A Importância da Informação Heurística Visibilidade para Algoritmos Baseados em Otimização por Colônia de Formigas Aplicados a Domínios Contínuos*

Cassio Rodrigo Conti[1], **Mauro Roisenberg**[1]

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brazil

{cassio,mauro}@inf.ufsc.br

Abstract. *Ant Colony Optimization (ACO) is an optimization metaheuristic based in the foraging behavior of ants. This metaheuristic was originally proposed to find good solutions to discrete combinatorial problems. Many extensions of the ACO for continuous domains have been proposed, but even those with close similarity with classical (discrete domain) ACO do not use the heuristic information called visibility commonly used in the original ACO algorithm. In this paper we show the importance of the visibility in ACO and propose its implementation for a continuous domain ACO algorithm. The improvement in convergence speed is shown in results of experiments when the visibility heuristic is used.*

Resumo. *Otimização por Colônia de Formigas (sigla em inglês, ACO) é uma meta-heurística baseada no comportamento das formigas na busca por alimento e foi originalmente desenvolvida para encontrar boas soluções em problemas de otimização combinatória (domínio discreto). Extensões do ACO para trabalhar diretamente no domínio contínuo têm surgido, entretanto as propostas mais similares à ideia clássica não usam a informação heurística chamada "visibilidade", geralmente presente em algoritmos de ACO discreto. Neste trabalho é mostrada a importância da visibilidade em domínio discreto e proposta sua implementação em domínio contínuo. Resultados dos experimentos mostram melhora na velocidade de convergência com o uso da visibilidade.*

1. Introdução

A otimização é uma área de pesquisa que apresenta rápido crescimento e onde o objetivo dos problemas é encontrar o melhor conjunto/combinação de elementos que compõem uma solução válida. Nestes problemas esse conjunto de valores para as variáveis, ou seja, essa solução, tem um custo numérico associado a cada possível conjunto e a complexidade destes problemas cresce exponencialmente à medida que cresce o número de variáveis a serem otimizadas. Em alguns casos, quando há um grande número de variáveis, o tempo necessário para encontrar a melhor solução por meio de uma busca exaustiva pode ser grande o suficiente para ser considerado inviável [Vannimenus and Mézard 1984].

Técnicas de otimização são propostas para resolver esta classe de problemas em uma quantidade de tempo viável. Estas técnicas normalmente não garantem que a melhor

*Este trabalho foi parcialmente financiado pela CAPES e pelo termo de cooperação PETROBRAS / UFSC 0050.0061468.10.9.

solução seja encontrada, mas, ao menos, uma boa solução. *Ant Colony Optimization* (ACO) é uma meta-heurística de otimização proposta por [Dorigo 1992] e que se baseia no comportamento de forrageamento das formigas. Esta técnica foi aplicada a vários problemas e mostrou ser um eficaz processo de otimização para problemas de otimização discreta [Dorigo et al. 1991] [Dorigo et al. 1996].

Na área da otimização, existem problemas em que os valores para as variáveis estão em um domínio contínuo. Para aplicar algoritmos de otimização discreta nestes problemas, as variáveis precisam ser discretizadas, o que não é sempre conveniente, especialmente se o intervalo de valores possíveis das variáveis é grande ou se é necessária uma precisão muito alta [Socha and Dorigo 2006].

Métodos baseados na meta-heurística de ACO para resolverem problemas em domínios contínuos foram propostos em diversos trabalhos, como por exemplo, [Bilchev and Parmee 1995], [Huang and Hao 2006], [Kong and Tian 2006], [Madadgar and Afshar 2008] e [Mathur et al. 2000]. Entretanto, de acordo com [Socha and Dorigo 2006], estas propostas não mantêm uma estreita semelhança com o ACO clássico ou não o seguem precisamente. Em [Socha 2004] foi proposta uma técnica de otimização baseada em ACO para domínio contínuo chamada $ACO_{\mathbb{R}}$. Embora essa implementação seja muito parecida com as definições do ACO clássico, ela não usa uma característica importante dos algoritmos de ACO; um termo conhecido como *visibilidade*. A visibilidade é qualquer informação do ambiente que possa guiar os agentes (formigas) para regiões melhores[1] do espaço de busca, a fim de aumentar a velocidade de convergência em bons resultados.

