Cruciform: Solving Crosswords with Natural Language Processing

Dragomir Radev[1], Rui Zhang[1], Steve Wilson[1], Derek Van Assche[1]
Henrique Spyra Gubert[2], Alisa Krivokapic[2], MeiXing Dong[1]
Chongruo Wu[1], Spruce Bondera[1], Luke Brandl[1], Jeremy Dohmann[3]
[1] University of Michigan
[2] Columbia University
[3] Harvard University
(radev@umich.edu, ryanzh@umich.edu, steverw@umich.edu, dvanassc@umich.edu
hs2807@columbia.edu, ak3533@columbia.edu, meixingd@umich.edu, chongruo@umich.edu,
spruceb@umich.edu, brandl@umich.edu, dohmann@college.harvard.edu)

Abstract

Crossword puzzles are popular word games that require not only a large vocabulary, but also a broad knowledge of topics. Answering each clue is a natural language task on its own as many clues contain nuances, puns, or counter-intuitive word definitions. Additionally, it can be extremely difficult to ascertain definitive answers without the constraints of the crossword grid itself. This task is challenging for both humans and computers. We describe here a new crossword solving system, Cruciform. We employ a group of natural language components, each of which returns a list of candidate words with scores when given a clue. These lists are used in conjunction with the fill intersections in the puzzle grid to formulate a constraint satisfaction problem, in a manner similar to the one used in the Dr. Fill system[4]. We describe the results of several of our experiments with the system.

1 Introduction

Crossword puzzles are popular word games that require not only a large vocabulary, but also a broad knowledge of topics. Answering each clue is a natural language task on its own as many clues contain nuances, puns, or counter-intuitive word definitions. Additionally, it can be extremely difficult to find definitive answers without the constraints of the crossword grid itself. Completing some of the more difficult crossword puzzles can be a formidable task for many humans. Working in a natural language domain makes the task similarly challenging for computers.

One early system designed to solve crossword puzzles, *Proverb* [6], uses a variety of modules that targeted different types of clues and combined them into a probabilistic list in order to place them into the puzzle. Later, *Dr. Fill* [4] treats the crossword problem as a probabilistic constraint satisfaction problem and uses various lexical features to rank the candidate answers. IBM's *Watson* [3] uses an enormous amount of computation power and engineering talent to successfully take on another difficult natural language game: *Jeopardy!*.

Compared with other Question Answering systems, crossword solving systems are generally more complex, because of the following properties of crossword puzzles: [7]:

- Clues are mostly formulated in a non-interrogative form.
- Clues can be voluntarily ambiguous and misleading.
- Topics of the questions are not limited to factoids.
- There is a unique and precise answer which is a word or compound word, while in QA, the answer is a sequence of words in order to be recognizable by humans.

There are, however, some aspects which make crossword puzzles more approachable. Crosswords have a bias towards shorter and more common words, as well as words with specific letters and specific bigrams. Another advantage is that the length of the answer is known.

We propose a new crossword solving system, *Cruciform*, which has a general structure similar to that of *Proverb* and *Dr. Fill*. We employ a group of natural language components that each return a list of candidate words with likelihood scores when given a clue. The candidate lists for a clue are merged to form a single list; this is done for every clue. Then, these lists are used in conjunction with the fill intersections in the puzzle grid to formulate a constraint satisfaction problem (CSP).

We performed all of our experiments on two months (June 2015 and July 2015) of New York Times puzzles to ensure reliable results. We show that filling in blanks using a vocabulary of answers and bigram reranking of generated solutions improves the baseline performance. Additionally, we propose a set of metrics to analyze each subcomponent of the overall crossword puzzle solver.

2 New York Times Crossword Puzzles

The New York Times (NYT) releases a new crossword puzzle daily[1]. Puzzles are submitted by puzzle composers, or *cruciverbalists*, to the editor of the daily puzzle. The puzzles progressively increase in difficulty throughout the week so Monday puzzles are the easiest and Saturday puzzles are the hardest, where each has a grid size of 15x15. The Sunday puzzle is roughly of the same difficulty as the Thursday puzzle, but is larger, typically with a size of 21x21. Each *fill*, or answer, is at least three letters long and all squares appear in exactly two words. Typically, the maximum word count for a Monday-Thursday puzzle is 78 words, 72 for a Friday or Saturday puzzle, and 140 words for a Sunday puzzle.

The NYT puzzles follow many additional set conventions. When a clue is an abbreviation or is tagged with with "abbr.", the fill is also an abbreviation. If a clue is phrased as a question and ends in a question mark, then the answer is a play on words. Answers that are in non-English languages will have a clue that is either tagged with the language (ex. "Fr." for French) or has a word from that language (e.g. "Month for Marcel" for "mai"). Clues and fills always match in part of speech, tense, number, and degree. The fill can consist of multiple words and will never appear in the clue corresponding to it. The first letter of a clue is always capitalized; this is sometimes used to introduce an ambiguity since there could be a significant difference between a word and the proper noun version of that word.

Some puzzles are rebus puzzles, where some squares have to be filled with symbols, multiple letters, or a word. For instance, a rebus puzzle by Jeff Chen required the letters "pH" to be written together in single squares in fills such as "t/r/i/u/m/pH" and "s/o/pH/o/c/l/e/s." These puzzles are often unique and require special rules to solve, so they are excluded from our analysis.

