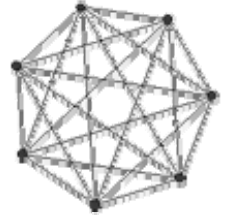




UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA



Análisis y Diseños de Algoritmos

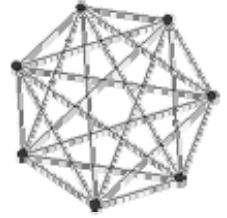
Aproximación con Dificultad NP

Br. Luis Gerardo Peña Camacho



UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA

Conceptos



P es la clase de problemas de decisión que se pueden resolver mediante un algoritmo de tiempo polinómico.

Un algoritmo en **tiempo polinómico** es eficiente si existe un polinomio $p(n)$ tal que el algoritmo puede resolver cualquier caso de tamaño n en un tiempo $O(p(n))$

- “Es Hamiltoniano el grafo G ” (HAMD) \Rightarrow decisión \Rightarrow **P**



Conceptos



Un ciclo hamiltoniano “HAM” en un grafo, no es un problema de decisión:

$$\text{HAM} \Rightarrow \text{NP} \quad \therefore \text{P} \subseteq \text{NP}$$

Si NP-completos se resuelven en $O(p(n))$ si todos los demás problemas NP se resuelven en $O(p(n))$, equivale a decir:

$$\text{P} = \text{NP}$$

Y los NP-completos que no se pueden resolver en tiempo polinómico

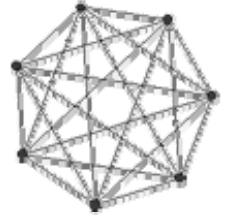
$$\text{P} \neq \text{NP}$$

“Todo problema que se pueda resolver en un tiempo polinómico está automáticamente en NP”



UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA

Conceptos



Si tenemos dos problemas, A y B. A es polinómicamente Turing reducible a B si existe un algoritmo para resolver A en un tiempo que sería polinómico si pudiéramos resolver casos arbitrarios de B con coste unitario $A \leq_T^P B$

Si $A \leq_T^P B$ y $B \leq_T^P A \Rightarrow B \equiv_T^P A$

$$\text{HAM} \equiv_T^P \text{HAMD}$$



Conceptos



Algoritmos heurísticos: Procedimientos que pueden producir una buena solución e incluso la solución óptima.

“Un método heurístico es un procedimiento para resolver un problema de optimización bien definido mediante una aproximación intuitiva, en la que la estructura del problema se utiliza de forma inteligente para obtener una buena solución.”

En contraposición a los *métodos exactos* que proporcionan una solución óptima del problema, los *métodos heurísticos* se limitan a proporcionar una buena solución del problema no necesariamente óptima. Lógicamente, el tiempo invertido por un método exacto para encontrar la solución óptima de un problema difícil, si es que existe tal método, es de un orden de magnitud muy superior al del heurístico (pudiendo llegar a ser tan grande en muchos casos, que sea inaplicable).



Conceptos



Algoritmo Aproximado: es un procedimiento que siempre proporcione algún tipo de solución para el problema, aun cuando quizá no llegue a encontrar una solución óptima, por tanto es posible calcular una cota bien de la diferencia, bien de la razón entre la solución óptima y la producida por el algoritmo aproximado.

Ningún algoritmo aproximado eficiente puede garantizar una cuota superior fija para el valor absoluto de su soluciones a no ser que $P = NP$



UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA

Factibilidad



Sea $opt(X)$ el valor de la solución óptima del caso X
optimización \Rightarrow X Colorado de un Grafo (G)
 $opt(G)$ = Número cromático de G:

Un algoritmo aproximado hallaría algún valor $opt^{\sim}(X)$ que podría ser subóptimo, pero requiere que sea factible.

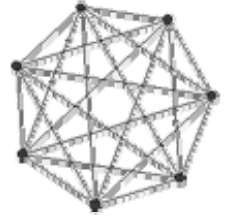
Requisito de factibilidad

$opt^{\sim}(X) \geq opt(X)$ para problemas de minimización

$opt^{\sim}(X) \leq opt(X)$ para problemas de maximización



Problema de Optimización P



A todo problema de optimización le corresponden problemas de aproximación absoluta c y relativa ε .

