Tarea 4

Universidad de Guanajuato
Eduardo Calvo Martínez

Optimización estocástica
29 de marzo de 2025

Para esta tarea se implementó un método de búsqueda guiada, el método que implementé fue búsqueda local iterada, a continuación se desscriben las pertubaciones usadas en el algoritmo y todo lo usado en el mismo.

Primer Perturbación

La primera perturbación fue extraíada de äsd; el proceso es extraer `r` nodos generados aleatoriamente, el primer paso es extraer tomar el nodo `v` tal que

$$\arg\min_{v \in T} c(v, v_1) - c(v, w),$$

donde el nodo v_1 es el nodo consecutivo a v_1 en el tour y w es el nodo más cercano a v. Es decir que estamos buscando el nodo en el tour tal que minimiza el costo de agregar (v, w) y eliminar (v, v_1). Una vez v está seleccionado tomamos al azar entre los aristas (w_{2i-1}, w_{2i}) para $i = 1, 2, 3$ donde los w_{2i-1} son los 3 más cercanos a v seleccionados entre k nodos generados aleatoriamente, todo lo anterior está reflejado en el siguiente pseudocódigo

Algorithm 1 Primer Perturbación

Entrada: Tour T, número de nodos a extraer r, parámetro k
Salida: Tour perturbado T'

Seleccionar aleatoriamente r nodos del tour T
for cada nodo v seleccionado **do**
    Sea v_1 el nodo consecutivo a v en el tour
    Sea w el nodo más cercano a v que no está en T
    Calcular $c(v, v_1) - c(v, w)$
    Seleccionar v si minimiza
end for
Escoger aleatoriamente aristas (w_{2i-1}, w_{2i}) para $i = 1, 2, 3$
Donde w_{2i-1} son los 3 nodos más cercanos a v entre los k de la muestra aleatoria
Construir el nuevo tour T' borrando las aristas seleccionadas de T
return T'

Para nuestras pruebas fijamos $k = r$ y usamos $r = \alpha N$ con N el número de ciudades en el tour y α con valores 0.023 y 0.03.

Segunda Perturbación

Para esta perturbación lo que hicimos fue seleccionar 3 índices aleatoriamente `i_0, i_1, i_2` y con esto dividir el tour en 4 partes. El segmento `tour[0, i_0]`, `tour[i_0 + 1, i_1]`, `tour[i_1 + 1, i_2]` y `tour[i_2 + 1, N-1]` y generar un nuevo tour pero con estos segmentos permutados de la siguiente manera, permutar `tour[0, i_0]` con `tour[i_0 + 1, i_1]` y `tour[i_1 + 1, i_2]` con `tour[i_2 + 1, N-1]`. Esto nos garantiza un cambio al menos $3 - opt$ y a lo más $4 - opt$ haciendo que nos salgamos de la vecindad que estamos explorando que es la $2 - opt$.

Búsqueda local iterativa

El algoritmo se implementó tal y como fue visto en clase, el historial de perturbación únicamente fue un set de vectores de perturbaciones tal que si la perturbación ya se había explorado, entonces se volvía a perturbar y continuaba la búsqueda local iterativa. El pseudocódigo de nuestra ILS es el siguiente

Algorithm 2 Búsqueda Local Iterativa

Entrada: Solución actual act_ans, costo $cost$, matriz de distancias $dstMat$, nodos cercanos $nearNodes$, parámetro α, índice de hilo $threadIdx$, tiempo límite T
Salida: Solución optimizada act_ans con costo $cost$
$N \leftarrow$ tamaño de act_ans
Inicializar conjuntos $pert\_history$
Iniciar temporizador $start$
Aplicar búsqueda local sobre act_ans y $cost$
while $CurrentExecutionTime(start) \leq T$ **do**
    $pertCost \leftarrow cost$
    Inicializar $perturbation$ como vector de tamaño N
    $perturbation \leftarrow pert(\alpha, act\_ans, dstMat, nearNodes, cost, pertCost)$
    if $perturbation \notin pert\_history$ **then**
        Agregar $perturbation$ a $pert\_history$
    else
        $perturbation \leftarrow pert(\alpha, act\_ans, dstMat, nearNodes, cost, pertCost)$
        Agregar $perturbation$ a $pert\_history$
    end if
    Aplicar búsqueda local sobre $perturbation$ y $pertCost$
    if $pertCost < cost$ **then**
        $act\_ans \leftarrow perturbation$
        $cost \leftarrow pertCost$
    end if
end while
return $act\_ans, cost$