Neste artigo, é mostrada a importância do termo visibilidade para um ACO clássico e estendido o conceito dessa informação heurística a um ACO de domínio contínuo. É descrita uma possível interpretação da visibilidade em domínio contínuo e a respectiva implementação é aplicada a alguns experimentos com a finalidade de mostrar melhora na velocidade de convergência quando esse recurso é usado.

2. Trabalhos Relacionados

2.1. Otimização por Colônia de Formigas

Introduzido por Marco Dorigo em sua tese de doutorado [Dorigo 1992], ACO é um método probabilístico que tem sido amplamente utilizado em uma vasta classe de problemas de otimização combinatória. Esta meta-heurística tenta explorar o comportamento das formigas no processo de forrageamento em que as formigas caminham entre o ninho e a fonte de alimento, utilizando caminhos diferentes, cada caminho com sua chance própria de ser escolhido e, no decorrer do tempo, cada caminho pode tornar-se mais ou menos desejável (menores caminhos ganham mais chances de serem escolhidos). Este comportamento emerge porque as formigas depositam uma substância chamada feromônio no caminho por onde passam e a quantidade desta substância pode ser notada por outras formigas que, probabilisticamente, escolhem o caminho a seguir usando essa informação. As formigas que utilizam os caminhos mais curtos vão e voltam em menos tempo e acumulam mais feromônio naquele percurso do que aquelas que utilizam caminhos mais longos. Mais feromônio significa mais chances de um caminho ser escolhido.

[1] Áreas do domínio onde os valores para as variáveis implicam em uma saída numericamente desejável para a função objetivo.

[Dorigo et al. 1996] explica com mais detalhes como o menor caminho emerge do comportamento das formigas e faz a analogia com as formigas artificiais que são utilizadas na meta-heurística ACO.

Existe outra importante característica dos algoritmos de ACO, que é o uso de um termo heurístico chamado *visibilidade* ou *atratividade*. Este termo é qualquer informação do ambiente que reflita uma expectativa da medida de custo sobre quão boa determinada escolha provavelmente será durante a construção de uma solução e é uma informação que melhora a busca das formigas artificiais. Em problemas como do Caixeiro Viajante (*Traveling Salesman Problem* - TSP) a distância entre as cidades é comumente usada em trabalhos na literatura como sendo essa informação heurística (por exemplo, [Dorigo et al. 1996]) e dessa forma cidades mais próximas da cidade em que a formiga se encontra recebem maiores chances de serem escolhidas, além da chance definida pelo feromônio. O mesmo ocorre para o Problema do Menor Caminho (*Shortest Path Problem*) onde a distância entre os nós/vértices é usada como informação heurística. No Problema da Mochila (*Knapsack Problem*) a relação custo/benefício de cada item pode ser usada como a visibilidade.

A escolha de cada componente da solução que está sendo construída é feita pela fórmula introduzida na Equação 1, onde p_{ij} é a probabilidade do componente j ser escolhido, dado que i foi escolhido na etapa anterior, f_{ij} é o feromônio referente a aresta/caminho ij, v_{ij} é a visibilidade[2], α e β controlam a importância do feromônio e da visibilidade respectivamente, e $permitidos$ é o conjunto de componentes que podem ser escolhidos pela formiga[3].

$$p_{ij} = \begin{cases} \frac{f_{ij}^{\alpha} \cdot v_{ij}^{\beta}}{\sum_{p \in permitidos} f_{ip}^{\alpha} \cdot v_{ip}^{\beta}} & \text{, se } j \in permitidos. \\ 0 & \text{, caso contrário.} \end{cases} \quad (1)$$

Se for colocada muita importância na visibilidade e pouca no feromônio, a busca se comportará como uma busca gulosa (*greedy search*) [Cormen et al. 2009]. Por outro lado, muita importância para o feromônio e pouca para a visibilidade implica que a informação heurística não influencia no algoritmo. Para mostrar a importância da visibilidade, o algoritmo *Ant System* (AS) [Dorigo et al. 1991] ACO foi implementado e aplicado ao TSP. Foram selecionadas algumas instâncias de problemas da TSPLIB[4]. A Tabela 1 mostra o percurso mais curto encontrado pelas formigas quando a visibilidade é usada e quando não é. Os valores estão em médias de 50 simulações independentes para cada caso e são os melhores percursos encontrados dentro de um limite de 1000 iterações. Os valores entre parênteses são o número médio de iterações necessárias até o melhor percurso da simulação ser encontrado e os valores utilizados para α e β foram 1 e 5 respectivamente, como sugerido em [Dorigo et al. 1991]. Note que a visibilidade melhorou tanto o comprimento do melhor caminho encontrado quanto o número de iterações

[2]No TSP, ij é o caminho entre as cidades i e j. Quanto maior é a distância entre as cidades, menor é o valor de v_{ij} implicando em uma menor chance de ser escolhido.

