# ALGORITMO EM COLÔNIA DE FORMIGAS COM MULTIFEROMÔNIOS PARA A SOLUÇÃO DO PROBLEMA DO CAIXEIRO ALUGADOR

**Paulo Asconavieta**
Instituto Federal Sul-Rio-grandense
Rua Gonçalves Chaves, 3798, Centro, Pelotas, RS
asconavietaph@hotmail.com

**Marco César Goldbarg**
Universidade Federal do Rio Grande do Norte
Campus Universitário, Lagoa Nova, Natal, RN
gold@dimap.ufrn.br

**Elizabeth Ferreira Gouvêa Goldbarg**
Universidade Federal do Rio Grande do Norte
Campus Universitário, Lagoa Nova, Natal, RN
beth@dimap.ufrn.br

## RESUMO

O presente trabalho relata um estudo algorítmico que visa avaliar a eficácia do uso de multiferomônios nos algoritmos em colônia de formigas na solução do problema do Problema do Caixeiro Alugador ou Car Renter Salesman Problem (CaRS). O problema objeto da pesquisa é constituído basicamente de três decisões independentes e que podem ser objeto de guiamento cooperativo. No trabalho é formalizado um algoritmo clássico em colônia de formigas e uma variante da versão clássica explorando o guiamento da rota do problema e que delega as demais decisões a um guiamento semiguloso. São formalizadas adicionalmente três versões multiferomônio para o problema. Duas versões empregando dois tipos de feromônio e que guiam duas das três decisões do problema. Uma terceira versão guiando todas as decisões através de feromônios. Um experimento computacional comprova a vantagem do uso da estratégia multiferomônio para guiar todas as decisões do problema.

**PALAVARAS CHAVE.** Problema do Caixeiro Alugador, Algoritmo de colônia de formigas, Multiferomônio.

## ABSTRACT

This paper describes an algorithmic study aimed to evaluate the effectiveness of using multipheromone in ant colony algorithms to solve the Car Renter Salesman Problem (CaRS). The problem object of the research consists basically of three independent decisions and may be the subject of cooperative guidance. The paper presents a classic ant colony algorithm and a variant of the classic version exploiting the guidance of the route of the problem, and delegating other decisions to a semigreedy guiding. Additionally, three multipheromone versions are formalized to the problem. Two versions use two types of pheromone and guide two of the three decisions of the problem. A third version guides all decisions through pheromones. A computational experiment demonstrates the advantage of using the multipheromone strategy to guide all decisions of the problem.

**KEYWORDS. Car Renter Salesman Problem. Ant Colony Algorithms, Multi-Pheronome Ant Colony Algorithms.**

## 1. Introdução

O Problema do Caixeiro Viajante (PCV) é um dos clássicos problemas da Otimização Combinatória, consistindo em determinar em um grafo ponderado $G=(N,M)$ onde $N=\{1,...,n\}$ representa o conjunto de vértices do grafo e $M = \{1,...,m\}$ o conjunto de arestas, um ciclo hamiltoniano de menor custo. O PCV é NP-difícil (Garey & Johnson, 1979), sendo um dos problemas de Otimização Combinatória mais intensamente pesquisados. O PCV possui várias e importantes aplicações práticas e diversas variantes (Gutin & Punnen, 2007). O presente trabalho introduz uma nova variante do problema denominada o Problema do Caixeiro Alugador ou *Car Renter Salesman Problem* (CaRS), recentemente descrita na literature em Goldbarg *et al.* (2012). Esse modelo tanto pode representar importantes aplicações na área do aluguel de transporte para turismo quanto em manufatura flexível (Goldbarg *et al*., 2012). Por admitir o PCV como um subcaso elementar, o modelo representa uma variante de complexidade seguramente desafiadora.

Adicionalmente, o presente trabalho aborda uma nova forma de controle para os algoritmos em colônia de formigas através do uso de mais de um tipo de feromônio, técnica presentemente denominada *regulagem multiferomônio*. A técnica pode se mostrar útil na solução do problema proposto tendo em vista o mesmo ser composto por três decisões independentes entre si e que são: a rota a ser percorrida, o carro a ser utilizado em cada trecho da rota e o ponto de troca dos carros. Nesse caso a cooperação exclusivamente baseada em uma das decisões necessárias, tende a ser, para um caso geral, uma estratégia que não contempla aspectos importantes das decisões do problema. A técnica permite a introdução de um mecanismo de controle na colônia capaz de considerar mais de uma possível decisão, conforme se exige no problema abordado.

A Seção 2 do presente trabalho define o Problema do Caixeiro Alugador. A Seção 3 formaliza o algoritmo em colônia de formigas tradicional e a técnica denominada regulagem multiferomônio. Para examinar o potencial da abordagem são desenvolvidos e comparados cinco diferentes algoritmos em colônia de formigas, a Seção 4 descreve estas cinco versões. A Seção 5 apresenta um experimento computacional englobando os algoritmos desenvolvidos visando comprovar o potencial da técnica proposta. A Seção 6 apresenta as conclusões do trabalho.

## 2. O Problema do Caixeiro Alugador

Existem hoje, no mercado mundial, mais de 90 companhias de locação de carros com porte econômico significativo. A importância do negócio de aluguel de carros pode ser medida tanto pelo faturamento do setor como através do porte das companhias prestadoras de serviço. Algumas destas companhias faturam bilhões de dólares (Conrad & Perlut, 2006). Todavia o aluguel dos veículos é apenas uma parte dos custos de transporte. Sobre esses custos se somam pelo menos as despesas com combustível, pedágios e seguros. Embora alguns trabalhos sejam dedicados a esta área, tais como os revisados por Yang et al. (2008), o ponto de vista do cliente não foi objeto de custo nestas publicações. Este trabalho apresenta um modelo relacionado ao ponto de vista do usuário de carros alugados.

