting player. These candidate moves strongly connect with target T1 and at worst weakly connect with target T2. A similar set of moves for the player lie near target T2. For the sake of simplicity, we only consider the set that lie near target T1.



**Fig. 5.** The MP deduction rule is a guided search of the game tree where the opponent's moves are in must play regions. The connecting player's moves build a strongly connected component on the targets that at worst weakly connect to the other target.

If the player extends target T1 with a move on cell 'a' and this move strongly connects with target T2 then the MP deduction rule has found a winning sequence of moves. The rule inserts the sequence into the tactical decision tree. In Fig. 5 right, if the move on cell 'a' weakly connects with target T2, then there is a must-play region  $Y$  between extended target T1 and target T2. The MP deduction rule reiterates the search for a move by the opponent in this new must-play region. If the MP deduction rule finds a solution for every possible move by the opponent in the must-play regions then there is a strong sub-game for targets T1 and T2.

The pseudo code in Fig. 10 is a recursive search that captures the duality between the opponent's moves in the must-play regions and the player's reinforcing moves near the targets. The search depth sets the maximum number of null deductions (see pseudo code). Unlike traditional search techniques, the search depth does not necessarily give better results. Setting the search depth too deep adds redundancy to the final sub-game and tactic. The search depth is a parameter that has to be fine-tuned for use in a competitive player.

#### 4. Results

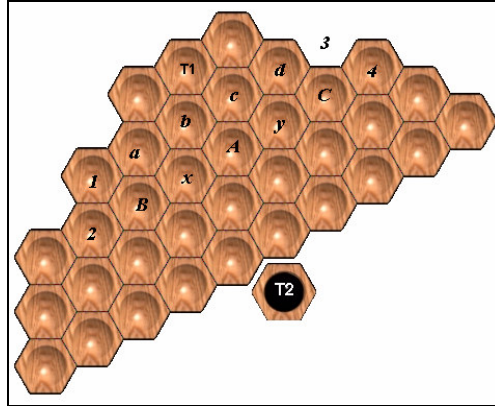
In this section, we show that our new deduction rule can deduce an edge template that the H-Search algorithm cannot deduce. The edge template of interest is the strong connection between a cell on the fifth row and a side (see Fig. 6). This edge template is one of the many template that can be found in the book "Hex Strategy: Making the Right Connections" by Cameron Browne [10].

**Theorem 3:** The H-Search Algorithm cannot deduce the edge template of Fig. 6.

**Proof:** Assume, on an empty board, the fifth row sub-game in Fig. 6 is H-Search deducible. Either the top rule is the AND rule or the top rule is the OR rule. Assume first that the AND rule is the top deducing rule. From Theorem 1, the AND rule deduces a strong sub-game when the common cell is a stone group. However, this is impossible here as the board is empty. Therefore, the AND rule is not the deducing rule.

The OR deduction rule must therefore be the deducing rule. From Theorem 2, the intersection of all weak sub-games with targets T1 and T2 must be empty. However, we will exhibit a move for the opponent that prevents the connecting player from making any weak connection.

The fatal opponent move is on the cell labelled 'A'. By Theorem 2, the player must have a weak sub-game with targets T1 and T2. Since the AND rule is the only rule that can deduce a weak sub-game the player can move on a cell that strongly connects to T1 and T2. Of all of the cells that strongly connect to T1, only those with the labels 'a', 'b', 'c' and 'd' could also strongly connect with target T2.



**Fig. 6.** An edge template; a strong connection is between T1 and T2 (T2 is a side).

Assume one of these cells also strongly connects with T2. Let that cell be either 'a' or 'b'. By Theorem 2, the intersection of weak sub-games between either 'a' or 'b' and the target T2 is empty. If the opponent moves on the cell with the label 'B' then either 'a' or 'b' is weakly connected to T2. However, neither 'a' or 'b' is weakly connected to T2 via the cells 'a', '1', '2' or the cells 'b', 'c', 'y'. Therefore, the cells 'a' and 'b' must weakly connect to T2 via the cell 'x'. However, the cells 'B' and 'A' have the opponent's stones such that 'x' is weakly connected to T2. Therefore, 'a' and 'b' neither weakly nor strongly connect with T2 via 'x', however, this contradicts the assumption that the player's move on 'a' or 'b' strongly connects T1 and T2 because Theorem 2 only deals with weak sub-games and there are none.

Let the cell be either 'c' or 'd' that strongly connect with T2. By Theorem 2, the intersection of weak sub-games between either 'c' or 'd' and the target T2 is empty. If the opponent's move on cell 'C' then either 'c' or 'd' weakly connects to T2. However, neither 'c' or 'd' weakly connects to T2 via the cells 'd', '3', '4' or the cells 'c', 'b', 'x'. Therefore, the cells 'c' and 'd' must weakly connect to T2 via the cell 'y'. However, the cells 'A' and 'C' have the opponent's stones such that 'y' is weakly connected to T2. Therefore, 'c' and 'd' neither weakly nor strongly connect with T2 via 'y', however, this contradicts the assumption that the player's move on 'c' or 'd' strongly connects T1 and T2 because Theorem 2 only deal with weak sub-games and there are none.

