

Instituto de Engenharia de Sistemas e Computadores de Coimbra
Institute of Systems Engineering and Computers
INESC – Coimbra

Luís A. Santos
lsantos@issb.pt

João M. Coutinho-Rodrigues
coutinho@dec.uc.pt

**Prestação de Serviços Distribuídos em Redes com Restrições
de Capacidade - Uma Heurística de Pesquisa de Caminho
Melhorada**

No. 13

2004

ISSN: 1645-2631

Instituto de Engenharia de Sistemas e Computadores de Coimbra
INESC – Coimbra
Rua Antero de Quental, 199; 3000-033 Coimbra; Portugal
www.inescc.pt

Prestação de Serviços Distribuídos em Redes com Restrições de Capacidade

Uma Heurística de Pesquisa de Caminho Melhorada

Luís Santos
Instituto Superior Bissaya Barreto, Bencanta, 3040 Coimbra, Portugal
lsantos@issb.pt

João Coutinho-Rodrigues
Departamento de Engenharia Civil, Faculdade de Ciências e Tecnologia,
PoloII, Universidade de Coimbra, 3030 Coimbra, Portugal
coutinho@dec.uc.pt

Abstract

Practical routing problems involving the distribution or collection of products, services or persons which are distributed along city streets, such as, collection of solid waste, delivering of goods or routing of school buses, are very common. This kind of real problems may be modelled by Capacitated Arc Routing Problem (CARP). The CARP is known to be a NP-Hard problem. Due the high computational complexity of the problem, the methods to obtain optimal solutions are only used for small instance problems. Therefore, several heuristics and metaheuristics have been developed. They provide good solutions in acceptable computer time. In this article we present some improvements for the path-scanning heuristic which has been developed for the CARP. 4 new criteria, 3 new probabilistic functions for selecting the criteria and a new rule of collecting inside an ellipse at the end of each route are proposed. The computational results obtained for two sets of test problems are presented.

Resumo

Os problemas localizados ao longo de ruas e que envolvem a prestação de serviços ou a distribuição/recolha de produtos ou pessoas são muito comuns em redes urbanas. Este tipo de problemas reais pode ser modelado pelo Problema de Circulação com Restrições de Capacidade (na terminologia anglófona *Capacitated Arc Routing Problem – CARP*). O CARP é um problema *NP-Hard*, ou seja, é extremamente difícil obter a solução óptima num tempo computacional aceitável através de algoritmos exactos, a não ser para problemas de dimensões muito reduzidas. Por isso, várias heurísticas e metaheurísticas têm sido desenvolvidas, permitindo obter soluções aproximadas (ou mesmo óptimas ocasionalmente), num tempo computacional aceitável. Neste relatório de investigação apresentam-se alguns melhoramentos introduzidos na heurística *path-scanning*, que foi desenvolvida para tratar o

CARP. Apresentam-se 4 novos critérios, 3 novas funções de probabilidade e uma nova regra impondo que apenas sejam servidos arcos no interior de uma elipse no final de cada circuito. Apresentam-se também os resultados obtidos para dois conjuntos de problemas de teste.

Palavras chave: Redes, circulação, heurísticas

1. Introdução

Um dos aspectos importantes no que concerne à circulação de veículos envolvendo a prestação de serviços localizados ao longo de ruas de uma cidade é a obtenção de circuitos otimizados. Os problemas teóricos que envolvem a circulação de veículos têm merecido grande atenção por parte da comunidade científica nos últimos anos, sobretudo porque permitem modelar vários problemas reais de enorme interesse, tais como, recolha de resíduos sólidos urbanos [4,11,24], recolha de alunos em autocarros escolares [8], limpeza de ruas [6,7], leitura de contadores eléctricos [27] ou distribuição de correspondência porta-a-porta [22,23]. Todos estes problemas têm em comum a particularidade de envolver a circulação de veículos com restrições de capacidade (a capacidade máxima do veículo e/ou a duração do turno de trabalho das equipas envolvidas).

Este tipo de problemas reais pode ser modelado pelo CARP. Seja $G = (V, E)$ um grafo não orientado, onde $V = \{v_0, v_1, \dots, v_n\}$ é o conjunto dos nós e $E = \{(v_i, v_j) : v_i, v_j \in V, i < j\}$ é o conjunto dos arcos não orientados. A cada arco (v_i, v_j) está associado um custo não negativo c_{ij} e uma quantidade não negativa q_{ij} que representa o serviço a prestar. O nó v_0 representa o depósito onde estão m veículos homogéneos com capacidade individual W ($W \geq \max q_{ij}$). O número de veículos pode ser fixo ou uma variável do problema. Neste relatório de investigação consideramos que este parâmetro constitui uma variável do problema. O objectivo do CARP é obter um conjunto de circuitos óptimos, de forma que:

- a) cada arco com serviço seja servido exactamente uma vez;
- b) cada circuito comece e termine no depósito;
- c) o serviço total em cada circuito não exceda W ;
- d) o custo total de circulação seja minimizado.

O CARP foi definido pela primeira vez por Golden e Wong [15], que demonstraram ser esse problema *NP-Hard*. Este problema pode ser formulado através de um programa linear e resolvido utilizando as técnicas de *branch-and-cut* ou *branch-and-bound*, mas apenas para problemas de reduzidas dimensões. Por exemplo, Hirabayashi *et al.* [19] apresentam um procedimento baseado na técnica *branch-and-bound* que foi apenas utilizado para problemas com, no máximo, 30 arcos orientados. Assim, para problemas com dimensão elevada, que é o que acontece em geral com os problemas reais, é

praticamente impossível obter a solução óptima num tempo computacional aceitável utilizando algoritmos exactos.

