Word Hypercubes are Fun, NP-Hard, and In General Undecidable

Barry Fagin
Leemon Baird

INTRODUCTION

Every year, the University of Chicago organizes a scavenger hunt on campus. This is like saying Genghis Khan organizes tea parties in central Europe. A UC scavenger hunt involves road trips, ridiculously hard-to-find items ("your appendix" was one), and silly stunts. "Scav" also includes plenty of brain teasers, because the University of Chicago is that kind of place.

The list of items participants must find and bring to the judges for scoring is routinely made public, so that parents and alumni can join in the madness. We call the readers' attention to item #345 from the 2012 list [3]:

> #345. A four-letter word. No, wait. A four-letter word square. No, wait! A four-letter word cube. NO, WAIT! A four-letter word tesseract. [4^1 points for a word cube, 4^2 points for a word tesseract]

A four-letter word square is a 4x4 grid filled in with four words such that all squares are symmetric about the diagonal. For example, the following is a four-letter word square:

S	A	F	E
A	R	E	A
F	E	A	R
E	A	R	N

Teams would be awarded four points for extending this idea in the obvious way to three dimensions, sixteen points if they could stretch it out to four.

This got us thinking about higher-order word objects. How many are there? What happens to the size of the solution space as the number of dimensions increases? As the word length increases? What is involved in coding up the problem for solution by computer? This article discusses what we have learned.

PREVIOUS WORK AND DEFINITIONS

Numerous articles in "Word Ways" [1,4,6,8,9,10,11,12] discuss word squares and word cubes of various sizes. Kon [12] discusses the concept of *word hypersolids*, the extension of word squares and cubes into different shapes and/or higher dimensions. Such objects can be either symmetric, where the words are the same in each dimension, or asymmetric, where they are allowed to differ. We will concern ourselves only with objects of equal length in all dimensions, which we will call *word hypercubes,* or simply *hypercubes* or even *cubes* when the context is clear.

Noting as Kon does that the terminology remains somewhat ambiguous, we restrict our analysis to what he calls "single" objects: Symmetric word squares extended in the obvious way to cubes, tesseracts, and so forth. For notational purposes of this article, (*w,d*) will denote a single word hypercube made from words of size *w* in *d* dimensions. Thus a (4,2) hypercube is a 4x4 word square, a (5,3) hypercube is a 5x5x5 word cube, and so forth.

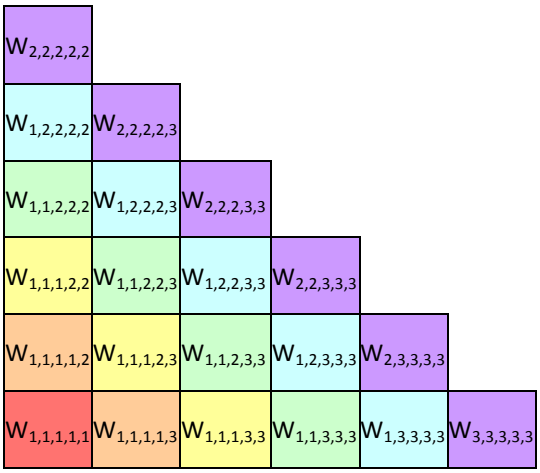
For example, a d=2, w=9 word square can be thought of as a square filled with letters, where $W_{r,c}$ is the letter on row r and column c. Then $W_{1,1}$ would be the upper-left corner, and $W_{9,9}$ the lower-right.

If d=4 and w=9, it becomes a 4-dimensional hypercube, with the letter $W_{1,1,1,1}$ at one corner, and $W_{9,9,9,9}$ at the opposite corner. If we leave off the last coordinate, we get a word. So $W_{4,2,7}$ is the 9-letter word made up of the letters $W_{4,7,2,1}$, $W_{4,7,2,2}$, ..., $W_{4,7,2,9}$. In a valid word hypercube, this word must be in the dictionary.

The word hypercube has words crossing each other, so there is one additional constraint: two letters must be equal if they have the same subscripts, but in a different order. So we must choose the letters such that $W_{4,7,2,1} = W_{1,2,4,7}$, and $W_{4,7,2,9} = W_{2,4,7,9}$. This would also imply that words must be equal when their subscripts are rearranged, so the word $W_{4,7,2}$ must be the same as word $W_{2,4,7}$.

Given these constraints, we could choose to fill a word hypercube by choosing only the words and letters whose subscripts are sorted in ascending order. So we would choose the letter $W_{1,2,4,7}$, but would not explicitly choose $W_{4,7,2,1}$, because the latter is determined by the former. Similarly, we would only check words against the dictionary if they have sorted subscripts. So we would check $W_{2,4,7}$, but not $W_{4,7,2}$.

