Ministério da Ciência e Tecnologia

BRASIL UM PAÍS DE TODOS GOVERNO FEDERAL

INPE-16609-RPQ/833

UM MÉTODO DE SOLUÇÃO BASEADO NA METAHEURÍSTICA COLÔNIA DE FORMIGAS E EM UMA TÉCNICA DE MINERAÇÃO DE DADOS PARA A RESOLUÇÃO DO PROBLEMA DO CAIXEIRO VIAJANTE

Rodolfo Ranck Júnior

Relatório final da disciplina Princípios e Aplicações de Mineração de Dados (CAP-359) do Programa de Pós-Graduação em Computação Aplicada, ministrada pelo professor Rafael Santos.

Registro do documento original:
<http://urlib.net/sid.inpe.br/mtc-m18@80/2009/10.21.23.43>

INPE
São José dos Campos
2009

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GB)
Serviço de Informação e Documentação (SID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3945-6911/6923
Fax: (012) 3945-6919
E-mail: pubtc@sid.inpe.br

CONSELHO DE EDITORAÇÃO:
Presidente:
Dr. Gerald Jean Francis Banon - Coordenação Observação da Terra (OBT)
Membros:
Drª Maria do Carmo de Andrade Nono - Conselho de Pós-Graduação
Dr. Haroldo Fraga de Campos Velho - Centro de Tecnologias Especiais (CTE)
Drª Inez Staciarini Batista - Coordenação Ciências Espaciais e Atmosféricas (CEA)
Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)
Dr. Ralf Gielow - Centro de Previsão de Tempo e Estudos Climáticos (CPT)
Dr. Wilson Yamaguti - Coordenação Engenharia e Tecnologia Espacial (ETE)
BIBLIOTECA DIGITAL:
Dr. Gerald Jean Francis Banon - Coordenação de Observação da Terra (OBT)
Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)
Jefferson Andrade Ancelmo - Serviço de Informação e Documentação (SID)
Simone A. Del-Ducca Barbedo - Serviço de Informação e Documentação (SID)
REVISÃO E NORMALIZAÇÃO DOCUMENTÁRIA:
Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)
Marilúcia Santos Melo Cid - Serviço de Informação e Documentação (SID)
Yolanda Ribeiro da Silva Souza - Serviço de Informação e Documentação (SID)
EDITORAÇÃO ELETRÔNICA:
Viveca Sant´Ana Lemos - Serviço de Informação e Documentação (SID)

RESUMO

Neste trabalho, propõe-se resolver o problema do caixeiro viajante através de um método de solução baseado na metaheurística Colônia de Formigas e em uma técnica de Mineração de Dados. Formula-se essa metaheurística e um procedimento para buscar associações que caracterizem a qualidade das soluções geradas por essa metaheurística. Iterativamente, a informação obtida por este procedimento é passada para a metaheurística, que obtém novas soluções baseadas nessa análise. Testes computacionais apresentados mostram que o procedimento proposto pode acelerar a convergência para uma boa solução e/ou melhorar as soluções obtidas por uma metaheurística colônia de formigas.

A SOLUTION METHOD BASED IN THE ANT COLONY METAHEURISTIC AND IN A DATA MINING TECHNIQUE TO SOLVE THE TRAVELLING SALESMAN PROBLEM

ABSTRACT

In this work, we propose to solve the Travelling Salesman Problem using a solution method based in the Ant Colony Metaheuristic and in a Data Mining technique. The metaheuristic is formulated and also a procedure to search associations that caracterize the quality of the solution generated by this metaheuristic. Step by step the information collected by this procedure is sent to the metaheuristic, that generate new solutions based on this analisys. The presented Computacional tests shows that the proposed procedure can accelerate the convergence to a good solution and/or better the solutions obtained by de ant colony metaheurist.

SUMÁRIO

Pág.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

LISTA DE SÍMBOLOS

1 INTRODUÇÃO 17

1.1 A metaheurística Colônia de Formigas 17

1.2 O Problema do Caixeiro Viajante 19

1.3 Mineração de Dados 20

2 O PROCEDIMENTO PROPOSTO 21

2.1 Modelagem do ACO para resolver o TCP..... 21

2.2 O procedimento proposto para melhorar o desempenho do ACO 26

3 RESULTADOS COMPUTACIONAIS..... 33

4 CONCLUSÃO 41

5 REFERÊNCIAS BIBLIOGRÁFICAS..... 43

LISTA DE FIGURAS

Pág.

1.1 - Formigas precisam decidir-se por um caminho até chegar ao alimento. Quanto maior a concentração de feromônio em um caminho, maior probabilidade dele ser escolhido 18
1.2 - Dentre os caminhos possíveis, com o passar do tempo, formigas tendem a trilhar o menor caminho (em destaque na cor mais escura)....19

LISTA DE TABELAS

Pág.

3.1 – Instâncias do TSP utilizadas neste trabalho 33
3.2 – Parâmetros e valores para o ACO e ACO-DM 34
3.3 – Parâmetros e valores exclusivos para o ACO-DM..... 34
3.4 – Resultados para a instância u16 35
3.5 – Resultados para a instância u22 36
3.6 – Resultados para a instância u48 37
3.7 – Resultados para a instância u76 38
3.8 – Resultados para a instância ch150..... 39

LISTA DE SIGLAS E ABREVIATURAS

ACO	*Ant Colony Optimization*
DM	*Data Mining*
ACO-DM	*Ant Colony Optimization with Data Mining*
TSP	*Travelling Salesman Problem*

