Wordle Is NP-Hard

Daniel Lokshtanov
University of California Santa Barbara, CA, USA

Bernardo Subercaseaux
Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

Wordle is a single-player word-guessing game where the goal is to discover a secret word w that has been chosen from a dictionary D. In order to discover w, the player can make at most ℓ guesses, which must also be words from D, all words in D having the same length k. After each guess, the player is notified of the positions in which their guess matches the secret word, as well as letters in the guess that appear in the secret word in a different position. We study the game of Wordle from a complexity perspective, proving NP-hardness of its natural formalization: to decide given a dictionary D and an integer ℓ if the player can guarantee to discover the secret word within ℓ guesses. Moreover, we prove that hardness holds even over instances where words have length $k = 5$, and that even in this case it is NP-hard to approximate the minimum number of guesses required to guarantee discovering the secret word. We also present results regarding its parameterized complexity and offer some related open problems.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases wordle, np-hardness, complexity

Digital Object Identifier 10.4230/LIPIcs.FUN.2022.19

Related Version *Full Version*: `https://arxiv.org/abs/2203.16713`

Funding *Daniel Lokshtanov*: supported by BSF award 2018302, and NSF award CCF-2008838.
Bernardo Subercaseaux: supported by NSF awards CCF-2015445, CCF-1955785, and CCF-2006953.

Acknowledgements We thank anonymous reviewers for helpful suggestions, Mark Polyakov for pointing out a minor bug in a previous proof of Theorem 1, and Wassim Bouaziz for pointing out some imprecisions in a previous version of this paper. The second author thanks Jérémy Barbay for his insightful conversation, and his awesome friends: Saranya Vijayakumar for proofreading the English in an earlier draft of this paper, and Bailey Miller for providing him a place (and tea) to work on this article.

1 INTRO

Wordle (https://www.nytimes.com/games/wordle/index.html) is a single-player web-based word-guessing game that combines principles of classic games such as Mastermind, Hangman, and Jotto, which have been studied from a computational perspective before [1, 6, 7, 12, 14]. Created by Josh Wardle for his partner [13], and published in October 2021, Wordle has reached an unprecedented level of virality and gained millions of daily players [4, 11]. As for the popularity of the game, Wardle suggested that part of the game's success is due to its artificial scarcity, as it can only be played once a day [9]. Others have pointed at the distinctive colorful patterns with which players share their results [10]. This article can be seen as yet another reason for Wordle's success: its intrinsic complexity. The relationship between in games and their computational complexity has been proposed by multiple authors [5, 8], and thus by establishing theoretical results of hardness for Wordle we can explain part of its challenging nature, and make the reader feel better about their unsuccessful guesses.

The game is played over a dictionary of words D, all of which have length k, that is fully known to the player. Formally, for a given alphabet Σ, $D \subseteq \Sigma^k$. The player, who we will denote the *guesser*, wishes to discover a secret word $w \in D$ through a series of at

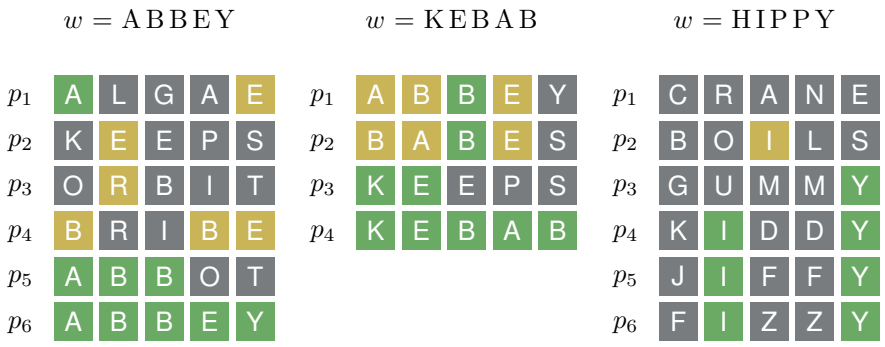

Figure 1 Illustration of three different instances of Wordle. In all of them $k = 5$ and $\ell = 6$. The first two games are won by the guesser, while the third one is lost.

