# Solução de um problema de roteirização com janelas de tempo através de um algoritmo de múltiplas colônias de formigas

Rafael Lorenzo Santos[1]; José Eugenio Leal[2]

**Resumo:** Sistemas de colônias de formigas (*Ant Colony Systems* - ACS) são modelos matemáticos baseados no comportamento de formigas quando imersas em colônias de indivíduos semelhantes. Formigas são indivíduos simples, porém capazes de interagir entre si, obtendo muitos benefícios desta prática. Estes modelos são muito úteis na resolução de grandes problemas de otimização combinatória, geralmente muito complexos para serem resolvidos por métodos exatos de otimização e representam um incipiente e importante campo de estudos da pesquisa operacional. Neste trabalho serão descritos alguns algoritmos de colônias de formigas, utilizados em problemas de otimização combinatória/discreta. Particularmente, o foco do trabalho será na aplicação destes algoritmos no problema de roteirização de veículos com janelas de tempo. Uma implementação do algoritmo no ambiente *Matlab* foi realizada e testada em problemas padrão usados como *benchmarking* na literatura.

**Abstract:** Ant Colony Systems are mathematical models based on the behavior of ants when immersed in colonies of likely individuals. Ants are simple individuals, however capable of interacting with each other, obtaining benefits from this practice. These models are very useful in solving large combinatory optimization problems, usually too complex to be solved by exact optimization methods, and represent an important and incipient field of study in operations research. This work aims to describe some ant colony algorithms, used in combinatory/discrete optimization problems. Particularly, the focus of this work will be in the application of these algorithms to the vehicle routing problem with time windows. The algorithm has been implemented in *Matlab* environment and tested in standard problems used as benchmarking in the literature.

## 1. INTRODUÇÃO

Sistemas de colônias de formigas (*Ant Colony Systems - ACS)* são modelos matemáticos baseados no comportamento de formigas quando imersas em colônias de indivíduos semelhantes. Estes algoritmos são muito úteis na resolução de grandes problemas de otimização combinatória, geralmente muito complexos para serem resolvidos por técnicas exatas de otimização, e representam um incipiente e importante campo de estudos da engenharia de produção. É um método ainda em desenvolvimento e com grande potencial de aplicação, observando-se que o procedimento faz uso de um grande número de parâmetros e que ainda não foram exaustivamente estudados e difundidos na literatura internacional. Neste trabalho pretende-se apresentar ao leitor, de forma detalhada, os fundamentos do método e a modelagem aplicada ao problema de roteirização com janelas de tempo, usando a abordagem proposta por Gambardella *et al.* (1999) sem, contudo, comparar os resultados com os obtidos por outros métodos já consagrados.

O trabalho descreve alguns algoritmos de colônias de formigas, utilizados em problemas de otimização combinatória/discreta. O foco do trabalho será na aplicação destes algoritmos no problema de roteirização de veículos com janelas de tempo. Este problema se mostra muito comum no mundo empresarial, sendo encontrado com muita freqüência hoje em dia. Muitos métodos de solução foram desenvolvidos ao longo da segunda metade do século XX, mas as colônias de formigas emergem como uma nova classe de algoritmos com grande potencial a ser desenvolvido.

Cada formiga, ao se locomover, deposita uma certa quantidade de feromônios em seu caminho. Os feromônios são substâncias que uma grande parte dos seres vivos é capaz de produzir, e que são detectáveis por outros indivíduos da mesma espécie. Essa espécie de comunicação entre as formigas através de feromônios é conhecida como stigmergia, palavra que vem do grego *stigma* (marca) e *ergon* (trabalho), ou seja, trabalho através de marcas ou sinais. Uma formiga que se lança à procura por alimento tende a seguir caminhos onde essa concentração de feromônios for maior, ou seja, por onde um maior número de formigas tenha passado.

Pode-se ilustrar o procedimento de forma simples através das duas primeiras formigas que partem para a procura de alimento, por dois caminhos distintos. Suponha-se que as formigas deixam a colônia ao mesmo tempo, com a mesma velocidade. A formiga que segue pelo caminho mais curto chega ao alimento após um intervalo $x$ de tempo, enquanto que a formiga que segue pelo caminho mais longo leva $x + \Delta t$ para chegar

[1] **Rafael Lorenzo Santos,** Departamento de Engenharia Industrial. PUC-Rio. Rio de Janeiro, RJ, Brasil (e-mail: rafalorenzo@globo.com).

[2] **José Eugenio Leal,** Departamento de Engenharia Industrial. PUC-Rio. Rio de Janeiro, RJ, Brasil (e-mail: jel@ind.puc-rio.br).

Manuscrito recebido em 1/6/2007 e aprovado para publicação em 29/10/2007. Este artigo é parte de TRANSPORTES, volume XV, número 2, de dezembro de 2007. ISSN: 1415-7713.

ao mesmo lugar. A primeira formiga retorna à colônia após um intervalo de tempo de *2x* (desconsiderando o tempo gasto para coletar o alimento, que deve ser igual para ambas as formigas), enquanto que a segunda leva *2(x + Δt)*. Ou seja, durante o intervalo de tempo [*2x, 2(x + Δt)*], de duração igual a *2Δt,* a situação que encontramos é uma densidade de feromônios duas vezes maior na seção inicial do caminho mais curto que na seção inicial do caminho mais longo, pois a formiga que optou pelo caminho mais curto já percorreu o percurso por inteiro por duas vezes (ida e volta). A segunda formiga provavelmente ainda estará no seu caminho de volta, fazendo com que a seção inicial de seu caminho só tenha sido percorrido uma única vez, durante a viagem de ida ao alimento. Então, para uma formiga que deixe a colônia em busca de alimento durante o dado intervalo de tempo de duração igual a *2Δt*, o caminho mais curto parecerá duas vezes mais atrativo, devido à sua densidade superior de feromônios. Esse processo deixa o caminho mais curto cada vez mais atrativo, conforme o tempo passa.

A partir do comportamento das formigas descrito acima, desenvolveu-se um campo de estudos de algoritmos que se utilizam de alguns desses princípios na resolução de problemas de otimização discreta. Para tornar estes algoritmos ainda mais eficientes, algumas funcionalidades e particularidades podem ser acrescentadas, mas a essência dos sistemas reside no comportamento observado na natureza. O assunto foi primeiramente abordado em 1991, por Dorigo, Maniezzo & Colorni (1991), que propuseram um modelo denominado *Ant System* (AS) para a resolução do problema do caixeiro viajante. Em 1996, Dorigo e Gambardella (1997) criaram o *Ant Colony System* (ACS) a partir do AS, o que representou um novo avanço no desempenho dos algoritmos, e aplicaram este modelo também ao, problema do caixeiro viajante, ou *Traveling Salesman Problem* (TSP), atingindo resultados motivadores. Em 1997, Bullnheimer, Hartl e Strauss (1997) aplicaram pela primeira vez a otimização baseada em colônias de formigas ao problema de roteirização de veículos (VRP), através de um modelo baseado no AS, que ficou denominado por AS-VRP. Muitos outros modelos surgiram ao longo da década de 90, em aplicações tais como o problema de seqüenciamento de tarefas, de coloração de grafos, e mesmo no TSP e no VRP. Entretanto, um dos modelos mais importantes, que é o foco principal deste trabalho, é o *Multiple Ant Colony System* aplicado ao problema de roteirização de veículos com janelas de tempo (MACS-VRPTW), proposto por Gambardella, Taillard e Agazzi (1999), como um desenvolvimento do ACS. Há também registro de aplicações de heurísticas de colônias de formigas combinadas com outras metaheurísticas (Bonasser, 2005). O MACS-VRPTW será estudado em detalhe ao longo deste trabalho. Os modelos AS e ACS também serão apresentados devido à importância que tiveram e ainda têm no desenvolvimento de novos algoritmos baseados em colônias de formigas, e devido ao papel fundamental que representam na compreensão do MACS-VRPTW.