LISTA DE SÍMBOLOS

$coincidBoa_{ij}$	Quantidade de vezes que uma aresta que liga as cidades *i* e *j* é repetida no grupo de soluções boas
$coincidRuim_{ij}$	Quantidade de vezes que uma aresta que liga as cidades *i* e *j* é repetida no grupo de soluções ruins
cSG	Custo da solução gulosa do problema iniciando da cidade 1
d_{ij}	Distância entre as cidades *i* e *j*
$fatAcrescimo$	Fator utilizado para incrementar a matriz de feromônios no melhor caminho encontrado
$fatDecrescimo$	Fator utilizado para decrementar a matriz de feromônios, simulando o processo de evaporação natural
$fAtualizacaoBoa_{ij}$	Fator de atualização do feromonio presente na aresta que liga as cidades *i* e *j* para as soluções boas
$fAtualizacaoRuim_{ij}$	Fator de atualização do feromonio presente na aresta que liga as cidades *i* e *j* para as soluções ruins
$fCoincidBoa$	Parâmetro de entrada do ACO-DM para determinar o peso do parâmetro $fAtualizacaoBoa_{ij}$
$fCoincidRuim$	Parâmetro de entrada do ACO-DM para determinar o peso do parâmetro $fAtualizacaoRuim_{ij}$
k	Representa uma formiga no ACO e ACO-DM
η_{ij}	Desejabilidade; dado pelo inverso da distância entre os nós *i* e *j*
$nBoasSol$	Número de soluções boas e consideradas na mineração dos dados;
$nCidades$	Número de cidades consideradas em uma instância do TSP
$nRuinsSol$	Número de soluções ruins e consideradas na mineração dos dados;
$nCoincidMaxBoa$	Maior valor possível para o parâmetro $coincidBoa_{ij}$
$nCoincidMaxRuim$	Maior valor possível para o parâmetro $coincidRuim_{ij}$
$nFormigas_i$	Número de formigas que iniciam seu circuito a partir de uma cidade *i*
$nPercMedRuim$	Quantidade média de vezes que uma aresta que liga as cidades

	i e *j* pode aparecer no grupo de soluções ruins
$nPercMedBoa$	Quantidade média de vezes que uma aresta que liga as cidades *i* e *j* pode aparecer no grupo de soluções boas
totalFormigas	Número total de formigas considerado no TSP
totalFormigasAux	Variável auxiliar contendo o número total de formigas considerado no TSP
$nTotalPercBoa$	Soma total do número de vezes que cada aresta aparece no grupo de soluções boas
$nTotalPercRuim$	Soma total do número de vezes que cada aresta aparece no grupo de soluções ruins
$p_{ij}^{k}(t)$	Probabilidade de estando no nó *i* ir ao nó *j* na iteração *t* com a formiga *k*
$pBoasSol$	Parâmetro de entrada do ACO-DM, informando a proporção de soluções boas na mineração dos dados.
$pRuinsSol$	Parâmetro de entrada do ACO-DM, informando a proporção de soluções ruins na mineração dos dados.
$solucao_i$	*I*-ésima cidade visitada por uma formiga e compõe uma solução para uma instância do TSP.
t	*Representa uma iteração no ACO e ACO-DM*
$totalC$	Número total de caminhos possíveis entre duas cidades adjacentes (número total de arestas).
α	Fator de ajuste utilizado para compatibilizar a grandeza dos fatores de interesse no cálculo de $p_{ij}^{k}(t)$.
β	Fator de ajuste utilizado para compatibilizar a grandeza dos fatores de interesse no cálculo de $p_{ij}^{k}(t)$.

1 INTRODUÇÃO

A seguir apresenta-se uma breve introdução sobre a metaheurística colônia de formigas, mineração de dados e o problema do caixeiro viajante.

1.1 A metaheurística Colônia de Formigas

Colônia de formigas, ou ACO (*Ant Colony Optimization*) é uma técnica de otimização baseada em um modelo biológico: no comportamento de algumas espécies de formigas na busca por alimento. O ACO é uma metaheurística[1] proposta originalmente por Dorigo et al. (1991) para solucionar o problema do caixeiro viajante e a partir daí, muitos autores tem desenvolvido variações do algoritmo original para diversas aplicações. Uma boa introdução sobre o assunto pode ser encontrada em Dorigo et al. (1991) e um estudo mais aprofundado pode ser encontrado em Bonabeau et al. (1999).

Formigas são capazes de encontrar um caminho entre sua colônia e uma fonte de alimento sem se basear em referências visuais uteis daquela região. Acontece que conforme formigas seguem um determinado caminho, depositam no solo uma substância química chamada de feromônio e outras formigas, ao captarem essa substância, sofrem mudanças em seus comportamentos. O feromônio, nesse caso, funciona como um elo de comunicação entre as formigas. Inicialmente, formigas partem de seu ninho em caminhos aparentemente aleatórios, e com o passar do tempo, tendem a seguir trilhas cuja concentração de feromônio seja maior. Conforme mais formigas são atraídas por esse caminho, maior a concentração de feromônio, e novamente cada vez mais formigas são atraídas por esse caminho. Esse mecanismo permite às formigas localizar o alimento de maneira eficiente.

[1] Uma definição de metaheurística, bem aceita nos meios acadêmicos, pode ser encontrada em Glover e Kochenberger (2002) “Uma metaheurística é um conjunto de conceitos que podem ser utilizados para definir métodos heurísticos aplicáveis a um extenso conjunto de diferentes problemas. Em outras palavras, uma metaheurística pode ser vista como uma estrutura algorítmica geral que pode ser aplicada a diferentes problemas de otimização com relativamente poucas modificações que possam adaptá-las a um problema específico”.

Entre um ninho de formigas e o alimento, existem diversos caminhos possíveis para uma formiga seguir. Quando em um caminho longo, formigas demoram

mais tempo para ir até a fonte de alimento (destino) e retornar ao seu ninho (origem), ao passo que em caminhos mais curtos, formigas realizam essa trajetória em menos tempo e passam mais vezes pelo mesmo local. Devido a esse maior transito de formigas, os caminhos mais curtos tendem a ter uma maior concentração de feromonio e, por isso, maior probabilidade de serem escolhidos por todo o grupo de formigas com o passar do tempo. A Figura 1.1 ilustra formigas decidindo-se por um caminho.

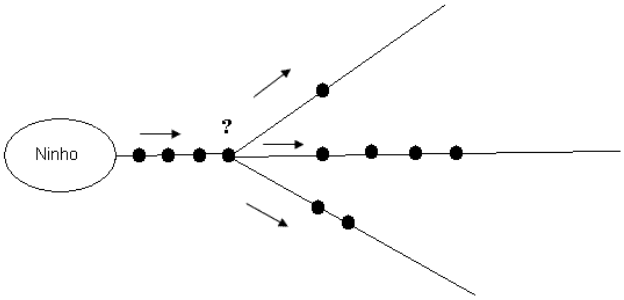

Figura 1.1 - Formigas precisam decidir-se por um caminho até chegar ao alimento. Quanto maior a concentração de feromônio em um caminho, maior probabilidade dele ser escolhido.

