A Reverse Hex Solver

Kenny Young* and Ryan B. Hayward

Dept. of Computing Science, UAlberta, Canada, hayward@ualberta.ca,
`http://webdocs.cs.ualberta.ca/~hayward/`

Abstract. We present Solrex, an automated solver for the game of Reverse Hex. Reverse Hex, also known as Rex, or Misère Hex, is the variant of the game of Hex in which the player who joins her two sides loses the game. Solrex performs a mini-max search of the state space using Scalable Parallel Depth First Proof Number Search, enhanced by the pruning of inferior moves and the early detection of certain winning strategies.
Solrex is implemented on the same code base as the Hex program Solver, and can solve arbitrary positions on board sizes up to 6×6, with the hardest position taking less than four hours on four threads.

Keywords: Hex, Reverse Hex, misère, Rex, solver, combinatorial game theory, proof number search

1 Introduction

In 1942 Piet Hein invented the two-player board game now called Hex [10]. The board is covered with a four-sided array of hexagonal cells. Each player is assigned two opposite sides of the board. Players move in alternating turns. For each turn, a player places one of their stones on an empty cell. Whoever connects their two sides with a path of their stones is the winner.

In his 1957 *Scientific American* Mathematical Games column, *Concerning the game of Hex, which may be played on the tiles of the bathroom floor*, Martin Gardner mentions the misère version of Hex known as Reverse Hex, or Rex, or Misére Hex: whoever joins their two sides *loses* [5]. See Figure 1.

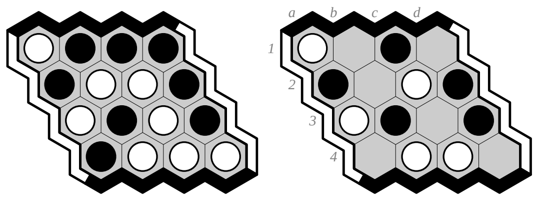


Fig. 1. Left: the end of a Rex game. White has joined both White sides, so loses. Right: a Rex puzzle by Ronald J. Evans. White to play and win.[7]

So, for positive integers n, who wins Rex on $n \times n$ boards? Using a strategy-stealing argument, Robert O. Winder showed that the first (resp. second) wins

* The authors gratefully acknowledge the support of NSERC.

when n is even (odd) [5]. Lagarias and Sleator further showed that, for all n, each player has a strategy that can avoid defeat until the board is completely covered [13].

Which opening (i.e. first) moves wins? Ronald J. Evans showed that for n even, opening in an acute corner wins [4]. Hayward et al. further showed that, for n even and at least 4, opening in a cell that touches an acute corner cell and one's own side also wins [9].

The results mentioned so far prove the existence of winning strategies. But how hard is it to *find* such strategies? In his 1988 book Gardner commented that "*4×4 [Rex] is so complex that a winning line of play for the first player remains unknown.* [7, 8]. In 2012, based on easily detected pairing strategies, Hayward et al. explained how to find winning strategies for all but one (up to symmetry) opening move on the 4×4 board [9].

In this paper, we present Solrex, an automated Rex solver that solves arbitrary Rex positions on boards up to 6×6. With four threads, solving the hardest 6×6 opening takes under 4 hours; solving all 18 (up to symmetry) 6×6 openings takes about 7 hours.

The design of Solrex is similar to the design of the Hex program Solver. So, Solrex searches the minimax space of gamestates using Scalable Parallel Depth-First Proof Number Search, the enhanced parallel version by Pawlewicz and Hayward [14] of Focussed Depth-First Proof Number Search of Arneson, Hayward, and Henderson [2]. Like Solver, Solrex enhances the search by inferior move pruning and early win detection. The inferior move pruning is based on Rex-specific theorems. The win detection is based on Rex-specific virtual connections based on pairing strategies.

In the next sections we explain pairing strategies, inferior cell analysis, win detection, the details of Solrex, and then present experimental results.

2 Death, pairing, capture, joining

Roughly, a dead cell is a cell that is useless to both players, as it cannot contribute to joining either player's two sides. Dead cells can be pruned from the Rex search tree. Related to dead cells are captured cells, roughly cells that are useless to just one player and so can be colored for the other player. In Hex, each player wants to capture cells; in Rex, each player wants to force the opponent to capture cells. In Rex, such opponent-forced capture can be brought about by pairing strategies. As we will see in a later section, pairing strategies can also be used to force the opponent to join their two sides.

