Tarea 3-Optimización Estocástica

1st Daniel Vallejo Aldana
Maestría en Ciencias de la Computación
Centro de Investigación en Matemáticas
daniel.vallejo@cimat.mx

***Resumen*—En el presente trabajo se propone una forma de crear vecindades usando programación dinámica (DP) sobre el algoritmo de búsqueda local de forma que contengan mejores soluciones candidatas que solamente utilizando el algoritmo de Stochastic Hill Climbing descrito en la tarea anterior. En este trabajo realizamos una comparación en cuanto a valores de la solución obtenidos y en cuanto a tiempo de ejecución de los dos algoritmos.**

***Index Terms*—p-medianas, grafos, búsqueda local, programación dinámica.**

I. Introducción

De acuerdo a [1], el problema de las p-medianas consiste en localizar p-instalaciones (medianas) dentro de una red (grafo) con $|V|$ nodos y $|E|$ aristas de tal forma que minimicemos la suma de todas las distancias de cada vértice v del grafo $G(V, E)$ a su instalación más cercana. Varios métodos se han utilizado para la resolución de este problema, en [1] se propone un método basado en relajación Lagrangiana, en [2] se analiza resolver este problema con algoritmos de métodos de partición como lo son la Variable Neighbor Search (VNS) o la búsqueda tabú. Así mismo el trabajo de Loranca et.al. [2], es usado como comparación de los resultados obtenidos en el presente trabajo.

En el presente trabajo se propone e implementa una mejora al algoritmo de Stochastic Hill Climbing basada en programación dinámica en donde se busca resolver subproblemas de tamaño $p = 2$ y resolver dichos subproblemas hasta resolver el problema completo para $p = k$. Dicha estrategia mostró una mejora en cuanto a valor de la solución, no obstante, el tiempo de ejecución creció considerablemente también como se verá en la sección de resultados.

II. Metodología

II-A. Estrategia de Programación Dinámica (DP)

Para la estrategia de programación dinámica (DP) de nuestro algoritmo consideremos el descriptor previamente definido en la Tarea 2 que tiene la siguiente estructura.

```
struct {
int u;
int nu;
} descriptor;
```

En el caso de la estrategia propuesta consideramos una nueva estructura definida como `Dpair` correspondiente a una pareja de descriptores que serán los dos nodos dentro del arreglo de p candidatos que optimizaremos en ese momento. Para determinar la pareja de nodos que otpimizaremos en un determinado momento, creamos una arreglo de números a de 0 a p el cual es reordenado de forma aleatoria. Posteriormente formamos una cola con las parejas $(a[i], a[i + 1]| - 1)$ y esto nos da las parejas de números que debemos optimizar en determinado momento.

Posteriormente aplicamos el algoritmo de búsqueda local sobre dichas posiciones del vector solución de forma parecida a la descrita en la Tarea 2. Una vez que se complete el proceso de optimización por búsqueda local sobre las posiciones definidas en donde ningún vecino mejore el costo de la solución actual entonces procedemos a optimizar la siguiente pareja de índices hasta que se encuentre en la cola creada. El algoritmo de optimización termina hasta que la cola que contiene las parejas de números a optimizar quede vacía.

III. Resultados

III-A. Instancia 1

Probamos los algoritmos diseñados en la presente tarea para la primera instancia la cual consta de 100 nodos, 200 aristas y cuyo valor de $p = 5$. El boxplot obtenido para cada uno de los métodos de encontrar las mejores soluciones candidatas se muestra a continuación en la figura 1

