Sinencio Granados Dante 316246019
Sandoval Mendoza Antonio 316075725
Reyes Martínez Antonio 316184931
Cruz Jiménez Alejandro 316008488

Práctica 1

Complejidad Computacional
Semestre 2022-1

Descripción del problema:

Empecemos describiendo cada problema:

Agente Viajero (TSP) :

Se tiene un número de nodos que deben ser visitados por una entidad, sin visitar dos veces el mismo nodo. Por ejemplo, si tenemos tres nodos: a, b y c, por visitar, entonces tendríamos una función de combinaciones sin repetición c(3,2), es decir, tendríamos 6 posibles soluciones: abc, acb, bac, bca, cab, cba, para el caso de 4 nodos tendríamos 12 combinaciones, y así sucesivamente.

MAX SAT :

Consiste en determinar el número máximo de cláusulas que se pueden satisfacer simultáneamente, de una fórmula booleana dada en forma normal conjuntiva, y que puede hacerse verdadera mediante una asignación de valores de verdad a las variables de la fórmula.

Es una generalización del problema de satisfacibilidad booleano , que pregunta si existe una asignación de verdad que hace que todas las cláusulas sean verdaderas. Además, este problema es NP-difícil , debido a que su solución conduce fácilmente a la solución del problema de satisfacibilidad booleano, el cual es NP-completo.

Descripción del esquema de codificación implementado:

- Para el problema del **TSP** utilizamos la siguiente codificación: $''a2b,\ b3c,\ c1a''$, donde las letras indican el nodo y los número el peso de la arista entre esos dos nodos.
- Para el problema del **MAX SAT** utilizamos la siguiente codificación: $''3(pu - q) * 5(rus)''$ donde los números son los pesos de las clausulas, 'u' es el símbolo para el $\vee$, '-' es la representacion de la negacion y el '*' es el símbolo para el $\wedge$.

Sinencio Granados Dante 316246019
Sandoval Mendoza Antonio 316075725
Reyes Martínez Antonio 316184931
Cruz Jiménez Alejandro 316008488

Práctica 1

Complejidad Computacional
Semestre 2022-1

Ejemplo de codificación de un ejemplar:

- Problema **TSP**:

  Un ejemplo de esta codificación puede ser: para la gráfica cuadrada con nodos: a, b, c, d y pesos de aristas: $2, 3, 4, 5$. La codificación seria la siguiente: $a2b$, $b3c$, $c4d$, $d5a$.

- Problema del **MAX SAT**:

  Un ejemplo de esta codificación puede ser: para las cláusulas: $(p \vee q) \wedge (r \vee s) \wedge (p \vee r)$, con pesos: 5, 6, 4. La codificación sería la siguiente: $5(p - q) * 6(r - s) * 4(p - r)$.

Ejemplos de ejecución del programa implementado:

Usaremos como ejemplo a la siguiente expresión para el **TSP**:

a	5	b
b	6	d
d	7	a
c	2	a
b	8	d

Primero sacaremos el numero de vértices, el programa hace la comparación de valores entre el primer y el segundo nodo (por ejemplo $a < b$), si se cumple se agrega un nodo a nuestro contador, si no se pasa a la siguiente línea, la única condición que tenemos al generar la codificación es que para líneas que creen más de un ciclo deben ponerse de manera inversa al orden alfanumérico (esto quiere decir que si se cumple la condición del ciclo entonces en lugar de poner axc pondremos cxa). Después, para el número de aristas solo contaremos la cantidad de pesos que hay, ya que toda arista debe tener un peso. Por último, con el peso máximo y mínimo sólo se irán comparando los pesos hasta obtener el valor mínimo y el máximo.

Por otra parte, para el **MAX SAT** usaremos la expresión:

$$2(PuQ) * 6(-QuRu - S) * 3(Su - T)$$

Primero dividiremos la expresión quitando los $\wedge$ (que en esta codificación son los $'*'$), de esta forma obtendremos cada cláusula en un arreglo y solo regresamos el tamaño de éste para obtener el total de cláusulas, ahora bien, para obtener el número de variables volveremos a separar cada string con las cláusulas en sub-arreglos pero ahora dividiéndolos por los $\vee$ (que en esta codificación son las $'u'$) y regresamos la cantidad de sub-arreglos para obtener el total

de variables a usar. Por último aprovechamos los arreglos con las cláusulas, las cuales se verían de la siguiente manera: $[\prime 2(PuQ\prime,\prime 6(QuRu - S)\prime, 3(Su - T)]$, como se puede ver los pesos son el primer carácter en todas las cadenas en el arreglo, así que separamos el primer carácter de todas las cadenas para parsearlo y sumarlo, y posteriormente dividirlo entre el tamaño del arreglo, logrando así obtener el peso promedio.

REFERENCIAS:

- Reinelt, G. (s. f.). TSPLIB 95. Universit¨at Heidelberg. Recuperado 13 de octubre de 2021, de http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp95.pdf
- https://en.wikipedia.org/wiki/Maximum_satisfiability_problem
- De Ita Luna, G., Morales-Luna, G. (1999). Propuestas Algorítmicas para la Resolución de los Problemas de Satisfacibilidad. Resumen de Tesis Doctoral.
- https://www.repositoriodigital.ipn.mx/bitstream/123456789/15111/1/ART%208.pdf

  Computación y Sistemas Vol. 2 No. 2-3 pp.140-145 © 1999, CIC - IPN. /SSN /405-5546 Impreso en México