PRACTICA 1

Jesús Campos Márquez

2018/2019

MODELOS DE BUSQUEDA Y HEURISTICAS DE BUSQUEDA

Universidad de Huelva

MB Y HB

Índice

Introducción a los algoritmos de búsqueda

Los algoritmos de búsqueda son aquellos que están diseñados para localizar un elemento con unas propiedades dentro de una estructura de datos, por ejemplo, en el problema del viajante, sería buscar aquellos caminos más cortos que resuelvan el problema con la mejor solución posible.

Para resolver problemas complejos se utilizan algoritmos aproximados, que proporcionan buenas soluciones, aunque no necesariamente la óptima, en un tiempo razonable.

En los algoritmos aproximados, encontramos dos tipos:

- Heurísticas constructivas que parten de una solución inicial vacía, y van añadiendo componentes hasta encontrar una solución, como es el caso del Greedy.

- Búsquedas locales, que parten de una solución inicial que podría ser una aleatoria, y que iterativamente intentan reemplazarla por otra solución mejor dentro de su vecindario. El problema que presentan es que pueden caer fácilmente en óptimos locales, que están alejados de un óptimo global, por lo que puede dar soluciones peores que una búsqueda Greedy.

Figura 1.1

Para logar encontrar solución óptima a problemas muy grandes se utilizan las heurísticas, que surgen como un intento de producir una sinergia, para que haciendo uso de una combinación de métodos podamos crear nuevos algoritmos a partir de conceptos derivados de la inteligencia artificial, de la evolución biológica y de los mecanismos estadísticos.

Definición teórica de los algoritmos

- Algoritmo de Búsqueda Aleatoria
  Es la más simple de todas, y genera aleatoriamente una solución en cada iteración usando una lista de nodos no visitados para evitar que se repitan.

- Búsqueda Greedy
  Sigue la heurística del vecino más cercano, es decir, que partiendo de una ciudad que podemos considerar aleatoria o fija(aunque siempre dará el mismo resultado), la elimina del conjunto de nodos y escoge aquella a menor distancia, repitiéndose el proceso hasta que no queden más nodos por visitar.

- Búsquedas Locales
  Nos encontramos con dos tipos a implementar:
  - Búsqueda del primer mejor vecino. Parte de una solución inicial aleatoria. Va generando paso a paso el entorno de la solución actual hasta que se obtiene una solución vecina que mejora al actual. Si la solución vecina mejora al actual el algoritmo finaliza.
  - Búsqueda del mejor de todos. A partir de una solución inicial aleatoria, genera todo el entorno completo de ella y selecciona la mejor solución vecina, y comprueba si esa solución es mejor que la propia actual en el caso que lo sea, entonces la sustituye y vuelve a iterar, finalizando el algoritmo dada una convergencia propuesta.

Ambos utilizan el operador de generación de nuevos vecinos 2-opt, que escoge dos posiciones y las intercambia.

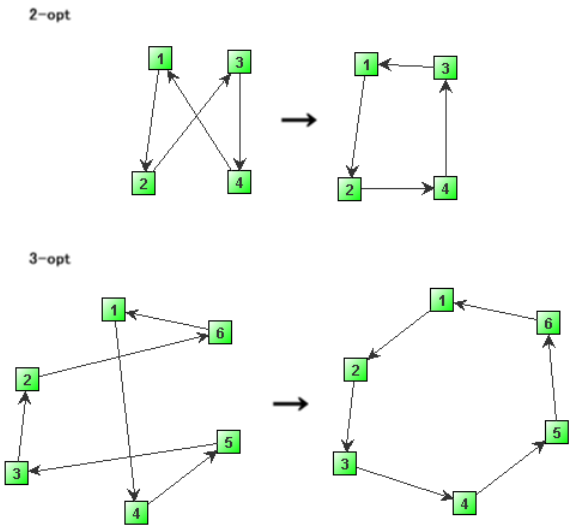

Figura 2.1

- Enfriamiento Simulado
  Es un método de Búsqueda Global, que permite algunos movimientos peores para escapar de óptimos locales. Para escapar de estos óptimos locales se debe controlar, para ello se implementa una función de probabilidad que hará disminuir la probabilidad de estos movimientos hacia soluciones peores conforme avanza la búsqueda.

  Cada iteración genera un número concreto de vecinos que puede ser fijo o depender de una iteración, y que cada vez que se genera un vecino se aplica un criterio de aceptación a ver si sustituye la solución actual, que consiste en una condición en la que sí es mejor que la actual se acepta, pero en el caso de ser peor tiene una probabilidad de que el vecino sustituya a la solución actual, donde la probabilidad depende de la diferencia de costes entra la solución actual y la vecino a comprobar, y de la temperatura.

  Utiliza el operador de generación de vecinos 2-opt (Figura 2.1).

- Búsqueda Tabú
  Es una búsqueda por entornos que utiliza memoria adaptativa y estrategias para resolver problemas.

  Permite movimientos de empeoramiento para escapar de óptimos locales y para evitar recorridos cíclicos incorpora una generación de vecinos modificando que evita la exploración de zonas que ya han sido visitadas.

  Además, emplea mecanismos de reiniciación para mejorar la capacidad del algoritmo para la exploración del espacio de búsqueda.

  Para la memoria a corto plazo se implementa una lista tabú que permite a la búsqueda determinar una solución organizando la manera en al que se explora el espacio. En nuestro caso se implementará una lista de movimientos tabú, en la que se eliminan del entorno aquellos movimientos utilizados anteriormente mediante el operador 2-opt (Figura 2.1).

Resultados de los algoritmos

Se mostrarán tablas gráficas del estudio realizada para cada algoritmo:

- Algoritmo de Búsqueda Aleatoria

	St70		Ch130		A280		p654		Vm1084		Vm1748	
#Ejecución y Seed	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev
1	3270	1	47662	1	32736	1	2022051	1	8600797	1	15039585	1
2	3306	1	44090	1	33971	1	2046487	1	8629254	1	14917610	1
3	3537	1	46736	1	33492	1	2001508	1	8743254	1	15091601	1
4	3554	1	47617	1	32782	1	1956608	1	8616299	1	15186690	1
5	3739	1	46059	1	33104	1	2053441	1	8593250	1	14743115	1
6	3790	1	45073	1	33442	1	2025476	1	8676759	1	14942129	1
7	3453	1	46507	1	33738	1	2056556	1	8607154	1	14809271	1
8	4050	1	45925	1	33150	1	2044331	1	8707822	1	15218796	1
9	3849	1	46059	1	32116	1	2028050	1	8355666	1	14641586	1
10	3965	1	48836	1	33244	1	1966576	1	8546644	1	15029041	1
Media	3651,3	1	46456,4	1	33177,5	1	2020108,4	1	8607689,9	1	14961942,4	1
Desviacion Típica	254,9235376	0	1290,56501	0	510,037106	0	33259,1371	0	100604,517	0	178887,492	0

Del algoritmo aleatorio no podemos sacar muchas cosas en claro, debido a que, si es cierto que devuelve un coste, pero sin un óptimo. Es utilizado como medida de inicialización en otros métodos de búsqueda

- Algoritmo Greedy

	St70		Ch130		A280		p654		Vm1084		Vm1748	
#Ejecución	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev
1	839	1	8611	1	3235	1	43701	1	299192	1	431158	1
2	832	1	7431	1	3137	1	47355	1	303297	1	413657	1
3	843	1	7358	1	3144	1	47034	1	307738	1	418157	1
4	815	1	7360	1	3193	1	45715	1	302737	1	433751	1
5	886	1	7383	1	3021	1	46510	1	299295	1	429444	1
6	886	1	7506	1	3175	1	43210	1	300685	1	427211	1
7	867	1	8482	1	3251	1	48301	1	303134	1	416974	1
8	835	1	7498	1	3169	1	44631	1	300414	1	438468	1
9	825	1	8280	1	2975	1	44945	1	297042	1	436041	1
10	843	1	7613	1	2989	1	44764	1	301986	1	422443	1
Media	847,1	1	7752,2	1	3128,9	1	45616,6	1	301552	1	426730,4	1
Desviacion Típica	23,26993769	0	473,499694	0	94,4462281	0	1570,20809	0	2801,02496	0	8117,46086	0

Da resultados muy buenos con una heurística muy simple. Además, se puede utilizar como inicialización para otros algoritmos de búsqueda.

- Algoritmo de Búsqueda Primer Mejor

	St70		Ch130		A280		p654		Vm1084		Vm1748	
#Ejecución	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev
1	1989	329	22780	857	2793	80627	101907	238242	2918201	18625	5332177	28807
2	1837	612	23882	924	2792	73003	97469	219625	2832347	19315	5218610	30207
3	1897	666	24858	735	2787	84714	100878	214656	2936190	16876	5195505	31125
4	1801	529	26386	539	2801	66689	101242	222867	2812217	20786	5273611	29521
5	1898	335	27217	672	2799	74788	101062	225844	2834451	18760	5279270	28805
6	1919	396	23246	964	2796	77750	100430	245154	2856995	18450	5313977	29318
7	1888	519	25442	592	2798	81984	98613	239889	2893111	19443	5246917	31663
8	1900	442	24845	784	2790	77594	101553	222302	2757102	20076	5151002	30548
9	1740	431	24942	800	2793	71051	96371	209597	2709657	20827	5317085	30029
10	1879	517	26749	553	2791	64084	102725	250877	2899220	19194	5320409	31099
Media	1874,8	477,6	25034,7	742	2794	75228,4	100225	228905,3	2844949,1	19235,2	5264856,3	30112,2
Desviacion Típica	64,72990036	105,820792	1392,14949	143,227092	4,17133072	6296,27018	1951,75296	13061,4113	68163,2044	1117,02827	57685,5286	948,838216

Sus resultados mejoran el aleatorio, pero no al Greedy ya que no busca por todo el espacio, si no que se queda con la primera solución encontrada mejor que la inicial cuya solución es generada aleatoriamente. Puede ocurrir debido a que se utiliza un operador de vecinos y un número de convergencia, podríamos hacer un estudio de convergencia y ver cuál sería el más recomendable.

- Algoritmo de Búsqueda Mejor de Todos

	St70		Ch130		A280		p654		Vm1084		Vm1748	
#Ejecución	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev
1	1003	112001	11516	208001	2784	448001	88709	1046401	1288266	1734401	2155158	2796801
2	1014	112001	10305	208001	2784	448001	94382	1046401	1262482	1734401	2078898	2796801
3	1093	112001	11617	208001	2784	448001	95234	1046401	1222061	1734401	2114825	2796801
4	957	112001	11128	208001	2790	448001	93473	1046401	1207041	1734401	2152987	2796801
5	1005	112001	10373	208001	2784	448001	88954	1046401	1260397	1734401	2069797	2796801
6	1182	112001	10966	208001	2784	448001	91392	1046401	1230002	1734401	2022865	2796801
7	1038	112001	10682	208001	2784	448001	91546	1046401	1201350	1734401	2082993	2796801
8	1023	112001	11488	208001	2784	448001	86661	1046401	1238047	1734401	2028865	2796801
9	1027	112001	10884	208001	2784	448001	90690	1046401	1236354	1734401	2060920	2796801
10	1026	112001	12369	208001	2784	448001	90535	1046401	1211068	1734401	1971923	2796801
Media	1036,8	112001	11132,8	208001	2784,6	448001	91157,6	1046401	1235706,8	1734401	2073923,1	2796801
Desviacion Típica	58,04101998	0	599,263348	0	1,8	0	2534,9133	0	26311,2582	0	54641,994	0

Es una mejora de la búsqueda del primer mejor, ya que sigue buscando por todo el vecindario hasta una convergencia determinada previamente. Sus resultados mejoran el anterior bastante.

- Algoritmo Enfriamiento Simulado

	St70		Ch130		A280		p654		Vm1084		Vm1748	
#Ejecución	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev
1	892	112069	9698	208129	3434	448279	129509	1047053	1050670	1735483	1725630	2798547
2	784	112069	9683	208129	3158	448279	125103	1047053	995556	1735483	1824314	2798547
3	864	112069	9822	208129	3159	448279	107941	1047053	1082238	1735483	1972873	2798547
4	872	112069	10045	208129	3060	448279	117093	1047053	1029334	1735483	1764815	2798547
5	891	112069	10156	208129	3665	448279	130405	1047053	994641	1735483	1818392	2798547
6	884	112069	9316	208129	3627	448279	125698	1047053	1001712	1735483	1834494	2798547
7	864	112069	9571	208129	3505	448279	121618	1047053	1034941	1735483	1792125	2798547
8	912	112069	9888	208129	2842	448279	120740	1047053	1021805	1735483	1823962	2798547
9	985	112069	9032	208129	3722	448279	115612	1047053	996428	1735483	1786399	2798547
10	922	112069	8755	208129	3019	448279	116913	1047053	961849	1735483	1862911	2798547
Media	887	112069	9596,6	208129	3319,1	448279	121063,2	1047053	1016917,4	1735483	1820591,5	2798547
Desviacion Típica	48,36941182	0	421,178395	0	293,417944	0	6570,45465	0	32473,1048	0	62699,3644	0

Da buenos resultados, aunque no da los óptimos debido a la probabilidad de los aleatorios utilizados para el resultado inicial y a los saltos hacia soluciones anteriores que también se apoya en la aleatoriedad. Para los datos en los que los puntos están muy separados unos de otros el resultado es bastante malo como puede verse en el vm1748 que ni siquiera se acerca al proporcionado por el Greedy, sin embargo, para el A2080 si ya que esos puntos están muy juntos en el espacio.

- Algoritmo de Búsqueda Tabú

	St70		Ch130		A280		p654		Vm1084		Vm1748	
#Ejecución	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev	Coste	#Ev
1	950	28669	11261	52590	2855	112380	107741	264273	1709533	434764	3101290	702755
2	931	28600	11471	52719	2823	112659	107821	262967	1673241	435847	3040290	704502
3	1028	28600	11402	52719	2818	112380	108684	263620	1675488	434764	3134989	701008
4	998	28600	11242	52848	2840	112659	108013	262967	1669151	435847	3044841	701008
5	943	28738	11759	52590	2828	112938	107836	262967	1675659	435847	3088229	701008
6	862	28669	11141	52719	2839	112659	107865	262967	1690787	436930	3138805	704502
7	949	28669	11724	52719	2830	112659	107754	263620	1656083	434764	3100191	704502
8	956	28600	11067	52719	2853	112938	108011	264926	1686538	435847	3166965	701008
9	914	28669	11530	52719	2850	112938	107837	262967	1692842	435847	3053907	704502
10	883	28600	11879	52719	2816	112938	107815	262967	1679664	435847	3032992	701008
Media	941,4	28641,4	11447,6	52706,1	2835,2	112714,8	107937,7	263424,1	1680898,6	435630,4	3090249,9	702580,3
Desviacion Típica	46,50204297	45,7694221	261,694555	66,3394726	13,62938	208,784482	263,606923	656,256878	14010,2396	649,8	44301,7133	1648,1165

La búsqueda debería dar mejores resultados debido a que guarda el mejor movimiento para utilizarlo en otra iteración del algoritmo, sin embargo, tiene probabilidades por lo que los resultados podrían mejorar con otras semillas.

Resultados globales y comparacion entre algoritmos

- St70.tsp

	St70 (675)			
Modelo	Media	Mejor	Desviacion Típica	Ev
Aleatorio	3651,3	3270	254,9235376	1
Greedy	847,1	815	473,4996938	1
Primer Mejor	1874,8	1740	1392,149493	477,6
Mejor deTodos	1036,8	957	599,2633478	112001
ES	887	784	421,1783945	112069
Tabu	941,4	862	261,6945548	28641,4

Para el fichero St70 podemos ver que Todos se parecen bastante, sin embargo, el enfriamiento simulado es capaz de ganar al Greedy en alguna ejecución, pero su media de resultados es mayor, aunque si es verdad que es más costoso computacionalmente, aunque en tiempo no se aprecia.

- Ch130.tsp

	Ch130 (6110)			
Modelo	Media	Mejor	Desviacion Típica	Ev
Aleatorio	46456,4	44090	1290,565008	1
Greedy	7752,2	7358	473,4996938	1
Primer Mejor	25034,7	22780	1392,149493	742
Mejor deTodos	11132,8	10305	599,2633478	208001
ES	9596,6	9596,6	421,1783945	208129
Tabu	11447,6	8755	261,6945548	52706,1

En este caso, el algoritmo Greedy es el mejor y gana por una diferencia que va siendo considerable a todos los demás. Es cierto que su desviación es mayor a la del Enfriamiento simulado o la búsqueda tabú, pero es debido a que empieza cada vez en sitios aleatorios.

- A280.tsp

	A280 (2579)			
Modelo	Media	Mejor	Desviacion Típica	Ev
Aleatorio	33177,5	32116	510,0371065	1
Greedy	3128,9	2975	94,44622809	1
Primer Mejor	2794	2787	4,171330723	75228,4
Mejor deTodos	2784,6	2784	1,8	448001
ES	3319,1	2842	293,4179442	448279
Tabu	2835,2	2816	263,6069233	263424,1

Para este fichero tenemos competencia entre todos exceptuando el aleatorio por razones obvias. Podemos ver que el algoritmo Mejor de todos es capaz de ser el que consigue su mínimo, con menos desviación y mejor media, aunque con un numero de evaluaciones bastante grande y con un tiempo de ejecución que va siendo notable. Eso puede deberse debido a que tiene los puntos muy juntos y prácticamente no hay cambio al elegir uno u otro en los mejores ya que no utilizamos números reales si no enteros.

- P654.tsp

	P654 (34.643)			
Modelo	Media	Mejor	Desviacion Típica	Ev
Aleatorio	2020108,4	1956608	33259,13712	1
Greedy	45616,6	43210	1570,208088	1
Primer Mejor	100225	96371	1951,752956	228905,3
Mejor deTodos	91157,6	86661	2534,9133	1046401
ES	121063,2	107941	6570,454654	1047053
Tabu	107937,7	107741	263,6069233	263424,1

De nuevo el algoritmo Greedy es el que se lleva el mejor resultado, mientras que otros con una complejidad de búsqueda mayores son peores.

- Vm1084.tsp y Vm1748.tsp

	Vm1084 (239.297)			
Modelo	Media	Mejor	Desviacion Típica	Ev
Aleatorio	8607689,9	8355666	100604,5169	1
Greedy	301552	297042	2801,024955	1
Primer Mejor	2844949,1	2709657	68163,20436	19235,2
Mejor deTodos	1235706,8	1201350	26311,2582	1734401
ES	1016917,4	961849	32473,10476	1735483
Tabu	1680898,6	1656083	14010,23961	435630,4

	Vm1748			
Modelo	Media	Mejor	Desviacion Típica	Ev
Aleatorio	14961942,4	14641586	178887,4916	1
Greedy	426730,4	413657	8117,460862	1
Primer Mejor	5264856,3	5151002	57685,52861	30112,2
Mejor deTodos	2073923,1	1971923	54641,99398	2796801
ES	1820591,5	1725630	62699,36445	2798547
Tabu	3090249,9	3032992	44301,71327	702580,3

Ambos ficheros de puntos son los más grandes, sin embargo, Greedy sigue dando la mejor solución, esto se debe básicamente a que su simple heurística obtiene el camino más corto que satisface el problema del viajero que estamos tratando. Se puede observar que otros algoritmos dan del orden del millón mientras que Greedy lo baja hasta casi su optimo conocido.

Posibles mejoras a los algoritmos

Como posibles mejoras al Greedy por no encontrar el óptimo global podrían introducirse unas heurísticas algo más trabajadas que solo una comparación, como por ejemplo que cada un número determinado de pasos elija la peor o que dada una probabilidad salte a uno peor.

Como mejora a los algoritmos de búsqueda del **Primer Mejor** y del **Mejor de Todos**, y para **Enfriamiento Simulado** la propuesta es simple, y es poner como solución inicial y partiendo d ellos puntos de la solución Greedy, ya que esta proporciona una buena solución de inicio y mejoraría al aplicar el algoritmo.

Por último, tenemos la búsqueda tabú, que debería de dar los mejores resultados, sin embargo, la tener en la re inicialización basada por frecuencias da bastante peor resultado que teniéndola basada en Greedy. Aun así, da unos resultados que podrían mejorarse mediante un número más alto de convergencia.

Para probar algunos cambios, se ha creado la siguiente tabla que muestra una comparación con iniciación basada en Greedy e iniciación basada en Aleatorio:

st70.tsp	Inicialización	
Algoritmo	Aleatorio	Greedy
Mejor de todos	957	811
Enfriamiento Simulado	784	957
Búsqueda Tabú	941,4	767

Vm1084.tsp	Inicialización	
Algoritmo	Aleatorio	Greedy
Mejor de todos	1201350	293447
Enfriamiento Simulado	961849	293447
Búsqueda Tabú	1656083	293844

Para cada algoritmo se han realizado 10 evaluaciones y cogida la mejor.

Puede verse que los saltos para data sets grandes el cambio de coste es muy significativo empezando desde Greedy y en el caso de la búsqueda tabú utilizando como re inicialización en un 0.5 de probabilidad el Greedy. Por otra parte para data sets pequeños en este caso el menor que se ha evaluado, se los saltos son pequeños.

Conclusiones

En definitiva, tras el estudio de los algoritmos con los distintos data sets puede verse que el algoritmo Greedy es muy bueno con una heurística simple ya que para este problema concreto cumple su función que es la de buscar el menor camino, sin embargo, otros algoritmos con mas complejidad no dan tan buenos resultados, o al menos empezando desde aleatorios.

Tras la comparación entre inicializar con aleatorio o con Greedy podemos sacar en claro que para data sets pequeños empezar de aleatorios puede proporcionarte buenas soluciones y que para data sets más grandes donde el número de combinatoria de ciudades es muy grande es mejor utilizar una función Greedy para la inicialización.

En cualquier caso, utilizar Greedy como inicialización ayuda a resolver el problema de mejor forma obteniendo mejores resultados, aunque para el problema del viajante en concreto, es mejor utilizar el algoritmo Greedy ya que es rápido y da una muy buena solución, para el problema QAP puede ser totalmente distinto y obtener mejores resultados con el Enfriamiento Simulado.

En definitiva, todo dependerá del problema al que haya que enfrentarse, y ahí es donde influye la capacidad del programador para saber que algoritmo podría ser mejor.