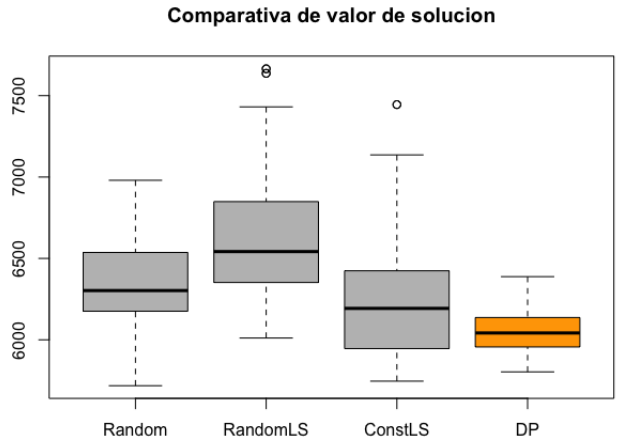

Figura 1. En naranja se muestra el boxplot del costo de las mejores soluciones encontradas con la estrategia de programación dinámica

Con base en la figura 1 podemos observar que la nueva heurística propuesta mejora las soluciones que se encuentran para el problema ya que se observa que la mediana de las observaciones es menor que en el resto de las heurísticas hasta el momento impelementadas.

La tabla de comparación se muestra a continuación

Método	Media	Min	Max
Aleatorio	6356	**5718**	6980
Aleatorio + búsqueda local	6613	6011	7665
Heurística constructiva + búsqueda local	6249	5746	7445
Aleatorio + DP en búsqueda local	**6055**	5803	**6388**
Mejor conocido		5891	

Cuadro I
TABLA COMPARATIVA DE RESULTADOS OBTENIDOS

En este caso vemos que en general la propuesta del uso de programación dinámica es en promedio mejor a las heurísticas que han sido previamente implementadas, así mismo vemos que la variación entre el valor de las mejores soluciones es menor a las demás heurísticas evaluadas lo que incrementa la generalidad del método.

Al comparar la tabla de tiempos de ejecución en cuanto a tiempo de búsqueda local en segundos podemos notamos lo siguiente.

Método	Media	Min	Max
Aleatorio + búsqueda local	0.04506	0.01757	0.11246
Heurística constructiva + búsqueda local	0.03492	0.01773	0.08106
Aleatorio + DP en búsqueda local	0.2486	0.1733	0.3612

Cuadro II
TABLA COMPARATIVA DE TIEMPOS DE BÚSQUEDA

Los boxplot correspondientes a los tiempos de búsqueda se muestran a continuación.

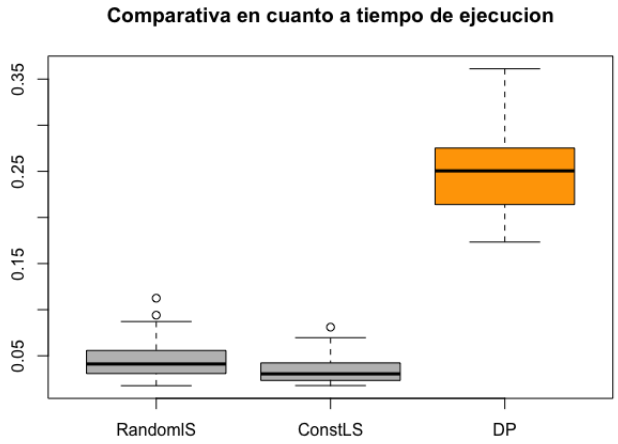

Figura 2. Comparación de tiempos que dura la búsqueda local en la instancia

En cuanto a tiempos de ejecución vemos que la nueva propuesta de heurística de solución es más lenta que las otras heurísticas evaluadas, en el caso de la primera instancia, lo anterior parecería una desventaja respecto a usar solamete búsqueda local por escalada. No obstante, como veremos en las otras instancias el rendimiento en cuanto a valor de la mejor solución es mejor en instancias más grandes.

III-A1. Instancia 2: La instancia 2 descrita en [1] corresponde a una colección de 200 nodos, 800 aristas y $p = 40$. Los resultados obtenidos por el algoritmo en cuanto a valor de la función objetivo se muestran a continuación.

