Tour de Murales

Integrantes:
Juan Camilo Falla & Nicolás Klopstock

Entrega 3: Propuesta de un algoritmo para comparar su resultado con el Modelo Matemático
Modelado, Simulación y Optimización

Departamento de Ingeniería de Sistemas y Computación
Universidad de Los Andes
Bogotá, Colombia
Semetre 2023-10

1 Descripción del Problema

Una familia de turistas viaja a una ciudad que no conocen durante la época de vacaciones. Una específica tarde que no tienen planeado nada que hacer, deciden salir a ver los famosos murales de la ciudad. Estos son grandes pinturas en diferentes, valga la redundancia, muros de toda la ciudad. La familia quiere desea visitar murales en una tarde, sin alejarse mucho de su hotel para poder disfrutar de sus vacaciones sin que estas sean extenuantes (minimizar la distancia de recorrida desde hotel).

Adicionalmente, ellos desean descansar en algún momento de la tarde. Teniendo en cuenta lo anterior, se desea hacer una única parada en la cual la familia sea capaz de comprar refrescos y tomar su descanso. La familia conoce las distancias que hay entre cualquiera de los puntos de interés, motivo por el cual este valor será tomado como el costo del camino.

2 Descripción de limitaciones y maximización/minimización

La familia quiere visitar un cierto número de murales (nodos), minimizando la distancia (costo) de cada distancia entre murales (enlaces). Además, entre los puntos por los que quiere pasar, debe haber lugares de venta de refrescos, donde la familia pueda descansar antes de seguir su tour.

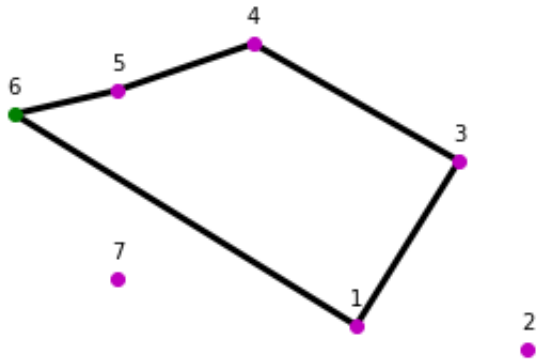


Fig. 1. Ejemplo de resultado del modelo con los valores que se verán en las restricciones. Nodos magenta: murales. Nodos verdes: refrescos.

3 Conjuntos, Parámetros y Variables

Table 1. Conjuntos, Parámetros y Variables de decisión.

Conjuntos y parámetros	**Descripción**
$i \in N$	Conjunto de nodos que se pueden visitar
$k \in M$	Conjunto de murales que se pueden visitar.
$l \in R$	Conjunto de lugares en donde pueden parar a descansar.
$f \in F$	Cantidad de enlaces parte del tour.
C_{ij}	Conjunto de distancias entre cualquier punto de interés (murales y sitios de refrescos).

Table 2. Variables de decisión

Variables	**Descripción**
X_{ij}^{f}	Variable binaria que determina la selección de un enlace.
W_i	Variable binaria que determina la selección de un nodo.

4 Función Objetivo y Restricciones

$$min \sum_{i \in N} \sum_{j \in N} \sum_{f \in F} C_{ij} * X_{ij}^{f} \tag{1}$$

4.1 Función Objetivo

La F.O hace referencia a que se quiere minimizar el costo de cada camino para ver un cierto número de murales y pasando por algunos lugares de descanso. Esta minimización logra encontrar el camino de menor peso (el camino más corto) entre algunos nodos de un mismo grafo.

4.2 Restricciones

La primera restricción hace referencia a no repetir enlaces:

$$x_{ij}^{f} + x_{ij}^{f} \leq 1 \qquad \forall_{ij \in N, f \in F} \tag{2}$$

La segunda restricción hace referencia a que el siguiente enlace escogido no sea el mismo anterior en dirección contraria (ej. enlace 2-5 y 5-2):

$$x_{ij}^{f} + x_{ij}^{g} \leq 1 \qquad \forall_{ij \in N, fg \in F} / f \neq g \tag{3}$$

Notar que g es solo una copia de f, para diferenciar ambos enlaces.