We note there is another way to write a word hypercube that may be easier to visualize in some cases. For example, when w=3 and d=5, the standard word hypercube is a 3x3x3x3x3 hypercube containing 243 letters, and 81 words, with each word consisting of 3 letters in a straight line. But many of those letters and words are constrained to be equal, so there are only 21 letters that can be chosen independently, and only 15 words that must be checked in the dictionary. If those 21 letters are written in this format:



then each 3-letter word will be L-shaped rather than a straight line. In this format, a word is the letter in a cell, followed by the letter in the cell immediately above it, followed by the cell immediately to the right. So the word starting at the lower-left consists of these three letters in this order: $W_{1,1,1,1,1}$, $W_{1,1,1,1,2}$, $W_{1,1,1,1,3}$. There are 15 such L-shaped words in this diagram. All 15 of those words are contained in the dictionary, if and only if this diagram represents a valid 3x3x3x3x3 word hypercube.

The validity of this format is clear. The letters that can be chosen independently are those that have their subscripts in sorted order. Since each subscript can be only a 1, 2, or 3 (for w=3), we can summarize the list of subscripts by giving only the number of 2's, the number of 3's, and the length of the list. In the above diagram, the number of 2's increases vertically, and the number of 3's increases horizontally.

This format also makes it easier to visualize what happens when the dimensionality d is changed. If d is decremented by one, then the top diagonal (purple) is deleted, and the first subscript (1) is deleted from all the remaining letters. This can be repeated. Each time d is decremented, one more diagonal is deleted, and another 1 subscript is deleted. Conversely, to increment d, first prepend a 1 to every list of subscripts, then add a new diagonal above the top one, with the subscripts following the same pattern as seen in the purple: all 2's at the top, with more 3's shifting into each cell as you move down, reaching all 3's at the bottom.

This pattern also continues for larger values of w. When w=3, it is a 2-dimensional triangle, as shown above. In general, it is a (w-1)-dimensional hyperpyramid. For w=3, a word is the letter in a cell, followed by the cells immediately adjacent to it in the positive horizontal and vertical directions. In general, a word is the letter in a cell followed by the cells immediately adjacent to it in the positive direction parallel to each of the (w-1) axes.

For a (*w,d*) cube, it is not difficult to show that the number of words *N(w,d)* required is given by

$$N(w,d) = \binom{w + d - 2}{d - 1}$$

where parentheses denote the binomial coefficient. Similarly, the number of letters *L(w,d)* is given by

$$L(w,d) = N(w,d+1)$$

WORD HYPERCUBES ARE NP-HARD

How hard is it to find a word hypercube of a given word size and dimensionality? Unfortunately for those who hope to discover new ones, it is NP-hard.

For this section, we assume the reader has a general familiarity with computational complexity theory. Readers who do not may skip this section, noting only that finding word hypercubes using a given word list is a member of a class of problems widely believed to require exponentially increasing time to solve as the word list gets larger.

Given a problem (w,d,D,A) where:

- w = the word length (a positive integer)
- d = dimensionality (a positive integer)
- A = an alphabet (set of symbols)
- D = a dictionary (a set of strings over A of length w)

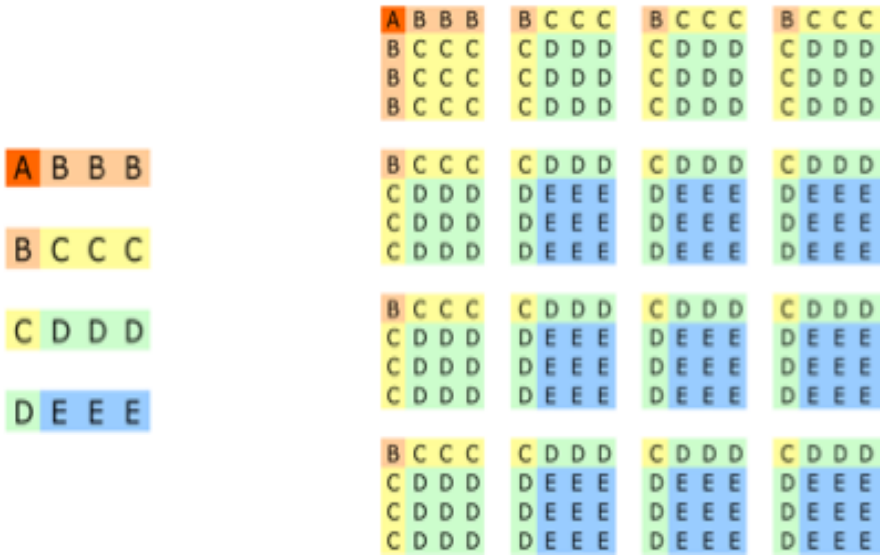
The Word Square Problem is the question: does there exist a word square of size w for dictionary D? This is the (w,2,D,A) problem. The Word Hypercube Problem is the general (w,d,D,A) problem in d dimensions. We believe the complexity of this problem was previously unknown, even for the d=2 Word Square Problem. Gary and Johnson [7] list the Crossword Puzzle Problem as NP-complete, but that problem lacks the symmetry requirements of the Word Square Problem, so it does not prove the Word Square Problem to be NP-complete. It is also unrelated to the higher-dimensional generalization.

