

## Crossword Puzzle Solver using as CSP

### Team Members:

<b>Hardik Udeshi (hvudeshi)</b> hvudeshi@ncsu.edu Department of Computer Science NC State University	<b>Het Patel (hpatel28)</b> hpatel28@ncsu.edu Department of Computer Science NC State University
---	---

### Problem Description:

A puzzle consisting of a grid of squares and blanks into which words are to be filled vertically and horizontally. In the crossword puzzle, we have a grid with blocked and unblocked cells and a dictionary of words. We want to assign a letter to each unblocked cell so that each vertical or horizontal contiguous segment of unblocked cells forms a word that appears in the dictionary. Given a word dictionary consisting of a list of words, we need to fill the blank squares with the alphabets such that every constraint like the word size and the overlapping word constraints are fulfilled. For example in the above image shown, the word dictionary will have a list of words like Cup, chaste, petal, tulip, etc. stored.

C	U	P		L	E	T	H	A	R	G	I	C
H		E		E		A		D		O	N	O
A	P	T	T	O	A	C	H	O	O		A	N
S		A						O			W	V
T	U	L	I	P		A	S	I	N	I	N	E
E					I		D		N		P	Y
	B	A	D	T	E	M	P	E	R	E	D	
C		L		H		A		P				C
H	A	L	C	Y	O	N		T	A	C	K	Y
E		Y				D				O		G
A	H		M	E	D	I	C	A	L	M	A	N
T	A	J		O		I		N		M		E
S	H	R	I	N	K	I	N	G		A	P	T

## Task Environment:

- **Fully Observable** - It is fully observable because at each state the agent can sense the whole environment.
- **Deterministic** - After inserting the values for one word we can determine the values to be inserted in adjacent blank spaces.
- **Sequential** - It is sequential because each word we put in a blank space can have long-term consequences
- **Static** - The environment is the same unless and until an agent acts.
- **Discrete** - It is discrete because a length of blank spaces can only have a finite number of words that can be inserted into blank spaces.
- **Single** - Only one agent working to solve the problem
- **Known**- The environment is known to the agent before the start of the game.

