
Algorithmes et méthodes de résolution statistiques pour Quordle



11 Avril au 16 Juin 2023
Stage de fin de deuxième année de BUT
Informatique

Etudiant : Thomas BREIL

Maitre de Stage : Costin BĂDICĂ
Professeure tutrice : Cristina ONETE

Remerciements

Je tiens à remercier avant de débiter mon rapport de stage l'Université de Craiova et en particulier M. BĂDICĂ pour la confiance qu'ils m'ont accordé.

Je tiens également à remercier M. BECHERU pour la mise en place du GitHub du projet.

Je remercie également l'Université de Limoges pour l'opportunité, Mme ONETE pour avoir supervisé mon stage durant ces dix semaines.

Table des matières

Introduction	6
1 Présentation du Stage	7
1.1 Contexte du Stage	7
1.1.1 Erasmus	7
1.1.2 Intelligent Distributed Systems Research Group	8
1.1.3 Organisation du Travail	8
1.2 Présentation du sujet du Stage	8
1.2.1 Introduction à Wordle	8
1.2.2 Elargissement à Quordle	9
1.2.3 Objectifs du Stage	10
2 Méthodes de Travail	11
2.1 Langage de Programmation : R	11
2.1.1 Présentation du R	11
2.1.2 R Studio	12
2.2 Latex	13
2.2.1 Présentation du Latex	13
2.2.2 Overleaf	13
2.3 Wordle	13
2.3.1 Package Wordle en R	13
2.3.2 Changements par rapport au jeu en ligne	14
2.4 Algorithme de Louis Monié sur Wordle	14
2.4.1 Fréquence des Lettres	15
2.4.2 Cas de lettre unique	15
2.4.3 Doublons de lettres	17
2.4.4 Conclusion du Wordle	17
2.5 Quordle	18
2.5.1 Ajout des quatre parties consécutives	18
2.5.2 Cas de lettre unique	19
2.5.3 Fonction de test	20
2.5.4 Sauvegarde des Données	20

3	Implémentation et analyse des différentes versions	21
3.1	Première version	21
3.1.1	Procédé de résolution	21
3.1.2	Désavantages de la solution	21
3.2	Seconde version	23
3.2.1	Ajout du principe d'aléatoire	23
3.2.2	Problèmes rencontrés	24
3.2.3	Analyse des résultats	24
3.3	Troisième version	25
3.3.1	Augmentation du nombre d'essais par défaut	25
3.3.2	Classement des mots mystères	27
3.3.3	Problèmes rencontrés	28
3.3.4	Analyse des résultats	29
3.4	Essai avec le dictionnaire officiel	30
3.4.1	Ajout du dictionnaire officiel dans les Helpers	32
4	Pistes d'évolution	34
4.1	Mots par défaut	34
4.2	Classement des mots	34
4.3	Généralisation du cas de lettre unique	34
	Conclusion	36
	Annexes	36

Introduction

Ce stage vient cloturer ma seconde année de BUT Informatique en mettant à l'épreuve les nombreuses compétences accumulées lors de ma formation. Ainsi, du 11 Avril au 16 Juin 2023, j'ai pu, grâce à l'aide de mon maitre de stage M. BĂDICĂ et de ma tutrice Mme. ONETE, mener à bien les tâches qui m'ont été confiées.

Lorsque l'on m'a présenté la possibilité d'effectuer mon stage dans un pays étranger, j'ai choisi de postuler à l'Université de Craiova, en Roumanie. Ainsi j'ai pu travailler avec l'Intelligent Distributed Systems Research Group, un groupe de recherche autour de l'intelligence artificielle dirigé par M. BĂDICĂ.

Ainsi, mes objectifs étaient de créer un solveur pour le jeu Quordle, dérivé du Wordle, un puzzle en ligne. Ce travail vient compléter les recherches de Louis Monié, étudiant en DUT Informatique l'année dernière en Erasmus à Craiova. J'ai ainsi pu utiliser ses travaux sur Wordle et les adapter à Quordle en comparant les stratégies.

Je vais vous présenter dans ce rapport les différentes étapes de mon travail selon le plan suivant : tout d'abord, je vais développer sur le contexte du stage, l'organisation de mon travail. Ensuite, je reviendrai sur les travaux de Louis Monié sur Wordle qui ont servi de bases pour mon stage, je développerai les stratégies et méthodes de travail qu'il a mis en place.

Enfin, je vous présenterai mes travaux sur Quordle. J'expliquerai les différentes méthodes mises en place qui m'ont permis de développer mon solveur pour le puzzle.

Pour finir, je présenterai des pistes d'amélioration afin de parfaire le solveur, dans le but d'obtenir un pourcentage de réussite le plus élevé possible.

Chapitre 1

Présentation du Stage

1.1 Contexte du Stage

Cette partie a pour but d'expliquer le contexte de mon stage. Je vais aborder le programme Erasmus, ainsi que l'organisation de mon travail.

1.1.1 Erasmus

Ce stage d'une durée de 10 semaines, du 11 Avril au 16 Juin 2023 consistait à aller travailler avec un groupe de recherche de l'Université de Craiova nommé "Intelligent Distributed Systems Research Group".



FIGURE 1.1 – Craiova, ville du Sud-Ouest de la Roumanie

J'ai pu partir en stage grâce au programme Erasmus +. J'ai donc pu cotoyer

pendant ces 10 semaines des étudiants venant de toutes l'Europe, ce qui est une expérience réellement enrichissante. De plus, j'ai également pu découvrir une nouvelle culture.

1.1.2 Intelligent Distributed Systems Research Group

L'Intelligent Distributed Systems Research Group est un groupe de recherche de l'Université de Craiova dirigé par M. BĂDICĂ. Ce groupe opère dans le domaine de l'intelligence artificielle et l'intelligence computationnelle.

1.1.3 Organisation du Travail

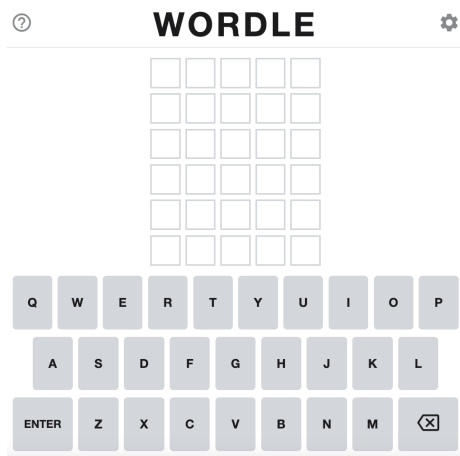
Le bâtiment de l'Université de Craiova étant en réhabilitation, je n'ai pas pu avoir de bureau de travail dédié. J'ai ainsi eu à travailler depuis le dortoir en distanciel. Nous nous sommes donc mis d'accord avec M. BĂDICĂ de se rencontrer toutes les deux semaines pour parler de mes avancements et fixer de nouveaux objectifs. J'avais également la possibilité de le contacter par mail en cas de problèmes afin d'éviter les pertes de temps.

De part cette méthode, j'ai pu avancer dans les meilleurs conditions, car la mise en place d'objectifs pour une durée de deux semaines m'ont permis d'ajuster mon travail en fonction de mes facilités et blocages.

1.2 Présentation du sujet du Stage

1.2.1 Introduction à Wordle

Wordle est un puzzle en ligne qui a été créé en 2021. Le but du jeu est de trouver un mot mystère en six essais. A chaque tentative, le jeu indique donne des indices au joueur pour l'aiguiller dans sa réflexion.



(a) Grille de Wordle vide



(b) Grille de Wordle complétée

FIGURE 1.2 – Exemple de Grille de Wordle

Comme on peut le voir sur les images ci-dessus, lorsqu'une lettre n'est pas présente dans le mot mystère, elle est grisée. Une lettre jaune signifie que la lettre est bien présente dans le mot, mais elle n'est pas bien placée. Enfin, la lettre verte est correctement placée dans le mot mystère.

Ainsi, avec ces indices, le joueur va avoir toutes les armes en main pour résoudre la grille.

1.2.2 Elargissement à Quordle

Quordle est un dérivé du jeu Wordle. Ici, ce n'est non pas une grille à résoudre mais quatre, avec 9 essais disponibles. Chaque essai donne des indices pour les quatre mots, cela apportant un nombre important de stratégies différentes.

