Using The Exact Cover to Solve The Pentomino Puzzle

I used the below reference in the PDF provided for this assignment. In this reference it speaks on how the original creators of the DLX algorithm creates it and implements it.

“Hitotumatu and Noshita used (1) to remove a column and (2) to restore it again; this meant that they could find an empty column without having to search for it.

This process causes the pointer variables inside the global data structure to execute an exquisitely choreographed dance; hence I like to call (1) and (2) the technique of dancing links.

1. And (2) being used back and forth moving the items in the list is how DXL got the nickname dancing linked lists.”

Reducing the Pentomino problem into the Exact Cover Problem was straightforward based on Knuth’s research on the DLX (Dancing Linked Lists). Here is how I came to my solution:

Initially, I recognized that the Pentomino problem is a problem of combinations, where the pieces are placed on a board. From there, I understood that there are a limited number of locations and orientations for each piece on the board. With this knowledge, I was then able to brute force the pieces into all possible positions by going through each piece and pushing it to the grid, checking the direction of the pieces, and checking to validate that the pieces will fit appropriately on the grid based on the rows and columns provided.

Knowing that pieces cannot overlap, and if they do, my solution would be invalid, I put logic in place to only add valid positions in the array that I would be encoding. I also realized that I needed to cover each spot in the board at least one time. Knowing this, I broke it down even further and started realizing that there is some combination of the finite possible positions of all the pieces, where each piece is picked once. With the directions of the pieces in mind, I ensured that every spot within the board was taken.

In the Exact Cover Problem, I had a collection (S) of subsets when I collected the finite positions of the pieces on the board. I had a set (X) when I decided on the pieces to use. I was able to break the Pentomino problem down through brute force, and some observations into a problem, which is NP Complete. I was able to efficiently find all subcollections (S\*) of S so that each Pentomino piece in X was used once, and each spot on the board was covered once.

Algorithm – X : nondeterministic algorithm will find all solutions to the exact cover problem defined by any given matrix A of 0s and 1s.







