Centro de Investigación en Cómputo
Instituto Politécnico Nacional

Metaheurísticas
Actividad No. 8
Solución de problemas mediante Recocido Simulado
Curso impartido por: Dra Yenny Villuendas Rey

Adrian González Pardo

26 de octubre de 2020

1. Recocido Simulado (Simulated Annealing, SA)

Ventajas	Desventajas
Permite obtener una solución aproximada al mínimo global	Puede llegar a pasar que en la siguiente iteración a la solución salga del mínimo global y este se alcance un resultado mínimo local
Permite encontrar soluciones razonables (En tiempo)	La solución encontrada puede no ser la mejor
Utiliza poca memoria	No permite vuelta atras en más de 1 paso

2. Pseudocódigo SA

```

s<-Solucion_inicial
e_old<-funcion_fitness(s)
t<-temp_max
while t< temp_min
  # temp_min es aproximado a 0
  i=0
  while i<imax
    # imax puede ser representativo dependiendo del problema que se aborda
    #  o en todo caso puede ser el numero de iteraciones maxima
    seleccion_solucion_sucesor_de(s)
    # seleccion_solucion_sucesor_de es una funcion que explora la vecindad del arbol
    #    elige aleatoriamente
    e_new<-funcion_fitness(s)
    delta=e_new-e_old
    if delta>0
      if rand(0..1) >= exp(-delta/(K*temp))
        deshacer_sucesor(s)
      else
        e_old=e_new
      end
    else
      e_old=e_new
    end
    i+=1
  end
  t*=alpha
```

```
  # alpha es una constante que va al inicio del programa entre [0.88,0.99]
end
```

3. SA vs RMHC

La diferencia más notable entre estos dos algoritmos, es notablemente que el RMHC se basa en la mutación aleatoria de un conjunto de datos, generando un camino de solución del arbol de soluciones, mientras que SA parte de una solución donde el mismo algoritmo esta bioinspirado en un proceso físico que busca un ablandamiento para la formación o moldeo de estructuras de metal, entonces si bien ambos son algoritmos heurísticos RMHC se aplica en problemas cuya solución puede ser mutada y en este caso no puede salir de un mínimo local si este cae en ese espacio, mientras que SA permite admitir una respuesta no optima para seguir explorando el espacio de soluciones.

4. Algoritmo de búsqueda de vecinos adaptativa

En el paper podemos observar el algoritmo metaheurístico, en el cual su área de solución es en un espacio de dominio discreto, pero el autor nos hace mención con respecto a hacer una hibridación del problema usando un dominio continuo, en el cual se permite la idea de hacer un uso de algoritmo donde dependiendo la obtención del valor se decide por usar el dominio discreto o el dominio continuo para con este se pueda dar idea a la traslación y rotación del objeto que mencionan en el paper, para así no desperdiciar regiones de área que puden ser utilizadas para acomodar P_i objetos.

5. Investigación de los ultimos 3 años aplicando el algoritmo de SA

1. Agosto de 2020. Diseño del controlador de regulación automática de voltaje usando SA hibrido con algoritmos de optimización de Forrajeo de mantarraya. Paper
2. Enero de 2020. Una nueva técnica híbrida de programación genética basada en SA para predecir la capacidad de carga máxima de las pilas. Paper
3. Junio de 2018. Optimización de SA evolutivo multiobjetivo para el equilibrado de la línea de desmontaje multirrobótico de modelo mixto con tiempo de procesamiento de intervalo. Paper

6. Aplicaciones SA

6.1. Knapsack

Modelación Matemática
Sean dos funciones

$$f(x) = \sum_{i=1}^{N} x_i^{'} h(x_i)$$

$$g(x) = \sum_{i=1}^{N} x_i^{'} p(x_i) \ <= \ peso\_maximo$$

Donde:
$X\ es\ un\ vector\ de\ la\ forma\ X = [x_1, x_2, \cdots, x_N]$

$X^{'}\ es\ un\ vector\ el\ cual\ es\ parecido\ a\ X\ pero\ sus\ valores\ x_i^{'} \in [0,1]\ y\ x_i^{'} \in \mathbb{Z}$

$h(x)\ es\ una\ función\ la\ cual\ devuelve\ el\ beneficio\ total\ de\ los\ objetos\ x_i\ en\ la\ mochila$

$p(x)\ es\ una\ función\ la\ cual\ devuelve\ el\ peso\ total\ de\ los\ objetos\ x_i\ en\ la\ mochila$

En el cual el algoritmo de SA busca iterar sobre cada contenido o casilla del vector $X^{'}$ de tal forma que se obtiene una solución y sobre esta se busca encontrar una mejor solución.
Por lo cual el test objetivo del algoritmo es encontrar el optimo global sin que este pueda caer en el optimo local o en otra zona.

