UE – Structures de Données

Jeanne Barriere – Chifaa Wehbe

sorbonne universite | 2019 - 2020

Projet Flood-It

Rendu intermediaire

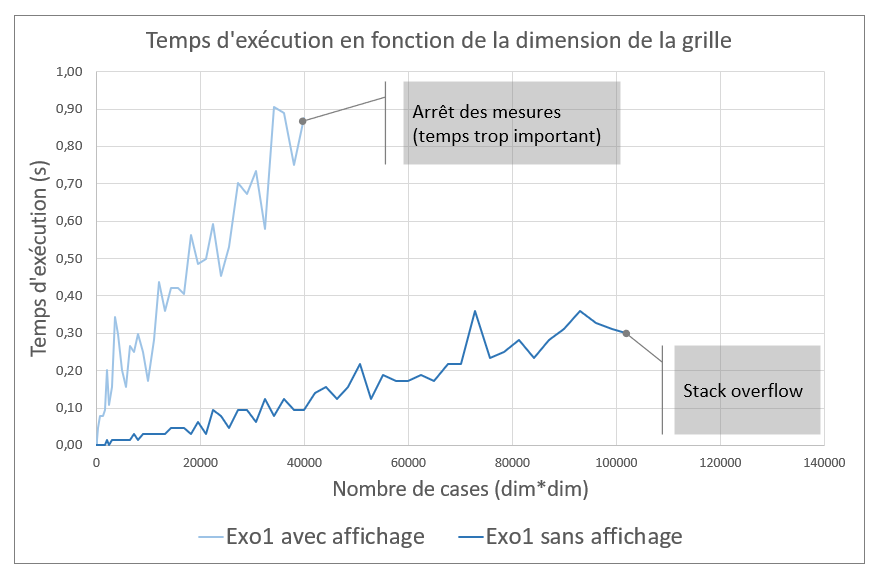


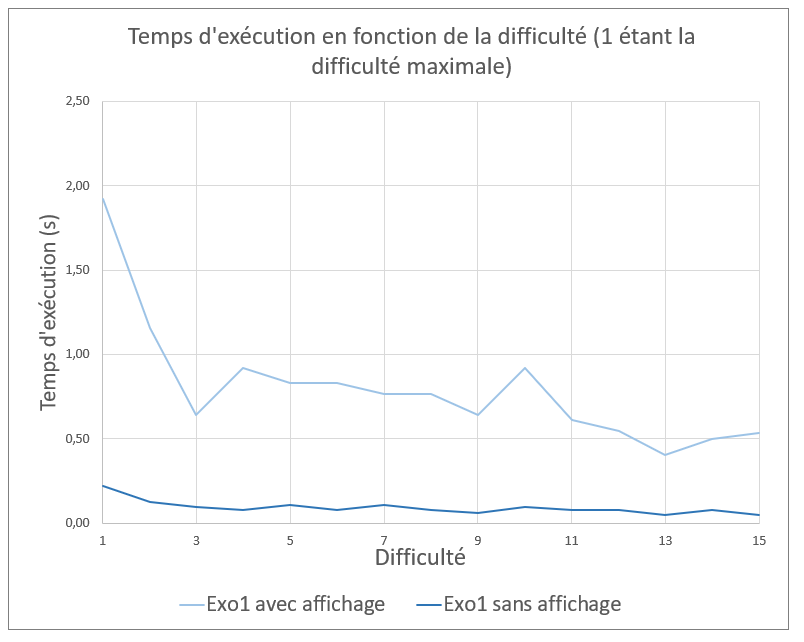
## Introduction

Notre travail a été fait sous Windows, avec l’application Ubuntu pour Windows pour la compilation. Les tests avec affichages ont été effectués en mode texte directement dans l’application Ubuntu. Nous avons travaillé et élaborer le code conjointement via git.

## Question 1.3

Le graphe ci-dessous présente les temps d’exécution du programme avec la fonction récursive en fonction de la dimension de la grille. Il a été réalisé avec 5 couleurs, un niveau de difficulté à 3, une graine à 2. La dimension de la grille a été augmentée de 5 après chaque mesure. Les mesures avec affichages ont été arrêtées à une dimension de 200\*200 car le temps de mesures devenait trop long.