La tercera restricción hace referencia a que, por cada enlace f, solo un par *(i,j)* debe ser escogido:

$$\sum_{ij \in N} x_{ij}^{f} = 1 \qquad \forall_{f \in F} \tag{4}$$

La cuarta restricción hace referencia a que, para cada par *(i,j* debe haber, a lo sumo, un enlace f activado:

$$\sum_{f \in F} x_{ij}^{f} \leq 1 \qquad \forall_{ij \in N} \tag{5}$$

La quinta restricción hace referencia a que, para el nodo inicial, debe salir un enlace:

$$\sum_{j \in N} x_{ij}^{f} = 1 \qquad \forall_{i \in N, f \in F} / i = INICIAL \wedge f = 1 \tag{6}$$

La sexta restricción hace referencia a que, si a un nodo le entra un enlace f_x, le debe salir un enlace f_{x+1}:

$$\sum_{k \in M} x_{jk}^{d} \geq x_{ij}^{f} \qquad \forall_{ij \in N, fd \in F} / d = f + 1 \wedge f < TOTAL_{VISITAR} \tag{7}$$

Notar que $TOTAL_{VISITAR}$ es nodos mural + nodos descanso + nodo hotel.

La séptima restricción hace referencia a la condición de que, si se escoge un par de nodos *(i,j)* y su enlace *f*, se debe activar también la variable w:

$$w_i \geq x_{ij}^{f} \qquad \forall_{ij \in N, f \in F} \tag{8}$$

La novena restricción hace referencia a que el total de murales visitados no se puede pasar del total de deseados:

$$\sum_{k \in M} w_k = DESEADOS \tag{9}$$

La décima restricción hace referencia a que el total de nodos seleccionados debe ser igual que los establecidos (nodo inicial + murales + un lugar de refresco):

$$\sum_{i \in N} w_i = MUST_{VISIT} \tag{10}$$

La decimoprimera restricción hace referencia a que el total de lugares de refresco visitados no se puede pasar del total de 1.

$$\sum_{l \in R} w_l = 1 \tag{11}$$

La decimosegunda restricción hace referencia a que el nodo origen no puede ser un nodo intermedio del camino. Además, ayuda a forzar que quede como nodo cierre del ciclo:

$$\sum_{i \in N} x_{ij}^{f} = 1 \qquad \forall_{j \in N, f \in F} / j = INICIAL \wedge f = MUST_{VISIT} \tag{12}$$

La decimotercera y última restricción hace referencia a que, los nodos activados, solo les salga un enlace:

$$\sum_{j \in N, f \in F} x_{ij}^{f} \leq 1 \qquad \forall_{i \in N} \tag{13}$$

5 Implementación y resultados del Modelo Matemático

En esta sección se van a describir dos escenarios de prueba para demostrar el funcionamiento del modelo matemático. Importante notar que, para las representaciones gráficas, magenta es un nodo mural, verde es un nodo de lugar de refresco y azul es el hotel.

5.1 Escenario 1

El escenario 1 consta de una versión sencilla del problema en el cuál la familia desea visitar un único mural en un día dado de sus vacaciones. En el grafo se encuentra: el hotel, 3 murales y un punto de refrescos. La limitación de una sola parada por un refresco se mantiene y los parámetros que entrarán al problema son:

Table 3. Parámetros del modelo

Parámetros	**Valores**
$\{n_1 * n_6\}$	Conjunto de nodos que se pueden visitar.
*Ver **Table. 4***	Conjunto de distancias entre cualquier punto de interés (murales y sitios de refrescos)
$\{t_1 * t_3\}$	Conjunto de tipos de nodos. $t_1 :=$ hotel, $t_2 :=$ mural, $t_3 :=$ refresco
1	Cantidad de murales a visitar: **DESIRED** $\in \mathbb{N}$
n_1	Nodo origen (indica el nodo hotel)
*Ver **Table. 5***	Conjunto que establece si hay un enlace entre un nodo y otro.