Beyond these constraints, there are also a number of more qualitative elements that make up a quality crossword puzzle. Proper names that lack specific distinction (e.g. "girl's name," "boy's name") are generally avoided. Letter intersections where both words are unlikely to be solved by most readers are also discouraged.

2.1 Example

An example puzzle by Peter Gordon (edited by Will Shortz) is shown in Figure 1.

A single word can have clues that reference many different senses and usages of a word. Even for one word sense, different clues can tie that sense to vastly different contexts. For example, "ear" can be clued as "Musician's asset", "Van Gogh had one later in life", or "Corn unit."

Sample clues for "TREE", "NOAH", "ASTRO", and "EAR".

TREE

[1] http://www.nytimes.com/crosswords/

- It has bark but no bite
- Its bark is silent
- Branch location
- Fort locale
- Type of house
- House for kids
- Lineage display
- Every family has one
- Family chart
- Cobbler's need
- Shoe stretcher
- It leaves in the spring
- Where to get dates
- Site of many a cat rescue
- Dendrophobe's fear
- Forbidden fruit source
- Ring holder
- Leaves home

NOAH

- Flood survivor
- Ararat lander
- Rider of the lost ark
- Biblical helmsman
- Guy who believed in "take two"
- Noted couples protector
- Early matchmaker
- Captain of a famous cruise for couples
- Life preserver
- Person known for double takes
- First name in lexicography
- One of the Websters
- Actor Wyle

ASTRO

- Houston Colt 45 today
- Enron Field player, once
- Houston player
- Texas leaguer
- Nolan Ryan, notably
- 2005 world series participant
- Player under a dome
- Pirate battler, at times
- Cartoon dog
- Space age hound
- Jetson canine
- Animated pooch
- Elroy's pet
- Bygone Chevrolet van

EAR

- The sense organ for hearing and equilibrium
- Musician's asset
- Teacup handle
- Corn serving
- Spike
- Elephant's floppy feature
- Van Gogh had one later in life
- Q-Tip target
- Hole in the head?

3 Related Work

3.1 The Dr. Fill System

The Dr. Fill system [4] starts with producing candidate answers for each clue. Candidate answers are extracted from various resources such as crossword databases, dictionaries, WordNet, and Wikipedia:

- A crossword database from various publishers.
- A small, basic dictionary containing common words.
- An extensive dictionary with manually generated merit scores, where words with a higher score are considered good fill for crosswords. For example, "BUZZ LIGHTYEAR" is considered excellent fill. It is a lively word, uses uncommon letters, and has positive connotations.

Wednesday, June 24, 2015 **The New York Times** Ian Livengood and J.A.S.A. Crossword Class / Edited by Will Shortz

ACROSS

1 "Rock and Roll All Nite" band
5 Crime chief
9 Region known for its black tea
14 Quechua speaker
15 Pike, e.g.
16 Big bang material, informally
17 Luxury hotel overlooking Central Park
19 Some airport transports
20 Like some cheaper tuition
21 Weak
22 Not yet available at press time, for short
23 First chairman of the Joint Chiefs of Staff, 1949
25 Labor Day deliveries
27 ___ bran muffin
28 Exam-administering org.
29 Hubbub
30 Red stone
33 Constellation visible in Melbourne and Sydney
38 Any of three author sisters
39 "Fine by me"
41 ___ deviation: Abbr.
44 Swiss canton
45 To no purpose
47 Flier over Tiananmen Square
51 Bo Derek, in a 1979 film
52 Election do-over
53 Richard who won an Emmy, Grammy, Oscar, Tony and Pulitzer
55 Pennsylvania Dutch speakers
56 What 17-, 23-, 33- and 47-Across each have
58 Places to stand and deliver?
59 Bambi and others
60 Feature of a big cake
61 "I Hated, Hated, Hated This Movie" author
62 Tosses in
63 Mönch and Eiger, for two

DOWN

1 Baby fox
2 Puts the brakes on
3 Plot outline
4 Drains, as energy
5 Wipes the floor with
6 Vessel with many branches
7 Kitchen doohickey
8 Keats's "To Autumn," e.g.
9 "B.C." animal that goes ZOT!
10 Fathered
11 35-Down quarters
12 Small sea projection
13 Shuffles (along)
18 Classic Langston Hughes poem
21 Swiss money

22 Common waiting area distractions
24 Shouldered
26 Chow
30 Robin Williams voiced one in "Aladdin"
31 ___ studio
32 Horn sound
34 "I agree"
35 Derby hopeful
36 Slippery slope?
37 Worker on commission
40 Nikkei 225 unit
41 Predicament
42 What a 5-Across is a boss in
43 What "/" means in math class
45 Sound control knobs
46 Popular sheepskin boots
48 Subordinate's refusal
49 Mentally pooped
50 Gave a glowing review, say
54 "At Last" singer James
56 "Protecting and promoting your health" org.
57 Many honorees at M.L.B.'s Old-Timers' Day

Figure 1: A New York Times puzzle.

- Word roots and synonyms from WordNet.
- Information like page titles Wikipedia.