Con c y ε constantes positivas $\Rightarrow c$ y $\varepsilon > 0$

c_abs_P, es el problema de aproximación absoluta que consiste que para cualquier caso X , una solución factible $opt^{\sim}(X)$ cuyo error absoluto en comparación con la solución óptima $opt(X)$ sea como máximo c , para problemas de minimización y maximización

$$opt(X) \leq opt^{\sim}(X) \leq opt(X) + c$$

$$opt(X) - c \leq opt^{\sim}(X) \leq opt(X)$$



UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA

Problema de Optimización P



ε -rel-P, es el problema de aproximación relativa, que consiste en hallar para cualquier caso X , una solución factible cuyo error relativo, comparado con la solución óptima, sea como máximo ε ,
Para problemas de minimización y maximización respectivamente:

$$opt(X) \leq opt^{\sim}(X) \leq (1 - \varepsilon)opt(X)$$

$$(1 - \varepsilon)opt(X) \leq opt^{\sim}(X) \leq opt(X)$$



UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA

Problema de Optimización P



Ejemplos:

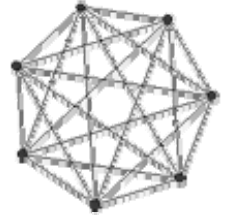
1-abs-P \Rightarrow determinar el número de objetos que se pueden almacenar en dos cajas.

1-rel-P \Rightarrow problema del Viajante Métrico

propiedad desigualdad triangular

$d(i,j) \leq d(i,k) + d(k,j) \Rightarrow$ euclídeos

Aproximación con dificultad absoluta



Sea MTSP el problema del viajante métrico

$$\text{MTSP} \leq^P_T c\text{-abs-MTSP}, c > 0$$

Para esto:

- Suponga que existe un algoritmo para resolver c -abs-MTSP
- Tenemos que buscar encontrar el problema exacto de MTSP, sin tener en cuenta el tiempo invertido en le algoritmo aproximado.
- Por tanto todo algoritmo eficiente aproximado \Rightarrow algoritmo eficiente para el problema exacto MTSP \Rightarrow el problema exacto es difícil puesto que es NP.
- Se concluye que no existe ningún algoritmo eficiente para poder hallar aproximaciones c absolutas del MTSP.



Aproximación con dificultad relativa



Sea TSP, el problema del viajante sin restricciones

$$\mathbf{TSP} \leq \frac{P}{T} \varepsilon \text{-rel-TSP}, \varepsilon > 0$$

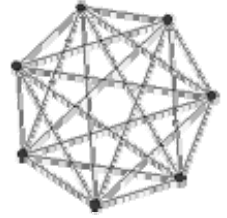
Con ε -rel-TSP, se puede resolver el problema HAMD, con una técnica similar a $\mathbf{HAMD} \leq \frac{P}{T} \mathbf{TSPD}$, por tanto para demostrar que la solución exacta de TSP se puede utilizar para resolver HAMD, por tanto se concluye: $\mathbf{TSP} \leq \frac{P}{T} \varepsilon \text{-rel-TSP}, \varepsilon > 0$

Y además que

$$\mathbf{HAMD} \leq \frac{P}{T} \varepsilon \text{-rel-TSP}, \varepsilon > 0$$



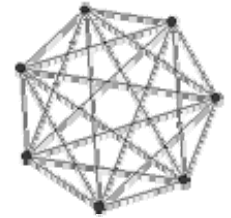
Problema del Racimo Mínimo



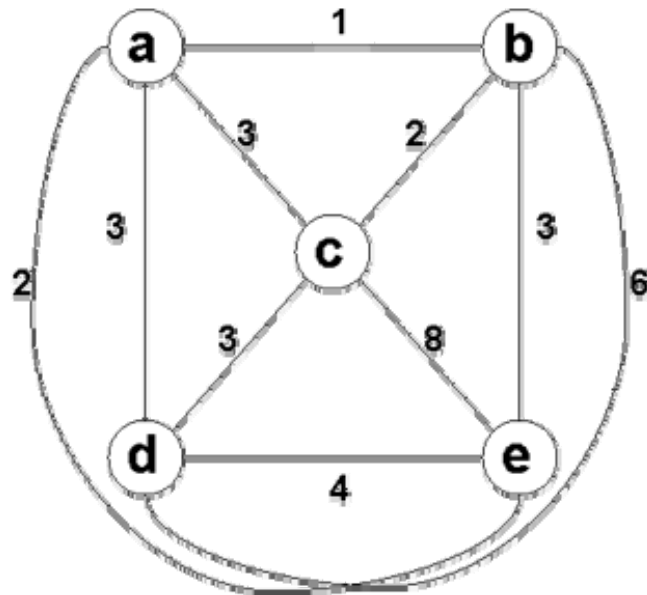
Sea $G = \langle N, A \rangle$, un grafo no dirigido y sea $c: A \rightarrow \mathbb{R}^+$ una función de coste. Considere tres sub conjuntos N_1, N_2, N_3 , que se llaman racimos, de tal manera que cada nodo N_i pertenezca exactamente a uno de los racimos además de ser las **aristas internas** aquellas que enlazan nodos de un mismo racimo y las **aristas cruzadas**, aquellas que enlazan nodos de racimos distintos, el problema esta en seleccionar N_1, N_2, N_3 , de tal manera que el coste de las aristas cruzadas sea máximo o equivalente de la forma que minimice el coste total la aristas internas. Por tanto este problema tiene dificultad NP