Theorem: The Word Hypercube Problem is NP-hard

Proof:

The proof is by reduction from 3SAT [7]. We will first construct a dictionary that is guaranteed to have exactly one word-hypercube solution. Then we will modify that dictionary to embed an arbitrary 3SAT problem.

Consider the word hypercube problem with w=d=4 and a dictionary of four words D = {ABBB, BCCC, CDDD, DEEE}. The following shows one solution, with the dictionary on the left, and the solution on the right, with letters colored for clarity:



In the solution, the hypercube element at location (1,1,1,1) is the red "A" in the upper-left corner of the upper-left square, and the element at location (4,4,4,4) is the blue "E" in the lower-right corner of the lower-right square.

The dictionary word ABBB appears in the hypercube four times: twice in the upper-left square, once going down through the layers shown by the top row of squares, and once going down through the layers shown by the left column of squares. This word could not have been placed anywhere else. If it had been placed at another location, then the "A" would have appeared somewhere other than location (1,1,1,1), and so would have had to intersect some word at a position other than its first letter. But no word in the dictionary has an "A" in any position other than the first position. So this word could not have appeared anywhere else.

In fact, the solution shown is the only possible solution for this dictionary. If the word ABBB is part of the solution, then it will have to appear in the position shown. And then BCCC will have to appear in the position shown, because its leading "B" must intersect with a word that has a "B" in a position other than first, and ABBB is the only word that has that. And then CDDD and DEEE must appear as shown, by the same argument. So if ABBB is used at all, then the only solution will be the one shown.

On the other hand, what if ABBB is not used? Then suppose BCCC is used. Since no other word contains a "B", the BCCC will have to appear where ABBB is shown in the diagram. That forces CDDD to move to where BCCC is shown, and DEEE to move to where CDDD is shown. But then there is no word that can go in the positions where DEEE are shown. A word in that position would have to start with E. But there

are no words in the dictionary starting with E. So there cannot be a solution that contains BCCC without ABBB.

By a similar argument, there is no solution with CDDD and no BCCC or ABBB. And there is no solution with DEEE without the other words. Therefore, the solution shown is unique.

This generalizes in the obvious way to word lengths other than w=4. The words simply expand or contract by using more or fewer repetitions of the final letter. It also generalizes to dimensions other than d=4 in the obvious way. The dictionary will contain d words, with each word starting with the letter that ended the previous word, followed by repetitions of a new letter.

Now we will delete the first 3 words of the dictionary and replace them with a longer list of words. In the process, the letters B and C will be removed from the alphabet, and several subscripted versions of them will be added. The new dictionary is:

$$
\begin{aligned}
D=\{\ & AB_1B_2B_3\ldots B_{w-1},\\
& B_1C_0C_0C_0\ldots C_0,\\
& B_2C_0C_0C_0\ldots C_0,\\
& \ldots\\
& B_{w-1}C_0C_0C_0\ldots C_0,\\
& C_0DDD\ldots D,\\
& C_1DDD\ldots D,\\
& DEEE\ldots E\}
\end{aligned}
$$

In high dimensions, this could be a very long list, but it will still only be the first 3 words that were changed. This new dictionary again has only a single possible solution. The first square in that solution for the w=4 case now looks like this:

A	B_1	B_2	B_3
B_1	C_0	C_0	C_0
B_2	C_0	C_0	C_0
B_3	C_0	C_0	C_0

This is identical to before, except for the subscripts. And it is unique for the same reason as before. Plus the single word starting with A forces all the B words to be in a particular order.

Now consider the word $B_2C_0C_0C_0$ in the dictionary. It is forced to be on the third row and the third column in the solution. If the word $B_2C_0C_1C_0$ were added to the dictionary, it could also be placed on the third row and column, so there would now be two valid solutions. In fact, any combination of the zero subscripts in $B_2C_0C_0C_0$ could be replaced with ones, and if the resulting words were added to the dictionary, each would give a valid solution if it were placed on the third row and column. Similar words starting with B_1 or B_3 could also be added, which might generate additional solutions, if they could be chosen such that their intersecting C letters matched.

