Inteligencia Artificial

Milk Collection with Blending using a Greedy+SA Approach

Sebastian Calderon Diaz

21 de julio de 2021

Evaluación

Mejoras 1ra Entrega (10 %):	_____
Código Fuente (10 %):	_____
Representación (15 %):	_____
Descripción del algoritmo (20 %):	_____
Experimentos (10 %):	_____
Resultados (10 %):	_____
Conclusiones (20 %):	_____
Bibliografía (5 %):	_____
Nota Final (100):	_____

Resumen

El problema de recolección de leche con mezcla busca hallar las rutas óptimas para un conjuntos de camiones que tiene que recolectar leche de distintas granjas. En este paper se estudia los avances en los métodos de resolución que ha tenido el problema, y se propone un enfoque greedy y SA para resolver el problema.

1. Introducción

El costo de recolectar leche en la cadena de producción de leche tiene un impacto significativo en la utilidad (Rojas and Lusa, 2005; Lahrichi et al., 2013). Los productores de leche por lo general están dispersos, y por esto mismo que se planifica una ruta para recolectar la leche. La planificación resulta extremadamente difícil para las empresas recolectoras, distintos factores como el coste de transporte, el beneficio por transportar leche de x calidad se deben considerar. Es mas, en un caso real influyen muchas mas variables como los costos de energía, los camiones podrian tener compartimientos con distintas capacidades, un limite de tiempo de entrega, etc.

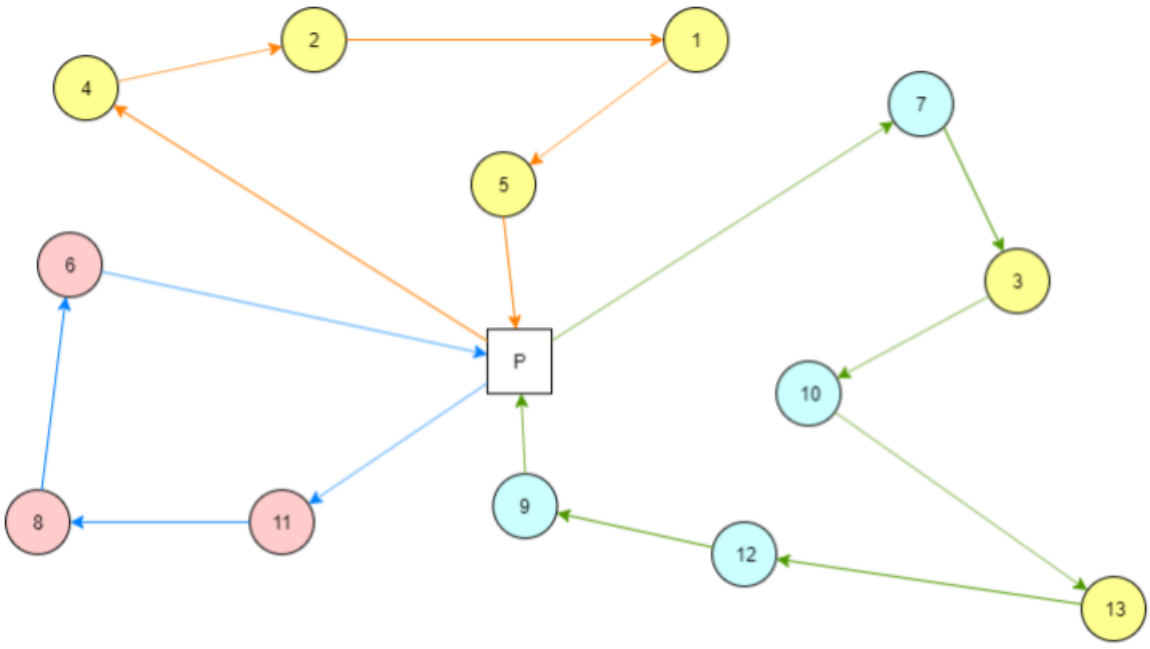
El problema de recolección de leche con mezcla consiste en buscar las rutas óptimas para un conjuntos de camiones que tiene que recolectar leche de distintas granjas, además se tienen que cumplir ciertos requerimientos.
La estructura del informe sera la siguiente: Primero definiremos el problema, luego estudiaremos los avances que se han tenido para luego proponer nuestro propio algoritmo y comparar resultados con la literatura.

2. Definición del Problema

El problema de recoleccion de leche con mezcla, consiste en hallar las rutas optimas para un conjunto de camiones, los cuales tienen que recolectar leche de distintas granjas. En este problema se considera que los camiones poseen un solo compartimiento, y además se pueden mezclar leches de distintos tipo, pero la calidad de la mezcla queda como la mas baja de las mezcladas.
Con el fin de hallar la ruta óptima se consideran los costos de viajar de un nodo a otro (granja-granja o granja-planta), y los beneficios de entregar la leche a la planta con cierta calidad.