O feromônio que é depositado em solo também sofre evaporação e com o passar do tempo, caminhos não transitados passam a ter quantidades cada vez menores dessa substância até não serem mais visitados por praticamente nenhuma formiga. Nota-se que as decisões individuais das formigas com essa influência coletiva, tendem a convergir para uma única decisão: a de trilhar o melhor caminho. A Figura 2 mostra caminhos possíveis entre um ninho e uma fonte de alimento; nessa figura, o caminho mais escuro é o caminho mais curto e o escolhido pelo grupo de formigas, após um determinado tempo de busca pelo alimento.

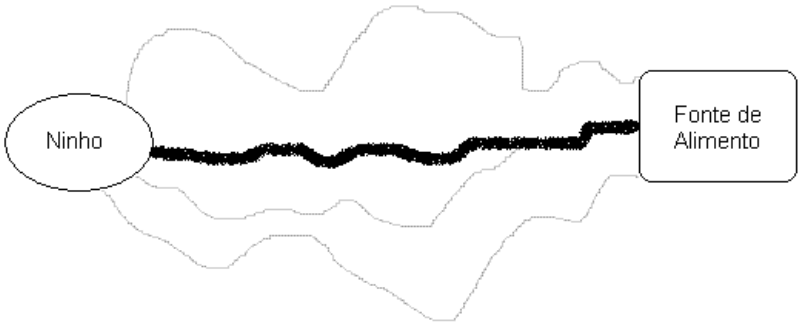

Figura 1.2- Dentre os caminhos possíveis, com o passar do tempo, formigas tendem a trilhar o menor caminho (em destaque na cor mais escura).

1.2 O Problema do Caixeiro Viajante

O problema do caixeiro-viajante, ou TSP (*Travelling Salesman Problem*) consiste na busca de um caminho de menor distância, começando em uma cidade qualquer, visitando cada cidade precisamente uma vez e retornando a cidade original (Nilsson, 1982). Caminhos como esse (não necessariamente o ótimo) são denominados de circuito Hamiltoniano, que, devido ao contexto, daqui em diante serão referenciados pelo termo “circuito”.

O TSP é um dos problemas de otimização mais investigados, por cientistas, matemáticos e pesquisadores de diversas áreas, tais como: logística, genética e produção (Applegate et al, 2006). O esforço computacional necessário para a sua resolução cresce em ordem fatorial com o tamanho do problema; de fato, esse problema pertence à categoria NP-difícil (mais detalhes sobre complexidade podem ser encontrados em Garey e Jonhson, 1978). A maior parte das instâncias de problemas do TSP são computacionalmente difíceis de serem resolvidas, isso é, difícil de obter soluções ótimas. Por isso, métodos de solução heurísticos são utilizados na obtenção de soluções aproximadas, na esperança dessas serem boas soluções. Busca-se com heurísticas, em particular metaheurísticas, o melhor que se poderia obter com recursos disponíveis como, por exemplo, de tempo e de processamento computacional.

1.3 Mineração de Dados

Mineração de dados, ou DM (*Data Mining*) pode ser entendida como uma tecnologia constituída de um conjunto de técnicas que permitem a extração de conhecimentos relevantes que estão escondidos em grandes volumes de dados. Algumas dessas técnicas são: Classificação; Predição; Agrupamento; Sumarização; Detecção de Desvios; e Identificação de Associações. (GOLDSCHMIDT; PASSOS, 2005); (NILSSON, 1982).

D entre essas técnicas, focaliza-se nesse trabalho a Identificação de Associações, por ser utilizada no procedimento heurístico proposto para resolver o problema do caixeiro viajante.

Algoritmos para encontrar associações, procuram identificar quais e/ou quando eventos estão associados, baseados em padrões nesses eventos. No caso do problema tratado nesse trabalho, deseja-se encontrar associações relacionando padrões de caminhos em um grupo de soluções factíveis geradas pela metaheurística ACO, com soluções boas para uma determinada instância do problema TSP.

2 O PROCEDIMENTO PROPOSTO

Na busca por boas soluções a metaheurística ACO e suas variações consideram, a cada iteração, informações dos melhores caminhos encontrados. Entretanto, não consideram uma análise de padrões existentes nas soluções geradas. Esses padrões podem indicar características de boas soluções, ou então, de soluções indesejadas. Em posse dessas informações, pode-se explorar melhor essas características na geração de novas soluções em iterações futuras**.** O procedimento proposto baseia-se nessa idéia.

Nas subseções seguintes, apresenta-se a modelagem da metaheurística ACO para resolver o TSP e também o procedimento proposto para melhorar o desempenho desta metaheurística na resolução deste mesmo problema. Embora o TSP seja o problema focalizado nesse trabalho, a técnica proposta pode ser diretamente estendida para resolver outros problemas de otimização combinatória.

2.1 Modelagem do ACO para resolver o TCP

Uma formiga pode ser entendida como um caixeiro viajante que precisa visitar todas as cidades apenas uma única vez, de maneira que a cidade de origem e a cidade de destino sejam as mesmas. Deseja-se encontrar o menor circuito que satisfaça a essas exigências. Para tal, isso é, para que uma formiga construa soluções para o TSP, fazem-se as seguintes considerações:

Ao se mover pelo caminho que liga um nó *i* até um nó *j*, uma formiga deposita nele uma determinada quantidade de feromônio;

É mantida uma lista *J* com as cidades que ainda não foram visitadas por uma formiga em seu caminho, evitando que a formiga passe por uma cidade mais de uma vez, o que geraria soluções infactíveis para o TSP.

Para que uma formiga possa escolher a próxima cidade a ser visitada, considera-se uma função de probabilidade que leva em conta a quantidade de

feromônio e também a distância entre nós, de maneira que caminhos mais próximos e/ou com maior quantidade de feromônio tenham maior probabilidade de serem escolhidos. A função de probabilidade utilizada é definida em (2.1).

$$p_{ij}^{k}(t) = \frac{\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{l \in j_i^k}[\tau_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}} \tag{2.1}$$

em que:

$p_{ij}^{k}(t)$ é a probabilidade de estando no nó *i* ir a um nó *j* na iteração *t* com a formiga *k;*

$\tau_{ij}(t)$ a quantidade de feromônio entre os nós *i* e *j* na iteração *t*;

η_{ij} é o parâmetro chamado desejabilidade dado pelo inverso da distância entre os nós *i* e *j* (quando maior a distância menor o desejo de ir a um nó), isso é $\eta_{ij} = \frac{1}{d_{ij}}$;

β e α são fatores de ajustes utilizados para compatibilizar a grandeza dos fatores de interesse.

