cmput 396 hex notes by Prof Hayward   part 1/2

solving a game

Two ideas are useful when solving Hex states: inferior cells and virtual connections.

First, some definitions. A *state* or *situation* $S = (P, X)$ is defined by the board *position* P and the *player-to-move (ptm)* X, ie. the player X whose turn it is to move next. An algorithm is *explicit* if, for any state reachable from the start state, it can computes the next move of the strategy in time polynomial in the boardsize.

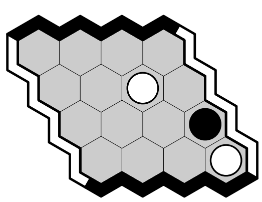
To *solve* S is to determine the outcome, assuming perfect (ie. minimax) play by both players. Hex has no draws, so the winner of S is either ptm (ie. 1st player to play from S), or the opponent of ptm (ie. 2nd player to play from S).

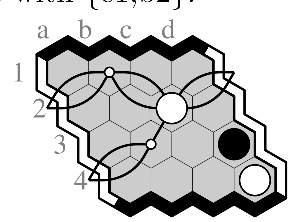
If we solve S and learn only the outcome, but not how to play to achieve that outcome, then we have *weakly solved* S. If in addition we find an explicit provably-winning strategy (ie. we can prove that it wins against all possible opponent strategies), then we have *strongly solved* S.

An example. Let S be the state consisting of the empty 100×100 board. By the strategy stealing argument, we know that there exists a winning strategy for the first player. So S is weakly solved.

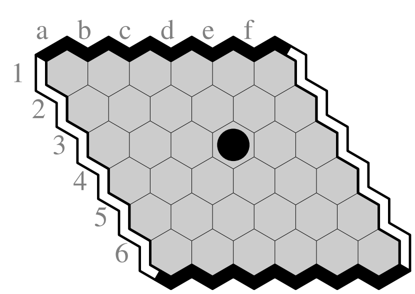
The only algorithms that I know that solve Hex positions take time at least exponential in the boardsize. As far as I know, no one has strongly solved S. Today, the best solver takes several months to solve the hardest 9x9 position.

Another example. Strongly solve this position, with Black to play. Hint: the second diagram shows you a winning strategy. The white stone at c2 is virtually connected to the right side using cell set {d1,c2}. On the left, if the opponent plays in any of {a1,a2,b1,b2,c1} then reply at b3, virtually connected to the left with {a3,a4}; if the opponent plays in any of {b3,a3,a4}, reply at b1, virtually connected to the left with {a1,a2} and to c2 with {c1,b2}.





Another example. Let S be the 6×6 position with a black stone on the centermost cell. It is not too difficult to find an explicit provably winning strategy for Black (so White, who plays next, loses). So S is strongly solved.

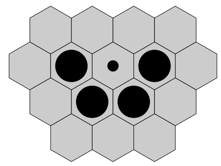


Challenge problem. White to play: find a winning Black strategy.

inferior cells

When searching for a winning move, sometimes some cells can be easily ruled out. The idea in inferior cell analysis is to identify cells that are inferior in some sense to other cells. In the search for a single winning move, moves to inferior cells can be ignored.

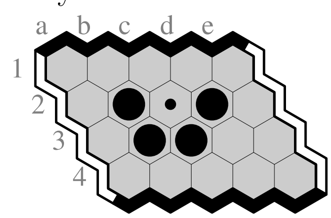
Consider the following part of a Hex position. The dotted cell is useless: neither player should play there. Can you see why?



dead cells

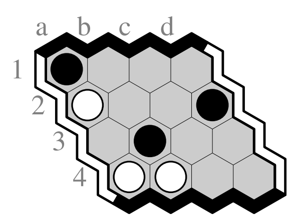
For a Hex position P and player X, a *joinset* is a set of empty cells which, when X-colored, joins X's two sides. A *joinset* is minimal if is not a proper subset of some other joinset.

Eg. below, find a joinset for Black, find a joinset for White, find a minimal joinset for Black, find a minimal joinset for White. Can you find a minimal joinset for Black that includes the dotted cell `c2`? For White? What does this tell you about the dotted cell?

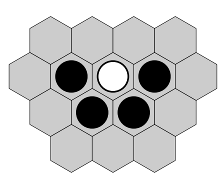


Eg. in the position below, the set { `a3 b2 c2 d1` } is a white joinset, but it is not minimal, because it is a superset of the white joinset {`b2 c2 d1` }. For a position P, a cell is *live* if it is in some minimal joinset (for either player), otherwise it is *dead*.

