.UBA fiuba
FACULTAD DE INGENIERÍA

Generar y probar: Particiones de conjunto y clustering

Teoría de Algoritmos I (75.29 / 95.06)

Ing. Víctor Daniel Podberezski
vpodberezski@fi.uba.ar

Generar y probar: Particiones de conjunto

- **Espacio de soluciones**
  - Trabajaremos sobre la separación de n elementos en diferentes subconjuntos.
  - Los elementos son únicos, indivisibles y deben pertenecer a un conjunto.
  - No interesa ni el orden de los elementos dentro de cada subconjunto, ni el orden de los subconjuntos.

- **Función generativa**
  - Generación de las partición es de los elementos
  - Corresponderá a las restricciones explicitas del problema

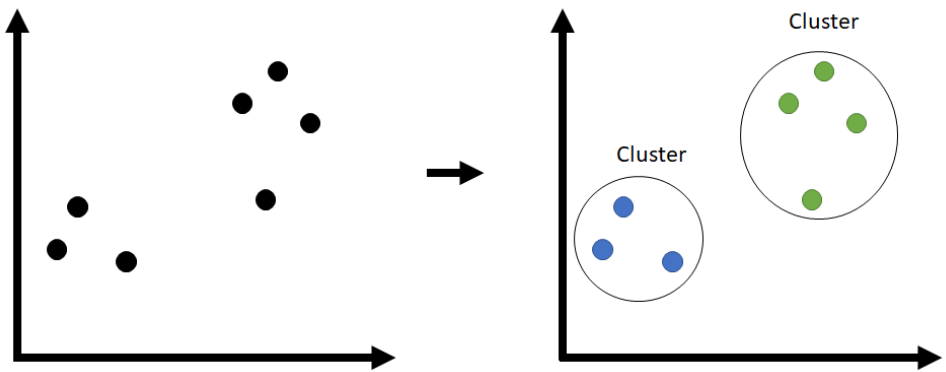
Ejemplo práctico: Clustering

Dados “n” elementos

con ciertas características mensurables y comparables

Queremos agruparlos en clusters

de forma de lograr un balance que minimice la distancia entre los puntos dentro de un mismo cluster
y maximizar la distancia entre los puntos entre clusters diferentes.



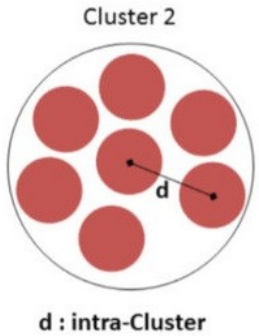
Ejemplo práctico: Clustering

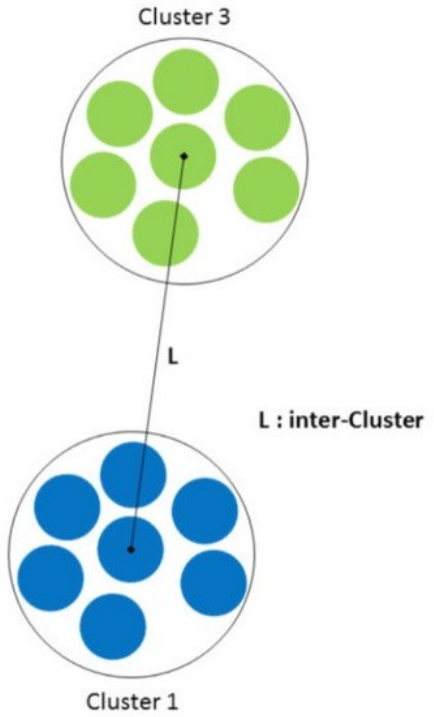
No se establece una cantidad requerida de clusters

- Se deben evaluar todos los clusters posibles para buscar el óptimo
- Todos los elementos deben estar en un cluster

Para evaluar una posible solución

- se establece una medida de similitud entre elementos
- se define una medida inter cluster