Uma formiga partindo de uma cidade decide aleatoriamente qual a próxima cidade a ser visitada (daquelas que não foram visitadas ainda) baseando-se na função de probabilidade dada em (2.1). Esse procedimento é feito iterativamente, até que a formiga tenha percorrido todas as cidades e a cidade de destino seja a mesma de origem.

Um pseudocódigo descrevendo como uma formiga realiza um circuito a partir de sua cidade de origem é descrito a seguir:

Início_Método geraCaminho(*atual,solucao)*

Marque todas as cidades como não visitadas;

Marque a cidade de origem como visitada;

Faça *i* = 0;

$solucao_i$= *atual*;

Para *i* =1 até *nCidades*-1 **Faça**

Calcule a probabilidade de escolher cada cidade não visitada a partir da cidade *atual* de acordo com (2.1);

Sorteie um número correspondente a cidade a ser visitada com base na probabilidade calculada;

Atribua o número sorteado à variável *atual*;

$solucao_i$=*atual*;

Marque a cidade *atual* como visitada;

Fim_Para

Fim_Método

Pseudocódigo 2.1 - método para gerar circuitos, soluções factíveis para o TSP.

em que:

nCidades é o número de cidades de uma instancia do TSP;

solução é um vetor que armazena uma solução para uma instância do TSP;

$solucao_i$ é a i-ésima cidade visitada por uma formiga e compõe uma solução para uma instância do TSP.

Para avaliar uma solução do TSP verificamos o seu custo, dado pelo comprimento total do circuito percorrido.

Para que seja calculada $p_{ij}^{k}(t)$ pela primeira vez é necessário conhecer $\tau_{ij}(t=0)$, a matriz inicial de feromônio. Neste trabalho, inicializa-se essa matriz de acordo com a sugestão de Bonabeau et al (1999) através da expressão (2.2).

$$\tau_{ij}(0) = \begin{cases} \dfrac{1}{nCidades * cSG}, \forall\, i,j \in \{1,\dots,nCidades\} \mid i \neq j \\ 0, \quad \forall\, i,j \in \{1,\dots,nCidades\} \mid i = j \end{cases} \tag{2.2}$$

em que:

cSG: custo da solução gulosa do problema partindo da cidade 1 (ou nó 1).

Uma iteração no ACO consiste em todas as formigas realizarem um circuito. A quantidade de formigas (*totalFormigas*) é um parâmetro de entrada para a metaheurística e cada uma delas está alocada a uma determinada cidade (que será origem e destino no seu caminho). Para esse trabalho procura-se distribuir a quantidade total de formigas igualmente por todas as cidades consideradas em um exemplar do TSP. Para isso, adiciona-se uma formiga para cada cidade, sequencialmente, iniciando pela primeira cidade. Após adicionar uma formiga, se houver, na última cidade retorna-se à primeira, e repete-se esse procedimento até que todas as formigas tenham sido distribuídas pelas cidades. Um pseudocódigo deste procedimento utilizado é representado a seguir.

Início_Método distFormigas(*totalFormigasAux*)
    Para i=0 até *nCidades*-1 **Faça**
        $nFormigas_i$ =0;
    Inicio_Enquanto (*totalFormigasAux*)
        Para i=0 até *nCidades*-1 **Faça**
            Se (*totalFormigasAux*)
                $nFormigas_i$= $nFormigas_{i+1}$;
                totalFormigasAux=*totalFormigasAux*-1;
            Caso contrario Fim do método;
        Fim_Para
    Fim_Enquanto
Fim_Método

Pseudocódigo 2.2 - método de distribuição de formigas pelas cidades do TSP

em que:

totalFormigasAux é uma variável auxiliar contendo o número de formigas considerado no problema;

$nFormigas_i$ é o número de formigas que iniciam seu circuito a partir de uma cidade *i*.

A cada iteração, quando todas as formigas tiverem realizado um único circuito cada, considera-se um acréscimo de feromônio nas arestas pertencentes ao melhor caminho encontrado, o caminho cujo custo (dado pela distância total) é o menor. Esse acréscimo é realizado de acordo com (2.3).

$$\tau_{ij}(t+1) = (1 + fatAcrescimo) * \tau_{ij}(t), \quad (2.3)$$
$$\forall\, i,j \in \{1, \dots, nCidades\} \mid Aresta_{ij} \in Solucao$$

em que:
$fatAcrescimo$ é um fator utilizado para incrementar a matriz de feromônios no melhor caminho encontrado;
$Aresta_{ij}$é uma aresta que liga os nós *i* e *j*

A cada iteração considera-se também um decréscimo do feromônio depositado em cada par de arestas do problema, de acordo com (2.4).

$$\tau_{ij}(k+1) = (1 + fatDecrescimo) * \tau_{ij}(k), \forall\, i,j \in \{1, \dots, nCidades\} \mid i \neq j \quad (2.4)$$

em que:
$fatDecrescimo$ é um fator utilizado para decrementar a matriz de feromônios, simulando o processo de evaporação natural.

A cada iteração armazena-se a melhor solução e custo encontrado, isso é, o menor circuito encontrado pelas formigas e seu respectivo tamanho. Armazena-se também a iteração na qual essa solução foi encontrada.

A seguir apresenta-se um pseudocódigo descrevendo a sistemática da metaheurística ACO considerada.

Início_Método principalACO()

Leia os parâmetros do problema e inicialize as variáveis utilizadas nesse método;

Inicialize a matriz de feromonios de acordo com (2.2);

```
    distFormigas(nTotalFormigas); // distribui as formigas pelas cidades de acordo
    com Pseudocodigo 2.2
    para t =0 até nIteracoes-1 Faça
       para i=0 até nCidades-1 Faça
            para k=0 até nFormigas_i-1 Faça
                geraCaminho(i, solucao); // Gera um circuito a partir da cidade i de
                acordo com o Pseudocódigo 2.1 atribuindo ao vetor solução;
                cSolucao=verificaCusto(solucao); //obtem o custo do caminho gerado;
                Se cSolucao<menorCusto;
                      menorCusto=cSolucao;
                Fim_Se
            Fim_Para
        Fim_Para
                Incremente a matriz de feromonio de acordo com (2.3);
                Decremente a matriz de feromonio de acordo com (2.4);
    Fim_Para
Fim_Método
```

Pseudocódigo 2.3 – método principal para o ACO

2.2 O procedimento proposto para melhorar o desempenho do ACO

O procedimento proposto para resolver o TSP pode ser entendido como uma variação do ACO em que as soluções geradas num certo número de iterações são mineradas em busca padrões que estejam associados a boas soluções. Vamos chamá-lo de ACO-DM. A seguir apresentam-se os detalhes deste procedimento.

