|  |
| --- |
| Trabajo final Técnicas de Optimización Programación metaheurística Problema de la mochila con algoritmo ACO  Farelo Panesso Camilo Enrique, Henao Henao Katherin Johana, Lozano Durango Jacksson Universidad de Antioquia camilo.farelo@udea.edu.co, katherin.henao@udea.edu.co, jacksson.lozano@udea.edu.co  http://www.regalospublicitariosm.com/UNIVERSIDAD%20DE%20ANTIOQUIA.jpg   1. Detalle del logotipo de JITEL 2013. |

***Resumen***- *En el presente trabajo se simuló el problema de la mochila mediante el método de la colonia de hormigas; para esto se utilizó el lenguaje Python. Se han variado algunos parámetros con el fin de comparar los diferentes resultados con los obtenidos con el software GLPK*.

# INTRODUCCIÓN

A través del tiempo han existido problemas que se pueden solucionar por medio de métodos exactos debido a su facilidad, sin embargo, hay otros a los cuales no se les ha encontrado solución de manera exacta, o en el mejor de los casos su solución exacta compromete bastantes recursos computacionales. Para este último caso es necesario usar algoritmos metaheurísticos con los cuales se hayan soluciones aproximadas con tiempos de ejecución menores.

En este trabajo se usa modelamiento por colonia de hormigas, el cual es un método basado en la forma en como las hormigas trabajan, realizando recorridos de manera aleatoria hasta encontrar comida donde van dejando rastro de feromona en sus diferentes caminos. Con el paso del tiempo todas las hormigas convergen a una misma ruta debido a que en los caminos poco visitados se evapora la feromona mientras que en los visitados recientemente se refuerza.

Este algoritmo se empezó a implementar en los años 90’s para encontrar el camino óptimo en un grafo y debido a su eficiencia se ha masificado como método de resolución en diferentes problemas.

# OBJETIVOS

* Implementar el problema de la mochila en GLPK asignando como parámetros una capacidad de la mochila, el peso de cada bloque y su beneficio.
* Comparar los diferentes resultados obtenidos de la simulación en GLPK con los obtenidos en Python, variando parámetros como el número de hormigas, el número de bloques y coeficientes de impacto como α y β.

# IMPLEMENTACIÓN

Se da una solución exacta al problema con el software GLPK buscando que se maximice el beneficio empacado en la mochila, teniendo como única restricción la capacidad de la mochila, la cual sería 15 Kg. Los otros parámetros utilizados fueron el peso y el beneficio de cada bloque, generados previamente de manera aleatoria. Por último, se crea una variable binaria que define si el bloque está o no dentro de la mochila.

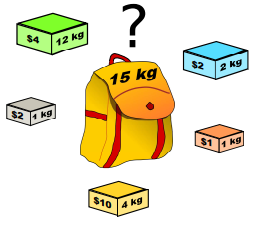


Ilustración . Problema de la mochila.

Ya teniendo listo el modelo, se realizó la ejecución del problema variando el número de bloques con 20, 50 y 100.

La función objetivo, la restricción, la variable y parámetro son las siguientes:

Función objetivo:

Sujeto a:

Variable:

variable binaria que define si el bloque i es seleccionado o no.

Parámetros:

número de ladrillos.

c: capacidad de la mochila

peso de cada bloque.

beneficio que tiene cada bloque

Posteriormente a esto, se creó el programa de la colonia de Hormigas en Python para solucionar el problema de la mochila. Se definió inicialmente parámetros de número de ladrillos, capacidad de la mochila, peso y beneficio de cada bloque utilizados para el GLPK. Ya en este caso hay dos parámetros nuevos que son propios del algoritmo de ACO los cuales son:

número de hormigas

la feromona que tiene cada bloque que inicialmente es 0.1.

la visibilidad de cada bloque

constante de influencia de la feromona  
 constante de influencia de la visibilidad.