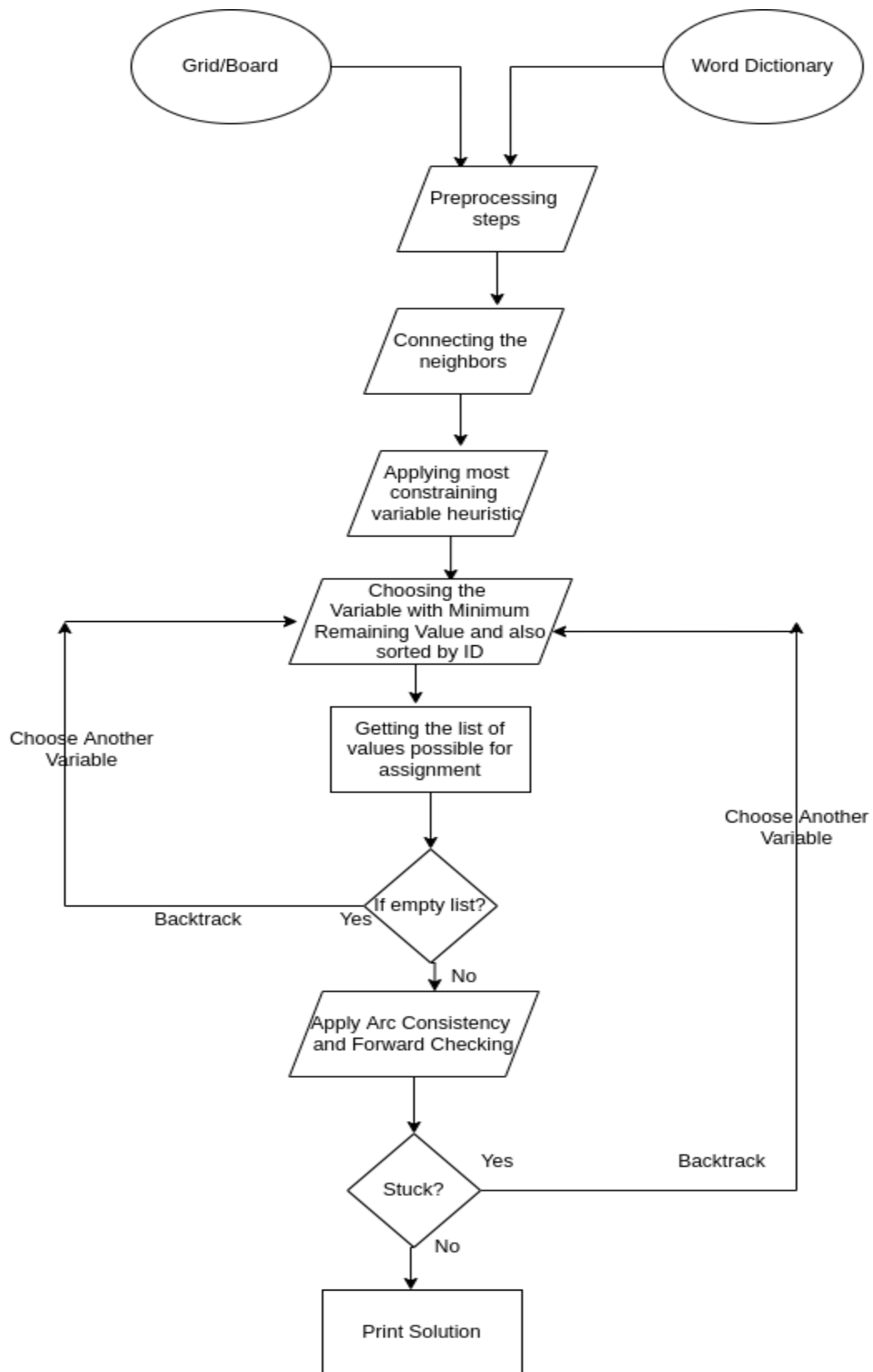
## Approach:

The crossword puzzle problem can be described as a Constraint Satisfaction problem as follows:-

**Variables:** All vertical or horizontal contiguous segments of unblocked cells.

**Domains:** The domain of each variable is the set of words in the dictionary of the same length as the corresponding contiguous segment.

**Constraints:** For each pair of vertical and horizontal contiguous segments of unblocked cells that intersect at an unblocked cell  $s$ , we add a constraint between them, constraining the words assigned to them to have the same letter at  $s$ .



**Pre-Processing Steps:**

- From word dictionary, for every possible word length, a 26 x (possible word length) combinations are created and stored in such a way that every alphabet at every possible index of the word length.
- All the horizontal and vertical blank spaces are found and will be considered as a Variable (in CSP term).

**Connecting the neighbors:**

- Traversed through all the horizontal and Vertical variables and those who are connected are stored in the form of an adjacency list.

**Applying Most Constraining Variable Heuristics:**

- The Most Constraining Variable is chosen as an heuristic. In most constraining variable heuristics, the horizontal or vertical variable which has the major constraints (connections) are chosen first.

**Choosing the Variable with Minimum Remaining Value:**

- Among all those variables which have the highest constraint, the variable having the minimum possible value is chosen and assigned first.
- Now again, if there exist some variables which satisfy both of these conditions, then this list is sorted based on the unique ID it is assigned.
- If there exists no possible remaining values for the variable, then we need to backtrack and choose again other variables.


