Algoritmos de Colonia de Hormigas para el Problema del Viajante de Comercio por Familias (FTSP) y para el Problema de Ruteo de Vehículos por Familias (FVRP)

Alexis Soifer
Directora: Irene Loiseau

Departamento de Computación - Universidad de Buenos Aires

alex1879@gmail.com

10/02/2015

Contenido

Contenido

Ejemplo FTSP

- Supongamos que estamos parados en la puerta un almacén que contiene familias de productos (yerba, galletitas, vino, etc.) distribuidos por el local. Suponiendo que tenemos un pedido con distintos productos el problema consiste en encontrar la ruta más corta para recoger todos los productos, y regresar a la puerta.

Ejemplo FTSP

- Supongamos que estamos parados en la puerta un almacén que contiene familias de productos (yerba, galletitas, vino, etc.) distribuidos por el local. Suponiendo que tenemos un pedido con distintos productos el problema consiste en encontrar la ruta más corta para recoger todos los productos, y regresar a la puerta.
- Por ejemplo: 3 paquetes de yerba, 2 de galletitas y 3 de vino.

FTSP

- Sea G un grafo tal que $G = \{V \cup \{v_0\}, E\}$ con $V = \{v_1, ..., v_n\}$ y $E = \{(v_i, v_j) : v_i, v_j \in V \cup \{v_0\}, i < j\}$. El nodo v_0 es el origen del recorrido y el resto de los nodos son los candidatos a recorrer.

FTSP

- Sea G un grafo tal que $G = \{V \cup \{v_0\}, E\}$ con $V = \{v_1, ..., v_n\}$ y $E = \{(v_i, v_j) : v_i, v_j \in V \cup \{v_0\}, i < j\}$. El nodo v_0 es el origen del recorrido y el resto de los nodos son los candidatos a recorrer.
- $K = \{F_1, ..., F_l\}$ una partición del conjunto V donde cada F_l contiene los nodos de la familia l.

FTSP

- Sea G un grafo tal que $G = \{V \cup \{v_0\}, E\}$ con $V = \{v_1, ..., v_n\}$ y $E = \{(v_i, v_j) : v_i, v_j \in V \cup \{v_0\}, i < j\}$. El nodo v_0 es el origen del recorrido y el resto de los nodos son los candidatos a recorrer.
- $K = \{F_1, ..., F_l\}$ una partición del conjunto V donde cada F_l contiene los nodos de la familia l.
- Consideramos a $nf_l = |F_l|$ la cantidad de visitas en la familia l, y a $KN = \sum_{l=1,...,|K|} nv_l$ la cantidad total de visitas en el recorrido.