Para poder buscar padrões nas soluções, inicialmente toda vez que uma solução é gerada em uma iteração *t* qualquer, ela é armazenada em um *pool* de soluções. Ao fim de *nIteracoesDM* iterações, aplica-se um procedimento para analisar as soluções armazenadas em busca de padrões. O número de

iterações (*nIteracoesDM*) antes de minerar as soluções é um parâmetro de entrada para o ACO-DM.

Para poder analisar as soluções armazenadas, inicialmente, essas são colocadas em ordem decrescente de acordo com seu custo associado, isto é, as melhores soluções ficam em primeiro.

Para as $nSolBoas$ melhores soluções, soma-se o total de vezes que cada uma das arestas do problema foi percorrida. Armazena-se essa quantidade em uma estrutura de dados. Esse mesmo procedimento é repetido para as $nSolRuins$ piores soluções, armazenando esse resultado em outra estrutura de dados.

Após esse procedimento, incrementa-se/decrementa-se a matriz de feromônios em um percentual calculado com base no número de vezes que cada aresta foi percorrida no grupo de soluções boas e no grupo de soluções ruins.

Para que uma aresta seja considerada como uma indicação de uma solução boa/ruim ela deve aparecer, respectivamente, no grupo de soluções boas/ruins mais do que o número médio de vezes que uma aresta é percorrida nesses grupos. A partir daí, quanto mais vezes um caminho aparece no grupo de soluções considerado, mais esse caminho caracteriza uma solução boa/ruim e, respectivamente, maior o incremento/decremento de feromonio nessa aresta.

O procedimento detalhado da atualização da matriz de feromônios é descrito a seguir:

1- Calcule os parâmetros a serem utilizados na atualização da matriz de feromônios de acordo com (2.5-2.13);

$$c = \lceil pSolBoas * nCidades \rceil \quad (2.5)$$

$$nSolRuins = \lceil pSolRuins * nCidades \rceil \quad (2.6)$$

$$nCoincidMaxBoa = nSolBoas - nPercMedBoa \quad (2.7)$$

$$nCoincidMaxRuim = nSolRuins - nPercMedRuim \quad (2.8)$$

$$nPercMedBoa = \frac{nTotalPercBoa}{totalC} \quad (2.9)$$

$$nPercMedRuim = \frac{nTotalPercRuim}{totalC} \quad (2.10)$$

$$nTotalPercBoa = nSolBoas * (nCidades - 1) \quad (2.11)$$

$$nTotalPercRuim = nSolRuins * (nCidades - 1) \quad (2.12)$$

$$totalC = nCidades^2 - nCidades \quad (2.13)$$

2- Determine um fator de atualização do feromônio para cada arestas do problema de acordo com (2.14, 2.15);

$$fAtualizacaoBoa_{ij} = fCoincidBoa\left(\frac{coincidBoa_{ij} - nPercMedBoa}{nCoincidMaxBoa}\right), \quad (2.14)$$

$$\forall\, i,j \in \{1, \dots, nCidades\} \mid coincidBoa_{ij} > nPercMedBoa$$

$$fAtualizacaoRuim_{ij} = fCoincidRuim\left(\frac{coincidRuim_{ij} - nPercMedRuim}{nCoincidMaxRuim}\right), \quad (2.15)$$

$$\forall\, i,j \in \{1, \dots, nCidades\} \mid coincidRuim_{ij} > nPercMedRuim$$

3- Atualize a matriz de feromônios de acordo com (2.16, 2.17).

$$\tau_{ij}(t+1) = \tau_{ij}(t) * \left(1 - fAtualizacaoRuim_{ij}\right), \forall\, i,j \in \{1, \dots, nCidades\} \quad (2.16)$$

$$\mid coincidBoa_{ij} > nPercMedBoa$$

$$\tau_{ij}(t+1) = \tau_{ij}(t) * \left(1 + fAtualizacaoBoa_{ij}\right), \forall\, i,j \in \{1, \dots, nCidades\} \quad (2.17)$$

$$\mid coincidRuim_{ij} > nPercMedRuim$$

em que:

$nBoasSol$ e $nRuinsSol$ são respectivamente o número de soluções boas e ruins consideradas na mineração dos dados;

$\mathrm{pBoasSol}$ e $p\mathrm{RuinsSol}$ são parâmetros de entrada para o ACO-DM, informando, respectivamente, a proporção de soluções boas e ruins consideradas na

mineração dos dados tal que $\mathrm{pBoasSol}, \mathrm{pRuinsSol} \in [0,1]$ (0% e 100%, respectivamente);

$\mathrm{fAtualizacaoBoa_{ij}}$ e $\mathrm{fAtualizacaoRuim_{ij}}$ são fatores de atualização tal que $\mathrm{fAtualizacaoBoa_{ij}}, \mathrm{fAtualizacaoRuim_{ij}} \in [0,1]$ (0% e 100%, respectivamente);

$\mathrm{fCoincidBoa}$ e $\mathrm{fCoincidRuim}$ são parâmetros de entrada para o ACO-DM para determinar o peso dos parâmetros $\mathrm{fAtualizacaoBoa_{ij}}$e $\mathrm{fAtualizacaoRuim_{ij}}$ na atualização da matriz de feromônios;

$\mathrm{coincidBoa_{ij}}$ e $\mathrm{coincidRuim_{ij}}$ são as matrizes que armazenam a quantidade de vezes que cada aresta aparece no grupo de soluções boas e ruins, respectivamente;

$nCoincidMaxBoa$ e $nCoincidMaxRuim$ são os maiores valores possíveis para os parâmetros $\mathrm{coincidBoa_{ij}}$ e $\mathrm{coincidRuim_{ij}}$ $\forall\, i, j \in \{1, \dots, nCidades\}$, respectivamente;

$\mathrm{nPercMedRuim}$ e $\mathrm{nPercMedBoa}$ são as quantidade média de vezes que uma aresta qualquer que liga cidades i e j pode aparecer no grupo de soluções boas e ruins, respectivamente;

$\mathrm{nTotalPercBoa}$ e $\mathrm{nTotalPercRuim}$ são, respectivamente, a soma total do número de vezes que cada aresta aparece no grupo de soluções boas e ruins;

totalC é o número total de caminhos possiveis entre duas cidades adjacentes (número total de arestas).