Before elaborating on these ideas, we give some basic terminology. Let $\overline{X}$ denote the opponent of player X.

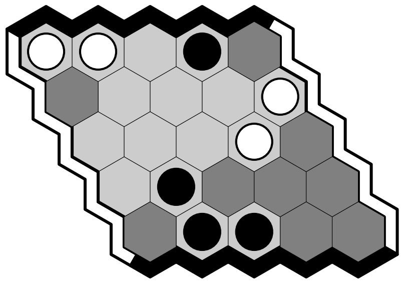


Fig. 2. Shaded cells are dead. All other uncolored cells are live.

For a given position, *player X colors cell c* means that player X moves to cell c, i.e. places a stone of her color on cell c. A cell is *uncolored* if it is unoccupied. To *X-fill* a set of cells is to X-color each cell in the set; to *fill* a set is either to X-fill or $\overline{X}$-fill the set.

A *state* $S = P^X$ is a position P together with the specified player X to move next. The *winner* of S is whoever has a winning strategy from S.

For a position P and a player X, a *X-joinset* is a minimal set of uncolored cells which when X-colored joins X's two sides; a *joinset* is an X-joinset or an $\overline{X}$-joinset; an uncolored cell is *live* if it is in a joinset, otherwise it is *dead*; a colored cell is *dead* if uncoloring it would make it dead.

For an even size subset C of uncolored cells of a position or associated state, a *pairing Π* is a partition of C into *pairs*, i.e. subsets of size two. For a cell c in a pair $\{c, d\}$, cell d is c's *mate*. For a state S, a player Y, and a pairing Π, a *pairing strategy* is a strategy for Y that guarantees that, in each terminal position reachable from S, at most one cell of each pair of Π will be Y-colored.

For a state $S = P^X$, *Last* is that player who plays last if the game ends with all cells colored, and *Notlast* is the other player, i.e. she who plays second-last if the game ends with all cells uncolored. So, Last (Notlast) is whoever plays next if and only if the number of uncolored cells is odd (even). For example, for $S = P^X$ with P the empty 6×6 board, Last is $\overline{X}$ and Notlast is X, since X plays next and P has 36 uncolored cells.

Theorem 1. *For state S and pairing Π, each player has a pairing strategy for S.*

Proof. It suffices to follow these rules. Proving that this is always possible is left to the reader.

First assume Y is Last. In response to $\overline{Y}$ coloring a cell in Π, Y colors the mate. Otherwise, Y colors some uncolored cell not in Π. Next assume Y is Notlast. In response to $\overline{Y}$ coloring a cell in Π with uncolored mate, Y colors the mate; otherwise, Y colors a cell not in Π; otherwise (all uncolored cells are in Π, and each pair of Π has both or neither cell colored), Y colors any uncolored cell of Π. □

For a player X and a pairing Π with cell set C of a position P or associated state $S = P^Y$, we say Π *X-captures* C if X-coloring at least one cell of each pair of Π leaves the remaining uncolored cells of C dead; and we say Π *X-joins* P if X-coloring at least one cell of each pair of Π joins X's two sides.

Notice that every captured set (as defined here, i.e. for Rex) comes from a pairing and so has an even number of cells, as does every X-join set.

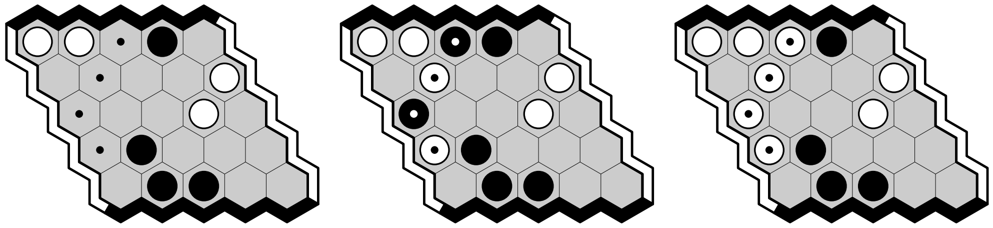


Fig. 3. Left: dots show a White-captured set (the top two cells form one pair, the bottom two form the other). Middle: Each player has colored one cell from each pair, and the two Black cells are dead. Right: original position after filling White-captured cells.

