1. Metaheurística Propuesta

Se propone un Iterated Local Search (ILS) que se presenta en el **Algorithm 1** que contiene 4 componentes: la solución inicial (línea 1); una estrategia de perturbación (línea 5); dos estrategias de búsqueda local (línea 2 y en las líneas 6–7) y finalmente un criterio de aceptación (línea 8-9). El algoritmo contempla una solución actual y la mejor solución encontrada, s y s^*, respectivamente. En la siguientes subsecciones se presentan con más detalles la representación, función objetivo, solución inicial, estrategia de perturbación, estrategias de búsqueda local, criterio de aceptación y parámetros.

Algorithm 1 Iterated Local Search

1: $s \leftarrow$ heurística-vecino-mas-cercano-P()
2: $s \leftarrow$3-opt(s)
3: $s* \leftarrow s$
4: **repeat**
5: $s \leftarrow$intercambio-aleatorio(s)
6: $s \leftarrow$2-opt(s)
7: $s \leftarrow$3-opt(s)
8: $s^* \leftarrow$ si $f(s) < f(s^*)$
9: $s \leftarrow$ criterio-aceptación$(f(s) \leq f(s^*) : s, s^*)$
10: **until** 100 iteraciones

1.1. Representación

La represetanción usada es la de permutación, donde cada índice corresponde a una ciudad única, $s = \{1, ..., n\}$.

1.2. Función Objetivo

Se utiliza la suma de las distancias euclidianas en cada ciudad como se presenta a continuación:

$$f(s) = \sum_{i=1}^{n-1} distance(s_i, s_{i+1}) + distance(s_n, s_1) \tag{1}$$

1.3. Solución inicial

Se utilizó la heurística del vecino más cercano, donde el nodo inicial era el depósito, la cual representaba al valor 0 en las representaciones del problema. Así, la idea del algoritmo es escoger la instalación i más cercana al depósito, luego i pasa a ser la nueva instalación escogida. Repetir el procedimiento de las instalaciones más cercanas hasta que no hayan más instalaciones disponibles.
Por otra parte, se añadieron dos iteraciones **for**, para encontrar un subconjunto de instalaciones que cumplieran con satisfacer el beneficio P de la respectiva instancia. La lógica era aplicar un filtro de los clientes que se estaban cubriendo, buscando no duplicar clientes que son cubiertos por dos instalaciones o más. Luego, se recorre esa lista con los clientes que realmente se cubren y se extraen las instalaciones pertinentes. El procedimiento anterior se presenta en el **Algorithm 2**.

Algorithm 2 VecinoMasCercano-P

Input: V: ciudades, d_{ij} : distancia de la arista (i, j), P : Beneficio
Output: $s* = \{1, ..., n\}$

Sol-mejor $\leftarrow \emptyset$
new-data $\leftarrow \emptyset$
seen $\leftarrow \emptyset$
$i \leftarrow 0$ (depósito)
$S \leftarrow S \cup \{i\}$
$S \leftarrow V \setminus \{i\}$
$p \leftarrow 0$
while $p < P$ **do**
    $k \leftarrow$ mín$\{c_{ij} : j \in V\}$
    $S \leftarrow S \cup \{k\}$
    $V \leftarrow V \setminus \{k\}$
    $i \leftarrow k$
    for i in I-clientes **do**
        first $\leftarrow i[0]$
        second $\leftarrow i[1]$
        if second in S **then**
            if first in seen **then**
                continue
                new-data $\leftarrow i$
                seen $\leftarrow$ first
                p $\leftarrow$ #{seen}
            end if
        end if
    end for
    for i in new-data **do**
        Sol-mejor $\leftarrow i[1]$
        s $\leftarrow$ Sol-mejor
    end for
    if p $\geq$ P **then**
        break

1.4. Estrategias de Perturbación

Se determinan dos enfoques para la estrategia de perturbación. El primero consiste en escoger una i aleatoriamente e intercambiar la instalación de esa posición con la instalación de la posición siguiente $(i + 1)$ de la ruta como se presenta en el **Algorithm 3**.