Desde que o CARP foi definido pela primeira vez, vários algoritmos para determinar o limite inferior de cada problema [1,2,5,15,24,25,29] e várias heurísticas [4,6,8,9,14,15,26] têm sido apresentadas. Estas heurísticas fornecem soluções aproximadas em tempos computacionais aceitáveis (para ver mais detalhes sobre estas heurísticas e o seu desempenho computacional consultar a referência [11]). Quando não é possível obter a solução óptima através de algoritmos exactos, para avaliar se uma solução obtida através de uma heurística é a solução óptima do problema é necessário compará-la com o limite inferior desse problema. Se a solução obtida é igual ao limite inferior então é óptima, caso contrário não é possível concluir nada. Daí, a importância de desenvolver bons algoritmos para determinar o limite inferior dos problemas.

Mais recentemente, algumas metaheurísticas têm sido propostas para tratar o CARP [16,17,18,20,21]. Estas metaheurísticas começam com uma solução aproximada, em geral obtida através de uma das heurísticas referidas no parágrafo anterior, sendo progressivamente melhorada com base em metodologias do tipo *tabu search*, *genetic algorithms* ou *ant colony system*. De um modo geral, as soluções obtidas pelas metaheurísticas têm um custo global bastante inferior às obtidas pelas heurísticas, embora o tempo computacional seja, em geral, mais elevado. Das metaheurísticas conhecidas da literatura para tratar o CARP, a que parece obter melhor desempenho é a desenvolvida por Lacomme *et al.* [20].

De todas as heurísticas referidas no penúltimo parágrafo a heurística *path-scanning*, introduzida pela primeira vez por Golden *et al.* [14], é uma das que obtém melhor desempenho em termos da qualidade das soluções obtidas e do tempo computacional necessário para as obter. Nesta heurística, os circuitos são construídos um de cada vez com base em determinados critérios de optimização. Cada problema é resolvido 5 vezes, utilizando cada um dos 5 critérios propostos por Golden *et al.* [14], e no final é seleccionada a melhor solução entre as 5 obtidas. Mais tarde Pearn [26] propôs uma modificação que consiste em seleccionar aleatoriamente em cada iteração um dos 5 critérios. Cada problema é resolvido 30 vezes e no final a melhor solução é seleccionada.

Neste relatório de investigação apresentam-se alguns melhoramentos introduzidos na heurística *path-scanning*, designadamente, 4 novos critérios, 3 novas funções de probabilidade (para a referida modalidade da escolha aleatória dos critérios) e uma nova regra impondo que apenas sejam servidos arcos no interior de uma elipse no final de cada circuito.

2. Descrição das Modificações Introduzidas

2.1 Novos Critérios

O procedimento utilizado na heurística *path-scanning* consiste em construir um circuito de cada vez baseado em determinados critérios de optimização. Para construir cada um dos circuitos vão se adicionando, um de cada vez, os arcos com serviço mais promissores segundo o critério seleccionado, até atingir a restrição de capacidade imposta. Cada percurso entre arcos com serviço ou entre o depósito e os arcos com serviço é obtido através do caminho mais curto. Cada problema é resolvido para cada um dos critérios individualmente e no final é seleccionada a melhor solução obtida.

Golden *et al.* [14] apresenta 5 critérios. Considerando um caminho já construído desde o nó v_0 (depósito) até ao nó $v_i, i = 1, \dots, n$, o próximo arco a ser servido (v_i, v_j) é escolhido com base nos seguintes critérios:

1. o quociente entre o custo c_{ij} e o serviço a prestar q_{ij} é minimizado;
2. o quociente entre o custo c_{ij} e o serviço a prestar q_{ij} é maximizado;
3. o custo de circulação entre o nó v_j e o nó v_0 é minimizado;
4. o custo de circulação entre o nó v_j e o nó v_0 é maximizado;
5. se a carga do veículo é inferior a metade da sua capacidade, selecciona-se o critério 4; caso contrário selecciona-se o critério 3.

O primeiro novo critério aqui proposto baseia-se no procedimento híbrido utilizado no critério 5 (proposto por Golden *et al.* [14]). Neste caso, o próximo arco a ser servido (v_i, v_j) é escolhido com base na seguinte regra:

6. se a carga do veículo é inferior a metade da sua capacidade, selecciona-se o critério 2; caso contrário selecciona-se o critério 1.

Para descrever os restantes novos critérios considere-se (v_n, v_m) o último arco servido no circuito e (v_p, v_i) o arco para o qual o nó v_i é o mais próximo de v_m com pelo menos um arco incidente ainda não servido. A escolha do próximo arco a servir (v_i, v_j) baseia-se nos seguintes critérios:

7. o custo c_{ij} mais o custo de circulação entre o nó v_j e o nó v_p é minimizado;
8. o custo c_{ij} mais o custo de circulação entre o nó v_j e o nó v_p é maximizado;
9. se a carga do veículo é inferior a metade da sua capacidade, selecciona-se o critério 8; caso contrário selecciona-se o critério 7.

Considerando por exemplo o critério 7, se o “custo” significar “comprimento” então o critério assegura que o veículo não se afasta muito do nó v_i o qual poderá conter ainda mais arcos incidentes para servir.

2.2 Regra da Elipse

Tendo por objectivo que os veículos no final de cada circuito (quando a sua carga se aproxima da sua respectiva capacidade de transporte) não se afastem demasiado do depósito, impõe-se que apenas sejam servidos arcos no interior de uma elipse, na qual os focos são representados pelo nó final do último arco servido e pelo próprio depósito. Este procedimento origina, em geral, mais circuitos mas permite reduzir o custo total de circulação (convém lembrar que o número de circuitos é uma variável de decisão do problema). Seja (v_n, v_m) o último arco servido no circuito, se a carga do veículo é maior que αW , com $0 < \alpha < 1$, então o próximo arco a ser servido (v_i, v_j) deve ser o que está mais próximo de (v_n, v_m) satisfazendo a seguinte condição:

$$SP(v_m, v_i) + c_{ij} + SP(v_j, v_0) \leq \mu SP(v_m, v_0), \quad (1)$$

onde, por exemplo, $SP(v_m, v_i)$ representa o caminho mais curto entre os nós v_m e v_i . Depois de efectuar vários testes em redes experimentais, concluiu-se que os valores dos parâmetros envolvidos que conduzem aos melhores resultados são $\alpha = 0.7$ e $\mu = 2$.

