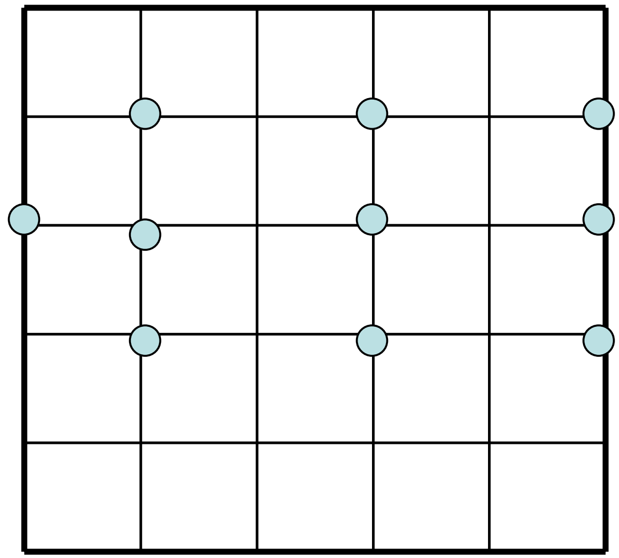
Problemas con Grafos

- Reconocer que es un problema con grafos
- Representar el grafo
- Recorrer ordenadamente el grafo

El Problema del escape

Una malla de n x n es un grafo no dirigido que consiste de n filas y n columnas de vértices. Denotamos el vértice en la i-esima fila y la j-esima columna como (i,j). Todos los vértices en la malla tienen exactamente 4 vecinos, excepto los que estan en la frontera (i=1, i=n, j=1 o j=n).

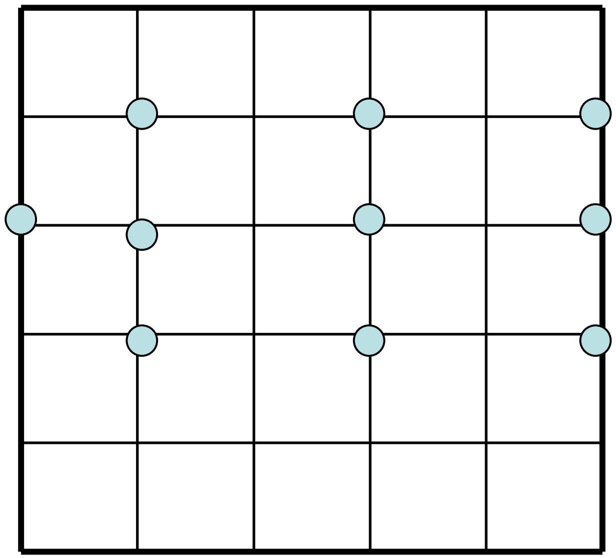
El Problema del escape



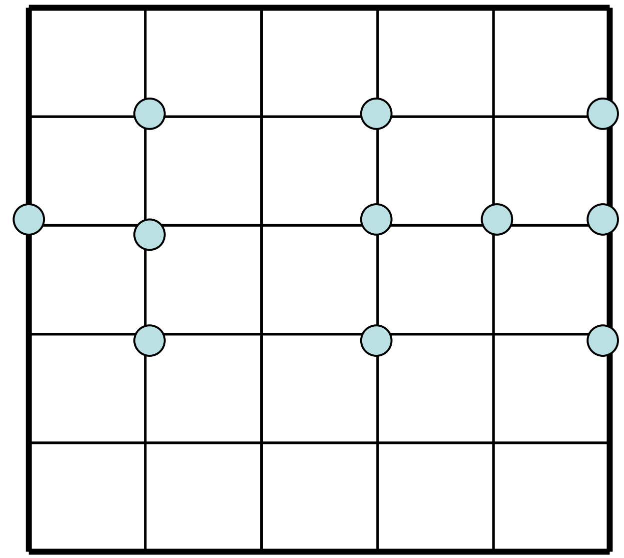
El Problema del escape

dados m<= n^2 puntos iniciales (vertices) en la malla, el problema del escape consiste en determinar si hay o no un camino desarticulado para llegar desde cada punto inicial hasta un punto en el borde de la malla.

El Problema del escape



Malla con solución.



Malla sin solución.

El Problema del escape (el verdadero problema)

-). Considerando una red de flujo en la que tanto vertices como aristas tienen capacidades, es decir, el flujo neto que entra en cualquier vértice está sujeto a un limite de capacidad. Muestre que determinar el flujo máximo en una red con vértices y aristas limitadas puede ser reducido a un ordinario problema de flujo máximo en una red de flujo de tamaño comparable.

El Problema del escape (el verdadero problema)

- Describa una algoritmo eficiente para resolver el problema del escape, y haga un análisis de su tiempo de ejecución.

Algoritmo

- ¿Cuál es la tasa mayor a la cual el material puede ser transportado de la fuente al resumidero sin violar ninguna restricción de capacidad?