Os algoritmos funcionam na base de iterações, ou passos, nos quais um certo número de formigas tem seu comportamento devidamente determinado pelo modelo e registrado. Estes passos são discretos, e a execução da simulação consiste em uma seqüência de passos, ou iterações, que representam o passar do tempo no mundo real.

Algumas habilidades extras são incluídas no modelo computacional, como, por exemplo, uma memória para cada formiga, que faz com que ela mantenha um registro dos lugares que já visitou. Outra habilidade incluída é o depósito de feromônios em quantidade proporcional à qualidade da solução encontrada por ela. Pode-se utilizar ainda um depósito de feromônios retroativo, ou seja, podemos avaliar o quão boa uma solução encontrada é, para só então realizar os depósitos de feromônios no caminho que a formiga considerada percorreu. Muitas outras modificações podem ser incluídas, que apesar de não corresponderem propriamente a características encontradas na natureza, consistem em melhoramentos visando um funcionamento mais eficiente do algoritmo.

O trabalho está organizado como se segue. Na seção 2, apresenta-se o problema de roteirização de veículos e algumas heurísticas de solução. Na seção 3, apresenta-se o primeiro modelo de colônias de formigas desenvolvido, o *Ant System* (AS). Na seção 4, é detalhado o procedimento *Ant Colony System* (ACS), que representa um aprimoramento do AS. Em seguida, na seção 5, mostra-se o *Multiple Ant Colony System* (MACS-VRPTW), onde são usadas mais de uma colônia de formiga, competindo para resolver o problema da roteirização com janelas de tempo, com diferentes objetivos. Finalmente, na seção 6, é mostrada a aplicação, destacando a capacidade para resolver problemas conhecidos da literatura internacional, usados como *benchmarking* e o trabalho apresenta suas conclusões na seção 7.

## 2. O PROBLEMA DE ROTEIRIZAÇÃO DE VEÍCULOS

Segundo *Lawler et al.* (1985), o problema do caixeiro viajante (PCV, em inglês *Traveling Salesman Problem* – TSP) teve sua origem com Merrill Flood na universidade de Princeton, e sua primeira solução foi proposta no artigo "*Solution of a Large Scale Traveling Salesman Problem*" (Dantzig, Fulkerson & Johnson, 1954), sendo a partir daí tema de inúmeras teses e

artigos pelo mundo. O problema tem sido objeto de aplicação e comparação das mais importantes heurísticas desenvolvidas nas últimas décadas. Várias aplicações da meta-heurística colônia de formigas tem sido desenvolvidas para o TSP. Abrahão (2006) faz uma revisão destas heurísticas e apresenta uma versão própria denominada ACSavings. O problema de roteirização de veículos (*Vehicle Routing Problem* – VRP) pode ser visto como o caso mais geral do problema do caixeiro viajante. O problema basicamente consiste em definir rotas de custo mínimo para veículos, partindo de um depósito, garantindo que todo cliente seja atendido por um e somente um veículo. Estes clientes estão dispostos geograficamente sobre uma dada área ou região, bem como o depósito, de onde todas as rotas se iniciam e terminam.

Pode-se reparar que o problema é muito aplicável ao mundo real, oferecendo um entendimento físico muito simples e direto. Para deixar o problema mais próximo ainda da realidade, algumas restrições podem ser adicionadas. O *Capacitated Vehicle Routing Problem* (CVRP) é uma variante do problema que define uma capacidade máxima de carga para cada carro ou caminhão, e uma demanda de cada cliente a ser atendido. Adicionalmente, pode-se definir ainda uma outra restrição: o tempo total de cada rota sendo que jornadas de trabalho devem ser respeitadas.

Uma outra restrição, que aumenta significativamente a dificuldade de implementação de algoritmos para o problema, é a utilização de janelas de tempo. Neste tipo de problema, cada cliente possui um intervalo de tempo durante o qual deve ser servido pelas rotas. Cada intervalo de tempo consiste em uma hora de início de atendimento e uma hora de fim, e um veículo que alcance um cliente antes da hora de início do atendimento deverá aguardar. Isso deixa o problema mais próximo ainda da realidade, uma vez que estabelecimentos comerciais e/ou industriais, por exemplo, possuem um horário de funcionamento definido, e portanto devem ser atendidos neste período. Outras restrições podem ser adicionadas, como por exemplo uma frota heterogênea de veículos, ou seja, diferentes carros ou caminhões com diferentes capacidades.

O problema de roteirização com janelas de tempo pode ser descrito, tal como o problema do caixeiro viajante, por um grafo *G(N,A)*, onde *N* é um conjunto contendo todos os nós, e *A* é um conjunto contendo todas as arestas. Os nós representam os clientes, e as arestas os caminhos que ligam estes clientes. O depósito de onde partirão as rotas também é considerado um nó – usualmente, o nó 1. Assim como no TSP, também considera-se o grafo como sendo completo, o que não representa de forma alguma uma redução da dificuldade do problema. Cada aresta *ij* possui um custo $C_{ij}$ associado. No caso do TSP, este custo é, usualmente, o comprimento de cada arco. No caso do VRPTW, este custo será o tempo necessário para se percorrer cada aresta como sendo o custo $C_{ij}$.

No depósito, existem *M* veículos à disposição, cada um com uma capacidade *cap*. Cada cliente *i* possui uma demanda $d_i$ associada, bem como um tempo de serviço $s_i$. Além disso, cada cliente possui também sua janela de tempo, durante a qual deverá ser servido. Esta janela é dada por $[b_i, e_i]$, sendo que $b_i$ representa o horário de início do atendimento e $e_i$ a hora limite para que um carro chegue ao cliente para ser atendido. O depósito possui demanda zero e tempo de serviço zero, e sua janela de tempo se estende desde a hora zero até uma hora grande o suficiente para que um veículo retorne de qualquer cliente que esteja atendendo.

O objetivo do problema é encontrar rotas de custo total mínimo que atendam todos os clientes, respeitando as restrições de capacidade dos veículos e das janelas de tempo. Além disso, deseja-se encontrar o número mínimo *MV* de veículos necessários para a realização da tarefa. Na literatura, é comum definir uma hierarquização de objetivos para este problema, na qual a minimização do número de veículos é mais importante que o custo variável das rotas.

Barán e Schaerer (2003) propuseram, uma modificação no algoritmo proposto por Gambardella *et al.* (1999) (conhecido como MACS-VRPTW), visando avaliar várias soluções simultaneamente, e não hierarquicamente, conforme comentado acima. Segundo os autores, deve-se procurar uma solução ótima de *Pareto*, o que é feito mantendo-se um vetor de soluções que otimizam os diversos objetivos do problema. Nesse trabalho, entretanto, o enfoque será o dado ao algoritmo original proposto por Gambardella *et al.* (1999), que representa a base para a maioria das futuras modificações. Antes disso, porém, mostra-se necessária a exposição de alguns métodos mais simples de construção de soluções para o VRP, conforme será visto na seção seguinte.