Se o “custo” significar “comprimento” e se considerar-mos teoricamente que o caminho mais curto entre os nós v_m e v_0 é uma linha recta, então a condição (1) impõe que o veículo apenas sirva arcos situados no interior da elipse representada na Figura 1, onde $c_2 + c_3 = 2c_1$.

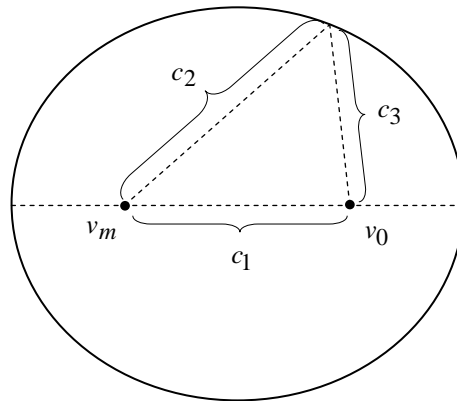


Figura 1 – Elipse definida pela condição (1).

2.3 Novas Funções de Probabilidade

Na versão original da heurística *path-scanning* cada problema é resolvido para cada um dos 5 critérios de cada vez e no final é seleccionada a melhor solução obtida. Mais tarde Pearn [26] propôs uma modificação que consiste em seleccionar aleatoriamente e em cada iteração um dos 5 critérios. Cada problema é resolvido k vezes e no final é seleccionada a melhor solução obtida. Nesse artigo são

sugeridas várias funções de probabilidade para seleccionar os critérios, tais como, (0.2,0.2,0.2,0.2,0.2), (0.6,0.1,0.1,0.1,0.1), (1/3,1/3,1/3,0,0) e (0.1,0.1,0.1,0.35,0.35). Os resultados aí apresentados foram obtidos para $k = 30$ e para a função de probabilidade (0.2,0.2,0.2,0.2,0.2), ou seja, os critérios têm todos a mesma probabilidade de serem seleccionados.

Depois de realizar alguns testes com 23 problemas propostos por DeArmon [12] e 34 problemas propostos por Belenguer e Benavent [3], utilizando a heurística *path-scanning* com os 9 critérios (descritos na secção 2.1) e com a regra da elipse (descrita na secção 2.2), identificou-se o melhor critério para cada problema. Esta informação, apresentada na Tabela 1, mostra, por exemplo, que o critério 5 foi o melhor para 36.8% dos 57 problemas testados. Baseados nestes resultados definiram-se as seguintes funções de probabilidade: $F_1 = 1/9(1,1,1,1,1,1,1,1,1)$, $F_2 = 1/57(9,3,1,8,21,4,2,2,7)$, $F_3 = 1/669(81,9,1,64,441,16,4,4,49)$ e $F_4 = 1/10953(729,27,1,512,9261,64,8,8,343)$.

A função de probabilidade F_1 significa que cada um dos 9 critérios tem igual probabilidade de ser seleccionado. As restantes funções de probabilidade F_2 , F_3 e F_4 foram obtidas com base na linha “Totais” da Tabela 1 e têm por objectivo aumentar a probabilidade de serem seleccionados os melhores critérios. Considerando f_i o número de problemas para os quais o critério i foi o melhor, então a probabilidade de seleccionar este critério é dada por: $f_i / \sum f_i$ considerando a função de probabilidade F_2 ; $f_i^2 / \sum f_i^2$ considerando a função de probabilidade F_3 ; e $f_i^3 / \sum f_i^3$ considerando a função de probabilidade F_4 .

3. Estrutura de Dados

A estrutura utilizada para armazenar as redes teóricas utilizadas nos testes baseia-se em arcos orientados. A vantagem está na sua fácil adaptação a problemas reais, que será numa fase posterior o objectivo destes estudos. De facto, em redes urbanas algumas mudanças de direcção são frequentemente proibidas, tais como, inversões de marcha ou viragens à esquerda. No exemplo representado na Figura 2 não é possível vindo do arco a_1 seguir para o arco a_4 . Se a rede estivesse armazenada numa estrutura baseada em nós, para calcular o caminho mais curto entre o nó s e o nó t seria necessário introduzir nós artificiais [10], porque o nó 1 apareceria mais de uma vez no caminho mais curto, como se pode ver na Figura 3. Este processo de introdução de nós artificiais pode tornar-se mais complexo quando aumentam as proibições associadas a um nó e o número de arcos que a ele chegam ou dele partem. Se a rede estiver armazenada numa estrutura baseada em arcos orientados não ocorrem problemas deste tipo, pois cada arco aparece apenas uma vez no caminho mais curto, como se pode ver na Figura 4.

Figure 2 – Exemplo de uma rede urbana com restrições de mudança de direcção.

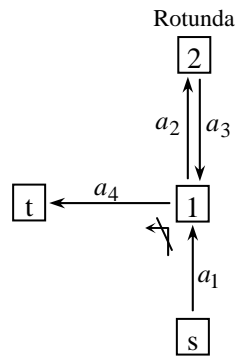


Figura 3 – Caminho mais curto utilizando uma estrutura baseada em nós.

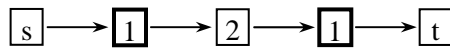
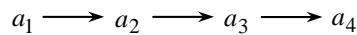


Figura 4 – Caminho mais curto utilizando uma estrutura baseada em arcos orientados.



Os caminhos mais curtos entre todos os arcos orientados foram calculados utilizando o algoritmo de Dial [13] adaptado para arcos. Para cada problema armazenou-se temporariamente o custo e o caminho em duas matrizes de dimensão igual ao número de arcos orientados. Este procedimento melhorou o tempo computacional especialmente para a heurística com escolha aleatória do critério, pois neste caso cada problema foi resolvido 1000, 10000 e 20000 vezes, calculando-se apenas uma vez os caminhos mais curtos entre todos os arcos orientados.