Con estos parámetros se calcula la probabilidad que tiene cada bloque de ser seleccionado por una hormiga definida de la siguiente manera:

Para seleccionar cada bloque se genera un número aleatorio entre 0 y 1, que de acuerdo al intervalo en el que se encuentre se selecciona dicho bloque y se realiza este proceso hasta que la mochila se llene. El proceso anterior es realizado por cada una de las hormigas; la hormiga que haya seleccionado el camino con el mayor beneficio total corresponde a la solución de esta iteración. En los bloques escogidos por “la mejor hormiga” se refuerza la feromona y en los restantes se evapora de acuerdo a las siguientes ecuaciones:

Para nuestro modelamiento:

Este proceso anterior solo sería una iteración para el problema.

Para las siguientes iteraciones se realiza el proceso mencionado, pero ya las probabilidades son calculadas con la feromona actualizada. En nuestro caso solo quisimos utilizar máximo 5 iteraciones para optimizar el tiempo de ejecución y simulación y poder recopilar más pronto los resultados obtenidos.

Para comparar los resultados obtenidos en GLPK con respecto a los de Python, se varió los parámetros de número de hormigas, número de iteraciones, número de bloques, el valor de α y de β. Los resultados obtenidos fueron anotados y posteriormente se realizó en otro script .py los gráficos que se mostrarán en la sección de analisis.

# RESUTADOS

Los resultados obtenidos en GLPK son los siguientes:

|  |  |
| --- | --- |
| Número de bloques | Valor de la F.O. |
| 20 | 53 |
| 50 | 78 |
| 100 | 88 |

Tabla . Resultados obtenidos con GLPK.

Los resultados obtenidos en Python son los siguientes:

* **Variación de hormigas y bloques:**

Como en el GLPK se halló la solución para 20, 50, 100 bloques, entonces en Python se decidió utilizar 5, 10, 15 y 20 hormigas en cada una de las instancias. Alfa y Beta equivalen a 1.

|  |  |  |
| --- | --- | --- |
| Número de bloques | Número de hormigas | Beneficio total |
| 20 | 5 | 46 |
| 10 | 49 |
| 15 | 50 |
| 20 | 50 |

Tabla . Resultados con python para 20 bloques.

|  |  |  |
| --- | --- | --- |
| Número de bloques | Número de hormigas | Beneficio total |
| 50 | 5 | 54 |
| 10 | 70 |
| 15 | 60 |
| 20 | 69 |

Tabla . Resultados con python para 50 bloques

|  |  |  |
| --- | --- | --- |
| Número de bloques | Número de hormigas | Beneficio total |
| 100 | 5 | 66 |
| 10 | 67 |
| 15 | 68 |
| 20 | 71 |

Tabla . Resultados con python para 100 bloques.

* **Variación de número de iteraciones:**

Para este caso, se mantuvo el número fijo de hormigas como 20 y el de número de bloques para escoger fue de 100. A partir de ahí se realizaron variaciones en el número de iteraciones desde 1 hasta 5. Alfa y beta con un valor 1.

|  |  |  |
| --- | --- | --- |
| Número de hormigas | Número de iteraciones | Beneficio total |
| 100 | 1 | 50 |
| 2 | 58 |
| 3 | 63 |
| 4 | 65 |
| 5 | 66 |

Tabla . Resultados con python para la variación de iteraciones.

* **Variación de alfa:**

Para este caso se varía el impacto que tiene la feromona en el criterio de selección del bloque a través de la modificación del coeficiente alfa entre 0 y 4 dejando como constantes el valor de beta como 2, el número de bloques que es 100, 5 iteraciones y 20 hormigas.

|  |  |  |
| --- | --- | --- |
| Número de bloques | alfa | Beneficio total |
| 100 | 0 | 47 |
| 1 | 60 |
| 2 | 58 |
| 3 | 56 |

Tabla . Resultados con python para variación de alfa.

* **Variación de beta:**

Para este caso se varía el impacto que tiene la visibilidad en el criterio de selección del bloque a través de la modificación del coeficiente beta entre 0 y 4 dejando como constantes el valor de alfa que es 2, el número de bloques que es 100, 5 iteraciones y 20 hormigas.

|  |  |  |
| --- | --- | --- |
| Número de bloques | beta | Beneficio total |
| 100 | 0 | 63 |
| 1 | 65 |
| 2 | 62 |
| 3 | 61 |
| 4 | 55 |

Tabla . Resultados con python para variación de beta.

