Random2

Ajoute N valeur générées aléatoirement entre 1 et maxVal au début d’un vector<int>.

Pour la complexité, on considère les opération insert().

//@param N nombre de donnée à générer

//@param maxVal valeur maximale des données

//@return vector rempli de N valeurs aléatoires.

vector<int> random2(size\_t N, int maxVal) {

vector<int> v;

for (size\_t i = 0; i < N; ++i) {

v.insert(v.begin(), 1 + rand() % maxVal);

}

return v;

}

L’algorithme ci-dessus va en premier lieux créer un vector d’entier puis ajouter une valeur aléatoire à au début de ce dernier grâce à la fonction insert(). La valeur sera créée aléatoirement entre 1 et valMax, paramètre de cette fonction. Le tout sera répété N fois afin de créer un tableau de taille N.

Selon cplusplus.com, la complexité de insert() est constante Le temps d’exécution peut par contre varier lors d’une réallocation de la mémoire. Il faut par contre différencier le temps d’exécution et le nombre d’opération insert(). En effet l’appel à insert est constant. Mais comme nous pouvons le constater sur le graphique ci-dessous, le temps est multiplié par N2. En effet lorsque nous ajoutons une valeur en début de tableau, il faut décaler toutes les autres valeurs vers la droite et donc faire « N » décalage. Ce qui prendra un temps de l’ordre de n2 .

le temps et donc du type n2 .

Pour ce qui est de l’estimation théorique du temps, nous avons effectué le même processus que pour la fonction random. Nous avons cependant pris en compte le « facteur carré ». De plus le tableau des réultats est également en annexe du au grand nombre de mesure que nous avons réalisé .

Finalement les piques dans la graphiques sont expliquées par la même cause que pour la fonction random.