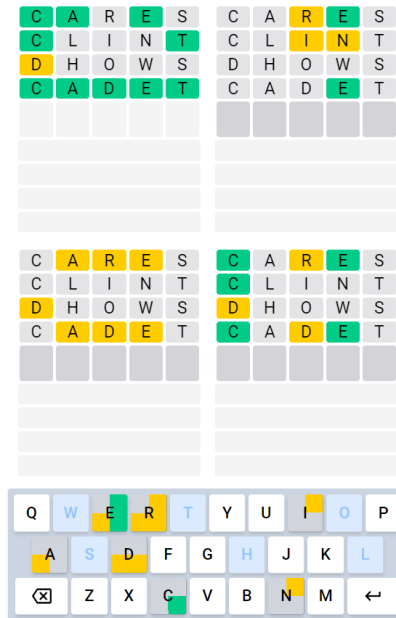


FIGURE 1.3 – Grille de Quordle

1.2.3 Objectifs du Stage

Le sujet de stage fait suite au travail de Louis Monié, élève de DUT Informatique qui a effectué son stage à l'Université de Craiova l'année dernière. L'objectif est donc de créer des algorithmes permettant de résoudre le puzzle Quordle en utilisant différentes stratégies et de les comparer.

Les résultats attendus que je développe plus bas sont donc la présentation des algorithmes montrant l'évolution des stratégies au fil du stage.

Chapitre 2

Méthodes de Travail

Je vais dans cette partie présenter le langage de programmation que j’ai utilisé durant mon stage. Je reviendrai sur le puzzle Wordle et je vais aborder les travaux de Louis qui m’ont servi de bases. Enfin, je présenterai comment j’ai adapté le package Wordle afin de pouvoir jouer des parties de Quordle.

2.1 Langage de Programmation : R

J’ai fait le choix de travailler sur le langage de programmation R, puisque Louis avait fait ses travaux sur ce dernier. De plus, le R possède de nombreux packages centrés autour du puzzle Wordle. Ce choix paraît donc logique puisque cela évite de devoir passer du temps sur la programmation d’un environnement de jeu Wordle. Ainsi, l’adaptation du Wordle en Quordle a été simplifiée.

Ce stage m’a donc permis de découvrir un nouveau langage de programmation et d’enrichir ma culture et mes compétences de développeur.

2.1.1 Présentation du R

Le R est un langage de programmation tourné autour des sciences des données et des statistiques. Créé en 1993 en Nouvelle-Zélande, ce langage basé sur le C est toujours aujourd’hui un langage populaire et utilisé dans le monde du travail. Il a notamment été classé 7e meilleur langage de programmation en 2021 par l’IEEE (Institute of Electrical and Electronics Engineers) et 11e au classement TIOBE en décembre 2022.

Rank	Language	Type	Score
1	Python	🌐 🖨️ 🌐	100.0
2	Java	🌐 📱 🖨️	95.4
3	C	📱 🖨️ 🌐	94.7
4	C++	📱 🖨️ 🌐	92.4
5	JavaScript	🌐	88.1
6	C#	🌐 📱 🖨️ 🌐	82.4
7	R	🖨️	81.7
8	Go	🌐 🖨️	77.7
9	HTML	🌐	75.4
10	Swift	📱 🖨️	70.4

(a) Classement 2021 de l'IEEE

Dec 2022	Programming Language	Ratings
1	Python	16.66%
2	C	16.56%
3	C++	11.94%
4	Java	11.82%
5	C#	4.92%
6	Visual Basic	3.94%
7	JavaScript	3.19%
8	SQL	2.22%
9	Assembly language	1.87%
10	PHP	1.62%
11	R	1.25%

(b) Classement Décembre 2022 TIOBE

FIGURE 2.1 – Classement de Langage selon différents organismes

2.1.2 R Studio

Un autre avantage du langage R est qu'il possède un environnement de développement dédié, R Studio.

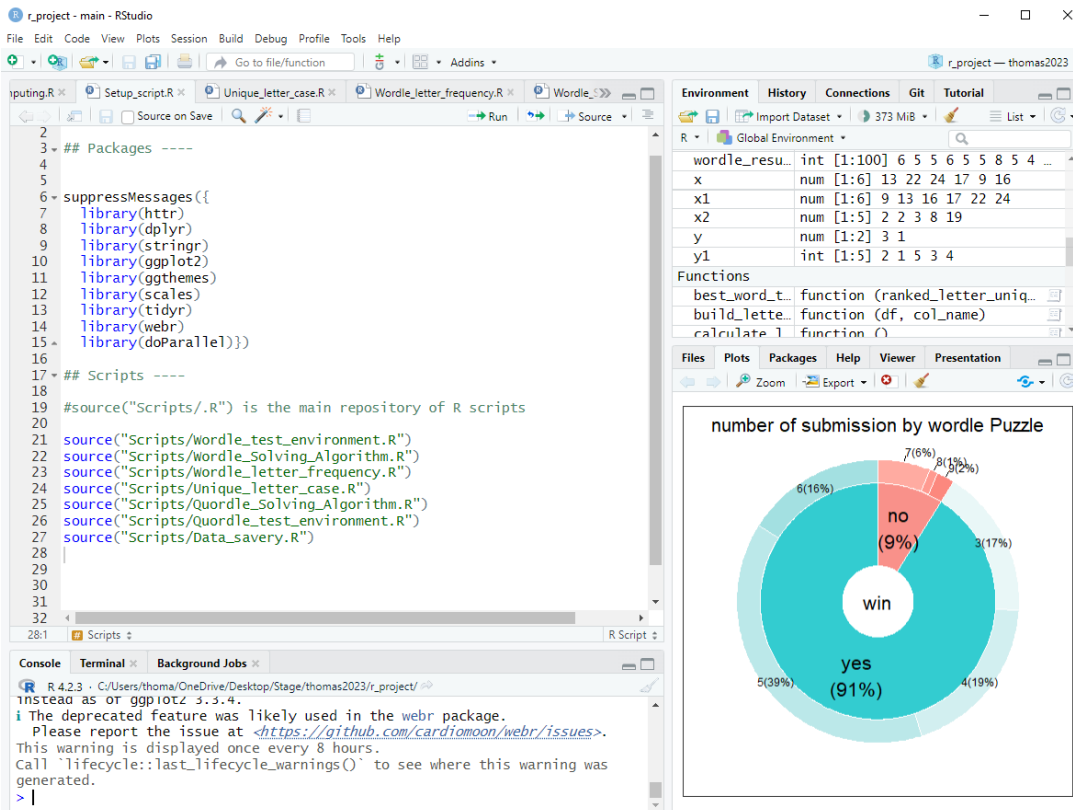


FIGURE 2.2 – Interface RStudio

Comme on peut voir sur la figure ci-dessus, RStudio facilite grandement le développement en R, en proposant une interface propre et claire afin de coder dans les meilleures conditions.

2.2 Latex

2.2.1 Présentation du Latex

Pour rédiger ce rapport, il m'a été demandé de composer en LaTeX. Le LaTeX est un langage de programmation basé sur des balises permettant la rédaction de document dont la mise en page est réalisée automatiquement à partir du type de document choisi. Par exemple, le LaTeX crée seul la table des matières à partir des chapitres et des sections que nous écrivons.

J'ai donc pu dans le cadre de stage apprendre la rédaction en LaTeX, ce qui me servira très probablement dans ma vie future.

2.2.2 Overleaf

Pour éditer en LaTeX, j'ai utilisé Overleaf. Overleaf est un éditeur LaTeX en ligne, revendiquant une communauté active de plus de onze millions de personnes.

L'un des avantages majeurs d'Overleaf est le partage de document. Comme un Google Doc, plusieurs personnes peuvent rédiger simultanément un même document LaTeX, notamment dans le cadre de la rédaction d'un article scientifique.

2.3 Wordle

2.3.1 Package Wordle en R

Le package utilisé pour ce stage est celui développé par CoolButUseless. On peut retrouver son travail sur son Github : <https://github.com/coolbutuseless/wordle/>

Ce package met en place un objet "Game", qui va nous permettre d'essayer des mots afin de résoudre notre grille, en nous retournant la couleur des lettres utilisées. On a aussi à disposition un Helper. Cet objet nous permet de réduire le dictionnaire

de mot en fonction des essais effectués sur l’objet Game. Ainsi, on va pouvoir adapter nos soumissions en fonction des mots restants dans le dictionnaire du Helper.

2.3.2 Changements par rapport au jeu en ligne

Le dictionnaire proposé par le puzzle Wordle en ligne comporte 10’638 mots. Cependant, le jeu est fait de telle façon que seulement 2’309 mots peuvent être choisis en tant que mots mystères. Ce choix est justifié par le créateur du jeu qui indique que les solutions représentent les mots les plus courants de la langue anglaise.