3 Inferior cell pruning and early win detection

We now present the Rex theorems that allow our solver to prune inferior moves and detect wins early.

For a position P, a player X, and a set of cells C, $P + C_X$ is the position obtained from P by X-coloring all cells of C, and $P - C$ is the position obtained from P by uncoloring all colored cells of C. For clarity, we may also write $P - C_X$ in this case where X is the player who originally controlled all the cells of C. Similarly, for a state $S = P^Y$, where $Y = X$ or $\overline{X}$, $S + C_X$ is the state $(P + C_X)^Y$. Also, in this context, when C has only one cell c, we will sometimes write c_X instead of $\{c\}_X$.

For states S and T and player X, we write $S \geq_X T$ if X wins T whenever X wins S, and we write $S \equiv T$ if the winner of S is the winner of T, i.e. if $S \geq_X T$ and $T \geq_X S$ for either player X.

An X-strategy is a strategy for player X.

Theorem 2. *For an even size set C of uncolored cells of a state S, $S \geq_X S + C_X$.*

Proof. Assume π^+ is a winning X-strategy for $S^+ = S + C_X$. Let π be the X-strategy for S obtained from π^+ by moving anywhere in C whenever $\overline{X}$ moves in

C. For any terminal position reachable from S, the set of cells occupied by $\overline{X}$ will be a superset of the cells occupied by $\overline{X}$ in the corresponding position reachable from S^+, so X wins S. □

Theorem 3. *For a position P with uncolored cell c, $(P + c_{\overline{X}})^{\overline{Y}} \geq_X P^Y$.*

Proof. First assume $Y = \overline{X}$. Assume X wins $S = P^{\overline{X}}$. Then, for every possible move from S by $\overline{X}$, X can win. In particular, X can win after $\overline{X}$ colors c. So X wins $(P + c_{\overline{X}})^X$.

Next assume $Y = X$. Assume X wins $S = P^X$. We want to show X wins $S' = (P + c_{\overline{X}})^{\overline{X}}$. Let c' a cell to which $\overline{X}$ moves from S', let $C = \{c, c'\}$, and let S'' be the resulting state $(P + C_{\overline{X}})^X$. X wins S so, by Theorem 2, X wins S''. So, for every possible move from S', X wins. So X wins S'. □

Theorem 4. *For an X-captured set C of a state S, $S + C_X \geq_X S$.*

Proof. Assume $\overline{X}$ wins $S^+ = S + C_X$ with strategy π^+. We want to show that $\overline{X}$ wins S. Let Π be an X-capture pairing for C, and modify π^+ by adding to it the Π pairing strategy for $\overline{X}$.

Let Z be a terminal state reachable from S by following π. Assume by way of contradiction that Z has an $\overline{X}$-colored set of cells joining $\overline{X}$'s two sides. If such a set Q^* exists, then such a set Q exists in which no cell is in C. (On C $\overline{X}$ follows a Π pairing, so in Z at most one cell of each pair of Π is $\overline{X}$-colored. Now X-color any uncolored cells of C. Now at least one cell of each pair is X-colored, and C is X-captured, so each $\overline{X}$-colored cell of C is dead, and these cells can be removed one at a time from Q^* while still leaving a set of cells that joins $\overline{X}$'s two sides. Thus we have our set Q.) But then the corresponding state Z^+ reachable from S^+ by following π^+ has the same set Q, contradicting the fact that $\overline{X}$ wins S^+. □

Corollary 1. *For an X-captured set C of a state S, $S \equiv S + C_X$.*

Proof. By Theorem 4 and Theorem 2. □

Theorem 5. *For a player X and a position P with uncolored dead cell d, $(P + d_X)^{\overline{X}} \geq_X P^X$. A move to a dead cell is at least as good as any other move.*

Proof. Coloring a dead cell is equivalent to opponent-coloring the cell. So this theorem follows by Theorem 3. □