Considerando um grafo $G=(N, M, W)$ onde $N=\{1,...,n\}$ representa o conjunto de vértices do grafo, $M =\{1,...,m\}$ o conjunto de arestas e $W=\{1,...,w\}$ o conjunto das distâncias entre os vértices de $G$ ou o comprimento das arestas do conjunto $M$, o problema possui as características listadas a seguir.

Estão disponíveis para o aluguel vários tipos de carro, cada qual possuindo características próprias, ou seja, custos de operação específicos. Tais custos englobam o consumo de combustível, eventuais taxas de pedágio e o valor do aluguel. Como as taxas de pedágio dependem, normalmente, tanto do tipo de carro como da extensão do trecho percorrido e o valor do aluguel pode ser associado ao quilômetro rodado, sem perda de generalidade, pode-se considerar que todos os custos são contabilizados em função de cada carro em um único valor associado ao peso ou cumprimento das arestas $(i,j)$ do grafo $G$.

Um carro alugado em uma dada operadora somente poderá ser devolvido em uma cidade que conta com pelo menos uma agência dessa mesma locadora ou agência conveniada. Consequentemente, não é permitido ao caixeiro alugar um carro de uma dada operadora para

cumprir um trecho de sua rota, se esse carro não puder ser devolvido na última cidade do trecho atribuído a esse carro – se nessa cidade não existir uma agência capacitada para executar seu recebimento.

Sempre que for possível alugar um carro em uma cidade *i* e devolve-lo em uma cidade *j* sendo $i \neq j$, haverá uma sobretaxa relativa ao retorno do carro à sua cidade de origem. O tour é iniciado e concluído na cidade em que o primeiro carro é alugado, a cidade base do caixeiro alugador.

O caso em que o caixeiro realiza seu tour com apenas um carro alugado corresponde ao PCV considerando-se as demais condições de custo associadas apenas ao carro selecionado.

Mesmo carros de características iguais e alugados em uma mesma rede de locação podem ser contratados sob diferentes custos, dependendo da cidade de locação e da negociação de contrato. Portanto, sem perda de generalidade, a designação de aluguel pode ser então eficazmente controlada por decisões associadas aos carros, abstraindo-se as locadoras. Presentemente o conjunto $K=\{1,...,k\}$, $|K|=k$ é o conjunto dos diferentes carros que podem participar da solução. O custo do retorno do carro alugado pode ser independente dos custos da rota.

O objetivo do problema proposto é encontrar o ciclo hamiltoniano que, partindo de um vértice inicial previamente conhecido, minimize a soma dos custos totais de operação dos carros utilizados no trajeto. Os custos totais do deslocamento são compostos por uma parcela que unifica o aluguel e demais despesas de trajetória de cada carro em um valor associado às arestas, e por uma parcela relativa à devolução de um veículo fora da sua cidade base, calculada para cada carro e para cada par de cidades origem / devolução existentes no ciclo.

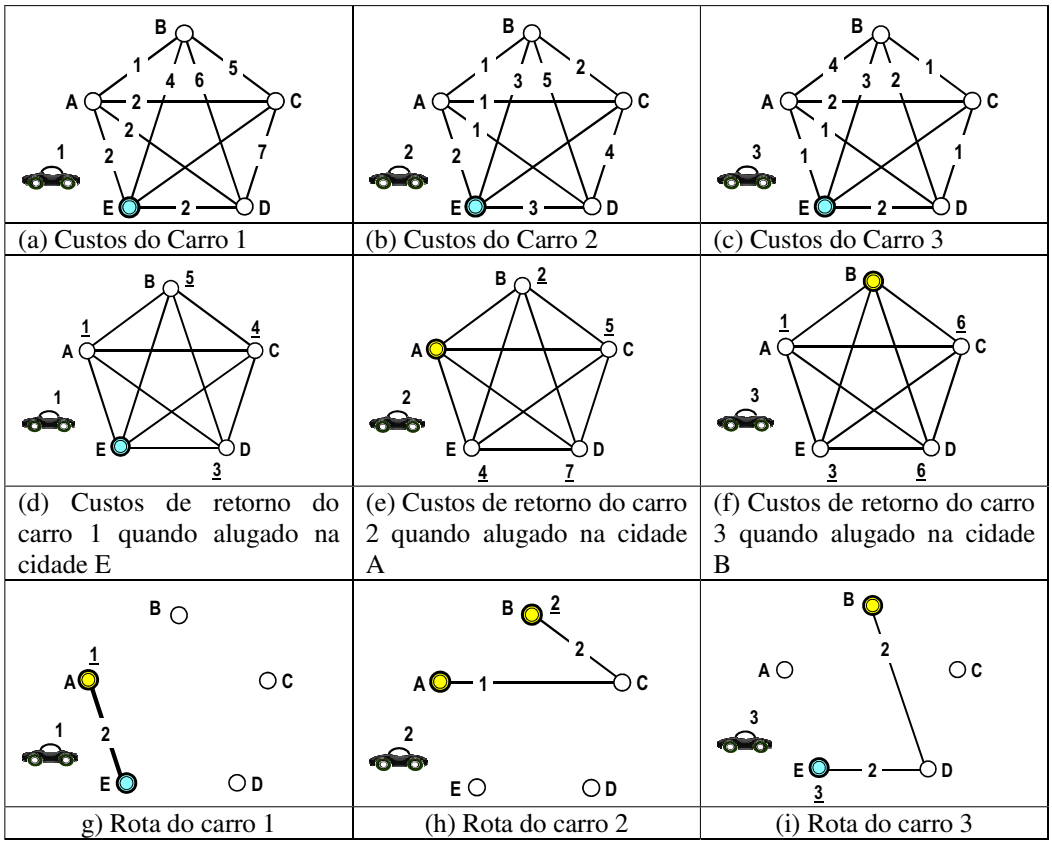


**Figura 1. Custos de rota e de retorno dos carros**

A Figura 1 exemplifica, em um grafo completo de cinco vértices, uma típica instância

