# Descripció de les estructures de dades i algorismes

Per a realitzar la pràctica ens hem basat en l'algorisme descrit a l'article adjunt amb l'anunciat. Aquest algorisme ens ha servit per a poder calcular la relavància d'una entitat amb les altres entitats. D'aquesta manera hem pogut muntar el perfil d'una entitat mostrant la seva informació rellavant, utlitzan la mesura anomenada Hetesim.

L'algorisme s'ha d'implementat usant matrius, així que per poder implementar l'esmentat algorisme hem hagut d'usar els algorismes bàsics per les matrius, com són la multiplicació de matrius, normalitzar per files, transposar una matriu i calcular la norma d'una matriu per files i per columnes.

En assignatures anteriors hem après a representar les matrius com a un vector de dos dimension, però en aquest cas utilitzar aquesta estructura de dades no era viable, a causa de la seva gran dimensió i a la quantiat de zeros que contindria la matriu (informació que no és necessaria guardar).
Per aquest motiu per a representar les matrius hem usat un HashMap, on la clau representava les files i el valor les seves columnes.
La clau era el id de l'entitat en qüestio i el valor un altre hashmap amb clau id de l'altra entitat (representant la columna) i el seu valor un 1 si hi havia relació (abans de normalitzar).

Per a poder realitzar el càlcul ens era necessari tenir guardades les 6 matrius associades aquesta pràctica.Per identificar quines matrius eren necessaries per a realitzar el càlcul de l'Hetesim fem servir un camí (ex. APC) que es descomposa amb una part esquerra i una dreta (AP-CP). Tenint en compte això, hem decidit guardar cada una de les 6 matrius en un HashMap, fent servir com a clau el nom de la matriu (ex. AP) i com a valor la matriu associada.

En el nostre cas un cop realitzat el càlcul de l'Hetesim ens interessa tenir per ordre de més rellevancia a menys els id associats aquella rellàvancia. Per guardar aquesta informació hem usat un TreeMap amb clau el valor del càlcul de l'Hetesim i amb valor una llista dels id associats. La desició d'usar un TreeMap és deguda al fet que automàticament ordena els resultats, però els ordena de forma creixent. Per aquest motiu hem usat un NavigableMap amb clau i valor igual que els descrits anteriorment. Hem usat aquesta estructura de dades

perquè donat un TreeMap i fent servir la funció descendingMap() aquest retorna un NavigableMap ordenat de forma decreixent. Tal com ens interessava.

Finalment, per a trobar tota la informació rellevant d'una entitat necesitavem calcular l'Hetesim amb cada una de la informació que voliem saber, quatre en aquest cas. Per aquest motiu, l'hetesim retorna un ArrayList dels esmentats NavigableMap, on cada posició correspon a un tipus d'entitat diferent.

Una de les funcions avançades que haviem proposat era suggerir el nom d'una entitat quan, en una cerca, no es troba cap entitat amb el nom introduit per l'usuari. Per suggerir aquest nom calculem la distància del nom introduit per l'usuari amb els noms que tenim a la base da dades, el nom que tingui una distància més petita, serà el nom que suggerim. Aquest nom com a mínim ha de tenir una distància igual o inferior a 3. L'algorisme que hem utilitzat per calcular la distància és la distància de levenshtein.