A modo de ejemplificar lo definido, se muestra una instancia resulta obtenida de [2]. Se cuenta con 13 productores de leche (nodos), 3 camiones y 1 planta (cuadrado). Cada productor produce leche de distinta calidad, es por eso mismo, que en la imagen cada nodo tiene un color distinto asociado con la calidad producida. Las flechas representan la dirección que debe tomar el camión, puesto que se cuenta con 3 camiones, hay 3 rutas óptimas.
Con esta instancia, nos podemos dar cuenta de cinco características importantes de este problema.

1. Los camiones parten en la planta, y terminan en la misma planta
2. Un productor solo puede ser visitado una vez.
3. Un camion puede visitar una granja que produce leche de distinta calidad, pero la leche queda mezclada.
4. Si el camion visita la granja, esta obligado a recoger la leche.
5. El camion solo puede visitar la granja, si tiene la capacidad para hacerlo.



El problema mencionado es un caso especial del problema de ruteo de vehiculos(VRP), que es un problema de optimizacion combinatoria, y ademas es NP. Sin embargo MILK, es considerado aun mas complejo, que VRP, a pesar de ser un caso especial. Por esto mismo, es complejo

resolver instancias grandes.

Las variantes mas conocidas en la planificacion de recoleccion de leche son las siguientes:

1. Se considera que los camiones pueden tener mas de un compartimiento para transportar la leche.
2. Los productores de leche almacenan la leche en un puntos de recolección.
3. Se considera que existe una sola calidad de leche.
4. Se considera una flota de camiones homogenea en capacidad.

3. Estado del Arte

El problema fue propuesto por Paredes-Belmar, G., Marianov, V., Bronfman, A., Obreque, C., Luer-Villagra, A. en el año 2016 [1]. Si bien el problema de recolección de leche se había planteado antes, ellos fueron los primeros en considerar la mezcla de la leche.

Proponen un modelo de programación entera mixta, y para resolverlo proponen dos algoritmos en base al tamaño del problema:

- Instancias pequeñas y medianas proponen usar branch and cut, y ademas proponen planos para relajar el problema.
- Instancias grandes ($\geq$ 100 granjas) proponen usar una heurística de tres etapas para dividir el problema en subproblemas, y resolverlos por separados:
    1. Primera etapa: Las granjas se separan en cluster (tecnica para separar grupos de elementos), usan una heuristica geografica para ello.
    2. Segunda etapa: Se asigna a cada cluster una cuota mínima de leche para garantizar que se cumpla la cuota mínima que necesita la planta.
    3. Tercera etapa: Se usa branch and cut para resolver cada cluster.

Resolvieron una instancia real en 23836 segundos (6 horas aprox.) con 500 granjas y 100 camiones, para resolverlo separaron las granjas en base a la geografía, formando 25 grupos. Luego compararon las ganancias entre las rutas óptimas obtenidas y las rutas usadas por la empresa, obteniendo un incremento en la ganancia de 217 %.

Por ultimo compararon la utilidad, considerando un compartimiento con y sin mezcla. También consideraron un camión con múltiples compartimiento con y sin mezcla. Concluyendo que se obtiene una mejor utilidad, si se considera la mezcla.

Jorge Villagran propone dos algoritmos basados en búsqueda local para resolver el problema en el año 2019[2]. Un algoritmo basado en hill-climbing y el otro es una variante del mismo llamado iterated local search.
Luego resuelven distintas instancias y las comparan con los que obtuvieron Paredes-Belmar, obteniendo en la mayoría de los casos mejores resultado en cuanto al tiempo de ejecución. Sin embargo en algunas instancias no pudieron llegar al óptimo, el algoritmo que tuvo mejor resultado en cuanto a tiempo de ejecución y a la calidad de la misma, fue iterated local search. Al finalizar las pruebas para el hill-climbing, suma los tiempos de las distintas instancia, y afirma que solo necesito usar 36 % del tiempo total que usaron Paredes-Belmar para resolver todas las instancias. En el caso del iterated local search solo necesito 33 %.
Con respecto a la instancia real con 500 granjas, ambos algoritmos obtuvieron una solución de mejor calidad y en menor tiempo que Paredes-Belmar.
Al final concluye que las tecnicas de busqueda local son buenas para obtener una solucion de buena calidad en un corto periodo de tiempo. A diferencia de tecnicas completas que son mas

costosas, y mas dificiles de implementar.