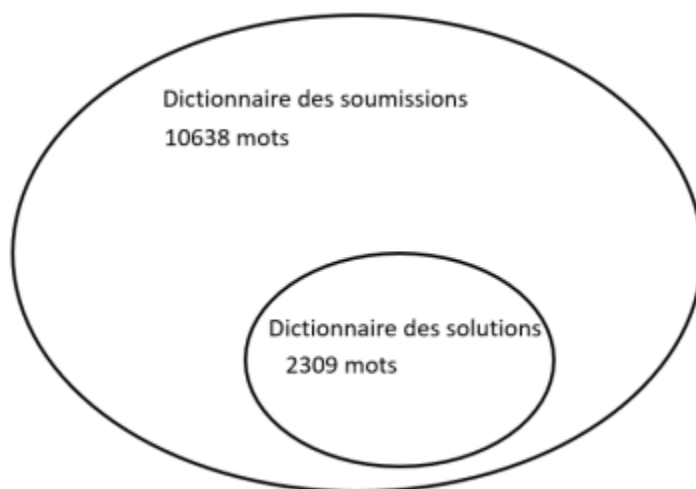


FIGURE 2.3 – Dictionnaire Wordle

Cependant, dans notre version du jeu, nous utilisons un dictionnaire de 12’972 mots, qui peuvent chacun être solutions. On a donc un nombre plus important de grille possible. Le choix d’agrandir le dictionnaire a été fait par le développeur de package Wordle de R que Louis et Moi avons utilisé. Cependant, ce choix ne nuit pas à notre projet. Au contraire, travailler sur un plus grand dictionnaire nous oblige à avoir des algorithmes et des stratégies plus pointilleux.

2.4 Algorithme de Louis Monié sur Wordle

Je vais présenter dans cette partie les travaux de Louis sur lesquels je me suis basé pour ce stage. Je vais expliquer ses stratégies et ses méthodes pour faciliter la compréhension de mon travail.

2.4.1 Fréquence des Lettres

La méthode de choix des soumissions de Louis est fait en utilisant la fréquence d'apparition des lettres de l'alphabet dans la langue anglaise.



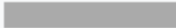





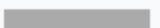

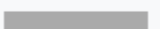

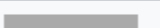

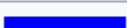

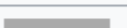









E	11%		P	2.8%	
S	8.7%		M	2.7%	
I	8.6%		H	2.3%	
A	7.8%		B	2%	
R	7.3%		Y	1.6%	
N	7.2%		F	1.4%	
T	6.7%		V	1%	
O	6.1%		K	0.97%	
L	5.3%		W	0.91%	
C	4%		Z	0.44%	
D	3.8%		X	0.27%	
U	3.3%		J	0.21%	
G	3%		Q	0.19%	

FIGURE 2.4 – Fréquence des lettres dans la langue anglaise

Pour Wordle, il va calculer la fréquence d'apparition de chaque lettre dans le dictionnaire du puzzle. Cependant, il va en plus calculer la fréquence d'apparition des lettres à chaque place du mot. Ainsi, il va pouvoir en déduire le mot qui va donner le plus d'indices.

Avec ses calculs, il a déduit un mot à soumettre en premier pour chaque partie, qui est "cares". Ensuite, il va recalculer les fréquences avec le dictionnaire des mots restants grâce aux indices donnés par la première soumission et ainsi de suite.

2.4.2 Cas de lettre unique

Afin d'optimiser son algorithme, Louis a mis en place une règle précise. Quand quatre des cinq lettres du mots sont vertes, plutôt que de tenter tous les mots restants,

qui peuvent être nombreux, il va déterminer un mot contenant plusieurs des lettres possibles pour la dernière place afin de gagner des coups.

```
> Wordle_solving_Algorithm("yills")
```

c	a	r	e	s
s	o	o	t	s
b	l	i	n	s
p	i	l	i	s
m	i	l	l	s

FIGURE 2.5 – Exemple de cas de lettre unique

On voit donc que dans ce cas, il manque seulement la première lettre de mot pour résoudre le Puzzle. Cependant, il reste des nombreuses possibilités :

```
"dills" "fills" "gills" "hills"
"jills" "kills" "lills" "vills"
"wills" "yills" "zills"
```

FIGURE 2.6 – Solutions possibles restantes dans le dictionnaire

Ainsi, plutôt que tenter toutes les solutions possibles ci-dessus, l'algorithme va définir un mot qui va tester le plus de lettres possibles. Ici, le mot est "fudgy", qui vérifie quatre lettres différentes en un seul coup. Ensuite, le solveur va indiquer que le "Y" se trouve dans notre mot, et ainsi résoudre la grille.

c	a	r	e	s
s	o	o	t	s
b	l	i	n	s
p	i	l	i	s
m	i	l	l	s
f	u	d	g	y
y	i	l	l	s

FIGURE 2.7 – Grille Résolue

Dans ce cas, le nombre de soumissions reste trop élevé pour que la partie soit gagnée. Cependant, cette règle nous a permis d'économiser dix essais. De plus, cela nous permet d'augmenter le taux de réussite de l'algorithme de plusieurs pourcent.

2.4.3 Doublons de lettres

Les lettres présentes plusieurs fois dans un même mot peuvent créer des problèmes de résolution pour l'algorithme, notamment du côté de la fréquence d'apparition des lettres. Par exemple, on a vu que les lettres "S", "E" ou encore "A" avaient une fréquence d'apparition élevées. Cependant, les mots contenant plusieurs fois ces lettres sont rares. Ainsi, afin de ne pas fausser le classement des solutions restantes, Louis a fait le choix de descendre le classement des mots avec des doublons de lettres. Cela permet de tester plus de lettres différentes afin de résoudre le puzzle plus efficacement.

2.4.4 Conclusion du Wordle

Ainsi, avec toutes ses stratégies, Louis a créé un algorithme permettant de jouer au Wordle en se basant sur les probabilités et les statistiques d'apparition des lettres dans la langue anglaise. Avec cet algorithme, Louis obtient un taux de réussite de 89% du dictionnaire de jeu.