## 2.1. Heurísticas de solução

Uma das heurísticas mais utilizadas para resolver o problema do VRP é o conhecido algoritmo de Clarke e Wright (1994). Uma outra heurística similar, que é utilizada como solução inicial em diversas meta-heurísticas, é o *Nearest Neighbor*. Este modelo é conhecido pela sua aplicação ao TSP. Para o caso do VRP, a idéia principal é a mesma, porém deve-se atentar para as restrições do problema. Isso faz com que o funcionamento do modelo seja muito próximo àquele do algoritmo de *Clarke e Wright*, com a diferença de que os arcos não são ordenados de acordo com seus ganhos; no *Nearest neighbor*, quem são or-

denados são os nós adjacentes ao nó atual, em ordem decrescente de custos para serem alcançados. Sendo assim, para cada nó visitado, uma nova lista dos nós mais próximos é construída. Por "próximos" entenda-se "com os menores custos para serem alcançados". Será utilizado o termo "distância" entre dois nós para indicar o custo da aresta que os conecta, embora este custo esteja relacionado ao tempo necessário para se percorrer tal arco. Parte-se do depósito que, no início, é o nó atual usado no procedimento. A partir daí, de forma mais detalhada, o algoritmo pode ser descrito por:

1- Construa uma lista com os nós adjacentes ao nó atual, ainda não visitados, em ordem crescente de custo
2- Tome desta lista o nó *i* com o menor custo
   a. Se o nó *i* puder ser incluído na rota atual sem que a solução se torne infactível
      i. Inclua o nó *i* na rota
      ii. Defina o nó *i* como o nó atual e reinicie o algoritmo
   b. Se o nó i não puder ser incluído na rota atual
      i. Elimine o nó *i* da lista de nós adjacentes ao nó atual e retorne ao passo 2
3- Se a lista estiver vazia, tome o depósito (inclua o nó inicial na rota), e reinicie as restrições de capacidade, tempo de ciclo, etc., com o depósito como nó atual
4- Se não houver mais nós a serem visitados, o algoritmo está terminado

Esta é a forma mais simples e intuitiva do *Nearest Neighbor*. Entretanto, para abordar o problema considerando janelas de tempo, é preciso fazer algumas alterações neste algoritmo. Basicamente, o que se deve fazer é utilizar um custo ponderado no lugar do mero tempo de viagem, de forma que ao se selecionar o nó com o menor custo ponderado, esta escolha leve em conta não só o custo de se percorrer o arco, mas também a espera que poderá incorrer caso se chegue a este dado nó antes do início de sua janela de tempo, e a urgência de se atender este nó, devido à proximidade do fim de sua janela de tempo. Esta urgência, pode ser vista como uma folga, mas será mantido o termo que é usado em outras referências importantes da literatura (Balakishnan, 1993 e Solomon, 1987). Conforme sugerido por Pellegrini (2005), deve-se dar pesos a cada uma destas parcelas. A Figura 1 ilustra duas situações distintas: na primeira, há espera, e na segunda não.

Sendo assim, a espera relacionada ao nó *i* pode ser caracterizada por: $(h\_inicial_i - (h\_atual + tempo\_serviço_{nó\ atual} + tvia_{nó\ atual,\ i}))$, onde $h\_inicial_i$ é o horário de início da janela de tempo do nó *i*, *h_atual* é a hora atual (que corresponde ao início do serviço do nó atual), $tempo\_serviço_{nó\ atual}$ é o tempo consumido no nó atual para realizar seu serviço, e $tvia_{nó\ atual,\ i}$ é o tempo do nó atual para o nó *i*. Se esta equação for menor que zero, tem-se uma situação em que não há espera, ou seja, define-se *espera = 0*. Formalmente, tem-se então:

$$espera = \max\left(h\_inicial_i - \left(h\_atual + tempo\_serviço_{nó\_atual} + tvia_{no\_atual,i}\right), 0\right) \quad (1)$$

Define-se urgência como:

$$urgência = \left(h\_final_i - \left(h\_atual + tempo\_serviço_{nó\_atual} + tvia_{nó\_atual,i}\right)\right) \quad (2)$$

sendo ela positiva se for possível chegar até o nó *i* antes que sua janela de tempo se feche. Na Figura 1, ambas as situações mostram uma urgência positiva, pois o final da janela de tempo *i*, *ei*, é posterior a $(h\_atual + tempo\_serviço_{nó\ atual} + tvia_{nó\ atual,\ i})$, ou seja, é possível atender o cliente *i* dentro da janela de tempo.

Dessa forma, no passo 2 do algoritmo de vizinho mais próximo (*Nearest Neighbor*), deve-se procurar pelo nó que minimize um custo ponderado $x = a \cdot tvia + b \cdot espera + c \cdot urgência$, onde *a, b* e *c* são parâmetros, e ainda que apresente *urgência > 0*, que ainda não tenha sido visitado e que não viole quaisquer outras restrições do problema, como capacidade, tempo de ciclo, etc. Esta forma de implementação do algoritmo será o modelo de *Nearest Neighbor* que foi adotada

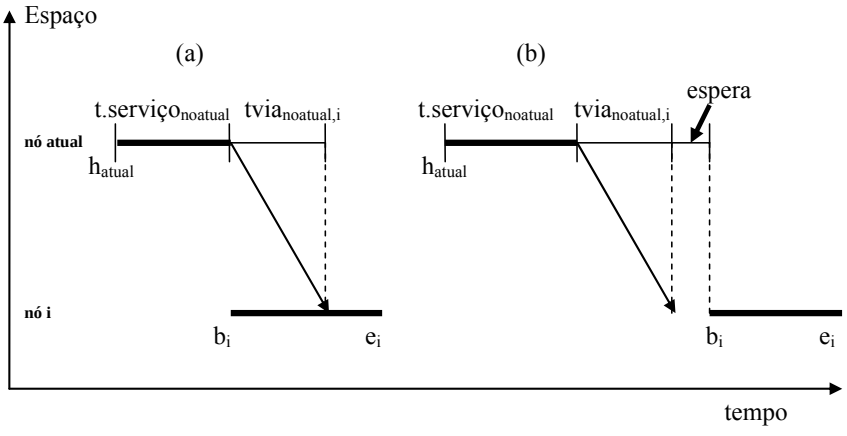


**Figura 1**. Intervalos de tempo no ciclo de serviço, para o VRPTW

neste trabalho, assim como sugerido por Gambardella *et al.* (1999). Outras heurísticas simples, como a heurística de inserção (Ghiani *et al.*, 2004), ou o método de Clarke e Wright modificado, são também conhecidas, mas não foram consideradas, uma vez que o desempenho de heurísticas de inicialização não são o foco deste trabalho.

## 3. O *ANT SYSTEM* (AS)

O AS foi o primeiro algoritmo baseado no comportamento de formigas divulgado, no ano de 1991, e serviu de base para o desenvolvimento de muitos outros modelos, como o ACS e o AS-VRP. Sua aplicação original, que será apresentada aqui, foi ao problema do caixeiro viajante (TSP). Este algoritmo, assim como os outros algoritmos baseados em formigas, simula ciclos de vida de cada formiga atuante no problema.

A regra probabilística de decisão para os algoritmos do tipo AS leva em consideração tanto a quantidade de feromônios acumulada em cada arco, quanto o comprimento (custo) destes arcos. Estas quantidades de feromônios representam o conhecimento acumulado pela colônia e é resultado de decisões tomadas por todas as formigas precedentes; os comprimentos dos arcos, por sua vez, representam as informações físicas do problema, que são estáticas e independem das formigas precedentes. Pode-se dar um peso diferente para a contribuição da densidade de feromônios e do comprimento do arco, através dos parâmetros *alfa* e *beta*, respectivamente. Matematicamente, pode-se descrever estes fatos através da expressão 3, que define como é feito o cálculo da atratividade de cada arco:

$$a_{ij}(t) = \frac{\left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\eta_{ij}\right]^{\beta}}{\sum_{l \in N_i} \left[\tau_{il}(t)\right]^{\alpha} \cdot \left[\eta_{il}\right]^{\beta}} \qquad (3)$$

Este cálculo da atratividade $a_{ij}(t)$ é realizado para uma formiga que se encontra em um nó *i* qualquer, e para todo nó *j* pertencente a $N_i$, onde $N_i$ é o conjunto dos nós adjacentes ao nó *i*. Na equação acima, a variável $\tau_{ij}$ representa a quantidade de feromônio acumulada no arco *(i,j)*, e a variável $\eta_{ij}$ é igual ao inverso do comprimento do arco *(i,j)*.Na expressão, o parâmetro *t*, indica que os valores estão variando a cada iteração *t*. Esta é a representação clássica do cálculo da atratividade. Na implementação, no entanto, dentro de cada iteração, o parâmetro de iterações não é explicitado e, por isso, neste trabalho, se faz referência a estas variáveis sem mencionar o parâmetro *t*.