4. Modelo Matemático

4.1 Modelo propuesto

El modelo propuesto es un MIP, el modelo considera 5 variables, de las cuales 3 son binarias y las otras 2 son positivas o igual a cero.
El modelo se obtuvo de [1].

4.2 Variables

$$x_{ij}^{k} = \begin{cases} 1, & \text{Si el camion k viaja directamente del nodo i al nodo j} \\ 0, & \text{en caso contrario} \end{cases}$$

$$y_{i}^{kt} = \begin{cases} 1, & \text{Si el camión k recoge leche de calidad t de la granja i} \\ 0, & \text{en caso contrario} \end{cases}$$

$$z^{kt} = \begin{cases} 1, & \text{Si el camión k entrega leche de calidad t a la planta} \\ 0, & \text{en caso contrario} \end{cases}$$

w^{kt} = Volumen de leche de calidad t que el camión k entrega a la planta

v^{tr} = Volumen de leche de calidad t entregada a la planta, mezclada para su uso como leche de calidad r

4.3 Parámetros

A : Conjunto de arcos que representan caminos entre granjas de leche

A^0 : Conjunto de arcos que representan caminos entre entre la planta y productores de leche

$N =$ { 0,...,n} Granjas de leche

N_0 : Conjunto de granjas y la planta

K : Conjunto de camiones

T : Conjunto de calidades de leche

N^t : Conjunto de granjas de leche de calidad $t \in T$

D^t : Resultado de la mezcla de leche de calidad r con leche de calidad t

IT : Conjunto de pares ordenados (i,t) de granjas i y leche de calidad t

Q^k : Capacidad de cada camión k

q_i^t : Cantidad de leche t producida por la granja i

c_{ij}^k : Costo de viaje de cada camion k sobre el arco $(i, j) \in A \cup A^0$

α^t : Ingreso por unidad de leche de calidad t

P^t : Requerimientos de leche de calidad t de la planta

$K_i = \{k \in K : t \in T, q_i^t \leq Q^k t \ \forall t \ \}$ Conjunto de camiones que pueden visitar la granja i

$K_{ij} = \{k \in K : q_i^t + q_j^t \leq Q^k \ \forall t \ \}$ Conjunto de camiones que pueden visitar de la granja i hasta la j recolectando la leche que ambas granjas producen sin exceder la capacidad del camion, para cada arco $(i, j) \in A \cup A^0$

0_k : Nodo en el que el camión k inicia la ruta

$AK = \{(i, j, k) : (i, j) \in A \cup A^0, K \in K_{ij}\}$ Conjuntos de elementos que contiene un arco que va de la granja i a la granja j y un camion k que pertenece a K_{ij}

4.4 Función objetivo

La funcion objetivo corresponde a la maximizaón del beneficio. Se considera la diferencia entre la ganancia producida por la leche y los costes de transporte.

$$Max \sum_{t \in T} \sum_{r \in T} \alpha^r v^{rt} - \sum_{(i,j,k) \in AK} c_{ij}^k x_{ij}^k$$

4.5 Restricciones

- Cada camión no puede recolectar más leche que su capacidad máxima.

$$\sum_{t \in T} \sum_{i \in N:(i,t) \in IT} q_i^t y_i^{kt} \leq Q^k, \quad \forall k \in K$$

- La recolección de la leche de cada granja debe ser realizada por exactamente un camión. Esto implica que se debe recolectar la leche de todas las granjas y que cada granja no puede ser visitada más de una vez.

$$\sum_{k \in K_i} y^{kt} = 1, \quad \forall i \in N, t \in T : (i, t) \in IT$$

- Cada camión debe tener como máximo una ruta, que comienza desde la planta.

$$\sum_{j:(0_k,j,k)\in AK} x^k_{0_k j} \leq 1 \quad \forall k \in K$$

- Control de flujo para el orden de las visitas de los nodos por parte de cada camión.

$$\sum_{i:(i,j,k)\in AK} x^k_{ij} = \sum_{h:(j,h,k)\in AK} X^k_{jh} \quad \forall k \in K_j, j \in N_0$$

- Cada camión que visita cada granja debe detenerse y recoger su leche.

$$\sum_{p:(p,i,k)\in AK} x^k_{pi} = y^{kt}_i \quad \forall k \in K_i, i \in N, t \in T : (i,t) \in IT$$

- Tipo de leche según las granjas que ha visitado.

$$z^{kt} \leq 1 - \sum_{r\in D^t: r\neq t,(i,r)\in IT} y^{kr}_i \quad \forall k \in K_i, i \in N, t \in T$$

- Cada Camión solo entrega un tipo de leche a la planta.