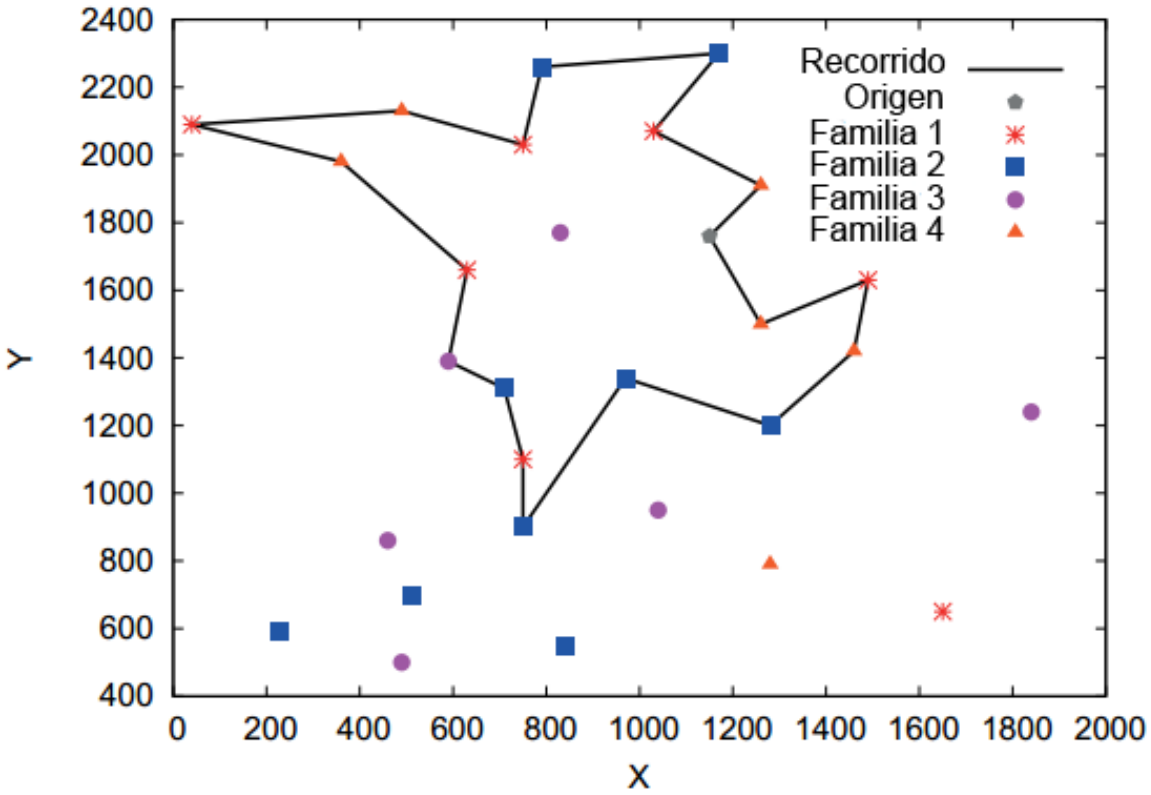
FTSP

- Sea G un grafo tal que $G = \{V \cup \{v_0\}, E\}$ con $V = \{v_1, ..., v_n\}$ y $E = \{(v_i, v_j) : v_i, v_j \in V \cup \{v_0\}, i < j\}$. El nodo v_0 es el origen del recorrido y el resto de los nodos son los candidatos a recorrer.
- $K = \{F_1, ..., F_l\}$ una partición del conjunto V donde cada F_l contiene los nodos de la familia l.
- Consideramos a $nf_l = |F_l|$ la cantidad de visitas en la familia l, y a $KN = \sum_{l=1,...,|K|} nv_l$ la cantidad total de visitas en el recorrido.
- Consideramos a $c(v_i, v_j) > 0, v_i, v_j \in V \cup \{v_0\}$ como la distancia entre cada par de nodos.

Objetivo FTSP

- El objetivo es minimizar la distancia total del recorrido que comience y termine en el origen y recorra nf_l nodos de cada familia $F_l \in K$.

Objetivo FTSP



Ejemplo de una posible solución óptima para un caso donde $|V| = 29, |K| = 4$ y $KN = 18$. Las 18 visitas se obtienen con $nv_1 = 6, nv_2 = 6, nv_3 = 1$ y $nv_4 = 5$.

FVRP

- Este problema generaliza al FTSP.
- A cada nodo le agregamos una demanda.
- Tenemos una flota ilimitada de vehículos con capacidad limitada.

FVRP

- Sea G un grafo tal que $G = \{V \cup \{v_0\}, E\}$ con $V = \{v_1, ..., v_n\}$ y $E = \{(v_i, v_j) : v_i, v_j \in V \cup \{v_0\}, i < j\}$. El nodo v_0 es el origen del recorrido y el resto de los nodos son los candidatos a recorrer, $K = \{F_1, ..., F_l\}$ una partición del conjunto V donde cada F_l contiene los nodos de la familia l y $nf_l = |F_l|$ la cantidad de visitas en la familia l.
- Consideramos a $c(v_i, v_j) > 0, v_i, v_j \in V \cup \{v_0\}$ como la distancia entre cada par de nodos.

FVRP

- Sea G un grafo tal que $G = \{V \cup \{v_0\}, E\}$ con $V = \{v_1, ..., v_n\}$ y $E = \{(v_i, v_j) : v_i, v_j \in V \cup \{v_0\}, i < j\}$. El nodo v_0 es el origen del recorrido y el resto de los nodos son los candidatos a recorrer, $K = \{F_1, ..., F_l\}$ una partición del conjunto V donde cada F_l contiene los nodos de la familia l y $nf_l = |F_l|$ la cantidad de visitas en la familia l.
- Consideramos a $c(v_i, v_j) > 0, v_i, v_j \in V \cup \{v_0\}$ como la distancia entre cada par de nodos.
- Consideramos a $d(v_i), v_i \in V$ como la demanda de cada nodo. Se asume que el depósito tiene demanda 0.