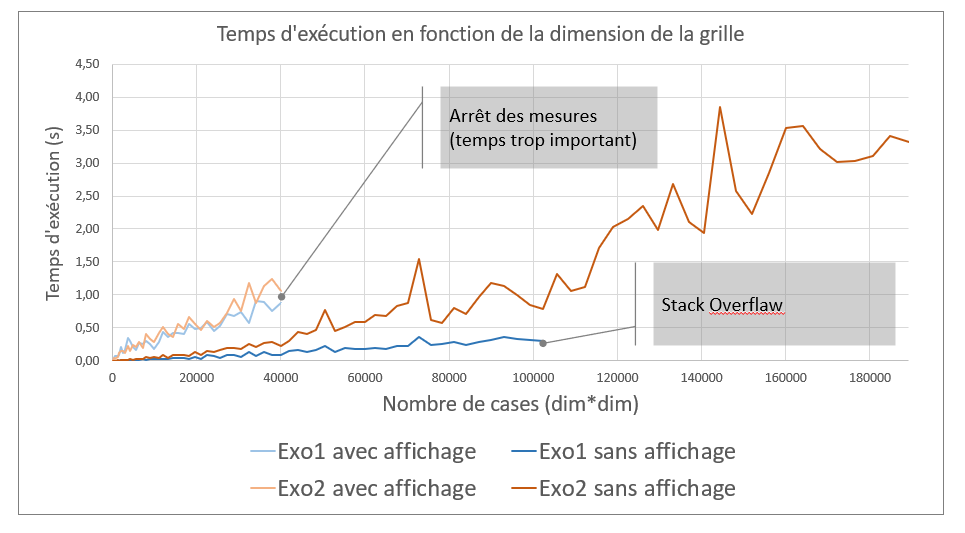
On peut voir sur le graphe que plus la grille est grande, plus le temps d’exécution est important. De plus, le temps d’exécution est plus important quand l’affichage est activé. Enfin, on observe un Stack Overflow pour une taille de grille de 325\*325. On constate donc que l’emploi de la récursivité limite la taille de la grille, car trop d’appels récursifs surchargent la pile d’exécution et conduisent à l’arrêt prématuré du programme.



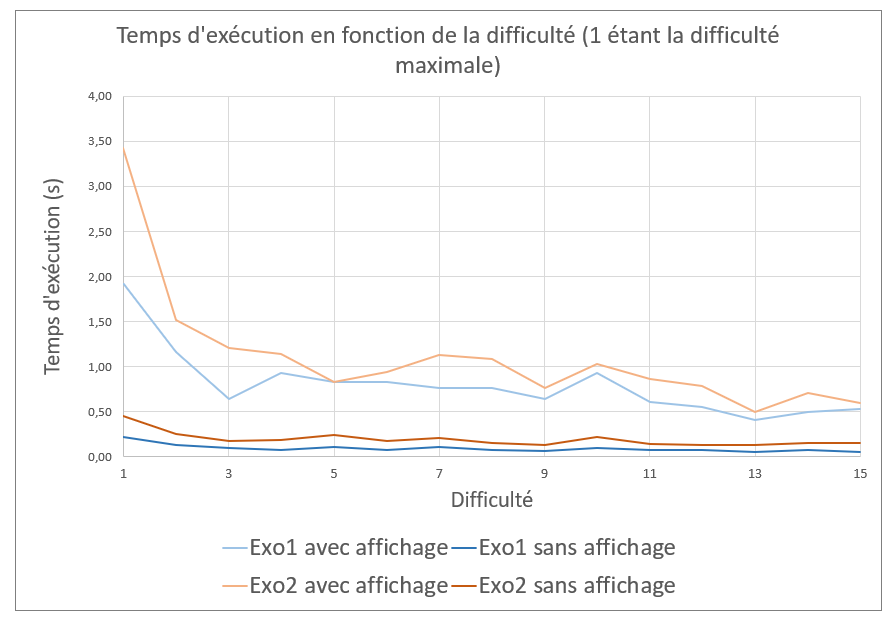
Dans le graphe ci-contre on a mesuré le temps d’exécution en utilisant la fonction récursive pour différentes difficultés, sachant que 1 est la difficulté maximale. Il a été réalisé avec 5 couleurs, une dimension de la grille de 200\*200 et une graine à 2. La difficulté est augmentée de 1 à chaque mesure.

On observe que le temps d’exécution augmente avec la difficulté (ainsi que le nombre d’essais nécessaires pour inonder l’ensemble de la grille). De même que précédemment, les temps sont plus importants quand on utilise l’affichage.

