DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 2

Modelando problemas problemas con grafos

21 de octubre de 2018 — Algoritmos y Estructura de Datos III

Integrante	LU	Correo electrónico
Buceta, Diego	001/17	diegobuceta35@gmail.com
Springhart, Gonzalo	318/17	glspringhart@gmail.com

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

Facultad de Ciencias Exactas y Naturales

Universidad de Buenos Aires

Ciudad Universitaria - (Pabellón I/Planta Baja)

Intendente Güiraldes 2610 - C1428EGA

Ciudad Autónoma de Buenos Aires - Rep. Argentina

Tel/Fax: (++54 +11) 4576-3300

http://www.exactas.uba.ar

1. Introducción al problema

2. Justificación teórica

3. Algoritmos presentados

3.1. Kruskal

KruskalSinPathComp(*ListaIncidencia* : *grafoCompleto*, *cantNodos* : *entero*)

```
1  padre ← vector de enteros de tamaño de cantNodos y cargado con el valor de su posición en
2  AGM ← lista de incidencia vacia, de tamaño cantNodos-1
3  OrdenarPorPeso(grafoCompleto)
4  for e:Arista ∈ grafoCompleto
5  if getPadre(indice(e.primerNodo), padre)=getPadre(indice(e.segundoNodo), padre)
6          then agregar(e,agm)
7
8
9  Devolver AGM
```

getPadre(*entero* : *indice*, *padre* : *vectordeenteros*)

```
1  if padre[indice] == indice
2          then Devolver indice
3      else
4          getPadre(indice(padre[indice]),padre)
5
```

getPadreConPathComp(*entero* : *indice*, *padre* : *vectordeenteros*, *altura* : *vectordeenteros*, *ni*
entero

```
1  if padre[indice] == indice
2          then altura[indice] = nivelesSubidos
3          Devolver indice
4      else
5          padre[indice] = getPadreConPathComp(indice(padre[indice]),padre,altura,nivelesSub
6          Devolver padre[indice]
7
```

unirPadres(*indiceNodo*1 : *entero*, *indiceNodo*2 : *entero*, *padre* : *vectordeenteros*)

```
1  padreNodo1 ← getPadre(indiceNodo1, padre)
2  padre[indiceNodo2]=padreNodo1
```

unirPadresConPathComp(*indiceNodo*1 : *entero*, *indiceNodo*2 : *entero*, *padre* :
vectordeenteros, *altura* : *vectordeenteros*, *nivelesSubidos* : *entero*)

```
1  padreNodo1 ← getPadreConPathComp(indiceNodo1, padre,altura,0)
2  padreNodo2 ← getPadreConPathComp(indiceNodo2, padre,altura,0)
3  padreMenosAltura ← min(altura[padreNodo1],altura[padreNodo2])
4  padreMasAltura ← max(altura[padreNodo1],altura[padreNodo2])
5  padre[padreMenosAltura]=padreMasAltura
```

armarGrafoCompleto($nodos : vectordeNodos$)

```
1  listaAristas ← inicializar lista de incidencia
2  matrizAristas ← inicializar matriz de adyacencia
3  for i ← 0 to tam(nodos)
4  for j ← i + 1 to tam(nodos)
5  armar arista con datos de v[i] y v[j]
6  agregar arista a listaAristas
7  armar matriz de adyacencia con la lista de incidencia
8  Devolver Matriz de adyacencia y Lista de incidencia
```

retirarEjesInconsistentes($listaAristas : listaincidencia, \sigma_T, profVecindario, f_T, forma, ca$
$vectordeenteros$)

```
1  for e : listaAristas
2  calcular media y desviacion respecto del vecindario de profVecindario de profundidad de cada
3  si es inconsistente
4  sacar e de las listas
5  recorrer en la lista de ady todos los nodos alcanzables de uno de los extremos y modificar su
6  aumentar en 1 el valor de cantidadDeClusters
7
```

3.2. Complejidad

4. Experimentación

4.1. Variaciones

Tomaremos los casos de test brindados en http://cs.joensuu.fi/sipu/datasets/ y correremos nuestro algoritmo para poder obtener las clusterizaciones.

Los experimentos estarán centrados en analizar las diferentes tipos clusterizaciones que pueden realizarse variando las definiciones de eje inconsistente. Intentaremos analizar las configuraciones necesarias para que se pueda alcanzar una clusterización lo más cercana a la de la percepción humana y los resultados interesantes al que pueden llegarse.

