Algoritmo de la mediana

```
Calcular la matriz de distancias D_{N×N} = [d_{ij}]
Min ← ∞
para i ← 1 hasta N hacer
   { Suma ← 0
     para j ← 1 hasta N hacer
        Suma ← Suma + D[i, j]
     si Suma < Min entonces
        Min ← Suma
     S[i] ← Suma
   }
Todos los elementos del vector S cuyos valores sean
iguales a Min son medianas.
```

Algoritmo del centro

```
Calcular la matriz de distancias D_{N×N} = [d_{ij}]
Min ← ∞
para i ← 1 hasta N hacer
   { Max ← -∞
     para j ← 1 hasta N hacer
        si D[i, j] > Max entonces
           Max ← D[i, j]
     si Max < Min entonces
        Min ← Max
     M[i] ← Max
   }
Todos los elementos del vector M cuyos valores sean
iguales a Min son centros.
```

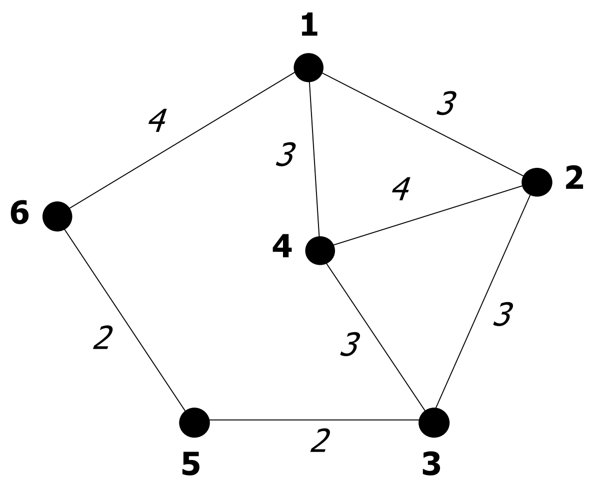


Figura 1: Grafo no dirigido.

D=

(i, j)	1	2	3	4	5	6	Max	Suma
1	*0*	*3*	*6*	*3*	*6*	*4*	***6****	*22*
2	*3*	*0*	*3*	*4*	*5*	*7*	*7*	*22*
3	*6*	*3*	*0*	*3*	*2*	*4*	***6****	***18****
4	*3*	*4*	*3*	*0*	*5*	*7*	*7*	*22*
5	*6*	*5*	*2*	*5*	*0*	*2*	***6****	*20*
6	*4*	*7*	*4*	*7*	*2*	*0*	*7*	*24*

Tabla 1: Centro y mediana para el grafo anterior.

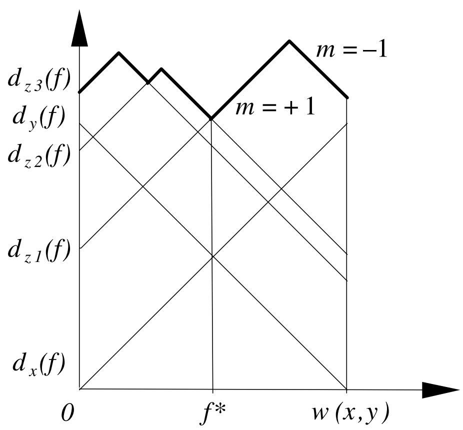


Figura 2: Envolvente superior de la función distancia d(f).

	Vértices							
	z_1	z_2	z_3	z_4	z_5	z_6	z_7	
$d(x, z_i)$	*40*	*38*	*36*	*32*	*30*	*29*	*29*	
$d(y, z_i)$	*32*	*29*	*30*	*27*	*33*	*34*	*38*	
r	*40*	*40*	*39*	*37*	***36***	***36***	*36.5*	*38*
f*	*0*	*2*	*3*	*5*	***6***	***7***	*7.5*	*10*

Tabla 3: Ejemplo numérico.