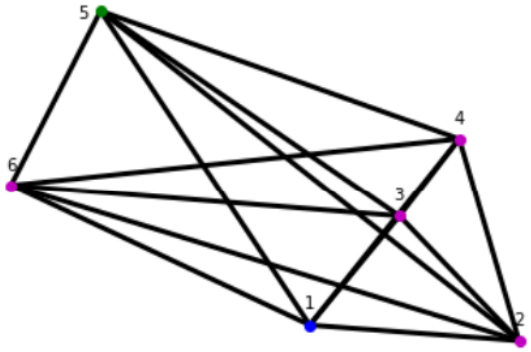
Table. 4. Conjunto de distancias.

	n1	**n2**	**n3**	**n4**	**n5**	**n6**
n1	999	2	5	5	2	9
n2	2	999	3	5	1	9
n3	5	3	999	2	3	11
n4	5	5	2	999	4	12
n5	2	2	3	4	999	4
n6	6	2	3	6	4	999

Table. 5. Conjunto de enlaces existentes.

	n1	**n2**	**n3**	**n4**	**n5**	**n6**
n1	0	1	1	1	1	1
n2	1	0	1	1	1	1
n3	1	1	0	1	1	1
n4	1	1	1	0	1	1
n5	1	1	1	1	0	1
n6	1	1	1	1	1	0

Fig. 2. Representación gráfica del escenario.



5.2 Resultados Escenario 1

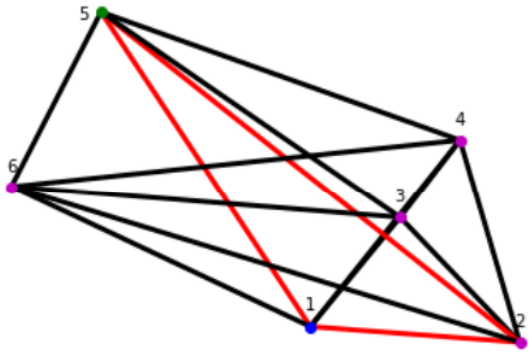
Se van a mostrar los resultados de este escenario:

Table. 6. Resultados Escenario 1

```
----    125 VARIABLE x.L  Indicates if the link i-j is selected or not.

               f1          f2          f3

n1.n2       1.000
n2.n5                   1.000
n5.n1                               1.000


----    126 VARIABLE w.L  Lleva cuenta de los nodos visitados.

n1 1.000,    n2 1.000,    n5 1.000


----    127 VARIABLE z.L                  =        5.000  Objective function.
```

Teniendo en cuenta los resultados del escenario 1 podemos ver que el modelo logra resolver un caso sencillo de manera correcta. Este elige todos los enlaces que le permiten cumplir con todas las restricciones de puntos a visitar deseados por la familia mientras que mantiene el camino de menor costo para que la familia se movilice la menor distancia posible. Esto se evidencia por distintos factores, en primer lugar, podemos ver que el enlace f1 se origina en el nodo 1 el cuál es considerado como el hotel en el cual se aloja la familia. Adicionalmente se puede evidenciar una secuencia que se relaciona entre los f y los enlaces elegidos para cada f_x. Se puede observar que el nodo destino del primer enlace corresponde directamente con el nodo origen del segundo y este patrón se repite para los tres enlaces. Adicionalmente, la variable $W(i)$ nos permite validar que se cumplan con las restricciones de cantidad murales deseados y de la parada y también que estos están siendo visitados en la matriz de resultados mostrada anteriormente.

Fig. 3. Representación gráfica de los resultados del escenario



5.3 Escenario 2

El escenario 2 consta de una versión más compleja del problema en el cuál la familia desea visitar 4 murales en un día dado de sus vacaciones. En el grafo se encuentra: el hotel, 6 murales y dos puntos de refrescos. La limitación de una sola parada por un refresco se mantiene y los parámetros que entrarán al problema son:

Table. 7. Parámetros del modelo

Parámetros	**Valores**
$\{n_1 * n_9\}$	Conjunto de nodos que se pueden visitar.
Ver ***Table. 4***	Conjunto de distancias entre cualquier punto de interés (murales y sitios de refrescos)
$\{t_1 * t_3\}$	Conjunto de tipos de nodos. $t_1 :=$ hotel, $t_2 :=$ mural, $t_3 :=$ refresco
1	Cantidad de murales a visitar: **DESIRED** $\in \mathbb{N}$
n_1	Nodo origen (indica el nodo hotel)
Ver ***Table. 5***	Conjunto que establece si hay un enlace entre un nodo y otro.

