PRÁCTICA 1
BIOINFORMÁTICA

Algoritmos de Optimización Basados en Colonias de Hormigas

2011/2012

Asier Erramuzpe Aliaga
44601460-B
erramuzpe@correo.ugr.es
Grupo A: Oscar Cordón. Lunes 10-12.

ÍNDICE

2. Descripción del problema.

El objetivo de esta práctica es estudiar el funcionamiento de los Algoritmos de Optimización Basados en Colonias de Hormigas (OCH)1 . Para ello, se utilizarán distintas variantes de estos algoritmos, correspondientes al Sistema de Hormigas, el Sistema de Colonias de Hormigas, Sistema de Hormigas Max-Min y Sistema de Hormigas Mejor-Peor, para resolver seis instancias del Viajante de Comercio con y sin búsqueda local.

3. Descripción de los algoritmos considerados.

Una de las áreas de estudio más importante y prometedora de los últimos años es la investigación de algoritmos basados en comportamientos observados en la Naturaleza . En concreto, nosotros vamos a utilizar algoritmos basados en colonias de hormigas (ACO), para resolver el problema del viajante. Los algoritmos utilizados son los siguientes:

a) El Sistema de Hormigas.

En el algoritmo AS, cada hormiga genera una ruta completa seleccionando las ciudades de acuerdo a una regla probabilística de transición entre estados (las hormigas prefieren moverse hacia ciudades más cercanas y con un alto contenido de feromonas en su camino). Esta regla de transición presenta unos parámetros configurables que determinan la importancia relativa de la feromona (rastro) frente a la longitud del trayecto (visibilidad). Posteriormente se aplica una regla de actualización global de feromonas: se evapora una fracción de la feromona de los caminos no recorridos y cada hormiga deposita una cierta cantidad de la misma en los caminos pertenecientes a su ruta, en proporción a lo corta que ésta resultara. Finalmente se itera el mismo proceso.

b) El Sistema de Colonias de Hormigas.

Dorigo y Gambardella (1997) desarrollaron, basándose en AS y con el objetivo de mejorar su eficiencia, el algoritmo ACS (Ant Colony System), cuya estructura ya presenta un cambio más sustancial con respecto al original. ACS difiere de AS en tres aspectos principales: i) la regla de transición entre estados proporciona una forma directa de balanceo entre exploración de nuevos caminos y explotación del conocimiento acumulado a priori; ii) la regla de actualización global es únicamente aplicada a los caminos pertenecientes a la mejor ruta y iii) mientras las hormigas construyen una solución se aplica una regla de actualización local de feromona. El algoritmo comienza con el posicionamiento de m hormigas en n ciudades elegidas de acuerdo a alguna regla de inicialización (por ejemplo de forma aleatoria). Cada hormiga construye una ruta aplicando la regla de transició entre estados. A medida que construye su ruta, la hormiga actualiza la cantidad de feromona en los caminos recorridos aplicando la regla de actualización local. Una vez que todas las hormigas han terminado su ruta, la cantidad de feromona en los caminos es modificada de nuevo aplicando la regla de actualización global.

c) El Sistema de Hormigas Max-Min.

Stüzle y Hoos (2000) propusieron el algoritmo MMAS (MAX-MIN Ant System) como una mejora del AS inicial. Sus características principales son que únicamente la mejor hormiga actualiza el rastro de feromonas y que el valor de la feromona está acotado, tanto superior como inferiormente. Estas cotas son típicamente obtenidas de forma empírica y ajustadas según el problema específico considerado.

d) El Sistema de Hormigas Mejor-Peor.

El Sistema de Hormigas Max-Min (Max-Min Ant System) es una nueva extensión del SH con una mayor explotación de las mejores soluciones y un mecanismo adicional para evitar el estancamiento de la búsqueda . Mantiene la regla de transición del SH y cambia el mecanismo de actualización (que es más agresivo, al evaporar todos los rastros y aportar sólo en los de la mejor solución). También define unos topes mínimo y máximo para los rastros de feromona y reinicializa la búsqueda cuando se estanca .

e) Uso de la búsqueda local.

Existe la posibilidad de hibridar los algoritmos de OCH con técnicas de búsqueda local para mejorar su eficacia . La búsqueda local es el mecanismo que se emplea para mejorar la calidad de las soluciones. Se busca entre X vecinos tratando de acotar la optimización local. Si se empezara a buscar entre todos los vecinos, la optimización global de las hormigas no nos serviría de mucho. La hibridación consiste en aplicar una búsqueda local sobre las soluciones construidas por todas las hormigas en cada iteración antes de actualizar la feromona. El aumento en la eficacia reduce la eficiencia por eso habitual emplear la búsqueda local junto con las llamadas Listas de Candidatos, que consisten en estudiar sólo las ciudades candidatas más prometedoras en cada paso de la hormiga . Al usar búsqueda local se obtienen muy buenos resultados con muchas menos hormigas que en la OCH básica (lo que implica mayor rapidez al hacer menos búsquedas locales) .