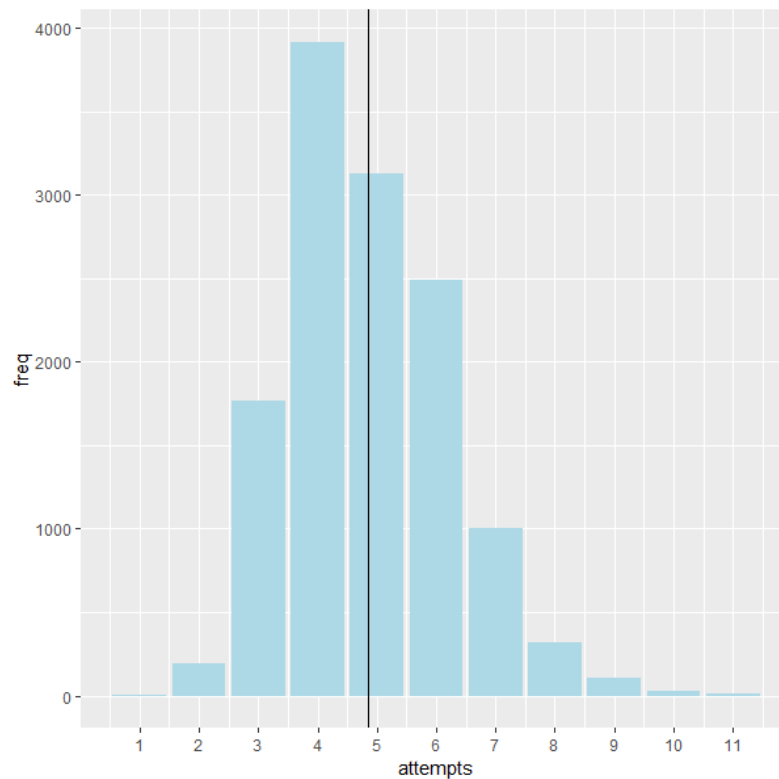


FIGURE 2.8 – Nombres de mots résolus en fonction du nombre d'essais pour Wordle

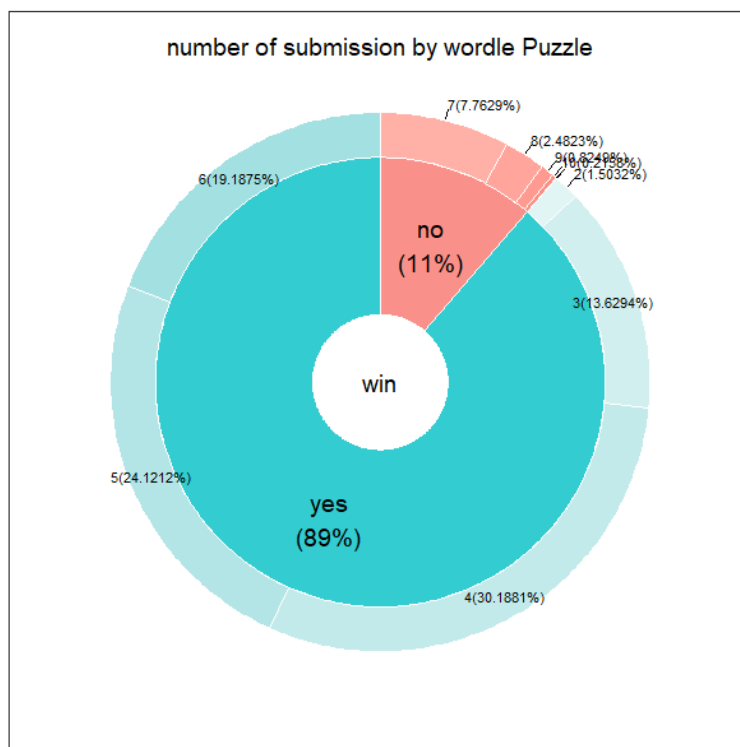


FIGURE 2.9 – Pourcentage de victoire du Wordle

2.5 Quordle

Je vais présenter dans cette partie les solutions que j'ai mises en place afin de pouvoir jouer des parties de Quordle avec le package Wordle de R évoqué plus haut.

2.5.1 Ajout des quatre parties consécutives

Le puzzle Quordle consiste à jouer quatre parties de Wordle simultanément. Pour cela, j'ai adapté le code de Louis afin d'avoir quatre Wordle qui se jouent en même temps. Le principal défi de cette partie était de réussir à faire jouer l'algorithme.

Puisque dans le code de Louis, le choix de la prochaine soumission dépend du dictionnaire des mots restants, il fallait faire un choix sur quel dictionnaire choisir parmi les quatre parties qui se jouent au même moment. Pour la première version, j'ai fait le choix de jouer les parties dans l'ordre. Ainsi, je garde la soumission par défaut qui est "cares", puis je résous la première partie et ainsi de suite.

```

> quordle_solving_algorithm()
c a r e s
b o o t y
p i n o n
n i n o n
[1] 4
n i n o n
a u l o i
a x i o m
[1] 6
a x i o m
r o i n s
[1] 7
r o i n s
r e b u s
[1] 8

```

FIGURE 2.10 – Résolution Quordle Première Version

Ainsi, on résout le premier mot, on affiche le nombre d'essais et on continue ainsi de suite. On remarque que le nombre d'essais n'est pas égal au nombre de ligne. Cela s'explique par le fait que le premier essai affiché d'un mot est la solution du mot précédent. On affiche cette ligne pour marquer le changement de mot à résoudre mais cela ne constitue pas un essai supplémentaire.

2.5.2 Cas de lettre unique

Pour ce qui est du cas des lettres uniques, je ne l'ai pas modifié puisque la résolution actuelle du Quordle consiste uniquement en quatre parties plus ou moins indépendantes si ce n'est les soumissions communes. Cependant, j'ai rencontré certains problèmes durant l'adaptation. En effet, dans la version de Louis, l'algorithme détermine s'il est nécessaire d'utiliser l'exception de lettre unique en fonction du Helper.

Cependant, sachant que nous avons quatre parties différentes, nous avons également quatre Helpers différents. Ainsi, j'ai du mettre en place une variable "Current_Helper", qui correspond au Helper du mot mystère que je résous actuellement. Cela permet donc de pouvoir utiliser le cas de lettre unique du Wordle.

2.5.3 Fonction de test

J’ai également eu à adapter une fonction de Louis permettant de jouer autant de parties de suite que voulu, afin d’obtenir une batterie de tests pour les diagrammes notamment. La fonction de base permettait de jouer une partie par mot dans le dictionnaire pour Wordle, puisque deux parties avec le même mot mystère se joueront de la même façon à chaque fois. Ainsi, cela lui permet d’obtenir un pourcentage de réussite dans lequel le facteur chance n’existe pas.

Pour adapter cette fonction, je n’ai pas pu faire une partie par grilles possibles de Quordle. En effet, avec un dictionnaire de 12’972 mots, il y aurait 12^{972^4} , ce qui représente évidemment beaucoup trop de parties jouables. Avec les conseils de Mme Onete, j’ai fait le choix de limiter à 10’000 parties. Ainsi le nombre de parties permet de minimiser le facteur chance et éviter d’avoir trop de variations d’un test à l’autre, mais également de limiter le temps de calcul. De cette façon, j’ai un temps de calcul d’environ une heure, ce qui est correct. Choisir plus de parties rendrait l’exécution de la fonction de test trop long et moins de parties accentuerait le facteur chance. C’est pourquoi j’estime que 10’000 parties est un nombre correct.

2.5.4 Sauvegarde des Données

Louis présentait également dans son code des fonctions permettant de sauvegarder les résultats de sa fonction de test et ainsi ne pas avoir à relancer son code à chaque fois.

Bien que les résultats peuvent légèrement changer d’un test à l’autre pour Quordle, j’ai tout de même adapter ses fonctions de lecture et de sauvegarde de données afin d’éviter de devoir relancer les fonctions de test trop souvent. Cela m’a permis de gagner un temps d’exécution précieux.

Chapitre 3

Implémentation et analyse des différentes versions

3.1 Première version

Dans cette partie, je vais développer les explications sur la première version de mon solveur pour Quordle. Je vais aborder comment je gère les quatre parties simultanées ainsi que le comptage des soumissions.

3.1.1 Procédé de résolution

Pour cette première version, j'ai fait le choix de résoudre les quatre mots dans l'ordre l'un après l'autre.

Après avoir entré le mot par défaut, qui est "cares", je vais ensuite me focaliser sur le premier mot mystère. Pour résoudre ce mot, je vais jouer une partie classique de Wordle. Grâce à l'adaptation du code de Louis pour Quordle, je peux résoudre cette première grille avec relativement peu d'essais. Ensuite, je vais changer de mot courant en résolvant le second mot de la même manière. Grâce aux indices donnés par le premier mot, la résolution du second est assez rapide. Enfin, je continue ainsi de suite pour les deux derniers mots.

3.1.2 Désavantages de la solution

Cette première version avait pour réel but une vérification de la fonctionnalité de mon solveur. J'ai pu, grâce à ce premier algorithme, voir que mon Quordle

fonctionnait correctement, mais également que ma base de solveur était limitée.

En effet, comme vous pouvez le voir ci-dessous, cette version du code ne me donne pas un pourcentage de réussite très convainquant, avec seulement 43%. Cependant, on peut voir des résultats encourageant. Assurément, on voit que près de 40% des parties se terminent avec dix ou onze essais. De ce fait, on voit que cette base d'algorithme est prometteuse pour la suite.

