About the Algorithms

3PixelRule.tiff ¬

Plain Search

Plain search is a vanilla depth first search, one letter at a time. The program checks after each letter to make sure that all words in the puzzle still have potential completions. When the current puzzle is deemed impossible, the algorithm backtracks to the last square it filled and tries a different letter.

If you watch the program as it uses plain search, you'll see that its simple minded backtracking is horribly inefficient. Quite often, the program repeatedly tries every combination of letters in parts of the puzzle that have nothing to do with its current impasse.

Backjumping

Backjumping extends depth first search by trying to backtrack intelligently. Rather than just changing the most recent letter that it filled, the algorithm erases letters until the current problem in the puzzle has been eliminated. The search then continues. Backjumping is an enormous improvement over plain search, but it has the drawback that it sometimes erases large portions of the puzzle that it has already successfully completed.

Leapfrog

Leapfrog search tries to eliminate this last difficulty of backjumping. It does so by caching any partial solution it finds and is forced to backjump over. After changing the problem square, it can try its cached solution, potentially saving all the time it would take to

rediscover the same solution from scratch.

As you watch the program, partial solutions that the program remembers (but has really erased!) are indicated by yellow squares with red letters.

paste.tiff ¬

Space does not allow a more elaborate explanation. A good introductory AI text will describe the first two algorithms in more detail. Leapfrog search arose as part of my research and is currently undocumented. If you watch the program you will quickly see why: sometimes the caching helps, and sometimes it hurts. Overall it appears to give little improvement over backjumping.