Each word of the appropriate length is scored by the following five criteria: 1) A match for the clue 2) Part of speech analysis 3) Crossword Merit 4) Abbreviation 5) Fill-in-the-blank. The final score is a weighted sum of the five criteria, where weights are tuned on a development set of puzzles.

Dr. Fill is evaluated (only) on the seven puzzles from the 2010 American Crossword Puzzle Tournament (ACPT) using ACPT's scoring rules to calculate a performance metric. However, it is not completely fair to use the ACPT score to compare their system with the other human solvers because the rules give bonus points for each minute of time remaining if a competitor finishes early. Since the rules were set for human solvers, we can expect that an automated system would finish solving puzzles much faster than humans and gain many extra points this way.

Since quantitative results on other puzzles are not provided and the system is not publicly available, we are unable to compare directly against the Dr. Fill system.

3.1.1 CSP Searching Heuristics

After generating the candidate answers, Dr. Fill employs the following heuristic to solve the underlying CSP.

- Value Selection: Prefer choices not only work well for the word slot in question, but also minimally increase the cost of the crossing clues. It is important to note that if Dr.Fill simply chooses the best candidate to fill, the performance drops significantly.
- Variable Selection: Pick the variable where the difference between the best candidate and the second best candidate is maximized.
- Limited Discrepancy Search (LDS): The search maintains a list of discarded value choices. Each element of the list is a pair of (v, x) indicating the value v should not be proposed for x. Discarded elements are not considered in the Value and Variable Selection heuristics and remain discarded in the rest of the search. For additional details, see [4].

The overall CSP algorithm in Dr. Fill is outlined in Algorithm 1: Given a crossword puzzle C, a fixed discrepancy limit n, a partial solution S, the best solution so far B, and previously pitched assignments P, solve(C, S, B, n, P) gives the best solution extending S with at most n discrepancies.

Algorithm 1 Dr. Fill CSP Algorithm

procedure SOLVE(C, S, B, n, P)
  if $cost(S) \geq cost(B)$ **then return** B
  if S assigns a value to every variable **then return** S
  $v \leftarrow$ a variable unassigned by S
  $d \leftarrow$ an element of v's domain such that $(v, d) \notin P$
  $S' \leftarrow S \cup (v = d)$
  $C' \leftarrow$ propagate(C,S')
  if propagation succeeded **then**
    $B \leftarrow$ solve(C', S', B, n, P)
  if $|P| < n$ **then**
    $B \leftarrow$ solve(C, S, B, n, $P \cup (v, d)$)

3.2 Proverb

Proverb [5, 6] uses thirty expert modules. Given a clue, each module produces a list of candidate answers of any length and numerical scores representing the confidence level for each word. The system also contains several modules to deal with uncommon answers that may not appear in the crossword database such as multi-word answers . Our approach is similar in that we also use a combination of expert modules.

3.2.1 Merging Lists and Grid Filling

The merging process is controlled by three parameters: scale(m), length-scale(m) and spread(m), where m refers to an expert module. For each module the weight of each candidate is adjusted by raising its power to spread(m) and then regularized, and the confidence of each module is multiplied by scale(m) and length-scale(m)$^{\text{targetlength}}$. The merger then combines all modules' candidates by summing together their probabilities weighted by the adjusted confidence and normalizing the sum to one. These parameters are optimized by hill-climbing with the objective function as the average log probability assigned to the correct targets.

3.3 WebCrow and SACRY

Webcrow [2] draws candidate words from web documents and previously solved crosswords. The candidate words are given confidence scores by various modules and combined into a solution using a CSP version of weighted A* search.

3.3.1 Candidate Answers Generation

WebCrow system draws its candidates primarily from web documents and previously solved crosswords. Using a novel Web Search Module (WSM), the system extracts potential words from web documents and ranks them using various filters. After parsing the document and identifying an unweighted list of candidate words, they are passed through both a statistical filter and a morphological filter. The statistical filter determines a score based on the distance between the candidate words and the search query. The morphological filter is composed of two parts, one of which attempts to determine the morphological class of each word in the document, while the other component identifies the most likely morphological classes of the clue's answer. The words are also ranked by a fairly standard crossword database module, a hard coded rule based module, a dictionary, and other similar modules. These scores are then merged additively by word and weighted by the confidence of each module. The grid is then filled by a CSP version of WA*.

The WebCrow system is evaluated on Italian crossword puzzles and not on English language puzzles. The structure and methodology for designing crossword puzzles varies a lot from one language to another. For instance, Italian puzzles can have letters that don't participate in crosses and include two letter fills that aren't actual words. Without running Webcrow on English crossword puzzles, we cannot determine how well the system would generalize to these puzzles or make a direct comparison with our system.

3.4 SACRY

SACRY is another crossword solving system introduced in [1] that introduces a new reranking module into the WebCrow system.

3.4.1 Reranking

SACRY attempts to relate various clues to each other using kernels in hopes that some higher level features of the clues can help to relate similar clues to a similar group of answers. Each answer is modeled by the set of clues associated with it, and the paper describes a several features which summarize this information. One of these features takes basic statistics the feature values of all such clues as determined by the reranking modules, as well as the frequency of the answer in the crossword database. The occurrences of each answer instance are modeled by a vector the size of the clue list with the values corresponding to the position in each clue's candidate list or zero if the word is not present. Additionally, the system attempts to determine the similarity between answer candidates and input clues using features based on word embeddings. These features are determined by (i) the similarity between the pair of clues, (ii) the target clue and the candidate answer and (iii) the candidate clue and the candidate answer. The end result is a reranked and aggregated list which can be input into WebCrow's CSP solver after a logistic regression step.