4. Resultados Computacionais

A heurística *path-scanning* foi testada num computador Pentium III a 1 Ghz e em 57 redes experimentais [30] (23 problemas propostos por DeArmon [12] mais 34 problemas propostos por Belenguer e Benavent [3]). O primeiro conjunto de problemas varia entre 7 e 27 nós e entre 11 e 55 arcos não orientados (os problemas 8 e 9 foram removidos dos 25 originais, pois contêm inconsistências [18,20,26]). O segundo conjunto de problemas varia entre 24 e 50 nós e entre 34 e 97 arcos não orientados. Em todos os problemas é necessário prestar serviço em todos os arcos. Estes dois conjuntos de problemas são os geralmente utilizados por outros autores para testar os seus algoritmos relacionados com o CARP.

De seguida apresentam-se as notações que irão ser usadas nesta secção:

- $|E|$: número de arcos não orientados;
- TD: quantidade total de serviço na rede;
- W: capacidade dos veículos (frota homogénea);
- LB: limite inferior para o problema;
- BK: melhor solução conhecida;
- PSG: heurística *path-scanning* implementada por nós tal como foi descrita originalmente por Golden *et al.* [14] (com 5 critérios);
- PSGM: heurística *path-scanning* modificada por nós com os 9 critérios (os 5 originais juntamente com os 4 novos) e a regra da elipse;
- PSP: heurística *path-scanning* implementada por nós tal como foi descrita originalmente por Pearn [26] (cada um dos 5 critérios propostos por Golden *et al.* [14] é seleccionado aleatoriamente com a mesma probabilidade);
- PSPM: heurística *path-scanning* aleatória implementada por nós (cada um dos 9 critérios é seleccionado aleatoriamente de acordo com a nova função de probabilidade F_2 e é aplicada a regra da elipse);
- CSP: heurística *construct and strike* modificada por Pearn [26] (uma das melhores heurísticas);
- CARPET: metaheurística proposta por Hertz *et al.* [18], baseada em *tabu search*;
- GA: metaheurística proposta por Lacomme *et al.* [20], baseada em algoritmos genéticos (uma das melhores metaheurísticas);
- TSCS: metaheurística proposta por Greistorfer [17], baseada em *tabu search*;
- ACO: metaheurística proposta por Lacomme *et al.* [21], baseada em *ant colony system*.

Relativamente à heurística PSGM, a Tabela 1 apresenta o número total de problemas para os quais cada um dos critérios obteve a melhor solução. É possível verificar que o critério 5 foi claramente superior aos restantes (a melhor solução foi obtida com este critério para 36.8% do 57 problemas) e que o critério 3 foi o pior (a melhor solução foi obtida com este critério para apenas 1.8% do 57 problemas). Para 26.3% dos 57 problemas a melhor solução foi obtida com os novos critérios 6, 7, 8 e 9.

Tabela 1 – Número de problemas para os quais o critério foi o melhor.

	Critérios propostos por Golden <i>et al.</i>					Novos critérios			
	Crit.1	Crit.2	Crit.3	Crit.4	Crit.5	Crit.6	Crit.7	Crit.8	Crit.9
23 problemas de DeArmon	7	0	1	3	5	2	0	2	3
34 problemas de Belenguer e Benavent	2	3	0	5	16	2	2	0	4
Totais	9	3	1	8	21	4	2	2	7
Totais em percentagem	15.8	5.3	1.8	14.0	36.8	7.0	3.5	3.5	12.3

No sentido de avaliar o desempenho da heurística *path-scanning* aleatória para as funções de probabilidade F_1 , F_2 , F_3 e F_4 (descritas na secção 2.3), cada problema foi resolvido 1000, 10000 e 20000 vezes, para cada função de probabilidade, e no final foi seleccionada a melhor solução obtida. O algoritmo termina quando é encontrada a solução óptima do problema (ou seja, quando a solução obtida é igual ao limite inferior) ou quando o número de iterações atinge o limite. Na Tabela 2, os valores nas colunas F_1 , F_2 , F_3 e F_4 , excluindo a última linha “Totais”, representam o total do desvio médio em percentagem para a melhor solução conhecida. Este procedimento de comparar a solução obtida com a melhor solução conhecida em vez de a comparar com o limite inferior tem sido adoptado nos artigos mais recentes. Na última coluna apresentam-se os tempos computacionais médios para cada problema.

Tabela 2 – Desempenho das funções de probabilidade.

	Número máximo de iterações	Funções de probabilidade				Tempo comput. médio* (seg)
		F_1	F_2	F_3	F_4	
23 problemas de DeArmon	1000	1.20	1.09	1.54	1.79	1.3
	10000	0.85	0.81	0.98	1.58	10.3
	20000	0.68	0.70	0.89	1.35	20.2
34 problemas de Belenguer e Benavent	1000	4.96	4.22	4.04	4.70	6.5
	10000	3.42	2.87	2.93	3.36	62.6
	20000	3.33	2.41	2.29	3.17	120.0
Totais		14.4	12.1	12.7	16.0	

* Tempo computacional médio para cada problema

Como é possível observar da última linha “Totais” o melhor desempenho global foi obtido pela função de probabilidade F_2 . Esta função de probabilidade melhorou o desvio médio para a melhor solução conhecida em 2.3% (14.4%-12.1%) relativamente a F_1 e em 0.6% (12.7%-12.1%) relativamente a F_3 . A função de probabilidade F_4 foi claramente a pior. Como esperado, a relação entre o número máximo de iterações e o tempo computacional médio é proporcional.