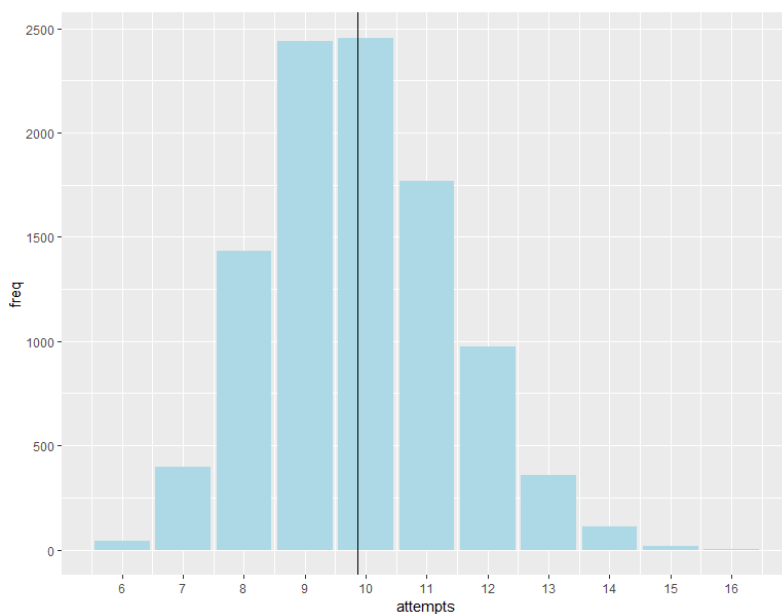


FIGURE 3.1 – Nombre de parties résolues en fonction du nombre d’essais de la première version Quordle

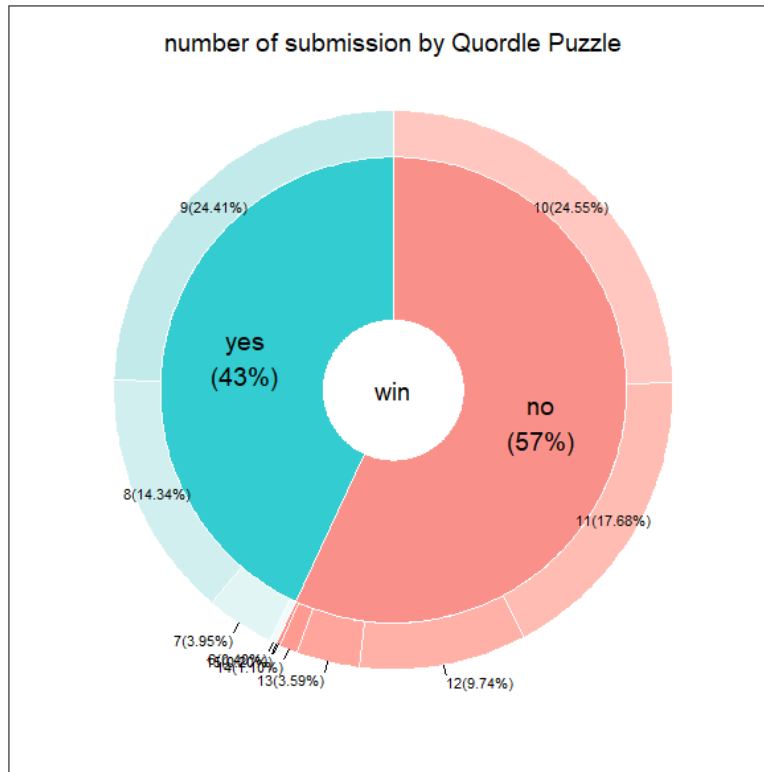


FIGURE 3.2 – Pourcentage de victoire de la première version Quordle

3.2 Seconde version

Je vais présenter dans cette partie la seconde version de mon solveur. Cette version a pour but de confirmer le fonctionnement de la précédente, mais également vérifier que notre Quordle fonctionne correctement.

Ainsi, on va ici jouer les parties dans un ordre aléatoire.

3.2.1 Ajout du principe d'aléatoire

Pour vérifier que les quatre mots de notre Quordle ne sont pas liés à leur placement, on va résoudre le puzzle en choisissant le mot à résoudre aléatoirement. Si tout fonctionne correctement, on devrait se retrouver avec des graphiques similaires.

Pour ce faire, j'ai fait le choix de stocker mes mots dans une liste, et de les retirer une fois qu'ils soient devenus "mot courant". Ainsi, une fois que le mot est résolu, je peux en choisir un autre dans la liste et ainsi de suite.

3.2.2 Problèmes rencontrés

Cependant, stocker les mots dans une liste peut présenter des limites. L'une d'entre elles qui m'a posé problème se trouve lors de la résolution involontaire d'un mot. En effet, en testant des soumissions pour trouver un mot, il se peut que l'on trouve un autre mot mystère sans le vouloir. Cependant, même s'il est résolu, ce mot se trouvera toujours dans la liste et pourrais donc être choisi ensuite comme mot courant.

Pour y remédier, j'ai du mettre en place une vérification afin de supprimer le mot de la liste s'il est résolu au hasard.

3.2.3 Analyse des résultats

Cette version qui n'apporte en soi pas d'amélioration au code se doit de donner les mêmes résultats que la précédente.

On voit donc ci-dessous que les graphiques sont très similaires. On trouve de petits changements dans les pourcentages, mais cela s'explique par le fait que les 10'000 parties de ce jeu d'essai sont différentes des 10'000 du précédent.

Ainsi, on peut en conclure que notre Quordle fonctionne correctement et que l'ordre de résolution n'impacte pas dans le cas d'une résolution indépendante et aléatoire.

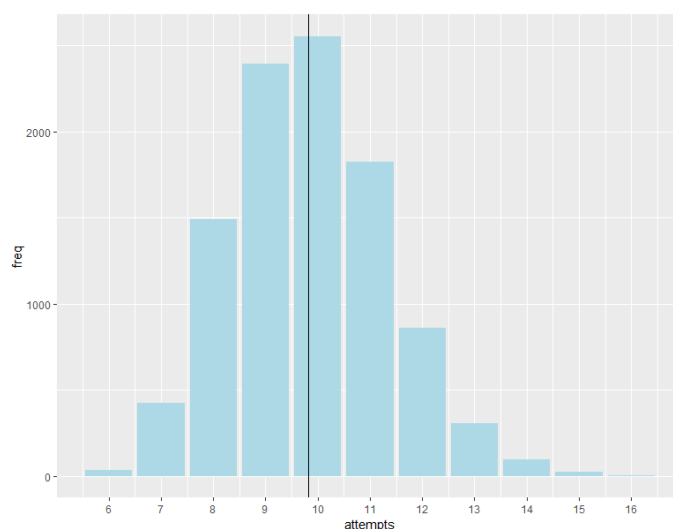


FIGURE 3.3 – Nombre de parties résolues en fonction du nombre d'essais de la seconde version Quordle

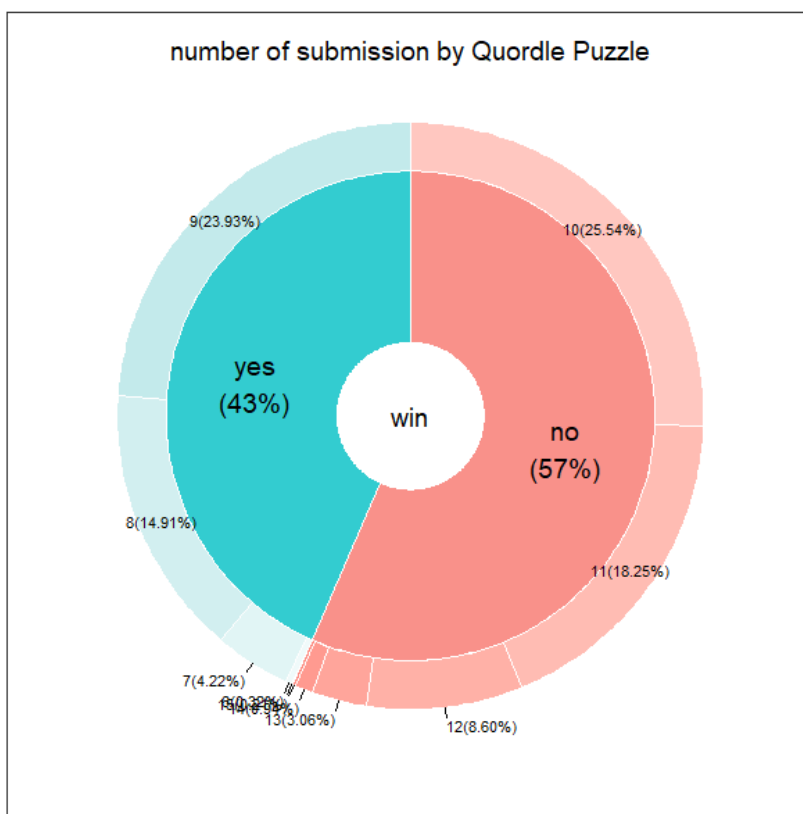


FIGURE 3.4 – Pourcentage de victoire de la seconde version Quordle

3.3 Troisième version

Cette version, qui est ma version actuellement au moment de la rédaction de ce rapport, vient apporter deux nouvelles fonctionnalités complémentaires.

3.3.1 Augmentation du nombre d'essais par défaut

La première fonctionnalité que j'ai ajouté dans cette version consiste à utiliser trois mots par défaut au lieu d'un seul. Le but ici est de tester plus de lettres avant de résoudre les mots mystères.