most ℓ guesses $p_1, p_2, \ldots, p_\ell$, all of which must be words belonging to D. The guesser is said to win if one of its guesses p_i is exactly equal to w, and she loses if no such match occurs after ℓ guesses. Whenever a guess p is made, the guesser receives some information about the relation between p and the secret word w: she is notified of every position i such that $p[i] = w[i]$, and also of positions i such that the letter $p[i]$ is present in the word w but not in position i. More precisely, to handle the case of repeated letters, we will use a notion of *marking* with colors. For an index $1 \leq i \leq k$, $w[i]$ marks $p[i]$ with green iff $w[i] = p[i]$. If $w[i] \neq p[i]$, and the set

$$S_i = \{j : p[j] = w[i],\ p[j] \text{ is unmarked}\}$$

is not empty, then $w[i]$ marks $p[\min(S_i)]$ with yellow. All letters $p[j]$ that were not marked at this point are marked with gray. A few games of Wordle are illustrated in Figure 1.

Despite its apparent simplicity, Wordle allows for encoding hard problems, as we present in the next section.

2 Complexity Results

In order to study the complexity of Wordle, we need to define an appropriate formal decision problem for it. Let us assume that the guesser wants to design a strategy that ensures winning the game (i.e., guessing correctly within ℓ attempts) regardless of what the secret word was chosen to be. Thus, we can pose the following problem:

PROBLEM :	Wordle
INPUT :	(D, ℓ) with $D \subseteq \Sigma^k$ the dictionary, and an integer $\ell \geq 1$, the number of guesses allowed.
OUTPUT :	**Yes**, if there is a winning strategy for the guesser, and **No** otherwise.

Note that Σ and k are not explicitly part of the input, but can trivially be deduced from D. We claim now that if a guesser knew a polynomial time algorithm A for this problem, then they could use A to win the game whenever it was possible, as we explain next. First, we will define the notion of feasible words. Observe that at any point in the game, the information the guesser has received thus far (i.e., the letters that have been marked or not in all the previous guesses) defines a dictionary $D' \subseteq D$ of words that could happen to be

the secret word, which we call *feasible* words. For example, after the first guess illustrated in the left-most game of Figure 1, the word `GAMES` would be infeasible with the information received for the first guess, as the letter `G` was not marked. Similarly, the word `KEEPS` used in the second guess of that same game is also infeasible with the result of the first guess, as it does not contain the letter `A` that was marked in the first position. The word `AMAZE` would not be feasible either, as in the first guess the letter `E` at the end is marked with yellow, and thus cannot appear in that position in the secret word. In contrast, the word `ANNEX` would be feasible, and naturally `ABBEY`, the actual secret word of that game, is feasible too. Formally, a word $p \in D$ is feasible after a sequence of guesses $p_1, \ldots, p_n$ if and only if the following conditions hold:

1. No letter $p[i]$ was marked gray in a previous guess p_j, for $1 \leq i \leq k, 1 \leq j \leq n$.
2. No letter $p[i]$ was marked yellow in a previous guess p_j, such that $p_j[i] = p[i]$, for $1 \leq i \leq k, 1 \leq j \leq n$.

Algorithm 1 WordleGuesser(D, ℓ).

```
1.1  if ℓ = 0 then
1.2      guesser loses.
1.3  if |D| = 1 then
1.4      guess the only word in D and win.
1.5  for p ∈ D do
1.6      potentialGuess ← true
1.7      for w ∈ D do
1.8          m ← marking for guess p if w is the secret word
1.9          D' ← {p' ∈ D | p' is feasible after m}
1.10         if WordleGuesser(D', ℓ − 1) is a loss for the guesser then
1.11             potentialGuess ← false
1.12             break
1.13     if potentialGuess then
1.14         Guesser can win using p as its next guess.
1.15 If this point is reached, then the guesser loses.
```