Algoritmo

```
Ford-Fulkerson(G,s,t) {
     for (cada arco (u,v) de E) { //inicializacion
           f[u,v]= 0;
           f[v,u]= 0;
    }
    while (exista un camino p desde s a t en la red residual Gf) { //arcos que soporten más flujp
           cf(p) = min{cf(u,v) : (u,v) está sobre p}; //mínimo a llevar
           for (cada arco (u,v) en p) {
                      f[u,v]= f[u,v] + cf(p) ;
                      f[v,u]= - f[u,v];
           }
    }
 }
//deja de existir un camino cuando tiene capacidad 0.
O(V^2*E)
```

Algoritmo Mejorado

PUSH(*u, v*)

/***Applies when**: *u* is overflowing, *cf* [*u, v*] > 0, and *h*[*u*] = *h*[*v*] + 1.

Action: Push *df* (*u, v*) = min(*e*[*u*], *cf*(*u, v*)) units of flow

From *u* to *v*.*/

df(*u, v*) <- min(*e*[*u*], *cf*(*u, v*))

f[*u, v*] <- *f*[*u, v*] +*df*(*u, v*)

f[*v, u*] <- - f[u, v]

e[*u*] <- *e*[*u*] - dphi (u, v)

e[*v*] <- *e*[*v*] + *df*(*u, v*)

end

Algoritmo Mejorado

LIFT(u)

/* **Applies when:** u is overflowing and for all v V, (u, v) Є Ef *implies* $h[u] <= h[v]$.

Action: Increase the height of u.*/

$h[u]$ <- $1 + \min \{h[v]: (u, v)$ Є Ef$\}$

Algoritmo Mejorado (inicialización)

$$f[u,v] = \begin{cases} c(u,v) & \text{if } u = s\,, \\ -c(v,u) & \text{if } v = s\,, \\ 0. & \text{otherwise}\,. \end{cases} \qquad (27.$$

$$h[u] = \begin{cases} |V| & \text{if } u = s\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

Algoritmo Mejorado

```
DISCHARGE(u)
    while e[u] > 0 do
        v <- current[u]
        if   v= NIL  then
                LIFT(u)
                current[u] <-  head[N[u]]
        elseif cf  (u,v ) > 0 and h[u] = h[ v] + 1 then
                PUSH(u, v )
        else
                current[u]<-   next-neighbor[ v]
        end
    end
end
```

Algoritmo Mejorado

```
LIFT-TO-FRONT(G, s, t)
    INITIALIZE-PREFLOW(G, s)
    L   V[G] - {s, t}, in any order
    for each vertex u   V[G] - {s, t} do
        current[u]  <- head[N[u]]
    u  <-  head[L]
    while u !=  NIL do
        old-height <-  h[u]
        DISCHARGE(u) //está linea se llama |V| veces
        if h[u] > old-height then
            move u to the front of list L
        u  <- next[u]
    end
end
O(V^3)
```

Volviendo al problema

- **Generaremos dos vértices ficticios (super-s y super-t), habrá una arista entre super-s y las s ordinarias con capacidad de uno (capacidad máxima de los s ordinarios), igualmente con super-f y f pero las aristas tienen capacidad infinita.**

Siguiendo con el problema

- **La restricción en los vértices debe estar en las aristas. Pero ¿Cómo?**
- **Construimos una red con dos veces la cantidad de vértices que el inicial, y para cada nodo original ahora tenemos dos vértices (el de entrada y el de salida) que están unidos por una arista con capacidad igual a la capacidad del nodo en cuestión.**

aún con el problema

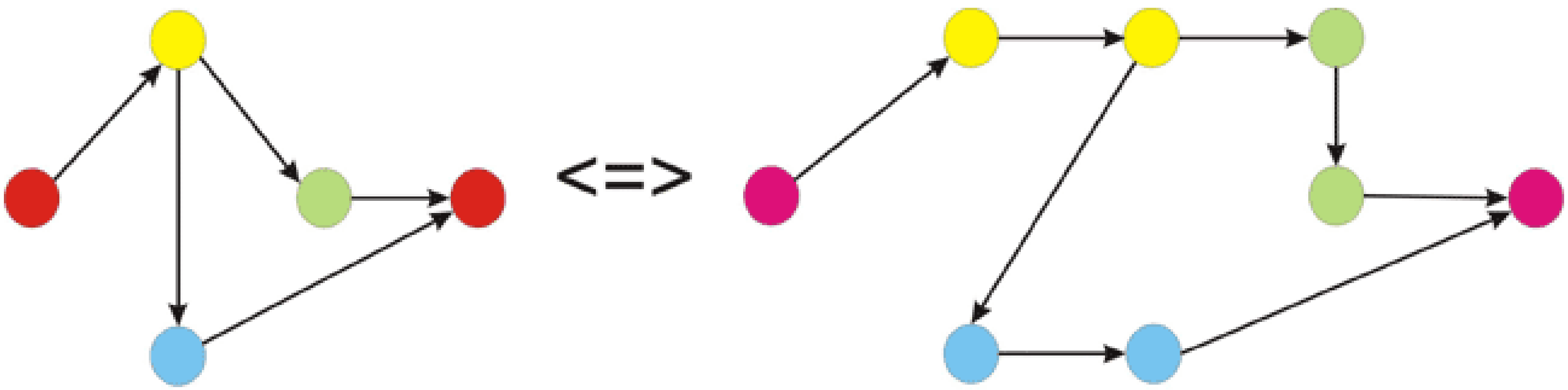


Figure 7 - Eliminating vertex-capacities

Gracias.