Le choix des trois mots par défaut pour démarrer la partie est un choix assez arbitraire. En effet, on peut choisir jusqu'à cinq mots par défaut, puisqu'il faut au minimum un essai par mot mystère pour pouvoir finir la partie. Cependant, de par la contrainte du temps de calcul et du nombre de combinaisons de cinq mots différents, je n'ai pas pu tester autant de possibilités que souhaité. Le choix de trois mots permet

d'avoir quelques soumissions en plus pour résoudre le puzzle dans les meilleures conditions, mais également car le choix des trois était plus simple, comme expliqué ci-dessous.

Les trois mots que j'ai choisi pour le moment sont "soare", "clint" et "dhows". J'ai choisi ces mots en lisant l'article "Solving N-dle using Information Entropy", rédigé par Dave Fetterman. Dans son article, il utilise une autre méthode pour jouer au Quordle. Dans sa version la plus concluante, il choisit deux mots pour commencer qui sont "soare" et "clint". Pour "Dhows", je me suis inspiré de l'article sur lequel se base l'article de Dave Fetterman, "On Optimal Strategies For Wordle", de Alexander D. Healy. Dans son article, il présente une version différente de celle de Louis pour Wordle, il teste avec le mot "dhows" en ouverture, avec lequel il obtient un pourcentage de réussite très élevé. De plus, "dhows" teste des lettres rares et donc peut donner des indices très importants pour la résolution des mots.

Ainsi, avec cette fonctionnalité, l'algorithme part de moins loin pour résoudre le premier mot, ce qui, comme on peut le voir ci-dessous, augmente considérablement le taux de réussite.

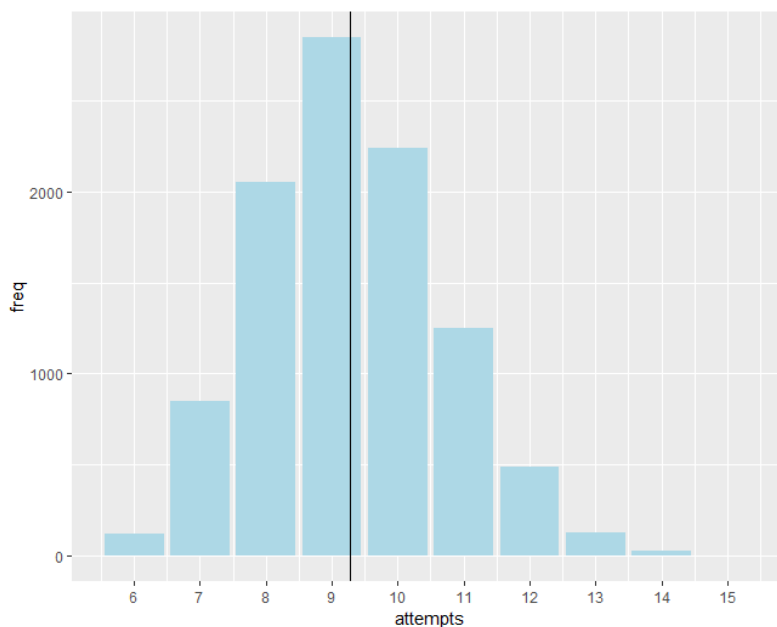


FIGURE 3.5 – Nombre de parties résolues en fonction du nombre d'essais de la troisième version Quordle

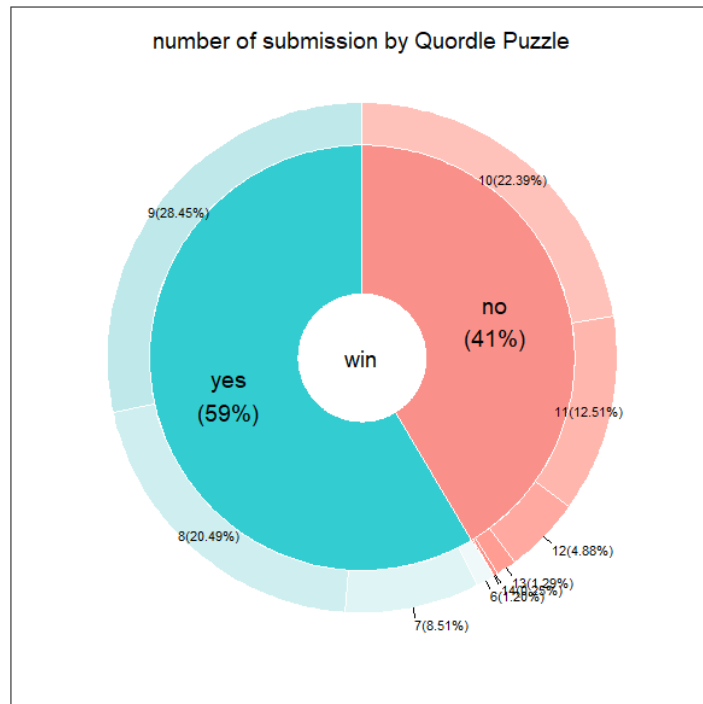


FIGURE 3.6 – Pourcentage de victoire de la troisième version Quordle

Cette fonctionnalité va permettre d'introduire la seconde amélioration de cette version.

3.3.2 Classement des mots mystères

Suite à l'amélioration du début de la partie évoquée précédemment, le second ajout consiste à choisir l'ordre de résolution des mots mystères en fonction de leur avancement. En effet, ici, on va classer les quatre mots mystères en fonction du nombre de mots possibles restants. Ainsi, on va résoudre les mots les plus avancés en premier et ainsi de suite.

Cette nouvelle fonctionnalité va nous permettre d'avancer tout en économisant les essais. Si cet ajout ne voit le jour qu'à cette étape du projet, c'est parce que cela aurait été difficile à mettre en place en ayant qu'un seul mot par défaut. Plus précisément, l'utilisation d'un seul mot par défaut n'apporte pas assez d'indices pour être suffisamment efficace sur un grille de Quordle.

La mise en place de cette fonctionnalité a été assez simple, puisque les Helpers liés aux quatre grilles de Wordle du puzzle contiennent un dictionnaire des mots

possibles restants. Ainsi, classer les mots mystères a été grandement facilité.

3.3.3 Problèmes rencontrés

Avec l'arrivée de cette nouvelle version, un problème précis est survenu. Dans le cas de lettre unique d'un mot mystère, si la lettre manquante des mots restants dans le dictionnaire du Helper est un doublon, l'algorithme n'arrive pas à créer un mot qui va donner des indices supplémentaires et donc retourner la même soumission à l'infini.

Par exemple, le mot mystère est "eater". Suite aux essais précédents, les mots possibles restants sont "eater", "rater" et "tater". Le problème dans ce cas de figure est que le -e, le -r et le -t sont déjà trouvés dans le mot mystère. Ainsi, notre algorithme va coincer car lorsqu'il testera ces trois lettres, il ne testera pas leur présence multiple dans le mot et donc n'obtiendra aucun indices supplémentaires.

```
> quordle_solving_algorithm("eater")
d h o w s
s o n i c
[1] 4
s o n i c
b a n d s
[1] 5
b a n d s
p a t e r
g e m o t
a f t e r
a f t e r
a f t e r
a f t e r
```

FIGURE 3.7 – Exemple du problème de lettres en doublons

Pour remédier à ce problème, j'ai fait le choix que lorsque la nouvelle soumission est identique à la précédente, le programme en conclu que nous sommes dans ce cas de figure. Ainsi, on va dans cette situation changer de mot courant et donc prendre le prochain mot dans la liste. Cela va permettre d'aller trouver des indices en résolvant les autres mots mystères.

Cette pratique est possible, puisque lors de mes tests, j'ai remarqué que ce problème n'arrivait jamais lors de la résolution du dernier mot. C'est-à-dire que

lorsque les trois premiers mots sont résolus, ce cas de figure n'existe plus. Ainsi, je n'ai plus rencontré ce problème depuis.

3.3.4 Analyse des résultats

Après avoir résolu tous les problèmes trouvés suite aux nouvelles implémentations, nous pouvons voir des résultats assez concluants. En effet, on voit une augmentation du taux de réussite de 10%, avoisinant ainsi les 70%.

