Compte rendue Projet Programmation

Groupe H: Paul Beziau
Clément Nerestan
Chloé Pathé

21 avril 2015

Table des matières

Chapitre 1

Introduction

Ce projet a pour but de développer le jeu 2048. Pour cela, on utilisera les librairies NCurses et Simple directmedia layer (SDL), ainsi que les librairies natives de C. Le 2048 se joue sur une grille de 4x4 tuiles(on appelera tuile une case ainsi que la valeur qui lui est affectée). Ces dernières ont chacune pour valeur une puissance de 2. Les tuiles sont déplacées selon les quatre directions grâce aux touches fléchées : gauche, haut, bas, droite. Lorsque deux tuiles de même valeur 2^n sont assemblées lors d'un mouvement, une tuile de valeur $2^(n+1)$ est créée. A la fin d'un mouvement, une tuile de valeur 2 ou 4 est placée de façon aléatoire sur une case vide (une chance sur dix d'obtenir une tuile de valeur 4). Le but du jeu est d'obtenir une tuile de la plus grande valeur la plus grande possible.

Chapitre 2

Projet

2.1 Le Problème

Notre projet est effectué en deux parties : la première consiste en l'implémentation du jeu 2048, la seconde en la création de stratégies permettant de gagner le plus souvent possible. Le jeu doit être jouable avec les touches fléchées du clavier. Il y a donc plusieurs problèmes à régler. Il faut recréer le jeu : la grille, les tuiles, la gestion de leur déplacements et de leurs fusions, les conditions de déplacement et de fin de partie ; il faut aussi implémenter la gestion du clavier. Une fois tout cela terminé, il faudra implémenter des stratégies viables.

2.2 L'implémentation du jeu

Notre jeu est jouable au clavier selon deux modes. Le premier se situe directement dans la console, le deuxième dispose d'une interface graphique.

2.2.1 Le Jeu

La taille de la grille est une constante définie en dehors des fonctions utilisées. La modification de cette variable n'affectera donc pas le fonctionnement du jeu. Cependant, sa valeur a été fixée à 4, afin de respecter le jeu original. La valeur des tuiles est gérée selon leur puissance de 2. Ainsi, un 2 vaudra 1, un 4 sera 2, ainsi de suite. On leur rendra leur vraie valeur à l'affichage.

Dans la première version de notre code, cette gestion des tuiles posait problème en cas de 0. En effet, 2 puissance 0 fait 1, et non pas zéro. Nous avons donc décidé qu'en cas de 0, la tuile affichée serait une case vide.

Le déplacement des tuiles selon une direction se fait en plusieurs étapes. On vérifie d'abord que le mouvement est possible. Un mouvement est possible si :

1) Au moins une case sur le bord vers lequel on déplace les tuiles est libre.

2) Si on fusionne des tuiles dans la direction demandée ; et donc qu'on libère un ou plusieurs case.

Si aucune de ces deux conditions ne sont remplies, le mouvement est impossible et ne sera pas réalisé.

Après cette vérification, on tente de décaler les tuiles dans la direction voulue. La fusion entre deux tuiles de même valeur se fait après ce décalage. Au cours de la fusion, le score est implémenté de 2 puissance la tuile créée. Après cette fusion, un nouveau décalage est effectué. Cependant, une seule fusion est effectuée. Ainsi quatre tuiles de même valeur N ne formeront que deux cases de valeur N+1, et non une seule case de valeur N+2. Pour obtenir cette case, un nouveau mouvement sera requis. A la fin de chaque mouvement, une tuile de valeur 2 (1) ou 4 (2) est placée aléatoirement dans une des cases vides de la grille. Il y a une chance sur 10 qu'une tuile de valeur 4 soit créée.

Quand aucun mouvement n'est possible, la partie est finie.

Afin d'éviter les duplications de code, nous avons décidé de n'effectuer la vérification des mouvements, les fusions et le décalage des tuiles dans un seul sens : vers le haut. Pour cela, nous faisons tourner la grille dans le sens antihoraire. La grille tournera d'un cran pour un mouvement vers la gauche, de deux pour un mouvement vers le bas, et de trois crans pour un mouvement vers la droite. Une fois les différentes étapes du mouvement effectué, la grille tourne de nouveau afin à se replacer dans sa position originelle.

2.2.2 Jouer par le biais de la console

L'affichage du jeu par la console se fait via NCurses. A chaque mouvement, la grille est effacée et recréée. La véritable valeur des tuiles est affichée à ce moment-là.