Theorem 6. *For a position P with uncolored cells c, k with c dead in $P + k_X$, $(P + c_X)^{\overline{X}} \geq_X (P + k_X)^{\overline{X}}$. Prefer victim to killer.*

Proof. $(P + k_X)^{\overline{X}} \equiv (P + k_X + c_X)^X \geq_X (P + c_X)^X$. □

Theorem 7. *For a position P with uncolored cells c, k with c dead in $P + k_{\overline{X}}$, $(P + c_X)^{\overline{X}} \geq_X (P + k_X)^{\overline{X}}$. Prefer vulnerable to opponent killer.*

Proof. Assume k is a winning move for X from P^X, i.e. assume X wins $S = (P + k_X)^{\overline{X}}$. Consider any such winning strategy π. We want to show c is also a winning move for X from P^X, i.e. that X wins $S' = (P + c_X)^{\overline{X}}$.

To obtain a winning X-strategy π' for S', modify π by replacing c with k: whenever X (resp. $\overline{X}$) colors c in π, X ($\overline{X}$) colors k in π'. In P, $\overline{X}$-coloring k kills c: so in P, if some X-joinset J contains c, then J must also contain k. But a continuation of π' has both k and c X-colored if and only if the corresponding continuation of π has them both X-colored. So, since X wins S following π, X wins S' following π'. □

Theorem 8. *For a position P with uncolored cell d and set C that is X-captured in $(P + d_x)^{\overline{X}}$, for all $c \in C$, $(P + c_x)^{\overline{X}} \geq_X (P + d_x)^{\overline{X}}$. Prefer capturee to capturer.*

Proof. $(P + c_X)^{\overline{X}} \geq_X (P + C_X + d_X)^{\overline{X}} \equiv (P + d_X)^{\overline{X}}$. □

Our next results concern *mutual fillin*, namely when there are two cells a,b such that X-coloring a $\overline{X}$-captures b and $\overline{X}$-coloring b X-captures a.

Theorem 9. *Let P be a position with sets A, B containing cells a, b respectively, such that A is X-captured in $(P + b_{\overline{X}})$, and B is $\overline{X}$-captured in $(P + a_X)$. Then $P \equiv P + a_X + b_{\overline{X}}$.*

Proof. By if necessary relabelling $\{X, a, A\}$ and $\{\overline{X}, b, B\}$, we can assume X plays next. We claim that a X-dominates each cell in $A + B$. Before proving the claim, observe that it implies the theorem, since after X colors a, all of B is Y-captured, so Y can then color any cell of B, in particular, b.

To prove the claim, consider a strategy that X-captures A in $P + b_{\overline{X}}$. Now, for all α in $A + B$,

$$\begin{aligned}
(P + \alpha_X)^{\overline{X}} &\leq_X & (P + b_{\overline{X}})^{\overline{X}} & \quad \text{(Theorem 3 twice: remove } \alpha_X\text{, add } b_{\overline{X}}) \\
&\equiv & (P + A_X + b_{\overline{X}})^{\overline{X}} & \quad \text{(capture)} \\
&\leq_X & (P + a_X + B_{\overline{X}})^{\overline{X}} & \quad \text{(Theorem 3, repeatedly for } X \text{ and then } \overline{X}) \\
&\leq_X & (P + a_X)^{\overline{X}} & \quad \text{(capture)}
\end{aligned}$$

So the claim holds, and so the theorem. □

Theorem 10. *Let c be any X-colored cell in a position P as described in Theorem 9. Then $(P - c + a_X)^{\bar{X}} \geq_X P^{\bar{X}}$. Prefer filled to mutual fillin creator.*

Proof. Define b' to be the mate of b in the $\overline{X}$-capture strategy for B in $(P + a_X)$.

$$\begin{aligned}
P^{\bar{X}} &\equiv & (P + a_X + b_{\bar{X}})^{\bar{X}} && \text{(Theorem 9)} \\
&\equiv & (P + a_X + B_X - b'_{\bar{X}})^{\bar{X}} && \text{(filling captured cells, now } b' \text{ dead)} \\
&\equiv & (P + a_X + B_X)^X && \text{(coloring } b'\text{)} \\
&\equiv & (P + a_X)^X && \text{(capture)} \\
&\leq_X & (P - c + a_X)^{\bar{X}} && \text{(Theorem 3)}
\end{aligned}$$