FVRP

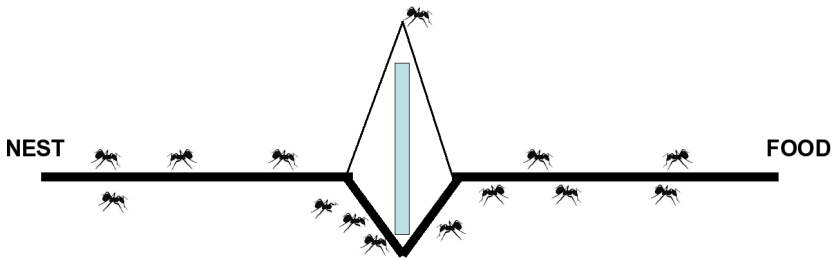
- Sea G un grafo tal que $G = \{V \cup \{v_0\}, E\}$ con $V = \{v_1, ..., v_n\}$ y $E = \{(v_i, v_j) : v_i, v_j \in V \cup \{v_0\}, i < j\}$. El nodo v_0 es el origen del recorrido y el resto de los nodos son los candidatos a recorrer, $K = \{F_1, ..., F_l\}$ una partición del conjunto V donde cada F_l contiene los nodos de la familia l y $nf_l = |F_l|$ la cantidad de visitas en la familia l.
- Consideramos a $c(v_i, v_j) > 0, v_i, v_j \in V \cup \{v_0\}$ como la distancia entre cada par de nodos.
- Consideramos a $d(v_i), v_i \in V$ como la demanda de cada nodo. Se asume que el depósito tiene demanda 0.
- Q es la capacidad de carga de los vehículos, m la cantidad de vehículos utilizados.

Ejemplo FVRP

- Una posible aplicación sería el problema de repartir mercadería en puertos. De cada país tenemos que escoger un subconjunto para recorrer.
- Por ejemplo: 3 puertos de Argentina, 2 de España, 3 de Brasil, etc.
- La solución sería un conjunto de rutas para nuestra flota, que recorran los puertos más convenientes.

Introducción a la metaheurística colonia de hormigas

- Inspirada en el comportamiento de ciertas especies de hormigas.
- Las hormigas se comunican a través de rastros que dejan a su paso.
- Los caminos más cortos hacia el alimento tendrán más rastros que el resto.



Experimento del puente

Movimiento de las hormigas

- Cada hormiga construye una solución completa al problema y en el proceso de construcción tienen en cuenta los rastros de feromona de hormigas anteriores e información heurística.
- La información heurística es información *a priori* y nos ayuda durante la ejecución del algoritmo para influir a la colonia para que converja más rápidamente, depende de cada problema en particular.
- La información de los rastros de feromona es un valor que se va acumulando con el transcurso del tiempo dependiendo de la calidad de los caminos se que encuentren.

Algoritmo básico

- Las hormigas realizan su recorrido.
- Se evapora parte de la feromona.
- Se deposita feromona en los ejes de los recorridos de la(s) mejor(es) hormiga(s).

Esquema clásico

Algoritmo 1: Proceso principal.

while *condición de corte* **do**
    Generación de hormigas y actividad();
    Evaporación de la feromona();
    Depósito de feromona();

Algoritmo básico - variantes

- Las distintas variantes de esta metaheurística se diferencian principalmente por cuando y quienes realizan el depósito de la feromona.

Sistema mejor-peor hormiga

- El movimiento de las hormigas y la evaporación de la feromona se conserva del algoritmo clásico de OCH.
- Feromona: se agrega en las aristas del mejor recorrido encontrado hasta el momento, y se resta del peor.
- Acciones extra: se realiza búsqueda local, y mutación. La mutación consiste en modificar el valor de la feromona al azar en distintas aristas.