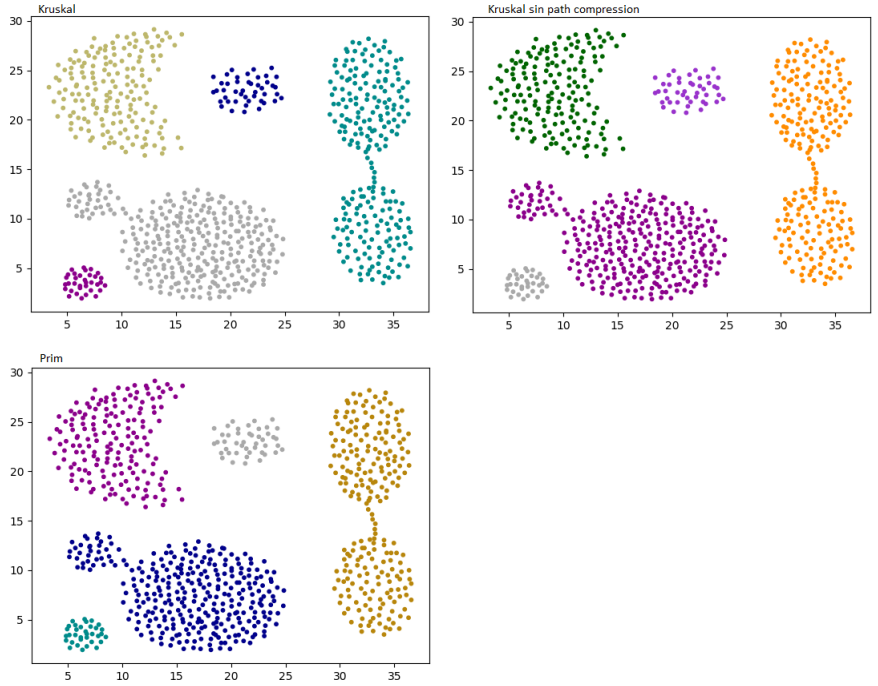
Dados los siguientes: f_T multiplicador del promedio, σ_T multiplicador de la desviación, y la profundidad del vecindario de los extremos del eje candidato, W(XY) el peso del eje candidato, y sea X e Y sus nodos extremos, definiremos un eje inconsistente:

- Forma 1: $\frac{W(XY)}{Promedio(Vecindario(X))} > f_T$ y $\frac{W(XY)}{Promedio(Vecindario(Y))} > f_T$, Es decir, la proporción entre el peso del eje candidato y el promedio de peso del vecindario de sus extremos es mayor al coeficiente dado.
- Forma 2: $W(XY) > Promedio(Vecindario(X)) + \sigma_T * desviacion(Vecindario(X))$ y $W(XY) > Promedio(Vecindario(Y)) + \sigma_T * desviacion(Vecindario(Y))$,

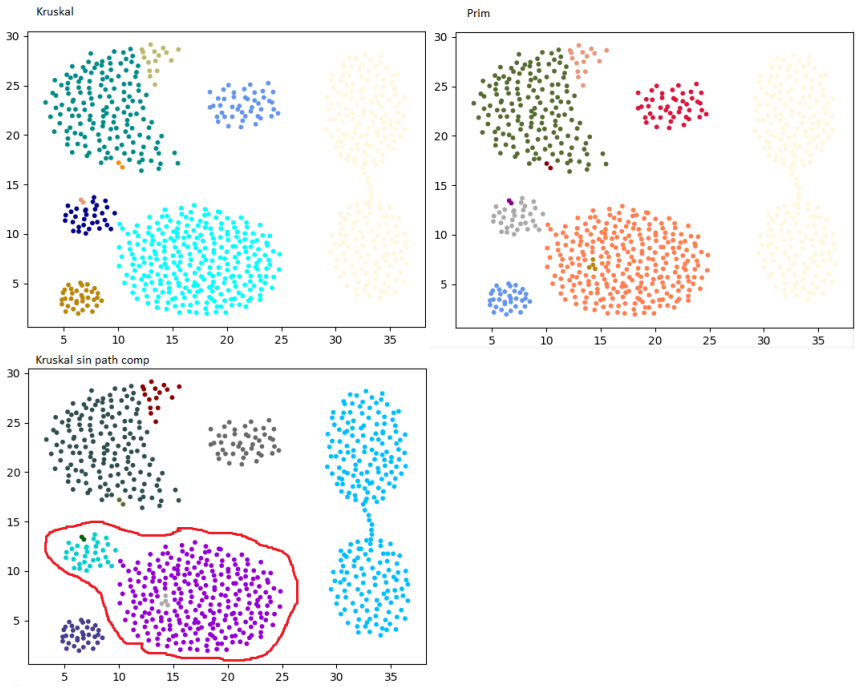
Es decir, que el peso del eje candidato supere al promedio del vecindario de sus extremos por al menos σ_T unidades de la desviación del vecindario del extremo.