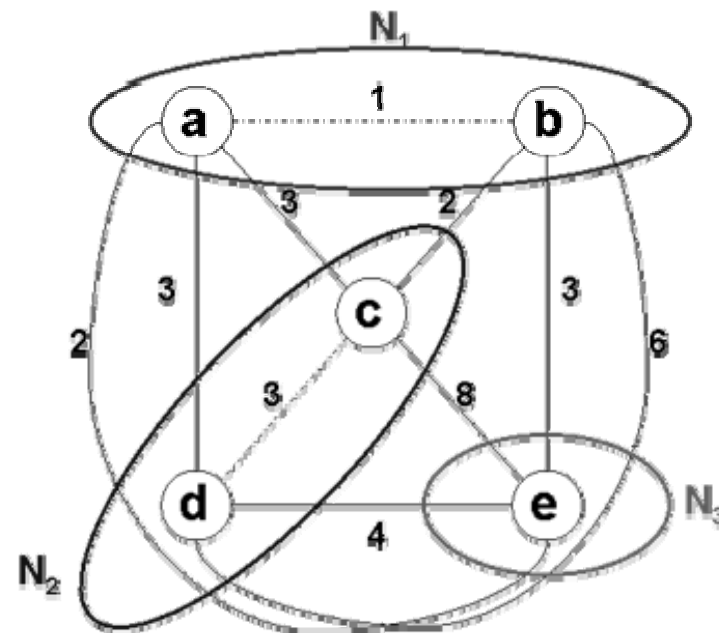
Problema del Racimo Mínimo



Se muestra un ejemplo de un grafo donde se selecciona la solución óptima que consiste en seleccionar $N_1=\{a,b\}$, $N_2=\{c,d\}$ y $N_3=\{e\}$, de tal manera que el coste total de las aristas internas es 4 y de las aristas cruzadas es 31.



Grafo propuesto

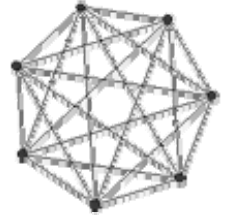


Solución Óptima



UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA

Problema del Racimo Mínimo



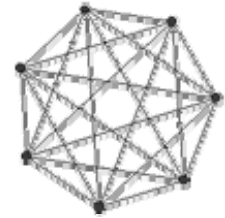
Es equivalente maximizar el coste total de las aristas cruzadas o minimizar el coste total de las aristas internas, por tanto se consideran dos problemas de optimización siguientes:

MAX-CORTAR: maximiza el coste total de las aristas cruzadas, para todas las particiones de N .

MIN-RACIMO: minimiza el coste total de las aristas internas, para todas las particiones de N .



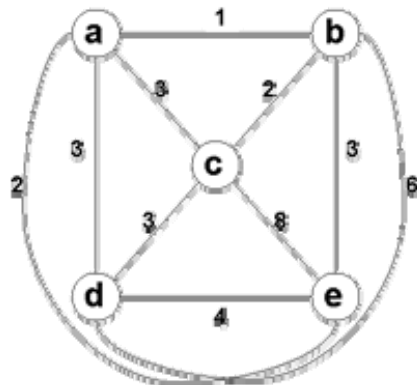
Problema del Racimo Mínimo



MAX-CORTAR_aprox($G=\langle N,A \rangle$, $c: A \rightarrow \mathbb{R}^+$)	
pre($c: A \rightarrow \mathbb{R}^+$)	pos($N_1, N_2, N_3 \neq \emptyset$) \wedge coste aristas cruzadas
<pre> 0 $N_1, N_2, N_3 \leftarrow \emptyset$ racimo, suma $\leftarrow 0$ 1 para todo $u \in N$ hacer costemin $\leftarrow \infty$ 2 para $i \leftarrow 1$ hasta 3 hacer coste $\leftarrow 0$ 3 para todo $v \in N_i$ hacer si $\{u, v\} \in A$ entonces coste \leftarrow coste + $c(\{u, v\})$ fsi frp suma \leftarrow suma + coste si coste < costemin entonces costemin \leftarrow coste k \leftarrow i fsi frp $N_k \leftarrow N_k \cup \{u\}$ racimo \leftarrow racimo + costemin 4 frp devolver suma – racimo </pre>	<p>-N_1, N_2, N_3: conjuntos donde se formaran una parición de N cuando concluya el algoritmo.</p> <p>-suma: acumula el coste total de todas las aristas de G.</p> <p>-racimo: acumula el coste de todas las aristas internas en la solución aproximada que seleccionamos</p>