Eg. cells `b2, d2, d1` are all live: they form a white joinset, and if we remove any one cell from the joinset, it is no longer a joinset. So the joinset is minimal, so each cell in it is live. Exercise: explain why `a3` is dead.

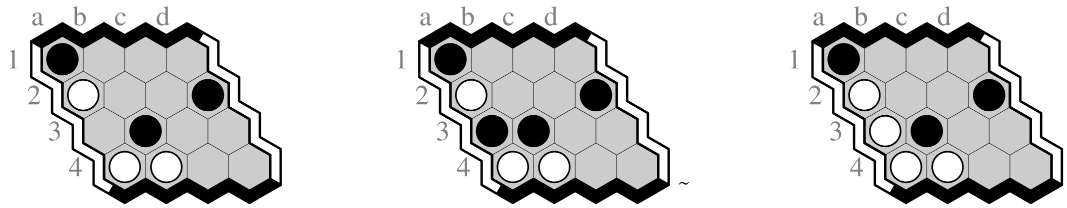


We extend the definition of dead from empty cells to colored cells: we call a colored cell *dead* if uncoloring it yields a dead cell. Eg. the white stone here is dead.



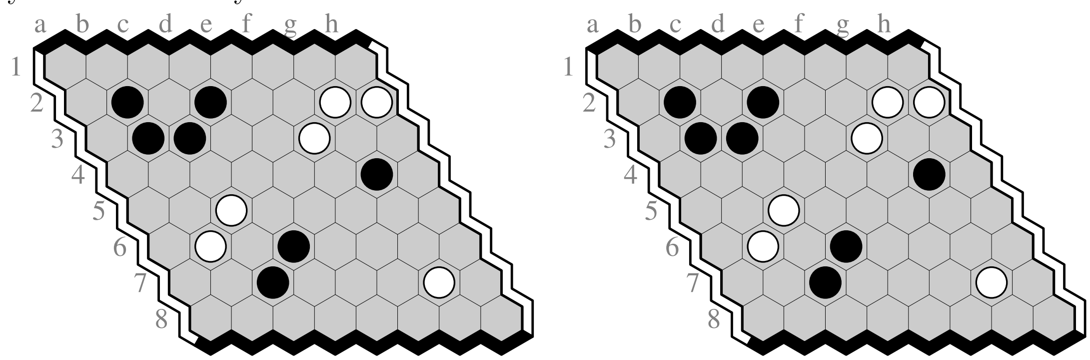
theorem For any Hex position, changing the color (empty, black, white) of a dead cell does not change the winner of the position.

Eg. fix the player to move: then the winner of each position below is the same.



So, when searching for a winning move, if we see that a cell is dead, we can prune it from our search.

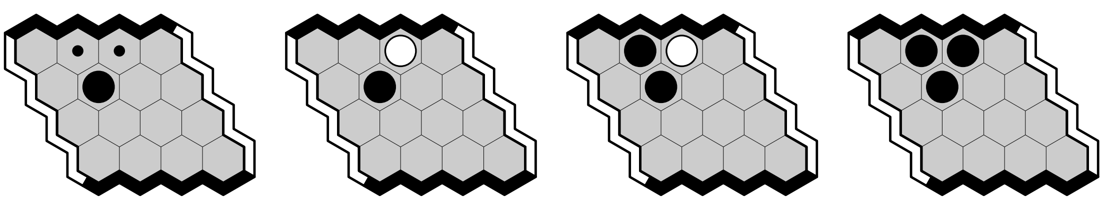
Sometimes, as above, we can tell that a cell is dead just by looking at the neighbouring cells. How many dead cells can you find here?



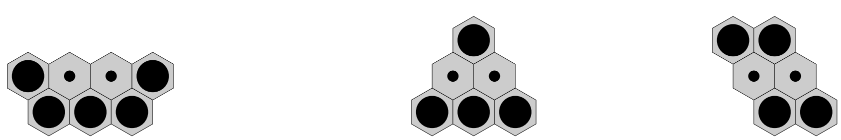
captured cells

In the 1st diagram below, if White ever plays at one dotted cell (as in the 2nd diagram), Black can reply at the other dotted cell (as in the 3rd diagram), and so kill the White cell. So: it is useless for White to play at either dotted cell. We call a set of cells *black-captured* if Black has a replying strategy that leaves every cell in the set black or dead.

useful in solving Hex states: fill each Black-captured sets with black stones. This reduces the size of the search tree.