Após a atualização da matriz de feromônios, pode-se pagar todas as soluções do *pool* ou apagar apenas as soluções desnecessárias, isso é, aquelas que não estão entre as $nBoasSol$ soluções boas nem entre $nRuinsSol$ ruins. Para esse trabalho, apagam-se todas as soluções do pool após utilizá-las, por ter apresentado os melhores resultados para o conjunto de parâmetros de entrada considerado.

A seguir apresenta-se um pseudocódigo descrevendo a sistemática do ACO-DM considerado.

Início_Método principalACO-DM()

Leia os parâmetros do problema e inicialize as variáveis utilizadas nesse método;

Inicialize a matriz de feromônios de acordo com (2.2);

distFormigas(*nTotalFormigas*); // distribui as formigas pelas cidades de acordo com Pseudocodigo 2.2

para *t* =0 até *nIteracoes*-1 **Faça**

para i=0 até *nCidades*-1 **Faça**

para *k*=0 até $nFormigas_i$ -1 **Faça**

geraCaminho(*i*, *solucao*); // Gera um circuito a partir da cidade i de acordo com o Pseudocódigo 2.1 atribuindo ao vetor solução;

cSolucao=verificaCusto(*solucao*); //retorna o custo do caminho gerado;

Se *cSolucao*<*menorCusto*;

menorCusto=*cSolucao*;

Fim_Se

Armazene a solução em uma estrutura de dados

Fim_Para

Fim_Para

Incremente a matriz de feromonio de acordo com (2.3);

Decremente a matriz de feromonio de acordo com (2.4);

Se o resto da divisão *t* /*nIteracoesDm* for zero **Faça**

Ordene as soluções armazenadas de forma decrescente de acordo com seu custo associado;

Defina os grupos de soluções boas e ruins sendo, respectivamente, as *nSolBoas* primeiras e as *nSolRuins* últimas do grupo de soluções ordenado;

Para toda aresta $Aresta_{ij}$ pertencente ao grupo de soluções boas e para toda aresta $Aresta_{ij}$pertencente ao grupo de soluções ruins armazene, respectivamente, em $coincidBoa_{ij}$ e $coincidRuim_{ij}$ o número de vezes que cada uma delas é repetida nos respectivos grupos de soluções;

```
            Atualize a matriz de feromônio de acordo com o procedimento descrito
            na seção 2.2;
            Apague todas as soluções da estrutura de dados;
        Fim_Se
    Fim_Para
Fim_Método
```

Pseudocódigo 3.1 – método principal para o ACO

3 RESULTADOS COMPUTACIONAIS

Nesta seção, apresentam-se resultados computacionais comparando as metaheurísticas ACO-DM e ACO na resolução do TSP.

O código utilizado em todas as implementação deste trabalho foi escrito em linguagem C++ e executado em um computador com processador Intel Celeron M, 1,73 Ghz com 2GB de memória RAM do tipo DDR2. O Compilador utilizado foi o Microsoft Visual C++ 2008 Express Edition. O Sistema Operacional utilizado foi o Microsoft Windows XP. Os números aleatórios foram gerados com a ajuda da função *rand*() do ANSI C.

Para realizar os testes com a heurística proposta utilizou-se as seguintes instâncias do TSP encontradas na literatura (Tabela 3.1):

Tabela 3.1 – Instâncias do TSP utilizadas neste trabalho

Nome da Instância do TSP	Número de Cidades	Descrição
u16	16	Odisséia de Ulysses (Groetschel e Padberg)
u22	22	Odisséia de Ulysses (Groetschel e Padberg)
u48	48	Odisséia de Ulysses (Groetschel e Padberg)
pr76	76	76 cidades (Padberg/Rinaldi)
ch150	150	problema de 130 cidades (Churritz)

Mais informações sobre essas instâncias podem ser encontradas em Siqueira (2005).

A Tabela 3.2 apresenta parâmetros e valores de entrada utilizados nas metaheurísticas ACO e ACO-DM para os testes computacionais apresentados nesta seção.

Tabela 3.2 – Parâmetros e valores para o ACO e ACO-DM

nIterações	***nFormigas/nCidade***	***fIncremento***	***fDecremento***	***Alfa***	***Beta***
100	2	0.07	0.08	1	5

A Tabela 3.3 apresenta outros parâmetros e valores de entrada exclusivos para a o ACO-DM para esses testes.

Tabela 3.3 – Parâmetros e valores exclusivos para o ACO-DM

nIteracoesDM	***pBoas***	***pRuins***	***fIncrementoDM***	***fDecrementoDM***
2	0,15	0,15	0,8	0,4

Nas Tabelas (3.4-3.8), apresentam-se as melhores soluções obtidas e o número de iterações necessário para encontrá-la. Essas tabelas apresentam, respectivamente, os resultados para as instâncias u16, u22, u48, pr76, ch150, descritas na Tabela 3.1. Essas tabelas apresentam também os custos da solução ótima e solução gulosa para cada uma das instâncias. O valor da solução gulosa é calculado iniciando da cidade número 1.

Nas Tabelas (3.4-3.8) s1,s2,s3,...,s25 são as sementes utilizadas para inicializar o gerador de números aleatórios das metaheurísticas. Essas 25 sementes foram geradas através de um outro gerador de números aleatórios no intervalo [0,1000], com a semente igual a “2009”, o ano de desenvolvimento deste trabalho. Nessas Tabelas, apresenta-se a média, desvio padrão e o maior e menor valor encontrado em todos os resultados para todas as sementes consideradas.