# ANÁLISIS DE RESULTADOS

**\*Variación de hormigas y bloques**

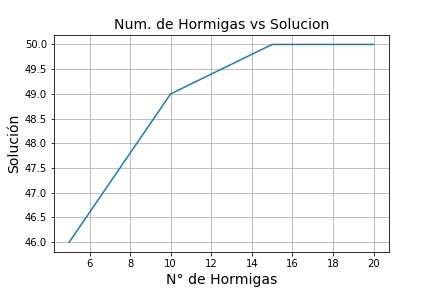


Ilustración . Curva de tendencia de la tabla 2.

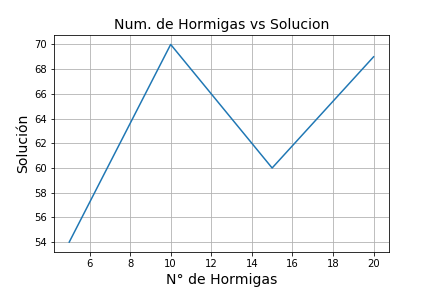


Ilustración . Curva de tendencia de la tabla 3.

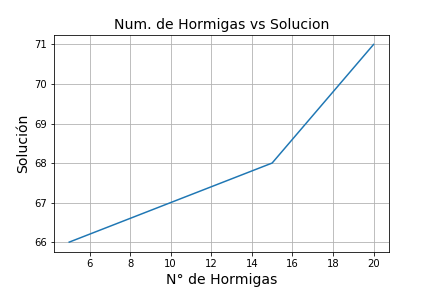


Ilustración . Curva de tendencia de la tabla 4.

De las ilustraciones 2,3 y 4 se observa como a medida que aumenta el numero de bloques aumenta el porcentaje de error, indicando que a mayor número de bloques el algoritmo se aleja más de la solución exacta.

De las ilustraciones 2,3 y 4 también se observa como el resultado va convergiendo a un valor único independientemente del número de hormigas que se tengan en la colonia.

**\*Variación de número de iteraciones:**

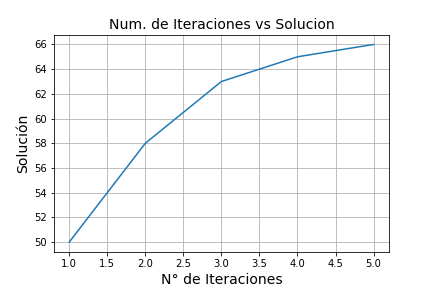
****

Ilustración . Curva de tendencia de la tabla 5.

De la gráfica anterior gráfica se puede ver que la relación del número de iteraciones con la solución tiene un comportamiento logarítmico debido a que de manera rápida converge a una única solución.

**\*Variación de alfa:**

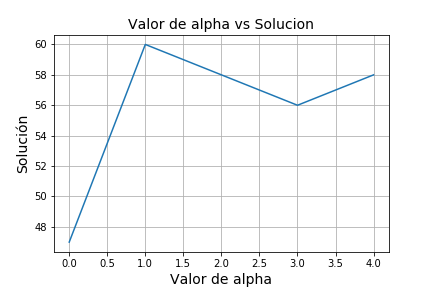
****

Ilustración . Curva de tendencia de la tabla 6.

A partir de cierto valor de α, el valor de la función objetivo tiende a converger a un rango de soluciones.

**\*Variación de beta:**

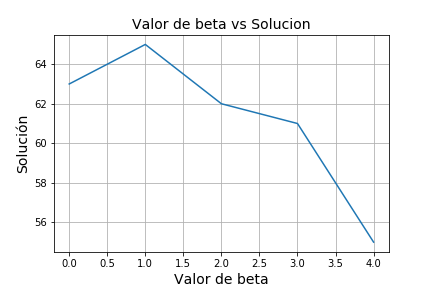
****

Ilustración . Curva de tendencia de la tabla 7.

De la ilustración 7 se analiza que darle más importancia a la visibilidad que a la feromona no es el mejor criterio porque el valor de la función objetivo va disminuyendo, cuando lo que se quiere es maximizar.

A modo de observación, se pude decir que aumentar el número de iteraciones realizadas por las hormigas o aumentar el número de hormigas produce un tiempo de ejecución elevado.