Nas Tabelas 3 e 4 apresentam-se os resultados obtidos pela heurística original implementada por nós (PSG e PSP – com escolha aleatória dos critérios) e os resultados obtidos pela heurística modificada por nós (PSGM e PSPM – com escolha aleatória dos critérios). A heurística PSG foi implementada por nós exactamente como Golden *et al.* [14] a descreve, no entanto os resultados obtidos são algo diferentes (este autor apenas testou a heurística para os 23 problemas propostos por DeArmon [12]). Para o desvio médio relativamente à melhor solução conhecida, nós obtivemos 10.51%, como é possível observar na Tabela 3, e Golden *et al.* [14] obtiveram 7.34%. Lacomme *et al.* [21] também implementaram a heurística *path-scanning* original e obtiveram resultados semelhantes aos nossos (para os 23 problemas propostos por DeArmon [12] obtiveram 10.08% e nós 10.51% e para os 34 problemas propostos por Belenguer e Benavent [3] obtiveram 16.14% e nós 15.55%).

Tabela 3 – Resultados computacionais para os 23 problemas propostos por DeArmon [12].

	E	TD	W	LB	BK	Heurísticas						Metaheurísticas					
						PSG	PSGM	PSP	seg	PSPM	seg	CSP	CARPET	seg	GA	ACO	seg
1	22	22	5	316	316 <i>b)</i>	350	316	316	0.7	316	0.0	323	316	17.1	316	316	0.2
2	26	26	5	339	339 <i>d)</i>	366	362	339	8.4	339	0.1	345	339	28.0	339	339	0.4
3	22	22	5	275	275 <i>c)</i>	293	279	275	5.6	275	0.2	275	275	0.4	275	275	0.2
4	19	19	5	287	287 <i>c)</i>	287	287	287	0.1	287	0.0	287	287	0.5	287	287	0.1
5	26	26	5	377	377 <i>d)</i>	438	383	383	10.2	383	12.1	386	377	30.3	377	377	0.2
6	22	22	5	298	298 <i>d)</i>	324	316	312	7.7	298	2.7	315	298	4.6	298	298	0.3
7	22	22	5	325	325 <i>b)</i>	363	350	325	0.1	325	0.0	325	325	0.0	325	325	0.0
10	46	249	27	344	348 <i>d)</i>	463	384	381	32.4	364	39.5	366	352	330.6	350	353	14.7
11	51	258	27	303	303 <i>e)</i>	390	341	326	36.7	316	44.9	346	317	292.2	303	306	76.1
12	25	37	10	275	275 <i>c)</i>	295	295	275	0.0	275	0.4	275	275	8.4	275	275	0.3
13	45	225	50	395	395 <i>d)</i>	432	432	410	27.2	395	7.8	406	395	12.4	395	395	4.1
14	23	212	35	450	458 <i>d)</i>	581	510	532	11.0	474	12.3	645	458	111.8	458	458	0.2
15	28	245	41	536	538 <i>e)</i>	593	552	540	13.2	544	15.4	544	544	13.1	540	544	0.3
16	21	89	21	100	100 <i>c)</i>	105	100	102	7.6	100	0.0	102	100	2.6	100	100	0.3
17	21	112	37	58	58 <i>a)</i>	62	58	58	0.3	58	0.0	58	58	0.0	58	58	0.1
18	28	116	24	127	127 <i>c)</i>	133	129	127	3.4	127	0.1	127	127	9.2	127	127	1.5
19	28	168	41	91	91 <i>a)</i>	91	91	91	0.0	91	0.0	91	91	0.0	91	91	0.1
20	36	153	37	164	164 <i>c)</i>	177	164	172	17.9	164	0.0	164	164	1.5	164	164	0.3
21	11	66	27	55	55 <i>c)</i>	57	55	55	0.0	55	0.0	63	55	1.1	55	55	0.1
22	22	107	27	121	121 <i>d)</i>	132	124	121	7.2	123	10.3	123	121	51.5	121	121	10.6
23	33	154	27	156	156 <i>c)</i>	176	158	159	16.6	156	9.5	156	156	6.1	156	156	9.2
24	44	205	27	200	200 <i>c)</i>	213	205	202	28.3	202	33.0	200	200	18.3	200	200	27.4
25	55	266	27	233	233 <i>c)</i>	251	241	240	37.8	235	48.8	233	235	186.3	235	235	2.1
Desvio médio para BK (%)						10.51	3.84	2.46		0.81		4.10	0.34		0.08	0.19	
Pior desvio para BK (%)						33.05	12.54	16.16		4.60		40.83	4.62		0.86	1.44	
Número de soluções óptimas						2	7	11		15		11	18		19	18	
Número de melhores soluções						2	7	11		15		11	19		20	19	
Tempo médio (seg)						0.0	0.0		11.8		10.3		49.0	21.0		6.5	

Os LB foram retirados de Hertz *et al.* [18], excepto o do problema 14 que foi retirado de Amberg e Vob [1].

As BK foram obtidas pela primeira vez por: a) Christofides [9], b) Golden *et al.* [14], c) Pearn [26], d) Hertz *et al.* [18] e e) Lacomme *et al.* [20] (alguns resultados referentes a d) e e) são diferentes das respectivas colunas “CARPET” e “GA” porque foram obtidos para diferentes valores dos parâmetros envolvidos).

Como é possível observar da Tabela 3, para os 23 problemas propostos por DeArmon [12] a heurística PSGM melhorou o desvio médio para a melhor solução conhecida em 6.67% (10.51%-3.84%), relativamente à heurística PSG e em 0.26% (4.10%-3.84%) relativamente à heurística CSP (uma das melhores heurísticas conhecidas). Os resultados obtidos para os 34 problemas propostos por Belenguer e Benavent [3], Tabela 4, mostram que a heurística PSGM melhorou o desvio médio para a melhor solução conhecida em 5.02% (15.55%-10.53%) relativamente à heurística PSG.

No sentido de comparar o desempenho das heurísticas PSP e PSPM, apresentam-se nas Tabelas 3 e 4 os resultados obtidos por ambas para 10000 iterações. Não se apresentam os resultados obtidos para 20000 iterações pois, como é possível observar na Tabela 2, o tempo computacional duplica e os ganhos em termos do desvio médio para a melhor solução conhecida não são muito significativos. Os resultados referentes à heurística PSPM foram obtidos para a função de probabilidade F_2 , pois como se referiu anteriormente, é aquela que apresenta o melhor desempenho global.