The following is the same square for a larger word length w, where an arbitrary rectangle has been chosen in the yellow region above the diagonal, and it has been outlined with a solid line, with its transpose outlined with a dotted line:

A	B_1	B_2	B_3	B_3	B_3	B_3	B_3	B_3	B_3
B_1	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0
B_2	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0
B_3	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0
B_4	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0
B_5	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0
B_6	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0
B_7	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0
B_8	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0
B_9	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0	C_0

Such a rectangle can be chosen arbitrarily, and then each row and column within that rectangle can be given a set of possible patterns by adding to the dictionary the words containing those patterns. Since the rectangle is above the diagonal, it will not overlap with its transpose, which simplifies the analysis.

Now it is possible to reduce 3SAT to the Word Hypercube problem of any desired dimensionality. Suppose we are given a 3SAT problem in conjunctive normal form with *v* variables and *n* clauses, and no repeated variables within a single clause. We can now generate a word hypercube problem of the desired dimensionality and with a word length long enough so that a rectangle with *v* rows and *n* columns can fit within it, as in the above figure.

Each row of the rectangle will correspond to one variable in the 3SAT problem. Each column of the rectangle will correspond to one clause in the 3SAT problem. The dictionary will contain 2 words for each row, and 7 words for each column

For each row in the rectangle, the dictionary has two words. One word has a C_0 for every position that falls within the rectangle. The other word is the same, except there is a C_1 in each column that corresponds to a 3SAT clause that contains that variable.

A 3SAT clause will contain 3 variables, each of which might be negated, so the clause will be satisfied by 7 of the 8 possible assignments of values to those variables. So for each column in the rectangle, the

dictionary will contain 7 words corresponding to the 7 satisfying assignments. Each word will have a C_1 in the position corresponding to each variable that is true in that assignment, and will have a C_0 for those variables that are false, and for the variables that don't appear in the clause at all.

Given this dictionary, it will be possible to construct a valid word-hypercube solution if and only if it is possible to make a valid word-square solution in the first square layer, which includes the first word in the dictionary. And it will be possible to make that word-square solution if and only if the 3SAT problem has a satisfying assignment. Thus 3SAT has been reduced to Word Hypercube, and so Word Hypercube is NP-hard.

Corollary: The Word Square Problem is NP-complete.

Proof:

The proof for the general Word Hypercube Problem works for any d>1, so it shows that even for d=2, it is still NP-hard. It is obvious that a Word Square Problem solution can be checked in polynomial time, and so is in NP. Therefore the Word Square Problem is NP-complete.

MAPPING THE SPACE OF WORD HYPERCUBES USING THE ENGLISH OPEN WORD LIST (EOWL)

Although finding word hypercubes is NP-Hard, most English words are sufficiently short and computers sufficiently fast that many parts of the (w,d) solution space can be mapped for a specific dictionary. For our results, we used the English Open Word List, or EOWL [5]. This list has the advantage of being in the public domain and therefore readily obtainable. It contains approximately 130,000 words, all under ten characters, with no words that contain hyphens, diacritical marks, or apostrophes.

For the EOWL and most values of w and d , our results show whether or not a word hypercube exists, and can give an example. For smaller values of d, we know all the solutions. Because the construction of a (w,d) hypercube requires at a minimum the existence of *w* hypercubes of dimensionality d-1, for larger values of d we can tell whether or not objects of a that dimensionality exist for some dictionary by merely inspecting the solution space of the dimension below. It is in the middle where the most mystery remains.

The results shown here were obtained using the programming language Python, run on the first author's computer or on the cloud at www.picloud.com. The use of word lists other than EOWL will of course produce different results. Running the code with larger word lists or word lists from different languages is one possible area of future work.