Contenido

Esquema básico OCH - SMPH

Algoritmo 2: Proceso principal.

```
Inicializar información heurística();
while condición de reinicio do
    Reiniciar feromona();
    while condición de corte do
        Generación de hormigas y actividad();      ▷ Algoritmo 3
        Evaporación de la feromona();              ▷ Algoritmo 4
        Acciones del demonio();                    ▷ Algoritmo 5
```

Movimiento de las hormigas

- La hormiga considera candidatos a aquellos nodos que no estén visitados y que se encuentren en una familia que aún tenga nodos por visitar.

Movimiento de las hormigas

- La hormiga considera candidatos a aquellos nodos que no estén visitados y que se encuentren en una familia que aún tenga nodos por visitar.
- Una de las modificaciones realizadas en esta implementación consiste en que la hormiga se mueva dentro de un vecindario restringido. Este consiste el 25 % de los nodos más cercanos a cada nodo.

Movimiento de las hormigas

- La hormiga considera candidatos a aquellos nodos que no estén visitados y que se encuentren en una familia que aún tenga nodos por visitar.
- Una de las modificaciones realizadas en esta implementación consiste en que la hormiga se mueva dentro de un vecindario restringido. Este consiste el 25 % de los nodos más cercanos a cada nodo.
- Si no tenemos ningún candidato vecino para visitar, realizamos el movimiento según el algoritmo clásico de OCH.

Movimiento de las hormigas

Algoritmo 3: Generación de hormigas y actividad.

```
for k = 0 to cantidad de hormigas a generar do
    Inicializar hormiga();
    while resten movimientos por realizar do
        v_i = nodo actual;
        forall the v_j ∈ candidatos vecinos(v_i) do
            Calcular la probabilidad de recorrer v_j();
        if No hay candidatos vecinos de v_i then
            forall the v_j ∈ candidatos(v_i) do
                Calcular la probabilidad de recorrer v_j();
        seleccionamos el siguiente nodo según su probabilidad de
        ser visitado y movemos la hormiga();
```

Evaporación

Algoritmo 4: Evaporación de la feromona.

forall the $e \in E$ **do**
    feromonas(e) = feromonas(e) * (1 - factor de evaporación);

Acciones extra

- Búsqueda local: consiste en partir de una solución inicial y aplicar reiteradamente alguna transformación hasta que deje de haber mejoras.
- Aplican a una única ruta (FTSP, y FVRP) o a múltiples rutas (FVRP).
- En esta implementación utilizamos varias transformaciones y el costo computacional de cada búsqueda es elevado en comparación con el resto del algoritmo.
- Por eso la aplicamos en el 50 % de las hormigas con mejores soluciones.

Acciones extra

- Búsqueda local: consiste en partir de una solución inicial y aplicar reiteradamente alguna transformación hasta que deje de haber mejoras.
- Aplican a una única ruta (FTSP, y FVRP) o a múltiples rutas (FVRP).
- En esta implementación utilizamos varias transformaciones y el costo computacional de cada búsqueda es elevado en comparación con el resto del algoritmo.
- Por eso la aplicamos en el 50 % de las hormigas con mejores soluciones.

- Mutación: consiste en modificar el valor de la feromona al azar en distintos ejes del grafo.
- También se aplica en el vecindario restringido.

Acciones extra

Algoritmo 5: Acciones del demonio.

forall the *hormiga entre las mejores hormigas* **do**
    Búsqueda Local (hormiga);

Actualizar mejor y peor hormiga hasta el momento();
Depositar feromona extra en el mejor camino, y retirar del peor();
Realizar mutación();

Mutación

- El valor de la feromona a depositar depende del promedio del valor depositado en el mejor camino.
- El porcentaje de aristas a modificar depende de la cantidad de iteraciones sin que se encuentren mejoras.
- La feromona se puede sumar o restar aleatoriamente a cada arista.

FVRP

- El algoritmo es similar al del FTSP.
- Cada hormiga representa una solución al problema, y determina los recorridos de todos los vehículos.
- Cuando un vehículo completa su capacidad, se agrega uno nuevo.

FVRP

- El algoritmo es similar al del FTSP.
- Cada hormiga representa una solución al problema, y determina los recorridos de todos los vehículos.
- Cuando un vehículo completa su capacidad, se agrega uno nuevo.
- Una mejora implementada es que se le permite regresar prematuramente al depósito a los vehículos que se encuentren cerca del mismo y sin vecinos disponibles.