Tabela 4 – Resultados computacionais para os 34 problemas propostos por Belenguer e Benavent [3].

	E	TD	W	LB	BK	Heurísticas						Metaheurísticas				
						PSG	PSGM	PSP	seg	PSPM	seg	CARPET	seg	GA	ACO	seg
1.A	39	358	200	173	173 a)	188	179	173	2.3	173	0.2	173	0.1	173	173	0.1
1.B	39	358	120	173	173 a)	204	186	180	21.0	179	24.9	173	50.2	173	173	57.5
1.C	39	358	45	235	245 a)	311	281	273	24.9	259	28.7	245	506.1	245	245	6.6
2.A	34	310	180	227	227 a)	250	246	231	16.2	227	1.5	227	0.9	227	227	0.4
2.B	34	310	120	259	259 a)	284	268	268	16.3	262	18.1	260	70.7	259	259	0.7
2.C	34	310	40	455	457 b)	516	516	483	17.5	487	19.6	494	171.6	462	463	14.8
3.A	35	137	80	81	81 a)	87	87	82	16.6	82	18.7	81	4.2	81	81	0.3
3.B	35	137	50	87	87 a)	99	99	92	17.0	88	19.6	87	15.1	87	87	6.4
3.C	35	137	20	138	138 b)	159	154	141	19.4	141	22.5	138	225.8	138	138	26.9
4.A	69	623	225	400	400 a)	440	440	414	60.7	406	70.9	400	153.5	400	400	13.3
4.B	69	623	170	412	412 a)	487	467	446	61.6	416	71.8	416	410.1	412	412	22.0
4.C	69	623	130	428	428 c)	510	510	463	68.4	447	73.7	453	379.7	428	436	116.9
4.D	69	623	75	520	530 c)	675	602	604	68.3	566	79.3	556	1265.9	541	546	239.1
5.A	65	614	220	423	423 a)	476	482	439	51.6	427	60.4	423	20.6	423	423	44.1
5.B	65	614	165	446	446 a)	496	464	471	52.2	450	61.0	448	224.3	446	446	17.9
5.C	65	614	130	469	474 a)	544	520	498	52.9	481	61.7	476	288.7	474	474	124.2
5.D	65	614	75	571	581 c)	719	665	641	56.4	629	65.8	607	1214.7	581	593	58.6
6.A	50	451	170	223	223 a)	267	249	227	31.5	223	2.9	223	21.1	223	223	0.8
6.B	50	451	120	231	233 a)	276	247	245	31.8	242	38.4	241	146.0	233	233	4.2
6.C	50	451	50	311	317 b)	385	367	351	35.3	335	41.1	329	461.7	317	317	59.4
7.A	66	559	200	279	279 a)	326	325	285	52.9	283	60.3	279	35.7	279	279	7.9
7.B	66	559	150	283	283 a)	331	321	307	52.8	290	62.8	283	0.1	283	283	7.9
7.C	66	559	65	333	334 a)	421	378	373	54.2	353	67.5	343	658.2	334	334	210.6
8.A	63	566	200	386	386 a)	426	417	399	48.4	389	56.1	386	20.8	386	386	27.0
8.B	63	566	150	395	395 a)	479	428	420	48.5	404	57.2	401	441.5	395	401	38.7
8.C	63	566	65	517	527 c)	578	578	571	49.6	559	58.6	533	798.9	533	550	59.5
9.A	92	654	235	323	323 a)	369	353	338	98.5	330	113.9	323	154.5	323	324	407.9
9.B	92	654	175	326	326 a)	359	355	350	99.7	332	119.9	329	324.6	326	327	184.6
9.C	92	654	140	332	332 a)	394	373	361	105.7	344	118.7	332	305.9	332	337	318.0
9.D	92	654	70	382	391 c)	485	435	446	105.1	417	129.0	409	1914.8	391	402	134.4
10.A	97	704	250	428	428 a)	450	454	443	108.3	432	124.2	428	29.9	428	428	383.3
10.B	97	704	190	436	436 a)	475	463	453	108.5	440	125.3	436	99.9	436	438	291.7
10.C	97	704	150	446	446 a)	486	480	472	112.4	453	126.5	451	506.6	446	450	233.7
10.D	97	704	75	524	530 c)	626	595	576	111.7	555	128.9	544	847.2	535	544	318.1
Desvio médio para BK (%)						15.55	10.53	6.15		2.87		1.39		0.15	0.68	
Pior desvio para BK (%)						27.36	19.16	14.07		8.26		8.10		2.08	4.36	
Número de soluções óptimas						0	0	1		3		16		23	16	
Número de melhores soluções						0	0	1		3		17		30	21	
Tempo médio (seg)						0.0	0.0		55.2		62.6		346.2	120.0		101.1

Os LB foram retirados de Hertz *et al.* [18].

As BK foram obtidas pela primeira vez por: a) Belenguer e Benavent [3], b) Hertz *et al.* [18] e c) Lacomme *et al.* [20] (alguns resultados referentes a b) e c) são diferentes das respectivas colunas “CARPET” e “GA” porque foram obtidos para diferentes valores dos parâmetros envolvidos).

Observando a Tabela 3, verifica-se que a heurística PSPM melhorou o desvio médio para a melhor solução conhecida em 1.65% (2.46%-0.81%) relativamente à heurística PSP e em 3.29% (4.10%-0.81%) relativamente à heurística CSP. De facto, a heurística PSPM é melhor que a CSP em todos os itens avaliados. Por exemplo, a heurística PSPM obtém a solução óptima para 65.2% dos 23 problemas enquanto a heurística CSP apenas para 47.8%. Observando a Tabela 4, verifica-se que a heurística PSPM melhorou o desvio médio para a melhor solução conhecida em 3.28% (6.15%-2.87%) relativamente à heurística PSP.

