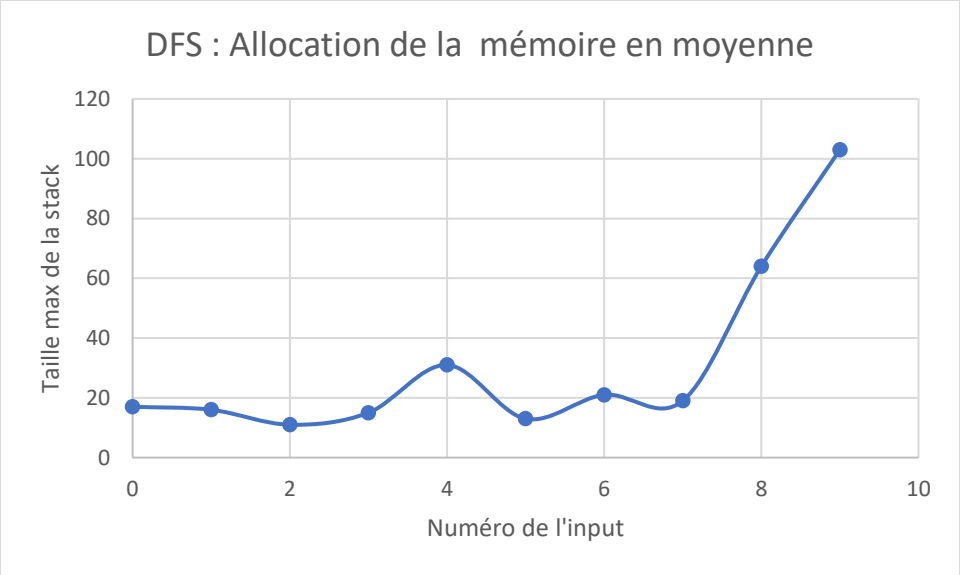
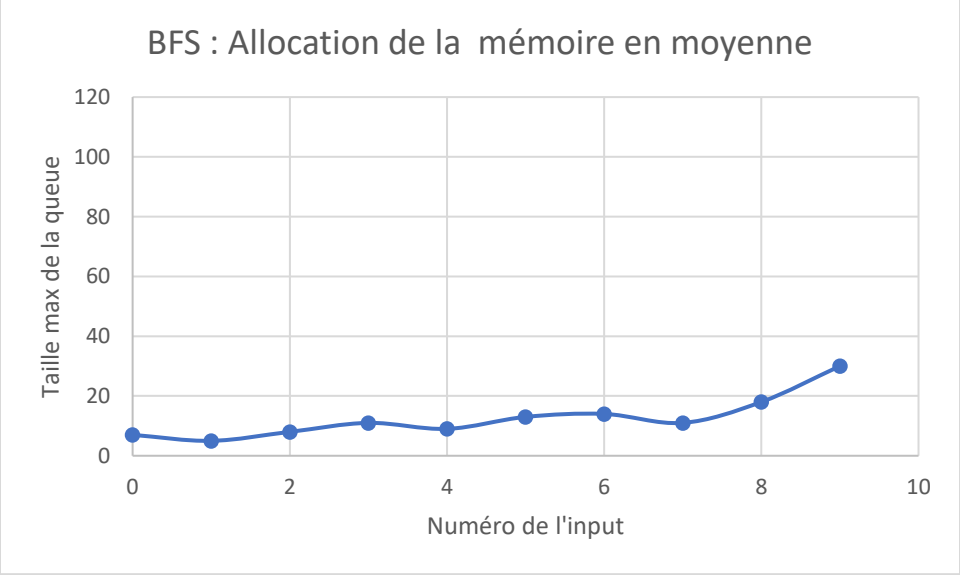
Partie 2.3