4. Experimentos y análisis de resultados.

a) Resultados obtenidos

Las tablas de resultados para cada uno de los 4 algoritmos analizados y sus respectivas búsquedas locales están en el “Apéndice 1”. En los distintos algoritmos con búsqueda local, las ejecuciones daban un mal comportamiento, parándose por “segmentation fault” o entrando en un bucle infinito por tener dos posiciones del mismo valor (mala implementación del algoritmo). Para estos casos, se ha usado ACOTSP-1.02 y se consiguen resolver todos los problemas.

A continuación se muestran las tablas globales para el análisis detallado:

Modelo	eil76 (538) Med	Mej	desv	kroA100 (21282) Med	Mej	desv	d198 (15780) Med	Mej	desv
SH	548.8	548	1.64	22214.6	22082	82.32	17011.4	16931	77.96
SCH	539.8	538	2.22	21318.6	21828	39.71	16034.8	15891	129.76
SHMM	538	538	0	21282	21282	0	15974	15954	15
SHMP	544.2	538	3.76	21627.4	21282	366.48	16203	16409.6	186.91
SH-BL	538	538	0	21282	21282	0	15780	15780	0
SCH-BL	538	538	0	21282	21282	0	15780	15780	0
SHMM-BL	538	538	0	21282	21282	0	15780	15780	0
SHMP-BL	538	538	0	21282	21282	0	15780	15780	0

Modelo	lin318 (42029) Med	Mej	desv	att532 (27686) Med	Mej	desv	rat783 (8806) Med	Mej	desv
SH	47424.6	47324	88.71	34184.4	33893	192.95	11273.6	11046	134.25
SCH	42669.4	42316	195.68	28465.8	28377	102.69	9146	9107	40.04
SHMM	42849	42677	101.67	28312	28182	80.24	9125	9080	36.99
SHMP	43383.8	42989	537.54	29414.8	28866	410.95	9074.8	9039	20.56
SH-BL	42096.4	42671	23.26	27864	27851	18.06	8939.4	8929	8.77
SCH-BL	42029	42029	0	27696.4	27686	13.48	8806	8806	0
SHMM-BL	42029	42029	0	27686	27686	0	8806	8806	0
SHMP-BL	42029	42029	0	27698.6	27686	11.18	8809.8	8806	7.6

Se han marcado con colores los resultados para una lectura más clara. En verde los mejores resultados para cada batería de datos o datos muy buenos, aunque no consigan ser los mejores. Los resultados marcados en rojo son resultados negativos muy por encima del error medio, me ha parecido significativo resaltarlo. El único dato marcado en amarillo se analizará más tarde.

Se pueden sacar dos conclusiones claras:

- Los sistemas con búsqueda local son mucho más eficientes que los que no la usan y su rapidez de convergencia hacia el resultado óptimo es mucho mayor. Pongamos como ejemplo SHMM con los datos eil76. En los dos se encuentra el óptimo, pero mientras que el SHMM necesita de una media de 4-5 minutos para encontrarlo, el SHMM-BL lo encuentra usando apenas 20-30 segundos. Ocurre lo mismo en todos los algoritmos.
- El algoritmo SHMM y SHMM-BL son los mejores, cada uno en sus respectivos grupos. No sólo encuentran la mejor solución (con casi todos los conjuntos excepto en 2, por la mínima), además son los que obtienen desviaciones menores. Este último aspecto es importante, acentúa que el SHMM es un algoritmo robusto sus resultados están muy balanceados, siempre en torno a la misma solución y no obtiene "picos" en sus resultados. Pasemos a analizar uno por uno los algoritmos:

SH es el peor algoritmo con diferencia. No consigue resolver ni un sólo problema y además, se queda lejos de los otros algoritmos de su misma categoría tanto en resultados como en la desviación (en ocasiones, demasiado alta). Puede que SCH o SHMP no sean tan buenos como SHMM, pero no se quedan tan lejos como lo hace SH, que claramente es un algoritmo muy pobre.

SH-BL no es tan malo como su implementación sin búsqueda local, resuelve correctamente los primeros 3 problemas y se queda realmente cerca y con una desviación muy baja en los 3 últimos. SH-BL no es tan pobre como SH, aunque tiene carencias respecto de los demás -BL, no son tan significativas como las que tiene SH con los de su misma categoría.