Como seria de esperar, quando se compara a heurística PSPM com as metaheurísticas os resultados são piores. Como a metaheurística TSCS, que apenas foi testada para os 23 problemas propostos por DeArmon [12], não melhora os resultados obtidos por qualquer uma das outras metaheurísticas, os seus

resultados não são apresentados na Tabela 3. Observando esta tabela, verifica-se que a heurística piorou o desvio médio para a melhor solução conhecida em 0.47% (0.81%-0.34%) relativamente à metaheurística CARPET e em 0.73% (0.81%-0.08%) relativamente à metaheurística GA (umas das melhores metaheurísticas conhecidas). Enquanto a heurística PSPM obtém a solução ótima para 65.2% dos 23 problemas, as metaheurísticas CARPET e GA obtém a solução ótima para 78.2% e 82.6%, respectivamente.

Observando a Tabela 4, verifica-se que a heurística PSPM piorou o desvio médio para a melhor solução conhecida em 1.48% (2.87%-1.39%) relativamente à metaheurística CARPET e em 2.72% (2.87%-0.15%) relativamente à metaheurística GA. Enquanto a heurística PSPM obtém a solução ótima apenas para 8.8% dos 34 problemas, as metaheurísticas CARPET e GA obtém a solução ótima para 47.1% e 67.6%, respectivamente.

Como é possível observar nas Tabelas 3 e 4, o tempo computacional médio para cada problema associado à heurística PSPM (o computador utilizado foi um Pentium III a 1 Ghz) é praticamente metade do associado à metaheurística GA (o computador utilizado foi um PC a 500 Mhz). Comparando com a metaheurística CARPET (o computador utilizado foi um Silicon Graphics Indigo2 a 195 Mhz), o tempo computacional é sensivelmente um quinto. Comparando com a metaheurística ACO (o computador utilizado foi um Pentium III a 800 Mhz) o nosso tempo computacional é superior para os 23 problemas propostos por DeArmon [12] e praticamente metade para os 34 problemas propostos por Belenguer e Benavent [3].

Um dos itens que raramente é referido nos artigos que descrevem algoritmos para tratar o CARP diz respeito ao número de circuitos gerados para obter cada uma das soluções. Embora nestes 57 problemas teóricos não exista nenhum custo adicional associado ao circuito, em problemas reais esse custo pode ocorrer (representando, por exemplo, o tempo que o veículo demora a descarregar ou um custo monetário associado à descarga do veículo) e, nesse caso, seria conveniente estudar com mais detalhe esta variável de decisão do problema. Observando a Tabela 5, verifica-se que a heurística PSGM gera mais circuitos que a heurística PSG (verifica-se um aumento de 6.2% para os 57 problemas) mas reduz o desvio médio para a melhor solução conhecida (verifica-se uma redução de 44.9% para os 57 problemas). A heurística PSPM gera mais circuitos que a heurística PSP (verifica-se um aumento de 3.3% para os 57 problemas) mas reduz o desvio médio para a melhor solução conhecida (verifica-se uma redução de 57.3% para os 57 problemas). A heurística PSPM gera mais 1 circuito que a metaheurística TSCS (única metaheurística para a qual o autor indica o número de circuitos gerados) e aumenta o desvio médio para a melhor solução conhecida.

Tabela 5 – Número total de circuitos gerados.

	23 problemas de DeArmon		34 problemas de Belenguer e Benavent	
	Nº total de circuitos	Desvio médio para a melhor solução	Nº total de circuitos	Desvio médio para a melhor solução
PSG	134	10.51	174	15.55
PSGM	142	3.84	185	10.53
PSP	133	2.46	174	6.15
PSPM	138	0.81	179	2.87
TSCS*	137	0.34	—	—

* Esta metaheurística apenas foi testada para os 23 problemas de DeArmon.

5. Conclusões

Neste relatório de investigação são apresentadas algumas modificações propostas para a heurística *path-scanning*. Estas modificações resumem-se à introdução de 4 novos critérios (a adicionar aos 5 critérios originalmente propostos por Golden *et al.* [14]), 3 novas funções de probabilidade para escolha aleatória dos critérios e uma nova regra impondo que apenas sejam servidos arcos no interior de uma elipse no final de cada circuito. Dos testes efectuados em 57 problemas experimentais (estes são os problemas geralmente utilizados para testar os algoritmos relacionados com o CARP), conclui-se que a heurística PSPM obtém melhores resultados que a heurística CSP (uma das melhores heurísticas conhecidas) em todos os itens avaliados.

Como esperado, quando comparada com as metaheurísticas CARPET, GA e ACO, a heurística PSPM obtém piores resultados. De facto, estas metaheurísticas que iniciam o processo com uma ou mais soluções obtidas por heurísticas, tais como a *path-scanning*, utilizam procedimentos computacionalmente mais complexos com o objectivo de melhorar essas soluções iniciais. Assim, comparando a complexidade computacional da heurística PSPM e das metaheurísticas CARPET, GA e ACO, com os resultados obtidos por cada um destes algoritmos, pode concluir-se, em termos globais, que a heurística tem um bom desempenho.

As vantagens associadas à utilização da heurística PSPM são, por um lado, a possibilidade de poder fornecer melhores soluções iniciais para as metaheurísticas e, por outro lado, o facto de ser simples de implementar e flexível na introdução de possíveis restrições adicionais quando aplicada a problemas reais. Uma eventual desvantagem está no facto de poder gerar um maior número de circuitos como consequência da aplicação da regra da elipse. Não entanto não é possível comparar este item com a maioria das metaheurísticas citadas neste relatório pois não é habitual os autores referirem o número total de circuitos gerados.

