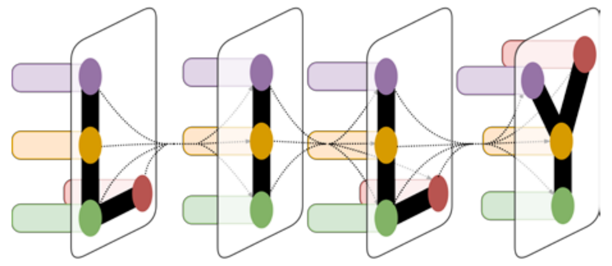
Yale

Generating 3D Crosswords as a Weighted Constraint Satisfaction Problem

Michael Menz[1]

[1]Yale College, New Haven, CT.



Introduction

Crossword constructors increasingly rely on computational tools to fill grids and finish puzzles. A solution to a crossword puzzle can be viewed as a constraint satisfaction problem, where each fill is a variable and the crossing points and lengths are constraints to be satisfied. With this framework, it is possible to build crosswords more complex than could be generated by human constructors. We apply methods used to solve weighted constraint satisfaction problems to generating three-dimensional crossword puzzles.

Why is this hard?

Figure 1 shows two graphs where the nodes in the graphs are words in a puzzle and an edge connects two nodes if the words cross. The graph on top is for a New York Times Thursday puzzle and the graph on the bottom is for a 4x4x4 3D puzzle. The NYT puzzle has densely connected pockets that are connected to each other by only one or two crossing constraints. Thus, these areas can be filled almost independently. This is not the case for our 4x4x4 puzzle.

Weighted CSPs

Filling a crossword can be considered as a constraint satisfaction problem, where we want to select some number of fills as variables such that they satisfy all the length and crossing constraints. Every variable has a large search space as every combination of letters could feasibly be an acceptable fill. Crossword constructors make use of large corpuses of previously used fills with associated “goodness” scores to assist in puzzle construction. Using these scores we can evaluate partially completed CSPs, turning our problem into a weighted CSP. We can then prune our search tree once we find any acceptable solution. This method is called branch & bound. The initial solution need not be very good in order to vastly improve our computation time. We want to sort our potential fills so that some solution can be found. This is done by giving fills an adjusted score based on the frequency of their letters and their goodness score and then bounding solutions based on the total goodness score.

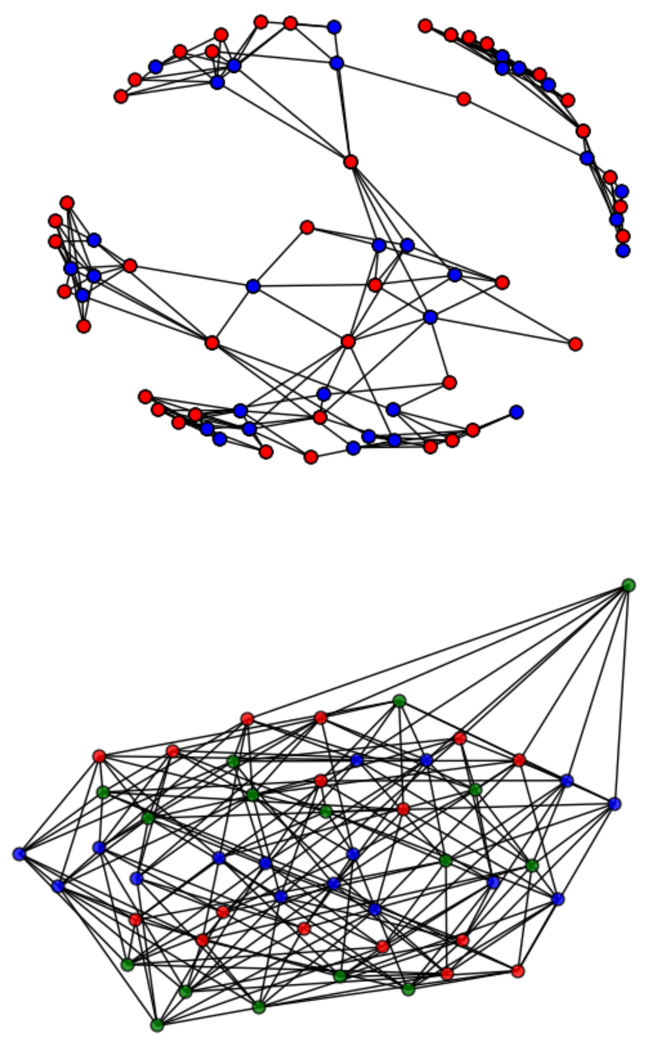
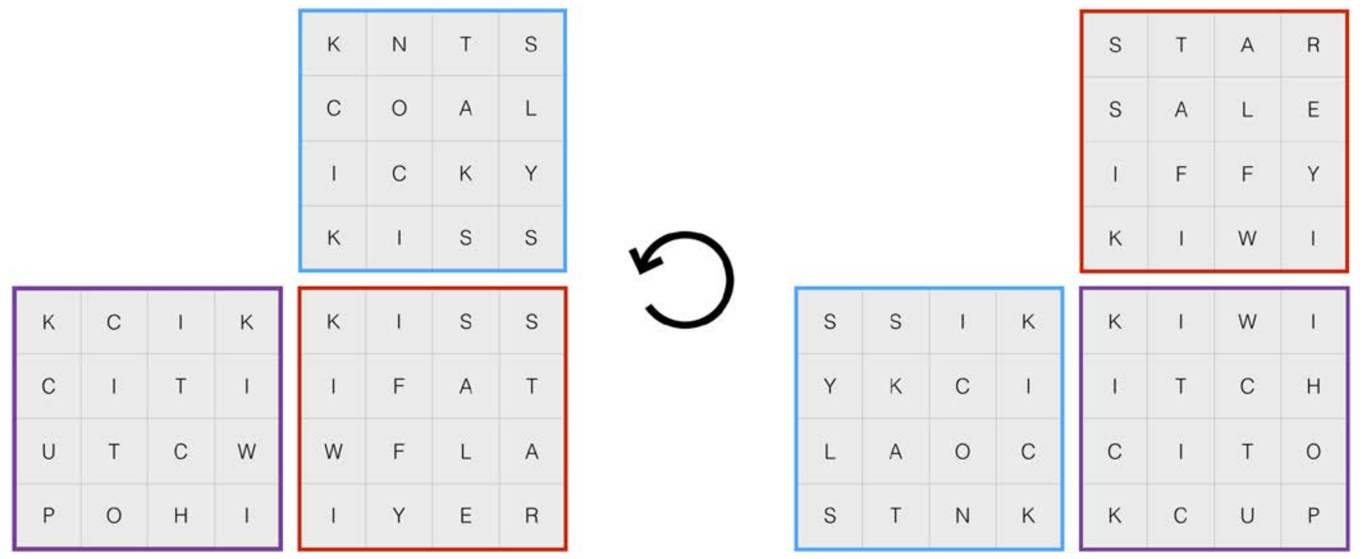