Now, provided a guesser can solve the Wordle problem efficiently, they can play optimally by following Algorithm 1. To see correctness, we can proceed by induction over ℓ. The base case $\ell = 0$ is trivially correct. For the inductive case, if $|D| = 1$ then also correctness is trivial. In any other case, if there is a winning strategy for the guesser within ℓ attempts, that implies that after the first guess, regardless of what the secret word is, it will be possible to win within $\ell - 1$ attempts over the dictionary restricted to the corresponding feasible words. Thus, the inductive hypothesis guarantees that calls to the algorithm with parameter $\ell - 1$ will be correct, thus implying correctness of the algorithm. This justifies the choice of Wordle as a decision problem. Our next step is to study its complexity. We will prove the following theorems:

▶ **Theorem 1.** *Wordle is* NP*-hard, and it is* W[2]*-hard when parameterized by ℓ, the number of guesses allowed.*

▶ **Theorem 2.** *Wordle is* NP*-hard even when restricted to instances where $k = 5$.*

▶ **Theorem 3.** *Wordle can be solved in polynomial time if the alphabet size $\sigma = |\Sigma|$ is constant.*

In order to prove Theorem 1, we will reduce from Almost Set Cover. An input of Almost Set Cover is a pair $(\mathcal{F}, c)$, where $\mathcal{F}$ is a family of sets $S_1, \ldots, S_{|\mathcal{F}|}$ whose union we denote by U, and c is an integer. The goal is to decide whether there is a sub-family $\mathcal{F}' \subseteq \mathcal{F}$ of at most c sets such that

$$\left| U \setminus \left(\bigcup_{S \in \mathcal{F}'} S \right) \right| \leq 1.$$

In other words, the goal is to decide whether it is possible to cover all the elements of the universe, except for perhaps one of them, with c sets. Almost Set Cover is W[2]-hard (parameterized by c), as we show next. It is a simple reduction from standard Set Cover, which is known to be W[2]-hard.

▶ **Lemma 4.** *Almost Set Cover is* W[2]*-hard, when parameterized by c.*

Proof. Let $(\mathcal{F}, c)$ be an instance of Set Cover, and let $U = \bigcup \mathcal{F}$ be its universe. Create first U' of size $2|U|$ in the following way: for each element $u \in U$ put elements u^+ and u^- in U'. Now, create $\mathcal{F}'$ with $|\mathcal{F}|$ sets as follows: for each set $S \in \mathcal{F}$, put into $\mathcal{F}'$ a set

$$S' = \{u^+, u^- \mid u \in S\}.$$

We claim $(\mathcal{F}, c) \in$ Set Cover iff $(\mathcal{F}', c) \in$ Almost Set Cover. For the forward direction, if $\mathcal{F}^* \subseteq \mathcal{F}$ is a set cover of size c for $\mathcal{F}$, then its corresponding sub-family

$$\mathcal{F}^{*'} = \{S' \mid S \in \mathcal{F}^*\}$$

is trivially a set cover for $\mathcal{F}'$. On the other hand, if no sub-family K of at most c sets in $\mathcal{F}$ covers $\mathcal{F}$, that means that for every such sub-family K, there is at least one element $u \notin \bigcup K$. But now if we consider

$$K' = \{S' \mid S \in K\},$$

then both u^+ and u^- are not covered by K'. This implies that every sub-family $K' \subseteq \mathcal{F}'$ of size at most c leaves at least two elements outside. This concludes the reduction, as it can be clearly computed in FPT-time. ◀

We are now ready to prove Theorem 1.