6. Bibliografia

- [1] Amberg, A., Vob, S., “A Hierarchical Relaxations Lower Bound for the Capacitated Arc Routing Problem”. Sprague, R. H. (Ed.), Proceedings of the 35th Annual Hawaii International Conference on System Sciences, IEEE, Piscataway, vol. DTIST02, 1-10 (2002).
- [2] Assad, A.A., Pearn, W.L., Golden, B.L., “The Capacitated Chinese Postman Problem. Lower Bounds and Solvable Cases”, *Amer. Journal Math. and Management Sci.* 7, 63-88 (1987).
- [3] Belenguer, J.M., Benavent, E., “A Cutting Plane Algorithm for the Capacitated Arc Routing Problem”, Research Report, Department of Statistics and Operational Research, University of Valência, Espanha (1997).
- [4] Beltrami, E., Bodin, L.D., “Networks and Vehicle Routing for Municipal Waste Collection”, *Networks* 4, 65-94 (1974).
- [5] Benavent, E., Campos, V., Corberán, A., Mota, E., “The Capacitated Arc Routing Problem: Lower Bounds”, *Networks* 22, 669-690 (1992).
- [6] Bodin, L.D., Kursh, S.J., “A Computer-Assisted System for the Routing and Scheduling of Street Sweepers”, *Operations Research* 26, 525-537 (1978).
- [7] Bodin, L.D., Kursh, S.J., “A Detailed Description of a Computer System for the Routing and Scheduling of Street Sweepers”, *Computers and Operations Research* 6, 181-198 (1979).
- [8] Chapleau, L., Ferland, J.A., Lapalme, G., Rousseau, J.M., “A Parallel Insert Method for the Capacitated Arc Routing Problem”, *Operations Research Letters* 3, 95-99 (1984).
- [9] Christofides, N., “The Optimum Traversal of a Graph”, *Omega* 1, 719-732 (1973).
- [10] Clossey, J., Laporte, G., Soriano, P., “Solving Arc Routing Problems With Turn Penalties”, *Journal of the Operational Research Society* 52, 433-439 (2001).
- [11] Coutinho-Rodrigues, J., Rodrigues, N., Clímaco, J., “Solving an Urban Routing Problem Using Heuristics – A Successful Case Study”, *Belgian Journal of Operations Research, Statistics and Computer Science*, edição especial “Combinatorial Optimization in Applications”, 33, 19-32 (1993).
- [12] DeArmon, J.S., *A Comparison of Heuristics for the Capacitated Chinese Postman Problem*, Master’s Thesis, University of Maryland at College Park, MD, USA, 1981.
- [13] Dial, R.B., “Algorithm 360: Shortest Path Forest With Topological Ordering”, *Communications of the ACM* 12, 632-633 (1969).
- [14] Golden, B.L., DeArmon, J., Baker, E.K., “Computational Experiments With Algorithms for a Class of Routing Problems”, *Computers and Operations Research* 10, 47-59 (1983).
- [15] Golden, B.L., Wong, R.T., “Capacitated Arc Routing Problems”, *Networks* 11, 305-315 (1981).
- [16] Greistorfer, P., “Computational Experiments With Heuristics for a Capacitated Arc routing Problem”, Working paper 32, Department of Business, University of Graz, Graz, Austria (1994).
- [17] Greistorfer, P., “A Tabu Scatter Search Metaheuristic for the Capacitated Arc Routing Problem”, Preprint submitted to Elsevier Science (2002).
http://www.kfunigraz.ac.at/ifwww/pg/downloads/jtscs_r4.pdf.
- [18] Hertz, A., Laporte, G., Mittaz, M., “A Tabu Search Heuristic for the Capacitated Arc Routing Problem”, *Operations Research* 48, 129-135 (2000).

- [19] Hirabayashi, R., Saruwatari, Y., Nishida, N., “Tour Construction Algorithm for the Capacitated Arc Routing Problems”, *Asia-Pacific Journal Operations Research* 9, 155-175 (1992).
- [20] Lacomme, P., Prins, C., Ramdane-Chérif, W., “A Genetic Algorithm for the Capacitated Arc Routing Problem and its Extensions”. In Boers, E.J.W., Gottlieb, J., Lanzi, P.L., Smith, R.E., Cagnoni, S., Hart, E., Raidl, G.R., and Tijink, H. (eds), *Applications of Evolutionary Computing, EvoWorkshops 2001*, vol. 2037 of Lecture Notes in Computer Science, Springer-Verlag, Berlin Heidelberg, 473-483 (2001).
- [21] Lacomme, P., Prins, C., Tanguy, A., “Optimisation par Colonies de Fourmis pour les Tournées sur Arcs”, 4^e Conférence Francophone de MODélisation et SIMulation “Organisation et Conduite d’Activités dans l’Industrie et les Services”, MOSIM’03, Toulouse, França (2003).
- [22] Levy, L.S., *The Walking Line of Travel: An Application of Arc Routing and Partitioning*, PhD Thesis, University of Maryland, EUA, 1987.
- [23] Levy, L.S., “Scheduling the Postal Carriers for the United States Postal Service: An Application of Arc Partitioning and Routing”, *Vehicle Routing: Methods and Studies*, edit. B. L. Golden e H. A. Assad, Elsevier Science Publishers B. V. North-Holland, 359-393 (1988).
- [24] Mourão, M.C., Almeida, M.T., “Lower-Bounding and Heuristic Methods for a Refuse Collection Vehicle Routing Problem”, *European Journal of Operational Research* 121, 420-434 (2000).
- [25] Pearn, W.L., “New Lower bounds for the Capacitated Arc Routing Problem”, *Networks* 18, 181-191 (1988).
- [26] Pearn, W.L., “Approximate Solutions for the Capacitated Arc Routing Problem”, *Computers and Operations Research* 16, 589-600 (1989).
- [27] Stern, H.I., Dror, M., “Routing Electric Meter Readers”, *Computers and Operations Research* 6, 209-223 (1979).
- [28] Ulusoy, G., “The Fleet Size and Mix Problem for Capacitated Arc Routing”, FBE – EM 84/04 Arastima Raporu Research Papers, Bogaziçi Universitesi, Bebek – Istanbul, Turquia (1984).
- [29] Win, Z., *Contribution to Routing Problems*, Ph.D. Dissertation, University of Augsburg, Augsburg, Alemanha, 1987.
- [30] <ftp://matheron.estadi.uv.es/pub/CARP>.