Problema del Racimo Mınimo



Matriz de adyacencia

	a	b	c	d	e
a	0	1	3	3	2
b	1	0	2	6	3
c	3	2	0	3	8
d	3	6	3	0	4
e	2	3	8	4	0

$N_1 = \{\}, N_2 = \{\}, N_3 = \{\}$

racimo = 0

suma = 0

Para $u = a$

i	1	2	3
coste=0	0	0	0
suma	0	0	0
Costemin= ∞	0	0	0
k	1	1	1

```

N1, N2, N3 ← ∅
racimo, suma ← 0
para todo v ∈ N hacer
  costemin ← ∞
  para i ← 1 hasta 3 hacer
    coste ← 0
    para todo v ∈ Ni hacer
      si {u, v} ∈ A entonces
        coste ← coste + c({u, v})
    fsi
  frp
  suma ← suma + coste
  si coste ≤ costemin entonces costemin ← coste
  k ← i
  fsi
frp
Nk ← Nk ∪ {u}
racimo ← racimo + costemin
frp
devolver suma - racimo

```

Finalmente para $u = a$

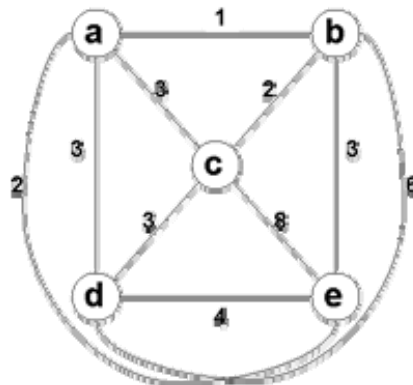
$N_1 = \{a\}, N_2 = \{\}, N_3 = \{\}$

racimo = 0

suma = 0



Problema del Racimo Mínimo



Matriz de adyacencia

	a	b	c	d	e
a	0	1	3	3	2
b	1	0	2	6	3
c	3	2	0	3	8
d	3	6	3	0	4
e	2	3	8	4	0

$N_1 = \{a\}, N_2 = \{\}, N_3 = \{\}$

racimo = 0

suma = 0

Para $u = b$

i	1	2	3
coste=0	1	0	0
suma	1	1	1
Costemin= ∞	1	0	0
k	1	2	2

```

N1, N2, N3 ← ∅
racimo, suma ← 0
para todo u ∈ N hacer
  costemin ← ∞
  para i ← 1 hasta 3 hacer
    coste ← 0
    para todo v ∈ Ni hacer
      si {u, v} ∈ A entonces
        coste ← coste + c({u, v})
    fsi
  frp
  suma ← suma + coste
  si coste ≤ costemin entonces costemin ← coste
  k ← i
  fsi
frp
Nk ← Nk ∪ {u}
racimo ← racimo + costemin
frp
devolver suma - racimo
  
```

Finalmente para $u = b$

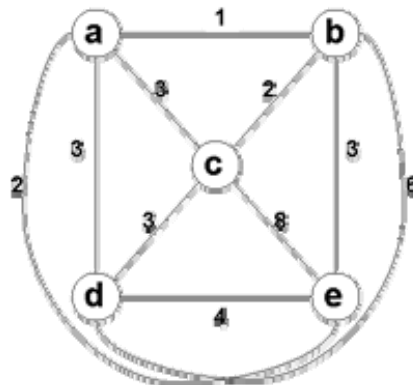
$N_1 = \{a\}, N_2 = \{b\}, N_3 = \{\}$

racimo = 0

suma = 1



Problema del Racimo Mínimo



Matriz de adyacencia

	a	b	c	d	e
a	0	1	3	3	2
b	1	0	2	6	3
c	3	2	0	3	8
d	3	6	3	0	4
e	2	3	8	4	0

$N_1 = \{a\}, N_2 = \{b\}, N_3 = \{\}$

racimo = 0

suma = 1

Para $u = c$

i	1	2	3
coste=0	3	2	0
suma	4	6	6
Costemin= ∞	3	2	0
k	1	2	3

```

N1, N2, N3 ← ∅
racimo, suma ← 0
para todo u ∈ N hacer
  costemin ← ∞
  para i ← 1 hasta 3 hacer
    coste ← 0
    para todo v ∈ Ni hacer
      si {u, v} ∈ A entonces
        coste ← coste + c({u, v})
    fsi
  frp
  suma ← suma + coste
  si coste ≤ costemin entonces costemin ← coste
  k ← i
  fsi
frp
Nk ← Nk ∪ {u}
racimo ← racimo + costemin
frp
devolver suma - racimo
  