Proof of Theorem 1. Let $(\mathcal{F}, c)$ be an instance of Almost Set Cover. We will build a dictionary $D_{\mathcal{F}}$ as follows. First, we will need $|U| + 2$ symbols, that we will denote $\Sigma_{\mathcal{F}} = \{\bot, 1, s_1, \ldots, s_{|U|}\}$. Now, $D_{\mathcal{F}}$ will contain two different kinds of words, *element-words* and *set-words*, both of which will have length $|U|$. For every element $u_i \in U$, build its corresponding word w_{u_i} as:

$$w_{u_i}[j] = \begin{cases} 1 & \text{if } i = j \\ s_i & \text{otherwise.} \end{cases}$$

All words w_{u_i} created this way constitute the set of element-words. Now, for every set $S_i \in \mathcal{F}$, build its corresponding set-word w_{S_i} as:

$$w_{S_i}[j] = \begin{cases} 1 & \text{if } u_j \in S_i \\ \bot & \text{otherwise.} \end{cases}$$

We now claim that $(\mathcal{F}, c) \in$ Almost Set Cover iff $(D_{\mathcal{F}}, c+1) \in$ Wordle. For the forward direction, if there is a family of sets $\mathcal{F}'$, with $|\mathcal{F}'| \leq c$ and such that $|U \setminus \bigcup \mathcal{F}'| \leq 1$, then the guesser can use the set-words w_S for every $S \in \mathcal{F}'$ to guarantee a win. If the secret word w was chosen to be a set-word, then we have two cases after the first guess $w_S, S \in \mathcal{F}'$. Either the secret word w is revealed to be exactly w_S, in which case the guesser wins, or it is another set-word $w_{S'}$, in which case we claim that after guess w_S is made, $w_{S'}$ will be the only feasible word remaining and thus the guesser will win in her second turn. Indeed, for every element $u_i \in U$ there are 4 cases:

1. $(u_i \in S, u_i \in S')$. In this case $w_S[i] = w_{S'} = 1$, and it will be marked green. This makes all words that do not have a 1 in their i-th position infeasible.
2. $(u_i \in S, u_i \notin S')$. In this case $w_S[i] = 1$, and it will not be marked green. This makes all words that have a 1 in their i-th position infeasible.
3. $(u_i \notin S, u_i \in S')$. In this case $w_S[i] = \perp$, and it will not be marked green. This makes all words that have a $\perp$ in their i-th position infeasible.
4. $(u_i \notin S, u_i \notin S')$. In this case $w_S[i] = w_{S'} = \perp$, and it will be marked green. This makes all words that do not have a $\perp$ in their i-th position infeasible.

Note that regardless of the case, words that do not agree with $w_{S'}$ on their i-th position become infeasible, and as this happens for every $i \in [U]$, the only remaining feasible word is $w_{S'}$. As $c+1 \geq 1$, and we have shown that if the secret word is a set-word then the guesser can win in at most 2 turns, we can safely proceed to the case where the secret word is an element-word. For this case the guesser can first make c guesses corresponding to all the words w_S for $S \in \mathcal{F}'$. Let w_{u_i} be the secret element word. As $|U \setminus \bigcup \mathcal{F}'| \leq 1$, there are two cases: in the first case, there is some $S_j \in \mathcal{F}'$ such that $u_i \in S_j$. Whenever the guess w_{S_j} is made, as it will happen that $w_{S_j}[i] = w_{u_i}[i] = 1$. This will reveal a 1 in the i-th position of the secret word, and as it is an element-word, this unambiguously reveals w_{u_i} to the guesser, who will win in the next turn, and thus in no more than $c+1$ guesses. The second case is when w_{u_i}, the secret word, is the only element of $U \setminus \bigcup \mathcal{F}'$, and thus the guesser simply say that word in its next turn, also ensuring a win. This concludes the forward direction.