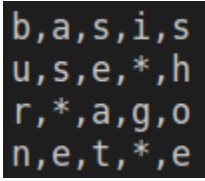

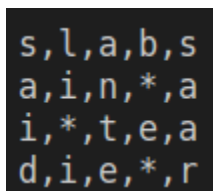
**Applying Arc Consistency and Forward Checking:**


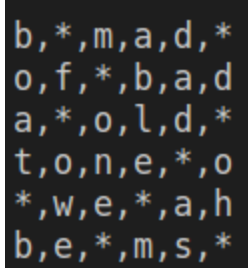

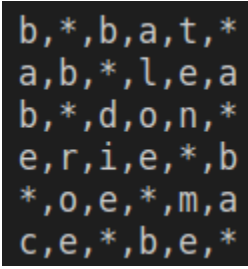
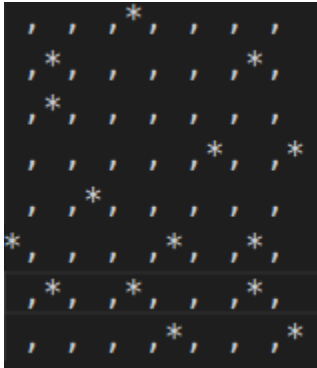
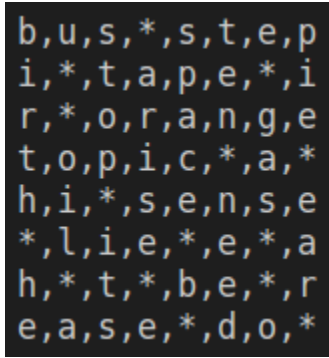
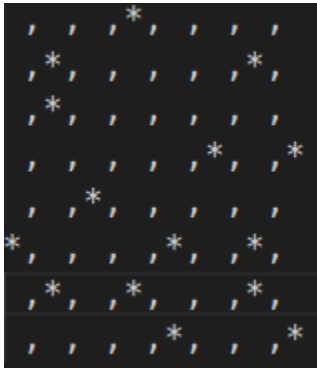
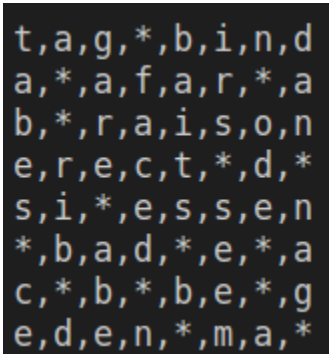
- **Forward Checking:**
  - After each successful assignment, we will loop over all of the neighbors and apply the limitations imposed by assigning the value.
  - Any neighbor whose values have become zero will suffice as a signal to assign a different value.
  - The other value will be the appropriate next value that satisfies the requirement for selecting a value.
  - There will be a backtracking if there is no other value.
- **Arc Consistency:**
  - After some of the neighbors' values are deleted during forward checking, the arc consistency stage will begin.
  - We will traverse all of the neighbors in arc consistency, and for each one, we will take the intersection cell of them and their neighbors.

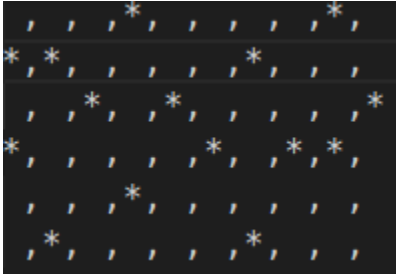
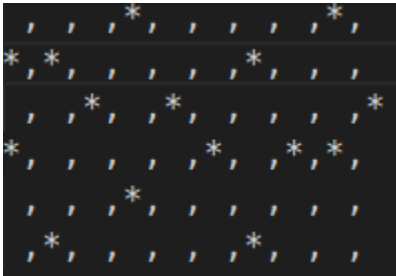
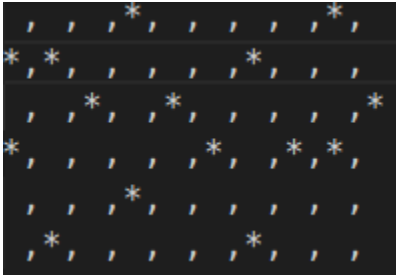
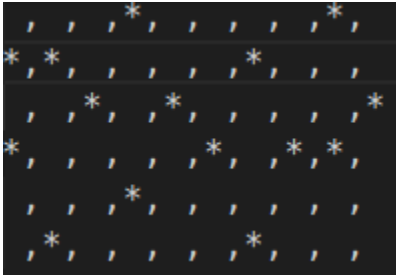
- Through arc consistency, we will add another constraint which will fix alphabets at the intersection cell and prune the domains of the neighbors and the neighbor's neighbor.
- If no such assignment of words is possible, then we have to backtrack and choose the other word from the list while keeping the constraints in mind.