Uma vez calculados os valores de *a* de todos os arcos incidentes a *i*, pode-se definir a probabilidade de uma formiga *k* escolher percorrer o arco *ij* como:

$$p_{ij}^{k}(t) = \frac{a_{ij}(t)}{\sum_{l \in N_i^k} a_{il}(t)} \qquad (4)$$

Na expressão 4, $N_i^k$ é o subconjunto de $N_i$ que contém somente os nós que ainda não foram visitados pela formiga *k*, ou seja, os nós que não constam na memória da formiga. A figura 2 ilustra o procedimento.

Na figura 2, o conjunto $N_i$ consiste dos nós 1, 2, 3 e 4, enquanto que $N_i^k$ possui apenas os elementos 1, 2 e 3. Cada um destes três nós tem, então, suas probabilidades p1, p2 e p3 determinadas e, com base nestas probabilidades, o nó seguinte será escolhido. A cada nó é associado um arco *(i,j)*. Para selecionar o arco, considerando as probabilidades é sorteado um número aleatório entre 0 e 1. Os arcos, ainda não percorridos na rota atual, são considerados seqüencialmente. A probabilidade de cada arco é calculada com a expressão 4. Depois, para cada arco considerado, a probabilidade acumulada é calculada. O arco que faz com que a probabilidade acumulada seja maior que o número aleatório sorteado é o selecionado.

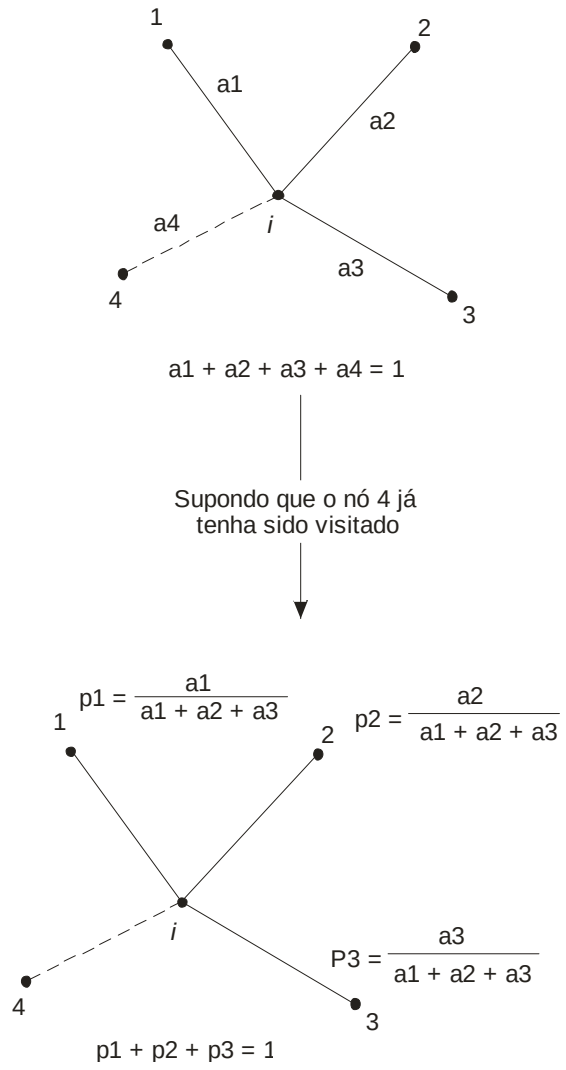


**Figura 2.** Cálculo das probabilidades de escolha de um arco, para o *Ant System*

No AS, uma vez que todas as *k* formigas já tenham percorrido seus caminhos por inteiro, parte-se para a atualização da quantidade de feromônios nos arcos. Dois fenômenos acontecem neste ponto: a evaporação dos feromônios e a deposição de novos feromônios pelas formigas. Ambos os fenômenos podem ser modelados por uma mesma equação:

$$\tau_{ij}(t) \leftarrow (1-\rho)\cdot\tau_{ij}(t) + \Delta\tau_{ij}(t) \qquad (5)$$

Na expressão 5, o parâmetro $\rho$ define o quão rápida será a evaporação dos feromônios, ou seja, ele atua como um coeficiente de decaimento. Já o termo $\Delta\tau_{ij}$, dado pela expressão 6, é o somatório de todos os $\Delta\tau_{ij}^{k}$ de cada arco (*i,j)*, ou seja, é a soma de todas as contribuições individuais de cada formiga *k* que tenha passado pelo arco (*i,j)* na iteração em questão; $\Delta\tau_{ij}^{k}$ é a quantidade de feromônios que a formiga *k* deposita no arco (*i,j),* e é definida como o inverso do comprimento total do percurso que a formiga *k* percorreu, ou seja, quanto menor comprimento $L_k$ do caminho encontrado, maior será a contribuição em feromônios da formiga *k* aos arcos que ela percorreu. Um arco que é percorrido por um número grande de formigas recebe um número maior de contribuições de feromônios, e o tamanho dessas contribuições é proporcional à qualidade das soluções encontradas. Matematicamente, tem-se:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{1}{L_k(t)}, & \text{se}(i,j) \in \text{percurso feito pela formiga } k \text{ na iteração } t \\ 0, & \text{caso contrário} \end{cases} \qquad (6)$$

onde $L_k(t)$ é o tempo total experimentado pela formiga $k$, na iteração $t$.

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t), \text{ onde } m \text{ é o número de formigas}$$

Esta é a idéia básica do AS, aplicado ao problema do caixeiro viajante. A seguir, apresenta-se uma outra meta-heurística que foi desenvolvida tendo como base o AS, conhecida como *Ant Colony System*.

## 4. O *ANT COLONY SYSTEM* (ACS)

O *Ant Colony System* representa um aperfeiçoamento do *Ant System*, e possui um desempenho notadamente superior ao AS e aplicabilidade a problemas de tamanhos bem maiores. Este algoritmo data de 1996, e foi inicialmente desenvolvido por *Dorigo* e *Gambardella* (1997). Assim como o AS, a aplicação original proposta para o ACS é o problema do caixeiro viajante (TSP).

Uma importante diferença entre o ACS e o AS se refere à forma com que é feita a atualização dos feromônios. No caso do AS, ela é feita de forma global e unificada, após o término de cada iteração, sendo realizada simultaneamente a evaporação de feromônios em todos os arcos e a deposição de novos feromônios por todos os caminhos percorridos, proporcionalmente à qualidade dos caminhos encontrados. No ACS, por sua vez, a atualização dos feromônios se dá tanto de forma local após a ação de cada formiga, representando a evaporação dos feromônios pelo seu caminho, quanto de forma global após o fim de cada iteração, recompensando apenas o melhor caminho encontrado pelas formigas com uma certa quantidade de feromônios proporcional à qualidade da solução encontrada. Esta pode ser considerada uma estratégia elitista para a atualização de feromônios. Outra estratégia é a baseada no ordenamento (Rank-Based-AS). Bonasser (2005) apresenta um bom resumo das versões de aplicações de colônias de formigas para o problema do TSP e para o PRV.

