Rapport IN104 - Wordle

Arthur Paradis et Matthieu Varenne

May 23, 2022

1 Introduction

Le projet Wordle est un travail qui a deux objectifs principaux : dans un premier temps, coder le jeu Wordle original en français qui puisse interagir avec un utilisateur avec des mots de 5 lettres uniquement.Puis dans un second temps, créer un résolveur qui puisse jouer le "meilleur mot possible" afin de trouver le plus vite possible le mot à trouver. Différentes options pouvaient également être ajoutés comme un mode difficile ou bien jouer avec des mots dont le nombre de lettre est différent de 5.

2 Implémentation du jeu Wordle

2.1 Choix du dictionnaire

Pour commencer le projet, il était nécessaire de charger un dictionnaire que nous voulons être sous la forme d'un tableau de chaînes de caractères (fonction *chargerdico* dans le fichier charger dico.c). Après des essais avec quelques dictionnaires, nous avons remarqué que les caractères accentuées étaient stockés avec des caractères étranges. On a donc récupéré un dictionnaire sans accents sur Internet. Dans la fonction *chargerdico* on a ensuite gardé uniquement les mots de 5 lettres.

A la suite de ça, nous avons crée un fichier "tentative-valide.c" comportant deux fonctions : l'une vérifiant si le mot est de la bonne longueur, l'autre vérifiant si le mot est bien dans le dictionnaire.

2.2 Affichage des indices

IL nous faut aussi trouver un moyen de renseigner les indices à l'utilisateur à chaque tour. Le fichier "afficher-indice.c" a été crée pour regrouper toutes

les fonctions utiles à l'affichage des indices. On a décidé de stocker les indices dans un tableau d'entiers de taille le nombre de lettres et dont les éléments valent : 0 si on a la bonne lettre à la bonne place, 1 si on a la bonne lettre dans le mot mais au mauvais endroit et 2 si la lettre n'est pas au bon endroit. Ce choix a été fait en prévision de la seconde partie où nous devons coder un résolveur, il sera alors nécessaire de stocker les informations des indices puis les afficher plutôt de les afficher directement. On a alors codé une fonction *indices* prenant en argument le mot à deviner et le mot tenté et renvoyant un tableau des entiers correspondants aux indices. Les premier et dernier cas ont pu être traités directement, en revanche le cas de la bonne lettre au mauvais endroit nous as posé plus de difficultés.

Le problème qui s'est posé est le suivant : si par exemple le mot à trouver comporte une fois la lettre 'a' et le mot tenté deux fois aux mauvaises places, les deux emplacements recevaient un 1. Or dans ce cas c'est uniquement le premier 'a' qui doit recevoir un 1, et l'autre un 2. Ce qui nous as amené à créer la fonction $tab - occ$ qui crée une table d'occurrences du mot. Nous avons ensuite pu obtenir le résultat souhaité en intégrant un calcul de la table d'occurrences du mot dans la fonction *indices*.

2.3 Création du jeu dans le main

Nous devons alors combiner tous les fonctions des différents fichiers crées pour les utiliser dans un fichier "jeu.c" intégrant un main.

On charge d'abord le dictionnaire utilisé à l'aide de la fonction *chargerdico*. Ensuite on commence une boucle de 6 tours correspondant à chaque essai. Au début du tour,l'utilisateur effectue une tentative. On vérifie ensuite si cette tentative est valide : si le nombre de mots est respecté et le mot existe bien. Dans le cas contraire, un message indique que la tentative n'est pas valide et le tour n'est pas compté. Si la tentative est bien valide, on affiche les indices et le tour est compté. Si le mot est trouvé, la partie s'arrête et un message indique la victoire. Au contraire, un message annonçant la défaite s'affiche si le mot n'est pas trouvé au bout de 6 tours.

3 Résolveur du Wordle

Le but du résolveur Wordle est de proposer à chaque tour au joueur un mot qui doit être le "meilleur mot à jouer" compte tenu des informations connues par le joueur.

