Programa 1

Ian Israel García Vázquez

15 de septiembre de 2022

1. Problema de Alcanzabilidad

Dada una gráfica no dirigida $G = (V, E)$, con dos vértices distinguidos s y t. ¿Existe un camino que no repite vérticcecs de s a t en G?

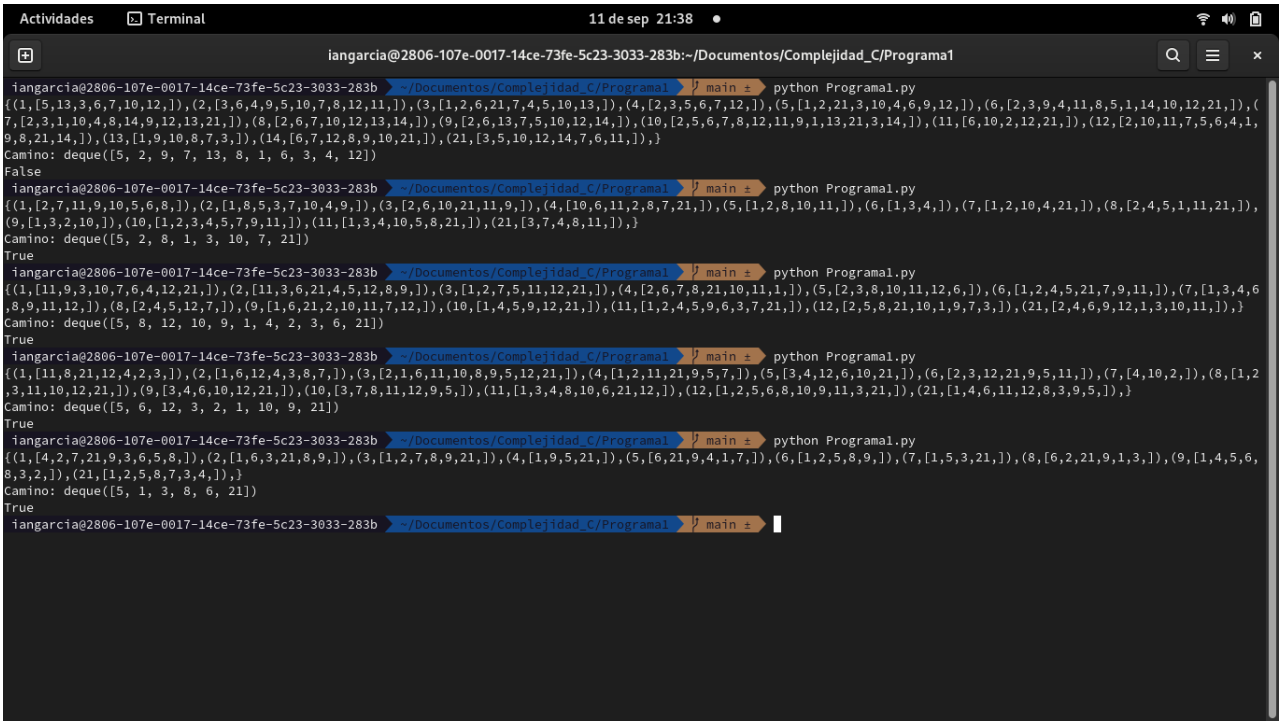
1.1. Forma canónica

Ejemplar genérico: Gráfica simple $G = (V, E)$
Enunciado de optimización: Determinar el camino C en G de s a t cuyo conjunto de vértices $|V_c| \leq |V|$.
Ejemplar genérico: Una gráfica simple $G = (V, E)$
Pregunta de decisión: ¿Existe un camino C en G con los vértices distinguidos s, t?

Algoritmo:

```
proceso generaRuta(Gráfica G, Vertice S, Vertice T):
IMPRIME(G)
contadorDeLecturas=0
Stack ruta=[]
Vertice V = S;
ruta.agrega(V)
mientras TIENEVECINOS(V):
        mientras V está en ruta:
            contadorDeLecturas++
            V=ESCOGE_VECINO_AL_AZAR[V]
            if contadorDeLecturas >= NoVecinos(V):
                break
            if V no está en ruta:
                ruta.agrega(V)
                contadorDeLecturas=0
            if T == V ó contadorDeLecturas>=NoVecinos(V):
                break

IMPRIME(ruta)
return ULTIMOAGREGADO(ruta)==T
```



En esta imagen tenemos la gráfica generada y las soluciones que nos otorgan el programa, así como su evaluación, nótese que para este caso debemos nuestra $s = 5$ y $t = 21$

2. Problema 3SAT

3-SAT
Una variante del problema de satisfacibilidad, dadas n-variables y m-clausulas, determinar si la preposición es satisfacible, en este caso cada causula deberá conformarse de al menos tres variables.

2.1. Forma canónica

Ejemplar genérico: Conjunto de Clausulas $C_1 * C_2 \cdots C_i$, tal que cada claúsula se conforma de k-variables, con $k = 3$

Pregunta de decisión: ¿Es satisfacible el conjunto de k claúsulas si cada clausúla debe componerse de al menos 3 claúsulas?

Algoritmo:

```
PROCESO 3SAT(LC <- Lista de clausulas):
    boolean evaluacion;
    boolean fst_time=true;
    for Clausula from LC:
        fst_time=true
        if fst_time then:
            evaluacion=Clausula.ValorBooleano
            fst_time=false
        else:
            evaluacion&=Clausula.ValorBooleano
        endif
    return evaluation
```


Actividades
Terminal
15 de sep 16:46
iangarcia@fedora:~/Documentos/Complejidad_C/Programa1
iangarcia@fedora:~/Documentos/Complejidad_C/Programa1
iangarcia@fedora:~
iangarcia@fedora ~/Documentos/Complejidad_C/Programa1 Programa1.2 python 3SAT.py
Ejemplar [+[y(False), q(True), r(True)], +[-y(True), h(True), r(True)], +[e(True), h(True), r(True)], +[s(True), b(False), r(True)], +[q(True), n(False), r(True)]]
Respuesta: True
iangarcia@fedora ~/Documentos/Complejidad_C/Programa1 Programa1.2 python 3SAT.py
Ejemplar [+[e(True), f(True), -i(False)], +[-e(False), f(True), -i(False)], +[l(True), f(True), -i(False)], +[-f(False), z(True), -i(False)], +[-f(False), y(False), -i(False)]]
Respuesta: False
iangarcia@fedora ~/Documentos/Complejidad_C/Programa1 Programa1.2 python 3SAT.py
Ejemplar [+[p(False), b(False), m(True)], +[f(False), n(True), m(True)], +[w(True), b(False), m(True)], +[-w(False), -n(False), m(True)], +[-b(True), n(True), m(True)]]
Respuesta: True
iangarcia@fedora ~/Documentos/Complejidad_C/Programa1 Programa1.2 python 3SAT.py
Ejemplar [+[b(True), f(False), -j(True)], +[-f(True), r(True), -j(True)], +[o(False), x(True), -j(True)], +[c(True), -f(True), -j(True)], +[v(True), f(False), -j(True)]]
Respuesta: True
iangarcia@fedora ~/Documentos/Complejidad_C/Programa1 Programa1.2 python 3SAT.py
Ejemplar [+[z(True), b(True), t(True)], +[b(True), m(False), t(True)], +[-b(False), m(False), t(True)], +[j(False), h(False), t(True)], +[h(False), -m(True), t(True)]]
Respuesta: True
iangarcia@fedora ~/Documentos/Complejidad_C/Programa1 Programa1.2