$$\sum_{t\in T} z^{kt} \leq 1 \quad \forall k \in K$$

- La cantidad de leche entregada de cada tipo por cada camión a la planta no supera su capacidad.

$$w^{kt} \leq z^{kt} Q^k \quad \forall k \in K, t \in T$$

- La cantidad recolectada de cada tipo de leche por cada camión, considerando las granjas que ha visitado, no supera su capacidad.

$$w^{kt} \leq \sum_{r:i\in D^r} \sum_{h\in N^r} q^r_h y^{kr}_h \quad k \in K, t \in T$$

- Cada camión debe llevarse toda la leche producida pro cada granja a la planta.

$$\sum_{k\in K} \sum_{t\in T} w^{kt} = \sum_{(i,t)\in IT} q^t_i$$

- Se debe equilibrar la cantidad de leche de cada calidad que llega la planta y a la cantidad restante después de la mezcla en planta.

$$\sum_{r\in D^t} v^{tr} = \sum_{k\in K} w^{kt} \quad \forall t \in T$$

- Se deben satisfacer las cuotas de leche de cada tipo.

$$\sum_{t\in T} v^{tr} \geq P^r \quad \forall r \in D^t$$

- Se debe evitar la mezcla de leche prohibidas

$$y^{kt}_i + y^{kr}_i \leq 1 \quad \forall (t,r) \in PM; (i,t), (j,t) \in IT$$

- Se debe evitar la aparición de sub-ciclos en las rutas de cada camión.

$$\sum_{i\in S} \sum_{j\in S} x^k_{ij} \leq |S| - 1 \quad \forall S \subseteq N, k \in K$$

4.6 Naturaleza de las variables

- Variables asociadas a los tipos de leche recolectados y mezclados en la ruta

$$y_i^{kt}, z^{kt} \in 0,1 \quad \forall i \in N, k \in K_i, t \in T : (i,t) \in IT$$

- Variable que controla la ruta del camión

$$x_{ij}^k \in 0,1 \quad \forall (i,j,k) \in AK$$

- Vólumenes de leche entregados y mezclados en la planta no negativos.

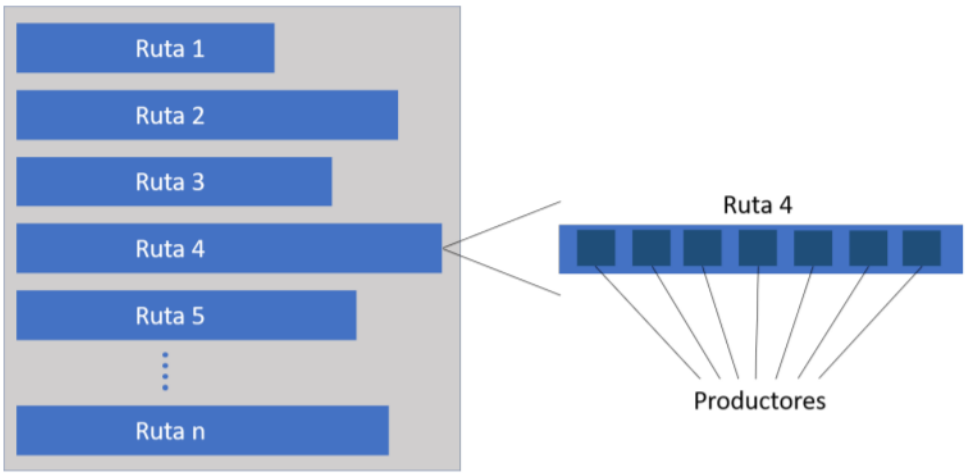
$$w^{kt}, v^{tr} \geq 0 \quad \forall k \in K; t, r \in T, r \in D^t$$

4.7 Espacios de búsqueda

El espacio de búsqueda esta condicionado por las variables que componen la función objetivo, V^{rt} y x_{ij}^k. La primera no tiene espacio de búsqueda finito, pero esta restringido por la segunda variable. La segunda tiene un espacio de búsqueda de $2^{|AK|}$, donde $|AK|$ corresponden a la cardinalidad del conjunto AK. y_i^{kt}, z^{kt} tienen un espacio de busqueda de $2^{|IT|}$ cada uno.

5. Representación

Dado que el objetivo de MCwb es encontrar las rutas optimas de cada camion, conocer que granjas se visitan y en que orden, la representacion que se utilizo fue de lista, puesto que puede representar las condiciones necesarios para una solucion.
Mas específicamente se uso una lista dinamica de tamańo n, donde n en el numero de camiones. En cada indice la lista contiene una ruta, que pertenece a un camion respectivamente.
En la imagen se puede apreciar mas claramente, la lista contiene n rutas, y cada ruta corresponde a un camión respectivamente. La ruta contiene una series de numeros enteros, que son los productores a visitar en orden.



