Lab 2 Artificial Intelligence

Shubham Chawla 239571

Crosswords generation as CSP

Introduction

The aim of the task was to become acquainted with the basic algorithms used to solve problems complying with restrictions (called Constraint Satisfaction Problem, CSP), by working on implementation and examining their properties of the problem.

Approach

Approach is focused on how to treat the space and the time well, accordingly, every step must be accurately measured and the decision of choosing the variables or the values must be done using heuristics.

Variable

Each square in the crossword board of dimension n X m is defined as variable.

Value

After choosing the variable, the next stage will be the value picking, we want to choose a value that maximizes other neighbours values. Not just maximize them in total but also no zero neighbor values. This is done simply by knowing how each letter frequency in each position using the array list in the value which we are linking to, and that's our heuristic for the value. If there's no such value, feedback will be sent to the DFS loop so that backtracking will occur.

Constraints

We need to find such a way of placing words across the board that one field includes at most one letter and every continuous sequence of letters read horizontally or vertically is a word from the selected subset of the size k of S. A solution must include exactly k words from S. Some fields of the board can be left empty.

Backtracking

A recursive approach used to print the crossword whose size is manually defined in .txt files.

The word-byword instantiation approach is used in this algorithm on repeatedly picking empty word patterns from the grid and filling them up with valid English words from the txt file. It requires lesser operation than letter by letter instantiation.

The efficiency of the algorithm depends on three factors which need to be considered at every step along the execution:

a) Which uninstantiated variable (empty word pattern) to fill in?

b) Which word to fill into the uninstantiated variables?

c) What point to backtrack to?

a) Fill Strategy – The Fill Strategy maximizes the number of choices for the remaining words on the grid, thus minimizing the number of backtracks to reach a solution.

I have applied this heuristic.

Most Constrained / Fail First: checks how many matches each word pattern has and chooses the one that has the least number of matches, for eg words of length = 2.

b) Pick Strategy –

The Pick Strategy chooses a word to fill a pattern with and maximizes the

number of choices for the remaining patterns in the grid. The heuristic used was

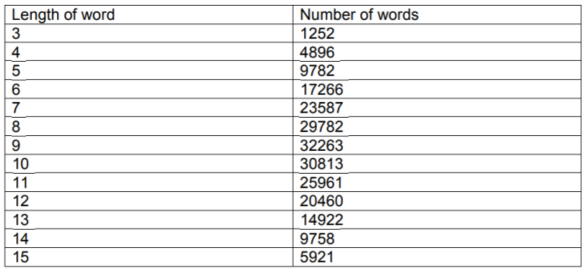
based on the 'first n' possible matches heuristic. Depending on the number of

words that are present in the database of a particular sized word, n is varied so

that it reflects the number of possible matches. For example there are 1252 words of length 3

whereas there are 32263 words of length 9 in the database. Thus n is chosen to

be 20 for 3 letter words and 100 for 9 letter words.



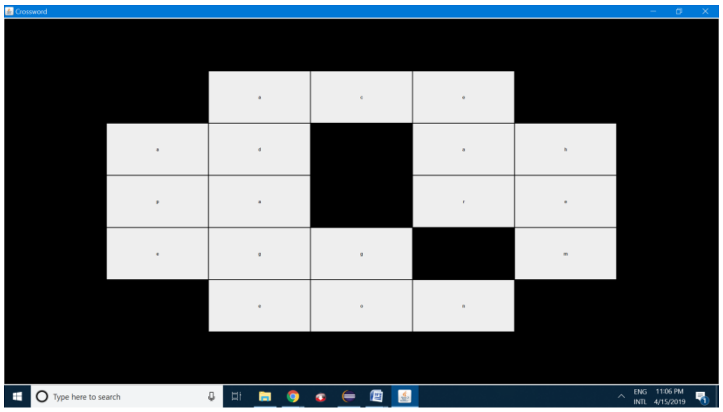
Length of word	Number of words
3	1252
4	4896
5	9782
6	17266
7	23587
8	29782
9	32263
10	30813
11	25961
12	20460
13	14922
14	9758
15	5921

c) Arc-consistency – Arc consistency allows determining whether the grid is consistent or there are word patterns which cannot be instantiated with words. Arc-consistency prevents having to search the entire grid for a word pattern to backtrack to when some word pattern cannot be instantiated. To implement arc-consistency, a hash table was used which has as keys each element belonging to the "across" and "down" word patterns and as values a list consisting of intersecting word patterns.

```
Manual creation of crossword board and words are loaded from a text file into it .
```

```
XXXXXXX
XX...XX
X..X..X
X..X..X
X...X.X
XX...XX
XXXXXXX
```

Output is as follows:



On console window:

```
X a d X a h X

X p a X r e X

X e g g X m X

X X e o n X X

X X X X X X X

Backtracking time: 4.046s

Forward Search time:1.555365029132E9s

true
```

The program calculates the time for backtracking approach i.e. 4.046 in this case.

Difficult Boards will take a rather long time.

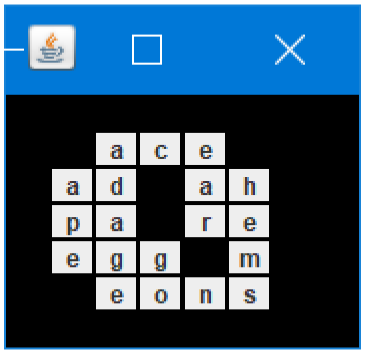
Another Solution for Same Problem

Forward search approach

As we all can see its way faster than the backtrack search.

Another example for

```
XXXXXXXX
XX...XXX
X..X..XX
X..X..XX
X...X.XX
XX....XX
XXXXXXXX
```

We get 

```
X X X X X X X X
X X a c e X X X
X a d X a h X X
X p a X r e X X
X e g g X m X X
X X e o n s X X
X X X X X X X X
Backtracking time:3.431s
Forward Search time:1.555365203907E9s
true
```

Analysis

The dependency of the time (s) algorithm to the size of the problem, for the crossword.

N	easy	normal	hard
BT	3.43	4.06	210.84
FC	1.55	1.6	180.3