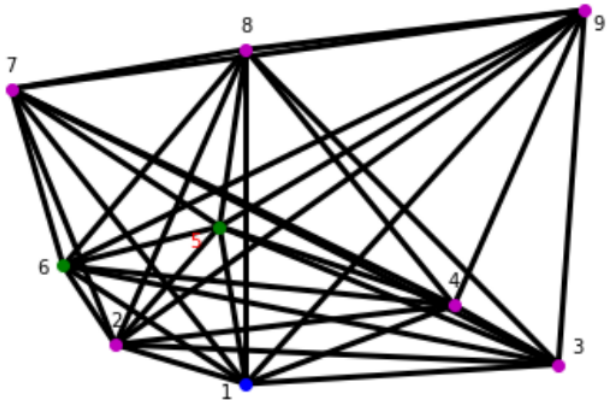
Table. 8. Conjunto de distancias.

	n1	**n2**	**n3**	**n4**	**n5**	**n6**	**n7**	**n8**	**n9**
n1	999	8	5	4	20	6	22	32	12
n2	8	999	4	10	13	10	23	42	22
n3	5	4	999	2	15	17	22	4	3
n4	4	10	2	999	4	14	6	30	11
n5	20	13	15	4	999	8	10	6	20
n6	6	10	17	14	8	999	10	8	10
n7	22	23	22	6	10	10	999	12	19
n8	32	42	4	30	6	8	12	999	8
n9	12	22	3	11	20	10	19	8	999

Table. 9. Conjunto de enlaces existentes.

	n1	n2	n3	n4	n5	n6	n7	n8	n9
n1	0	1	1	1	1	1	1	1	1
n2	1	0	1	1	1	1	1	1	1
n3	1	1	0	1	1	1	1	1	1
n4	1	1	1	0	1	1	1	1	1
n5	1	1	1	1	0	1	1	1	1
n6	1	1	1	1	1	0	1	1	1
n7	1	1	1	1	1	1	0	1	1
n8	1	1	1	1	1	1	1	0	1
n9	1	1	1	1	1	1	1	1	0

Fig. 4. Representación gráfica del escenario.



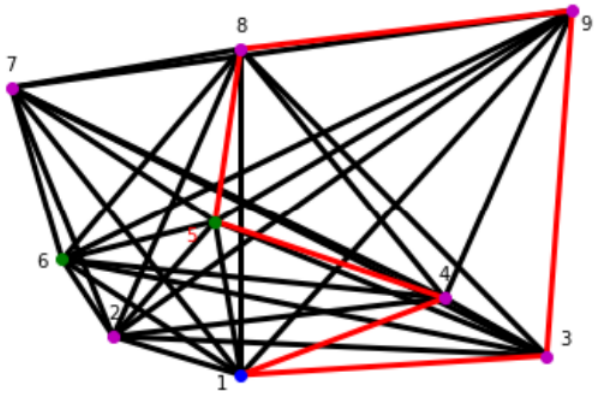
5.4 Resultados Escenario 2

Table. 10. Resultados Escenario 1

```
----    130 VARIABLE x.L  Indicates if the link i-j is selected or not.

                f1          f2          f3          f4          f5          f6

n1.n3        1.000
n3.n9                    1.000
n4.n1                                                                  1.000
n5.n4                                                      1.000
n8.n5                                          1.000
n9.n8                                1.000


----    131 VARIABLE w.L  Lleva cuenta de los nodos visitados.

n1 1.000,    n3 1.000,    n4 1.000,    n5 1.000,    n8 1.000,    n9 1.000


----    132 VARIABLE z.L                  =       30.000  Objective function.
```

Teniendo en cuenta los resultados del escenario 2 podemos ver que el modelo logra resolver un caso más complejo y grande de manera correcta. Este elige todos los enlaces que le permiten cumplir con todas las restricciones de puntos a visitar deseados por la familia mientras que mantiene el camino de menor costo para que la familia se movilice la menor distancia posible. Esto se evidencia por distintos factores, en primer lugar, podemos ver que el enlace f1 se origina en el nodo 1 el cuál es considerado como el hotel en el cual se aloja la familia. Adicionalmente se puede evidenciar una secuencia que se relaciona entre los f y los enlaces elegidos para cada f_x. Se puede observar que el nodo destino del primer enlace corresponde directamente con el nodo origen del segundo y este patrón se repite para los seis enlaces. Adicionalmente, la variable $W(i)$ nos permite validar que se cumplan con las restricciones de cantidad murales deseados y de la parada y, también, que estos están siendo visitados en la matriz de resultados mostrada anteriormente.