□

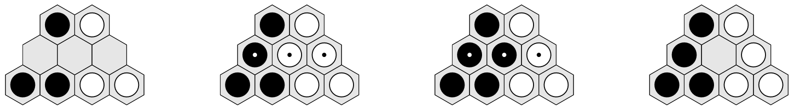


Fig. 4. Mutual fillin. If B colors left cell, the other two cells are W-captured. If W colors right cell, the other two cells are B-captured. So we can replace first position with this.

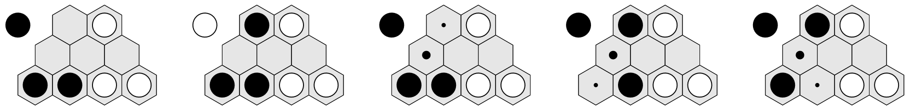


Fig. 5. Mutual fillin domination. Off-board stone shows B to play. Black move would create mutual fillin pattern. So, for these three states, Black prefers large dot to small.

Finally, we mention join pairing strategies.

Theorem 11. *For a state $S = P^X$ with an $\overline{X}$-join pairing Π, X wins P^X.*

Proof. It suffices for X to follow the Π strategy. In each terminal state Z player X will have colored at most one cell of Π. From Z obtain Z' by $\overline{X}$-coloring any uncolored cells: this will not change the winner. But in Z' at least one cell of each pair of Π is $\overline{X}$-colored, and Π is an $\overline{X}$-join pairing. So in Z' $\overline{X}$'s two sides are joined, so in Z $\overline{X}$'s two sides are joined. So X wins. □

4 Early win detection

For a position P, a *X-join-pairing strategy* is a pairing strategy that joins X's two sides, and an *X-pre-join-pairing strategy* is an uncolored cell k together with an

X-join-pairing strategy of $P+k_X$; here k is the *key* of this strategy. The key to our algorithm is to find opponent (pre)-join-pairing strategies. When it is clear from context that the strategies join a player's sides, we call these simply *(pre)-pairing strategies.*

Theorem 12. *Let P be a position with an X-join-pairing strategy. Then $\overline{X}$ wins P^X and also $P^{\overline{X}}$.*

Proof. This follows from Theorem 7 in [9]: $\overline{X}$ can force X to follow the X-join-pairing strategy.

Theorem 13. *Let P be a position with an X-pre-join-pairing strategy and with X = Last. Then $\overline{X}$ wins P^X and also $P^{\overline{X}}$.*

Proof. This follows from Theorem 6 in [9]. $\overline{X}$ can avoid playing the key of the pre-pairing strategy, forcing X to eventually play it.

5 Solrex

Solrex is based on Solhex, the Hex solver of the Benzene code repository [3]. The challenge in developing Solrex was to identify and remove any Hex-specific, or Rex-unnecessary, aspects of Solhex — e.g. permanently inferior cells apply to Hex but not Rex — and then add any Rex-necessary pieces. E.g., it was necessary to replace the methods for finding Hex virtual connections with methods that find Rex (pre-) pairing strategies.

Search follows the Scalable Parallel Depth First variant of Proof Number Search, with the search focusing only on a limited number of children (as ranked by the usual electric resistance model) at one time [14].

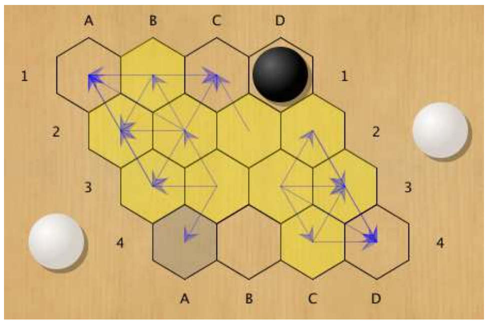


Fig. 6. Inferior cells of a Rex position. Each arrow end is inferior to its arrowhead.