	Vértices			
	z_1	z_2	z_3	
$d(x, z_i)$	*10*	*8*	*6*	
$d(y, z_i)$	*11*	*9*	*5*	
r	*10*	*10*	*9*	*11*
f*	*0*	*2*	*3*	*1*

Tabla 2: Otro ejemplo numérico.

Algoritmo de Minieka

```
Calcular la matriz de distancias D_{N×N} = [d_ij]
para todo vértice x∈V hacer
   Ordenar todos los vértices de V en orden decreciente
   según su distancia a x
Radio_Min ← ∞
para toda arista (x, y)∈E hacer
   { Sea X ← {z_1, z_2,...,z_N} el conjunto de los vértices
     ordenados para el vértice x
     Y ← ∅
     r* ← d(x, z_1)
     f* ← 0
     X ← X – {z_1}
     Y ← Y ∪ {z_1}
     Ymax ← d(y, z_1)
     para todo elemento z∈X hacer
        { si d(y, z) ≥ Ymax entonces
             { r ← [d(x, z) + Ymax + w(x, y)] / 2
               f ← r – d(x, z)
               si r < r* entonces
                  { r* ← r
                    f* ← f
                  }
               Ymax ← d(y, z)
             }
        }
     si Ymax < r* entonces
        { r* ← Ymax
          f* ← w(x, y)
        }
     si r* < Radio_Min entonces
        Radio_Min ← r*
   }
Todos los puntos f* cuyo radio r* sea igual a Radio_Min
son centros absolutos del grafo.
```

Algoritmo de Kariv & Hakimi

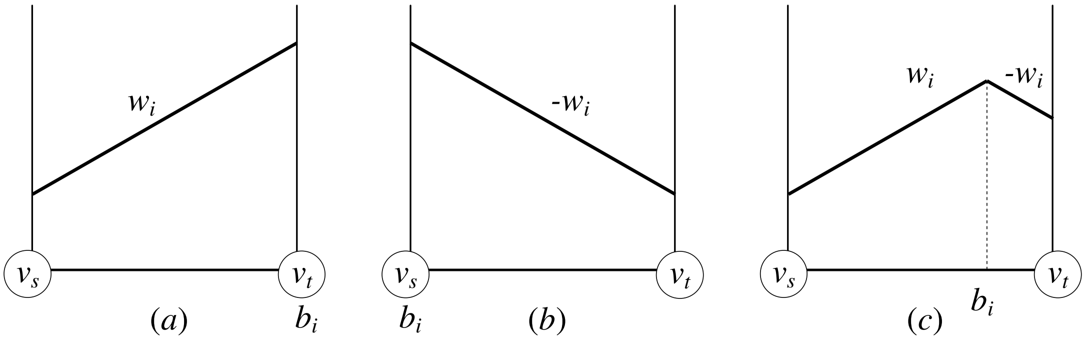
```
para cada vértice v∈V hacer
   Construir la lista L(v) con todos los vértices de la
   red en orden decreciente con respecto a sus
   distancias a v.
para cada arista e=(v_r, v_s)∈E hacer
   { v* ← Primer vértice de la lista L(v_r)
     v̂ ← Primer vértice de la lista L(v_s)
     si D_e(v*, 0) ≤ D_e(v̂, l(e)) entonces
        { t* ← 0
          r ← D_e(v*, 0)
        }
     en otro caso
        { t* ← l(e)
          r ← D_e(v̂,l(e))
        }
     si v* = v̂ entonces PARAR
     v_m ← v*
     i ← 1
     mientras (i < N) y (D_e(v*, 0) = D_e(v_m, 0)) hacer
        { si D_e(v*, l(e)) > D_e(v_m, l(e)) entonces
              v_m ← v*
          i ← i + 1
          v* ← Vértice i-ésimo de la lista L(v_r)
        }
     v̄* ← v_m
     mientras i ≠ N hacer
        { v_m ← v*
          i ← i + 1
          v* ← Vértice i-ésimo de la lista L(v_r)
          mientras D_e(v*, 0) = D_e(v_m, 0) hacer
             { si D_e(v*, l(e)) > D_e(v_m, l(e)) entonces
                   v_m ← v*
               i ← i + 1
               v* ← Vértice i-ésimo de la lista L(v_r)
             }
```

```
        si las funciones Dₑ(vₘ,·) y Dₑ(v̄*,·) se cruzan
        entonces
           { Calcular el punto tₘ donde se cruzan
             si Dₑ(vₘ, tₘ) < r entonces
                { t* ← tₘ
                  r ← Dₑ(vₘ, tₘ)
                }
           }
        si Dₑ(v*, l(e)) > Dₑ(v̄*, l(e)) entonces
           v̄* ← vₘ
     }
   si las funciones Dₑ(v*,·) y Dₑ(v̄*,·) se cruzan
   entonces
     { Calcular el punto tₘ donde se cruzan
        si Dₑ(vₘ, tₘ) < r entonces
           { t* ← tₘ
             r ← Dₑ(v*, tₘ)
           }
     }
}
```