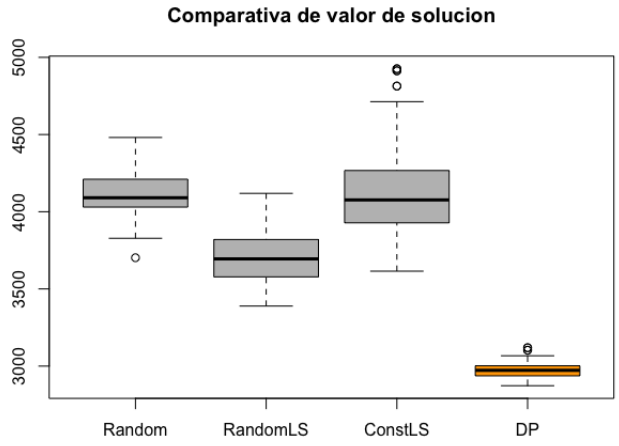

Figura 3. En naranja se muestra el boxplot del costo de las mejores soluciones encontradas con la estrategia de programación dinámica

La tabla comparativa en cuanto a valor de la función objetivo se muestra en el cuadro III

Método	Media	Min	Max
Aleatorio	4090	3702	4481
Aleatorio + búsqueda local	3709	3389	4119
Heurística constructiva + búsqueda local	4126	3615	4926
Aleatorio + DP en búsqueda local	**2972**	**2873**	**3119**
Mejor conocido		2837	

Cuadro III
TABLA COMPARATIVA DE RESULTADOS OBTENIDOS

De la tabla III podemos observar la gran mejora de la estrategia de programación dinámica respecto a las demás heurísticas evaluadas, con esta nueva estrategia casi es alcanzable el mejor valor conocido para la solución del problema. Así mismo en promedio dicha heurística se comporta mejor que las anteriormente evaluadas en cuanto a valor de la solución objetivo.

Al comparar la tabla de tiempos de ejecución en cuanto a tiempo de búsqueda local en segundos podemos notamos lo siguiente.

Método	Media	Min	Max
Aleatorio + búsqueda local	0.4124	0.1452	1.9832
Heurística constructiva + búsqueda local	0.8740	0.1663	2.3605
Aleatorio + DP en búsqueda local	18.44	15.79	24.93

Cuadro IV
TABLA COMPARATIVA DE TIEMPOS DE BÚSQUEDA

Los boxplot correspondientes a los tiempos de búsqueda se muestran a continuación.

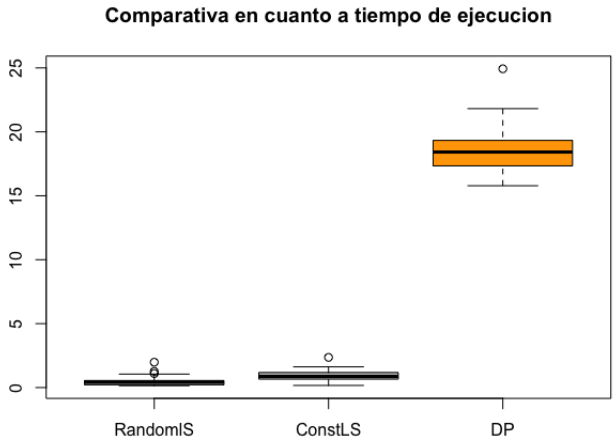

Figura 4. Comparación de tiempos que dura la búsqueda local en la instancia

Nuevamente vemos que los tiempos de ejecución aumentan de forma considerable respecto a las demás heurísticas en esta instancia. Veremos que en instancias más grandes el tiempo de ejecución puede llegar a casi una hora.

III-A2. Instancia 3: La instancia 3 es la instancia más grande evaluada en el presente problema, consta de 800 nodos con 12800 aristas y valor de $p = 80$. Los resultados de la función objetivo se muestran a continuación.