Since no move by the player on cells 'a', 'b', 'c' or 'd' strongly connects to T2 given the opponent's first move was on the cell 'A' and these cells are the only possible candidates, the assumption that the OR deduction rule did deduce this fifth-row sub-game is a contradiction. Therefore, the H-Search algorithm cannot deduce a fifth row sub-games on an empty board. ■

**Theorem 4:** The MP deduction rule can deduce the edge template of Fig. 6.

One way to prove this theorem would be to display the tactical tree returned by our program. However, a printout of such a tree would exceed the page limit for this

paper (The full proof tree is available via e-mail request and an extract can be found in the appendix). Our proof is a computer program that demonstrates the deductive property of the MP deduction rule. This computer program is a test-bed for the MP deduction rule. It provides a view of the proof trees and a window where the user can make moves in a sub-game and the computer returns a response from the sub-game tactic. In Fig. 8, a screen shot of that computer program places emphasis on strong connections with the side of the board. In addition, we present an argument using the MP deduction rule that proves a key solution path for the sub-game in Fig. 6.

**Proof:** In Fig. 7, if the opponent makes move ‘1’ in the must-play region then the connecting player can make move ‘2’ and strongly extend target T1. The MP deduction rule continues a search path that has the opponent’s moves in must-play regions as ‘1’, ‘3’, ‘5’ and ‘7’ and the player’s moves as ‘2’, ‘4’, ‘6’ and ‘8’. The moves ‘2’, ‘4’ and ‘6’ by the player strongly extend the target T1 and move eight strongly extends the target T2. In addition, the cells with the label ‘a’ form a weak carrier between the T1 and T2 extensions and the cell with the label ‘b’ forms a separate weak carrier between the T1 and T2 extensions. Since, these two carriers are disjoint, by Theorem 1 T1 and T2 strongly connect.

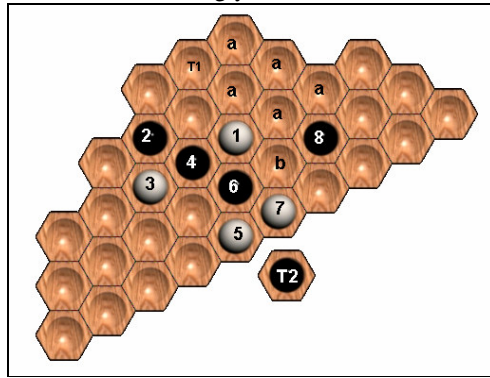


Fig. 7. A solution path discovered by the MP deduction rule.

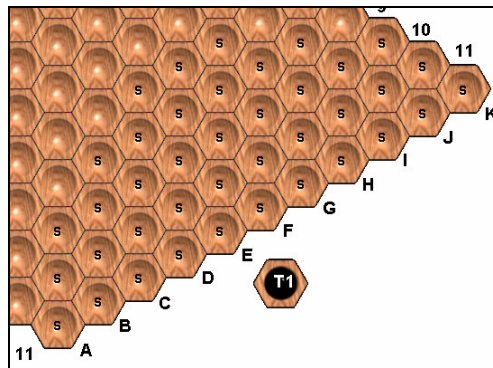


Fig. 8. A test run of the MP deduction rule implementation gives the cells with label ‘S’ as strong connections with the side target T1. There are three fifth-row strong connections.

## 5. Conclusion

Hex is the game that kick-started the connection game genre in the middle of the twentieth century. Despite its simple rules, Hex is a subtle strategic game that can provide a test-bed for the development of general principles of combining tactics into strategies. In this paper, we describe an extension of the OR deduction rule that allows the automatic discovery of new connection. The MP deduction rule is an efficient search of the game tree of a failed OR deductions. Indeed, to prove that a sub-game is strong, this rule only considers the relevant candidate moves of the opponent. This rule dramatically prunes the game tree of the connection by exploiting the MP region. For example, since the carrier of the edge-template of Fig. 6 has 35 empty cells, its full game tree is of depth 35 with 35-factorial different possible games. The MP deduction rule in this case is equivalent to looking ahead 35 moves in advance. The MP deduction rule is efficient because it also uses heuristics to generate a set of candidate moves for the connecting player. The MP deduction rule could be made complete by testing all possible reply-moves of the connecting player, but at a computational cost that would not be a good trade-off for an artificial player, as testing many failed OR-deductions is more worthwhile in competitive conditions.