When reaching a leaf node, using a database of fillin and inferior cell patterns, we apply the theorems of §3. We find dead cells by applying local patterns and by searching for any empty cells whose neighbourhood of empty cells, after stone groups have been removed and neighbouring empty cells contracted, is a clique. We iteratively fillin captured cells and even numbers of dead cells until no more fillin patterns are found. We also apply any inferior cell domination that comes from virtual connection decompositions [2, 11].

We then look into the transposition table to see if the resulting state win/loss value is known, either because we previously solved, or because of color symmetry (a state which looks the same for each player is a win for Notlast). Then inferior cells are pruned. Then, using H-search [1] in which the or-rule is limited to combining only 2 semi-connections, we find (pre)-join-pairing strategies. Then, for X the player to move, we prune each key of every X-pre-join-strategy.

H-search is augmented by observing that semi-connections that overlap on a captured set of endpoints do not conflict and so can be combined into a full connection [2, 11]. Notice that augmented H-search is not complete: some pairing strategies (e.g. the mirror pairing strategy for the $n \times (n-1)$ board [6]) cannot be found in this way.

Figure 7 shows the start of Solrex's solution of 1.Bd1, the only unsolved 4×4 opening from [9]. First, inferior cells are found: White b1 captures a1,a2; a2 kills a1; b2 captures a2,a3; c2 leaves c1 dominated by b2; d2 captures d3,d4; etc. See Figure 6. Only 5 White moves remain: a1,c1,a4,b4,d4. After trying 2.Wa4, a White pre-join-pairing strategy is found, so this loses. Similarly, 2.Wb4 and 2.Wd4 also lose. Now 2 White moves remain: a1,a3. From 2.Wa1, search eventually reveals that 3.Bc1 wins (a2 also wins). From 2.Wc1, search reveals that 3.Ba1 wins (b2 and d4 also win). The deepest line in solving this position is 1.Bd1 2.Wc1 3.Bd4 4.Wc4 5.Bb2 6.Wa3 7.Ba4 8.Wb4.

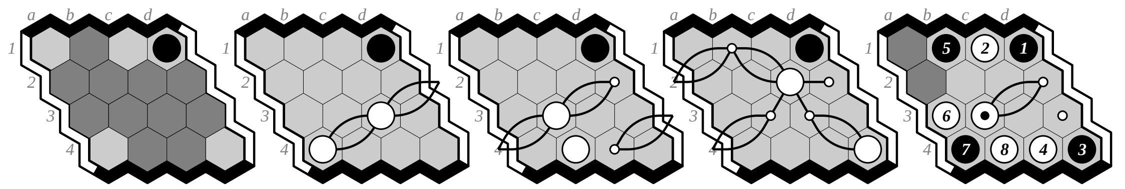


Fig. 7. Solving 1.Bd1. Left: White inferior cells after 1.Bd1. Then White pre-join-pair strategies after 2.Wa4, 2.Wb4, 2.Wd4. Search reveals that 2.Wa1 loses. Search reveals that 2.Wc1 loses. So 1.Bd1 wins. The last diagram shows the deepest line of search and the final pre-join-pair strategy: the shaded cells are Black-captured.

6 Experiments

We ran our experiments on Torrington, a quad-core i7-860 2.8GHz CPU with hyper-threading, so 8 pseudo-cores. For 5×5 Rex, our test suite is all 24 replies to opening in the acute corner:[1] this takes Solrex 13.2s. For 6×6 Rex, our test suite is all 18 (up to symmetry) 1-move opening states: this takes Solrex 25900s. To measure speedup, we also ran the 18 1-move 6×6 openings on a single thread, taking 134635s.

To show the impact of Solrex's various features, we ran a features knockout test on the 5×5 test suite. For features which showed negligible or negative contribution, we ran a further knockout test on the hardest 6×6 position, 1.White[d2], color-symmetric to 1.Black[e3]. The principle variation for this hardest opening is shown in Figure 8. The results are shown below. Figure 9 shows all losing moves after the best opening move on 5×5 (all opening 5×5 moves lose), and all losing opening moves on 6×6.

Figure 10 shows three new Rex puzzles we discovered by using Solrex. The middle puzzle was the only previously unsolved 4×4 position. The other two were found by using Solrex to search for positions with few winning moves.