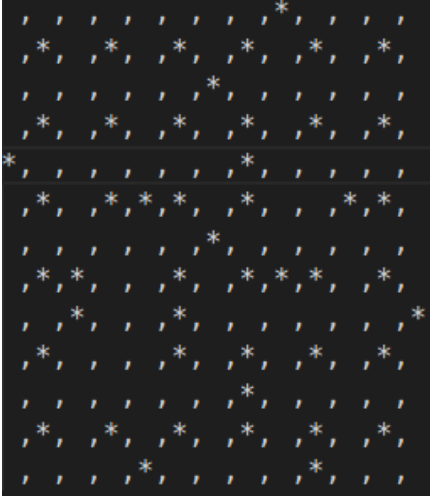
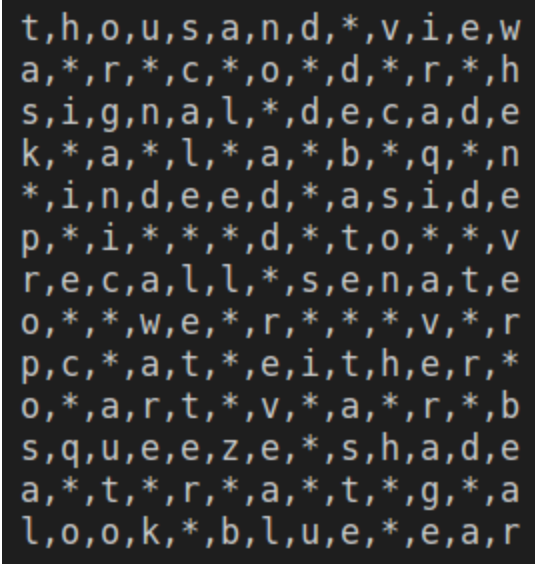
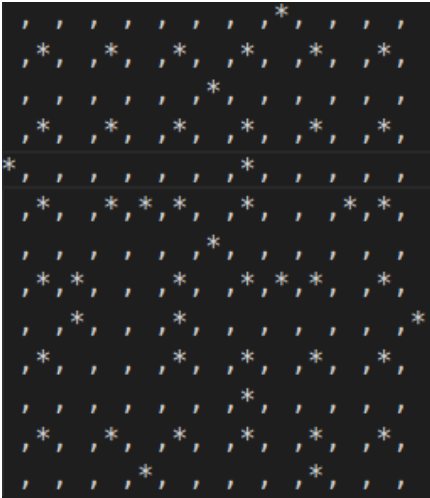
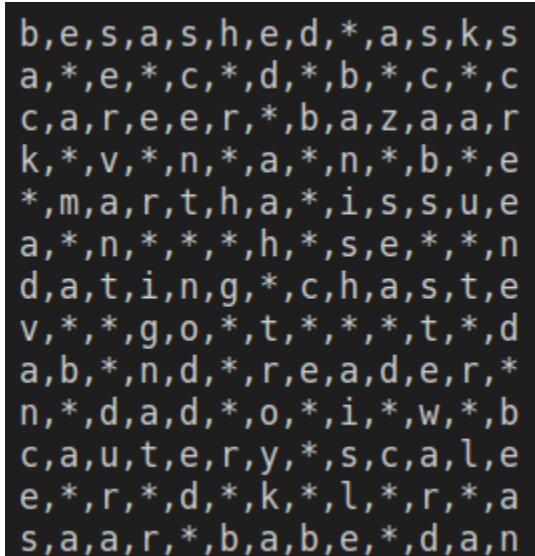
If after applying forward checking and arc consistency, if we find any solution satisfying all the constraints, then we got the solution and the solution is printed. Also, if after trying all the possible values and we could not find any possible assignment, then we print that there doesn't exist any possible solution for the particular grid and given a specific word dictionary.