For the backward direction, assume every sub-family $\mathcal{F}' \subseteq \mathcal{F}$ leaves at least two elements u_1 and u_2 of U uncovered. Then we claim that no strategy for the guesser can guarantee a win within $c+1$ attempts. Indeed, for any set of c initial guesses the guesser can make, we claim that there are at least two feasible words remaining, and thus the guesser cannot ensure a win in her last turn. To see this, consider any sequence $W = w_1, \ldots, w_c$ of c guesses, and then define

$$\mathcal{F}_{\text{sets}} = \{S \in \mathcal{F} \mid w_S \in W\} \quad ; \quad \mathcal{F}_{\text{elem}} = \{\{u\} \subseteq U \mid w_u \in W\}.$$

Now observe that every element-word w_u such that $u \notin \bigcup (\mathcal{F}_{\text{sets}} \cup \mathcal{F}_{\text{elem}})$, is feasible after the sequence of guesses W. Also notice that for every $S \in \mathcal{F}_{\text{elem}}$ there is a set $S' \in \mathcal{F}$ such that $S \subseteq S'$. Thus, if for every set $S \in \mathcal{F}_{\text{elem}}$ we choose a set $S' \in \mathcal{F}$ such that $S \subseteq S'$, we get a collection of sets $\mathcal{F}'$ such that $(\bigcup \mathcal{F}_{\text{elem}}) \subseteq (\bigcup \mathcal{F}')$ and $|\mathcal{F}'| = |\mathcal{F}_{\text{elem}}|$. This implies that $(\mathcal{F}_{\text{sets}} \cup \mathcal{F}') \subseteq \mathcal{F}$ and $(\mathcal{F}_{\text{sets}} \cup \mathcal{F}')$ is a collection of at most c sets, and thus there are at least two elements $u_1, u_2 \in U$ that are not on its union. We thus also have that

$$\{u_1, u_2\} \cap \left(\bigcup \mathcal{F}_{\text{sets}} \cup \mathcal{F}_{\text{elem}}\right) = \emptyset,$$

and consequently, both w_{u_1} and w_{u_2} are feasible words, which implies it is not possible to guarantee a win in the next guess. This concludes the proof. ◀

Theorem 2 captures an arguably essential part of Wardle's Wordle: difficulty does not require long words. Its proof is inspired by that of coNP-hardness for Evil Hangman by Barbay and Subercaseaux [1]. We will use a result in hardness of approximation to prove that Wordle cannot be solved efficiently unless P = NP. In particular, we will use that $\gamma(G)$, the size of the smallest dominating set in a 4-regular graph G, is NP-hard to approximate within $1 + \frac{1}{390}$ [2, 3]. For our purpose it will be enough to use that is NP-hard to compute a $(1 + \epsilon)$-approximation.

Proof of Theorem 2. Let G be a 4-regular graph. We will work with words of length $k = 5$. The alphabet Σ will be the vertex set of G. We build a dictionary D_G from G as follows: for every vertex $v \in G$, we create two words, w_v and w'_v: the word w_v has v as its first symbol is v, and its 4 remaining symbols are the neighbors of v (the order of these 4 symbols can be chosen arbitrarily), while the word w'_v consists simply of the symbol v repeated 5 times. Given a dictionary D, let $W(D)$ be the minimum value of ℓ such that (D, ℓ) is a positive instance of Wordle. We now claim that $\gamma(G) \in [W(D_G) - 4, W(D_G)]$. Note that this claim is enough to prove the theorem, as a polynomial time algorithm for Wordle would imply that $W(D_G)$ can be computed in polynomial time, and thus $\gamma(G)$ could be approximated up to an additive constant, contradicting its $1 + \frac{1}{390}$ hardness of approximation [2, 3]. In order to prove $\gamma(G) \geq W(D_G) - 4$, we show that a dominating set of size d for G implies a guessing strategy that guarantees a win within $d + 4$ guesses. Assume G has a dominating set S of size at most d and the secret word is either w_u or w'_u for some $u \in V(G)$. Now consider the sequence of guesses $w_v, v \in S$. If $u \in S$ we have two cases, either the secret word was w_u, and thus the secret word was already guessed in d guesses, or the secret word was w'_u, in which case the first symbol of guess w_u was marked green, leaving only w'_u as a feasible word, and thus allowing a guaranteed win within $d + 1$ guesses. If $u \notin S$, then as S is a dominating set u must be a neighbor of some vertex $v^\star$ in S. If the secret word was w_u, then the symbol u in guess $w_{v^\star}$ must have been marked yellow, which allows to find w_u by guessing the sequence of words $w_{u'}$ for $u' \in N(v^*)$, which must contain w_u, as u is a neighbor of v. This guessing strategy has at most $d + 4$ guesses and it is guaranteed to succeed. On the other hand, if the secret word was w'_u, then the symbol u in guess $w_{v^\star}$ must have been marked green, which makes w'_u the only feasible word, and thus allows to win in $d + 1$ guesses. In order to prove $\gamma(G) \leq W(D_G)$, we show that is not possible to guarantee a win within $\gamma(G) - 1$ guesses. Indeed, any sequence σ of $\gamma(G) - 1$ guesses induces a set S of vertices by considering the first symbol of each guess. As $|S| \leq \gamma(G) - 1$, there needs to be a vertex v that is not dominated by S, the word w'_v is still a feasible word that is not part of σ, and thus in the case where every request in σ was entirely marked with gray, then σ does not make the guesser win. As this is true for any sequence σ of length $\gamma(G) - 1$, we conclude $W(D_G) \geq \gamma(G)$. ◀