A principal diferença entre os algoritmos, entretanto, diz respeito à regra de decisão das formigas. No modelo AS, esta regra é considerada meramente probabilística, de forma que os arcos com maiores concentrações de feromônios sejam mais prováveis de serem escolhidos pelas formigas. No ACS, a regra é um pouco mais complexa, sendo dita pseudo-aleatória, pois envolve uma parte determinística, ou exploratória, e uma parte probabilística, ou desbravadora, parte esta que é definida nos mesmos moldes do AS.

No ACS, tem-se um parâmetro extra chamado *q0*, variando de 0 a 1, e que define quanto poder de exploração de novos caminhos queremos dar às formigas. Antes de cada decisão ser tomada por uma formiga, um número aleatório *q* de 0 a 1 é sorteado. Se este número for maior que *q0*, a regra de decisão da formiga é equivalente àquela do modelo AS, ou seja, a formiga decide para qual nó irá se direcionar com base nas probabilidades calculadas para cada arco factível, usando as expressões (3) e (4) com a diferença que o expoente α da variável $\tau_{ij}$, que representa a importância dada à densidade de feromônios em cada arco, é igual a 1. Esta é a parte probabilística do modelo, e permite que a formiga tenha a possibilidade de experimentar novos caminhos, viabilizando a descoberta de possíveis rotas de comprimentos menores e mais interessantes. Quanto menor for o valor do parâmetro *q0*, maior é esse poder desbravador das formigas.

Por outro lado, se o número aleatório sorteado *q* for menor que *q0*, a formiga tomará sua decisão baseada no conhecimento disponível sob a forma de depósitos de feromônios e distâncias. Em outras palavras, a regra de decisão da formiga será, para todo *j* pertencente a $N_i^k$:

$$p_{ij}^{k}(t) = \begin{cases} 1, \text{ se } \; j = \arg\max a_{ij}, \\ \quad \text{onde } \; a_{ij} = \dfrac{[\tau_{ij}(t)] \cdot [\eta_{ij}]^{\beta}}{\sum_{l \in N_i} [\tau_{il}(t)] \cdot [\eta_{il}]^{\beta}} \\ 0, \text{caso contrário} \end{cases} \quad (7)$$

Ou seja, a formiga escolherá o nó *j* que maximize o valor de $a_{ij.}$ Essa é uma regra determinística, que define que a formiga irá explorar o conhecimento já adquirido ao invés de procurar por novos caminhos melhores. De forma mais concisa, pode-se resumir a regra de decisão do modelo ACS como:

$$j = \begin{cases} \arg\max_{j \in N_i^k} \left\{ [\tau_{ij}(t)][\eta_{ij}]^{\beta} \right\}, \; se \; \; q \le q0 \\ S, \; \text{caso contrário} \end{cases} \quad (8)$$

Na equação (8), S é uma variável aleatória discreta com a probabilidade calculada com a fórmula 4, com o $a_{ij}$ calculado pela fórmula 3, com o parâmetro α igual a 1.

A outra grande diferença para o modelo AS, como dito anteriormente, é na atualização dos feromônios. Primeiramente, o modelo ACS faz uma atualização local em cada arco, ou seja, sempre que uma formiga percorre um arco, a quantidade de feromônios em tal arco é modificada de acordo com a regra:

$$\tau_{ij}(t) \leftarrow (1-\rho) \cdot \tau_{ij}(t) + \rho \cdot \tau_0 \quad (9)$$

Na equação acima, $\tau_0$ é a quantidade inicial de feromônios em cada arco. Esta quantidade inicial de feromônios é definida como o inverso do comprimento da solução encontrada através do método do *Nearest Neighbor*, como foi visto anteriormente ($1/L_{nn}$). Adicionalmente, pode-se multiplicar este comprimento $L_{nn}$ pelo número de nós do problema, diminuindo ainda mais esta concentração inicial de feromônios ($1/n.L_{nn}$). A expressão acima faz com que a densidade de feromônios de um arco sofra uma redução cada vez que tal arco é percorrido por uma formiga. Isso garante que as informações mais antigas vão gradativamente assumindo um peso cada vez menor na decisão atual de cada formiga, e que novas e melhores soluções possam surgir.

No AS, toda formiga era responsável por depósitos de feromônios, que seriam tão maiores quanto melhores fossem os caminhos encontrados por cada formiga. No caso do ACS, esta regra é diferente. O reforço de feromônios neste caso não é feito por cada formiga, mas sim de forma unificada e centralizada. Ao final de cada iteração, o melhor caminho encontrado é o único que tem sua densidade de feromônios incrementada, representando uma diferença importante para o modelo AS. Agora, todos os arcos que compõem o caminho mais curto encontrado têm sua densidade de feromônios atualizada segundo a expressão 5, onde, agora, $\Delta\tau_{ij}(t)$ é igual ao inverso do comprimento do caminho mais curto. Ou seja, quanto melhor for a qualidade da solução, ou quanto menor for o custo desta solução, maior é a contribuição de feromônios que será adicionada aos arcos desta melhor solução.

Apesar de ter sido criado originalmente para a aplicação ao TSP, o ACS foi a base para o desenvolvimento do MACS-VRPTW, algoritmo que será visto na seção seguinte e que é aplicado ao problema de roteirização de veículos com janelas de tempo.

## 5. O *MULTIPLE ANT COLONY SYSTEM* (MACS)

O MACS-VRPTW é a extensão do ACS para o problema de roteirização de veículos com janelas de tempo. O objetivo do problema consiste em minimizar tanto a quantidade de veículos utilizada quanto a distância total percorrida por eles (custo total). Estes objetivos possuem uma hierarquia, sendo o primeiro mais importante que o segundo. Esta hierarquia é comumente empregada na literatura, e será aqui também utilizada. O MACS-VRPTW foi adaptado por Sabino (2004) para a solução de um problema de programação de locomotivas em pátios de manobras.

A idéia central do modelo proposto reside na adoção de duas colônias de formigas, cada uma visando a otimização de um dos objetivos do problema. As melhores soluções são compartilhadas entre as duas colônias, mas as informações de feromônios são independentes. Dessa forma, ambas as colônias trabalham de forma distinta, porém simultânea, na procura por soluções mais interessantes, tendo os resultados de uma colônia grande influência nos caminhos tomados pela outra. Esta é a essência do MACS-VRPTW.

Idealmente o algoritmo deveria usar computação paralela, ou seja, ambas as colônias estariam ativas e procurando por melhores soluções ao mesmo tempo. Esta é a idéia proposta por Gambardella *et al.* (1999). Na prática, porém, visando facilitar a implementação do algoritmo, pode-se construir um algoritmo que trabalhe com ciclos, nos quais as colônias estão ativas alternadamente, e a solução encontrada por uma é utilizada em seguida pela outra, e vice-versa. Esta será a forma de implementação utilizada nas simulações.

Cada uma das duas colônias será então simulada separadamente. Basicamente, cada colônia será simulada por uma função, que deverá construir caminhos e analisar a qualidade destes caminhos. A diferença nos objetivos das duas colônias se refletirá principalmente na forma com que esta análise dos caminhos será feita. Entretanto, a construção dos caminhos em ambas as colônias será feito de forma muito parecida, pois a regra de decisão de cada formiga é sempre a mesma, independentemente de a qual colônia ela pertence.