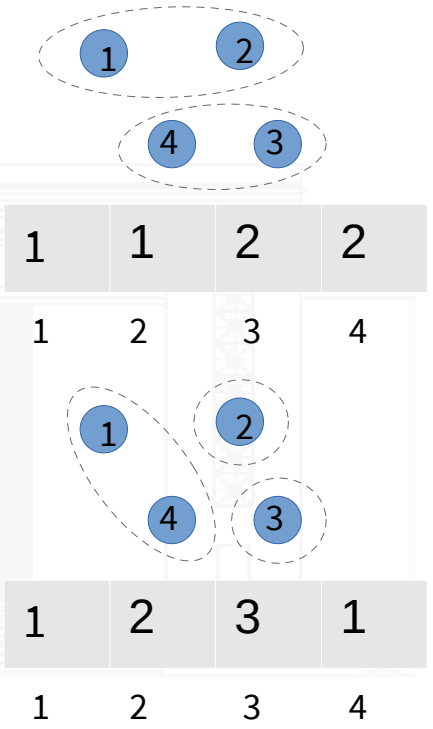




Ejemplo práctico: Clustering

Podemos expresar una posible solución con un vector P de “n” posiciones.

- La posición i corresponde al elemento i
- Su valor numérico corresponde al cluster (partición) donde ese asigna.
- Al primer elemento siempre se asigna a la partición 1.
- El valor en la posición j es menor o igual al máximo valor de los elementos anteriores más uno (cadena de crecimiento restringido o “restricted growth string”)



La cantidad de posibles particiones de “n” elementos

Se puede acotar a $\Theta\left(\frac{n}{\ln n}\right)^n$

Generación de las particiones

- **Se generarán en orden lexicográfico.**
  - Se comenzará con todos los elementos en un mismo conjunto (el 1)
- **Utilizará un vector auxiliar.**
  - En la posicion i tendra el valor máximo que aparece en alguna posición anterior
- **Generar la siguiente partición**
  - Buscará el primer elemento incrementable comenzando por la derecha
  - Lo llevará al siguiente conjunto y pondrá los recorridos en el conjunto 1
- **Finaliza el proceso**
  - Cuando cada elemento se encuentra en un conjunto diferente

1	1	1	1
1	1	1	2
1	1	2	1
1	1	2	3
1	2	1	1
1	2	1	2
1	2	1	3
...			
1	2	3	4
1	2	3	4

Generación de las particiones: Inicialización

- **Para inicializar:**
  - Comenzar con el vector P y Auxiliar A con todos sus valores en 1
  - Corresponde al tener todos los elementos en el mismo conjunto

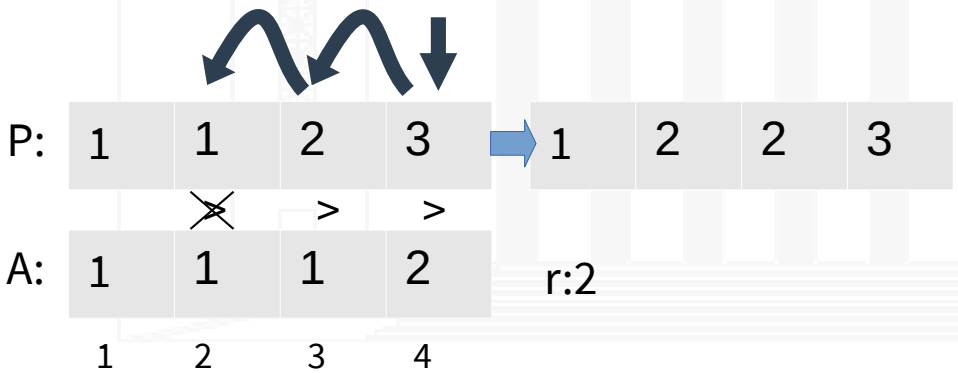
P:	1	1	1	1
A:	1	1	1	1
	1	2	3	4

```
Sea P un vector con la partición
Sea A un vector auxiliar

Inicializar P:
     Desde i=1 a n
          Establecer P[i]=1
          Establecer A[i]=1
```

Generación de las particiones: Obtener siguiente – Paso 1

- **Buscar el primer elemento incrementable**
  - Corresponde al primer elemento tal que no existe un anterior en un conjunto con identificador mayor o igual



