Tarea: Algoritmo Set Covering

Lic. Arnoldo Del Toro Peña

8 de febrero de 2022

Resumen

Uso de un método metahurístico para la solución a un problema de set covering utilizando programación en python.

Palabras clave: python,set, covering, metahurística.

SECCIÓN 1

Introducción

En este documento se presentarán los resultados a las instancias obtenidas la siguiente dirección: people.brunel.ac.uk, se compararán con los resultados óptimos que se presentan a continuación:

Instancias de Prueba	*Nombre de la instancia*	*m*	*n*	*Óptimo*
4	scp41	200	1000	429
5	scp51	200	2000	253
6	scp61	200	1000	138
A	scpa1	300	3000	253
B	scpb1	400	3000	69
C	scpc1	400	4000	227
D	scpd1	400	4000	60
E	scpe1	50	500	5

Más adelante analizaremos las diferencias entre estos resultados y los que obtuvimos.

SECCIÓN 2

Descripción

El algoritmo que se utilizó fue modificado, la causa de ello es que uno de los archivos a trabajar (scpe1) tenía un conjunto de costos con la cualidad de que todos tenían valor unitario, lo cual en el algoritmo anterior lo que estaba haciendo era seleccionar de manera casi aleatoria cualquier centro, claro esta que no era lo deseado.

SECCIÓN 3

Descripción del algoritmo

Las modificaciones que se hicieron fueron las siguientes:

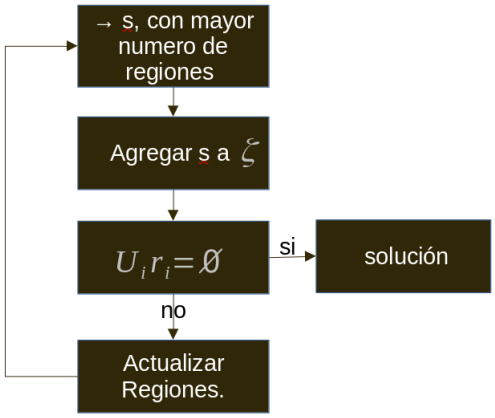

Figura 1: Algoritmo corregido

Lo primero que hay que señalar en la figura 1 es que ahora en primera instancia eligiremos al centro que cubra el mayor número de regiones, en segundo lugar agregaremos este centro a nuestro conjunto solución, tomaremos la decisión de avanzar o finalizar dependiendo si hemos cubierto todas las regiones, ya por último repetiremos esto hasta que terminemos de cubrir todas las regiones

SECCIÓN 4

Implementación

El algoritmo se programó en lenguaje python con referencias en Van Rossum und Drake Jr (1991), Van Rossum und Drake Jr (2017) y Chun (2001), y se puede verificar en el siguiente repositorio de: git-hub.

SECCIÓN 5

Resultados

Los resultados se pueden ver en el mismo enlace de git-hub, en los documentos txt, sin embargo se presentarán a continuación en una forma más ordenada:

Instancias de Prueba	*Nombre de la instancia*	m	n	*Óptimo*	Aproximado	% Error
4	scp41	200	1000	429	819	91 %
5	scp51	200	2000	253	1091	331 %
6	scp61	200	1000	138	450	226 %
A	scpa1	300	3000	1445	1445	0 %
B	scpb1	400	3000	69	926	1242 %
C	scpc1	400	4000	227	1798	692 %
D	scpd1	400	4000	60	1269	2015 %
E	scpe1	50	500	5	5	0 %
					promedio	575 %

Cuadro 1: Tabla con resltados obtenidos

SECCIÓN 6

Conclusiones

Si observamos los resultados obtenidos en el cuadro 1 podemos observar tanto los resultados obtenidos como su error obtenido en porcentaje, y si nos enfocamos al final de esta misma podemos observar que el promedio es de 575 % el cual es demasiado alto; ahora si observamos los resultados obtenidos en las instancias: spca1 y scpe1 podemos ver que se obtuvo un porcentaje del 0 % que es muy bueno, pero en las instancias scpd1 y scpb1 los porcentajes son demasiados altos en consecuencia esto contrarresta la eficacia obtenida en las primeras dos mencionadas.

Referencias

[Chun 2001] Chun, Wesley: *Core python programming*. Bd. 1. Prentice Hall Professional, 2001

[Van Rossum und Drake Jr 1991] Van Rossum, Guido ; Drake Jr, Fred L.: Guía de aprendizaje de Python. In: *Release* 2 (1991)

[Van Rossum und Drake Jr 2017] Van Rossum, Guido ; Drake Jr, Fred L.: *El tutorial de Python*. Python Software Foundation, 2017