


Metaheurísticas
4º Curso de Grado
en Ingeniería Informática
Especialidad en Computación

Área de Ciencias de la
Computación e Inteligencia Artificial
Departamento de
Tecnologías de la Información

PRÁCTICA 2 (Versión 2021, 1.0)

Algoritmos de Búsqueda Multiarranque: GRASP, GRASP Extendido, ILS y VNS

Objetivos

El objetivo de esta práctica es estudiar el funcionamiento de los *Algoritmos de Búsqueda Multiarranque: Procedimiento de Búsqueda Voraz Aleatorio Adaptativo (GRASP), Búsqueda Local Reiterada (ILS) y Búsqueda Basada en Entornos Cambiantes (VNS)*. Para ello, se requerirá que el alumno implemente estos algoritmos, para resolver el *Problema del Viajante de Comercio (TSO)*. El comportamiento de los algoritmos implementados deberá compararse entre sí y, al menos, también con las técnicas *Greedy* y Búsqueda Local del Mejor Vecino de la Práctica 1. El análisis de los resultados influirá decisivamente en la calificación de la práctica.

En nuestro caso trabajaremos con 3 instancias del problema obtenidas de la biblioteca TSPLIB, todas ellas correspondientes al TSP simétrico (misma distancia o coste independientemente de si es en un sentido u otro). Serán las siguientes:

- St70: Tamaño 70 ciudades. Coste de la solución óptima: 675
- Ch130: Tamaño 130 ciudades. Coste de la solución óptima: 6.110
- A280: Tamaño 280 ciudades. Coste de la solución óptima: 2.579

Enunciado de la práctica

Es importante destacar que, puesto que los algoritmos considerados en esta práctica se basan en generar múltiples soluciones, refinarlas mediante un algoritmo de búsqueda local, y devolver la mejor solución encontrada, no será necesario ejecutarlos varias veces

con distintos valores de la semilla para el generador aleatorio, tal como se hacía con los algoritmos probabilísticos experimentados en la práctica anterior.

Dado que en la Práctica 1, igualmente se utilizó el problema de TSP, asuma los mismos elementos que no se especifiquen de forma diferente en este enunciado para cada algoritmo.

- *Algoritmo de Búsqueda Local*: Utilice la Búsqueda Local del Mejor Vecino con el intercambio como operador de generación de vecinos (algoritmo 2-opt) propuesto asimismo en la práctica anterior.

- *Función objetivo*: Como en la Práctica 1.

  No deje de emplear la variante optimizada de la función objetivo: Sea *S'* el vecino generado a partir de *S* mediante el intercambio con el operador 2-opt de los valores contenidos en las posiciones *r* y *s*.

GRASP

Algoritmo

El algoritmo GRASP constará de dos componentes: construcción de soluciones *Greedy* probabilísticas, y optimización de las mismas mediante el algoritmo de búsqueda local. El algoritmo *Greedy* probabilístico es el siguiente:

- *Construcción de soluciones Greedy probabilísticas*: El mecanismo GRASP de generación se basa inicialmente en seleccionar las ciudades con. Para aplicarlo se mantendrán dos listas, una que almacena mayor potencial. Los elementos seleccionados serán eliminados de la lista, reemplazándose por los siguientes elementos inmediatamente siguientes y aún no incluidos. De esta forma se mantiene constantes las lista mientras haya suficientes elementos.

Una vez generada cada solución *Greedy* probabilística inicial se aplicará el algoritmo de Búsqueda Local como de costumbre.

Valores de los Parámetros

Se generarán 10 soluciones *Greedy* probabilísticas. Estas soluciones serán posteriormente optimizadas mediante una única ejecución del algoritmo de Búsqueda Local para cada una de ellas. El parámetro $l = 0.1 \cdot n$.

Ejecuciones

Se realizará un total de tres ejecuciones, correspondientes a ejecutar el algoritmo GRASP a los tres casos (*data sets*) del problema.

ILS

Algoritmo

El algoritmo ILS consistirá en generar una solución aleatoria inicial y aplicar el algoritmo de Búsqueda Local. Una vez obtenida la solución optimizada, se estudiará si es mejor que la mejor solución encontrada hasta el momento y se realizará una mutación sobre la mejor de las dos, volviendo a aplicar el algoritmo de búsqueda local sobre esta solución mutada. Este proceso se repite un determinado número de veces, devolviéndose la mejor solución encontrada en todo el proceso. Por tanto, se sigue el *Criterio del Mejor* como criterio de aceptación.

El *operador de mutación* se implementará de la siguiente forma: Haremos uso de un operador que provoque un cambio más brusco en la solución actual que el considerado en la Búsqueda Local. Para ello concretamente, usaremos el operador de modificación por *Sublista Aleatoria de Tamaño Fijo*: este proceso consiste en generar aleatoriamente dos posiciones que determinen una sublista de tamaño *s*, analizar las asignaciones efectuadas entre ambas y reasignarlas aleatoriamente. No se considerarán sublistas cíclicas y, por tanto, la posición inicial de la sublista siempre será inferior a la posición final.

Valores de los Parámetros

Se aplicará 50 veces el algoritmo de Búsqueda Local, la primera vez sobre una solución aleatoria y las 49 restantes sobre soluciones mutadas. Se usará un tamaño de sublista s = n/4 en la mutación.

Sólo se ejecutará el algoritmo una vez para cada uno de los tres problemas.

VNS

Algoritmo

El algoritmo VNS que consideraremos se compone de los siguientes pasos:

1. Generar la solución actual aleatoria S_{act} y hacer $k = 1$, $bl = 0$
2. Si ($k > k_{max}$) hacer $k = 1$.
3. Generar una solución vecina (S_{vec}) de S_{act} con el operador de generación de vecino para el valor del parámetro k, $S_{vec} = N_k(S_{act})$.
4. Aplicar la Búsqueda Local sobre la solución S_{vec}, obteniendo S'. Hacer $bl = bl + 1$.
5. Si S' es mejor que S_{act}, hacer $S_{act} = S'$ y $k = 1$. Si no, hacer $k = k + 1$.
6. Si ($bl < bl_{max}$) volver a 2. Si no, devolver S_{act} y terminar.

El operador de generación de vecino hará uso de nuevo de la *Sublista Aleatoria de Tamaño Fijo* anteriormente descrita para el algoritmo ILS (aunque en aquel caso, se usó para implementar el operador de mutación).

Valores de los Parámetros

El algoritmo de búsqueda local se aplicará cincuenta veces (bl_{max} = 50). Se trabajará con k_{max} = 5, es decir, cinco entornos diferentes. El valor de *s* (tamaño de la sublista) en el operador de generación de vecino definirá el tamaño del movimiento según el valor de *k*. Irá aumentándose de la siguiente forma:

- *k* = 1: Se aplica un tamaño de *s* = n/8.
- *k* = 2: Se aplica un tamaño de *s* = n/7.
- *k* = 3: Se aplica un tamaño de *s* = n/6.
- *k* = 4: Se aplica un tamaño de *s* = n/5.
- *k* = 5: Se aplica un tamaño de *s* = n/4.

En resumen, para *k* se aplica el operador de vecino sublista aleatoria con $s = n=(9-k)$.

Los resultados que deben aportarse

El comportamiento de los algoritmos implementados deberá compararse entre sí y con las técnicas *Greedy* y Búsqueda Local del mejor vecino de la Práctica 1.

Fecha y Método de Entrega;

El día 5 de Mayo durante la sesión de prácticas: Código fuente y documentación (con análisis), con una extensión máxima de 3 folios. Tenga en cuenta que el análisis que realice, cuenta más que las propias implementaciones.