5×5 knockout tests	
version	time ratio
all features on	1.0 (13.9s)
no dead clique cutset	.97
unaugmented H-search	.99
no mutual fillin	1.00
no color symmetry pruning	1.01
no VC decomp	1.06
no dead fillin	1.07
no resistance move ordering	1.62
no capture fillin	2.02
no inferior pruning	2.30
no H-search	89.83

6×6 knockout test	
version	time ratio
all features on	1.0 (13646 s)
unaugmented H-search	1.10
no color symmetry pruning	1.13
no dead clique cutset	1.37
no mutual fillin	1.44
no VC decomp	1.95

[1] All opening 5×5 Rex moves lose, so we picked all possible replies to the presumably strongest opening move.

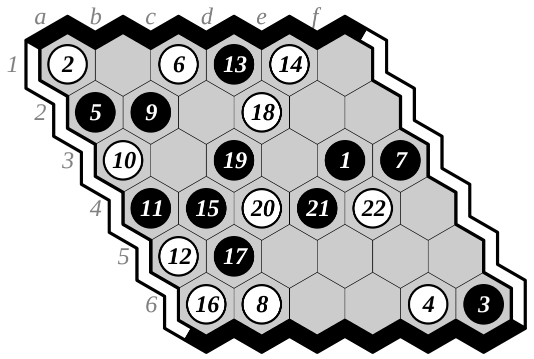


Fig. 8. Principle variation of 1.Black[e3], hardest 6×6 opening. From here Black forces White to connect with pairs {C4,C5} {D6,E5}{F4,F5} and last cell D5.

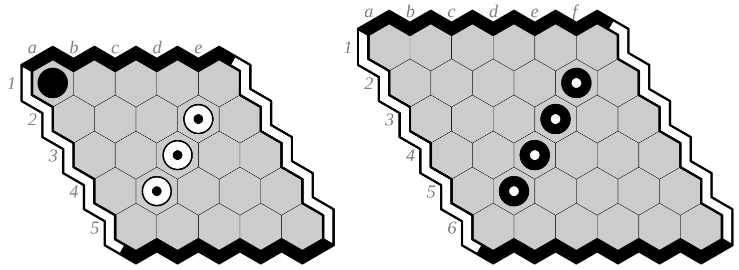


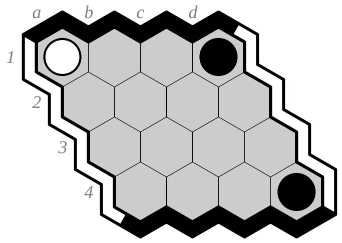
Fig. 9. Left: all losing replies. Right: all losing openings.

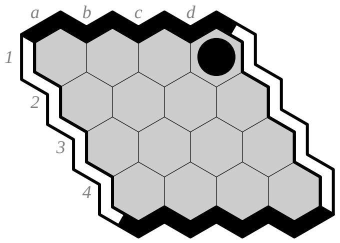
7 Conclusions

All features listed in the knockout tests contributed significantly to shortening search time: the four features that contributed no improvement on 5×5 boards all contributed significantly on 6×6 boards. The effectiveness of these pruning methods – which exploit pruning via local patterns in a search space that grows exponentially with board size — explained by Henderson for Hex, is clearly also valid for Rex [11]:

> *In almost all cases, we see that feature contributions improved with board size. We believe this is partly because the computational complexity of most of our algorithmic improvements is polynomial in the board size, while the corresponding increase in search space pruning grows exponentially. Furthermore, as the average game length increases, more weak moves are no longer immediately losing nor easily detectable via previous methods, and so these features become more likely to save significant search time.*

Of these features, by far the most critical was H-search, which yielded a time ratio of about 90 on 5×5 Rex when omitted. The enormous time savings resulting from H-search is presumably because our general search method does not learn





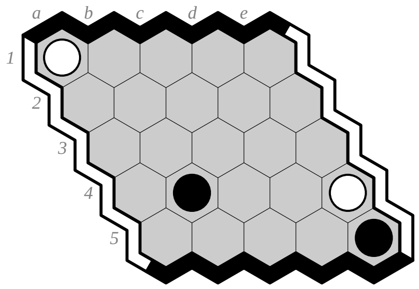


Fig. 10. Three new Rex puzzles. Left: White to play: find the only winning move. Middle: White to play: find White's best move, and Black's best reply. Right: White to play: find the only winning move.

to recognize the redundant transpositions that correspond to the discovery of a (pre-) pairing strategy. So H-search avoids some combinatorial explosion.

