Problema de Floyd

Descripción: *Encontrar el camino mínimo entre dos vértices de un grafo*

Técnica: *Programación Dinámica Sin Filtro*

Tipos

- S – GraphPath<V,E>
- A – {Y,N}

Propiedades Compartidas	*g, Graph<V,E>, básica* *vt, List <V>, lista de vértices del grafo, derivada* *n, Número de vértices, derivada,* *or, V, origen, básica* *d, V, destino, básica* i_0*, Integer, origen, derivada, or = vt(*i_0*)* j_0*, Integer, origen, derivada, d = vt(*j_0*)*
Propiedades Individuales	*i, entero en [0,n)* *j, entero en [0,n)* *k, entero en [0,n)*

Solución: GraphPath<V,E>

Tamaño: *n-k*

Solucion Parcial: *(a,v)*

Siendo v el valor de camino mínimo de i a j usando los vértices en el conjunto [k,n) en el camino

Alternativas: $A = \{Y, N\}$

Instanciación

$Inicial = (i_0, j_0, 0)$,

Problema Generalizado

(i, j, k)

Ir del vértice i al j, usando un camino que contenga vértices ubicados en el conjunto [k,n).

Caso Base

$g.ea(i,j) || k = n$

Solución Caso Base

$$\begin{cases} (N, g.pa(i,j)), & g.ea(i,j) \\ \perp, & !\, g.ea(i,j) \end{cases}$$

Subproblemas

$$p = (i,j,k) \xrightarrow{N,0} p_{N,0}\ (i,j,k+1)$$
$$p = (i,j,k) \xrightarrow{Y,0} p_{Y,0}(i,k,k+1)$$
$$p = (i,j,k) \xrightarrow{Y,1} p_{Y,1}(k,j,k+1)$$

Esquema Recursivo

$sp(p) = \begin{cases} (N, g.pa(i,j)), & g.ea(i,j) \\ \perp, & !\ g.ea(i,j) \\ min\{(Y, sp\left(p_{Y,0}\right).v + sp\left(p_{Y,1}\right).v), (N, sp\left(p_{N,0}\right).v)\}, & en\ otro\ caso \end{cases}$	
$\boldsymbol{cS}(Y, (a', v'), (a'', v'')) = (Y, v' + v'')$ $\boldsymbol{cS}(N, (a', v)) = (N, v)$)	
$\boldsymbol{sA}$: Elige la alternativa que proporciona la solución parcial menor según el orden natural	
Solución reconstruida $sr(N, v) = (g.ar(i,j))$, es un caso base $sr(N, p) = p$, es un caso recursivo en cuya solución parcial era la alternativa N $sr(Y, p1, p2) = p1 + p2$, es un caso recursivo en cuya solución parcial era la alternativa Y	
Complejidad	n^3

Notación GraphPath<V,E>

- $p1 + p2$, *Unión de las aristas en p1 y p2*
- (ar), *Camino formado por la arista ar*

Notación Listas

- *s(i), Posición i en la lista s*

ProblemaFloyd *g, vt, n,or,d, (i,j,k)*

- *g, Grafo inicial*
- *vt, Vértices del grafo*
- *n, Número de vértices*
- *or, Vértice origen del problema inicial*
- *d, Vértice destino del problema inicial*
- *i0, Vértice origen del problema original*
- *j0, Vértice destino del problema original*
- *i, Vértice origen*
- *j, Vértice destino*
- *k, Los vértices en posiciones [k,n) pueden ser vértices intermedios*

Notación

- $\perp$, *No existe solución*
- *vt(i), Vértice en la posición i de la lista vt*
- $g.ea(i,j)$, *Existe arista en el grafo g entre los vértices en posiciones i, j*
- $g.ar(i,j)$, *Arista, si existe, en el grafo g entre los vértices en posiciones i, j*
- $g.pa(i,j)$, *Peso de la arista en el grafo g entre los vértices en posiciones i, j*

Operadores y Funciones

- *sA: Selecciona alternativa*
- *cS: Combina soluciones de subproblemas*
- *sr: Solución Reconstruida*