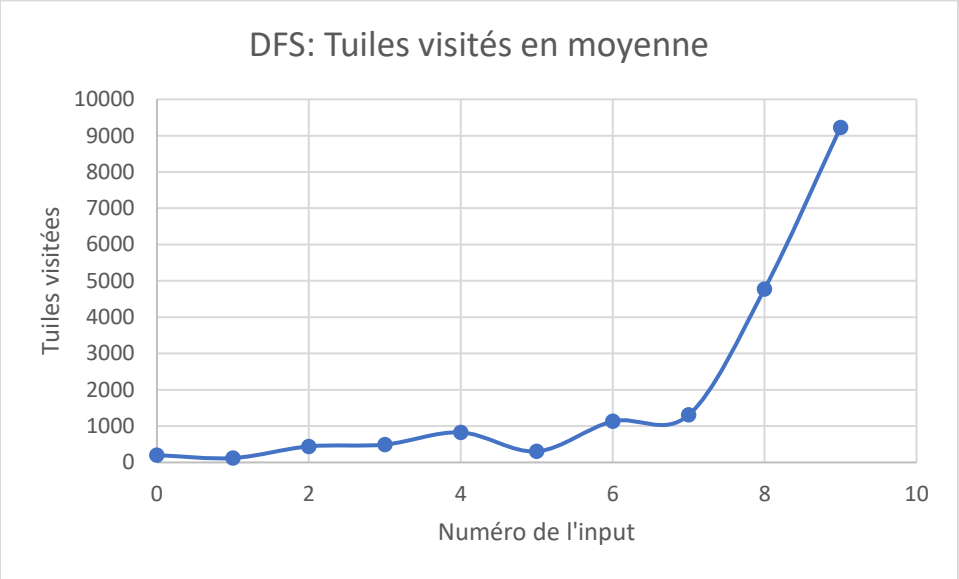
Observations

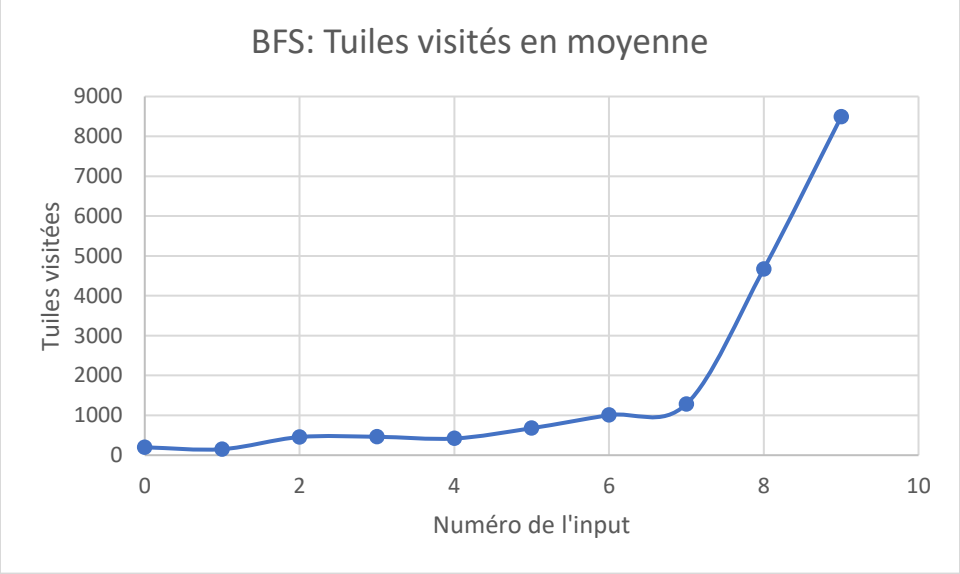
Pour commencer, l'allocation dynamique en moyenne pour les deux algorithmes est assez similaire en fonction des labyrinthes qui ont été fournis. En regardant la taille maximale de la stack pour l'algorithme DFS et la taille maximale de la queue pour l'algorithme BFS, on remarque que certains inputs du labyrinthe donne une taille maximale de la stack/input plus grande que d'autres. L'algorithme DFS donne des tailles maximales de la stack plus grande que l'algorithme BFS avec une taille maximale de 103 versus une taille maximale de 30 pour l'algorithme BFS comme le montre les tableaux suivants.





Par la suite, le nombre de tuiles visités par les algorithmes est quasi similaire. On observe que le nombre de tuiles visitées lors de la recherche pour la sortie des algorithmes est légèrement plus grande pour l'algorithme DFS en comparaison avec l'algorithme BFS tel que montrés dans les tableaux suivants :





Conclusions

En mettant en parallèle les résultats obtenus pour le nombre de tuiles visités en moyenne est l'allocation de la mémoire moyenne, on observe une relation de proportionnalité entre les deux caractéristiques. Ainsi, plus le nombre de tuiles visitées augmente plus la taille maximale de la stack/queue augmente aussi. Cette corrélation peut être expliquée par le fait que l'algorithme DFS a une plus grande complexité spatiale lorsque la hauteur de l'arbre est plus grande et qu'un nombre de tuiles visité plus grand peut signifier un labyrinthe plus grand. De façon analogue, l'allocation de la mémoire augmente avec la largeur de l'arbre et donc en fonction du nombre de tuiles visitées pour l'algorithme BFS.

En conclusion, puisque le nombre de tuiles visitées lors de la recherche de la sortie est similaire pour les deux algorithmes, on peut conclure que les labyrinthes utilisés sont construits davantage en largeur qu'en hauteur, l'algorithme BFS ayant une plus petite allocation de la mémoire que l'algorithme DFS.