We naturally obtain hardness of approximation as a corollary.

▶ **Corollary 5.** *Let $W(D)$ be the minimum number of guesses ℓ such that the guesser can guarantee to win a game of Wordle over dictionary D within ℓ guesses, i.e., $(D, \ell) \in$ Wordle. Then it is* NP*-hard to approximate $W(D)$ within $1 + \frac{1}{390}$.*

Interestingly, both the proof of Theorem 1 and Theorem 2 crucially depend on variable length alphabets. This if further confirmed by Theorem 3, which will we prove next. First, consider the following lemma:

▶ **Lemma 6.** *Wordle restricted to instances where $\ell \leq c$, for some fixed constant c, can be solved in polynomial time.*

Proof. This can be seen by analizing the branching factor and depth of the associated game-tree. The game-tree has a branching factor of $O(D)$, as at any point the guesser can choose one of the $O(D)$ feasible words remaining, and for each such word there are at most $O(D)$ possible responses the guesser can get, depending on what the secret word w was at that point. The game-tree has also depth at most ℓ, which implies thus that the size of the game-tree is at most $O(D^\ell) = D^{O(1)}$, and thus the optimal guess at any given point can be computed in polynomial time. The result follows from this. ◀

Now we can a prove a key lemma relating $\sigma = |\Sigma|$ and ℓ.

▶ **Lemma 7.** *On a game of Wordle over an alphabet Σ of size $\sigma = |\Sigma|$, the guesser can always win in at most σ guesses.*

Proof. Consider the following simple algorithm for the guesser: *at any point of the game, choose any feasible word as a guess.* We will show that this algorithm requires at most σ guesses to find the secret word. Indeed, for each index $1 \leq i \leq k$, let us define sets $S(i) \subseteq \Sigma$ as:

$$S(i) = \{s \in \Sigma \mid \text{there is a feasible word } w \in D, \text{ such that } w[i] = s\}.$$

Note that, before the first guess, $S(i) = \Sigma$ for every i. The key idea is now the following: after a feasible word w is guessed by the aforementioned strategy, the following holds for every $1 \leq i \leq k$: $w[i]$ can either be marked green, in which case $S(i) \leftarrow \{w[i]\}$, or it can be marked with gray or yellow, in which case $S(i) \leftarrow S(i) \setminus \{w[i]\}$. This implies that after every feasible guess each set $S(i)$ either becomes a singleton or decreases its size by 1. Therefore, by doing $\sigma - 1$ feasible guesses, either the secret word has been found (in which case we are done), or it happens that every set $S(i)$ must be a singleton, and thus there will be only one remaining feasible word, which allows the guesser to win within σ guesses in total. ◀

We are now ready to use the previous lemmas and prove Theorem 3.