```

Finalmente para $u = c$

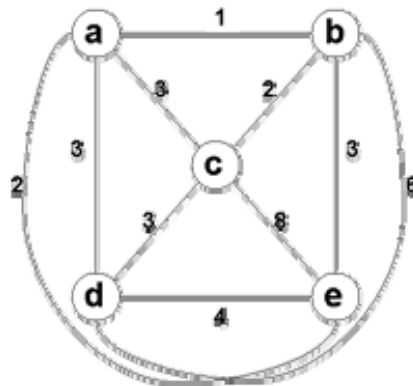
$N_1 = \{a\}, N_2 = \{b\}, N_3 = \{c\}$

racimo = 0

suma = 6



Problema del Racimo Mínimo



Matriz de adyacencia

	a	b	c	d	e
a	0	1	3	3	2
b	1	0	2	6	3
c	3	2	0	3	8
d	3	6	3	0	4
e	2	3	8	4	0

$N_1 = \{a\}, N_2 = \{b\}, N_3 = \{c\}$

racimo = 0

suma = 6

Para $u = d$

i	1	2	3
coste=0	3	6	3
suma	9	15	18
Costemin= ∞	3	3	3
k	1	1	1

```

N1, N2, N3 ← ∅
racimo, suma ← 0
para todo u ∈ N hacer
  costemin ← ∞
  para i ← 1 hasta 3 hacer
    coste ← 0
    para todo v ∈ Ni hacer
      si {u, v} ∈ A entonces
        coste ← coste + c({u, v})
    fsi
  frp
  suma ← suma + coste
  si coste ≤ costemin entonces costemin ← coste
  k ← i
  fsi
frp
Nk ← Nk ∪ {u}
racimo ← racimo + costemin
frp
devolver suma - racimo
  
```

Finalmente para $u = c$

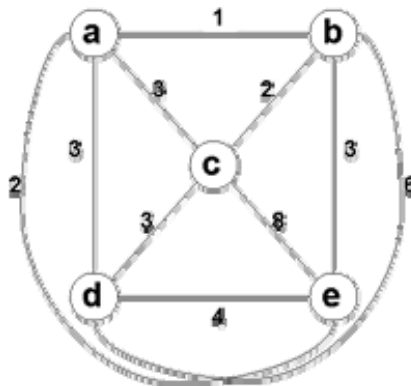
$N_1 = \{a, d\}, N_2 = \{b\}, N_3 = \{c\}$

racimo = 3

suma = 18



Problema del Racimo Mínimo



Matriz de adyacencia

	a	b	c	d	e
a	0	1	3	3	2
b	1	0	2	6	3
c	3	2	0	3	8
d	3	6	3	0	4
e	2	3	8	4	0

$N_1 = \{a, d\}$, $N_2 = \{b\}$, $N_3 = \{c\}$

racimo = 3

suma = 18

Para $u = e$

i	1	2	3
coste=0	2 6	3	8
suma	24	27	35
Costemin= ∞	6	3	3
k	1	2	2

```

N1, N2, N3 ← ∅
racimo, suma ← 0
para todo u ∈ N hacer
  costemin ← ∞
  para i ← 1 hasta 3 hacer
    coste ← 0
    para todo v ∈ Ni hacer
      si {u, v} ∈ A entonces
        coste ← coste + c({u, v})
    fsi
  frp
  suma ← suma + coste
  si coste ≤ costemin entonces costemin ← coste
  k ← i
  fsi
frp
Nk ← Nk ∪ {u}
racimo ← racimo + costemin
frp
devolver suma - racimo
  
```

Finalmente para $u = e$

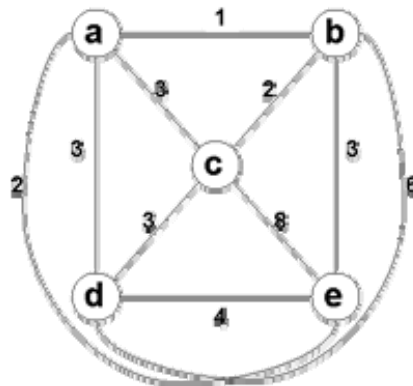
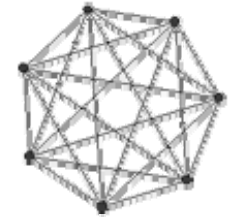
$N_1 = \{a, d\}$, $N_2 = \{b, e\}$, $N_3 = \{c\}$

racimo = 6

suma = 35



Problema del Racimo Mnimo



Matriz de adyacencia

	a	b	c	d	e
a	0	1	3	3	2
b	1	0	2	6	3
c	3	2	0	3	8
d	3	6	3	0	4
e	2	3	8	4	0

Los conjuntos formados
 $N_1 = \{a, d\}$, $N_2 = \{b, e\}$, $N_3 = \{c\}$
 racimo = 6
 suma = 35

Las aristas cruzadas = 29
 Con un 93.54%, de la solucin
 ptima que es 31

Las aristas internas = 6
 Con un 50% por arriba del
 ptimo, por tanto las aristas
 internas dista del 2/3 del
 ptimo que es 4

$$6 \times \frac{2}{3} = 4$$

```

N1, N2, N3 ← ∅
racimo, suma ← 0
para todo u ∈ N hacer
  costemin ← ∞
  para i ← 1 hasta 3 hacer
    coste ← 0
    para todo v ∈ Ni hacer
      si {u, v} ∈ A entonces
        coste ← coste + c({u, v})
    fsi
  frp
  suma ← suma + coste
  si coste ≤ costemin entonces costemin ← coste
  k ← i
  fsi
frp
Nk ← Nk ∪ {u}
racimo ← racimo + costemin
frp
devolver suma - racimo

