Práctica 7. Problema de rutas de vehículos (VRP)

Daniel Dóniz García

3 de mayo de 2022

1. Introducción.

En esta práctica se implementarán algoritmos heurísticos para el problema de rutas de vehículos (VRP). El programa se implementará en C++.

2. Algoritmos usados para resolver el problema.

Para ello usaremos 3 tipos de algoritmos que iré explicando más adelante. Voraz, GRASP, GVNS.

2.1. Voraz.

El algoritmo Voraz es una estrategia de búsqueda por la cual se sigue una heurística. Consiste en elegir la opción óptima en cada paso local con la espereanza de llegar a una solución óptima. El pseudocódigo que se ha seguido es el siguiente:

```
1: Elem = S;
2: S=8:
3: Obtener s c = centro(Elem):
4: repeat
5:    Obtener el elemento s * E Elem ms alejado de s c:
6:    S=SU{s*};
7:    Elem = Elem-{S*}:
8:    Obtener s c= centro(S);
9: until (S| = m)
10: Devolver S:
```

2.2. GRASP.

El algoritmo GRASP Greedy Randomized Adaptive Search Procedure es un algoritmo constructivo, es decir va anadiendo elementos a la solucion que es inicialmente vacia, adaptativo, un elemento se evalua teniendo en cuenta los elementos previamente incluidos en el solucion, se mide la conveniencia de incluir el elemento como parte de la solucion y tiene una estrategia greedy o voraz el cual escoge el elemento que optimiza la Funcion objetivo. El algoritmo tiene la siguiente estructura:

```
1: Begin
2: Preprocesamiento
3:    Repeat
4:       Fase Gonstructiva(Solucion);
5:       PostProcesamiento(Solucion);
6:       Actualizar(Solucion, MejorSolucion);
7:    Until (Criterio de parada);
8: End.
```

2.3. GVNS.

Este algoritmo se basa en la búsqueda por entornos variable. Se inicia desde un solucion aleatoria y se genera su entorno, una vez generado el entorno pueden ocurrir dos cosas, la solucion es un minimo/maximo local dependiendo de la Funcion objetivo, si esto es asi el siguiente paso es generar un entorno mes grande que contenga la solucion actual. En caso de que se encuentre una maxima local distinto a la solucion actual se tomara esta solucion como mejor solucion y se generara su entorno, y se buscara un minimo maxim° local. El numero de entornos generados es un valor dictado por el usuario, suelen ser tres entornos como maxima. Este proceso se repite hasta que se alcanza un nUmero de iteraciones maximo o un ralimero de iteraciones maximas sin mejora. El pseudocodigo de este algoritmo es el siguiente:

```
1: 5 = Generarinicial();
2: repeat
3:    repeat
4:    X = agitacion(k, S);
5:    busquedaLocal(X)
6:    if X mejor que S
7:       S = X;
8:       reset del tamao del entorno
9:    else
10:      ampliamos entorno
11:   hasta tamao entorno == maximo
12: haste condicin de parada
```

3. Resultados obtenidos.

En este apartados estaremos ejecutando los diferentes problemas con los algoritmos implementados.

3.1. Voraz.

Algoritmo voraz

Problema	n	Ejecución	Distancia$_{TotalRecorrida}$	CPU$_{Time}$
I40j_2m_S1_1		1	217	83
I40j_2m_S1_1		2	217	118
I40j_2m_S1_1		3	217	97
I40j_2m_S1_1		4	217	113
I40j_2m_S1_1		5	217	133
I40j_4m_S1_1		1	280	126
I40j_4m_S1_1		2	280	111
I40j_4m_S1_1		3	280	131
I40j_4m_S1_1		4	280	87
I40j_4m_S1_1		5	280	99
I40j_6m_S1_1		1	362	160
I40j_6m_S1_1		2	362	140
I40j_6m_S1_1		3	362	205
I40j_6m_S1_1		4	362	120
I40j_6m_S1_1		5	362	147
I40j_8m_S1_1		1	444	127
I40j_8m_S1_1		2	444	112
I40j_8m_S1_1		3	444	144
I40j_8m_S1_1		4	444	123
I40j_8m_S1_1		5	444	121

3.2. GRASP.

GRASP

Problema	n	LCR	Ejecución	Distancia$_{TotalRecorrida}$	CPU$_{Time}$
I40j_2m_S1_1		2	1	150	366338
I40j_2m_S1_1		2	2	187	365587
I40j_2m_S1_1		2	3	166	438073
I40j_2m_S1_1		2	4	194	255669
I40j_2m_S1_1		2	5	205	380810
I40j_2m_S1_1		3	1	188	616424
I40j_2m_S1_1		3	2	189	612584
I40j_2m_S1_1		3	3	198	463571
I40j_2m_S1_1		3	4	171	514067
I40j_2m_S1_1		3	5	214	474361
I40j_4m_S1_1		2	1	222	248670
I40j_4m_S1_1		2	2	276	141429
I40j_4m_S1_1		2	3	250	213296
I40j_4m_S1_1		2	4	265	216753
I40j_4m_S1_1		2	5	256	243809
I40j_4m_S1_1		3	1	297	277509
I40j_4m_S1_1		3	2	286	205586
I40j_4m_S1_1		3	3	231	306049
I40j_4m_S1_1		3	4	282	391419
I40j_4m_S1_1		3	5	281	222153
I40j_6m_S1_1		2	1	315	162861
I40j_6m_S1_1		2	2	302	181802
I40j_6m_S1_1		2	3	328	104840
I40j_6m_S1_1		2	4	317	122739
I40j_6m_S1_1		2	5	282	147252
I40j_6m_S1_1		3	1	357	225510
I40j_6m_S1_1		3	2	381	180370
I40j_6m_S1_1		3	3	335	169212
I40j_6m_S1_1		3	4	323	239936
I40j_6m_S1_1		3	5	345	256631
I40j_8m_S1_1		2	1	368	88583
I40j_8m_S1_1		2	2	424	84964
I40j_8m_S1_1		2	3	377	91671
I40j_8m_S1_1		2	4	433	112534
I40j_8m_S1_1		2	5	429	124284
I40j_8m_S1_1		3	1	369	115835
I40j_8m_S1_1		3	2	437	115241
I40j_8m_S1_1		3	3	424	111545
I40j_8m_S1_1		3	4	435	121542
I40j_8m_S1_1		3	5	440	90556

3.3. GVNS.

GVNS

Problema	m	k_{max}	Ejecución	$Distancia_{TotalRecorrida}$	CPU_{Time}
I40j_2m_S1_1		2	1	279	6343769
I40j_2m_S1_1		2	2	268	6247031
I40j_2m_S1_1		2	3	261	6290606
I40j_2m_S1_1		2	4	267	6305427
I40j_2m_S1_1		2	5	293	6122472
I40j_2m_S1_1		3	1	255	8596277
I40j_2m_S1_1		3	2	270	8473586
I40j_2m_S1_1		3	3	263	8682803
I40j_2m_S1_1		3	4	257	8707754
I40j_2m_S1_1		3	5	254	8573133
I40j_4m_S1_1		2	1	315	11539051
I40j_4m_S1_1		2	2	321	11574764
I40j_4m_S1_1		2	3	288	11806880
I40j_4m_S1_1		2	4	322	11236430
I40j_4m_S1_1		2	5	305	11586749
I40j_4m_S1_1		3	1	292	14067567
I40j_4m_S1_1		3	2	302	14102629
I40j_4m_S1_1		3	3	313	14066576
I40j_4m_S1_1		3	4	295	14237904
I40j_4m_S1_1		3	5	306	14490445
I40j_6m_S1_1		2	1	362	16221422
I40j_6m_S1_1		2	2	394	16203910
I40j_6m_S1_1		2	3	363	16063362
I40j_6m_S1_1		2	4	403	16106800
I40j_6m_S1_1		2	5	392	16250142
I40j_6m_S1_1		3	1	406	19249636
I40j_6m_S1_1		3	2	392	19753339
I40j_6m_S1_1		3	3	397	19501254
I40j_6m_S1_1		3	4	364	19374294
I40j_6m_S1_1		3	5	381	19023063
I40j_8m_S1_1		2	1	448	16587181
I40j_8m_S1_1		2	2	464	16626043
I40j_8m_S1_1		2	3	434	16487019
I40j_8m_S1_1		2	4	433	16625966
I40j_8m_S1_1		2	5	473	17224251
I40j_8m_S1_1		3	1	417	20504009
I40j_8m_S1_1		3	2	425	19953786
I40j_8m_S1_1		3	3	468	20278174
I40j_8m_S1_1		3	4	462	20029300
I40j_8m_S1_1		3	5	418	19830009