4 Methodology used in the Cruciform system

Given a crossword puzzle to solve, our system first uses different components (Section 4.1) to generate candidate answers for each clue. After merging the answers from different components, we compile a list of candidate answers along with confidence scores. These answers are then used to formulate a CSP solved by the algorithm similar to Dr.Fill (Section 4.2.2). Specifically, the CSP algorithm uses a variable selection heuristic by choosing the variable with the maximum confidence score difference between the best candidate and the second best candidate and then choosing that variable's best candidate following the value selection heuristic. To introduce some variations, the CSP algorithm also searches with non-best values with a predefined depth limit. The CSP search procedure generates a number of solutions (typically around 20) for the whole puzzle. We then refine each solution with post-processing steps including filling in the blanks and bigram probability reranking (Section 4.3).

4.1 Component Mechanism

Since many clues follow a predetermined pattern, it can be advantageous to prepare specialized programs that are crafted for the purpose of providing answers to certain types of clues. By setting up the system in a modular fashion, more components can easily be added at any time, which can continually increase the overall performance of the system. The task of each component is to recognize a specific clue type, understand the clue in some way, and provide an (optionally) ranked list of candidate answers that are the correct length to be placed into the puzzle.

The input to each component consists of a list of all clues in a puzzle along with the clue ID (e.g., 1A, 2D, 3A...) and the length of the correct answer, which is inferred from the puzzle grid layout. The output of each component is a list of candidate answers for each clue, designated with the corresponding clue ID as well as a score indicating the quality of the answer produced. The clue types that the system handles along with the approaches used by each component operates are described below in detail.

4.1.1 Lucene Components

The Lucene components are used as the baseline components that can handle any clue pattern in a generic way, accomplished by providing an interface to the Lucene indexing system.[2] Clues are converted into queries that are used to search through datasets of known clues as well as Wikipedia titles. The top results from each data source are compiled, and the entries of the wrong length are removed from the list of candidate answers. Lucene automatically generates a score for each result, which is returned along with the list of candidate answers. The three components we consider are CWG, OTSYS, and WIKI which search through the text of the CWG, OTSYS, and Wikipedia datasets detailed in Section 5.

4.1.2 Specialized NLP Components

We initially incorporated components that focused on very specific tasks such as identifying missing last names (ex. Clue: Titanic actor Billy (4) Fill: ZANE), missing first names (ex. Clue: "Wrecking Ball" singer Cyrus (5) Fill: MILEY), world capitals, acronyms, and other tasks. Unfortunately, these components caused substantial time increases for modest performance improvements. For example, adding components focused on first names, last names, state capitals, and country capitals added less than 1% to the average accuracy for June 2015 puzzles, at the cost of a time increase from 82.4 seconds to 494 seconds per puzzle. As such, we decided to exclude them from our final system.

Different NLP components with examples:

- Missing last name: Titanic actor Billy (4) ZANE, James who wrote "A Death in the Family" (4) AGEE
- Missing first name: "Wrecking Ball" singer Cyrus (5) MILEY, Stuntman Knievel (4) EVEL

[2] https://lucene.apache.org/core/

- Roman years: Year John Dryden died (4) MDCC, Early third-century year (4) CCIV
- Common foreign words: "Thank you," in Hawaii (6) MAHALO, When the day's done, to Denis (4) NUIT
- Prefix: Puncture preceder (3) ACU, Intro to physics? (4) META
- Suffix: Suffix with miss and dismiss (3) IVE, Fox tail? (4) TROT
- Direction: New Orleans-to-Detroit dir. (3) NNE
- World capitals: Samoa's capital (4) APIA
- Brands: Hyundai and Kia (5) AUTOS, Honda model (6) ACCORD
- Competitors: Marriott alternative (4) OMNI, Reebok alternative (4) NIKE
- Acronyms: Canoodling in a restaurant, e.g. (abbr.) (3) PDA, Chemists' org (3) AIC
- Hypernyms: Platinum, for example (5) METAL, Cronus and Hyperion (6) TITANS
- Synonyms: Overshoot, say (4) MISS, Gait (4) PACE
- Roles: 1963 Elizabeth Taylor role (9) CLEOPATRA, Tom Cruise's 'Risky Business' co-star (15) REBECCADEMORNAY
- Cities: Largest city in Nebraska (5) OMAHA
- Works of art: "Owner of a Lonely Heart" band (3) YES, Tom Cruise film set in Chicago (13) RISKY-BUSINESS
- Partners and counterparts: Gentleman's partner (4) LADY, Bull's counterpart (4) BEAR
- Dictionary definitions: Coffee dispenser (3) URN

4.2 Solution Generation

4.2.1 CSP

We formulate this task as a Constraint Satisfaction Problem (CSP). A CSP instance can formally be defined with the triple (X, D, C) [8]. Here, X is the set of variables $\{X_1, X_2, ..., X_n\}$, D is the set of domains for each variable, $\{D_1, D_2, ..., D_n\}$, where each $D_i = \{v_1, v_2, ..., v_n\}$ specifies allowable values for the corresponding variable X_i, and C is the set of constraints between variables. Each constraint is a tuple $(scope, rel)$, where $scope$ contains a list of the variables affected by the constraints, and rel describes the restrictions enforced by the constraint.