```



UNIVERSIDAD
DE LOS ANDES
MERIDA VENEZUELA

Problema del Racimo Mínimo



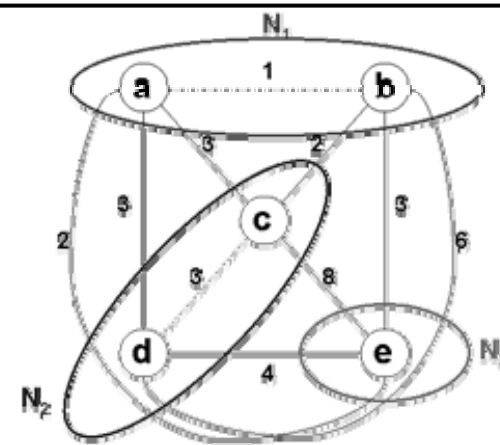
Dado que racimo y suma poseen valor inicial nulo, se tiene $\text{suma} \geq 3 \times \text{racimo}$, $35 \geq 18$, el coste total de las aristas cruzadas dada por el algoritmo es:

$$\text{opt}^{\sim}(G, c) = \text{suma} - \text{racimo} \geq \frac{2}{3} \text{suma}$$

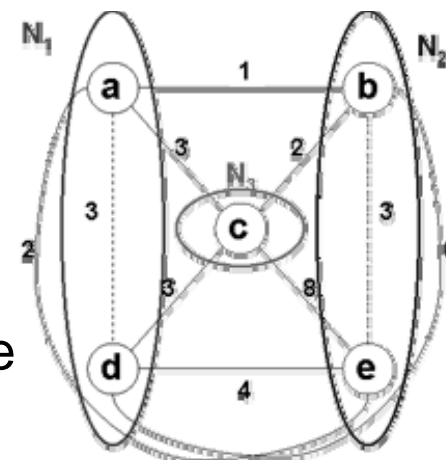
Sabemos que el coste total de las aristas cruzadas del óptimo no puede ser menor que el coste de la solución aproximada del algoritmo y no puede ser mayor que la suma de todas las aristas del grafo, por tanto:

$$(1 - 1/3) \text{opt}'(G, c) \leq 2/3 \text{suma} \leq \text{opt}^{\sim}(G, c) \leq \text{opt}'(G, c)$$

El algoritmo proporciona una aproximación de 1/3-rel-MAX-CORTAR



opt(G)

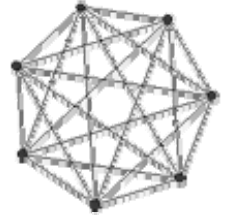


opt~(G)



UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA

Problema del Racimo Mínimo



MIN-RACIMO: minimiza el coste total de las aristas internas, para todas las particiones de N .

$$MIN-RACIMO \leq \frac{P}{T} \varepsilon\text{-rel-MIN-RACIMO}, \varepsilon > 0$$

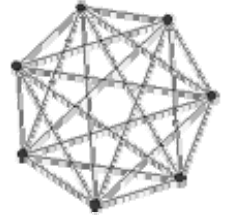
No se puede garantizar que ningún algoritmo eficiente encuentre una buena aproximación $\varepsilon\text{-rel-MIN-RACIMO}$, a no ser que $P=NP$

Demostración: se hace asumiendo el problema de colorear un grafo con tres colores 3COL



UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA

Llenado de cajas - II parte

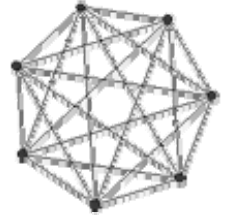


Un algoritmo aproximado voraz eficiente para el problema de llenado de cajas, que garantiza una solución que no utilice mas de $4 + 11/9 \text{ opt}$ cajas para almacenar objetos.

No existe un esquema de aproximación completamente polinómico para este problema de dificultad NP si $P \neq NP$



Problema de la Mochila (6)



- Algoritmo de programación Dinámica

$$V[i, j] = \max(V[i - j, j], V[i - j, j - w_i] + v_i) \rightarrow \Theta(nW)$$

devuelve el valor máximo de los objetos que se transporta en la mochila si el límite de peso es j

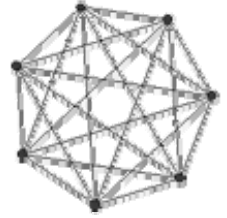
- Se aproxima en una tabla $U[1..n, 0..M]$, donde M es una cuota superior del valor óptimo que se puede transportar. Donde $U[i, j]$ nos da el peso mínimo de los objetos que podemos transportar para lograr exactamente el valor de j .

$$U[i, j] = \min(U[i - 1, j], U[i - 1, j - v_i] + w_i)$$

para valores fuera de límites se toma $+\infty$ o cualquier valor mayor q W , con $U[0, 0] = 0$