The results below shows what is known of the EOWL word hypercube space so far. In some cases, the number of hypercubes is known exactly. In other cases (based on our self-imposed limit of no more than one week of CPU time per problem) it is estimated, and in at least one case the existence of a solution is not yet known. We elaborate on some details in the paragraphs that follow.

w	dictionary size	d=2	d=3	d=4	d>4
3	1,083	132,454	37,375,325	22 x 10^9 (est.)	?
4	4,447	4,144,949	36 x 10^9 (est.)	427 x 10^{12} (est.)	?
5	9,636	10,724,619	610 x 10^6 (est.)	19 x 10^6 (est.)	?
6	16,219	1,652,918	22	0	0
7	22,923	9,744	0	0	0
8	26,265	0	0	0	0
9	25,626	0	0	0	0
10	22,679	0	0	0	0

Table 1: The EOWL Word Hypercube Space

For w=3 and w=4, the time requirements for d=4 and higher exceed our patience thresholds to enumerate the search space. However, the inclusion in the EOWL of word cycles of length w like {EAT,ATE,TEA} and {ANAN, NANA, ANAN, NANA} means that hypercubes of arbitrary dimensionality can be constructed for w=3 and w=4. Thus if we allow solutions that reuse words, we would expect these search spaces to be quite large. Intermittent computational runs with samples of the solution space suggest that for w={3,4}, the number of solutions does indeed grow rapidly for d=4 and higher.

For w=5, the computing requirements exceed our resources and patience to enumerate the search space beyond d=2. Partial runs with sampling suggest that the number of solutions for (5,3) is approximately thirty times that for (5,2). Continued partial computation for (5,4) suggests that the number of solutions falls off by about two orders of magnitude. We do not yet know if a (5,5) hypercube can be formed from words in the EOWL.[1]

We will say more about hypercubes with w<=5, d >=3 shortly.

Although it may seem that one must exhaustively search the entire space to determine whether or not a (w,d) hypercube exists, that is not necessarily true. If the space of all (w,d-1) and (w,d-2) cubes is known, and it is sufficiently small, that space can be examined to determine if it contains cubes that can be assembled into a (w,d) solution. This is done by attempting to place w (w,d-2) cubes in a w x w grid G such that a) G(i,j) = G(j,i) for all i,j between 1 and w, and b) all rows of G are (w,d-1) cubes. (Note that the first requirement means that the columns of G will be those same cubes). For larger values of d with completely enumerated (w,d-1) and (w,d-2) solution spaces, this is more efficient than exhaustive search of the (w,d) space.

For example, a 5-cube requires its first and second 5-square slices to share a word at their second and first positions, respectively. If all the 5-squares are known, and no two have this property, than no 5-

[1] Borgmann [2] claims the longest English word cycle is {ESTER, STERE, TERES, EREST, RESTE}. Any list containing these words can therefore be used to construct a 5-cube of arbitrary dimensionality. Of these five words, only ESTER and STERE are in the EOWL. The EOWL does contain the tantalizing {PESTO, ESTOP, STOPE, TOPES] partial cycle, but no word list known to the authors contains OPEST.

cubes exist. Clearly this is not the case for (5,3) cubes, but the technique generalizes to higher values of w and d where it can be more efficient. And of course in some cases the solution set for (w,d-1) is so small that the nonexistence of (w,d) cubes can be concluded by inspection.

Note that our results show no word squares for w > 7. Such objects are in fact known, but are drawn from different word lists [1,14]. The closest example that can be obtained from the EOWL is below, in which all but one letter was successfully placed:

```
D   I   S   S   U   A   D   E
I   M   P   I   N   G   E   S
S   P   A   N   *   E   S   S
S   I   N   F   O   N   I   A
U   N   *   O   S   T   L   Y
A   G   E   N   T   I   V   E
D   E   S   I   L   V   E   R
E   S   S   A   Y   E   R   S
```

The closest approximation to an EOWL 9-square places all but 5 letters. For the 10-square, the best attempts leave 12 letters unplaced.

RESTRICTING THE SOLUTION SPACE TO DIAGONAL HYPERCUBES

Because the solution space for w=3 and w=4 is so large, we consider making the problem a little more interesting by requiring one or both diagonals to be words. For higher dimensional objects, this requirement extends to all its lower dimensional slices. Unfortunately, this requirement is too stringent for d>=3, as no hypercubes meet it. For d=2, the solution space is as follows:

w	d=2
3	2083
4	7717
5	862
6	16
>= 7	0

Table 2: EOWL Dual Diagonal Hypercubes

If we relax the requirement so that only the main diagonal must be a word (from upper left to lower right in the square case with the obvious generality for d > 2), the space becomes populated for three dimensions, but not beyond:

w	d=2	d=3	d>=4
3	9,521	4,897	0
4	140,377	2,039	0
5	67,198	0	0
6	1,506	0	0
7	2	0	0

Table 3: EOWL Main Diagonal Hypercubes

SOME INTERESTING WORD HYPERCUBES

The EOWL solution space contains 22 (6,3) word hypercubes, grouped into two *families.* A family is a set of two or more word cubes closed under a set of single-letter substitutions. These families are given below:

Family of 2:

PRESTO	RUSHES	ESCORT	SHOVEL	TERETE	OSTLER
RUSHES	UNTAME	STALER	HALITE	EMETIN	SERENE
ESCORT	STALER	CAGILY	OLIVES	RELENT	TRYSTS
SHOVEL	HALITE	OLIVES	VIVERS	ETERNE	LESSEE
TERETE	EMETIN	RELENT	ETERNE	TINNER	ENTERA
OSTLER	SERENE	TRYSTS	LESSEE	ENTERA	RESEA1

1 = {L,T}

Family of 20:

1ACHES	ACHENE	CHARRS	HEREAT	ENRACE	SESTET
ACHENE	CYANIN	HAREEM	ENERVE	NIEVES	ENMESH
CHARRS	HAREEM	ARGALA	REARER	RELENT	SMARTY
HEREAT	ENERVE	REARER	ERRORS	AVERSE	TERSER
ENRACE	NIEVES	RELENT	AVERSE	CENS2R	ESTERS
SESTET	ENMESH	SMARTY	TERSER	ESTERS	THYRS3

1 = {C,L,N,R,T}

2 = {E,O}

3 = {E,I}

Previously, only one 6-cube from any word list was known to the authors [10], so we believe these are new.

The EOWL word list yields a family of two 7-squares where the main diagonal is also a word:

1	R	O	P	P	E	R
R	E	C	L	I	N	E
O	C	R	E	A	T	E
P	L	E	A	S	E	D
P	I	A	S	T	R	E
E	N	T	E	R	E	R
R	E	E	D	E	R	S

1 = {C,D}

8-squares with main diagonals as words are known [14], but require the use of a different word list.

AN UNUSUAL (3,4) HYPERCUBE

If we expand our list of 3-letter words slightly, we can construct a (3,4) word hypercube in which all main diagonals are words as well. The (3,3) slices of a such an object are as follows:

CUBE 1:

PRO	RHO	OOT	
RHO	HEP	OPE	(cube diagonal is PET)
OOT*	OPE	TET	

*The diagonal of this square is PHT, which appears in some word lists

CUBE 2:

RHO	HEP	OPE	
HEP	EEL	PLY	(cube diagonal is REE)
OPE	PLY	EYE	

CUBE 3:

OOT	OPE	TET	
OPE	PLY	EYE	(cube diagonal is OLE)
TET	EYE	TEE	

This object uses the words OOT, TET, PHT and OLE, which do not appear in the EOWL but do appear in other sources. The main diagonal of the entire (3,4) cube is the upper left back corner of cube 1 = 'P', the center letter of cube 2 = 'E', and the lower right front corner of cube 3 = 'E'. While regrettably scatological, the word is nonetheless included the EOWL.

RESTRICTING THE SOLUTION SPACE TO HYPERCUBES WITH UNIQUE WORD PLACEMENT

To reduce the solution space for shorter word lengths and make the problem more interesting, we may add the restriction that a word can only be placed once. This ensures that for any given word list, there is sufficiently high dimensionality beyond which no cubes exist. Our results and estimates for small d are shown below:

w	d=2	d=3	d=4	d>=5
3	132,187	35,774,512	2×10^9 (est.)	?
4	3,989,868	34×10^9 (est.)	121×10^{12} (est.)	?
5	9,509,258	571×10^6 (est.)	14×10^6 (est.)	?
6	1,308,120	22	0	0
7	9,744	0	0	0

Table 4: EOWL Single Word Placement Hypercubes

AN UNDECIDABLE PROBLEM

Returning to the problem as originally stated (in which the reuse of words is allowed), given a dictionary of w-letter words, it is natural to ask what dimensionalities of word hypercubes are possible. Clearly, a d=1 solution can be found: it's just a single word from the dictionary. If the dictionary contains a word that is just repetitions of a single letter, then solutions will exist for all positive d. If the dictionary looks like the first one that was constructed in the NP-completeness proof, then solutions will exist for all d up to a certain threshold, and for no d above that. So it is natural to ask the question: does a given dictionary allow solutions for all positive d, or only for d up to some threshold value? That question isn't merely NP-hard. It's actually undecidable. No computer program can be written that will correctly answer this question for all dictionaries, even if that program is given unlimited time and memory.

Theorem: The question of whether a given dictionary with words of length w allows for word hypercube solutions for all positive d is undecidable.

Proof:

The proof is by reduction from the Wang tiling problem [13].

A Wang tiling problem consists of a given, finite set of square tiles, with each edge of each tile colored a single color. Different edges can have different colors or the same color, but a single edge is assigned only one color. The problem is to tile the infinite plane with duplicates of these tiles, without rotating them, such that edges in contact are the same color. The question of whether this is possible for a given set of tiles is undecidable. The problem remains undecidable if the question is whether the infinite quarter-plane can be tiled (i.e., the first quadrant in a Cartesian system).