Por ejemplo tomemos la siguiente solución [[1,2],[3,4],[5,6]]. La solucion se lee de la siguiente manera, el primer camion parte en la planta nodo 0 luego visita a los productores 1 y 2, en ese orden, y se devuelve a la planta. Luego, el segundo camion parte en la planta luego visita a los productores 3 y 4, en ese orden, y se devuelve a la planta. Finalmente el tercer camion parte en la planta nodo 0 luego visita a los productores 5 y 6, en ese orden, y se devuelve a la planta.

6. Funcion de evalucion

La calidad de la solucion se determina por la funcion de evalucion, la funcion esta compuesta por la funcion objectivo y dos terminos que penalizan la solucion en caso que sea infactible.
La funcion de evaluacion primero suma las ganancias obtenidas por cada tipo de leche, luego resta los coste de transporte respectivamente. En caso que la solucion sea infactible calcula dos tipos de penalizacion. El primero consiste en penalizar los litros que no fueron entregados segun los requerimientos, esto se multiplica por un factor que va de 0 a 1. El segundo tipo penaliza los litros excedidos de cada camion.

$$calidad = profit\_leche - coste\_transporte - (litros\_bajocuota * f\_pen\_sobre + sobre\_capacidad * f\_pen\_cap) \quad (1)$$

7. Movimiento

El movimiento usado fue de un 2-opt. En un inicio se eligen dos rutas random (pueden ser las misma), luego en cada ruta se elige el nodo a intercambiar nuevamente de manera random. Por ultimo se intercambian los nodos.
El movimiento fue elegido porque permite explorar los distintos valores que puede tomar una solucion, y ademas al ser un grafo completo no se generan problemas de inconsistencia con respecto a la dirección.

8. Descripción del algoritmo

Se comienza generando una solucion inicial con un enfonque pseudo-greddy, luego se procede con el algoritmo de simulated annealing.

Con el fin de generar una solucion inicial greddy, podemos pensar en dos decisiones greddy para ello. Primero elegir a las granjas que esten mas cerca, y segundo elegir a las granjas que producen leche de mejor calidad.
El algoritmo para generar la solucion inicial greedy, tiene dos pasos. El primero consiste en suplir la demanda, y el segundo se encarga de visitar a los nodos que no fueron tomados en cuenta en la primera parte. En la primera parte del algoritmo comenzamos armando rutas que recolecten leche de tipo A. Una vez satisfecho la demanda de leche de tipo A, comenzamos con otro camion y empezamos a recolectar leche de tipo B, y asi sucesivamente. En caso que el camion se llene, procedemos con otro camion, en el algoritmo esta condicion se traduce a que la lista de candidatos generados sea igual a cero.

La generacion de los candidatos en cada iteracion esta compuesta por todos los nodos de calidad igual o superior a leche elegida, a la vez que el nodo incluido no supere la capacidad del camion, y el nodo no haya sido incorporado anteriormente a solución. Una vez superado estos requerimientos se procede a filtrar la lista segun la distancia, es decir, se genera una nueva lista con los nodos que esten mas cerca al ultimo nodo elegido. El largo de esta lista es definido por un parametro L, el cual por defecto es 10.
Una vez terminado este proceso se elige un candidato de lista generada con un procedimiento probabilístico que favorece a los nodos con mas producción. Este procedimiento esta basado en la ruleta que asigna probabilidad a cada nodo segun la siguiente formula:

$$P_x = \frac{p_x}{\sum_{i=1}^{n} p_i} \tag{2}$$

De esta forma los nodos con mas producción son mas probable que se alijan primero, que los, que no.
Una vez terminada esta etapa, el algoritmo comienza con la etapa de reparación. Esta etapa es muy similar a la anterior, la unica diferencia es que termina solo cuando el camión esta lleno, o se visitaron todos los nodos de cierto tipo de leche.