Fig. 5. Representación gráfica de los resultados del escenario



6 Algoritmo propuesto

Teniendo en cuenta el código utilizado, se puede afirmar que el codigo creado utiliza una heurística totalmente nueva. Este realiza una búsqueda exhaustiva de todos los caminos posibles que cumplen con la cantidad de nodos que deberían ser visitados según los parámetros del problema. Una vez se obtengan todas las permutaciones de nodos, se procede a hacer una verificación exhaustiva, en donde se busca validar qué rutas cumplen con las restricciones de m cantidad de murales y 1 nodo de refresco junto con el inicio y el fin de la ruta muestren el nodo hotel. Si la permutación cumple con estas restricciones, se procede a calcular el costo de la ruta. Si este costo es menor al costo mínimo actual, se actualiza el resultado. El algoritmo termina una vez se recorren todas las permutaciones posibles.

Se considera una heurística totalmente nueva debido a que el problema planteado se encuentra restringido por restricciones altamente específicas. Sin embargo, es importante aclarar que el método exhaustivo no es una novedad.

6.1 Pseudocódigo del algoritmo

(En la siguiente página.)

Algorithm 1 Pseudocódigo.

```
Inicializar permutaciones_caminos → posibles permutaciones entre caminos
num_murales = 1
num_parques = 1
num_hoteles = 1

function validarPermutaciones(permutaciones, num_murales, num_parques, num_hoteles)
    validos = []
    for camino, recorrer cada camino de cada permutacion do
        cont_murales = 0
        cont_parques = 0
        cont_hoteles = 0
        for nodo, recorrer cada nodo de cada camino do
            if tipo del nodo == mural then
                cont_murales+ = 1
            else if tipo del nodo == parque then
                cont_parques+ = 1
            else
                cont_hoteles+ = 1
            end if
        end for
        if contador de cada tipo es igual al parametro dado then
            agregar camino al arreglo de permutaciones validas
        end if
    end for
    for camino, recorrer caminos supuestamente validos do
        for j, act_node, recorrer los elementos de cada camino do
            if j < (tamanio de camino − 1) then
                sig_nodo = nodo siguiente al actual
                if sig_nodo no esta en las conexiones del nodo actual then
                    break
                end if
            else if j == (tamanio de camino − 1) then
                sig_nodo = nodo origen del camino
                if sig_nodo no tiene conexion con act_nodo then
                    break
                end if
            end if
        end for
    end for
    return validos
end function

function calculoCostoCamino(permutacion)
    cost_camino = 0
    for j, act_nodo, recorrer los elementos de cada camino do
        if j < (tamanio de camino − 1) then
            sig_nodo = nodo siguiente al actual
            cost_camino = costo del camino entre act_node y sig_nodo
        else if j == (tamanio de camino − 1) then
            sig_nodo = nodo origen
            cost_camino = costo del camino entre act_node y sig_nodo
        end if
    end for
    return cost_camino
end function

function caminoMinimoCosto(permutaciones)
    costo_min = ∞
    camino_min = None
    costo_camino = 0
    for camino, recorrer cada camino do
        costo_camino = calculoCostoCamino(camino)
        if costo_camino < costo_min then
            costo_min = costo_camino
            camino_min = camino
        end if
    end for
    return (costo_min, camino_min)
end function
```

6.2 Explicación del algoritmo

Lo primero que se hace es sacar las diferentes permutaciones posibles entre los nodos del grafo que entra como parámetro. Estas representan posibles caminos que se van a evaluar (la creación de las permutaciones se hace con herramientas externas, por lo que no se incluyó en el pseudocódigo). Estas permutaciones se deben, luego, validar, para confirmar que cumplen con el número requerido de cada tipo de nodo; además de revisar que sean caminos posibles (según las conexiones del grafo). Eso hace la función *validarPermutaciones()*.
Luego, se debe sacar el camino con el costo mínimo entre todas las permutaciones válidas. Para esto, se recorren todas las mencionadas, calculando individualmente el costo de cada camino (esto se hace en la función *calculoCostoCamino()*. Esta función se llama en *caminoMinimoCosto()* para ejecutar lo explicado hace un par de lineas. Cuando se encuentra el costo mínimo y el camino al que le corresponde este costo, se retorna.

