Breu explicació del algoritme

Introducció

En aquesta entrega, per tal de resoldre el problema de assignació quadràtica hem utilitzat dos algoritmes, el Branch&Bound i el Greedy, que tots dos estan a la classe BranchBound.java.

Les classes que utilitza el nostre algoritme son:

* Clase node.java
* Clase nodeComparador.java
* Clase BranchBound.java

La clase Node, es una clase auxiliar de la clase BranchBound, on es guardaran las possibles soluciones del nostre problema. Cada Node conté 3 atributs: un vector de les assignacions pendents, un vector de les assignacions ja realitzades i el cost de aquesta solució.

La clase nodeComparador, es una clase on hi ha implementat un metode que ordena els nodes segons els seus costos, de tal forma que els que tenen menys cost estiguin al principi

Y la clase BranchBound, es la clase principal de l’algoritme, els seus atributs són:

* Una cua de prioritat de nodes amb ordre ascendent
* La mida de la solució
* La matriu de estadístiques (o de flux)
* La matriu de distancies
* Un node amb la millor solució
* El millor Cost

Funcionament del algoritme

Al inicialitzar un BranchBound, se li passen les dues matrius, la de estadístiques i la de distancia, amb aquestes matrius s’inicialitzarà node que tindrà un vector buit de assignacions realitzades amb la mida de les matrius, un vector amb totes les assignacions pendents i un cost = 0 i aquest node s’afegirà a la cua de prioritats. A continuació, s’executarà l’algoritme greedy.

Greedy

L’algoritme durant tres iteracions assignarà de forma aleatòria les assignacions i desprès calcularà el seu cost, i si el cost de la solució aleatòria és millor que el millor cost, el millor cost serà igual al cost de la solució aleatòria. Al sortir del greedy tindrem una bona cota per tal de poder “podar” les rames que ja siguin un cost pitjor que el del greedy.

Cos principal del algoritme ( mètode solve () )

Aquest es el mètode que realitza la principal tasca de l’algoritme i donarà la solució del problema.

Aquest mètode s’executarà mentre quedi algun node en la cua de prioritat, per cada iteració del bucle es traurà un node de la cua, aquest node serà el de menor cost de la cua, i es comprovarà si es una possible solució ( el cost del node < el millor cost ).

Si és una solució parcial ( encara te assignacions pendents), es creant tants fills com assignacions pendents tingui i per cada un dels fills se li assignarà una assignació diferent, és calcularà el nou cost i si el fill té un cost superior al millor cost no s’afegirà a la cua.

En canvi, si la solució és total (no li queden assignacions pendents) i el seu cost es millor que el millor cost fins al moment, la millor solució serà per el moment serà aquesta.