Figure 1. The two graphs above shows crossing constraints of a NYT puzzle (top) and a 4x4x4 puzzle (bottom).



K	N	T	S
C	O	A	L
I	C	K	Y
K	I	S	S

K	C	I	K
C	I	T	I
U	T	C	W
P	O	H	I

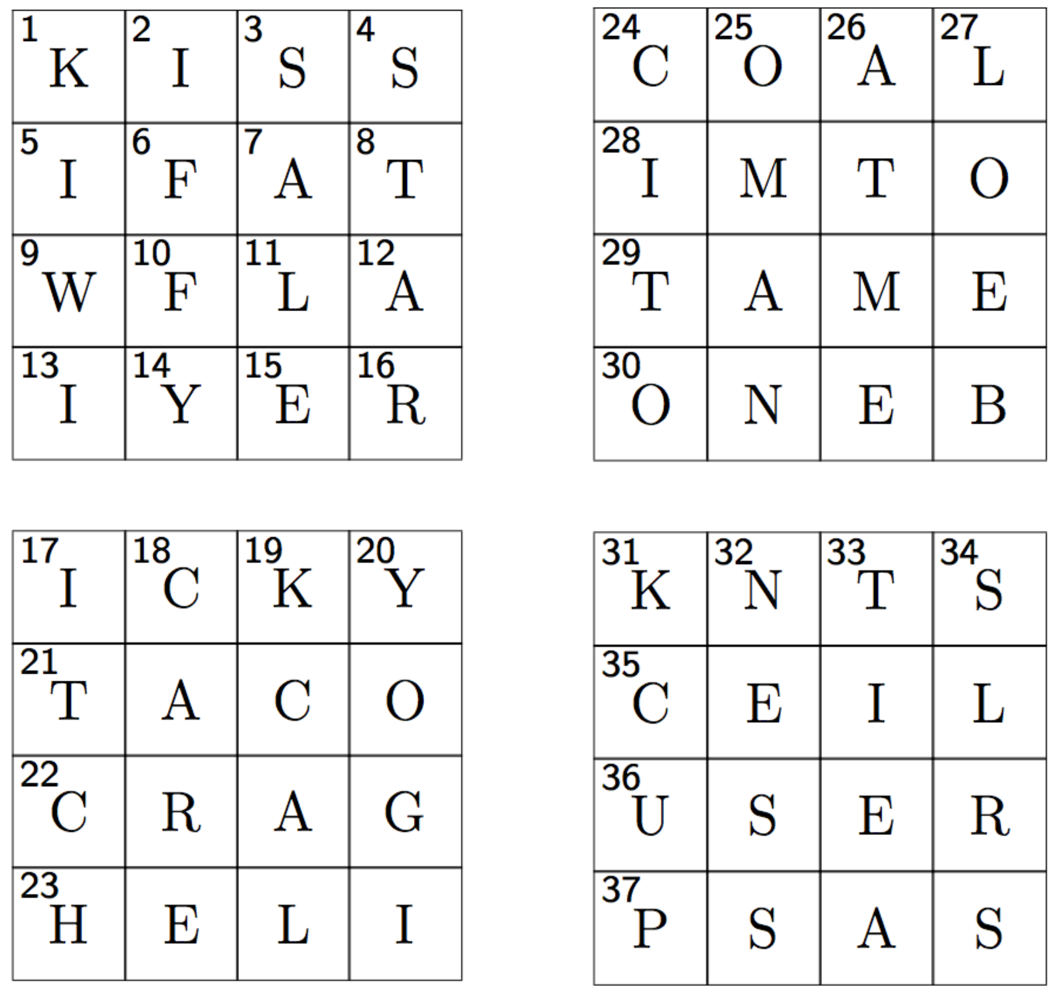
K	I	S	S
I	F	A	T
W	F	L	A
I	Y	E	R

S	T	A	R
S	A	L	E
I	F	F	Y
K	I	W	I

S	S	I	K
Y	K	C	I
L	A	O	C
S	T	N	K

K	I	W	I
I	T	C	H
C	I	T	O
K	C	U	P

Figure 2. The crossword of Figure 3 shown from the front (red), the side (purple), and the top (blue). A rotation can be applied to the crossword to obtain a new 4x4x4, with the front as the top, the side as the front, and the top as the side. We can continuously permute in this manner, to progressively stack 4x4 puzzles from the front and the fill the cube. If we have added *k* puzzles in the each of the other directions, then we will have already filled the first *k* down and across clues in the next 4x4 we add to the front. This greatly restricts our search space, and these puzzles can be found with a dictionary lookup.



1 K	2 I	3 S	4 S
5 I	6 F	7 A	8 T
9 W	10 F	11 L	12 A
13 I	14 Y	15 E	16 R

24 C	25 O	26 A	27 L
28 I	M	T	O
29 T	A	M	E
30 O	N	E	B

17 I	18 C	19 K	20 Y
21 T	A	C	O
22 C	R	A	G
23 H	E	L	I

31 K	32 N	33 T	34 S
35 C	E	I	L
36 U	S	E	R
37 P	S	A	S

Through 1 Recoil **2** “My Computer, ” e.g. **3** Card game **4** Parts of a wd. **5** Suffix with arthr- **6** Hall of ___ **7** When Romeo meets Juliet **8** Jimmy or jack **9** Alcohol-eschewing org. for the ladies **10** Monk monikers **11** Not cool **12** Antiquing agent **13** Eatery with a “Two x Two x Two ” option **14** Achings **15** Ancient home of Parmenides **16** Barbecue grub

Down 1 Fuzzy fruit **2** Far from certain **3** Bait for shoppers **4** Sky twinkler **17** Pruritus **18** Feel concern **19** Heat meas. **20** ___ Bear **24** Baseball manager Gaston **25** Arabian sultanate **26** “Don’t look ___! ” **27** Leopold’s accomplice **31** Single-serving coffee option **32** Faithful follower? **33** “___ yellow ribbon... ” **34** Cameras for pros

