TOPOLOGICAL PERSISTENCE

TD5 - INF556

Octobre 2018

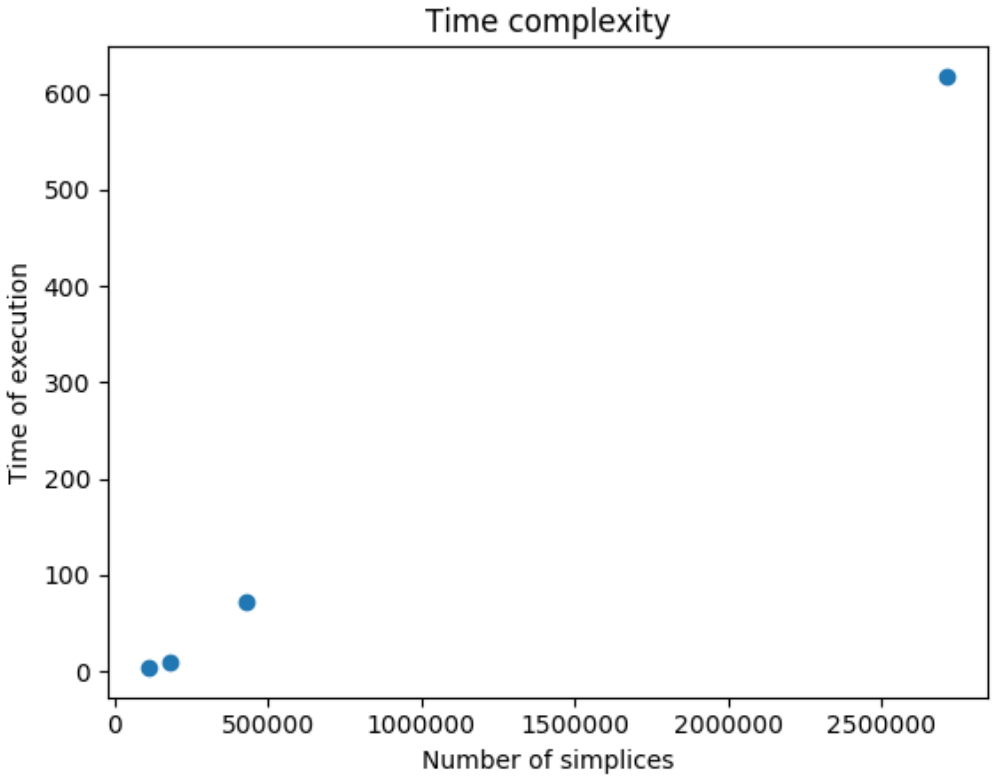
Antoine GROSNIT et Yassin Hamaoui

1
STRUCTURE DU CODE

— La méthode $buildMatrix$ permet d'obtenir une représentation matricielle de la fonction $boundary$ appliquée à chaque simplex de la filtration. Cette matrice est représentée par une $HashMap < Integer, Set < Integer >>$, la clé correspondant à l'identifiant de chaque simplex de la filtration, les valeurs étant le set des indices i pour lesquels l'élément $B_{ij} \neq 0$ où j est la valeur de la clé. B est ainsi une sparse matrix dans la mesure où l'on ne stocke que les éléments non nuls de la matrice.
La table d'association simplex-identifiant, $simplToInd$, est stockée dans une $HashMap < Set < Integer >, Integer >$ où chaque clé correspond au set des entiers correspondant aux vertex du simplex auquel il associe une valeur entière.
La construction de la matrice fait appel à la méthode $getBoundaries$ de la classe Simplex, qui prend en paramètre $simplToInd$ et qui renvoie la liste des identifiants des simplex obtenus en appliquant la fonction boundary.
Complexité de $buildMatrix$: on considère chaque simplex de la filtration et on calcule pour chacun sa frontière, or on a l'inégalité $d \leq log(n)$ où d est la dimension du simplex et n est le nombre d'éléments de la filtration. On a donc une complexité en $O(nlog(n))$

— La méthode $reduceMatrix$ implémente l'algorithme vu en cours, avec une sparse matrix. On a donc, pour chaque colonne, une recherche du pivot potentiel (en $O(n)$), et on simplifie la colonne si une colonne précédemment considérée contenait le même pivot (on opère cette simplification en $O(n)$). On réitère cette opération (au plus n fois) jusqu'à ce que l'on obtienne un nouveau pivot ou que la colonne soit nulle.
La complexité de $reduceMatrix$ est donc bien en $O(n^3)$.

— La méthode $buildBarcode$ itère sur chacune des colonnes : si la colonne j contient un pivot (ligne i), on ajoute la barre $[i,j)$, sinon, on regarde si la ligne j contient un pivot et on ajoute la barre $[i,\text{inf})$.
La complexité de $buildBarcode$ est donc de $O(n)$.


Time complexity
Time of execution
Number of simplices
0
100
200
300
400
500
600
0
500000
1000000
1500000
2000000
2500000