Solrex takes about 7 hours to solve all 18 (up to symmetry) 6×6 boardstates; by contrast, Solhex takes only 301 hours to solve all 32 (up to symmetry) 8×8 boardstates [12]. So why is Solhex faster than Solrex?

One reason is because Hex games tend to be shorter than Rex games: in a balanced Rex game, the loser can often force the winner to play until the board is nearly full. Another reason is there are Hex-specific pruning features that do not apply to Rex: for example, the only easily-found virtual connections for Rex that we know of are pairing strategies, and there seem to be far fewer of these than there are easily-found virtual connections in Hex. Also, in Hex, if the opponent can on the next move create more than one winning virtual connection, then the player must make a move which interferes with each such connection or lose the game; we know of no analogous property for Rex.

The general approach of Solhex worked well for Solrex, so this approach might work for other games, for example connection games such as Havannah or Twixt.

Solutions to puzzles. Evans' puzzle: b1 (unique). Three new puzzles: Left: a2 (unique). Middle: Black wins; best move for White is a1, which leaves Black with only 2 winning replies (a2, c1); all other White moves leave Black with at least 3 winning replies (e.g. c1 leaves a1, b2, d4). Right: e3 (unique).

Acknowledgments. We thank Jakub Pawlewicz for helpful comments.

References

1. Vadim V. Anshelevich. The game of Hex: An automatic theorem proving approach to game programming. In *AAAI/IAAI*, pages 189–194, Menlo Park, 2000. AAAI Press / The MIT Press.
2. Broderick Arneson, Ryan B. Hayward, and Philip Henderson. Solving Hex: Beyond humans. In H.J̃aap van den Herik, Hiroyuki Iida, and Aske Plaat, editors, *Computers and Games 2010*, volume 6515 of *LNCS*, pages 1–10. Springer, 2011.

3. Broderick Arneson, Philip Henderson, and Ryan B. Hayward. Benzene, 2009. `http://benzene.sourceforge.net/`.
4. Ronald J. Evans. A winning opening in reverse Hex. *Journal of Recreational Mathematics*, 7(3):189–192, 1974.
5. Martin Gardner. Mathematical games: Concerning the game of hex, which may be played on the tiles of the bathroom floor. *Scientific American*, 197(1):145–150, June 1957.
6. Martin Gardner. Mathematical games: Four mathematical diversions involving concepts of topology. *Scientific American*, 199(4):124–129, October 1958.
7. Martin Gardner. Mathematical games: Games of strategy for two players: star nim, meander, dodgem, and rex. *Scientific American*, 232(6):106–111, June 1975.
8. Martin Gardner. *Hexaflexagons and Other Mathematical Diversions: The First Scientific American Book of Puzzles and Games*, chapter 8, pages 73–83. University of Chicago Press, Chicago, USA, 1988.
9. Ryan B. Hayward, Bjarne Toft, and Philip Henderson. How to play reverse hex. *Discrete Mathematics*, 312:148–156, 2012.
10. Piet Hein. Vil de laere Polygon? *Politiken*, December 1942.
11. Philip Henderson. *Playing and solving Hex.* PhD thesis, UAlberta, 2010. `http://webdocs.cs.ualberta.ca/~hayward/theses/ph.pdf`.
12. Philip Henderson, Broderick Arneson, and Ryan B. Hayward. Solving 8x8 Hex. In Craig Boutilier, editor, *IJCAI*, pages 505–510, 2009.
13. Jeffrey Lagarias and Daniel Sleator. *The Mathemagician and Pied Puzzler: A Collection in Tribute to Martin Gardner, editors Elwyn Berlekamp and Tom Rodgers*, chapter 3, pages 237–240. A.K. Peters, 1999.
14. Jakub Pawlewicz and Ryan B. Hayward. Scalable parallel dfpn search. In *Computer and Games*, Springer LNCS 8427, pages 138–150, 2013.