En el algoritmo se creara conjunto S cuya evaluación es una solución aleatoria al problema, en el cual al interactuar con el algoritmo SA se generara un conjunto $S^{'}$ el cual sera retornado como una mejor solución a S

Estructura propuesta para su solución:
Si bien en este problema podemos resolverso en un lenguaje de alto nivel podemos hacer uso de un arreglo dinámico en el cual podamos añadir elementos cuyos atributos son los siguientes:

- Costo: Valor entero o valor real
- Beneficio: Valor entero o valor real
- Selección: Valor booleano

De modo que a traves de este arreglo se pueda realizar la operación producto para la obtención del costo total y la obtención del peso total.

6.2. Travel Salesman Problem TSP

Modelación Matemática
Sea la función

$$f(x) = \sum_{i=1,j=1}^{N,N} x^{'}_{i,j} g(x_{i,j})$$

Donde:
$X\ es\ una\ matriz\ de\ NxN\ la\ cual\ trabajara\ para\ obtener\ el\ costo\ con\ la\ función\ g(x)$

$X^{'}\ es\ una\ matriz\ de\ NxN\ la\ cual\ contiene\ valores\ que\ permiten\ saber\ si\ el\ nodo\ es\ considerado\ o\ no\ para\ la\ solución$

$La\ notación\ (i,j)\ significa\ \ i\ como\ nodo\ origen\ que\ va\ a\ j$

$x^{'}_{i,j}\ es\ un\ valor\ de\ la\ matriz,\ donde\ x^{'}_{i,j} \in [0,1]\ y\ x_{i,j} \in \mathbb{Z}$

$Si\ x_{i,j} = -1\ significa\ que\ no\ hay\ una\ conexión\ de\ i\ a\ j$

$g(x)\ es\ una\ función\ la\ cual\ devuelve\ el\ valor\ costo\ de\ ir\ de\ i\ a\ j$

De tal forma que este algoritmo de SA busca iterar sobre el contenido de una solución cualquiera y sobre cada vertice que este modifique el algoritmo buscara realizar una operación de intercambio de la forma $x_{i,j}$ cambia con $x_{j,i}$ de tal forma que el movimiento sea valido y este modifique la solución.
El test objetivo de esta solucion es encontrar una solución de minimización de costo de tal forma que la solución recorra todos los nodos.
Finalmente el valor inicial del algoritmo es una matriz S cuyos valores realizaran el producto sobre su semenjante $S^{'}$ en el cual podran conocerse el costo inicial de la solución, al entrar al algoritmo de SA obtendremos una matriz que operara sobre $S^{'}$ y modificara los valores del camino tomado y de forma teorica se obtendra una matriz $T^{'}$ la cual al interactuar con el producto de S es mejor que $S^{'}$.

Estructura propuesta para la solución:
Para este problema podemos hacer uso de una estructura arreglo dinámico en el cual tendre los siguientes atributos arreglo de costos para cambiar al siguiente vertice, el arreglo de selección booleano que permitira conocer si el camino es tomado. Ahora bien ese arreglo dinámico dependiendo de la entrada de vertices podemos evolucionarlo a una estructura bicola circular en el caso de que sea un grafo completo es decir que todos sus vertices estan conectados entre si, de modo que al obtener una solución S este pueda rotar los indices formando una solución $S^{'}$.

6.3. Función de Minimización en D dimensiones

Modelación Matemática
Sea la función

$$f(x) = \sum_{i=1}^{N} x_i^2$$

Donde:
$x_i \in \mathbb{R}$

Descrito en los intervalos $x_i \in [-10, 10]$

Donde sabemos que los puntos mínimos de cada x_i los encontramos cuando el valor asignado a el es $x_i = 0$ por lo tanto el ir variando los valores para que el programa se acerque hacia 0
De tal manera que el algoritmo de SA seleccionara 1 indice en el cual se le asignara un valor a x_i.
En el test objetivo es encontrar 1 punto aproximado en el que la función se aproxime a 0 o en el que al menos una coordenada $x_i \sim 0$
En ello generaremos un conjunto solución S el cual contendra los puntos x_i de modo que al ser evaluado en la función matemática obtendremos un valor Y que sera el costo total de evaluar ese punto y considerarlo punto mínimo, una vez pasando por el algorimto SA obtendremos un valor $S^{'}$ en el cual al pasar por la función a evaluar se obtendra un $Y^{'}$ de modo que teóricamente al comparar dichos valores obtenidos $Y^{'}$ es mejor y por lo tanto $S^{'}$ es un mejor punto mínimo de la función.