do CaRS. No exemplo existem três diferentes carros de aluguel, e os carros estão disponíveis para serem alugados e entregues em todas as cidades do grafo. As Figuras 1(a), (b) e (c) exibem a contabilidade dos custos envolvidos no deslocamento de cada tipo de carro. Observe-se que, diferentemente do ciclo do caixeiro viajante clássico, a solução do CaRS depende da cidade escolhida para o início do tour, cidade base do caixeiro. Esse fato decorre da taxa de retorno poder estar vinculada tanto à cidade que inicia o ciclo, quanto ao próprio sentido de percurso desse ciclo. No exemplo essa cidade é representada pelo vértice E.

As Figuras 1(d), (e) e (f) mostram, para o exemplo alguns dos custos do retorno dos carros a suas bases. Os custos do retorno dos carros aparecem como números sublinhados junto aos vértices. A Figura 1(d) exibe o grafo de retorno do carro 1 quando alugado no vértice E. A Figura 1(e) exibe o grafo de retorno do carro 2 quando alugado no vértice A e a Figura 1(f) o retorno do carro 3 quando alugado no vértice B. No caso geral são conhecidos os custos de retorno de todos os carros quando alugados em qualquer uma das cidades.

Uma solução do problema exemplificado nas Figuras 1 de a-f é exibida nas Figuras 1 de g-h. Essa solução considera um caso em que todos os carros disponíveis são alugados e não há carros alugados mais de uma vez. O custo de percorrer o ciclo, segundo o esquema de solução da Figura 1, corresponde ao custo do caminho E-A para o carro 1, somado ao custo do caminho A-C-B para o carro 2, somado ao custo do caminho B-D-E do carro 3, tem um total de 9 unidades. A esse valor deve se acrescentar o custo dos retornos dos carros a suas bases. No caso do carro 1, o retorno do vértice A ao vértice E, custa uma unidade. Para o caso do carro 2, o retorno do vértice B ao vértice A custa duas unidades e, para o carro 3, o retorno ao vértice B quando o carro é entregue no vértice E custa três unidades. Assim, a solução final custa 15 unidades.

Goldbarg *et al*. (2012) apresentam dois algoritmos para CaRS. O primeiro é um GRASP (*Greedy Randomized Adaptive Search Procedure*) (Feo & Resende, 1995) hibridizado com um VND (*Variable Neighborhood Descent*) (Mladenovic & Hansen, 1997) na fase de busca local. A segunda heurística é um Algoritmo Memético (Moscato, 1989).

## 3. Otimização por Colônia de Formigas

Os algoritmos da classe Colônia de Formigas são baseados em população e inspiram-se no comportamento forrageiro de formigas. Tais algoritmos buscam mimetizar tanto a cooperação dos indivíduos de uma colônia de formigas como os mecanismos que permitem coordenar a atuação de cada formiga.

### 3.1 O Sistema de Formigas Artificiais

O *Ant Colony System* (ACS) é um algoritmo não-determinístico composto de heurísticas construtivas baseadas em uma população de agentes (formigas). Os agentes movem-se simultaneamente e de forma independente construindo estocasticamente soluções com base nas informações da própria heurística e da trilha de feromônio que tenha sido anteriormente formada. As informações heurísticas referem-se às especificidades do problema a ser resolvido. A informação da trilha de feromônio reflete a experiência já obtida durante a busca.

### 3.2 O Pseudocódigo Genérico do Algoritmo Colônia de Formigas

O algoritmo apresentado no Quadro 1 inicia com a distribuição de $k$ formigas em nodos aleatórios do grafo $G=(V,E)$, onde $V$ é o conjunto de vértice $V=\{1,2,..,n\}$ e $E$ é o conjunto de arestas $E=\{1,2,...,m\}$. Cada formiga irá construir uma solução individualmente visitando os diversos nodos do grafo. Começando de um vértice $i$, a formiga escolhe probabilisticamente o próximo nodo $j$ entre os vizinhos exequíveis. A probabilidade da formiga $k$ que está no vértice $i$ de escolher $j$ como seu próximo vértice é calculada pela equação 1.

$$p_{ij}^{k} = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{j \in \aleph_i^k} (\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}, \text{ se } j \in \aleph_i^k \quad (1)$$

Quadro 1 – Pseudocódigo Genérico do Algoritmo Colônia de Formigas

```
Localizar uma formiga em um vértice do grafo G=(V,E)
Para t←1 até o número de iterações (ou colônias)
  Para k←1 até m
    Enquanto a formiga k não construir o caminho S_k
      Selecione o próximo vértice pela regra P_ij^k
    Fim Enquanto
    Calcule a distância L_k do caminho S_k
    Se L_k<L* então
      S*← S_k, L*←L_k
    Fim Se
  Fim Para
  Atualize os feromônios
Fim Para
Retornar S*
```

Na equação 1 $\tau_{ij}$ representa a quantidade de feromônio associado à aresta $(i,j)$ e $\eta_{ij}$ reflete o valor heurístico que representa a atratividade da formiga visitar o vértice $j$ após visitar o nodo $i$. Este valor é inversamente proporcional à distância $d_{ij}$ entre os vértices $i$ e $j$, como mostrado na equação 2.

