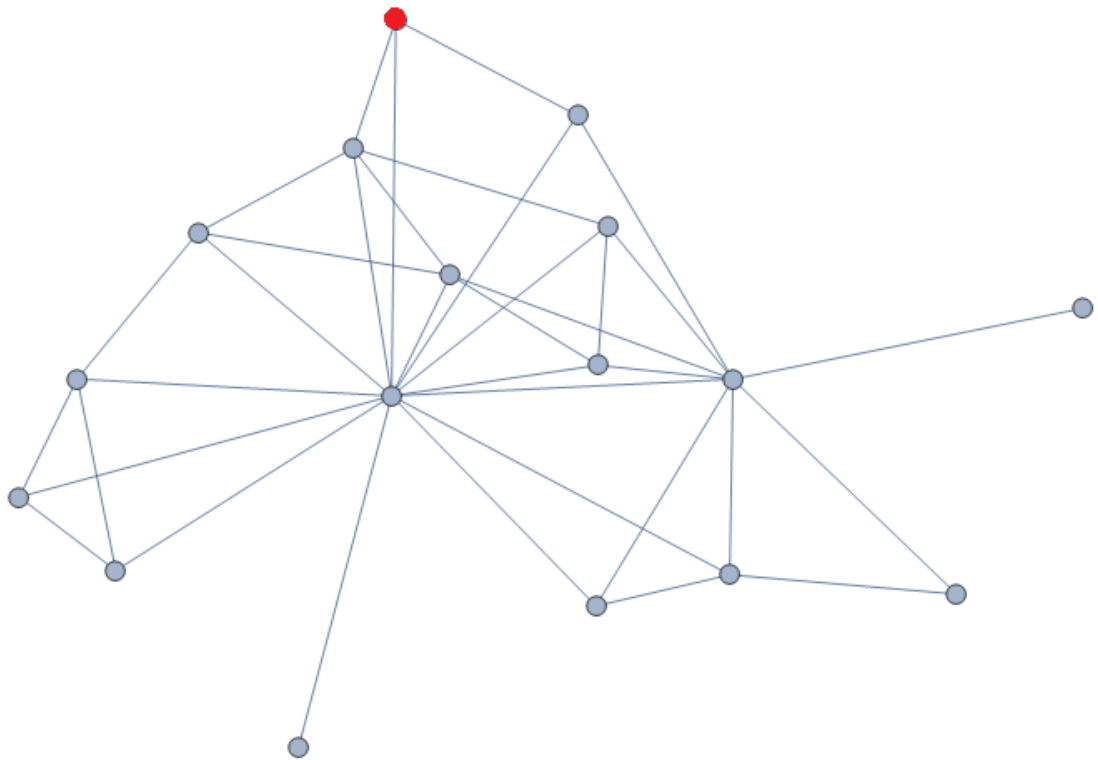


TAREA 1

Quinta Entrega: Grafos



INTEGRANTES:

Andrés Monetta
Alejandro Clara
Sebastián Daloia

Contenido

- Ejercicio 1.....3
 - Parte a).....3
 - Parte b).....4
 - Parte c).....4
 - Parte d).....4
- Ejercicio 2).....5
 - Parte a).....5
 - Parte b. i).....5
 - Parte b.ii).....6
- Ejercicio 3).....7

Ejercicio 1

Parte a)

/*Pre: Portador ya infectado como activo. Inoculado ya infectado como inmune*/
PROCEDIMIENTO CantidadSanos(comunidad : Grafo, portador : vertice, inoculado : vertice)

```
VAR contador : entero
COMIENZO
    contador=0
    propagacion_BFS(portador)
    sanos_BFS(inoculado)
    RETORNAR contador
FIN
```