Sendo assim, ambas as colônias usarão uma mesma função auxiliar para a construção dos caminhos percorridos. Este processamento dos caminhos percorridos por cada formiga por esta função auxiliar será feito de forma equivalente ao funcionamento do ACS para o TSP, segundo o qual uma matriz de distâncias/custos e uma de feromônios são analisadas na procura pelo arco mais atrativo. Entretanto, como o VRP envolve rotas múltiplas, e não mais um único caminho passando por todos os nós, é preciso que o processamento das matrizes seja adaptado. Isto é feito através da inclusão de depósitos virtuais nestas matrizes. Estes depósitos são verdadeiras cópias do depósito real, nas mesmas coordenadas cartesianas, e as matrizes deverão possuir tantos depósitos quantos veículos forem utilizados. Por exemplo, um problema de 10 nós, que será resolvido com 3 veículos, precisará de matrizes não mais de 10x10, mas de 12x12, onde os nós 11 e 12 serão cópias do depósito (nó 1). Dessa forma, o funcionamento do ACS no VRP pode ser equivalente àquele no TSP, resultando em um caminho que incluirá, em sua seqüência proposta de nós a serem percorridos, cópias virtuais do depósito, que representarão retornos ao depósito. Quando um veículo retorna ao depósito, a capacidade e o tempo de viagem são zerados, possibilitando que outra rota seja iniciada, a partir da hora atual. O número de veículos utilizados é igual ao número de visitas ao nó 1, menos um.

O modelo precisa ser inicializado com alguma solução inicial. Ter uma solução inicial evita um procedimento inicial de busca de uma solução pelo MACS-VRPTW, sem referência de depósito de feromônios, o que obrigaria a um certo tempo relativamente grande de cálculo. Isto normalmente é feito com um método simples, como o *Nearest Neighbor*. Conforme foi visto anteriormente, este método possui algumas peculiaridades para que possa ser aplicado ao VRPTW, mas mantém sua simplicidade de entendimento e facilidade de implementação. Tendo em mãos esta solução inicial, o MACS-VRPTW entra na sua fase cíclica, na qual as duas colônias de formigas entrarão em funcionamento. A solução inicial, primeiramente, passa pelo chamado ACS_VEI, que é um algoritmo que visa encontrar alguma solução factível que utilize um veículo a menos que na solução inicial. Este algoritmo usa uma primeira colônia de formigas para tanto. Feito isso, a solução encontrada passa para a segunda colônia de formigas, materializada sob a forma do algoritmo ACS_TIME, que tem como objetivo minimizar o tempo total gasto pelo dado número de veículos encontrado pela primeira colônia. Terminada a execução deste segundo algoritmo, o ciclo é reiniciado, com a primeira colônia procurando uma nova solução com um veículo a menos que a melhor solução encontrada pela segunda colônia. Esse ciclo continua até que algum critério de término seja atendido. Por exemplo, este critério pode ser o número de iterações, ou o tempo de processamento. No algoritmo, apresentado a seguir, será usado um critério variável, no qual o algoritmo só se dá por terminado após um certo número de ciclos sem que haja nenhuma melhora na solução.

A idéia básica do MACS-VRPTW é a descrita acima. Sua implementação, porém, possui muitas peculiaridades que precisarão ser vistas com mais detalhes, o que será feito na próxima seção, na qual estudar-se-á a fundo uma proposta de implementação do algoritmo.

### 5.1. Geometria do problema

Será visto, agora, o funcionamento do algoritmo aqui proposto para o problema de roteirização de veículos com janelas de tempo. A descrição do problema foi feita na seção 2. Todas as simulações foram elaboradas com o auxílio do software *Matlab 6.0 v12*. Primeiramente, é preciso definir de que forma serão os dados de entrada do algoritmo MACS-VRPTW, e qual será a forma de sua saída.

Como dados de entrada, é necessário ter as características físicas do problema, bem como os dados das janelas de tempo e capacidade de carga dos veículos. Sendo assim, serão usados como dados de entrada: uma matriz $N$ x $N$ contendo todos os custos $C_{ij}$ entre todos os pares de nós do problema (matriz *dist)*; dois vetores 1 x $N$ contendo os valores iniciais $b_i$ e finais $e_i$ das janelas de tempo de cada um dos $N$ nós (vetores *h_inicial* e *h_final*); um vetor 1 x $N$ com as demandas $d_i$ de cada nó, lembrando que a demanda do nó 1, que convencionou-se ser o depósito, deve ser igual a zero (vetor *demanda*); uma variável com o valor da capacidade máxima de carga dos veículos, aqui considerados todos iguais (*cap*); finalmente, um vetor 1 x $N$ contendo os tempos de serviço $s_i$ de cada nó, inclusive do depósito (vetor *tempo_serv*). Estes dados são suficientes para identificar completamente o problema. A saída do algoritmo, por sua vez, será composta de três partes: o vetor *caminho*, contendo a seqüência de nós do melhor caminho encontrado; a variável *comp*, com o custo total de tal caminho (comprimento total do caminho, que conforme mencionado acima, é equivalente ao tempo total que um veículo precisará rodar para percorrer o caminho encontrado) e a variável *carros*, com o número de veículos utilizados na solução encontrada.

O algoritmo criado para a rotina MACS-VRPTW é mostrado a seguir. Primeiramente, com base nos seis dados físicos do problema, é chamada a função auxiliar *nearest_neighbor*, que determina um caminho factível através da heurística homônima, aplicada ao VRP (como descrita na seção 3.2.1). Sendo assim, as variáveis de saída *caminho, comp* e *carros* são inicia-

lizadas com os valores referentes ao caminho encontrado pelo *Nearest Neighbor*.

Algoritmo MACS-VRPT

1. Encontre uma solução inicial, aplicando o *Nearest Neighbor*
2. Enquanto não se cumpre a condição de parada faça:
    a. Chame o ACS-veículo
    b. Chame o ACS-tempo

Fim enquanto

Na seção seguinte apresenta-se uma aplicação do método.

## 6. APLICAÇÃO

O objetivo desta seção é mostrar o desempenho do MACS_VRPTW, no tocante à qualidade da solução, mas sobretudo da sua capacidade de captar as influências dos fatores considerados essenciais para o problema: O horizonte de tempo de planejamento; a disposição geográfica de nós e as características das janelas de tempo. Também foi analisada brevemente, a influência do parâmetro *q0* na qualidade da solução e no tempo de processamento. Não é objetivo do trabalho analisar o desempenho computacional deste procedimento com a ferramenta computacional escolhida: o *Matlab*.

Um conjunto de problemas obtidos em www.top.sintef.no (2006) foi usado para testar o modelo. Estes problemas são instâncias desenvolvidas por Solomon. Como não se pretende avaliar o desempenho computacional e dada a limitação do software utilizado, são tomados apenas 25 nós, ou 50 nós, dos 100 nós propostos nas instâncias de Solomon. Os problemas são classificados de acordo com os alguns fatores, mencionados no início da seção. Os valores que caracterizam e combinam os fatores são:

- características da distribuição geográfica: R, representa uma distribuição aleatória (random), C, uma distribuição agrupada (cluster) e RC uma combinação das duas;
- horizonte de tempo: horizonte estreito, 1 e horizonte amplo, 2;
- características das janelas de tempo: janela estreita, 01 e janela mais larga, 02.

Dessa forma, o problema R101, por exemplo, será um problema com distribuição aleatória de pontos, com horizonte de tempo estreito, e com as janelas de tempo segundo a configuração 01. A Tabela 1 resume os tipos de problema resolvidos e comparados. São seis tipos diferentes de problemas, sendo que todos tem 25 nós, com exceção do R103 que tem 50 nós. Os dados podem ser consultados em Santos (2006).

Para aplicação dos algoritmos do MACS-VRPTW aos problemas, foram definidos os parâmetros *rho* = 0,1, *beta* = 1, *q0* = 0,9, e os parâmetros *a, b* e *c* do *Nearest Neighbor* iguais a 0,6, 0,2 e 0,2 respectivamente. Além disso, cada simulação das funções *acs_time* e *acs_vei* foi feita com até 20 iterações com $k$ = 10 formigas cada uma. Apresenta-se a seguir, a título de ilustração, o resultado da aplicação ao problema R201. A capacidade do veículo é de 200 unidades.