$$\eta_{ij} = \frac{1}{d_{ij}} \quad (2)$$

Os parâmetros $\alpha$ e $\beta$ são ajustados heuristicamente para determinar a influência do feromônio e da informação heurística. A vizinhança factível da formiga $k$, ou seja, o conjunto de nodos que ainda não foram visitados pela formiga $k$, é representado por $\aleph_i^k$. No final do tour, quando todas as formigas construíram a sua solução, ocorre a atualização do feromônio $\tau_{ij}$ representada em dois eventos: o depósito e a evaporação. Ao longo da trilha $(i,j)$ a formiga deposita na aresta uma quantidade da substância definida conforme a equação 3.

$$\tau_{ij} = \underbrace{(1-\rho)\tau_{ij}}_{\textbf{Evaporação}} + \underbrace{\sum_{k=1}^{m} \Delta\tau_{ij}^{(k)}}_{\textbf{Depósito}} \quad (3)$$

A evaporação evita que acúmulos de feromônios cresçam indefinidamente e possibilita o esquecimento de decisões realizadas no passado e consideradas fracas. O parâmetro $\rho$ é a uma informação heurística referente a taxa de evaporação do feromônio cujo valor está contido no intervalo [0,1]. A quantidade de feromônio depositada pela formiga $k$ na trilha entre $i$ e $j$ é representado por $\Delta\tau_{ij}^{(k)}$ e é obtida segundo a equação 4.

$$\Delta\tau_{ij}^{(k)} = \begin{cases} \dfrac{Q}{L_k}, & se\ (i,j) \in S_k \\ 0, & caso\ contrário \end{cases} \quad (4)$$

O parâmetro $Q$ é uma constante do projeto e $L_k$ é o comprimento total do tour da *k-ésima* formiga, portanto o depósito é realizado caso o trecho $(i,j)$ pertença ao tour construído pela

formiga *k*. O critério de parada pode ser definido por um número máximo de iterações ou pelo evento da estagnação. O algoritmo torna-se estagnado a partir do momento em que todas as formigas convergem para o mesmo percurso em consequência do excessivo crescimento de feromônio nas arestas pertencentes a uma rota subótima.

### 3.3 Algoritmos de Colônia de Formigas com Multiferomônios

A utilização de vários feromônios no desenvolvimento de algoritmos em Colônias de Formigas é recente na literatura. Como regra geral a técnica associa-se ao uso de múltiplas colônias. Em Kawamura *et al.* (2000), uma dos primeiros trabalhos abordando o tema, várias colônias depositam diferentes feromônios, todavia os feromônios apresentam efeitos de atração e de repulsão para as várias colônias. No trabalho de Greenfield (2006) os feromônios das diferentes formigas marcam diferentes cores de uma pintura. Martens (2007) seleciona regras de classificadores através de diferentes formigas e diferentes feromônios. Todavia o efeito acumulado de diferentes feromônios influencia todas as colônias. No trabalho de Xia *et al.* (2008) os feromônios sinalizam atributos dos arcos de uma rede. Na pesquisa de Ou (2008) são propostos feromônios de atualização local e de atualização global. Em Chircop & Buckingham (2011) um mesmo tipo de feromônio é depositado por mais de uma colônia com regras diferentes para cada colônia. No trabalho de Hsin *et al.* (2011) são o utilizados *k* feromônios que se distinguem pelas diferentes velocidades de evaporação. Moncayo-Martínez & Zhang (2010) e Doerner *et al.* (2008), dentre outros, modelam diferentes objetivos através de diferentes feromônios. Devido as características dos modelos sugeridos na literatura, suas possíveis adaptações ao processo de solução do CaRS não se mostra trivial.

## 4. Algoritmos em Colônia de Formiga (ACS) com uso de Multiferomônios para CaRS

O presente item descreve cinco algoritmos em colônia de formigas, F1 a F5, sendo o algoritmo F1 um algoritmo clássico, visando estabelecer o desempenho do método sem a técnica multiferomônio. O Quadro 2 resume as técnicas empregadas para orientar cada uma das três decisões do algoritmo. O símbolo * estabelece que o método utilizado é o mesmo implementado no trabalho de Goldbarg *et al.* (2012).

Quadro 2 - Resumos das estratégias adotadas nas decisões dos algoritmos

| Algoritmos | Rota | Vértices de troca dos carros | Sequência dos Veículos na rota |
|---|---|---|---|
| $F_1$ | Feromônio | Semiguloso* | Semiguloso* |
| $F_2$ | Feromônio | Semiguloso modificado | Semiguloso* |
| $F_3$ | Feromônio | Feromônio | Semiguloso* |
| $F_4$ | Feromônio | Semiguloso modificado | Feromônio |
| $F_5$ | Feromônio | Feromônio | Feromônio |

Os algoritmos desenvolvidos visam avaliar o potencial de melhoria embutido na consideração do histórico associado às várias decisões que são tomadas pelas formigas, bem como a vantagem da consideração simultânea da contabilidade de todos esses históricos. Como alternativa ao processo guiado pelos feromômios o experimento examina a alternativa do uso de decisões semigulosas, na linha do algoritmo GRASP/VNS proposto em Goldbarg *et al* (2012).

### 4.1 Algoritmo ACS - F1

O algoritmo do Quadro 3 mostra o pseudocódigo do algoritmo F1. Os parâmetros de entrada são: o nome da instância, *nomeInstancia*, o número de cidades, *n*, o número de colônias de formigas, *#Col*, o número de formigas por colônia, *#Fmg*, os valores de $\alpha$ e $\beta$ que determinam respectivamente a influência do feromônio e da informação heurística no cálculo da atratividade de cada vértice vizinho factível utilizado para a movimentação das formigas, o coeficiente de

evaporação da matriz de feromônios, $\Phi$, o valor representativo da quantidade de feromônio, $Q$, que é depositado em cada aresta percorrida pela formiga na solução campeã de cada colônia e o número de carros disponíveis *nCar*.