Algorithm 3 perturbacion

Input: $s = \{1, ..., n\}$
Output: s^*

escoger aleatoriamente una posición i
$s^* \leftarrow$ nueva ruta considerando el intercambio de las instalaciones en las posiciones i y $i + 1$ =0

El segundo enfoque utilizado es una heurística de mejora que consiste en intercambiar instalaciones aleatoriamente 30 veces. Así, se tiene el **Algorithm 4**

Algorithm 4 intercambio-aleatorio

Input: $s = \{1, ..., n\}$
Output: s^*

repeat
    escoger aleatoriamente una posición i de s
    escoger aleatoriamente una posición j de s
    $s \leftarrow$ intercambiar las instalaciones de las posiciones i y j
until 30 veces

1.5. Estrategia de Búsqueda Local

En esta estrategia se utilizan dos enfoques. La primera es la búsqueda local 2-opt, la cual consiste en recorrer iterativamenete pares de arista e intercambiarlos. En caso de que con el intercambio la ruta mejore su costo, entonces, se mantiene, caso contrario se deshace. En particular, en esta implementación se utiliza la versión que si se encuentra una mejora el algoritmo se detiene. El algoritmo antes descrito se presentan en el **Algorithm 5**

Algorithm 5 2-Opt

Input: $S = \{v_1, v_2, ..., v_n\}$, c_{ij} : pesos de las aristas
Output: S' = Nueva Ruta

for $i \leftarrow 1$ hasta $n-2$ **do**
    for $j \leftarrow i+1$ hasta $n-2$ **do**
        if $c_{v_i v_{i+1}} + c_{v_j v_{j+1}} > c_{v_i v_{i+1}} + c_{v_{i+1} v_{j+1}}$ **then**
            Se desconecta (v_i, v_{i+1}) y (v_j, v_{j+1})
            Se reemplaza por (v_i, v_j) y (v_{i+1}, v_{j+1})
            Se actualiza el costo de la solución actual.
        end if
    end for
end for

El segundo enfoque fue una extensión del 2-Opt, ya que se utilizó el 3-Opt. Por simplicidad, se evita escribir el algoritmo. Sin embargo, hay que especificar que dicho algoritmo tiene 3 iteraciones de **for**.

1.6. Criterio de aceptación

Si la solución candidata es mejor que la actual, entonces candidata se acepta como solución actual, caso contrario se usa la mejor encontrada, vale decir, $f(s) \leq f(s^*)$.

1.7. Parámetros

El número de iteraciones se define en 100 y los demás datos son respectivos de la instancia. Ya sea $|I|$,$|V|$,$|C|$ y el beneficio P. También, las coordenadas entre instalaciones vienen dadas en dos dimensiones, es decir, como un vector (x_i, y_i)

2. Resultados Computacionales

El algoritmo propuesto es implementado en Python 3.7. Por otra parte, las pruebas se realizaron en un Intel(R) Core(TM) i5-7300HQ CPU 2.50 GHz y 16 GB de RAM (ejecutado en un solo un hilo), en el sistema operativo Windows 10.

Cuadro 1: Resultados instancias

instancias	**V**	**I**	**costo optimo**	**ERP**	**promedio**	**tiempo**
S1	51	15	71,33	37,39 %	98	0,0209
S2	51	15	105,98	30,21 %	138	0,1625
S3	51	15	172,36	2,11 %	176	0,4478
S4	51	20	75,29	2,27 %	77	0,0209
S5	51	20	102,59	22,82 %	126	0,1806
S6	51	25	38,49	0,00 %	38,49	0,0099
S7	51	25	82,38	28,67 %	106	0,1874
S8	51	25	140,62	8,09 %	152	0,6781
S9	52	16	1378,45	62,94 %	2246	0,0458
S10	52	16	2198,91	53,53 %	3376	0,0987
S11	52	21	669,76	53,79 %	1030	0,0239
S12	52	21	1554,5	45,51 %	2262	0,1002
S13	52	21	3910,04	7,36 %	4198	0,6252
S14	52	26	572,13	62,90 %	932	0,0505
S15	52	26	1240,67	45,57 %	1806	0,2838
S16	52	26	2958,78	10,89 %	3281	0,8803
S17	52	16	1340,29	10,95 %	1487	0,0449
S18	52	16	2291,58	71,72 %	3935	0,2749
S19	52	21	878,3	81,03 %	1590	0,0967
S20	52	21	1521,64	4,49 %	1590	0,0970
S21	52	21	3590,08	4,09 %	3737	0,4309
S22	52	26	479,76	93,22 %	927	0,0219
S23	52	26	1334,81	29,83 %	1733	0,1915
S24	52	26	2894,96	13,20 %	3277	0,6229
S25	76	23	98,94	20,27 %	119	0,2828
S26	76	23	147,89	19,01 %	176	0,8764
S27	76	30	86,16	20,71 %	104	0,1635
S28	76	30	119,54	51,41 %	181	2,5626
S29	76	38	57,79	34,97 %	78	0,1800
S30	76	38	106,39	51,33 %	161	0,4399
S31	76	38	173,59	21,55 %	211	2,1011
S32	76	38	20279,16	17,18 %	23763	0,2842
promedio	57,75	23,875	1583,53625	31,84 %	1972,234063	0,3902