Algorithm 1: Construccion de la solucion inicial greedy

```
   Result: Solucion factible pseudo-greedy
 1 cola camiones = init_camiones();
 2 verificar(cola de camiones);
 3 while cola de camiones no este vacia do
 4 |   current_camion = cola camion front;
 5 |   leche_elegida = eleccion_greddy();
 6 |   while la demanda de leche elegida no ha sido satisfecha do
 7 |   |   candidatos = generar candidatos(leche elegida, current camion) ;
 8 |   |   if si el largo de la lista de candidatos generados es igual a cero then
 9 |   |   |   break;
10 |   |   end
11 |   |   candidato = random_choose(candidatos);
12 |   |   ruta_camion = ruta_camion + candidato;
13 |   end
14 |   cola de camiones pop ;
15 end
16 Reparar solucion;
17 cola de camiones = init_camiones();
18 while cola de camiones no este vacia do
19 |   current_camion = cola camion front;
20 |   leche_elegida = eleccion_greddy();
21 |   while true do
22 |   |   candidatos = generar candidatos(leche elegida, current camion) ;
23 |   |   if si el largo de la lista de candidatos generados es igual a cero then
24 |   |   |   break;
25 |   |   end
26 |   |   candidato = random_choose(candidatos);
27 |   |   ruta_camion = ruta_camion + candidato;
28 |   end
29 |   cola de camiones pop ;
30 end
```

Una vez terminado el proceso de construcción procedemos con el clásico algoritmo de simulated anneling, la unica diferencia con el algoritmo clasico es la condicion de termino, en este caso son el numero de iteraciones.

Algorithm 2: Simulated anneling

Result: Solucion
Input: Solucion inicial, iteraciones

```
current temp = initial temp;
best_solucion = solucion inicial;
best_quality = evaluate(solucion inicial);
c = 0;
while C ≤ iteraciones do
    seleccionar nuevo punto de la vencidad;
    diferencia = calidad vecino - calidad actual ;
    if diferencia ≥ 0 then
        Moverse al vecino
    end
    else
        aceptar con una probabilidad exp(diferencia/current_temp) > random_number
    end
    current_temp = current_temp * alpha;
    c = c+1;
end
```

9. Experimentos

Los experimentos consistirán en tres etapas. La primera etapa se encarga de obtener experimental la mejor temperatura para las instancias.
La segunda etapa consistirá en obtener experimentalmente el mejor factor de penalización para el factor de bajo cuota y la sobre cuota respectivamente.
Una vez obtenido la temperatura y los factores de penalización que son adecuados para el conjunto de problemas elegidos se comparan los resultados obtenidos por Paredes-Belmar, G., Marianov, V., Bronfman [1] y por [2].

Las instancias elegidas fueron eil22, eil23, eil30, eil76, a36, a44, a55, a64, a80. Los parametros de temperatura, factor de bajo cuota y factor de sobrecuota se modifican directamente en el codigo fuente, específicamente en la funcion main(). Con respecto al parámetro de iteraciones este se pasa como parámetro al ejecutar el programa.
Las instancias fueron ejecutadas en una maquina virtual con 4 nucleos y 6gb de ram. El procesador de la maquina host es un i5 10300h. Se eligo el parametro de iteraciones porque puede ser reproducido en cualquier computador con resultados similares.

10. Resultados

Temperatura variable					
Instancia	N	Z	Z_O	Iter	Temp
eil22	22	15947	15947	30k	500
	22	15947	15902.9	30k	1000
	22	15947	15875	30k	1500
	22	15947	15868	300k	1500
	22	15947	15947.3	300k	2000
	22	15947	15902.7	3000k	2000
	22	15947	15935.6	3000k	2500
eil23	23	7207	7142.98	30k	500
	23	7207	6742.94	30k	1000
	23	7207	6624.55	30k	1500
	23	7207	7087.15	300k	1500
	23	7207	6824.24	300k	2000
	23	7207	7119.59	3000k	2000
	23	7207	6581.64	3000k	2500
eil30	23	7117	7030.17	30k	500
	23	7117	6742.94	30k	1000
	30	7117	6624.55	30k	1500
	30	7117	7087.15	300k	1500
	30	7117	6824.24	300k	2000
	30	7117	6969.38	3000k	2000
	30	7117	6030.32	3000k	2500
eil33	33	20409	20353.7	30k	500
	33	20409	20322.7	30k	1000
	33	20409	20234.3	30k	1500
	33	20409	20359.8	300k	1500
	33	20409	20331.7	300k	2000
	33	20409	20289.7	3000k	2000
	33	20409	20281.9	3000k	2500
eil76	76	91461	91160.9	30k	500
	76	91461	91192.7	30k	1000
	76	91461	91137.2	30k	1500
	76	91461	91242	300k	1500
	76	91461	91231.7	300k	2000
	76	91461	91170.5	3000k	2000
	76	91461	91231.1	3000k	2500
a36	36	29233	29077.4	30k	500
	36	29233	28735.3	30k	1000
	36	29233	29138.7	30k	1500
	36	29233	29032.8	300k	1500
	36	29233	28978.1	300k	2000
	36	29233	28997.3	3000k	2000
	36	29233	28997.3	3000k	2500
a44	a44	38771	38816.4	30k	500
	a44	38771	38465.9	30k	1000
	a44	38771	38594.6	30k	1500
	a44	38771	38526.1	300k	1500
	a44	38771	38573.1	300k	2000
	a44	38771	38556.3	3000k	2000
	a44	38771	38596.9	3000k	2500
a55	55	24694	24314.7	30k	500
	55	24694	24412.1	30k	1000
	55	24694	24315.7	30k	1500
	55	24694	24385.4	300k	1500
	55	24694	24470.1	300k	2000
	55	24694	24459	3000k	2000
	55	24694	24320.3	3000k	2500
a64	64	24100	23388.3	30k	500
	64	24100	23526.6	30k	1000
	64	24100	23586	30k	1500
	64	24100	23378.3	300k	1500
	64	24100	23455.4	300k	2000
	64	24100	23629.7	3000k	2000
	64	24100	23438.8	3000k	2500
a80	80	29977	29576.5	30k	500
	80	29977	29549.4	30k	1000
	80	29977	29422	30k	1500
	80	29977	29471.9	300k	1500
	80	29977	29475.2	300k	2000
	80	29977	29423.1	3000k	2000
	80	29977	29482.2	3000k	2500