Quadro 3 – Algoritmo Colônia de Formigas F1

```
1. main(nomeInstancia,n,#Col, #Fmg,α,β,Φ,Q,nCar)
2.   LeInstancia(nomeInstancia)
3.   Para i ←1 to nCar
4.     feroRota[i] ← atualizaFeroRota(Concorde(i),10*Q)
5.   Fim para
6.   f(Sol*) ←∞
7.   Para i ← 1 até #Col faça
8.       f(Sol) ← ∞
9.       Para k ← 1 até #Fmg faça
10         lista_cidTroca ←sorteiaTrocas(Sol)
11.        lista_seqCarros ←sorteiaCarros(Sol)
12.        Sol.rota[0] ←0 ; Sol.car[0] ←lista_seqCarros[0]
13.        k ←1
14.        Para j ← 1 até n
15.          Sol.rota[j] ← fazRoleta(Sol.rota[j-1], α,β)
16.          Se (sol.rota[j] ∈ lista_cidTroca)
17.            Sol.car ←lista_seqCarros[k]
18.            k ←k+1
19.          fim se
20.        fim para
21.        Se (f(Sol) < f(Sol*))
22.          Sol* ← Sol; f(Sol*) ← f(Sol)
23.        fim se
24.      fim para
25.      fero ←atualizaFeroRota(Sol*,Φ,Q)
26.  fim para
27. retorna (Sol*)
```

É criada uma matriz para cada carro disponível para armazenar o histórico das arestas percorridas pelas formigas na melhor solução de cada colônia. Estas matrizes de feromônios de rotas são inicializadas, no procedimento *atualizaFeroRota*( ) com as *nCar* soluções ótimas do problema quando solucionado para um único carro e obtidas pelo algoritmo exato apresentado em Applegate *et al.*(2006). Visto que correspondem às soluções ótimas de cada carro, o aquecimento é reforçado com o valor 10**Q*. Ao final de cada iteração do laço que inicia na linha 7, a solução de menor adequação é usada para atualizar as matrizes de feromônio.

Cada colônia possui #*Fmg* formigas que, saindo de seu formigueiro, percorrem um ciclo hamiltoniano e retornam ao ponto de partida, construindo assim uma solução para o problema.Os feromônios reforçam as arestas percorridas nesta solução com o valor de $Q$ e evaporando um percentual do conteúdo das arestas restantes conforme o valor da variável de entrada $\Phi$. As futuras colônias irão se beneficiar deste histórico.

O procedimento *sorteiaTrocas*( ) inicia a construção de uma nova solução sorteando a quantidade de carros que será utilizada no percurso, respeitando o intervalo [2,*nCar*]. Em seguida sorteia com igual probabilidade quais serão os vértices de troca de veículos, guardando esta informação na variável *lista_cidTroca*. O procedimento *sorteiaCarros*( ) sorteia, com igual probabilidade, todos os carros que serão utilizados no percurso nos trechos entre as cidades estabelecidas por *sorteiaTrocas*( ) e guarda esta informação na variável *lista_seqCarros.*

A partir destes procedimentos iniciais a formiga começa a construir sua trilha, tendo

como ponto de partida a cidade 0 e utilizando como transporte o primeiro veículo da *lista_seqCarros*. O caminho é construído de forma iterativa, sempre sorteando a próxima cidade da trilha (*j*) em um procedimento probabilístico de roleta descrito no Quadro 4. Para realizar o cálculo anteriormente citado, leva-se em consideração $\tau_{ij}$ e $\eta_{ij}$. O valor de $\eta_{ij}$ é inversamente proporcional à distância $d_{ij}$ entre os vértices *i* e *j*. No passo 3 do procedimento do Quadro 4 a variável *z* representa o resultado do somatório das atratividades de toda a vizinhança de *i* utilizando o carro *k*. No passo 6 é calculado o percentual correspondente de escolha de cada vértice vizinho a *i* quando utiliza o carro *k*, este valor é representado por $p_{ij}^{k}$. O procedimento enfim sorteia probabilisticamente o próximo vértice em um vetor composto pelos percentuais correspondentes de cada vizinho de *i*.

Quadro 4 – Pseudocódigo do procedimento *fazRoleta*( )

1. *fazRoleta*(*i*,α,*β*,*k*)
2. para *j* até *nCid* faça
3. $z \leftarrow calcula\ (\sum_{l \in \aleph_i^k} (\tau_{il})^{\alpha}\ (\eta_{il})^{\beta})$, se $j \in \aleph_i^k$
4. fim para
5. para *j* até *nCid* faça
6. $p_{ij}^{k} \leftarrow calcula\ (\ (\tau_{ij})^{\alpha}\ (\eta_{ij})^{\beta}/z)$, se $j \in \aleph_i^k$
7. *preparaVetorRoleta*($p_{ij}^{k}$)
8. fim para
9. retorna (*vetorRoleta*[rand()%100])

Após a escolha do próximo vértice, o algoritmo do Quadro 3 verifica se o vértice escolhido pertence à lista de cidades de troca (*lista_cidTroca*). Caso afirmativo, o próximo carro da sequência inicialmente definida é inserido na solução como meio de transporte do próximo trecho do alugador. O procedimento continua de forma iterativa (passos 14 a 24) inserindo cada novo vértice na solução até que não existam mais cidades a serem visitadas, completando-se o ciclo. E ao final de cada colônia de formigas, todas as arestas pertencentes à solução de melhor adequação são utilizadas para marcação e evaporação das matrizes de feromônio que representam o histórico das melhores rotas percorridas. No procedimento *fazRoleta*( ), partindo da cidade *i* com o carro *k*, é realizado um sorteio probabilístico para escolha da próxima cidade a ser visitada. Para cada cidade *j*, sendo *j* pertencente à vizinhança factível $\aleph_{ij}^{k}$ de *i*, verifica-se qual probabilidade de *j* ser escolhida como próxima cidade do percurso.