In the case of a crossword puzzle, let X be the set of entries in the puzzle that must be filled, where each entry X_i has exactly one clue and one correct answer. Both the length of the correct answer, $\ell(X_i)$, and the constraints, C, can be inferred from examining the puzzle grid layout. Each C_i defines an intersection between two answers ($scope$) in the puzzle, one in the "across" direction and the other in the " down" direction, where rel states that either 1) the letter in the square of the intersection must be the same, or 2) at least one of the two variables currently has a "blank" square in the location of the intersection. Finally, D is generated by aggregating the output of all components and removing any values from D_i with length not equal to $\ell(X_i)$ for a given i. Figure 2 illustrates such an example.

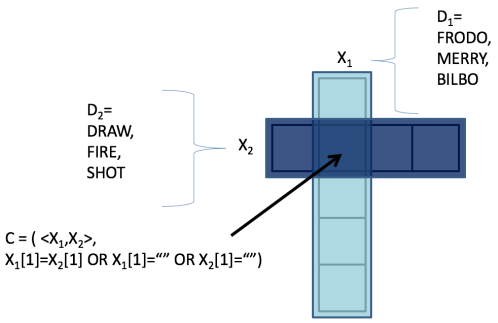

Figure 2: Crossword as a Constraint Satisfaction Problem

4.2.2 Filling in the Grid

Initially, all entries are set to "-", which indicates missing values. This means our CSP is technically solved from the beginning. Instead of looking for a solution that just fulfills the constraints, we search for one that maximizes the likelihood of the words. Every candidate word has some score reflecting how likely it is for the given clue. These are provided by the NLP components.

For the algorithm to choose the next variable, we employ the following variable selection heuristic: choose the variable that has the largest difference between the confidence scores of the best and second best candidates. This difference indicates how confident we are in filling that variable. For our value selection heuristic, we choose the candidate with the highest score. However, the best candidate is not always the correct answer and always choosing the candidate with the highest score can lead to poor solutions. To work around this problem, we also implemented Limited Discrepancy Search, where "discrepancy" is defined as the number of times that the heuristic is violated. We use LDS with a value selection heuristic where non-best candidates are also chosen. This search procedure introduces variations of solutions.

After CSP, we have a list of possible solutions to the crossword puzzle, each of which has a score by summing up the candidate scores for each filled word.

4.3 Post-processing

We employ the following two post-processing steps.

4.3.1 Filling in the blanks

In many cases, single letters are missing from the solution. For each solution, we perform the following Algorithm 2. Here, the function SORT($Entries$) will sort the list of entries with missing letters by the number of letters missing. Entries with fewer missing letters will be filled first. The function MATCH(e) returns a word that matches the length and the existing letters of entry, or None if none are found.

Algorithm 2 Fill in the blank

```
procedure FILLBLANK
    Entries ← a list of incomplete entries
    SortedEntries ← SORT(Entries)
    for e in SortedEnries do
        w ← MATCH(e)
        if w is not None then
            e ← w
```

	CWG	OTSYS
Number of words	174,648	251,595
Number of clues	1,692,482	4,325,828

Table 1: Data Statistics

Avg % in database	Clues	Answers	C/A pairs
April 2008	86.7%	90.3%	80.4%
May 2009	86.6%	89.7%	79.8%
June 2015	38.6%	84.8%	28.2%
July 2015	40.3%	84.7%	28.6%
January 2016	37.3%	84.6%	25.7%

Table 2: Average Percentage of Coverage in the Database for Select Months

4.3.2 Bigram Probability Reranking

Lastly, we perform bigram probability reranking over the possible solutions. We first build a bigram language model based on the clue data set described in Section 5.1. Then, we calculate the bigram probability of each solution after filling in the blank. The score of the solution is then rescaled by the bigram probability. We pick the solution with the highest score.

5 Datasets

5.1 Crossword Clues

We use puzzles from Crossword Giant (CWG),[3] an online crossword resource with clues and answers, and a dataset of clues collected by Matthew Ginsberg (OTSYS).[4] CWG and OTSYS were downloaded around May 2013 and March 2014, respectively.

Statistics including the number of words and clues for each data set are summarized in Table 1. The word with the most clues is "EAR" with 1,808 different accompanying clues, followed by "SEA" (1,707), "TREE" (1,578), "ERA" (1,578), and "ARIA" (1,428).

5.2 Wikipedia

We collect the title and the first sentence from Wikipedia pages from 2013.

5.3 Dataset Coverage

Our main training dataset used by the Lucene components consists of the clues and answers in CWG and OTSYS. We examine the overlap of clues and answers in the database and NYT clues and answers. If a clue shows up in a puzzle being evaluated and is also in the database, then it is included in the average percentage of clues in the database; this is also done for answers. If a puzzle clue and its corresponding answer both are in the database, then it is included in the average percentage of clue/answer pairs in the database. The percentage of clues and answers from the puzzles of the specified month that are found in the database tells us how much of the puzzle we have seen before.