factor de penalizacion variable					
Instancia	N	Z	Z_O	FS	FE
eil22	22	15947	15853.2	1	0.1
	22	15947	15874.3	1	0.5
	22	15947	15924	1	1
	22	15947	15917	0.1	1
	22	15947	15932.6	0.5	1
eil23	23	7207	7140.09	1	0.1
	23	7207	6564.3	1	0.5
	23	7207	7156.92	1	1
	23	7207	6816.66	0.1	1
	23	7207	6767.15	0.5	1
eil30	23	7117	6922.26	1	0.1
	30	7117	6864.91	1	0.5
	30	7117	6988.97s	1	1
	30	7117	6824.24	0.1	1
	30	7117	6059.42	0.5	1
eil33	33	20409	20250.4	1	0.1
	33	20409	20292.4	1	0.5
	33	20409	20341	1	1
	33	20409	20329.8	0.1	1
	33	20409	20355.1	0.5	1
eil76	76	91461	91297.2	1	0.1
	76	91461	91152.1	1	0.5
	76	91461	91157.6	1	1
	76	91461	91257.4	0.1	1
	76	91461	91131.9 v	0.5	1
a36	36	29233	29026.2	1	0.1
	36	29233	28947	1	0.5
	36	29233	29064.8	1	1
	36	29233	29124	0.1	1
	36	29233	29066.1	0.5	1
a44	a44	38771	39271.1	1	0.1
	a44	38771	39207.6	1	0.5
	a44	38771	38465	1	1
	a44	38771	38732.4	0.1	1
	a44	38771	38499.1	0.5	1
a55	55	24694	24360.3	1	0.1
	55	24694	24503	1	0.5
	55	24694	24526.4	1	1
	55	24694	24415.3	0.1	1
	55	24694	24373.7	0.5	1
a64	64	24100	23343.7	1	0.1
	64	24100	23606.1	1	0.5
	64	24100	23340.9	1	1
	64	24100	23250.2	0.1	1
	64	24100	23643.9	0.5	1
a80	80	29977	29483.3	1	0.1
	80	29977	29474.2	1	0.5
	80	29977	29615.3	1	1
	80	29977	29553.2	0.1	1
	80	29977	29399.7	0.5	1

En el primer experimentos las variables definidas son las siguientes: N corresponde al tamaño de nodos, Z corresponde al óptimo obtenido por [1], Z_0 corresponde a lo obtenido por nuestro algoritmo, iter corresponde al numero de iteraciones, temp corresponde a la temperatura.

Analizando el experimento nos podemos dar cuenta que obtuvimos mejores resultados con tem-

peraturas bajas, especialmente con la temperatura de 500. Esto puede deberse a que al aumentar la temperatura se favorece mas la exploración que la explotación, ocasionando que se obtengan soluciones peores.

En la segunda tabla FS indica el factor de penalizacion por no cumplir con la demanda y FE es el factor de penalizacion por sobreexcederse con la capacidad de los camiones. En este experimento obtuvimos resultado similares, pero dos factores destacaron el 1,0.1 y el 1,1. Por factores de seguridad elegimos 1,1 para los factores de bajo cuota y sobre cuota respectivamente. Cuando hablamos de "seguridad"nos referimos a que es menos probable que la solución que se genere se infactible. Por ejemplo en al instancia a44 con factor de 1,0.1 obtuvimos una solución infactible debido a la baja penalización del factor sobre cuota.