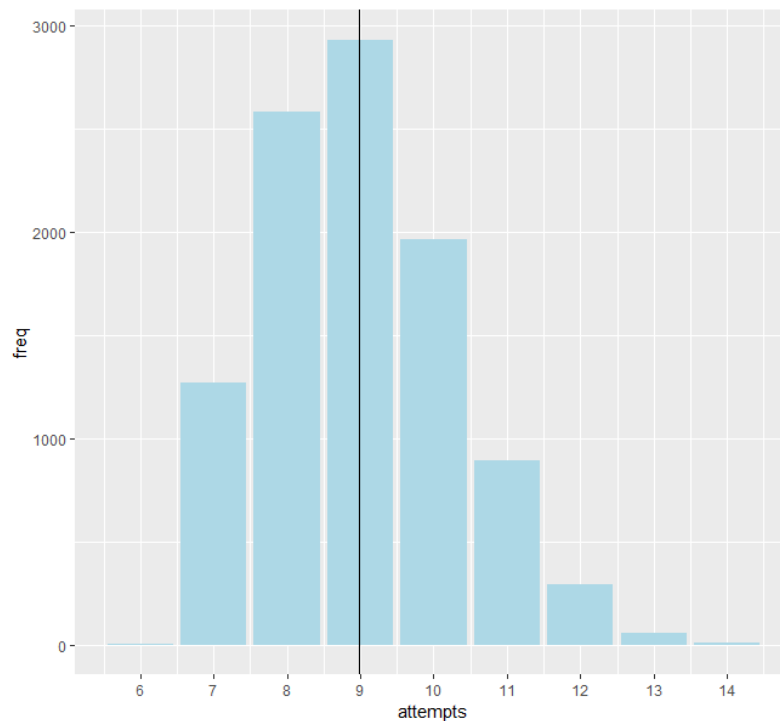


FIGURE 3.8 – Nombre de parties résolues en fonction du nombre d'essais de la dernière version Quordle

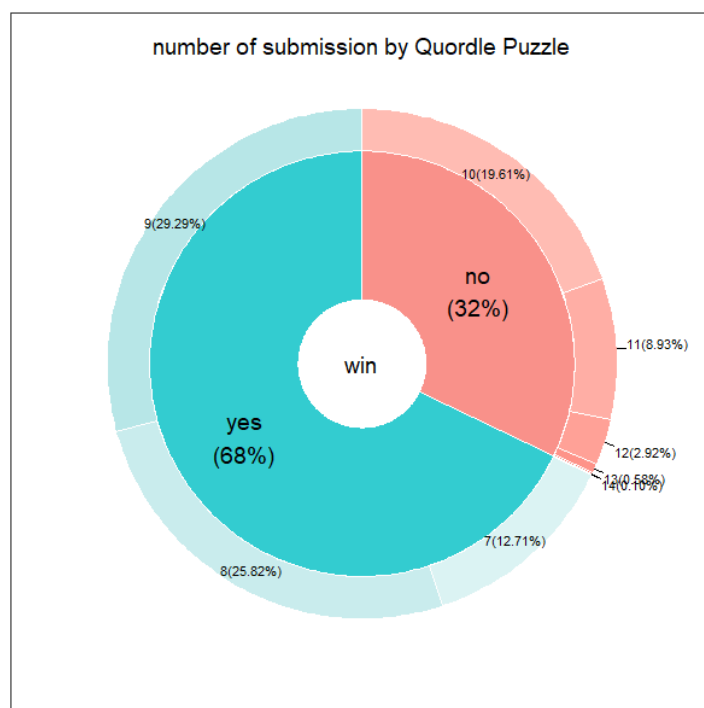


FIGURE 3.9 – Pourcentage de victoire de la dernière version Quordle

Ainsi, on voit que l'on se rapproche d'un algorithme avec un taux de réussite important. Sachant que nous utilisons un dictionnaire bien plus complexe que celui du jeu de base, on peut déduire que notre algorithme commence à être bon.

3.4 Essai avec le dictionnaire officiel

Dans cette partie, je vais utiliser le dictionnaire officiel du puzzle Wordle. Ainsi, il sera plus simple d'évaluer le taux de réussite de l'algorithme.

On voit donc qu'avec le dictionnaire officiel, le taux de réussite est autour des 80%. Ainsi, on se rapproche d'un taux très correct. Cependant, il reste encore de nombreux axes d'amélioration que je vais évoquer plus bas.

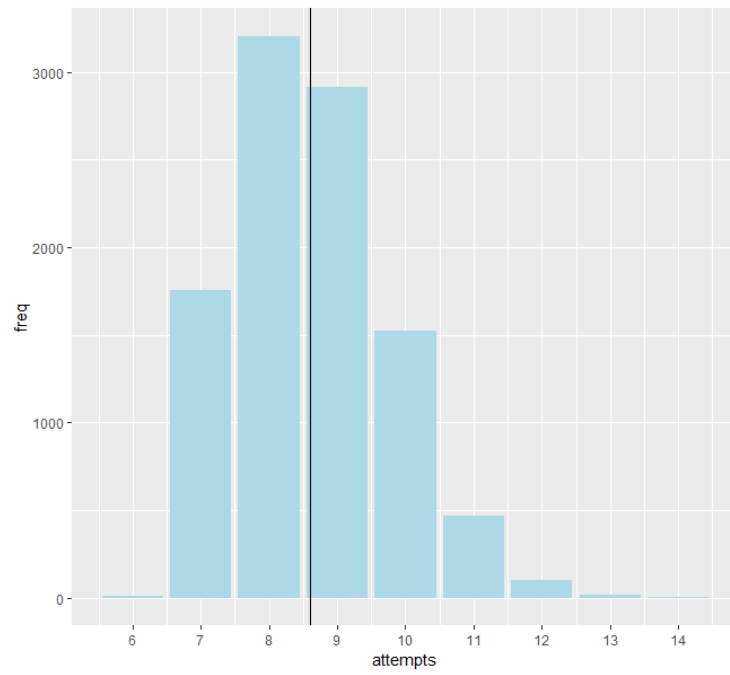


FIGURE 3.10 – Nombre de parties résolues en fonction du nombre d’essais avec le dictionnaire officiel

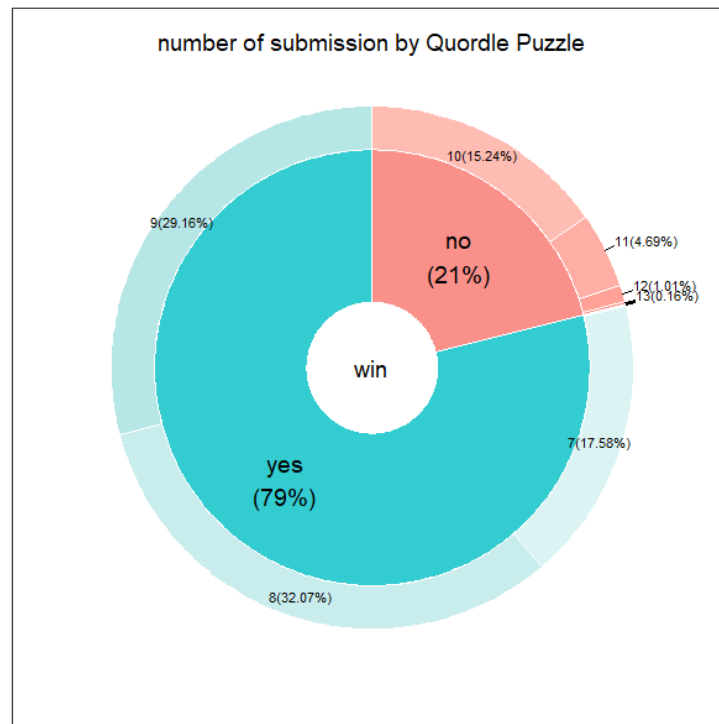


FIGURE 3.11 – Pourcentage de victoire avec le dictionnaire officiel

3.4.1 Ajout du dictionnaire officiel dans les Helpers

Ici, on va également utiliser le dictionnaire officiel des solutions du Wordle dans les Helpers. Ainsi, lors du cas classique, l'algorithme soumet uniquement des mots pouvant être solution. La seule exception faite est pour le cas de lettre unique. En effet, afin d'obtenir un mot qui peut tester le plus de lettres possibles, on va utiliser le dictionnaire du package Wordle de R. Ainsi, on a plus de chance d'avoir un mot qui teste beaucoup de lettres.

On voit donc ci-dessous qu'avec ce changement de dictionnaire, le pourcentage de réussite est très élevé. On obtient donc des résultats très satisfaisant pour le dictionnaire classique de Wordle.