- Forma 3: Que se cumpla ambas

Comencemos señalizando que la clusterización no es un problema matemáticamente definido y por ello podemos definirlo de forma tal que se obtenga diferentes resultados para mismas situaciones. La clusterización es la organización y detección de patrones en nuestros datos en grupos que tengan algún significado. Por lo general, a partir de un lote de datos, se busca caracterizarlos en base a la información que se pueda obtener de ellos y no por un patrón o clase de datos conocida dada. Es decir, no se separan los datos en función de su característica en propiedades o atributos previamente fijados, sino que se evalúa la correlación de cada uno de ellos entre todos los datos y en función de esto se agrupan. Estos analisis suelen dividirse como analisis supervisado y no supervisados (o clusteing), donde en los supervisados se cuenta inicialmente con las caracterizaciones que se esperan obtener y se analizan los datos para el reconocimiento de los patrones que permitan situarlo en el modelo correcto. Algunos ejemplos son redes neuronales, árboles de decisión, etc. Discutir el significado de correlación y significancia depende en gran medida de la naturaleza de los datos que se organicen. En nuestro caso, tenemos datos números correspondientes a puntos en el plano y en donde vamos a tomar como relación primaria entre ellos como su distancia euclídea. Experimentaremos con casos ya conocidos de clusterización y evualuaremos fuertemente la habilidad del modelo y sus parámetros para acercarse a los resultados de nuestra percepción, aunque siempre teniendo en mente la literatura del problema de clusterización y particularmente los analisis de los principios de agrupación propuestos por la filosofía de Gestalt previamente analizados.



Analizando el siguiente caso podemos ver que generando el AGM con cualquiera de los métodos elegidos obtenemos la misma clusterización. Elegimos como parámetros que un eje inconsistente debe superar en 2.5 al promedio de los vecinos de 2 pasos de sus extremos. La clasificación resulta en 5 clusters. Como primera observación, la percepción humana podría arrojar que el cluster de la esquina derecha en realidad podrían ser dos clusters diferentes, al igual que el cluster de la esquina izquierda. Si intentamos conseguir algo más cercano a esto, podemos intentar clasificar ahora también teniendo en cuenta la desviación. Como todos los puntos tienen una densidad de vecinos parecida, no tiene mucho sentido modificar la distancia del vecindario de los extremos (actualmente dos). Si reducimos el coeficiente del promedio de 2.5 hasta 1.4 y el coeficiente de la desviación lo fijamos en 2. Después de varios intentemos, encontramos que en ese punto obtenemos lo siguiente:



Por un lado, conseguimos aumentar los clusters. Sin embargo, las modificaciones que hagamos para dividir clusters anteriores resultan en divisiones no deseadas en otros que ya estaban 'bien' según nuestra percepción. Ahora pudimos encontrar una forma de clusterizar más parecida a nuestra percepción el cluster señalado. Sin embargo, esto produce ciertas inconsistencias. Por ejemplo, poder identificar más de un grupo nos cuesta encontrar ahora cuatro. Excepto en kruskal con path compression, en los otros dos encontramos un mini-cluster dentro del cluster grande marcado con rojo, a su ve también el nuevo cluster ahora en turquesa dentro de la marcación tiene también un mini-cluster. En los demás fuera de la marca también encontramos este tipo de cosas. Podríamos pensar entonces que en este tipo de situaciones en donde hay grandes grupos de cluster bien definidos y las densidades de puntos similares en cada uno, podemos fácilmente clasificarlos de forma similar a nuestra percepción. Sin embargo, cuando queremos perfeccionarlo, nos encontramos con dificultades para poder seguir manteniendo la consistencia en los demás clusters.
Una posible solución a esto podría ser hacer un 'zoom' a la zona de clusters que nos interesa. De esa forma conseguiríamos abstraernos de la clasificación de los demás y centrarnos sólo en los que nos interesa.
Entonces una de las cosas que vamos a ver es si esto es realmente factible y entonces podemos utilizar esta técnica para poder ver a grandes rasgos los clusters y en función de nuestro interes enfocarnos en una zona en particular. Esto nos facilita porque al momento de volver a clasificar esa zona, lo haríamos restringiendo nuestro domino de patrones (anteriormente de todo el conjunto de puntos) hacia uno más pequeño.