**Tabela 1.** Resumo dos problemas estudados

| *Fator a ser analisado* | *Problemas a serem analisados* |
|---|---|
| Horizonte de tempo | R101 e R201 |
| Disposição geográfica dos nós | R101, C101 e RC101 |
| Arranjo das janelas de tempo | R101 e R111 |
| Influência do tamanho do problema | R103 |

Aplicando os algoritmos ao problema R201, tem-se como melhor solução encontrada o caminho 1-6-15-17-7-23-5-25-2-14-18-1-3-16-24-22-13-12-20-8-19-9-10-4-21-11-1, de custo 538,1 e utilizando dois veículos. O custo médio das três simulações realizadas foi de 550,88. A restrição de capacidade poderia influir neste resultado, mas como pode ser visto na Tabela 2, o limite de 200 unidades por veículo não foi atingido em nenhuma das rotas, ficando evidenciada a influência de diferentes horizontes de simulação na solução. Informações sobre o tempo e a carga de cada uma das duas rotas estão na Tabela 2.

**Tabela 2.** Características das rotas encontradas para o problema R201

| | *Tempo Total* | *Carga* |
|---|---|---|
| Rota 1 | 856,1 | 143 |
| Rota 2 | 642,3 | 183 |

Aplicando apenas o *Nearest Neighbor*, chega-se a um custo total de 682,04, com 2 veículos. Este valor é 26,82% maior que a melhor solução encontrada pelo MACS-VRPTW, e 23,81% maior que a média do MACS-VRPTW. A melhor solução conhecida (www.top.sintef.no, 2006) é 463,3, porém com um número de veículos igual a 4, o que inviabiliza comparações quanto à qualidade, uma vez que a hierarquia dos objetivos utilizada para se chegar a esta solução foi diferente daquela utilizada neste trabalho– uma solução com um número inferior de veículos é melhor que uma solução com um custo menor. Na figura 4 ilustra-se a melhor solução encontrada pelo MACS-VRPTW. O nó circulado corresponde ao depósito (nó 1).

### 6.1. Efeito do horizonte de tempo

Os problemas R101 e R201 diferem quanto ao horizonte de tempo menor (1) e maior (2). Para o problema R101, a melhor solução encontrada pelo MACS-VRPTW corresponde ao seguinte conjunto de nós: 1-

15-16-23-5-1-6-17-7-14-1-3-22-4-25-1-13-10-21-2-1-12-20-11-1-24-1-8-9-18-1-19-1, com um custo total de 616,38 e um número de veículos utilizados igual a 8. Para o problema R201, tem-se, como melhor solução encontrada, o caminho 1-6-15-17-7-23-5-25-2-14-18-1-3-16-24-22-13-12-20-8-19-9-10-4-21-11-1, de custo 538,1 e dois veículos. O custo médio das três simulações realizadas foi de 550,88. Pode-se ver que o horizonte maior de tempo possibilitou um número muito maior de clientes em cada rota. Fica evidente o fato de que com um horizonte mais amplo, as rotas podem ser mais longas, envolvendo um número maior de clientes e, portanto com menos veículos. A restrição de capacidade poderia influir neste resultado, mas o limite de 200 unidades por veículo não foi atingido em nenhuma das rotas, ficando evidenciada a influência de diferentes horizontes de simulação na solução.

### 6.2. Efeito da disposição geográfica dos nós

O problema C101, com nós agrupados em 3 clusters, conduziu a uma solução com três rotas, com custo médio de 191,42, por rota, tal como encontrado pela melhor solução, que também chegou a três rotas, com custo médio de 191,3.

O problema RC101 combina separação dos clientes em três grupos, mas com uma disposição aleatória entre eles. O MACS-VRPTW chegou a uma solução com quatro rotas com um custo médio de 459,99 por rota. A melhor solução encontrada também apresentava 4 rotas com um custo médio de 461,1, bem próxima a encontrada pelo MACS-VRPTW.

Como foi mostrado a instância R101, sem agrupação de nós leva a necessidade de uso de 8 veículos e rotas, com comprimento de 616,38. Portanto o MACS-VRPTW comporta-se de forma coerente no tratamento da disposição geográfica dos nós.

### 6.3. Arranjo das janelas de tempo

O problema R111 apresenta um padrão de janelas bem mais abertas que o R101, aonde todos os nós tinham janelas de 10 unidades de tempo. Para este problema foi encontrada a solução com 4 rotas e custo de 446,34. A melhor solução encontrada tem um custo de 428,5, mas usa 5 veículos. Como esta solução buscava apenas a minimização do tempo total, não se pode concluir sobre a vantagem de uma solução sobre a outra. Fica em aberto se a diferença de cerca de 8% na solução é compensada pelo uso de um veículo a menos, encontrado pelo MACS-VRPTW.

Lembra-se que no problema R101, foram usados 8 veículos e o custo total foi de 616,38 unidades.

Portanto o MACS-VRPTW foi capaz de representar coerentemente a influência do arranjo das janelas de tempo na solução do problema.

### 6.4. Influência do tamanho do problema

Para o problema R103, com 50 nós, o MACS-VRPTW, chegou a uma solução de custo 896,34, porém com apenas 8 veículos. A melhor solução para o problema indicada no website possui custo de 772,9, mas com 9 veículos. Uma comparação com a melhor solução conhecida, novamente, é inviabilizada devido a diferenças nos objetivos. O processamento do algoritmo (executado três vezes) levou de 20 a 22 minutos, o que é aproximadamente dez vezes o tempo médio que os problemas de 25 nós levaram.

A aplicação dos algoritmos a problemas maiores que os apresentados (100 nós) não se mostra muito eficiente, levando mais de uma hora de execução para a obtenção de soluções de qualidade mediana. Entretanto, isso provavelmente acontece porque os *scripts* criados nesta pesquisa não visaram uma otimização de performance, mas tão somente um funcionamento correto e simples, seguindo a idéia central do modelo proposto. Estes *scripts* podem certamente passar por um processo de otimização, a começar pela utilização de alguma linguagem de programação de nível mais baixo que o *Matlab*, como C++ ou outra, que apresentará um desempenho e uma velocidade muito superior.

### 6.5. Análise de influência de variação de parâmetros do MACS-VRPTW

Além das análises realizadas, também se mostra relevante uma análise de sensibilidade do parâmetro *q0*. Este parâmetro foi, para todas as simulações realizadas até agora, fixado em 0,9. Porém, é importante avaliar o impacto que uma variação deste valor causa na qualidade das soluções e nos tempos de processamentos. Para isso, foi simulado o problema R101 com $q0 = 0,7$ e com $q0 = 0,5$. Os resultados que emergem de uma simulação com $q0 = 0,9$ já são conhecidos.

No primeiro caso, as três simulações com $q0 = 0,7$ levaram a um custo médio das soluções de 616,13, o que é praticamente igual ao que se tinha quando $q0 = 0,9$. As simulações, entretanto, levaram em média cerca de 3 minutos, um pouco mais longas que anteriormente. No segundo caso, com $q0=0,5$, o custo médio encontrado foi de 621,40, e o tempo médio das simulações ficou em torno de 4 minutos. Ou seja, isto sugere que o valor ideal de *q0* deve ser maior que 0,5 para que o desempenho do algoritmo como um todo se mostre mais eficiente. Em outras palavras, se as formigas forem desbravadoras em excesso (*q0* baixo), elas deixam de tirar o máximo do proveito do conhecimento acumulado, reduzindo a eficiência da busca como um todo.

Análises deste tipo poderiam ser estendidas para outros tipos de problemas, para que fosse possível avaliar se esta influência de *q0* se dá de forma equivalente; aqui, entretanto, o estudo de sensibilidade restringe-se

ao problema R101, ficando a sugestão para futuros desenvolvimentos.