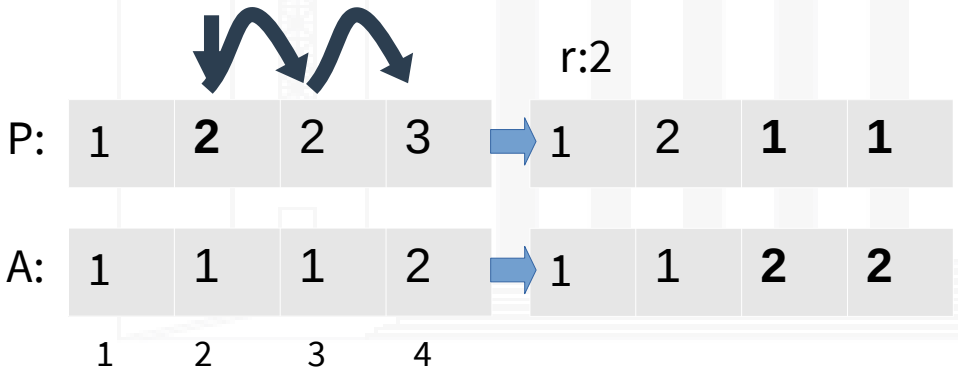

```
Incrementar P:
     Sea j=n
     Mientras P[j]>A[j]
          j-=1
     Si j==1
          retornar 'fin'

     P[j]+=1
     r = max(P[j],A[j])

     ...
```

Generación de las particiones: Obtener siguiente – Paso 2

- **Pasar los elementos posteriores al incrementado a 1**
  - Actualizar el vector con el mayor entre el valor del incrementado o el máximo anterior



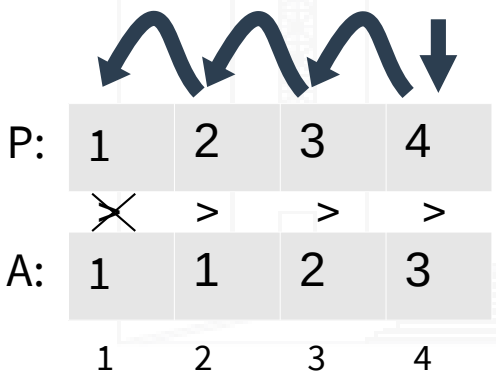

```
...
j += 1

Mientras j<=n
     Establecer P[j] = 1
     Establecer A[j] = r
     j +=1

retornar P
```

Generación de las particiones: Final

- **El procedimiento finaliza cuando no quedan elemenos incrementables**
  - En ese caso se llega al elemento en la posición 1




```
Incrementar P:
     Sea j=n
     Mientras P[j]>A[j]
          j-=1
     Si j==1
          retornar 'fin'

     P[j]+=1
     r = max(P[j],A[j])

     ...
```

Verificación de la solución

Distancia intra cluster y inter cluster

Para cada cluster se debe calcular su distancia intra cluster

Entre cada cluster se debe calcular su distancia intra cluster.

Existen diferentes alternativas para estos indicadores (excede el alcance de nuestra explicación)

Calculo de calidad de la partición

En base a los indicadores de la solución factible se debe calcular un valor de calidad de la misma.

Nuevamente existen diferentes alternativas

Se evaluará su valor y se utilizará para comparar entre particiones

Se seleccionará a aquella partición que maximice la calidad

Generar y probar: Particiones de conjunto

Unificando:

Comenzar con la primera partición posible

Mientras quedan particiones posibles obtener el próximo y evaluar si su calidad es superior al mejor previamente encontrado.

Complejidad temporal: $\Theta\left(\frac{n}{\ln n}\right)^n * \Theta(CalculoCalidad())$

```
Inicializar P

Sea valorSolucion = Evaluar P
Sea mejorSolucion = P

Mientras permutar C <> 'fin'
     Si valorSolucion  > Evaluar P
          minimoCosto = Evaluar P
          mejorSolucion = P

retornar mejorSolucion y valorSolucion
```