## References

- [1] M. Gardener, "The Game of Hex," in *The Scientific American Book of Mathematical Puzzles and Diversions*. New York: Simon and Schuster, 1959.
- [2] S. Even, Tarjan R.E., "A Combinatorial Problem Which is Complete in Polynomial Space," *Journal of the Association for Computing Machinery*, vol. 23, pp. 710-719, 1976.
- [3] W. J. Duffin, "Electricity and Magnetism," 4th ed. London: McGraw-Hill, 1990, pp. 46-81.
- [4] J. Van Rijswijck, "Are Bees Better Than Fruitflies?," in *Department of Computing Science*. Edmonton: University of Alberta, 2002.
- [5] J. Van Rijswijck, "Search and Evaluation in Hex," in *Department of Computing Science*. Edmonton: University of Alberta, 2002.
- [6] V. V. Anshelevich, "A Hierarchical Approach to Computer Hex," *Artificial Intelligence*, vol. 134, pp. 101-120, 2002.
- [7] V. V. Anshelevich, "An Automatic Theorem Proving Approach to Game Programming," presented at Proceedings of the Seventh National Conference of Artificial Intelligence, Menlo Park, California, 2000.
- [8] G. Melis, Hayward, R., "Hex Gold at Graz: Six Defeats Mongoose," *to appear ICGA Journal*, 2003.
- [9] S. Russell, Norvig, P., *Artificial Intelligence a Modern Approach*, Second ed. Upper Saddle River: Pearson Education, 2003.
- [10] C. Browne, *Hex Strategy: Making the Right Connections*. Natick: A. K. Peters, 2000.
- [11] C. Browne, "Connection Games," A. K. Peters, 2004.
- [12] R. Hayward, Björnsson, Y., Johanson, M., Kan M., Po, N., Van Rijswijck, J., *Advances in Computer Games: Solving 7x7 HEX: Virtual Connections and Game-State Reduction*, vol. 263. Boston: Kluwer Academic Publishers, 2003.

## Appendix



**Fig. 9** A screen shot of the part of the tree derived by the MP deduction rule for the edge template of Fig. 6. The target T1 is at position I7 (see Fig. 8 for the coordinates). The depth of the tree is 11 and the number of nodes of 1008. The “Red’s adjacent stones” leaves correspond to adjacent cells (elementary strong connections).

```

function MP_Deduction(targetA, targetB, Depth) returns a SubGame, Tactic

Inputs:
targetA: The first target
targetB: The second target
Depth: The deepest level in the search

if not hasWeak(targetA, targetB) or hasStrong(targetA, targetB) then
return null

Must_Play := GetMustPlayRegion(targetA, targetB)

Played_Moves := { targetA, targetB }
Result := null
for cell  $\in$  Must_Play do
  Played_Moves := Played_Moves  $\cup$  {cell}
  Result := Search_On_Player(Stack_A, Stack_B, Played_Moves, Depth, 1)
  Played_Moves := Played_Moves - {cell}
  if Result  $\neq$  null then
    TheSubGame := BuildSubGame(Result, targetA, targetB)
    TheTactic := BuildTactic(Result, targetA, targetB)
    return TheSubGame, TheTactic

end // for
return null

function Search_On_Player(StackA, StackB, Played, Depth, Level) returns a Tactic

Inputs:
StackA: The stack of strong connections that extends the first target group
StackB: The stack of strong connections that extends the second target group
Played: The set of moves that the search has made
Depth: The depth of the search
Level: The current level of the search

if Level > Depth then
return null

if Exists_A_Cell_That_Strongly_Connects(StackA, StackB) then
return BuildStrongConnectionTactic(StackA, StackB)

for stack  $\in$  { StackA, StackB} do
  for cell := Strongly_Connected_AND_Deducible_With(stack) do
    StrongTactics := Get_StrongTactics(cell, stack)
    for game  $\in$  StrongTactics do
      if IsEmpty({cell  $\cup$  game}  $\cap$  Played_Moves) then
        Push(stack, game)
        Played_Moves := Played_Moves  $\cup$  {cell}
        Result := Search_On_Opponent(StackA, StackB, Played, Depth, Level + 1)
        Played_Moves := Played_Moves - {cell}
        Pop(stack)
        if Result not null then
          return Result
      end // for
    end // for
  end // for

function Search_On_Opponent(StackA, StackB, Played, Depth, Level) returns a Tactic

Inputs:
StackA: The stack of strong connections that extends the first target group
StackB: The stack of strong connections that extends the second target group
Played: The set of moves that search has made
Depth: The depth of the search
Level: The current level of the search

TheConnectionTactic := Create_ConnectionTactic()
Must_Play := Get_Must_Play_Region_Between_Stacks(StackA, StackB)
Result := null
for cell  $\in$  Must_Play do
  Played_Moves := Played_Moves  $\cup$  {cell}
  Result := Search_On_Player(Stack_A, Stack_B, Played_Moves, Depth, Level + 1)
  Played_Moves := Played_Moves - {cell}
  if Result = null then
    return null
  end // for
  Add_Transition_To(TheConnectionTactic, Result)
end // for

```

Fig. 10. Pseudo code for the MP deduction rule.