### 4.2 Algoritmo ACS - F2

O algoritmo F2 é uma variante do algoritmo F1, porém a principal diferença está no procedimento conjugado de escolha dos vértices de troca e devolução dos carros, e de escolha da sequência dos veículos na solução. O passo 11 do pseudocódigo do algoritmo colônia de formigas F1 (Quadro 3) é modificado em F2 para:

11. *lista_seqCarros* ← *escolheTrocasLRC*(*Sol*)

O procedimento *escolheTrocasLRC*( ) inicia sorteando, com igual probabilidade, a quantidade de carros (*qCar*) que será utilizada no percurso e em seguida constrói uma lista restrita de candidatos (LRC) com tamanho δ, que é dado como parâmetro de entrada do algoritmo. Esta LRC contém uma lista de pares (*iCar, dCid*), representando as melhores δ opções de escolha de carro e cidade destino, respectivamente, para as trocas de veículo durante o percurso. Dada uma cidade origem, *iCid*, um custo pode ser designado para cada par (*iCar, dCid*), sendo a taxa de retorno paga quando *iCar* é alugado na cidade *iCid* e devolvido na cidade *dCid*. Um par é escolhido baseado no método da roleta, onde a menor das taxas de retorno possui a mais alta probabilidade associada. A cada par (*iCar, dCid*) escolhido, a LRC é atualizada pela

remoção de todos os elementos cujo primeiro elemento corresponde a *iCar* ou o segundo elemento corresponde a qualquer cidade já designada ao caminho *Sol.rota*[*iCar*], até que a quantidade de carros (*qCar*) seja atingida. Como resultado deste procedimento obtém-se as listas *lista_cidTroca* e *lista_seqCarros* que serão utilizadas por F2 da mesma forma como relatado no pseudocódigo de F1 (Quadro 3).

Outra importante modificação em F2 é a atualização das matrizes de feromônio de rotas durante a execução da colônia e não apenas no final da colônia, como em F1. Ou seja, cada vez que uma solução de melhor adequação é encontrada, imediatamente ela é registrada no feromônio de rotas a fim de que as formigas da mesma colônia também sejam beneficiadas com a nova descoberta. Portanto, entre os passos 22 e 23 do Quadro 3 é inserida a seguinte linha de código: *fero* ← *atualizaFeroRota*(*Sol**,*Φ*,Q). O procedimento de evaporação das matrizes de feromônio de rotas não é realizado durante a execução da colônia e sim ao final, tal como em F1.

### 4.3 Algoritmo ACS – F3

O F3 é bem semelhante ao F1, porém a principal diferença está no modo de escolha dos vértices de troca e devolução dos carros. A forma de escolha da sequência dos veículos na solução continua a mesma de F1.

O procedimento de escolha dos melhores vértices de troca e devolução de carros no algoritmo F3 emprega a mesma idéia tradicional de escolha de vértices dos algoritmos colônia de formigas, o uso de feromônios de marcação das escolhas pregressas. Isto é, além do feromônio de rotas que é usado como recurso facilitador na escolha da sequência de vértices da rota do caixeiro em F1, F2 e F3, também é utilizado em F3 uma matriz de feromônios relativos aos melhores vértices de troca de veículos – o feromônio de trocas. Sendo assim, o passo 10 do pseudocódigo do algoritmo colônia de formigas F1 (Quadro 3) é modificado em F3 para:

10. *lista_cidTroca* ← *roletaFEROTrocas*(*Sol*)

O procedimento *roletaFEROTrocas*( ) do algoritmo F3, inicia sorteando, como em F1, a quantidade de carros (*qCar*) que será utilizada no percurso. As cidades de troca e devolução dos veículos da solução são escolhidas através do método de roleta, conforme demonstrado no Quadro 4, com o diferencial de se usar a matriz de feromônios de trocas de veículos. O aquecimento inicial é feito de forma uniforme com 10**Q*.

O feromônio de rota, tal como em F2, e o feromônio de trocas são atualizados toda vez que uma nova solução de melhor adequação é descoberta. A evaporação só ocorre no final.

### 4.4 Algoritmo ACS – F4

Nesta seção o quarto algoritmo de Colônia de Formigas (F4) proposto para o CaRS é apresentado. O F4 é semelhante aos anteriores. Tomando como base o algoritmo de F1 (Quadro 3), as diferenças em F4 encontram-se no método de escolha dos vértices de troca e de devolução dos carros por LRC, tal como descrito no algoritmo F2 e como novidade, a criação de uma matriz de feromônios para o registro histórico das escolhas na sequência de carros utilizada na solução. Sendo assim, os passos 10 e 11 do pseudocódigo (Quadro 3) do algoritmo F1 são modificado em F4 para:

10. *lista_cidTroca* ← *escolheTrocasLRC*(*Sol*)

11. *lista_seqCarros* ← *roletaFEROCarros*(*Sol*)

O procedimento *roletaFEROCarros*( ) do algoritmo F4, escolhe as cidades de troca e devolução dos veículos da solução através do método de roleta, conforme demonstrado no Quadro 4, com o diferencial de se usar a matriz de feromônios de sequência de veículos. O aquecimento inicial é feito de forma uniforme com 10**Q*. O feromônio de rota e o feromônio de sequência de carros são atualizados toda vez que uma nova solução de melhor adequação é descoberta, tal como em F2. A evaporação só ocorre no final.