3.1 Stockage des informations recueillies lors de la partie

Afin de pouvoir chercher le meilleur mot possible en fonction des données connues, nous avons décidés de stocker les données dans un tableau de taille $26 * n$, où n est le nombre de lettres du mot à deviner. Elle donne pour chaque position de lettre dans le mot et pour chaque lettre de l'alphabet si une lettre est, n'est pas ou peut être à une certaine position dans le mot.

3.2 Actualisation du dictionnaire

Les données récupérées à partir de chaque itération vont se révéler très utiles. En effet le but est d'éliminer le plus de mots possibles après chaque tour afin de faciliter la recherche et le calcul du meilleur mot. On crée alors un nouveau fichier actualise-dico.c" où il y aura notamment une fonction *motvalide* vérifiant si un mot est valide ou non d'après les données actualisés, et une fonction $actualise - dico$ qui renvoie un nouveau dictionnaire composés des mots jouables.

3.3 Recherche du meilleur mot

Une première notion qu'il est nécessaire de préciser est la notion de "meilleur mot jouable". Comment savoir quel est le meilleur mot jouable? Le "meilleur mot jouable" est le mot qui apporte le maximum d'information au joueur en moyenne sur l'ensemble des réponses possibles que peut renvoyer le Wordle au joueur. Afin de quantifier la notion d'information nous avons utilisé la théorie de l'information et plus précisément l'entropie de Shannon qui a l'expression suivante pour un mot M du dictionnaire.

$$H(M) = \sum_{réponse du Wordle} -p_{rep} ln(p_{rep})$$

, où p est la proportion de mot du dictionnaire encore possible pour la réponse du Wordle lorsque M est joué.

On utilise la fonction $actualise - dico$ afin de simuler pour chaque mot l'actualisation du tableau de données.

On sélectionne par la suite de jouer le mot pour lequel l'entropie calculée est maximale. La fonction qui réalise cette tâche est la fonction $best - word$.

Elle stocke dans un tableau de la même taille que le dictionnaire l'entropie moyenne associée à chaque mot et en choisi le maximum. La fonction $best - word$ est coûteuse en temps de calcul car elle est en $O(3^n \times Taille_{Dictionnaire})$, où n est le nombre de lettres du mot à deviner.

Dans un premier temps, les réponses possibles données par le Wordle étaient générées par 5 boucles for lorsque le nombre de lettres était fixé à 5. Par la suite, pour permettre d'utiliser le résolveur pour un nombre quelconque de lettres, nous avons décidé de générer les 3^n réponses possibles du Wordle en utilisant une décomposition en base 3.

4 Fonctionnalités supplémentaires

4.1 Choix du nombre de lettres

Dans le but d'offrir à l'utilisateur plus de possibilités de jeu, nous avons décidé d'adapter notre code en permettant à l'utilisateur de choisir le nombre de lettres avec lequel il veut jouer.

Ce que nous avons fait en passant ainsi le nombre de lettres en paramètre de toutes les fonctions déjà crées, ainsi que la taille du dictionnaire correspondant (ces deux variables étaient anciennement des variables globales et donc fixés).

Nous modifions également le main pour que l'utilisateur puis renseigner le nombre de lettres choisis avant chaque tour.

4.2 Ajout d'un mode difficile

Pour corser la difficulté, nous nous sommes inspiré du Wordle original en ajoutant un mode difficile : dans ce mode il faut tenir compte des indices déjà dévoilés et ne jouer que des mots jouables. De plus l'utilisateur n'a plus accès à l'aide de l'ordinateur .

Il nous suffit alors de rajouter une condition dans le main, et d'utiliser la fonction $verifie - dico$ non pas avec le dictionnaire chargé au début, mais avec le dictionnaire actualisé pour chaque tour.

Nous avons fait en sorte que l'utilisateur puisse choisir de prendre le mode difficile en début de partie. Également, s'il ne choisit pas le mode difficile il a toujours la possibilité de choisir l'aide de l'ordinateur ou non.