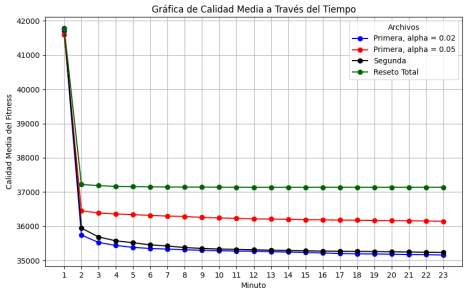
En la siguiente página se encuentran los resultados de las ejecuciones del código

Resultados

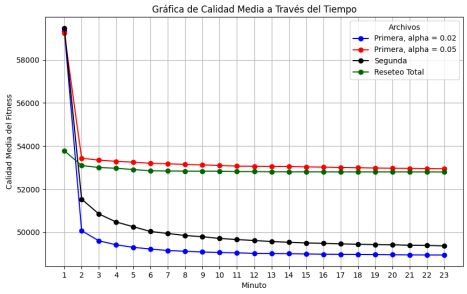
A continuación la tabla de estadísticos, los plots de medias y los boxplots por perturbación

Estadístico	**Instancia d493**	**Instancia d657**	**Instancia dsj1000**
Mejores soluciones halladas			
N/A	35002	48912	18660188
Primera perturbación $\alpha = 0.02$			
Mínimo	34927	48716	18851503
Máximo	35406	49169	19063675
Promedio	35155.8	48943.53	18953666.2
Primera perturbación $\alpha = 0.05$			
Mínimo	35861	51819	19977883
Máximo	36546	53261	20494738
Promedio	36143.66	54514.51	20296778.63
Segunda perturbación			
Mínimo	35051	49014	19091957
Máximo	35428	49751	19425252
Promedio	35235.66	49355.73	19280190.46
Segunda perturbación			
Mínimo	35051	49014	19091957
Máximo	35428	49751	19425252
Promedio	35235.66	49355.73	19280190.46