## Question 2.2

Le graphe ci-dessous présente les temps d’exécution du programme avec la fonction récursive et avec la fonction dérécursifiée, en fonction de la dimension de la grille. Il a été réalisé avec 5 couleurs, un niveau de difficulté à 3, une graine à 2. La dimension de la grille a été augmentée de 5 après chaque mesure. Les mesures avec affichages ont été arrêtées à une dimension de 200\*200 car le temps de mesures devenait trop long.

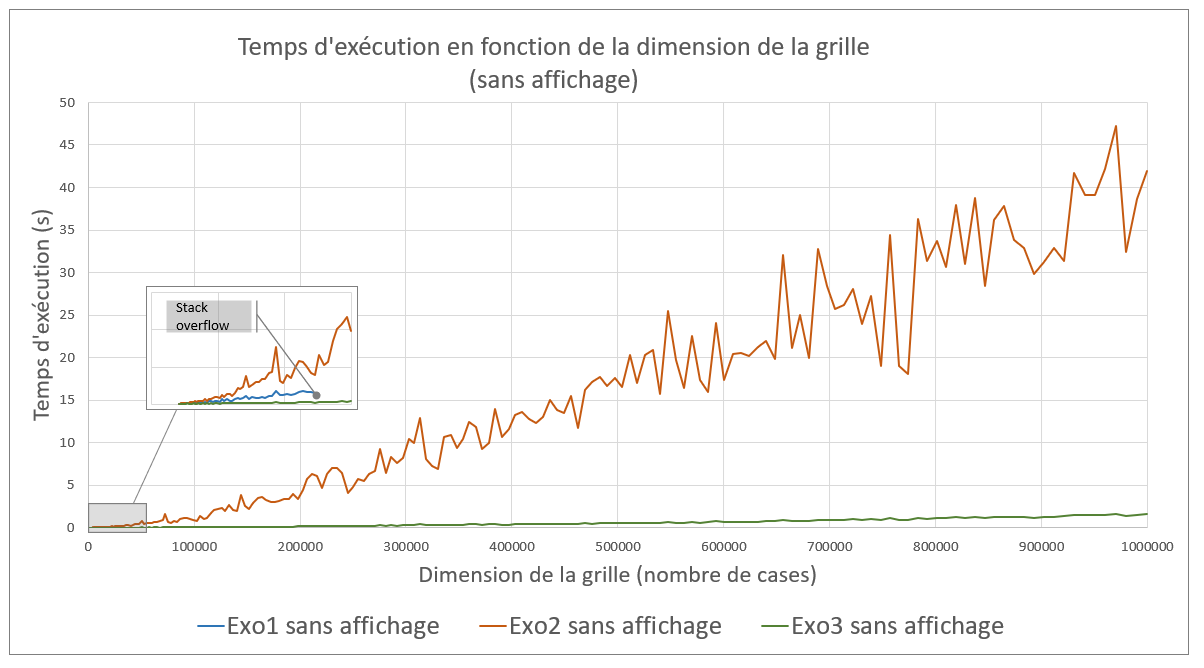
On obtient des résultats semblables pour la fonction dérécursifié à ceux obtenus avec la fonction récursive. Le temps d’exécution augmente avec la taille de la grille et est plus important lorsqu’on affiche la grille. On remarque cependant que les temps d’exécution sont plus importants avec la version dérécursifié qu’avec la version récursive. De plus, on n’observe plus de Stack Overflaw pour la version dérécursifié.