SCH es un buen algoritmo, no consigue resolver ningún problema, pero consigue en ciertas ejecuciones el óptimo para el primer y el segundo problema. También tiene mérito el conseguir el mejor resultado para el cuarto problema, por encima de SHMM. La explicación del dato marcado en amarillo es que pese a que SCH logra un mejor resultado que SHMM en este problema, la desviación de SCH es mayor. No podemos asegurar que siempre vaya a resolver mejor que SHMM en ejecuciones aisladas.

SCH-BL resuelve todos los problemas menos el quinto (att532), pero se queda muy cerca con una desviación bajísima. Si miramos la tabla del apéndice, se aprecia que ha resuelto correctamente 3/5 ejecuciones y las dos no resueltas se a quedado realmente cerca. SCH con búsqueda local es un muy buen algoritmo. En las ejecuciones también se aprecia que no converge tan rápido como SHMM, pero sigue siendo muy válido para resolver problemas.

SHMM es el mejor algoritmo de todos los analizados. El más rápido en converger, el más robusto (mínima desviación, incluso en los 2 casos que no gana de las 8 ejecuciones). De todas las ejecuciones sólo en 1 no consigue la desviación menor. En el sexto problema, que le gana SHMP.

SHMM-BL es el único que resuelve todos los problemas. Además lo hace a una velocidad más alta que sus competidores, utiliza entre 5 segundos (en los mejores casos) y 90 en los peores.

De SHMP podemos decir muchas cosas; pese a ser un muy buen algoritmo, parecido a SCH, sorprende su pésima desviación en ciertos problemas. En el segundo, en alguna ejecución logra el óptimo global, en otras se dispara la solución y tiene problemas para converger correctamente. Le pasa lo mismo con los problemas 4 y 5, logrando resultados incluso peores que SH. Pese a este problema, por sorpresa, en el problema 6 barre a sus competidores y logra no sólo el mejor resultado, si no que también gana a SHMM en robustez de resultados, cuando su punto débil en los anteriores problemas era precisamente la gran desviación. Es el único algoritmo que logra vencer a SHMM. Podemos decir que es un buen algoritmo, pero su falta de robustez le hace no estar a la altura de SHMM. Tal vez se mejorara la convergencia si además de reforzar la mejor solución, penalizáramos también la peor. Planteé esto en clase, pero no he sido capaz de implementarlo para

poder comprobarlo.

SHMP-BL resuelve todos los problemas menos el quinto (att532) y el sexto (rat783), pero se queda muy cerca con una desviación bajísima (al igual que SCH-BL). Si miramos la tabla del apéndice, se aprecia que ha resuelto correctamente 1 ejecución del att532, quedándose muy cerca en las no resueltas. Y resuelve perfectamente todas menos una de las ejecuciones de rat783. Eso sí, converge bastante más lento que SHMM. Con búsqueda local, no tiene problemas para resolver los problemas, pero siguen latentes su problema de convergencia, reflejado en el tiempo necesario para encontrar las soluciones.

5. Bibliografía.

- M. Dorigo & T. Stützle, Ant Colony Optimization, MIT Press, 2004

- M. Dorigo, T. Stützle, Ant Colony Optimization: Overview and Recent Advances. M. Gendreau and Y. Potvin, editors, Handbook of Metaheuristics, 2nd edition. Vol. 146 in International Series in Operations Research & Management Science, pp. 227--263. Springer, Verlag, New York, 2010.

- O. Cordon, F. Herrera, and T. Stützle, A Review on the Ant Colony Optimization Metaheuristic: Basis, Models and New Trends. Mathware and Soft Computing, 9(2-3), pp. 141--175, 2002. MSC-Intro-prel.ps.gz

- ”Estado del arte de algoritmos basados en colonias de hormigas”. David de la Fuente, Jesús Lozano, Eva Ochoa de Olano, Magín Díaz, XV Congreso de Ingeniería de Organización Cartagena, 7 a 9 de Septiembre de 2011 .

APENDICE 1 : Tablas de resultados.

SH	eil76	kroA100	d198	lin318	att532	rat783
Ejecución 1	548	22082	17097	47457	34348	11437
Ejecución 2	548	22264	17114	47324	34013	11377
Ejecución 3	550	22220	16931	47353	34347	11263
Ejecución 4	551	22327	16969	47413	34321	11046
Ejecución 5	547	22180	16946	47576	33893	11245,
Media	548.8	22214.6	17011.4	47424.6	34184.4	11273.6
Desv. Típ. (n) Desv. Típ. (n-1)	1.46 1.64	82.32 92.04	77.96 87.17	88.71 99.19	192.95 215.72	134.25 150.1

Tabla 1.1: Resultados del algoritmo SH en los distintos casos del problema

SH-BL	eil76	kroA100	d198	lin318	att532	rat783
Ejecución 1	538	21282	15780	42126	27851	8938
Ejecución 2	538	21282	15780	42080	27851	8929
Ejecución 3	538	21282	15780	42071	27853	8931
Ejecución 4	538	21282	15780	42082	27868	8949
Ejecución 5	538	21282	15780	42123	27898	8950
Media	538	21282	15780	42096.4	27864	8939.4
Desv. Típ. (n) Desv. Típ. (n-1)	0 0	0 0	0 0	23.26 26.01	18.06 20.19	8.77 9.81

Tabla 1.2: Resultados del algoritmo SH-BL en los distintos casos del problema

SCH	eil76	kroA100	d198	lin318	att532	rat783
Ejecución 1	543	21282	16181	42316	28400	9114
Ejecución 2	538	21305	15939	42740	28469	9146
Ejecución 3	538	21305	16201	42653	28662	9147
Ejecución 4	542	21396	15891	42910	28377	9220
Ejecución 5	538	21305	15962	42728	28421	9107
Media	539.8	21318.6	16034.8	42669.4	28465.8	9146
Desv. Típ. (n) Desv. Típ. (n-1)	2.22 2.49	39.71 44.4	129.73 145.05	195.68 218.78	102.69 114.81	40.04 44.77

Tabla 2.1: Resultados del algoritmo SCH en los distintos casos del problema

SCH-BL	eil76	kroA100	d198	lin318	att532	rat783
Ejecución 1	538	21282	15780	42029	27686	8806
Ejecución 2	538	21282	15780	42029	27686	8806
Ejecución 3	538	21282	15780	42029	27719	8806
Ejecución 4	538	21282	15780	42029	27705	8806
Ejecución 5	538	21282	15780	42029	27686	8806
Media	538	21282	15780	42029	27696.4	8806
Desv. Típ. (n) Desv. Típ. (n-1)	0 0	0 0	0 0	0 0	13.48 15.08	0 0

Tabla 2.2: Resultados del algoritmo SCH-BL en los distintos casos del problema

SHMM	eil76	kroA100	d198	lin318	att532	rat783
Ejecución 1	538	21282	15970	42677	28430	9085
Ejecución 2	538	21282	15954	42796	28182	9080
Ejecución 3	538	21282	15967	42956	28296	9141
Ejecución 4	538	21282	15980	42891	28307	9144
Ejecución 5	538	21282	15999	42927	28345	9176
Media	538	21282	15974	42849	28312	9125
Desv. Típ. (n) Desv. Típ. (n-1)	0 0	0 0	15 16.78	101.67 113.68	80.24 89.71	36.99 41.36

Tabla 3.1: Resultados del algoritmo SHMM en los distintos casos del problema

SHMM-BL	eil76	kroA100	d198	lin318	att532	rat783
Ejecución 1	538	21282	15780	42029	27686	8806
Ejecución 2	538	21282	15780	42029	27686	8806
Ejecución 3	538	21282	15780	42029	27686	8806
Ejecución 4	538	21282	15780	42029	27686	8806
Ejecución 5	538	21282	15780	42029	27686	8806
Media	538	21282	15780	42029	27686	8806
Desv. Típ. (n) Desv. Típ. (n-1)	0 0	0 0	0 0	0 0	0 0	0 0

Tabla 3.2: Resultados del algoritmo SHMM-BL en los distintos casos del problema

SHMP	eil76	kroA100	d198	lin318	att532	rat783
Ejecución 1	544	21379	16256	42989	30084	9098
Ejecución 2	538	21438	16736	43106	29275	9079
Ejecución 3	543	21747	16203	43033	29619	9068
Ejecución 4	547	21282	16413	44427	29230	9090
Ejecución 5	549	22291	16440	43364	28866	9039
Media	544.2	21627.4	16409.6	43383.8	29414.8	9074.8
Desv. Típ. (n) Desv. Típ. (n-1)	3.76 4.21	366.48 409.74	186.41 208.42	537.54 600.99	410.95 459.46	20.56 22.99

Tabla 4.1: Resultados del algoritmo SHMP en los distintos casos del problema

SHMP-BL	eil76	kroA100	d198	lin318	att532	rat783
Ejecución 1	538	21282	15780	42029	27715	8806
Ejecución 2	538	21282	15780	42029	27686	8825
Ejecución 3	538	21282	15780	42029	27686	8806
Ejecución 4	538	21282	15780	42029	27703	8806
Ejecución 5	538	21282	15780	42029	27703	8806
Media	538	21282	15780	42029	27698.6	8809.8
Desv. Típ. (n) Desv. Típ. (n-1)	0 0	0 0	0 0	0 0	11.18 12.5	7.6 8.5

Tabla 4.2: Resultados del algoritmo SHMP-BL en los distintos casos del problema

FIN