Una ves obtenidos los mejores parametros para nuestro conjuntos de problemas, los usamos para comparar los resultados obtenidos por la literatura y lo obtenido por nuestro algoritmo. Por cada instancia se ejecuto 5 veces el algoritmo con una temperatura de 500, 30.000 iteraciones, FS=1 Y FE=1. Una vez terminada las ejecuciones se eligió la solución con mejor calidad, y posteriormente se compararon los resultados.

Resultados finales con temp=500 y iter = 30k								
Instancia	N	Z	[S]	Z_H	[S]	Z_G	[S]	Δ %
eil22	22	15947	12	15947	-	15944.2	2.5	0.017 %
eil23	23	7207	6	7207	-	7207	2.5	0
eil30	23	7117	99	7117	-	7099.1	2.5	0.25 %
eil33	33	20409	58	20409	-	20352.8	2.5	0.28 %
eil76	76	91461	1700	91461	-	91224.8	2.5	0.26 %
a36	36	29233	110	29233	-	29118	2.5	0.39 %
a44	44	38771	101	38771	-	38557	2.5	0.56 %
a55	55	24694	270	24694	-	24459.5	2.5	0.94 %
a64	64	24100	5395	24100	164	23454.5	2.5	2.68 %
a80	80	29977	5626	29977	-	29661.4	2.5	1.05 %

En la primera columna tenemos el nombre de la instancia, en la segunda columna tenemos el numero de nodos, en la tercera y cuarta columna tenemos la calidad y el tiempo que obtuvieron [1]. En la cuarta y quinta columna tenemos la calidad y el tiempo que obtuvieron(El signo - significa que obtuvieron una solución en menos de un segundo)[2]. Finalmente en las columnas posteriores tenemos la calidad, los segundos y la variacion porcentual que obtuvimos con nuestro algoritmo.

A pesar de obtener buenos resultados para instancia pequeñas y medianas comparados con [1], no obtuvimos buenos resultado con respecto [2]. Sin embargo en la instancia a64 obtuvimos un resultado peor, pero un tiempo considerable menor. Con esto en mente podemos concluir que el algoritmo de simulated anneling, es muy sensible a la solucion inicial, y si bien, una solucion inicial con un enfoque greddy es una buena solucion, al momento de obtener una mejor solucion se convierte en un problema, pues la exploracion se nos dificulta, debido justamente al enfoque que usamos para generar la solucion inicial. Por esto mismo desde un inicio se considero añadir un comportamiento no determinista para generar la solucion, sin embargo el resultado no fue lo esperado.

11. Conclusiones

En conclusion el enfoque greedy+SA para resolver el problema de recoleccion de leche con mezcla, es buen enfoque para obtener una buena solución en corto tiempo. Sin embargo no se obtuvieron mejores resultados que heurísticas como hill climbing, principalmente debido al enfoque greedy, que dificulta la exploracion. Otra posible causa puede atriburse al movimiento usado. A futuro se propone experimentar con nuevos movimientos, asi como, incorporar la mecanica de restart al SA.

El problema de recolección de leche con mezcla es reciente, por esto mismo, la literatura es escasa. Debido a que el problema es NP, usar tecnicas completas como branch and cut resulta muy costoso para instancias grandes, por este motivo ambos autores proponen usar heurísticas como hill-climbing o una de tres etapas. La primera heurística mencionada es buena para encontrar soluciones de manera rápida y con un bajo costo, pero no garantiza obtener un óptimo global. Técnicas como Iterated local search llegan a solucionar este problema , al añadir pertubacion. Sin embargo, aun asi no se soluciona el problema completamente.
La segunda heurística mencionada es buena para encontrar una solución de buena calidad sin iterar. La calidad de su solución va a depender mucho del procedimiento que se uso para hacer el cluster, no hay una bala de plata para elegir el mejor procedimiento va ha depender mucho del contexto del problema.

Para el futuro, se propone usar algoritmos genéticos, puesto que permiten explorar el espacio de búsqueda de mejor manera, y por ende se espera obtener una solución mejor.

12. Bibliografía

Referencias

[1] Paredes-Belmar, G., Marianov, V., Bronfman, A., Obreque, C., Luer-Villagra, A. (2016) *milk collection problem with blending.* Addison-Wesley, Reading, Massachusetts, 1993. Transportation Research Part E: Logistics and Transportation Review, 94, 2643.

[2] Jorge Andres Villagran Munoz (2019) *acercamientos basados en busqueda local para el problema de recoleccion de leche con mezcla.* Transportation Research Part E: Logistics and Transportation Review, 94, 2643

[3] Kanchana Sethanan,Rapeepan Pitakaso (2015) *Differential evolution algorithms for scheduling raw milk transportation.* Transportation Research Part E: Logistics and Transportation Review, 94, 2643