### 4.5 Algoritmo ACS – F5

O F5 é semelhante aos anteriores, porém, tomando como base o algoritmo de F1 (Quadro 3), a diferença é que em F5 todas as principais escolhas, ou seja, as escolhas da rota, das cidades de troca e da sequência de veículos na solução, todas elas são auxiliadas por matrizes de feromônios distintas. Esta versão do algoritmo colônia de formigas, portanto diferencia-se por ser multiferomônio.

Para as escolhas de definição de rota, o algoritmo F5 é auxiliado pelas matrizes de feromônio de rotas, tal como em F1. As escolhas referentes aos vértices de troca e devolução de veículos são auxiliadas pelo feromônio de trocas, tal como em F3. E por fim as escolhas referentes à sequência de carros utilizada no ciclo do caixeiro são auxiliadas pelo feromônio de sequência de veículos, tal como em F4. Desta forma, os passos 10 e 11 do pseudocódigo (Quadro 3) do algoritmo F1 são modificado em F5 para:

10. *lista_cidTroca* ← *roletaFEROTrocas*(*Sol*)

11. *lista_seqCarros* ← *roletaFEROCarros*(*Sol*)

Todos os feromônios de F5 são atualizados toda vez que uma nova solução de melhor adequação é descoberta. A evaporação só ocorre no final.

## 5. Experimentos Computacionais

Os testes preliminares para afinação de parâmetros das versões do algoritmo colônia de formigas propostos foram executados sobre um conjunto de 20 instâncias CaRSLIB, com um número de cidades na faixa de 14 a 300 e 2 a 5 veículos.

Os valores definidos para os parâmetros do algoritmo colônia de formigas são: o número de colônias ($\#Col$ = 700), o número de formigas por colônias ($\#Fmg$ = 50), o valor do peso do feromônio ($\alpha$ = 1) e do peso da informação heurística ($\beta$ = 0.5), o valor do coeficiente de evaporação das matrizes de feromônio ($\Phi$ = 0.1), o valor representativo da quantidade de feromônio depositada ($Q$ = 10) e o tamanho da LRC ($\delta$ = 0.25) para os algoritmos F2 e F4.

Foram realizadas 30 execuções independentes de cada algoritmo utilizando-se a plataforma PC *Intel Xeon QuadCore* W3520 2.8 GHz, 8G RAM rodando *Scientific Linux* 5.5 64 bits com *C++*. Os experimentos foram realizados em uma amostra de 40 instâncias, sendo 20 do conjunto Euclidiano e 20 do conjunto não-Euclidiano do CaRSLIB (Goldbarg *et al*., 2012). O número de cidades varia entre 14 e 300 e o número de veículos varia entre 2 e 5. Os algoritmos foram executados para um tempo fixo. O tempo de teste de cada instância foi dado pela média do tempo de processamento obtido em 30 execuções do algoritmo F5. Este último utilizou como regra de parada 700 iterações ou 210 iterações sem melhoria da melhor solução. Os testes estatísticos foram realizados em todas as comparações, utilizando-se os testes Mann-Whitney, também conhecido como *U-test* (Conover, 2001) e o proposto por Taillard *et al.* (2008) para comparar estatisticamente as taxas de sucesso entre dois métodos algoritmo, todos os testes empregando o nível de significância 0,05. A Tabela 1 resume os resultados alcançados. As colunas *Venceu* e *Perdeu* mostram resultados do primeiro algoritmo em relação ao segundo.

Os resultados da Tabela 1 mostram que o algoritmo F5 é superior aos demais, tendo obtido consistentemente um maior número de sucessos. Além disso, o teste de comparação de proporções de Taillard et al. (2008) indica que a proporção de sucessos é estatisticamente melhor, para o nível de significância 0,05, conforme a mostra coluna 1º correspondente ao algoritmo F5. O algoritmo F3 apresenta desempenho superior ao F1, F2 e F4 em número de sucessos. Entretanto, a estatística de comparação de proporções entre F3 e F4 não atinge o nível de significância fixado. Portanto, não é possível afirmar que F3 é superior ao F4. O algoritmo F4 é

superior a F1 e F2. Finalmente, o algoritmo F2 apresenta desempenho significativamente superior que o F1.

**Tabela 1 – Resultados dos testes *U-test* e *Taillard* aplicados às versões ACS**

| Algoritmos | *U-test* | | | *Taillard* | |
|---|---|---|---|---|---|
| | Venceu | Perdeu | Empate | 1º | 2º |
| F5xF1 | 22 | 6 | 12 | 1 | 0 |
| F5xF2 | 21 | 8 | 11 | 1 | 0 |
| F5xF3 | 11 | 6 | 23 | 0,97 | 0,03 |
| F5xF4 | 17 | 5 | 18 | 1 | 0 |
| F4xF1 | 19 | 9 | 12 | 1 | 0 |
| F4xF2 | 14 | 4 | 22 | 1 | 0 |
| F4xF3 | 3 | 10 | 27 | 0,08 | 0,92 |
| F3xF1 | 23 | 6 | 11 | 1 | 0 |
| F3xF2 | 20 | 4 | 16 | 1 | 0 |
| F2xF1 | 18 | 15 | 8 | 0,86 | 0,14 |

Os resultados dos testes estatísticos, portanto, mostram que existem evidências de o algoritmo F5, aparelhado pelo guiamento cooperativo em todas as decisões do problema, logrou obter os melhores resultados qualitativos dentre os demais algoritmos desenvolvidos. Observa-se também que os algoritmos F3 e F4, os quais realizam duas de suas três decisões baseadas no viés produzido pelos feromônios, obtêm melhor desempenho qualitativo que o algoritmo F1, que realiza apenas uma dessas decisões. A reunião desses dois resultados permite sustentar que o guiamento cooperativo proporcionado pela marcação de feromônio interferiu significativamente no resultado estatístico dos algoritmos, criando um viés positivo para a busca algorítmica. Uma alteração no procedimento construtivo proposto em Goldbarg *et al.* (2012) é testada no experimento de F1xF2, visando avaliar a sensibilidade dessa estratégia de tomada de decisão semigulosa utilizada nos algoritmos. O resultado estatístico desse experimento não permite concluir com grau de confiança de 95% que a alteração resulta em um algoritmo de desempenho superior.