Problema de la Mochila (6)



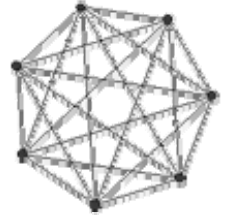
- Esta aproximación requiere un tiempo $O(n \log n + nM)$
- Y se invierte un tiempo al final de $O(M)$ para explorar la n -ésima fila U .
- Esta tabla U se puede utilizar para determinar no solo su carga óptima, sino su composición, al igual que el algoritmo de programación dinámica
- Al emplear esta aproximación resulta ligeramente mas complicada y menos natural que el algoritmo de programación dinámica
- Es preferible cuando los valores son mas pequeños que los pesos

$$V_i < W_i$$

ya que el tiempo de la solución óptima depende mas de los valor total de la solución, en vez de la capacidad a efectos de pesos



Coloreado de grafos



Problema de decisión:

Dados un grafo G y un número entero positivo m , ¿es G m -coloreable?

Es decir, ¿se puede pintar con colores los nodos de G de modo que no haya dos vértices adyacentes con el mismo color y se usen sólo m colores?

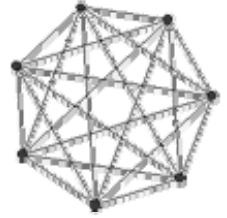
Problema de optimización:

Dado un grafo G , ¿cuál es su número cromático?

Es decir, ¿cuál es el menor número m de colores con el que se puede colorear G ?



Coloreado de grafos



El problema que consideraremos aquí:

- Dado un grafo cualquiera, determinar todas las formas posibles en las que puede pintarse utilizando no más de m colores.
- Representación elegida: matriz de adyacencias.

```
tipo grafo = vector[1..n,1..n] de bool
```

- Justificación de la elección: sólo necesitaremos saber si un arco existe o no.
- Representación de los colores: enteros de 1 a m .

```
tipo color = 0..m
```

- Representación de la solución: vector de colores.

```
tipo sol = vector[1..n] de color
```



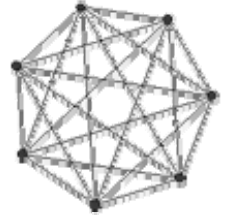
Coloreado de grafos



```
algoritmo_m_coloreado(ent k:entero;  
                      entsal x:sol)  
{Se usa una variable global g de tipo grafo.  
 En x se tiene la parte de la solución ya calculada  
 (es decir, hasta x[k-1]) y k es el índice del  
 siguiente vértice al que se va a asignar color.}  
principio  
  repetir  
    {generar todos los colores 'legales' para x[k]}  
    siguienteValor(x,k); {x[k]:=un color legal}  
    si x[k]≠0 {se ha encontrado un color legal}  
      entonces  
        si k=n  
          entonces escribir(x)  
          sino m_coloreado(k+1,x)  
        fsi  
      fsi  
    hastaQue x[k]=0  
fin
```



Coloreado de grafos



```
algoritmosiguienteValor(entsal x:sol;  
                        ent k:entero)  
{x[1]...x[k-1] tienen colores asociados de forma  
 que todos los vértice adyacentes tienen distinto  
 color.  
 x[k] tiene el anterior color para el que se ha  
 probado (0 si no se ha probado con ninguno).  
 Se calcula el siguiente color para x[k]  
 diferente del de todos los vértices adyacentes  
 a k (0 si no hay ninguno).}  
variableencontrado:booleano; j:entero  
principio  
  repetir  
    x[k]:=(x[k]+1) mod (m+1); {siguiente color}  
    si x[k]≠0  
      entonces  
        encontrado:=verdad;  
        j:=1;  
        mq (j≤n) and encontrado hacer  
          si g[k,j] and (x[k]=x[j])  
            entoncesencontrado:=falso  
            sino j:=j+1  
          fsi  
        fmq  
      fsi  
    hastaQue(x[k]=0) or encontrado  
fin
```



UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA

Coloreado de grafos



```
...  
{g contiene la matriz de adyacencia del grafo}  
para i:=1 hasta n hacer  
  x[i]:=0  
fpara  
m_coloreado(1,x);  
...
```



Coloreado de grafos



Cota superior del coste temporal:

- Número de nodos internos del árbol del espacio de estados:

$$\sum_{i=0}^{n-1} m^i$$

- Coste de 'siguienteValor' para cada nodo interno:

$$O(mn)$$

- Cota del tiempo total:

$$\sum_{i=1}^n m^i n = n(m^{n+1} - 1)/(m - 1) = O(nm^n)$$

Recordar que ya vimos una heurística voraz muy eficiente...