El radio mínimo de todas las aristas de la red es el 1-radio de la red, y cualquier centro local que tenga como radio el 1-radio es un 1-centro de la red.

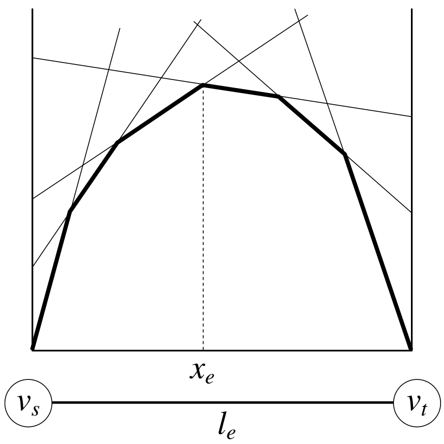
$$d(x,v_i) = \min\{x + d(v_s,v_i), l_e - x + d(v_t,v_i)\}$$

$$b_i = \frac{d(v_t,v_i) + l_e - d(v_s,v_i)}{2}$$



$$f(x) = \min_{v_i \in V} w_i d(x,v_i)$$

$$\max_{x \in N} \min_{v_i \in V} w_i d(x,v_i) = \max_{x \in N} f(x)$$



$$f(x_e) = \max_{x \in e} f(x)$$

$$d_k = d(v_s,v_k) - d(v_t,v_k)$$

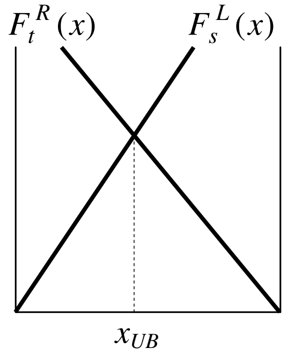
$$L = \{v_k \in V : d_k < l_e\}, \qquad R = \{v_k \in V : -d_k < l_e\}$$

$$v_i \in V, \qquad \begin{aligned} F_i^L(x) &= w_i(x + d(v_s,v_i)) \\ F_i^R(x) &= w_i(l_e - x + d(v_t,v_i)) \end{aligned}$$