## Results:

Sr. No.	Crossword board	Difficulty Level	Word List#	Total Backtracks	Time Taken
1	grid1 (4x5)	Easy	wordList1 (3026 words)	0	30ms
					
2	grid1 (4x5)	Easy	wordList2 (29162 words)	0	75ms
					
3	grid2 (6x6)	Medium	wordList1 (3026 words)	37	106ms

					
4	grid2 (6x6)	Medium	wordList2 (29162 words)	3277	1389ms
					
5	grid3 (8x8)	Medium	wordList1 (3026 words)	4397	1526ms
					
6	grid3 (8x8)	Medium	wordList2 (29162 words)	0	155ms
					

7	grid4 (6x10)	Hard	wordList1 (3026 words)	Infinite	Infinite
			No Solution Found Running Infinitely.		
8	grid4 (6x10)	Hard	wordList2 (29162 words)	18848	9536ms
			<pre> a,f,t,*,c,e,n,t,*,d *,*,a,r,e,a,*,a,b,e a,m,*,a,*,t,a,t,e,* *,e,a,t,s,*,c,*,*,b a,n,n,*,a,t,t,i,c,a b,*,a,b,l,e,*,d,a,t </pre>		
9	grid4 (6x10)	Medium	wordList3 (28 words)	0	60ms
			<pre> a,g,o,*,t,a,p,e,*,o *,*,n,e,a,t,*,a,b,d t,o,*,a,*,m,o,r,e,* *,n,o,t,e,*,w,*,*,t g,e,t,*,c,o,l,u,m,n o,*,p,o,o,l,*,p,o,t </pre>		
10	grid4 (6x10)	Medium	wordList3 (27 words)	9 (Solution not found)	53ms
			Solution Doesn't Exist		
11	grid5 (13x13)	Hard	wordList1 (3026 words)	49	466ms

					
12	grid5 (13x13)	Hard	wordList2 (29162 words)	0	877ms
					

### Discussion on results:

- From the results, we are getting some assignments where the number of backtracks are 0, because we are choosing the variable in such a way that it has the most constraining variable and the minimum remaining value possible and in that way the number of backtracks will be minimum.
- When the number of backtracks is zero (grid1), the time taken for wordlist1 is lower than the time taken for wordlist2, because the number of words are less.



- As the number of words in the dictionary increases, chances of getting the solution is high as there can be many possible domains.
- For grid 4, we tried all the dictionaries:
  - For wordlist1, the number of backtracks are Infinite and the time taken are infinite, because the domain is quite big and it will take a lot of time to try out all the possible domains in the word.
  - For wordlist2, we are getting the solution after 18848 backtracks as the number of words in the dictionary are 29162. Now, here, we got the solution, because chances of getting the possible assignment for this particular wordlist are high.
  - For wordlist3, the list of words is quite small, hence it is finding a solution in a small amount of time.
  - For wordlist4, there is no possible combination which can satisfy the constraints, and the word list is small too, so the number of backtracks are less and through this also, we can see that through our approach, the number of backtracks are less when there is no possible assignment.

## References:

1. Crossword Puzzles and Constraint Satisfaction by James Connor, John Duchi, and Bruce Lo.
2. <http://www.cs.toronto.edu/~sheila/384/w14/Lectures/csc384w14-Lecture-04-Backtracking-Search.pdf>
3. Artificial Intelligence A Modern Approach (4th Edition) (Pearson Series in Artificial Intelligence) by Stuart Russell, Peter Norvig (z-lib.org).
4. Constraint Satisfaction Problems:-  
[http://idm-lab.org/intro-to-ai/problems/solutions-Constraint\\_Satisfaction.pdf](http://idm-lab.org/intro-to-ai/problems/solutions-Constraint_Satisfaction.pdf)
5. Introduction to CSP:-  
<https://www.cs.ubc.ca/~kevinlb/teaching/cs322%20-%202008-9/Lectures/CSP1.pdf>