In Table 2, we randomly pick five months of New York Time Puzzles (April 2008, May 2009, June 2015, July 2015, January 2016) and calculate overlaps for each month. As we can see, the percentage of clues previously seen decreases over time while the percentage of answers previously seen stays about the same. This implies that although our database contains many clues and fills, there are always new clues being

[3] http://crosswordgiant.com

[4] http://www.otsys.com/clue/

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	Average
Clues	50.30	42.46	38.17	34.39	33.62	24.88	40.58	37.77
Answers	90.58	90.92	86.26	83.09	81.26	75.77	82.84	84.30
C/A pairs	42.74	37.02	29.21	20.91	19.93	12.22	26.62	26.95
Clues	48.67	43.40	44.03	42.10	35.30	27.80	41.45	40.39
Answers	87.02	87.14	87.81	82.12	83.85	80.50	83.95	84.63
C/A pairs	40.87	33.28	35.93	26.28	20.41	12.27	31.05	28.58

Table 3: Average Percentage of Coverage in the Database on NYT June 2015 and July 2015

written for the same words. If our system can do well on newer puzzles, then the system is generalizable to unseen clues.

For June 2015 and July 2015, we also zoom in by grouping puzzles by weekdays, as summarized in Table 3. We see that the more difficult and the larger puzzles (e.g. Friday, Saturday, and Sunday puzzles) also tend to have more unseen clues and answers than the easier puzzles from other days of the week. Being able to do well on these puzzles requires generalization as well.

6 Experimental Results

Our test set is composed of the 61 crossword puzzles from June and July of 2015. In this section, we describe the baseline system based on Dr.Fill [4], and show that our post-processing (fill in blank and bigram reranking) and additional component based on Wikipedia consistently improve the accuracy of the solver.

6.1 Baseline

We consider the following setting as our baseline, which matches the system of Dr.Fill [4] as much as possible. We use two Lucene components based on our crossword datasets, CWG Lucene Component and OTSYS Lucene Component, to retrieve candidate answers. We replicate Dr.Fill's CSP algorithm with LDS depth limit of 4.

6.2 System Evaluation

Finally, to evaluate the performance of the whole system, we use the percentage of correct squares filled by the system compared with golden standards.

6.2.1 Post-processing

We evaluate our post-processing steps by measuring the performance of our baseline system with and without the two post-processing steps. The results are in Table 4. Our baseline system uses only the CWG and OTSYS Lucene components. Fill and Rerank refer to the Fill-in-the-Blank and Bigram Reranking steps respectively.

We see that using the Fill-in-the-Blank step consistently improves over no post-processing for all days. Bigram Rerank also achieves improvement on average, and it can help Friday and Saturday puzzles by a large margin.

6.2.2 Additional Lucene Component

We add the use of the Wikipedia Lucene component (WIKI) to the baseline system that uses only the CWG and OTSYS Lucene components. As we can see from the results in Table 4, the Wikipedia Lucene component give large improvements for the Wednesday, Saturday and Sunday puzzles. However, it decreases

the performance for Thursday and Friday puzzles. The best accuracy for both June 2015 and July 2015 is achieved when we combine two post-processing steps with WIKI.

7 Analysis and Discussion

Evaluating the output of a crossword puzzle solver is straightforward: we count how many words or squares are filled correctly as given by the solution. However, crossword puzzle solvers often involve multiple stages and a single final accuracy cannot quantify the contribution of each stage. Therefore, intermediate evaluation metrics are crucial to understand different pieces of the system and point out meaningful directions for future improvements. In this section, we propose several evaluation metrics for individual components, for CSP algorithms, and for the whole system.

7.1 Individual Component Evaluation

All components are run and evaluated individually using the the following metrics.

- Mean Reciprocal Answer Rank (MRAR).

$$MRAR = \frac{1}{|C'|} \sum_{i \in C'}^{|C'|} \frac{1}{r_i}$$

  where C' is set of queries for which the component gives the correct answers, and r_i is the rank of the i-th answer.

- Average Precision (AP).

$$AP = \frac{1}{N} \sum_{i}^{N} \begin{cases} 1 & \text{if } a_i \in C_i \\ 0 & \text{otherwise} \end{cases}$$

  For i-th clue, a_i is the correct answer, and C_i is the list of candidate answers produced by the component.

- Attempt Ratio (AR)

$$AR = \frac{1}{N} \sum_{i}^{N} \begin{cases} 1 & \text{if answered} \\ 0 & \text{otherwise} \end{cases}$$

A good component should be able to achieve a relatively high MRAR and AP. A high MRAR implies that the component gives helpful results to the solver and will not mislead it by giving low scores to valuable answers. A high precision means that there is a good chance the solver actually has the correct answer somewhere in its candidate lists, which is a requirement in order for it to produce a correct solution. Attempted Ratio (AR) quantifies the coverage ability of the component.