FVRP

- El algoritmo es similar al del FTSP.
- Cada hormiga representa una solución al problema, y determina los recorridos de todos los vehículos.
- Cuando un vehículo completa su capacidad, se agrega uno nuevo.
- Una mejora implementada es que se le permite regresar prematuramente al depósito a los vehículos que se encuentren cerca del mismo y sin vecinos disponibles.
- La búsqueda local contiene transformaciones que aplican a varios recorridos simultáneamente.

Contenido

Resultados FTSP

- Morán-Mirabal, Gonzalez-Velarde, y Resende presentaron la formulación para el problema.
- Lo resolvieron utilizando las metaheurísticas GRASP, y BRKGA (una variante de los algoritmos genéticos).

Resultados FTSP

- Morán-Mirabal, Gonzalez-Velarde, y Resende presentaron la formulación para el problema.
- Lo resolvieron utilizando las metaheurísticas GRASP, y BRKGA (una variante de los algoritmos genéticos).

- Publicaron los resultados óptimos hasta las instancias de 48 nodos y para el resto utilizaron heurísticas para determinar un valor con el cual comparar la eficacia de sus algoritmos.
- Utilizaremos sus mejores valores publicados para comparar los resultados del algoritmo OCH-SMPH.

Resultados FTSP

Instancia	Obj.	BRKGA	GRASP+evPR	OCH-SMPH	Dif.
burma14_3_1001_1001	13,93(*)	13,93	13,93	13,93 %	0,00 %
burma14_3_1001_1002	25,66(*)	25,66	25,66	25,66 %	0,00 %
burma14_3_1001_1003	11,89(*)	11,89	11,89	11,89 %	0,00 %
bayg29_4_1001_1001	5345,86(*)	5345,86	5345,86	5345,86	0,00 %
bayg29_4_1001_1002	5791,01(*)	5791,01	5791,01	5791,01	0,00 %
bayg29_4_1001_1003	5544,33(*)	5544,33	5544,33	5544,33	0,00 %
att48_5_1001_1001	23686,02(*)	**23686,02**	**23686,02**	23834,13	0,63 %
att48_5_1001_1002	20609,09(*)	**20609,09**	20635,57	20679,09	0,34 %
att48_5_1001_1003	9024,58(*)	9024,58	9024,58	9024,58	0,00 %
bier127_10_1001_1001	36800,39	36913,74	36800,39	**34500,98**	-6,25 %
bier127_10_1001_1002	97615,41	98216,1	97615,41	**92135,29**	-5,61 %
bier127_10_1001_1003	50513,10	50513,1	50715,49	**48104,21**	-4,77 %
a280_20_1001_1001	1891,16	2126,34	1891,16	**1807,16**	-4,44 %
a280_20_1001_1002	1697,48	1925,28	1697,48	**1616,53**	-4,77 %
a280_20_1001_1003	1597,25	1720,23	1597,25	**1479,74**	-7,36 %
gr666_30_1001_1001	1817,06	2625,69	1817,06	**1670,47**	-8,07 %
gr666_30_1001_1002	1443,05	2275,8	1443,05	**1370,13**	-5,05 %
gr666_30_1001_1003	1358,12	2426,59	1384,18	**1292,71**	-4,82 %
pr1002_40_1001_1001	163461,79	421061,63	163461,79	**149774,43**	-8,37 %
pr1002_40_1001_1002	182144,13	421761	182144,13	**162302,5**	-10,89 %
pr1002_40_1001_1003	149456,63	284856,22	149456,63	**138031,59**	-7,64 %

La columna objetivo corresponde a los mejores valores publicados.
(*) valores óptimos

GVRP

- Es un caso particular de FVRP donde solo hay que recorrer 1 cliente de cada conjunto.
- Como en FVRP no disponemos de instancias conocidas, utilizamos este problema para comparar el algoritmo de FVRP con resultados existentes.