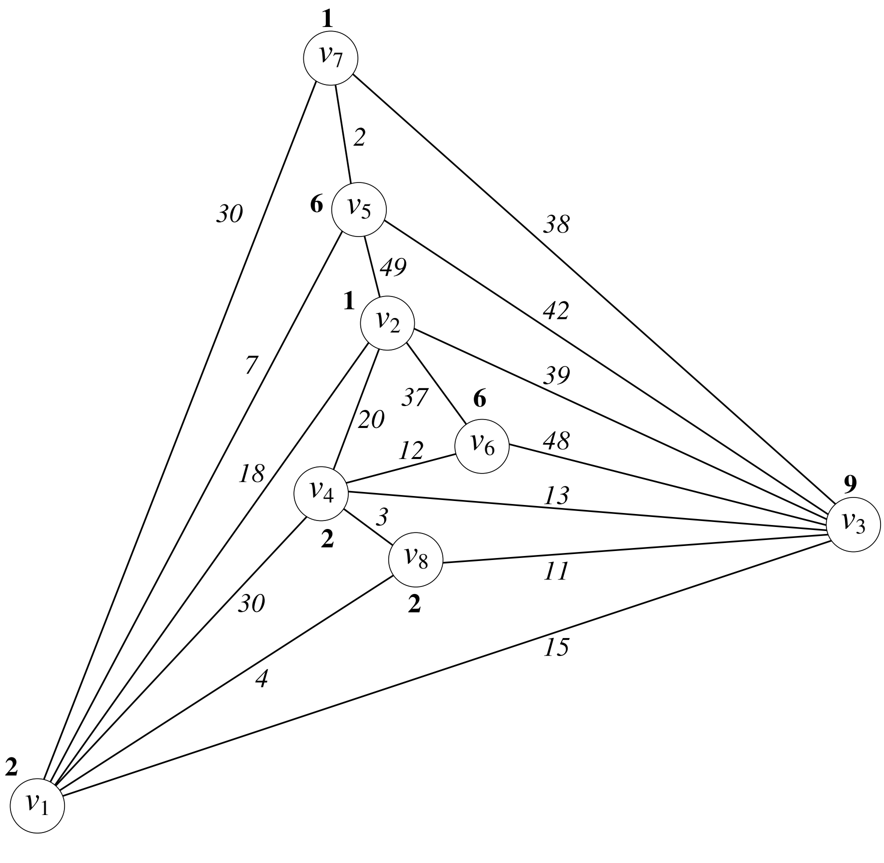
$$v_i \in L,\ v_j \in R, \qquad X(v_i,v_j) = \frac{w_j(l_e + d(v_t,v_j)) - w_i d(v_s,v_i)}{w_i + w_j}$$

$$P_e = \{X(v_i,v_j) : \forall v_i \in L,\ \forall v_j \in R\}$$

$$P_N = \bigcup_{e \in E} P_e, \quad |P_N| \le mn^2$$



$x_{UB} = X(v_s, v_t)$ $\qquad\qquad$ $F_{UB} = F_s^L(x_{UB}) = F_t^R(x_{UB})$



Solución $F_N = 50$, $S = \{(26, e_{36})\}$

Algoritmo del UnCenter

```
Calcular la matriz de distancias d_{n×n} = [d_ij]
F_N ← 0
S ← ∅
para toda las aristas e=(v_s,v_t)∈E hacer
   { x_UB ← X(v_s,v_t)
     F_UB ← F_s^L(x_UB)
     si F_N > F_UB entonces
        Continuar con la siguiente arista
     Crear los conjuntos L y R (todas las lineas deben
     estar por debajo de F_UB)
     x_e ← x_UB
     F_e ← F_UB
     mientras (F_e ≥ F_N) y (L ≠ ∅ o R ≠ ∅) hacer
        { Emparejar todos los nodos en L y R usando un
           emparejamiento de cardinalidad max{|L|,|R|}
          Guardar en (x_e,F_e) el punto de intersección
           con mínimo valor de función
          Proyectar el valor x_e en la envoltura inferior
           usando:
```

$$v_a \in L : F_a^L(x_e) = \min_{v_k \in L} F_k^L(x_e), \quad v_b \in R : F_b^R(x_e) = \min_{v_k \in R} F_k^R(x_e)$$

```
          x_e ← X(v_a,v_b)
          F_e ← F_a^L(x_e)
          Eliminar de L y R todos los nodos cuyas
           lineas estén por encima de F_e
        }
     si F_e ≥ F_N  entonces
        { F_N ← F_e
          Guardar (x_e,e) en S
        }
   }
retornar (F_N,S)
```

Ejemplo de emparejamiento de maxima cardinalidad:

$$L = \{v_1, v_3, v_4\}, \; R = \{v_2, v_3, v_5, v_7, v_8\}$$

$$(v_i \in L, v_j \in R) : (v_1, v_2), \; (v_3, v_3), \; (v_4, v_5), \; (v_1, v_7), \; (v_3, v_8)$$