Across 1 “Sealed With a ___ ” (Brian Hyland hit) **5** “___ first you don’t... ” **9** Tampa’s area, gazetteer-style **13** Travel writer Pico **17** Revolting **21** Filled tortilla **22** Rugged rock **23** Prefix for pad or port **24** Shade of black **28** “For ___ be Queen o’ the May,... ” **29** Gentle **30** First-floor apartment **31** Clothing category **35** Cover, in a way **36** Exploiter **37** Anti-DUI spots

Figure 3. 4x4x4 Crossword. Clues are taken from a database of syndicated crosswords.

Rotational Properties

In generating three-dimensional puzzles, the breadth of the search tree turns out to be much more problematic than the depth. After a large number of fills have been placed, due to the high number of crossing constraints, there are very few options for the remaining fills. In order, to reduce the breadth of our search tree at the lower levels, we take advantage of the rotational symmetries of our puzzles. In particular, a solution forms a 3D puzzle when viewed from the top, left, or right (Figure 2). Rather than placing fills one at a time, we place 2D solutions one at a time in each of these directions. These solutions enforce immediate constraints on each other. Furthermore, these constraints can be applied in constant time (Figure 2).

Limited Discrepancy Search

One of the issues with branch & bound is that the search can become stuck in some large branch of the tree. This is usually bad because we believe that our heuristics will allow us to find the best solutions in any branch relatively quickly. One way out of this is to stop searching a branch after some number of “discrepancies”. In our application, a discrepancy occurs when we try to place a fill that has already been placed in the puzzle. When the number of discrepancies caused by a fill exceeds some threshold, we leave the branch of the tree that first involved that fill.

Results

The methods we apply allows us to construct 4x4x4 puzzles in fractions of a second. We also constructed a few 5x5x5 puzzles, but this required much more computation time and had significantly lower scored fills. One of the best scoring 4x4x4 puzzles is shown in Figure 3.

Acknowledgement

Thank you to Matt Ginsberg for providing the code for Dr.Fill which inspired many of the techniques used in this project. Also thank you to Dragomir Radev for advising the research.