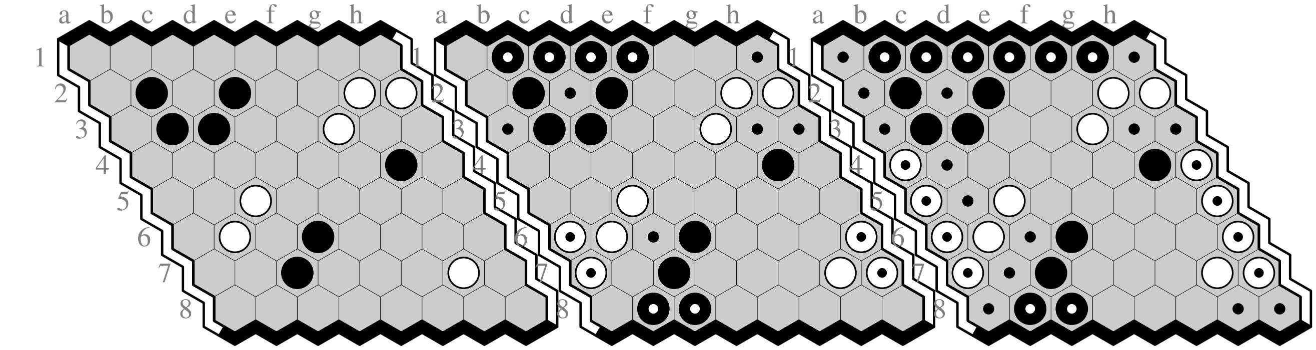


Here are three common captured set patterns. Page 55 of this paper `http://www.cs.ualberta.ca/~hayward/papers/bergeParis.pdf` shows others.

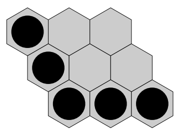


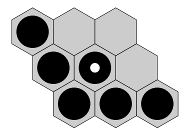
On the 8x8 board above, fill in as many dead and captured cells as you can.

Here, with each step I fill in more dead and captured cells. Whoever wins the first state wins the last state. Obviously, this kind of fill-in reduces the search space.

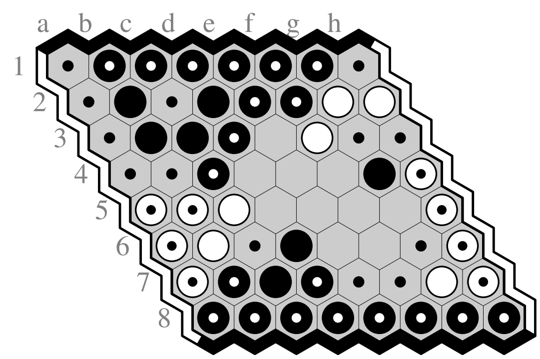


Finally, here is one more fillin pattern, called *permanently inferior*: whenever you see the pattern at the left, you can add a black stone as on the right. If you want to know why this is ok, read this paper `https://www.cs.ualberta.ca/~hayward/papers/revDom.pdf`.





If you fill in the above 8x8 board using dead, captured, and permanently inferior fillin, you get end up with something like this:



solving 7x7 Hex

Here is how mustplay analysis works. When it is your turn to play, consider the opponent's threats. Is there any empty cell, such that if the opponent plays there, they then have a winning virtual connection? If yes, then the union of that cell plus the cells used by the virtual connection is an *opponent threat-set*. So, find all opponent threat-sets that you can, and then take the intersection of all of them: this combined intersection is your *mustplay region*. Your next move must be in this region: if it is not, then you have missed at least one opponent threat-set, and after you move your opponent can win.

Around 2003, a group at UAlberta (Björnsson, Hayward, Johanson, Kan, Po, Van Rijswijck) were trying to solve Hex positions.

By pruning dead cells, computing virtual connections and using mustplay analysis they could solve positions on 6x6 but not 7x7.

And then they realized that filling captured cells with stones of the capturing player does not change the outcome of the game. So they added that feature to their program and then could solve all 7x7 positions. More details are here

`http://webdocs.cs.ualberta.ca/~hayward/talks/hex.solve7x7.pdf`

so far

To date, all opening moves on all board sizes up to 9x9 have been solved, and two opening moves on 10x10. More details are here

`http://webdocs.cs.ualberta.ca/~hayward/talks/hex.sol10.pdf`

It would be great if someone could find a winning 11x11 Hex strategy. This is the original boardsize introduced by Piet Hein in 1942 when he introduced the game (then called Polygon) to readers of the Danish newspaper *Politiken*.