### 6.6. Resumo das aplicações

Na Tabela 3 resumem-se todos os resultados obtidos através da simulação dos seis problemas sendo cinco com 25 pontos e um com 50 pontos. O custo indicado na coluna referente ao MACS-VRPTW corresponde à média dos custos das soluções encontradas em cada uma das três simulações feitas para cada problema.

## 7. CONCLUSÕES

O objetivo deste trabalho foi apresentar alguns algoritmos baseados em colônias de formigas e ilustrar sua utilidade prática na resolução de um dos problemas mais intrigantes da pesquisa operacional: o problema de roteirização de veículos com janelas de tempo (VRPTW). Um algoritmo de solução foi proposto e seu funcionamento foi minuciosamente estudado, como uma forma de se compreender mais profundamente o próprio funcionamento da meta-heurística. Através de aplicações dos algoritmos a problemas propostos e de análise de seus resultados foi possível concluir que os modelos são realmente capazes de chegar a uma solução próxima da ótima, ou até mesmo a ela própria. A aplicação a instâncias clássicas da literatura internacional mostra que o algoritmo, além de mostrar uma boa qualidade de solução para problemas de 25 nós, foi capaz de captar corretamente a influência das características dos problemas sobre a solução de cada caso. Um objetivo secundário do trabalho foi o de implementar o modelo em ambiente *Matlab*, o que abre interessantes possibilidades de uso da modelagem para fins didáticos, observando que um ambiente similar ao *Matlab*, o *Scilab*, é livre e pode ser obtido na internet. Detalhes da programação do algorítmo em *Matlab* podem ser obtidos em Santos (2006). Para instâncias de 50 nós não foi feita uma análise completa, o que não permite conclusões definitivas e, para instância de 100 nós, o tempo de computação e a qualidade da solução não foram satisfatórios. Cabe observar dois aspectos importantes. Em primeiro lugar o *Matlab*, ou o *Scilab* não podem ser considerados linguagens eficientes de programação. Em segundo lugar, a meta-heurística colônia de formigas é relativamente recente e está em processo de disseminação e de experimentação no meio científico, e portanto não deve ser comparada de forma definitiva a outras abordagens como a meta-heurística tabu-search. No entanto, experiências como as descritas neste trabalho, mostram um grande potencial de desenvolvimento e aplicação.

### REFERÊNCIAS BIBLIOGRÁFICAS


Abrahão, F.T.M. (2006). *A meta-heurística colônia de formigas para solução do problema de programação de manutenção preventiva de uma frota de veículos com múltiplas restrições: aplicação na Força Aérea Brasileira*. Tese de Doutorado USP.

Balakrishnan, N. (1993). Simple Heuristics for the Vehicle Routeing Problem with Soft Time Windows. *The Journal of the Operational Research Society* 44,3, 179-287.

Barán, B.; Schaerer, M. (2003) A multiobjective ant colony system for vehicle routing problem with time windows. In: *Proceedings of the 21st IASTED International Conference of Aplied Informatics*, Innsbruck, Austria, pp 10-13.

Bonasser, U. O. (2005). *Meta-Heuristicas híbridas para solução do problema de roteirização de veículos com múltiplos depósitos e frota heterogênea fixa: aplicação na Força Aérea Brasileira*. Tese de Doutorado USP.

Bräysy O.; Gendreau, M. (2001) *Metaheuristics for the vehicle routing problem with time windows*. Oslo: SINTEF Applied Mathematics, Research Council of Norway, Noruega, Relatório Técnico.

Bullnheimer, B.; Hartl, R. F.; Strauss, C. (1999) An improved Ant System algorithm for the Vehicle Routing Problem. *Annals of Operations Research*, Vol 89, No. 0. pp. 319-328.

Bullnheimer, B.; Hartl, R. F.; Strauss, C. (1999) Applying the ant system to the vehicle routing problem. In: *2nd International Conference on Metaheuristics MIC-97*, Sophia-Antipolis, França, Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization, pp. 285-296.

Clarke, G.; Wright, J. (1964) Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, Vol. 12, No.4. pp.568-581.

Dorigo, M.; Di Caro, G.; Gambardella, L. M. (1999) Ant algorithms for discrete optimization. *Artificial Life*, Vol. 5, No. 3, pp. 137-172.

Dorigo, M.; Gambardella, L. M. (1997) Ant colonies for the travelling salesman problem. *Biosystems*, Vol. 43, No. 2. pp. 73-81.

Dorigo, M.; Maniezzo, V.; Colorni (1991) A. *Positive feedback as a search strategy*. Milão: Dipartimento di Elettronica e Informatica, Politecnico di Milano, Itália, Relatório Técnico

Gambardella, L. M. *et al.* (2003) Ant Colony Optimization for vehicle routing in advanced logistics systems. In: *Proceedings of MAS 2003 – International Workshop on Modelling and Applied Simulation*, Bergeggi, Itália.

Gambardella, L. M.; Taillard, É.; Agazzi, G. (1999) MACS-VRPTW: A Multiple Ant Colony System for vehicle routing problems with time windows. In: *New Ideas in Optimization. Londres: McGraw-Hill*, pp. 63-76.

Ghiani, G, Laporte, G, Musmanno, R (2004) *Introduction to Logistics Systems Planning and Control*. West Sussex, England: John Wiley & Sons, 352 p.


**Tabela 3.** Resumo dos resultados obtidos nos problemas abordados

| | | *MACS-VRPTW* | | *Nearest Neighbor* | | *Melhor solução* | |
|---|---|---|---|---|---|---|---|
| | ***Problema*** | ***Custo*** | ***Veículos*** | ***Custo*** | ***Veículos*** | ***Custo*** | ***Veículos*** |
| | R101 | 616,38 | 8 | 648,54 | 8 | 617,1 | 8 |
| | R201 | 550,88 | 2 | 682,04 | 2 | 463,3 | 4 |
| 25 nós | C101 | 191,42 | 3 | 236,94 | 3 | 191,3 | 3 |
| | RC101 | 461,85 | 4 | 515,14 | 5 | 461,1 | 4 |
| | R111 | 475,30 | 4 | 536,36 | 5 | 428,5 | 5 |
| 50 nós | R103 | 896,34 | 8 | 1009,9 | 9 | 772,9 | 9 |

http://www.top.sintef.no/vrp/benchmarks.html. *Benchmarks-vehicle routing and travelling salesperson problems*. Acesso em 22 de novembro de 2006.

Lawler, E. L. *et al*.. (1985) *The traveling salesman problem: a guided tour of combinatorial optimization*. 1. ed. New York: John Wiley & Sons, 476p.

Neves, T. A.; Souza, M. J. F.; Martins, A. X. (2004) *Resolução do problema de roteamento de veículos com frota heterogênea e janelas de tempo*. Ouro Preto: Departamento de Computação, UFOP, Relatório Técnico.

Pellegrini, P. (2005) *Application of two nearest neighbor approaches to a rich vehicle routing problem*. Bruxelas: IRIDIA, Université Libre de Bruxelles. Relatório Técnico.

Sabino, J.A. (2004). *Competição entre colônias de formigas aplicada à designação de locomotivas de manobras em pátios ferroviários*. Dissertação de Mestrado. UFES.

Santos, R. L. (2006) *Uma aplicação de algoritmos de colônias de formigas em problemas de roteirização de veículos com janelas de tempo.* Rio de Janeiro, 86p. Dissertação de Mestrado – Departamento de Engenharia Industrial, PUC-Rio.

Solomon, M. M. (1987) Algorithms for the vehicle routing and scheduling problems with time windows constraints. *Operations Research*, 35, 2, 254-2.