Settings	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	Average
CWG+OTSYS	97.17	92.85	83.40	75.35	64.88	54.87	80.73	79.68
+Fill	**98.64**	93.92	83.92	75.53	65.77	55.90	82.77	80.87
+Fill+Rerank	**98.64**	**94.98**	85.23	**77.11**	**66.41**	58.97	81.00	81.38
+Fill+Rerank+WIKI	98.32	94.77	**89.54**	72.89	58.62	**62.69**	**82.97**	**81.39**
CWG+OTSYS	98.14	94.77	91.41	71.24	59.03	50.46	78.22	77.39
+Fill	98.94	**96.65**	92.15	72.28	60.47	51.24	79.38	78.51
+Fill+Rerank	98.54	96.25	**94.14**	70.60	60.57	**56.08**	82.81	79.88
+Fill+Rerank+WIKI	**99.47**	96.38	93.93	**73.70**	**62.53**	53.73	**83.49**	**79.95**

Table 4: Square Accuracy on NYT June 2015(upper) and July 2015(lower)

	Lucene CWG			Lucene OTSYS			Lucene WIKI		
	MRAR	AP	AR	MRAR	AR	AP	MRAR	AR	AP
Monday	0.68	0.89	0.96	0.73	0.92	0.96	0.07	0.17	0.95
Tuesday	0.60	0.87	0.96	0.61	0.89	0.96	0.05	0.11	0.95
Wednesday	0.55	0.81	0.97	0.60	0.86	0.98	0.04	0.13	0.95
Thursday	0.42	0.73	0.97	0.49	0.78	0.97	0.04	0.09	0.96
Friday	0.39	0.73	0.97	0.43	0.76	0.98	0.04	0.10	0.97
Saturday	0.29	0.59	0.96	0.32	0.64	0.96	0.04	0.11	0.96
Sunday	0.47	0.78	0.95	0.52	0.82	0.95	0.04	0.11	0.95
Average	0.50	0.78	0.96	0.54	0.82	0.97	0.05	0.12	0.96

Table 5: Mean Reciprocal Answer Rank (MRAR), Average Precision (AP), Attempt Ratio (AR) of Lucene Components on NYT June 2015

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	Average
$FromComponents$	0.93	0.88	0.87	0.79	0.78	0.66	0.81	0.82
$AverageRank$	11.52	12.77	10.15	27.47	22.66	24.07	21.23	17.53
$IntoCSP$	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
$IntoSolution$	1.00	0.95	0.90	0.72	0.51	0.65	0.83	0.83
$FromComponents$	0.93	0.88	0.87	0.79	0.72	0.68	0.83	0.82
$AverageRank$	8.01	10.55	8.47	20.56	19.57	43.11	15.65	16.19
$IntoCSP$	1.00	1.00	1.00	0.99	0.99	0.96	0.99	0.99
$IntoSolution$	0.99	0.96	0.92	0.70	0.60	0.50	0.78	0.80

Table 6: CSP Evaluation on June 2015(upper half) and July 2015(lower half)

Table 5 lists evaluation results for three Lucene according to the above-mentioned three metrics. The evaluation is obtained from NYT puzzles from June 2015. As we can see, Lucene components achieve high attempt ratios around 97%, indicating that it attempts to answer most clues. Lucene CWG and OTSYS have high MRAR and AP, while Lucene Wiki has much lower numbers.

7.2 CSP Evaluation

To examine the CSP system portion specifically, we compute the following performance statistics:

- $FromComponents$ - This is the percentage of clues where the correct word is present in at least one of the component candidate lists.
- $AverageRank$ - This is the average rank of the correct word in the final candidate list for CSP if it was produced for a clue.
- $IntoCSP$ - This is the percentage of correct words presented by the components that are placed into the final candidate list for CSP.
- $IntoSolution$- This is the percentage of correct words in the candidate list for CSP that are placed into the solution by CSP.

Here, $FromComponents$ provides an overview of the quality of candidate answers generated by different components. $AverageRank$ and $IntoCSP$ quantify the input quality of CSP, while $IntoSolution$ evaluates the quality of CSP algorithm itself. A good CSP algorithm will pick out correct answers from the input candidate lists, resulting in a small gap between $IntoCSP$ and $IntoSolution$.

Table 6 summarizes our evaluation of the CSP algorithm's performance on the NYT June 2015 and July 2015 puzzles. For Monday, Tuesday, and Wednesday puzzles, the CSP algorithm demonstrates strong performance even though the Average Rank is around 10. However, as the difficulty of puzzles increases for the following weekdays, the Average Rank surges to 25, which lowers the CSP algorithm's performance.

8 Web Application

We have constructed a web interface for our puzzle solver.[5] As shown in Figure 3, one can select a puzzle to be solved from all NYT puzzles between 1999 and 2015. One can also choose which candidate generation components will be used by the solver. When a puzzle is loaded, both the empty and correctly filled grids are displayed, along with all of the clues. The empty grid is filled with the generated solution after the puzzle has been solved. Evaluation results such as square and word accuracy are also displayed. Figure 4 fives an output example for our interface.

