ASD2 Laboratoire 4

## Partie 1

Pour cette partie, nous avons récolté les données qui sont affichés par le programme et nous les avons mis sous forme de graphe. Voici les 2 graphes représentant respectivement le pourcentage de collisions et de buckets vides ainsi que les performances du temps d’insertion et de recherche associé pour chaque algorithme. Voici les correspondances des numéros d’algorithmes :

|  |  |
| --- | --- |
| Numéros d’algorithme | Algorithme |
| 1 | Int |
| 2 | Long |
| 3 | PolILj2EE |
| 4 | PolILj31EE |
| 5 | PolILj37EE |
| 6 | Stl |
| 7 | Sha256 |
| 8 | City |
| 9 | WithoutAVS |

On voit ici que le nombre de collisions et de buckets est approximativement semblable partout sauf pour le 9ième algorithme qui est clairement derrière en terme de collisions (Un facteur d’environ 3 fois plus de collisions que les autres algorithmes). Cela s’explique par le fait que cet algorithme est un algorithme très simple (Uniquement le hash du nom) ce qui peut provoquer beaucoup de collision due au fait que le nom est similaire dans plusieurs entrées (Une famille aura tous le même nom de famille). En réalité, le nom n’est pas une clé primaire à un individu, autrement dit cela ne caractérise pas obligatoirement un individu en particulier.

La raison pour laquelle le nombre de buckets vides croit avec le nombre de collisions est que si on ajoute un élément qui existe déjà, cela créés une collision mais compte également comme une nouvelle paire clé-valeur alors l’algorithme le compte dans le nombre d’entrée alors qu’en réalité cela n’a pas augmenté le nombre de buckets utilisé.

Ici, on voit clairement que l’algorithme 7 est le moins performant