Given a particular set of n Wang tiles using m colors, we will construct a dictionary for the word hypercube problem, where the alphabet has 4n+m+3 letters, each word is w=3 letters long, the dictionary contains 8n+1 words. We will then show that this dictionary has solutions for all positive d, if and only if the given set of Wang tiles can tile the quarter plane.

For the word hypercube problem, the alphabet will contain one letter for each color used by the Wang tiles, plus the three letters {X,Y,Z}, plus the 4n letters $\{A_i, B_i, C_i, D_i\}$ for $1 \leq i \leq n$. The dictionary will contain the single, 3-letter word "XYZ", plus the 8n words given by the following, for all $1 \leq i \leq n$:

W_i X A_i

A_i Z B_i

B_i N_i E_i

Y W_i C_i

C_i A_i D_i

D_i B_i Y

Z_i C_i S_i

S_i D_i X

This defines a group of 8 words associated with each value of i. Each group together corresponds to the ith Wang tile. The four variables $\{N_i, S_i, E_i, W_i\}$, represent the four colors that are on the North, South, East, and West side of the ith Wang tile, respectively.

For the word hypercube problem for a given dimensionality d, a solution exists if and only if it is possible to fill in a triangular table of d+1 rows and columns, similar to the one in the previous section, such that every L-shape of 3 cells (a cell, the one above, the one to the right, in that order) corresponds to a word in the dictionary. For the problem of deciding whether there exist hypercubes for all positive d, this corresponds to expanding the triangle in the diagram forever. In other words, the problem is to choose letters for every cell in the infinite quarter plane (i.e., the first quadrant) such that every such L-shape contains a valid word from the dictionary.

Note that this problem cannot be solved using only the letters {X, Y, Z}, because there is only one word that uses only those letters, and it uses Y and Z but starts with X. So it will be necessary to use at least one word from the tile groups.

If at least one word from the tile group i is to be used, then notice the effect of the letters $\{A_i, B_i, C_i, D_i\}$, given how they are distributed among the 8 words in that group. These 4 letters act as "glue". Every

word in the group includes at least one of those 4 glue letters. If any word from that group is used, then at least one glue letter will appear in the table, and that will require that at least one word be used that starts with that letter, ends with that letter, and has that glue letter in the middle. And the same is true for any glue letters that appear in those words, and so on. It quickly becomes apparent that if even a single word in a tile group is used, then all 8 words in that group must be used, and they must be arranged exactly like this:

X	Z	N_i	X
W_i	A_i	B_i	E_i
Y	C_i	D_i	Y
X	Z	S_i	X

The 4 glue letters appear in the center, and ensure that all the letters in this diagram must be as shown. The X in the lower-left corner is forced by the fact that there is a Y above it and a Z to its right, and the only dictionary word ending in "YZ" is the word "XYZ". The X in the upper-right corner is forced by process of elimination, given the reasoning below. The rest of the letters are forced by the words in the tile group.

Since the above 4x4 pattern must appear anywhere a tiling group word is used from the dictionary, then the only way to tile the quadrant will be to tile it using copies of this pattern, where the rightmost column of one pattern overlaps the leftmost column of the pattern to its right, and the top row of one pattern overlaps the bottom row of the pattern above it. For example, a 7x7 region might be filled like this:

X	Z	N_1	X	Z	N_2	X
W_1	A_1	B_1	$E_1=W_2$	A_2	B_2	E_2
Y	C_1	D_1	Y	C_2	D_2	Y
X	Z	$S_1=N_3$	X	Z	$S_2=N_4$	X
W_3	A_3	B_3	$E_3=W_4$	A_4	B_4	E_4
Y	C_3	D_3	Y	C_4	D_4	Y
X	Z	S_3	X	Z	S_4	X

In this example, the upper-left region is filled with words from tile group 1, the upper right is 2, and the bottom is 3 and 4. Since the patterns are overlapping, this will only be a legal configuration if it happens to be the case that E1=W2, and S1=N3, and S2=N4, and E3=W4. Similarly, larger regions can be filled with letters only by laying them out in the same way that tiles would have covered a larger region in the Wang tiling problem.

In other words, a region (e.g., the infinite quarter plane quadrant) can be filled with letters that satisfy the constraints of the word hypercube problem, if and only if they are chosen to correspond to tiles that satisfy the constraints of the Wang tiling problem. Thus the two problems are equivalent, and if one is undecidable, then so is the other. The Wang tiling problem is undecidable, so the word hypercube problem is undecidable, and the proof is complete.