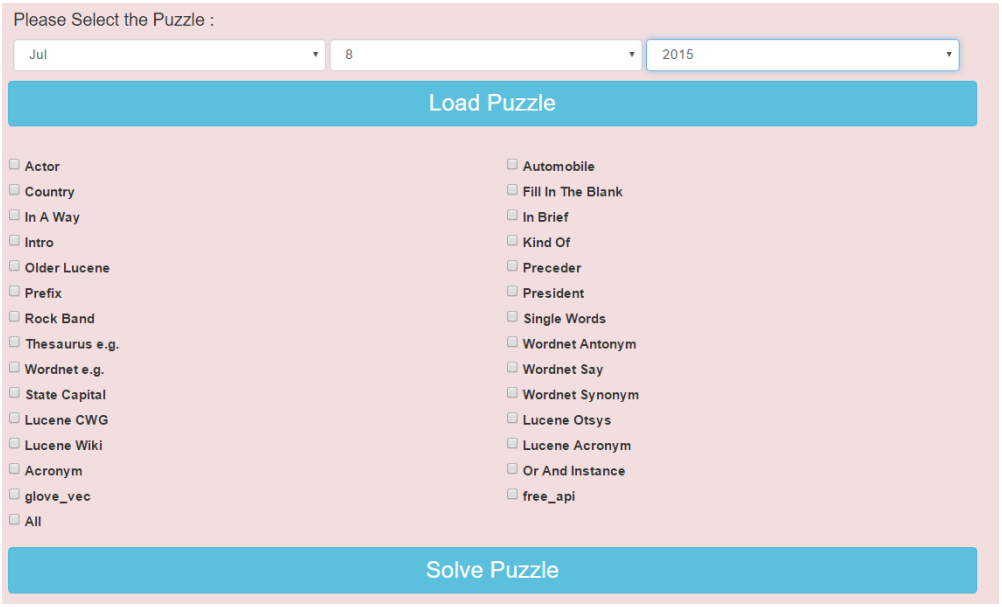

Figure 3: Cruciform Web Interface Controls

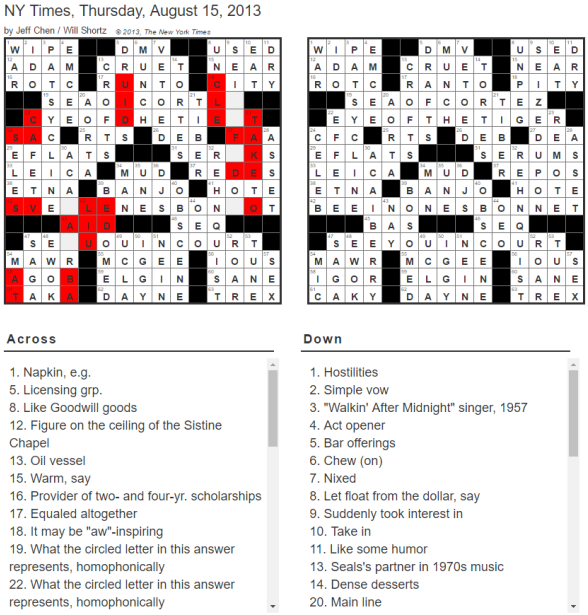

Figure 4: Cruciform Web Interface Output

[5] http://clair.si.umich.edu/crossword2015/

9 Conclusion

In this paper, we propose Cruciform, a new crossword solving system. We employ a group of natural language components that each return a list of candidate words with likelihood scores when given a clue. Then, these lists are used in conjunction with the fill intersections in the puzzle grid to formulate a constraint satisfaction problem (CSP). Additional stages, such as fill-in-the-blanks using bigram probabilities, round out the components of the system.

Acknowledgments

We thank Di Chen, Yanni Gu, Malcolm MacLachlan, Benjamin Englard, Cody Hansen, Anthony Vito, Seunbum Park, Jonathan Juett, Yue Xu, Jonathan Kummerfeld for helpful discussions and feedback.

References

[1] Gianni Barlacchi, Massimo Nicosia, and Alessandro Moschitti. Sacry: Syntax-based automatic crossword puzzle resolution system. *ACL-IJCNLP 2015*, page 79, 2015.

[2] Marco Ernandes, Giovanni Angelini, and Marco Gori. Webcrow: A web-based system for crossword solving. In *AAAI*, pages 1412–1417, 2005.

[3] David A Ferrucci. Ibm's watson/deepqa. In *ACM SIGARCH Computer Architecture News*, volume 39. ACM, 2011.

[4] Matthew L Ginsberg. Dr. fill: Crosswords and an implemented solver for singly weighted csps. *Journal of Artificial Intelligence Research*, pages 851–886, 2011.

[5] Greg A Keim, Noam M Shazeer, Michael L Littman, Sushant Agarwal, Catherine M Cheves, Joseph Fitzgerald, Jason Grosland, Fan Jiang, Shannon Pollard, and Karl Weinmeister. Proverb: The probabilistic cruciverbalist. *New York Times (NYT)*, 792(10):70, 1999.

[6] Michael L Littman, Greg A Keim, and Noam Shazeer. A probabilistic approach to solving crossword puzzles. *Artificial Intelligence*, 134(1):23–55, 2002.

[7] Sara Manzoni. *AI* IA 2005: Advances in Artificial Intelligence: 9th Congress of the Italian Association for Artificial Intelligence Milan, Italy, September 21-23, 2005, Proceedings*, volume 3673. Springer, 2005.

[8] Stuart Jonathan Russell, Peter Norvig, John F Canny, Jitendra M Malik, and Douglas D Edwards. *Artificial intelligence: a modern approach*, volume 2. Prentice hall Upper Saddle River, 2003.