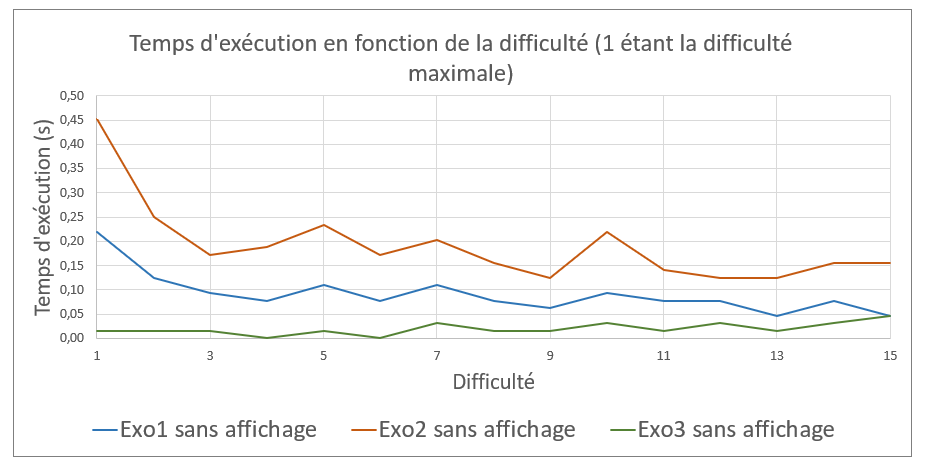
Dans le graphe ci-contre on a mesuré le temps d’exécution en utilisant la fonction récursive et la fonction dérécursifié, pour différentes difficultés, sachant que 1 est la difficulté maximale. Il a été réalisé avec 5 couleurs, une dimension de la grille de 200\*200 et une graine à 2. La difficulté est augmentée de 1 à chaque mesure.

Comme pour le graphe précédant, les résultats pour la fonction dérécursifié à ceux obtenus avec la fonction récursive. De plus, on remarque encore que l’utilisation de la fonction dérécursifié entraîne une exécution du programme plus longue.

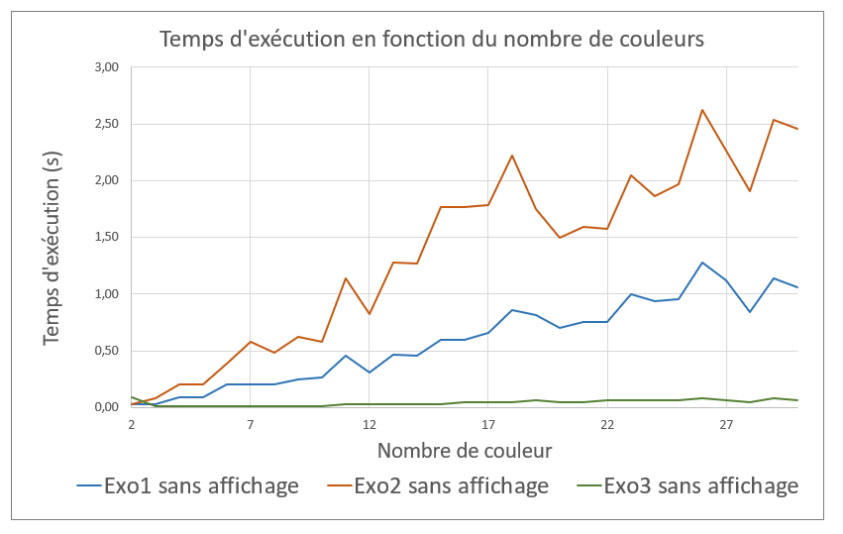
# III – Structure acyclique (Allons plus vite)

Le graphe ci-dessous présente les temps d’exécution du programme avec les fonctions des trois exercices, en fonction de la dimension de la grille. Il a été réalisé avec 5 couleurs, un niveau de difficulté à 3, une graine à 2. La dimension de la grille a été augmentée de 5 après chaque mesure.

On voit que la version de l’exercice 3 permet une exécution beaucoup plus rapide que les deux autres versions et ne provoque pas d’erreurs contrairement à celle de l’exercice 1 (testé jusqu’à 1000\*1000 cases).



Dans le graphe ci-contre on a mesuré le temps d’exécution pour chacune des versions du code, pour différentes difficultés, sachant que 1 est la difficulté maximale. Il a été réalisé avec 5 couleurs, une dimension de la grille de 200\*200 et une graine à 2. La difficulté est augmentée de 1 à chaque mesure.

Comme pour le graphe précédant, les temps d’exécution sont plus faibles pour la version de l’exercice 3.

Dans le graphe ci-contre on a mesuré le temps d’exécution pour chacune des versions du code, pour un nombre croissant de couleurs. Il a été réalisé avec une difficulté de 3, une dimension de la grille de 200\*200 et une graine à 2.

De même que dans les graphes précédants, on observe que la troisième version s’exécute beaucoup plus rapidement que les deux autres versions.

Pour conclure, la version du programme utilisant une structure acyclique est beaucoup plus rapide que les deux autres versions. De plus, la version récursive est elle-même plus rapide que la version dérécursifié. Cependant cette dernière version permet d’éviter de surchager la pile d’éxécution, contrairement à la version récursive.