Coloreado de grafos

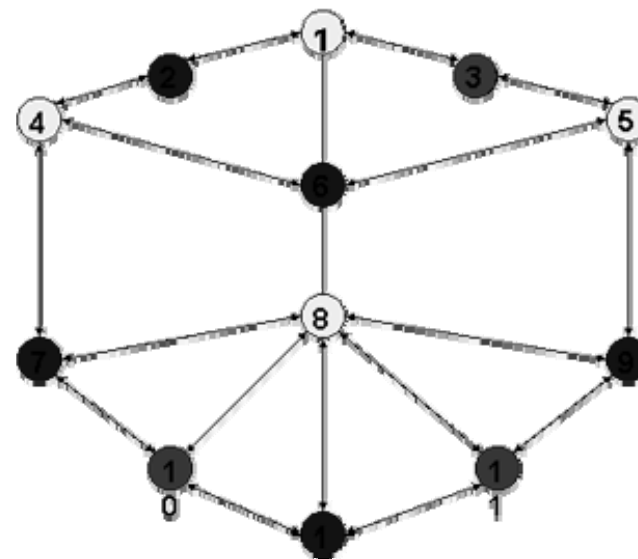


El Código del programa se anexa al presente informe dentro del archivo con el nombre de **mapaKcoloreado.cpp**, de igual forma el archivo de entrada de la matriz de adyacencia (grafo.dat) y el la salida del programa el grafo.txt.

Los casos de pruebas que se hicieron al programa son los siguientes:

EL GRAFO 1 SE PINTO DE LA SIGUINETE MANERA

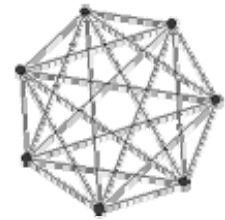
```
Colores: 3          K: 5
Nodo 1 color:      Amarillo
Nodo 2 color:      Azul
Nodo 3 color:      Rojo
Nodo 4 color:      Amarillo
Nodo 5 color:      Amarillo
Nodo 6 color:      Azul
Nodo 7 color:      Azul
Nodo 8 color:      Amarillo
Nodo 9 color:      Azul
Nodo 10 color:     Rojo
Nodo 11 color:     Rojo
Nodo 12 color:     Azul
```





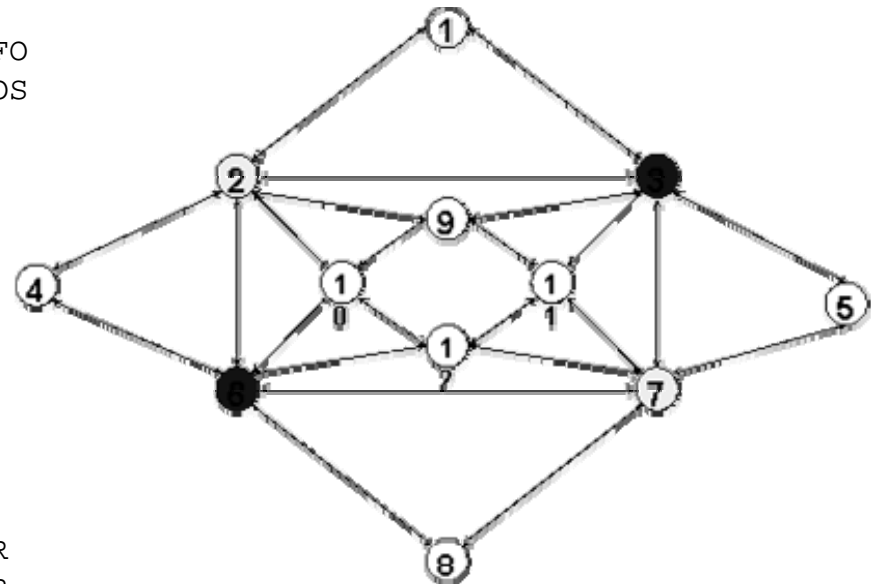
UNIVERSIDAD
DE LOS ANDES
MERIDA VENEZUELA

Coloreado de grafos



No Hay Colores suficientes para Colorear el GRAFO
!!!!GRAFO 10 SE COLOREARON LOS SIGUIENTES NODOS

Colores: 2 K: 2
Nodo 1 Color: NO COLOREADO POR FALTA DE COLOR
Nodo 2 color: Amarillo
Nodo 3 color: Azul
Nodo 4 Color: NO COLOREADO POR FALTA DE COLOR
Nodo 5 Color: NO COLOREADO POR FALTA DE COLOR
Nodo 6 color: Azul
Nodo 7 color: Amarillo
Nodo 8 Color: NO COLOREADO POR FALTA DE COLOR
Nodo 9 Color: NO COLOREADO POR FALTA DE COLOR
Nodo 10 Color: NO COLOREADO POR FALTA DE COLOR
Nodo 11 Color: NO COLOREADO POR FALTA DE COLOR
Nodo 12 Color: NO COLOREADO POR FALTA DE COLOR





UNIVERSIDAD
DE LOS ANDES
MÉRIDA VENEZUELA



?

Gracias 😊