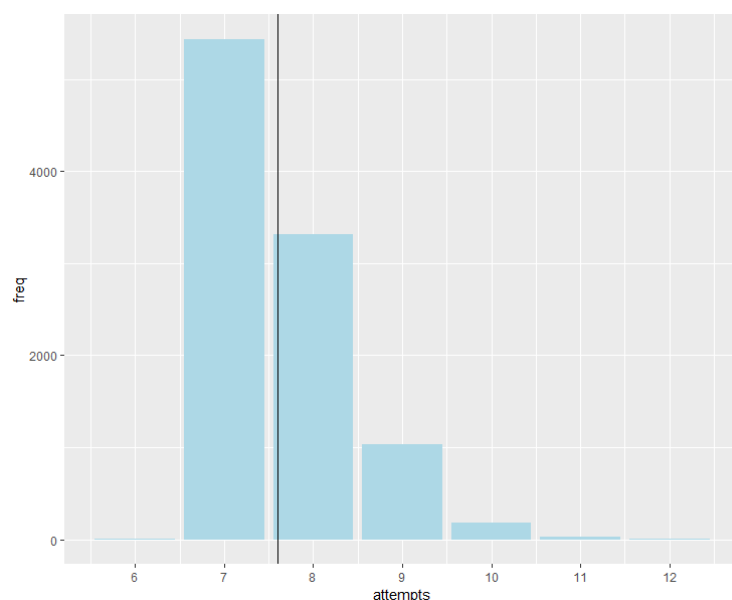


FIGURE 3.12 – Nombre de parties résolues en fonction du nombre d'essais avec le dictionnaire officiel dans le Helper

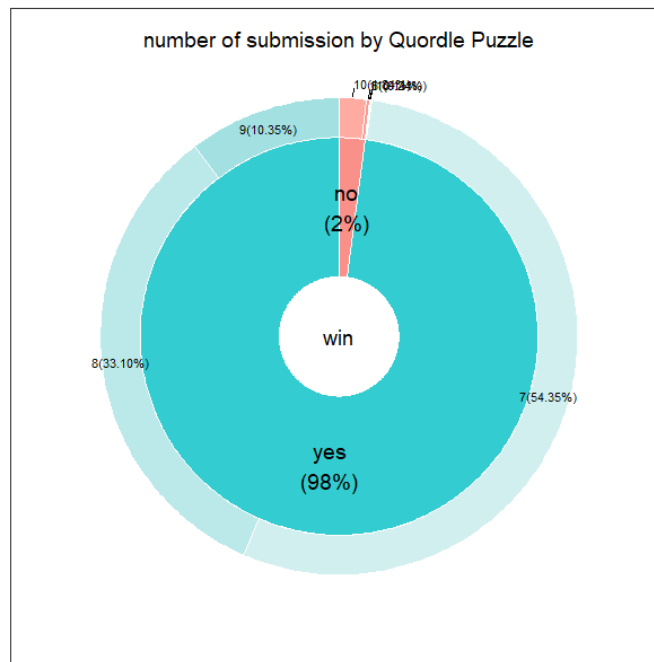


FIGURE 3.13 – Pourcentage de victoire avec le dictionnaire officiel dans le Helper

On voit donc que notre solveur est très bon pour jouer des parties de Quordle qui utilisent le dictionnaire officiel.

Chapitre 4

Pistes d'évolution

4.1 Mots par défaut

Une des améliorations qui pourraient apporter beaucoup est le changement des mots par défaut. Comme expliqué plus haut, par manque des tests, rien n'indique que les mots par défaut utilisés actuellement soient les plus optimaux, de même pour le nombre de mots. Ainsi, avec plus de temps et un environnement de test plus optimisé, on pourrait sûrement déduire un meilleur départ possible pour nos parties de Quordle.

4.2 Classement des mots

Le classement des mots peut également être amélioré. Ici, on classe en fonction du nombre de mots possibles restants. Cependant, il serait possible de prendre en compte les fréquences d'apparition des lettres également.

4.3 Généralisation du cas de lettre unique

Comme Louis l'a expliqué dans son rapport, il serait également possible, pour améliorer notre algorithme, de généraliser le cas de lettre unique. En effet, il serait judicieux de saisir des soumissions qui testent le plus grand nombre de lettres possibles restantes, sans se soucier des lettres déjà trouvée.

Si on généralise encore plus, on pourrait également créer les soumissions en fonction de chaque Helper, et ainsi tester le plus grand nombre de lettres.

Table des figures

1.1	Craiova, ville du Sud-Ouest de la Roumanie	7
1.2	Exemple de Grille de Wordle	9
1.3	Grille de Quordle	10
2.1	Classement de Langage selon différents organismes	12
2.2	Interface RStudio	12
2.3	Dictionnaire Wordle	14
2.4	Fréquence des lettres dans la langue anglaise	15
2.5	Exemple de cas de lettre unique	16
2.6	Solutions possibles restantes dans le dictionnaire	16
2.7	Grille Résolue	16
2.8	Nombres de mots résolus en fonction du nombre d'essais pour Wordle	17
2.9	Pourcentage de victoire du Wordle	18
2.10	Résolution Quordle Première Version	19
3.1	Nombre de parties résolues en fonction du nombre d'essais de la première version Quordle	22
3.2	Pourcentage de victoire de la première version Quordle	23
3.3	Nombre de parties résolues en fonction du nombre d'essais de la seconde version Quordle	24
3.4	Pourcentage de victoire de la seconde version Quordle	25
3.5	Nombre de parties résolues en fonction du nombre d'essais de la troisième version Quordle	26
3.6	Pourcentage de victoire de la troisième version Quordle	27
3.7	Exemple du problème de lettres en doublons	28
3.8	Nombre de parties résolues en fonction du nombre d'essais de la dernière version Quordle	29
3.9	Pourcentage de victoire de la dernière version Quordle	30
3.10	Nombre de parties résolues en fonction du nombre d'essais avec le dictionnaire officiel	31
3.11	Pourcentage de victoire avec le dictionnaire officiel	31
3.12	Nombre de parties résolues en fonction du nombre d'essais avec le dictionnaire officiel dans le Helper	32
3.13	Pourcentage de victoire avec le dictionnaire officiel dans le Helper	33

Conclusion

Dans ce mémoire, je présente le déroulé de mon stage. Je commente la réflexion autour de mon algorithme de solveur et explique mes choix.

Durant ce stage, j'ai expérimenté la recherche en Informatique. C'est un rythme de travail que j'ai beaucoup apprécié, relativement peu de temps à coder en tant que tel, mais beaucoup plus de réflexion.

Pour ce qui est du solveur, de nombreuses pistes d'amélioration sont encore disponibles pour se rapprocher de plus en plus de la perfection.

Bien que ce stage touche à sa fin, Mme ONETE et M. BĂDICĂ m'ont proposé de participer à la rédaction d'un article scientifique sur le sujet du stage. C'est une opportunité très intéressante que j'ai évidemment acceptée.

Au cours de ces dix semaines de stage, j'ai pu mettre à l'épreuve les compétences acquises lors de ces deux années de BUT Informatique. Ce stage en Roumanie était une expérience très intéressante, qui m'a fait grandir d'un point de vue professionnel et humain.

Glossaire

LaTeX : LaTeX est un langage de programmation permettant la saisie de texte à partir de balises et de macros.

R : R est un langage de programmation basé sur du C, orienté autour des probabilités et des statistiques.

Helper : un Helper est un objet présent dans le package Wordle de R. Son but est de stocker la liste de toutes les solutions possibles de la partie et de réduire cette liste à chaque soumission. Il est présent pour aider à la résolution.

indice : Un indice représente le résultat d'une soumission. Ici, chaque lettre de la soumission sera colorer pour nous indiquer son implication dans le mot mystère.

soumission : Une soumission est un essai que l'on va soumettre au Wordle. Ainsi, son but est de tester le mot mystère.

mot courant : Le mot courant est le mot principal choisi lors de la résolution du Quordle. Ce mot est celui sur lequel l'algorithme se concentre en particulier.

mot mystère : Un mot mystère est un des mots à trouver dans la grille de Quordle.

dictionnaire : Un dictionnaire est une liste de mots. Ici, on utilise comme dictionnaire la liste des mots possibles en tant que mot mystère.

Sources

Louis Monié. *Mémoire de stage 2022*

Alexander D. Healy. *On Optimal Strategies For Wordle*

[http ://www.alexhealy.net/papers/wordle.pdf](http://www.alexhealy.net/papers/wordle.pdf)

Dave Fetterman. *Solving N-dle using Information Entropy*

[http ://fettermania.com/math/ndle.pdf](http://fettermania.com/math/ndle.pdf)