Corollary: The word hypercube problem remains undecidable, even if the word length w is constrained to be w=3.

Proof:

The above proof only used w=3, so this is sufficient.

Theorem: The word hypercube problem is decidable for w<3.

Proof:

Since w is a positive integer, this can only be w=1 or w=2. For w=1, a single dictionary word can solve the problem for all d by filling the quadrant with its single letter, so the problem is trivially decidable (since it's always true). For w=2, the dictionary words are all 2-letter words, and the problem can be found by discovering whether there exists a sequence of n words from the dictionary that form a cycle of the form $\{L_1L_2,\ L_2L_3,\ L_3L_4,\ \ldots,\ L_{n-1}L_n,\ L_nL_1\}$. There are solutions for all d if and only if such a cycle exists. And its existence can be found in polynomial time by constructing a finite directed graph and looking for directed cycles. Therefore, the w=2 case is decidable, too.

CONCLUSIONS

As before, let

- w = the word length (a positive integer)
- d = dimensionality (a positive integer)
- A = an alphabet (set of symbols)
- D = a dictionary (a set of strings over A of length w)

For fixed w, A and D, with unique word placement, some interesting questions include:

1) For what value of d does the size of the solution space begin to decrease?

2) What is the lowest value of d for which the solution space is known to be empty?

3) What is the highest value of d for which a (w,d) cube is known?

4) To what extent can the empty regions of Tables 1 and 4 be populated through the use of different dictionaries?

5) What are the effects of different hypersolids (non-symmetric, non-cubical, etc.)?

For symmetric hypercubes using the EOWL with w >= 6, the answers are known and were presented in previous sections. We show our current answers for w < 6 in the table below:

word length w	highest value of d for which (w,d) object found	# words placed for (w,d+1) object after 1 week CPU time
3	17	167/171
4	6	80/84
5	4	50/70

Table 5: Known Limits for EOWL Hypercubes

For w=3 with the EOWL, we have found solutions up to d=17. For d=18, we were able to place 167 out of 171 words after one week of computation time, so we suspect (3,18) hypercubes exist and perhaps higher dimensional objects as well. For w=4, we have found solutions up to d=6. For d=7, we were able to place 80 out of 84 words, so again we suspect objects of higher dimensionality exist, given that the search space is so large.

For w=5, the question is more problematic. We were only able to place 50 out of 70 words in a week-long search for a (5,5) hypercube. On the other hand, during that time we were only able to search an estimated 1/20th of one percent of the solution space. The existence of a (5,5) hypercube for EOWL, or indeed any word list, remains an open question. [2]

REFERENCES

1. Albert, E. (2012) "The Best 9X9 Square Yet," *Word Ways*: Vol. 24: Issue 4, Article 2
2. Borgmann, D., “Language on Vacation: An Olio of Orthographical Oddities”, Charles Scribner & Sons, 1965.
3. The 2012 University of Chicago Scavenger Hunt List: Available online at http://scavhunt.uchicago.edu/scavlist2012.pdf

[2] This work was sponsored in part by the Air Force Office of Scientific Research (AFOSR). This material is based on research sponsored by the United States Air Force Academy under agreement number FA7000-14-2-0009. The U.S. Gov- ernment is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the United States Air Force Academy or the U.S. Government.

4. Eckler, A. (2009) "Cubic Word Squares," *Word Ways*: Vol. 42: Issue 1, Article 4
5. EOWL: Available online at http://dreamsteep.com/projects/the-english-open-word-list.html
6. Francis, D. (1971) "From Square to Hyperhypercube," *Word Ways*: Vol. 4: Issue 3, Article 8
7. Garey, M. and Johnson, D., "Computers and Intractability: A Guide to the Theory of NP-Completeness", W.H. Freeman, 1979, ISBN 978-0716710455.
8. Gordon, L. (2012) "Designing a List for Word Squares," *Word Ways*: Vol. 29: Issue 3, Article 4
9. Knuth, D. E. (2012) "5X5X5 Word Cubes By Computer," *Word Ways*: Vol. 26: Issue 2, Article 12.
10. Grant, J. (1978) "Cubism Revisited," *Word Ways*: Vol. 11: Issue 3, Article 1
11. Grant, J. (1979) "More Word Cubes," *Word Ways*: Vol. 12: Issue 2, Article 3
12. Kon, R. (2009) "Solid and Hypersolid Forms," *Word Ways*: Vol. 42: Iss. 4, Article 14
13. Wang Tiling Problem: http://en.wikipedia.org/wiki/Wang_tile
14. Word Square: http://en.wikipedia.org/wiki/Word_square