PROJECT

Write a program for computation of the shortest trajectories in ABG.

Input: set X (a 2D table with or without obstacles), an element p for which the trajectories should be calculated, relations of reachability for this element (if necessary*), location of the element (the start of trajectories), the end of trajectories, the length of trajectories.

*) Remark: If these are usual relations of reachability of a chess piece for a standard chess board they do not have to be defined explicitly. Instead you can refer to them by the respective number of the table 15x15, which can be stored in advance, e. g., the numbers 1-6.

Algorithm: You can use grammar $\mathbf{G_t^{(1)}}$ but it generates only one trajectory. Your program should add something to it to generate ALL the shortest trajectories.

Output:

Print shortest trajectories for sample locations for all the chess pieces (for 8x8 board with and without obstacles): Pawn (assume that Pawn can move straight ahead only), Knight, Bishop, Rook, Queen, King.

Include at least one example of computation of trajectories for the element with unusual relations of reachability (different from chess pieces) and unusual board defined by you as part of the input of this program.

The output should be both: a graph "drawn on the board" and a list of coordinates of locations (stops) along those trajectories. A reasonable explanation (or a proof) that all the required trajectories have been generated should be included.

In particular, as a simple test generate all the trajectories for the King from a5 to h5 of the length 7. You do not have to print them all (if your output is not a graph) but print their total number.

It is not necessary but you can also read my book
“Linguistic Geometry: From Search to Construction”, Kluwer, 2000.
It might be helpful.