Por limitaciones en los tiempos de ejecución solo se consiguió llevar a cabo un experimento con 10 ejecuciones por lo que dichas ejecuciones serán usadas para la generación de los boxplots. No obstante no es una comparación justa respecto a las 50 ejecuciones de las demás heurísticas.

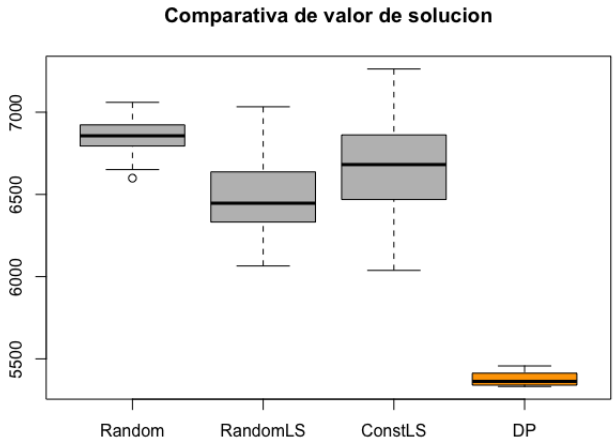

Figura 5. Boxplot de las soluciones encontradas con programación dinámica, nuevamente se aclara que no es una comparación del todo justa debido a las pocas observaciones con las que se cuenta debido al elevado tiempo de ejecución del algoritmo

La tabla comparativa en cuanto a valor de la función objetivo se muestra en el cuadro III

Método	**Media**	**Min**	**Max**
Aleatorio	6856	6599	7060
Aleatorio + búsqueda local	6446	6065	7033
Heurística constructiva + búsqueda local	6682	6038	7263
Aleatorio + DP en búsqueda local	**5377**	**5332**	**5457**
Mejor conocido		5057	

Cuadro V
TABLA COMPARATIVA DE RESULTADOS OBTENIDOS

En este caso y aunque la comparación se llevó a cabo con un menor número de soluciones obtenidas al solicitado podemos ver que la media de dichas soluciones es considerablemente menor a la media de las soluciones generadas con las heurísticas de las tareas pasadas. Lo anterior nos sugiere que el método propuesto con programación dinámica si mejora de forma considerable la función de costo a minimizar respecto a las heurísticas propuestas.

Al comparar la tabla de tiempos de ejecución en cuanto a tiempo de búsqueda local en segundos podemos notamos lo siguiente.

Método	**Media**	**Min**	**Max**
Aleatorio + búsqueda local	31.67	8.26	75.70
Heurística constructiva + búsqueda local	46.60	8.54	111.53
Aleatorio + DP en búsqueda local	≥ 3600	≥ 3600	≥ 3600

Cuadro VI
TABLA COMPARATIVA DE TIEMPOS DE BÚSQUEDA

Podemos ver que por cada solución el algoritmo necesita alrededor de una hora por lo que se propone agregar una restricción de tiempo al algoritmo para que encuentre si bien una solución no tan buena como sin restricciones de tiempo, si una solución que pueda ser obtenida en un tiempo razonable.

IV. CONCLUSIÓN

Con base en los experimentos realizados en el presente trabajo podemos concluir que el método de búsqueda local combinado con programación dinámica DP mejoró mucho los resultados comparándolo con las heurísticas evaluadas en la Tarea 2. No obstante, dicha mejora viene acompañada de un aumento en el tiempo computacional requerido para obtener una solución. En futuras tareas se buscará hacer más eficiente la obtención de soluciones sin sacrificar el rendimiento obtenido en la presente tarea.

REFERENCIAS

[1] John E Beasley. A note on solving large p-median problems. *European Journal of Operational Research*, 21(2):270–273, 1985.

[2] María Beatríz Bernábe Loranca, Jorge A Ruiz-Vanoye, Rogelio González Velázquez, Marco Antonio Rodríguez Flores, and Martín Estrada Analco. P-median: A performance analysis. *Res. Comput. Sci.*, 88:19–30, 2014.