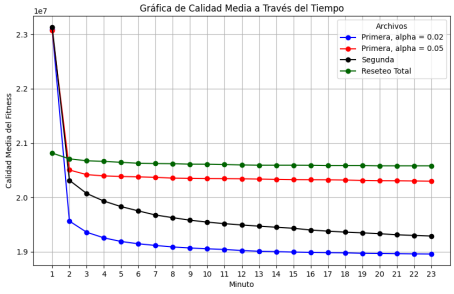
Cuadro 1: Comparación de fitness



(a) d493

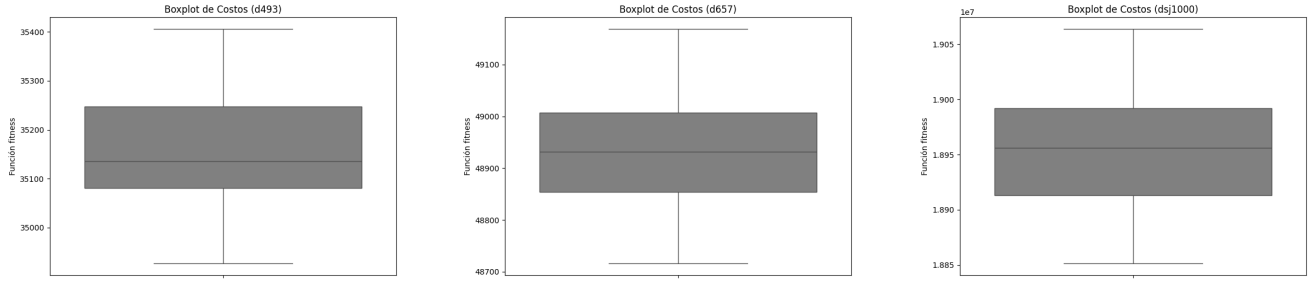


(b) d4657



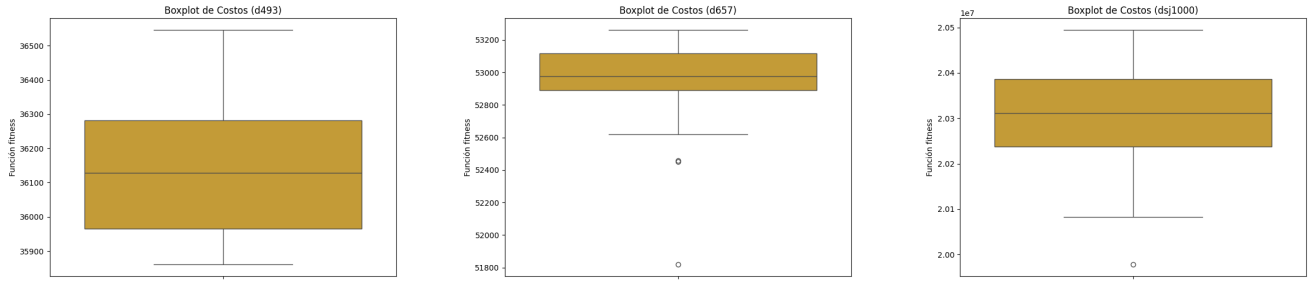
(c) dsj1000

Figura 1: Gráficas de calidad media.



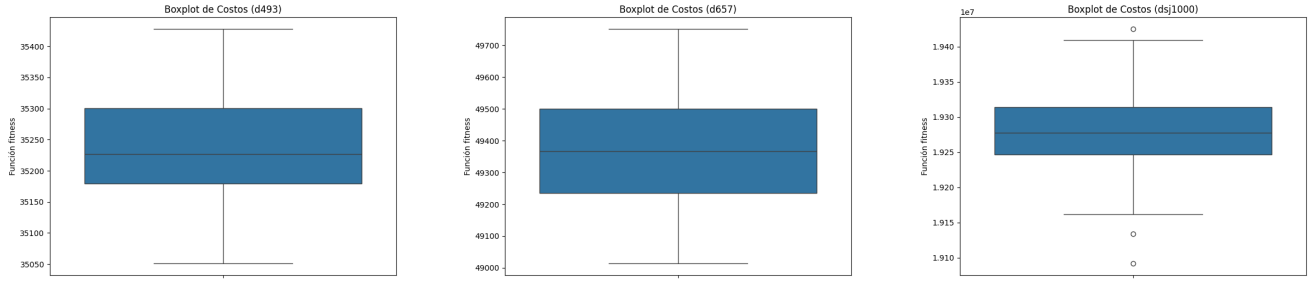
(a) d493 (b) d4657 (c) dsj1000

Figura 2: BoxPlots de la primera perturbación $\alpha = 0.02$.



(a) d493 (b) d4657 (c) dsj1000

Figura 3: BoxPlots de la primera perturbación $\alpha = 0.05$.



(a) d493 (b) d4657 (c) dsj1000

Figura 4: BoxPlots de la segunda perturbación.

Conclusiones

Como podemos notar la primer perturbación es la que mejores resultados da cuando se tiene un parámetro $\alpha = 0.02$, en el caso del reseteo completo no se encuentra muchas mejoras. Y las otras dos perturbaciones se quedan algo lejos de los resultados de esta. Podemos notar que la búsqueda local iterada presenta una

mejora muy importante en este problema y en el caso de las dos primeras instancias se logra incluso mejorar lo mejor hallado y en la tercera nos acercamos bastante.