PROCEDIMIENTO propagacion_BFS(portador : vertice)

```
VAR Q : Cola
    u : Vertice
COMIENZO
    marcar(portador)
    insBack(a, Q)
    Mientras no_vacia(Q)
        u=primeroCola(Q)
        desencolar(Q)
        Para cada w adycente a u
            Si w no marcado entonces
                Si w esInmune entonces
                    marcar(w)
                Fin si
                Si esSano(w) y u esActivo(w) entonces
                    setLatente(w)
                    marcar(w)
                    insBack(w)
                Fin Si
                Si esSano(w) y u esLatente(w) entonces
                    setActivo(w)
                    marcar(w)
                    insBack(w)
                Fin Si
            Fin si
        Fin para
    Fin Mientras
FIN
```

```

PROCEDIMIENTO sanos_BFS(inoculado: vertice)
    VAR Q : Cola
        u : Vertice
    COMIENZO
        insBack(inoculado)
        Mientras NoVacia(Q)
            u=primeroCola(Q)
            desencolar(Q)
            Para cada w adyacente a u
                Si w no marcado entonces
                    marcar(w)
                    contador++
                    insBack(w)
            Fin si
        Fin Para
    Fin Mientras
FIN

```

Parte b)

Todo camino que comience en un infectado y culmine en un sano debe contener entre sus vértices al inmune (inoculado), en caso contrario la infección no se detendría y continuaría el recorrido por la comunidad.

Por lo que el subgrafo obtenido al borrar el vértice inmune contiene más componentes conexas que el original.

Es decir el vértice inmune es un punto de articulación.

Parte c)

Realizar una recorrida para cada vértice, salvo el Activo inicial, guardar la cantidad de sanos que quedarían para el vértice que es inmunizado en cada pasada. Luego devolver al individuo que mayor de individuos sanos deje.

Parte d)

El grafo es conexo y no dirigido, se puede plantear la siguiente solución.

Hacer una recorrida DFS desde el activo.

Como el grafo es no dirigido no habrá aristas cross entre las ramas generadas del árbol.

Contar para cada rama del árbol generado por DFS la cantidad de vértices que tienen.

Elegir para inocular el vértice adyacente al activo en el árbol DFS que tenga más vértices en el camino hasta la hoja (sumidero).

Ejercicio 2)

Parte a)

Es posible solamente en el caso de que exista un camino simple que comunique a todos los activos y si existiese otro camino simple que comunique a todos los infectados latentes.

En otro caso existe al menos un infectado del otro estado que esté actuando como punto de articulación y evite la propagación de la cura.

Parte b. i)

EsPeorEscenario(comunidad : Grafo) : boolean

VAR esPeor: boolean

COMIENZO

```
    esPeor=verdadero
    Para cada v que pertenece a V
        inicializar v como no marcado
    Fin para
    Para cada v
        Si v no marcado entonces
            esPeor=PeorEscenario_BFS(v)
            Si esPeor== falso entonces
                retornar esPeor
        Fin si
    Fin si
    Fin para
    retornar esPeor
```

FIN

```

PeorEscenario_BFS(v: vertice) : boolean
    VAR Q : Cola
        u : Vertice
        esPeor : boolean
COMIENZO
    esPeor= verdadero
    marcar(v)
    insBack(v, Q)
    Mientras no_vacia(Q)
        u=primeroCola(Q)
        desencolar(Q)
        Para cada w adyacente a u
            Si estado w == a estado u entonces
                esPeor=falso
            Fin si
            marcar(w)
            insBack(w)
        Fin para
    Fin Mientras
    RETORNAR esPeor
FIN

```

Parte b.ii)

El sugrafo inducido por los vértices en estado latente y el inducido por los vértices en estado activo cada uno de ellos no tiene aristas.

Ejercicio 3)

PROCEDIMIENTO `propagacion_DFS`(v, ik: `vertice`, lista : `Lista de vértices`,
encontrado: `boolean`)

COMIENZO

`marcar`(v)

`insLista`(v, lista)

 Para cada w adyacente a v

 Si w no `marcado` y w diferente de ik y encontrado == `falso` entonces

`propagacion_DFS`(w, ik, lista, b)

`borrarLista`(v, lista)

 Sino si w igual a ik entonces

`insLista`(w, lista)

 encontrado=`verdadero`

 Fin Si

 Fin para

FIN