Si un «game over» est obtenu, c'est-à-dire, si le jeu est terminé car plus aucun mouvement n'est possible, le programme se coupe, et l'affichage de la grille et du score obtenu perdure pendant quelques secondes.

2.2.3 Jouer avec l'interface graphique

L'interface graphique est gérée par la librairie SDL. Afin de rendre le jeu lisible, plus la valeur d'une tuile est composée de nombres, plus la taille de la police sera faible. Ainsi, 2048 est écrit en plus petit que 64, et 64 est écrit plus petit que 2, par exemple. De plus, les tuiles ont des couleurs différentes selon leur valeur. Nous pouvons aussi joué à la souris.

L'interface fait 400x450 pixels, chaque tuile fait 100x100 pixels.

Nous avons aussi ajouter de la musique qui se lance et qui s'arrête en même temps que le jeu. La seule véritable difficulté de cette partie a été de comprendre comment utiliser la librairie SDL.

Chapitre 3

Les stratégies

3.1 Stratégie "ExpectedMax"

Cette stratégie est similaire aux coups d'un joueur de dame ou d'échec. L'algorithme «regarde» trois coups en avant. Pour ce faire, on crée une nouvelle grille sur laquelle les différentes possibilités de mouvements. Ces mouvements sont évalués grâce à un entier. Plus cet entier est grand, plus le mouvement est rentable. Cet entier est incrémenté en fonction du score effectué lors du mouvement, du nombre de bord que la tuile la plus élevée touche et l'homogénéité de la grille, c'est-à-dire si les tuiles autours de la tuile maximum ont des valeurs proches.

Anticipant le fait que le score influencerait de plus en plus la décision de l'algorithme, nous avons décidé d'attribuer une très grande valeur au placement des tuiles sur les bords. Cette valeur est une constante attribuée de façon arbitraire, après de nombreux tests.

Par défaut, l'algorithme n'a qu'une profondeur de trois et cela pour des questions de temps d'exécution. Cependant, cette profondeur peut être changée.

3.2 Statégie "Fast"

La stratégie que nous avions privilégiée était la suivante : l'algorithme essaye de déplacer les tuiles vers la gauche à chaque coup. Si un mouvement est possible et que cette direction n'est pas disponible, il essaye d'aller en haut. Si haut n'est pas un mouvement possible, la direction essayée est le bas.

Et si cette troisième direction n'est pas disponible non plus, le mouvement s'effectue vers la droite.

Cependant, cette stratégie n'était pas satisfaisante. En effet, les tuiles maximum, au moment de perdre, oscillait entre 256 et 512. Nous avons donc décidé de reprendre la stratégie «ExpectedMax», cette fois-ci avec une profondeur de 2.

Chapitre 4

Critique

Le principal problème que nous avons rencontré est un problème d'organisation. En effet, nous avons la plupart du temps travaillé en même temps sur les mêmes fichiers. Même si nous ne travaillions pas sur les mêmes fonctions, les différents «commit» sur GitHub entrainaient des duplications ou la suppression de certains morceaux de code.

De plus, le fait de tous travailler sous des systèmes d'exploitation différents nous ont obligé de trouver des astuces pour pouvoir compiler les différentes librairies. Nous avons décidé de passer par un système d'exploitation Unix, plus pratique.

Chapitre 5

Bibliographie

Sujet du Projet : http://dept-info.labri.u-bordeaux.fr/ENSEIGNEMENT/edd/projet/sujet.pdf

2048 game : http://gabrielecirulli.github.io/2048/

Doc SDL : http://www.libsdl.org/release/SDL-1.2.15/docs/

Doc SDL-ttf : http://www.libsdl.org/projects/SDL_ttf/docs/SDL_ttf.pdf

Chapitre 6

Annexes

Construction des exécutables La construction des exécutables se fait de la façon suivante :

1. A partir de la console et dans le dossier ProjetProgS4, faire « cmake.»
2. Après la génération du makefile, faire make.
3. Un dossier «bin» a été créé, s'y déplacer.
4. Faire «./ nomdel'exécutable». Les différents éxecutables sont

a. «2048» qui correspond au 2048 en mode console.

b. «2048-gui», qui est le 2048 avec l'interface graphique.

c. «strat-main», qui permet de lancer les stratégies en mode console.

d. «test-grid», qui correspond aux tests de la grille et des stratégies