Estructura propuesta para la solución:
Para este problema podemos pensar en un vector o arreglo dinámico de D dimensiones en el cual evalualeremos por la función matemática que nos arrojara la suma de los cuadrados de cada punto x_i de modo que podremos trabajar sobre la misma para obtener un mínimo mejor evaluado que otro.

7. Implementación y ejecución

7.1. Knapsack

La interfaz por línea de comandos es la siguiente para el programa:

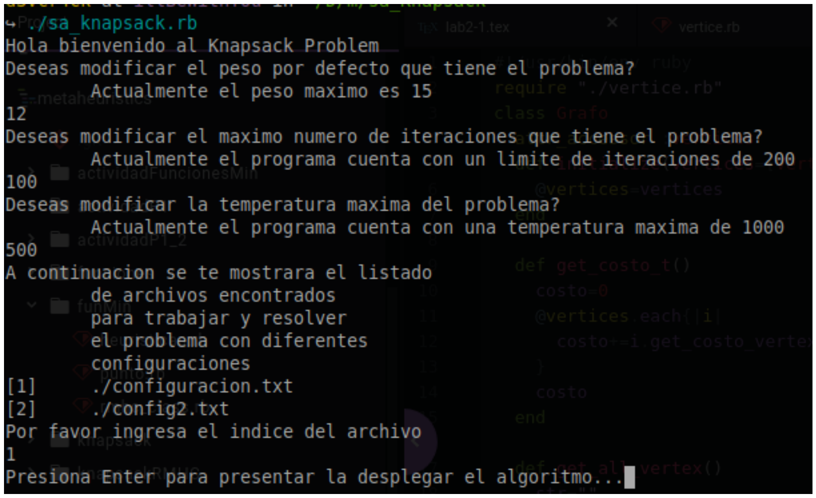
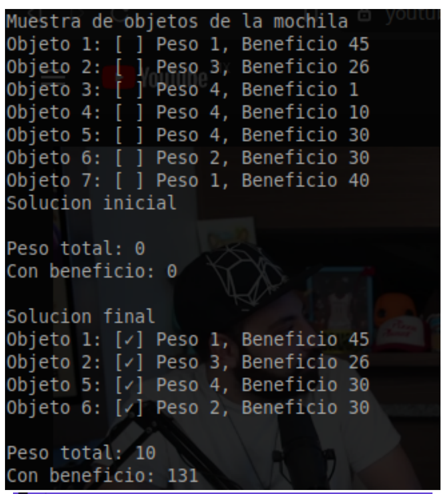
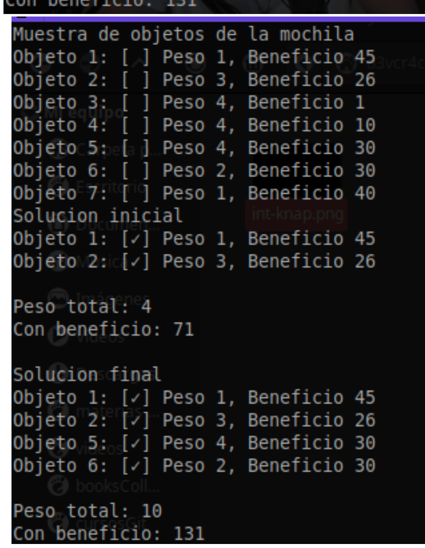


Figura 1: Interfaz del problema Knapsack en el cual se puede modificar máximo de iteraciones, temperatura máxima, peso máximo de la mochila, y el archivo de configuraciones

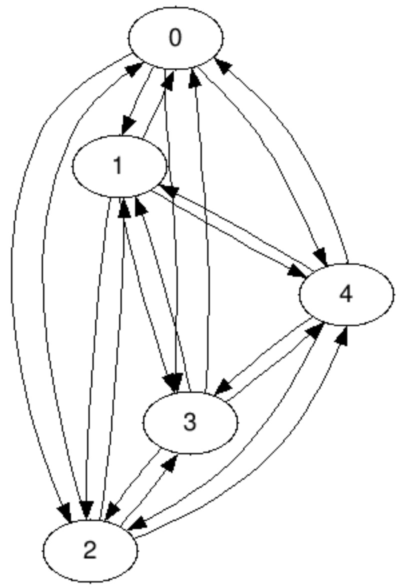
La ejecución del problema arrojo los siguientes resultados de dos corridas:



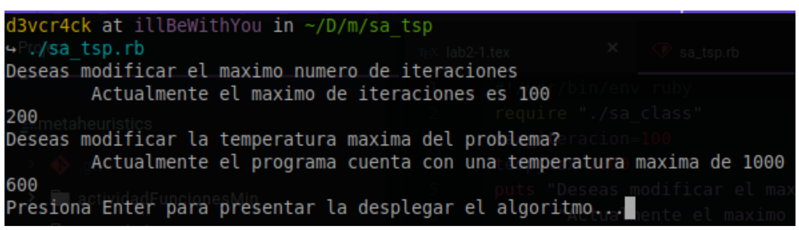


7.2. TSP

La configuración del grafo es:



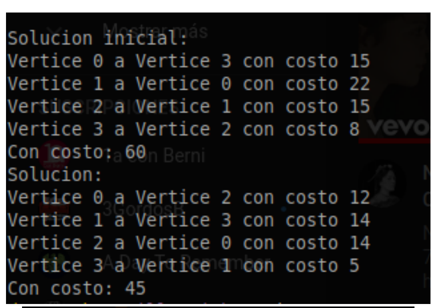
La interfaz por línea de comandos es la siguiente:



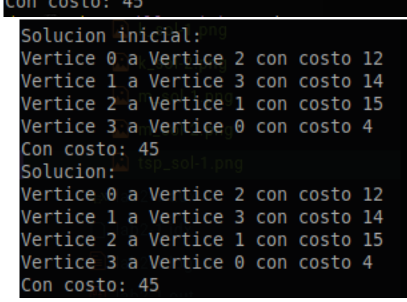
```
d3vcr4ck at illBeWithYou in ~/D/m/sa_tsp
↪ ./sa_tsp.rb
Deseas modificar el maximo numero de iteraciones
        Actualmente el maximo de iteraciones es 100
200
Deseas modificar la temperatura maxima del problema?
        Actualmente el programa cuenta con una temperatura maxima de 1000
600
Presiona Enter para presentar la desplegar el algoritmo...
```

Figura 2: Interfaz del TSP en cual se puede modificar el máximo de iteraciones, y temperatura máxima, destacando que la implementación se busca la solución a una entrada de un grafo completo, es decir 1 vertice apunta a los demas de igual forma.

La ejecución arrojo los siguientes resultados de la configuración:



```
Solucion inicial:
Vertice 0 a Vertice 3 con costo 15
Vertice 1 a Vertice 0 con costo 22
Vertice 2 a Vertice 1 con costo 15
Vertice 3 a Vertice 2 con costo 8
Con costo: 60
Solucion:
Vertice 0 a Vertice 2 con costo 12
Vertice 1 a Vertice 3 con costo 14
Vertice 2 a Vertice 0 con costo 14
Vertice 3 a Vertice 1 con costo 5
Con costo: 45
```



```
Solucion inicial:
Vertice 0 a Vertice 2 con costo 12
Vertice 1 a Vertice 3 con costo 14
Vertice 2 a Vertice 1 con costo 15
Vertice 3 a Vertice 0 con costo 4
Con costo: 45
Solucion:
Vertice 0 a Vertice 2 con costo 12
Vertice 1 a Vertice 3 con costo 14
Vertice 2 a Vertice 1 con costo 15
Vertice 3 a Vertice 0 con costo 4
Con costo: 45
```

7.3. Función mínimos

La interfaz por línea de comandos es la siguiente:

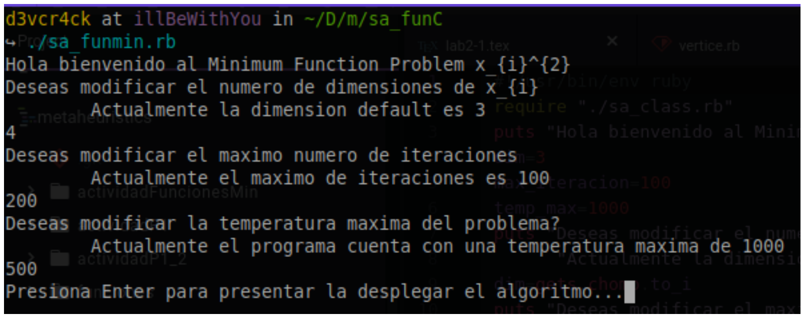


Figura 3: Interfaz del problema de la función $f(x) = \sum_{i=1}^{D} x_i^2$ en el cual se puede modificar máximo de iteraciones, temperatura máxima y número de dimensiones de la función

La ejecución arrojo los siguientes resultados: