UNIVERSIDAD SIMÓN BOLÍVAR
Dpto. de Computación y Tecnología de la Información
CI-2693 - Lab. de Algoritmos y Estructuras de Datos III

Proyecto II

Un algoritmo heurístico para resolver el Problema del Cartero Rural

Autor:
Christopher Gómez (18-10892)
Ka Fung (18-10492)

Profesor:
Guillermo Palma

17 de enero de 2022

1. Introducción

El siguiente estudio experimental consiste en implementar un algoritmo heurístico para obtener soluciones aproximadas al Problema del Cartero Rural (RPP), modelado mediante el uso de grafos no dirigidos. Para ello, se empleará el algoritmo presentado por Pearn y Wu, el cual es una modificación del algoritmo de Christofides et al. El RPP consiste en encontrar un ciclo de costo mínimo en un grafo no dirigido, tal que atraviese un conjunto de lados requeridos al menos una vez.

En el algoritmo de Pearn y Wu requiere de la implementación de un algoritmo de apareamiento perfecto de costo mínimo. Por ello, se implementará dos algoritmos heurísticos para obtener un apareamiento perfecto de un grafo completo (un algoritmo ávido y el Vertex-Scan presentado por David Avis [1]), que proporcionan una solución aproximada y que sirven como una alternativa mucho más eficiente y menos compleja que el algoritmo de Edmonds, el cual proporciona una solución exacta al problema.

Luego, el objetivo de este estudio consiste en comparar la eficacia y eficiencia de estos dos algoritmos aplicados a solucionar 36 distintas del problema RPP.

2. Diseño de la solución

Para resolver el *problema del cartero rural* se usó el algoritmo constructivo proveído por el profesor del curso, basado en el presentado por Pearn y Wu en [2].

La idea principal de dicho algoritmo consiste en modelar la instancia del problema con un grafo no dirigido $G = (V, E)$ y un subconjunto $R \subseteq E$ que representa las aristas por las que el cartero debe pasar en el ciclo buscado. Luego, construir a partir del conjunto R un grafo $G' = (V_R, R)$, al cual se le añaden lados y vértices en varias ocasiones hasta obtener finalmente un grafo par y conexo del cual obtener un ciclo euleriano que servirá como solución aproximada del problema.

Los procedimientos para añadir lados al grafo G' en dos ocasiones se pueden resumir y simplificar como los siguientes:

- Se construye un grafo G_t completo donde cada vértice es una componente conexa de G' y el costo de cada lado corresponde al del camino de costo mínimo entre dichas componentes en G.
- Se obtiene el conjunto E_{MST} de los lados del arbol mínimo cobertor de G_t.
- Se construye E_t como el conjunto de lados de los caminos de costo mínimo asociados a cada lado de E_{MST}.
- Se agrega al conjunto de vértices de G' los vértices que aparecen E_t, y luego los lados admitiendo duplicados.

En este punto se ha convertido G' en un grafo conexo, pues hemos añadido a cada componente conexa del grafo G' inicial
Construcción de grafo GR y G' (mapeo G -¿GR)

Construcción del grafo G0 (mapeo GR -¿G0)

Implementación del algoritmo de Dijkstra para grafos no dirigidos

Implementación del ciclo euleriano para grafos no dirigidos

3. Detalles de la implementacion

Algoritmo para obtener las componentes conexas en grafos no dirigidos.

Algoritmo de Dijkstra para obtener los caminos de costo mínimo en grafos no dirigidos.

Algoritmo de Prim para obtener el árbol mínimo cobertor.

Algoritmo para obtener el ciclo euleriano de grafos no dirigidos.

Algoritmo de apareamiento perfecto ávido.

Algoritmo de apareamiento perfecto Vertex-Scan.

Programa cliente para la solución del problema RPP.

4. Detalles de la plataforma

Los siguientes son los detalles de la máquina y el entorno donde se ejecutaron las implementaciones:

- **Sistema operativo**: GNU/Linux (Debian 11 Bullseye 64-bit).
- **Procesador**: Intel(R) Core(TM) i3-2120 CPU @3.30GHz.
- **Memoria RAM**: 4,00 GB (3,88 GB usables).
- **Compilador**: kotlinc-jvm 1.6.10 (JRE 11.0.12+7-post-Debian-2).
- **Entorno de ejecución**:OpenJDK Runtime Environment (build 11.0.12+7-post-Debian-2).

5. Resultados experimentales

Para realizar una comparación entre la solución obtenida y la solución óptima de una instancias del problema de RPP, se calculó el porcentaje de desviación con la fórmula:

$$\%desv = \frac{\text{valor obtenido} - \text{valor óptimo}}{\text{valor óptimo}} * 100 \quad (1)$$

6. Análisis de los resultados

7. Conclusiones

-

Nombre de la instancia	Valor óptimo	Desviación (%)		Promedio Resultado V-S
		Alg. Ávido	Vertex-Scan	
UR132	**23913**	16,271	20,655	28852
UR135	**33088**	14,335	16,814	38651
UR137	**42797**	10,347	13,594	48615
UR142	**25548**	15,935	19,035	30411
UR145	**39008**	7,342	12,603	43924
UR147	**55959**	6,346	7,604	60214
UR152	**28975**	11,365	17,336	33998
UR155	**49156**	4,669	7,078	52635
UR157	**70231**	3,91	4,595	73458
UR162	**32341**	10,998	12,462	36371
UR165	**58800**	4,978	6,2	62446
UR167	**82481**	2,917	3,572	85427
UR532	**17277**	16,438	22,261	21123
UR535	**23635**	16,306	16,839	27615
UR537	**30098**	9,253	12,324	33807
UR542	**17830**	14,173	20,269	21444
UR545	**29648**	6,365	8,835	32267
UR547	**38692**	4,784	7,877	41740
UR552	**20097**	12,579	16,875	23488
UR555	**34488**	4,126	8,069	37271
UR557	**48307**	4,651	3,97	50225
UR562	**24556**	7,188	10,541	27144
UR565	**42828**	4,056	5,17	45042
UR567	**58971**	4,087	4,015	61339
UR732	**21114**	18,751	21,313	25614
UR735	**28663**	10,623	15,26	33037
UR737	**36588**	6,308	11,45	40777
UR742	**22557**	13,078	17,916	26598
UR745	**32493**	10,19	10,96	36054
UR747	**47764**	5,481	6,385	50814
UR752	**25131**	12,614	15,796	29101
UR755	**41774**	4,517	8,357	45265
UR757	**58416**	4,002	5,876	61849
UR762	**27880**	8,691	13,155	31548
UR765	**50492**	4,565	5,696	53368
UR767	**72950**	3,413	3,911	75803

Cuadro 1: Desviación de los resultados obtenidos para cada algoritmo

Nombre de la instancia	Promedio del tiempo (seg.)	
	Alg. Ávido	Vertex-Scan
UR132	0,673	0,637
UR135	1,096	0,965
UR137	1,217	1,05
UR142	1,025	0,81
UR145	1,164	1,058
UR147	1,373	1,192
UR152	1,169	0,929
UR155	1,312	1,223
UR157	1,352	1,169
UR162	1,224	1,313
UR165	1,375	1,215
UR167	1,416	1,148
UR532	0,319	0,296
UR535	0,45	0,416
UR537	0,47	0,467
UR542	0,334	0,327
UR545	0,529	0,457
UR547	0,449	0,443
UR552	0,388	0,391
UR555	0,505	0,464
UR557	0,516	0,445
UR562	0,487	0,431
UR565	0,516	0,501
UR567	0,693	0,511
UR732	0,492	0,479
UR735	0,737	0,666
UR737	0,822	0,746
UR742	0,596	0,525
UR745	0,802	0,688
UR747	0,807	0,75
UR752	0,647	0,693
UR755	1,047	0,815
UR757	0,909	0,866
UR762	0,873	0,732
UR765	1,001	0,968
UR767	0,907	0,858
Total	29,692	26,644
Promedio	0,825	0,74

Cuadro 2: Tiempo de ejecución de la implementación para cada instancia

-

8. Referencias (en caso de tenerlas)

ñema.