## 6. Conclusões

Este trabalho propôs uma abordagem por multiferomônios para algoritmos de Colônia de Formigas. A abordagem proposta foi aplicada a um problema NP-difícil denominado Problema do Caixeiro Alugador. Foi apresentado um estudo experimental de cinco algoritmos Colônia de Formigas, três dos quais utilizaram uma abordagem multiferomônio. Tais algoritmos foram testados contra outros dois que não utilizavam tal estratégia, mostrando resultados estatisticamente superiores que os demais. Os experimentos foram realizados com um grupo de 40 casos testes. Das três abordagens o algoritmo denominado F5, o qual realiza as principais escolhas baseadas em matrizes de feromônios distintas, foi o que apresentou o melhor desempenho em termos de qualidade de solução. Esse resultado mostra que a abordagem é promissora em comparação com a abordagem clássica de algoritmos em Colônia de Formigas.

## Referências Bibliográficas

**Applegate, D. L., Bixby, R.E., Chvátal, V., Cook, W. J**. (2006). The Traveling Salesman Problem: A Computational Study, Princeton University Press.

**Chircop, J. & Buckingham, C. D. A** (2011). Multiple Pheromone Algorithm for Cluster

Analysis, ICSI 2011: International conference on swarm intelligence, id-1-id-10.
**Conover, W.J**. (2001). Practical Nonparametric Statistics. John Wiley & Sons, 3rd Ed.
**Conrad, C. & Perlut, A.** (2006). Enterprise Rent-A-Car Hits New Billion-Dollar Revenue Mark for 3rd Consecutive Year. Enterprise rent-a-car, 2006. Disponível em: http//www.enterpriseholdings.com/NewsReleases/Enterprise_FYO6_Sept06.pdf. Último acesso em Nov/2011.
**Doerner, K., Gutjahr, W.J., Hartl, R.F., Strauss, C. & Stummer, C.** (2008). Nature-inspired metaheuristics for multiobjective activity crashing. Omega 36 (6):1019–1037.
**Feo, T. A. & Resende, M. G. C.** (1995). Greedy randomized adaptive search procedures. Journal of Global Optimization, 6:109-133.
**Garey, M. R. & Johnson, D. S.** (1979). Computers and Intractability. A Guide to the Theory of NP-completeness. W.H. Freeman and Company, San Francisco.
**Goldbarg, M. C., Asconavieta, P. H., & Goldbarg, E. F. G.** (2012). Memetic Algorithm for the Traveling Car Renter Problem: An Experimental Investigation. Memetic Computing, 2012.
**Greenfield, G**. (2006). On Evolving Multi-Pheromone Ant Paintings, 2006 IEEE Congress on Evolutionary Computation, 2006, 2072-2078.
**Gutin, G. & Punnen, A. P.** (2007). The Traveling Salesman Problem and Its Variations. Series: Combinatorial Optimization 12, Springer, ISBN: 978-0-387-44459-8.
**Hsin H-K., Chang, E-J., Chao, C-H., Lin, S-Y. & Wu, A-Y**. (2011). Multi-Pheromone ACO-based Routing in Network-on-Chip System, Inspired by Economic Phenomenon, 2011 IEEE International Conference OC (SOCC), 273-277. 2011.
**Kawamura, H., Yamamoto, M., Suzuki, K. & Ohuchi, A.** (2000). Multiple Ant Colonies Algorithm Based on Colony Level Interactions, IEICE Trans, Fundamentals, E83A(2):371-379.
**Martens, D., Backer, M., Haesen, R., Vanthienen, J. Snoeck, M. & Baesens, B**. (2007). Classification With Ant Colony Optimization, *IEEE Transaction on Evolutionary Computation* 11(5): 651-665.
**Mladenovic, N. & Hansen, P.** (1997). Variable Neighborhood Search. Computers and Operations Research, 24:1097-1100.
**Moncayo-Martínez, L. A. & Zhang, D; Z**. (2010). Multi-objective ant colony optimisation: A meta-heuristic approach to supply chain design.
**Moscato, P.** (1989). On evolution, search, optimization, genetic algorithms and martial arts: Towards Memetic Algorithm. Caltech Concurrent Computation Program. California Institute of Technology, USA.
**Ou Y-p., Wei, Z-wen, & Jilang, H-c.** (2008). An Ant Colony Algorithm Based on Multi-Pheromones, Journal of Guangxi Academy of Sciences 24(3).
**Taillard, E., Waelti, P. & Zuber, J.** (2008). Few statistical tests for proportions comparison. European Journal of Operational Research, 185: 1336-1350.
**Xia, Y-m., Chen, A. J-l., & Meng, X-w.** (2008). On the Dynamic Ant Colony Algorithm Optimization Based on Multipheromones, Seventh IEEE/ACIS International Conference on Computer and Information Science, 630-635.
**Yang, Y., Jin, W., & Hao, X.** (2008). Car rental logistics problem: A review of literature, in: Proceedings of the IEEE International Conference on Service Operations and Logistics, and Informatics, 2: 2815-2819.