Tabela 3.4 – Resultados para a instância u16

Instância:	u16					
	ACO-DM		ACO		Ótima	Gulosa
	Iteração	Custo	Iteração	Custo	Custo	Custo
s1	25	74,3735	69	73,9998	73,9876	104,7349
s2	30	74,2333	43	74,6148	73,9876	104,7349
s3	40	73,9876	86	74,1087	73,9876	104,7349
s4	37	74,1087	29	74,1087	73,9876	104,7349
s5	49	74,0013	58	74,0013	73,9876	104,7349
s6	38	74,6008	46	74,6148	73,9876	104,7349
s7	54	74,0013	73	74,0013	73,9876	104,7349
s8	80	74,0013	47	74,6148	73,9876	104,7349
s9	33	74,0013	63	74,0013	73,9876	104,7349
s10	23	73,9998	67	74,2333	73,9876	104,7349
s11	65	73,9998	70	74,0013	73,9876	104,7349
s12	25	74,2333	49	74,6148	73,9876	104,7349
s13	78	73,9876	85	73,9998	73,9876	104,7349
s14	60	74,0013	90	73,9998	73,9876	104,7349
s15	27	74,1087	82	74,0013	73,9876	104,7349
s16	43	74,1087	91	74,1087	73,9876	104,7349
s17	29	74,6008	98	73,9876	73,9876	104,7349
s18	33	74,0013	58	74,6008	73,9876	104,7349
s19	42	74,0013	37	74,6148	73,9876	104,7349
s20	50	74,0013	53	74,0013	73,9876	104,7349
s21	69	73,9876	85	74,0013	73,9876	104,7349
s22	29	74,0013	91	74,0013	73,9876	104,7349
s23	39	74,1087	88	74,1087	73,9876	104,7349
s24	96	73,9876	72	74,3775	73,9876	104,7349
s25	28	74,0013	28	74,0013	73,9876	104,7349
MÉDIA:	**44,88**	**74,09758**	**66,32**	**74,18876**		
DESVIO P.:	19,73432	0,180293	20,77602785	0,258556		
MAIOR:	96	74,6008	98	74,6148		
MENOR:	23	73,9876	28	73,9876		

Tabela 3.5 – Resultados para a instância u22

Instância:	u22					
	ACO-DM		ACO		Ótima	Gulosa
	Iteração	Custo	Iteração	Custo	Custo	Custo
s1	91	75,3097	100	75,4545	75,3097	89,6407
s2	73	75,3097	95	76,0949	75,3097	89,6407
s3	59	75,3097	61	75,3984	75,3097	89,6407
s4	64	75,8805	73	75,8805	75,3097	89,6407
s5	86	75,3097	91	75,9543	75,3097	89,6407
s6	57	75,3984	84	75,7568	75,3097	89,6407
s7	84	75,3097	90	76,5043	75,3097	89,6407
s8	74	75,8805	74	75,8805	75,3097	89,6407
s9	43	75,6243	98	76,043	75,3097	89,6407
s10	66	75,3097	91	75,8099	75,3097	89,6407
s11	57	75,4897	68	76,3022	75,3097	89,6407
s12	72	75,3097	94	75,9543	75,3097	89,6407
s13	75	75,3984	86	75,6345	75,3097	89,6407
s14	56	75,3984	99	76,0416	75,3097	89,6407
s15	55	75,3097	86	76,1224	75,3097	89,6407
s16	69	75,3984	76	76,1304	75,3097	89,6407
s17	66	75,6243	100	75,8835	75,3097	89,6407
s18	43	75,3097	84	76,0325	75,3097	89,6407
s19	41	75,3984	99	75,5432	75,3097	89,6407
s20	59	75,3984	92	76,4445	75,3097	89,6407
s21	65	75,3097	83	75,9683	75,3097	89,6407
s22	37	75,3097	71	75,3097	75,3097	89,6407
s23	59	75,9321	97	75,9543	75,3097	89,6407
s24	50	75,8805	82	75,9692	75,3097	89,6407
s25	91	75,3097	77	75,9104	75,3097	89,6407
MÉDIA:	**63,68**	**75,45675**	**86,04**	**75,91912**		
DESVIO P.:	14,99089	0,214262	11,07354	0,292778		
MAIOR:	91	75,9321	100	76,5043		
MENOR:	37	75,3097	61	75,3097		

Tabela 3.6 – Resultados para a instância u48

Instância:	u48					
	ACO-DM		ACO		Ótima	Gulosa
	Iteração	Custo	Iteração	Custo	Custo	Custo
s1	39	33792,1	98	33792,1	33523,7	40526,42
s2	47	33856,8	89	34510,8	33523,7	40526,42
s3	38	33826	95	33724,4	33523,7	40526,42
s4	98	34036,9	64	35494,6	33523,7	40526,42
s5	42	34745,4	93	34049,2	33523,7	40526,42
s6	42	34292,6	87	34733,4	33523,7	40526,42
s7	77	34175,1	96	33659,7	33523,7	40526,42
s8	56	34448,8	82	33967,8	33523,7	40526,42
s9	61	34562,4	97	34423,3	33523,7	40526,42
s10	80	33743,7	84	34955,1	33523,7	40526,42
s11	47	34292,6	99	34043,6	33523,7	40526,42
s12	57	34069,4	95	34647,9	33523,7	40526,42
s13	66	34317,7	80	34325,4	33523,7	40526,42
s14	73	34300,3	66	34325,4	33523,7	40526,42
s15	88	34173,5	82	34655,2	33523,7	40526,42
s16	32	33869	76	34163,4	33523,7	40526,42
s17	43	34852,5	97	34367,2	33523,7	40526,42
s18	52	34036,7	91	34659,3	33523,7	40526,42
s19	44	34349,2	91	33736,6	33523,7	40526,42
s20	55	34119,6	81	34119,6	33523,7	40526,42
s21	52	33953,6	47	34613,7	33523,7	40526,42
s22	54	33826	93	33743,7	33523,7	40526,42
s23	56	34423,7	94	34457,4	33523,7	40526,42
s24	51	33903,1	94	33853,7	33523,7	40526,42
s25	90	34292,6	77	33689,6	33523,7	40526,42
MÉDIA:	**57,6**	**34170,37**	**85,92**	**34268,48**		
DESVIO P.:	17,59498	297,5674	12,55959	459,7228		
MAIOR:	98	34852,5	99	35494,6		
MENOR:	32	33743,7	47	33659,7		

Tabela 3.7 – Resultados para a instância pr76

Instância:	pr76					
	ACO-DM		ACO		Ótima	Gulosa
	Iteração	Custo	Iteração	Custo	Custo	Custo
s1	92	115786	94	115548	108159	153461,9
s2	96	112896	97	118463	108159	153461,9
s3	97	111209	91	118715	108159	153461,9
s4	99	113646	87	114929	108159	153461,9
s5	91	112714	82	118821	108159	153461,9
s6	81	112957	84	118496	108159	153461,9
s7	100	111020	95	118700	108159	153461,9
s8	97	113689	85	115519	108159	153461,9
s9	80	112880	93	116836	108159	153461,9
s10	80	115378	82	116399	108159	153461,9
s11	100	112784	92	114500	108159	153461,9
s12	77	109775	89	116337	108159	153461,9
s13	98	115610	97	116860	108159	153461,9
s14	100	111020	95	118700	108159	153461,9
s15	97	113689	85	115519	108159	153461,9
s16	80	112880	93	116836	108159	153461,9
s17	80	115378	82	116399	108159	153461,9
s18	100	112784	92	114500	108159	153461,9
s19	77	109775	89	116337	108159	153461,9
s20	98	115610	97	116860	108159	153461,9
s21	100	111020	95	118700	108159	153461,9
s22	97	113689	85	115519	108159	153461,9
s23	80	112880	93	116836	108159	153461,9
s24	80	115378	82	116399	108159	153461,9
s25	100	112784	92	114500	108159	153461,9
MÉDIA:	**91,08**	**113089,2**	**89,92**	**116689,1**		
DESVIO P.:	9,210139	1779,991	5,227491	1453,438		
MAIOR:	100	115786	97	118821		
MENOR:	77	109775	82	114500		