Proof of Theorem 3. Consider an instance (D, ℓ) of Wordle where $|\Sigma| \in O(1)$. By Lemma 7, if $|\Sigma| \leq \ell$, then simply answer **Yes**. Otherwise $\ell \leq |\Sigma| = O(1)$, and thus by Lemma 6 the result follows. ◀

3 OPEN Problems

We now offer problems that our analysis leaves open.

1. Is Wordle in NP, or is it PSPACE-complete? It is not hard to see that Wordle is in PSPACE, by simply computing its associated game-tree, which has polynomial depth by Lemma 7.
2. Is Wordle in FPT when parameterized by $\sigma = |\Sigma|$? The proof of Theorem 3 provides an algorithm in time $D^{O(\ell)}$, and it seems challenging to obtain a $\text{poly}(D) \cdot f(\ell)$ algorithm for some computable function ℓ.
3. How does is the complexity of Wordle affected by having a dictionary that is not presented as a list of word, but rather in some more compact representation as a finite automaton? We can envision D to be provided as a finite automaton, for which all accepted words of length k are considered part of the dictionary. This affects for example the proof of Theorem 3, as now a branching factor of $|D|$ can be exponential in the input size.

References

1. Jérémy Barbay and Bernardo Subercaseaux. The computational complexity of evil hangman. In *10th International Conference on Fun with Algorithms (FUN 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.
2. M. Chlebík and J. Chlebíková. Approximation hardness of dominating set problems in bounded degree graphs. *Information and Computation*, 206(11):1264–1275, November 2008. doi:10.1016/j.ic.2008.07.003.
3. Miroslav Chlebík and Janka Chlebíková. Approximation hardness of edge dominating set problems. *Journal of Combinatorial Optimization*, 11(3):279–290, May 2006. doi:10.1007/s10878-006-7908-0.
4. New York Times Company. Wordle is joining the new york times games. https://www.nytco.com/press/wordle-new-york-times-games/, 2022. Accessed: 2022-02-17.
5. David Eppstein. Computational complexity of games and puzzles. https://www.ics.uci.edu/~eppstein/cgt/hard.html, 1997. Accessed: 2022-02-17.
6. Sam Ganzfried. Computing strong game-theoretic strategies in jotto. In *Lecture Notes in Computer Science*, pages 282–294. Springer Berlin Heidelberg, 2012. doi:10.1007/978-3-642-31866-5_24.
7. Michael T. Goodrich. On the algorithmic complexity of the mastermind game with black-peg results. *Information Processing Letters*, 109(13):675–678, June 2009. doi:10.1016/j.ipl.2009.02.021.
8. Robert A. Hearn and Erik D. Demaine. *Games, Puzzles, and Computation*. A. K. Peters, Ltd., USA, 2009.
9. Nicole Holliday and Ben Zimmer. Wordle's creator thinks he knows why the game has gone so viral. https://slate.com/culture/2022/01/wordle-game-creator-wardle-twitter-scores-strategy-stats.html, 2022. Accessed: 2022-02-17.
10. Meeta Malhotra. Wordle's viral success is based on design. https://thehardcopy.co/wordles-viral-success-is-based-on-design/, 2022. Accessed: 2022-02-17.
11. Erin Sebo. Codecracking, community and competition: why the word puzzle wordle has become a new online obsession. https://theconversation.com/codecracking-community-and-competition-why-the-word-puzzle-wordle-has-become-a-new-online-obsession-174878, 2022. Accessed: 2022-02-17.
12. Jeff Stuckman and Guo-Qiang Zhang. Mastermind is np-complete, 2005. arXiv:cs/0512049.
13. Daniel Victor. Wordle is a love story. https://www.nytimes.com/2022/01/03/technology/wordle-word-game-creator.html, 2022. Accessed: 2022-02-17.
14. Giovanni Viglietta. Hardness of mastermind. In Evangelos Kranakis, Danny Krizanc, and Flaminia Luccio, editors, *Fun with Algorithms*, pages 368–378, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.