GVRP

- Es un caso particular de FVRP donde solo hay que recorrer 1 cliente de cada conjunto.
- Como en FVRP no disponemos de instancias conocidas, utilizamos este problema para comparar el algoritmo de FVRP con resultados existentes.
- Bektas, Erdogan, Ropke generaron distintas clases de instancias y las resolvieron con un algoritmo de branch and cut, y la metaheurística LNS.
- Ha, Bostel, Langevin y Rousseau resolvieron las mismas instancias con una metaheurística híbrida basada en GRASP y ELS.

Resultados GVRP

En esta tabla se muestra en cuales de las instancias publicadas previamente se alcanza el mejor valor conocido.

Tipo de instancia	OHC-SMPH alcanza el mejor valor publicado
A	54/54
B	46/46
P	48/48
M	4/8
G	0/2

Resultados GVRP

En esta tabla se muestra en cuales de las instancias publicadas previamente se alcanza el mejor valor conocido.

Tipo de instancia	OHC-SMPH alcanza el mejor valor publicado
A	54/54
B	46/46
P	48/48
M	4/8
G	0/2

Detalle de los resultados para las instancias de tipo G

	Bektas et. al.			Ha et. al.				
Nombre de instancia	m_1	LNS	Branch & Cut	m_2	Metaheurística	m_3	ACO-SMPH	Dif.
G-n262-k25-C131	12	3249	-	13	3303	13	3458	6,43 %
G-n262-k25-C88	9	2476	-	9	2477	9	2560	3,39 %

Generación de instancias para FVRP

- Se utilizaron como base las mismas instancias que en FTSP, agregando una demanda para cada cliente.
- Para comparar los resultados se utiliza como objetivo el valor obtenido por ejecuciones de 1:30hs. de duración.

Resultados FVRP

Instancia	Obj.	CH-SMPH	Dif.
burma14_3_1001_1001_2	16,35	16,35	0,00 %
burma14_3_1001_1002_2	35,71	35,71	0,00 %
burma14_3_1001_1003_2	11,04	11,04	0,00 %
bayg29_4_1001_1001_2	9526,61	9526,61	0,00 %
bayg29_4_1001_1002_2	13127,05	13128,05	0,01 %
bayg29_4_1001_1003_2	10100,44	10100,44	0,00 %
att48_5_1001_1001_2	66871,44	66871,44	0,00 %
att48_5_1001_1002_2	56325,95	56325,95	0,00 %
att48_5_1001_1003_2	17127,95	17127,95	0,00 %
bier127_10_1001_1001_2	100878,28	100779,87	-0,10 %
bier127_10_1001_1002_2	217376,55	216421,97	-0,44 %
bier127_10_1001_1003_2	137983,46	138072,24	0,06 %
a280_20_1001_1001_2	15707,51	15750,98	0,28 %
a280_20_1001_1002_2	12912,01	13093,27	1,40 %
a280_20_1001_1003_2	11867,62	12022,72	1,31 %
gr666_30_1001_1001_2	14847,04	15039,83	1,30 %
gr666_30_1001_1002_2	13333,87	13300,78	-0,25 %
gr666_30_1001_1003_2	13309,33	13273,73	-0,27 %
pr1002_40_1001_1001_2	2060679,36	2061861,39	0,06 %
pr1002_40_1001_1002_2	2936641,74	2952446,41	0,54 %
pr1002_40_1001_1003_2	2198170,55	2189537,48	-0,39 %

Contenido

Conclusiones y trabajo futuro

- Utilizamos una variante de colonia de hormigas para resolver FTSP, GVRP y FVRP. Evaluamos cada característica del algoritmo y tratamos de explotarla al máximo.
- En el caso de FTSP superamos la mayor parte de los resultados conocidos hasta el momento.
- En GVRP nuestro algoritmo demostró buenos resultados alcanzando los mejores valores conocidos en la mayoría de las instancias chicas y medianas. Por otro lado en las grandes instancias nuestro algoritmo no superó el 6,50 % de diferencia con el mejor valor.

Conclusiones y trabajo futuro

- Presentamos una formulación matemática para el FVRP.
- Una mejora al algoritmo podría ser la utilización de otra heurística para determinar el nro. de vehículos a utilizar.
- Una posibilidad para continuar investigando sería la de no establecer la cantidad de visitas por familia a priori. La única restricción podría ser la de cumplir con la demanda de la familia y que la cantidad de visitas a la misma surja en la resolución del problema.

Gracias!