7 Resultados del Algoritmo vs Modelo Matemático

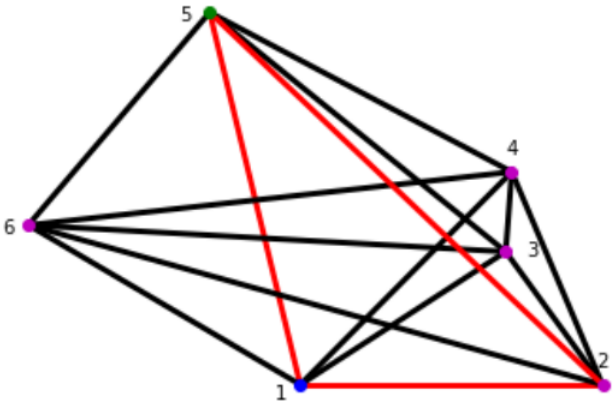
Se van a evaluar dos casos implementados en el algoritmo (mostrado anteriormente) y el modelo matemático.
Nota: Los casos son los mismos a los explicados en la sección del modelo matemático, pero ajustados al lenguaje usado para implementar el algoritmo.

7.1 Escenario 1

Al observar los resultados del escenario 1, se puede observar que la ruta y el costo de la ruta es el mismo que en el modelo matemático planteado. Esto nos permite validar que el algoritmo propuesto fue desarrollado de manera correcta y que este puede llegar a ser una alternativa viable de solución. Tanto en los resultados del modelo matemático como en los resultados del algoritmo, se puede observar que la ruta seleccionada es 1-2-5-1 con un costo de 15 en la distancia entre los puntos. Es por esto que podemos decir que el algoritmo heurístico planteado nos provee una solución consistente, después de varios intentos, y al igual que el modelo matemático este logra cumplir con el objetivo de la familia, moverse la menor distancia posible mientras que logran visitar la cantidad de murales deseados.

7.2 Resultados Escenario 1

Fig. 6. Representación gráfica de los resultados del escenario

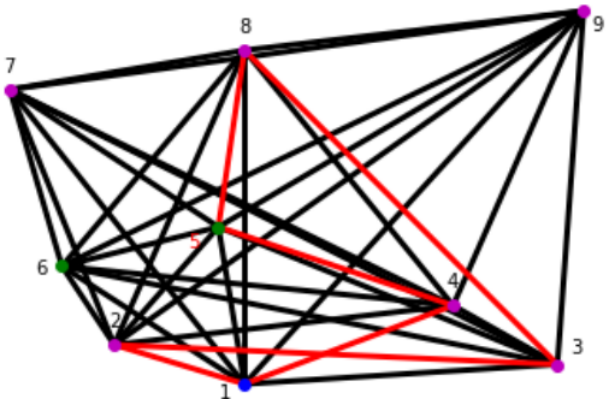


7.3 Escenario 2

En este escenario, nuevamente se puede observar que el algoritmo provee la misma solución que el modelo matemático planteado previamente, sin embargo, esta solo es idéntica en el costo, no en la ruta propuesta. Sin embargo, la ruta propuesta cumple con las restricciones establecidas por el problema. Es importante aclarar que esta diferencia en los resultados puede radicar en el hecho del orden en el cuál esta permutación es creada y en el orden el cual su costo es calculado. Debido a que esta ruta alterna se calcula primero que la ruta propuesta por el modelo matemático, esta se mantiene como la ruta de menor costo, aún si hay dos rutas con el mismo costo. Por último, se puede concluir que el algoritmo en este escenario también logra solucionar de manera óptima el problema propuesto.

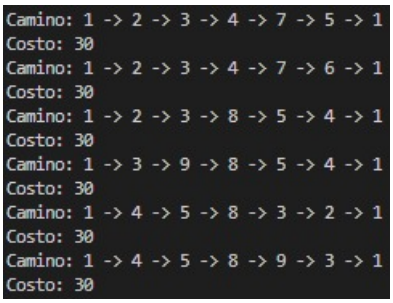
7.4 Resultados Escenario 2

Fig. 7. Representación gráfica de los resultados del escenario



Algo curioso que se puede ver en este escenario es que el algoritmo encuentra varios caminos (en caso de que varios tengan el mismo costo), pero muestra solo el primero encontrado (se puede modificar para retornar el último encontrado):

Fig. 8. Total de resultados encontrados por el algoritmo



Como se puede ver, el cuarto camino encontrado por el algoritmo es el mismo que el encontrado por el modelo matemático.