Tabela 3.8 – Resultados para a instância ch150

Instância:	ch150					
	ACO-DM		ACO		Ótima	Gulosa
	Iteração	Custo	Iteração	Custo	Custo	Custo
s1	76	6689,56	84	7062,47	6532.28	8194,614
s2	86	6715,53	81	7237,66	6532.29	8194,614
s3	93	6713,48	94	7199,2	6532.30	8194,614
s4	49	6782,21	95	6848,65	6532.31	8194,614
s5	57	6706,23	93	7202,28	6532.32	8194,614
s6	96	6736,87	99	6844,9	6532.33	8194,614
s7	88	6680,3	99	7260,64	6532.34	8194,614
s8	53	6719,31	96	7115,79	6532.35	8194,614
s9	82	6746,03	99	7022,98	6532.36	8194,614
s10	85	6721,59	97	6951,12	6532.37	8194,614
s11	76	6726,45	71	7056,98	6532.38	8194,614
s12	70	6756,66	91	6915,74	6532.39	8194,614
s13	51	6727,52	97	7124,36	6532.40	8194,614
s14	78	6690,93	100	7058,36	6532.41	8194,614
s15	63	6803,06	98	6892,67	6532.42	8194,614
s16	80	6794,92	73	7050,02	6532.43	8194,614
s17	84	6729,6	99	7063,16	6532.44	8194,614
s18	97	6695,89	84	6916,5	6532.45	8194,614
s19	73	6695,89	100	6884,32	6532.46	8194,614
s20	92	6696,66	89	7132,32	6532.47	8194,614
s21	96	6794,35	95	7322,62	6532.48	8194,614
s22	57	6694,66	92	7124,12	6532.49	8194,614
s23	98	6718,05	96	6970,31	6532.50	8194,614
s24	81	6750,01	98	7122,28	6532.51	8194,614
s25	66	6723,67	100	7337,27	6532.52	8194,614
MÉDIA:	**77,08**	**6728,377**	**92,8**	**7068,669**		
DESVIO P.:	15,3159	35,07263	8,190442	142,6657		
MAIOR:	98	6803,06	100	7337,27		
MENOR:	49	6680,3	71	6844,9		

Das Tabelas (3.4-3.8) observa-se que para todas as instâncias avaliadas o ACO-DM obteve, em média, soluções melhores do que o ACO, e a diferença percentual entre essas soluções, com exceção da instância u48, aumenta em função do tamanho do problema (com o aumento do número de cidades).

Para a grande maioria dessas instâncias, o ACO-DM encontra a melhor solução em um menor número de iterações quando comparado com o ACO.

Em quase todas as instâncias avaliadas, o ACO-DM obtém melhores resultados para os menores e maiores valores de solução e também para os menores e maiores números de iterações. O desempenho com relação a esses valores aumenta quando instâncias maiores são consideradas.

Para o caso das instâncias u16 e u22, tanto o ACO quando o ACO-DM obtém a solução ótima em pelo menos uma das sementes consideradas, enquanto que para as instâncias u48, pr76 e ch150 a diferença das melhores (menores) soluções obtidas e suas respectivas soluções ótimas no caso do ACO-DM são, respectivamente 0,65%, 1,49%, 2,26% e para o ACO são, respectivamente: 0,40%, 5,86%, 4,78%.

Para todos os casos o desvio padrão do ACO-DM foi maior que o do ACO.

4 CONCLUSÃO

Neste trabalho, apresentou-se um procedimento baseado em uma técnica de DM com o objetivo de melhorar o desempenho do ACO na resolução do TSP. A metaheurística ACO considerando esse procedimento, ACO-DM, apresenta, em média, melhores resultados comparado a quando esse procedimento não é utilizado.

A partir desses resultados, o ACO-DM demonstrou convergir para uma solução mais rapidamente para a maioria dos casos considerados e essas soluções são, em média, melhores que aquelas geradas pelo ACO.

Como trabalho futuro pretende-se:

- tentar melhorar o procedimento proposto considerando a busca por padrões de mais de 2 elementos;
- avaliar outras metaheurísticas clássicas como Algoritmos Genéticos, *Simulated Annealing* e algumas mais recentes como Bando de Pássaros e Vermes Australianos;
- verificar o desempenho do procedimento proposto no caso de outros problemas de otimização combinatória.

5 REFERÊNCIAS BIBLIOGRÁFICAS

APPLEGATE, D. L.; BIXBY, R. E.; Chvátal, V.; & Cook, W. J. (2006). **The travelling salesman problem:** a computational study. New Jersey: Princeton University Press, 2006.

BONABEAU, E.; DORIGO, M.; THERAULAZ, G. **Swarm intelligence:** from natural to artificial systems. New York: Oxford University Press, 1999.

DORIGO, M.; MANIEZZO, V.; COLORNI, A. **Ant system**: an autocatalytic optimizing process**.** Italia: Dipartimento di Elettronica e Informazione, Politenico di Milano, 1991. 91-016. Relatório técnico nº 91-016.

GAREY, M. R.; JOHNSON, D. S. **Computers and intractability:** a guide to the theory of NP-completeness. New York: W.H. Freeman & Company,1979

GLOVER, F.; KOCHENBERGER, G. A. **Handbook of metaheuristics**. Boston: Kluwer Academic Publishers, 2002.

GOLDSCHMIDT R.; PASSOS, E. **Data mining**: um guia prático. Editora Campus. Rio de Janeiro: Elsevier, 2005.

NILSSON, N. **Principles of artificial intelligence**. Heidelberg: Springer, 1982.

SIQUEIRA, H. P. **Uma nova abordagem na resolução do problema do caixeiro viajante**. Tese (Doutorado em Métodos Numéricos para Engenharia), UFPR, Curitiba. 2005.