[3]No TSP, $permitidos$ é o conjunto de cidades que ainda não foram visitadas.

[4]TSPLIB é uma biblioteca de instâncias de exemplo para o Problema do Caixeiro Viajante. Pode ser acessada em http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.

necessárias para encontrá-lo. Quanto mais complexo é o problema, mais significante é a influência da visibilidade para acelerar a convergência e melhorar os resultados.

Tabela 1. Médias dos valores de menor caminho encontrado na simulação e da iteração em que o menor caminho foi encontrado para o AS ACO aplicado ao TSP.

Problema	Sem visibilidade	Com visibilidade
oliver30.tsp	898,72 [559,58]	416,28 [373,92]
eil51.tsp	1208,58 [536,28]	443,14 [373,68]
a280.tsp	28938,58 [575,34]	2892,92 [490,2]

É importante ressaltar que o mesmo número de avaliações por iteração é feito quando a visibilidade é usada e quando não é. Os valores dos demais parâmetros do algoritmo (número de agentes, taxa de evaporação, etc.) utilizados nestas simulações (Tabela 1) são os sugeridos em [Dorigo et al. 1991].

2.2. Otimização por Colônia de Formigas para Domínios Contínuos

Em problemas de otimização contínua existem variáveis que podem assumir qualquer valor real dentro de um determinado intervalo e o objetivo é maximizar ou minimizar a saída da função/problema. Algumas técnicas de otimização baseadas em ACO foram desenvolvidas visando a sua utilização diretamente em problemas de otimização contínua. [Bilchev and Parmee 1995] propuseram uma versão contínua do ACO (CACO), em que é representado um número finito de direções (vetores) a partir de um ponto base (ninho) e que evoluem com o tempo de acordo com o percurso das formigas. [Huang and Hao 2006] implementaram uma versão do ACO com uma codificação discreta das variáveis contínuas (CACO-DE). [Mathur et al. 2000] propuseram algumas alterações ao modelo CACO básico. Outros modelos foram propostos por [Monmarchè et al. 2000] (API), [Dréo and Siarry 2002] (CIAC) e por [Kong and Tian 2005] (BAS). No entanto, [Socha 2004] afirma que todas essas técnicas não representam de forma muito similar o ACO clássico e então ele propõe um modelo chamado *Extended ACO* ($\text{ACO}_{\mathbb{R}}$). O feromônio nesse modelo está relacionado a combinações de diferentes distribuições de probabilidade para cada variável [Kong and Tian 2006].

Outra extensão é proposta neste trabalho, com base no modelo desenvolvido por [Socha 2004] [Socha and Dorigo 2006] incluindo a visibilidade a fim de guiar o algoritmo de otimização para áreas promissoras do espaço de busca mais rapidamente, reduzindo o número de iterações e avaliações necessárias para alcançar boas soluções, como mostrado na seção anterior para o domínio discreto. O $\text{ACO}_{\mathbb{R}}$ é apresentado a seguir e a visibilidade proposta neste trabalho é detalhada na Seção 3.

2.2.1. $\text{ACO}_{\mathbb{R}}$

Na técnica $\text{ACO}_{\mathbb{R}}$ proposta por [Socha 2004] existe um vetor de soluções chamado *arquivo população* que é mantido durante a execução do algoritmo e que contém um número determinado de k soluções para o problema a se otimizar. Esse vetor com caminhos (soluções) das formigas é uma estrutura análoga a *matriz de feromônio* do ACO

de otimização discreta e representa o feromônio no ambiente. Além disso, o *arquivo população* é mantido ordenado o tempo todo pela qualidade da solução e as melhores soluções (que estão nas primeiras posições do vetor) são preferencialmente escolhidas, assim como acontece com os melhores caminhos encontrados em um ACO clássico.

No início do algoritmo o *arquivo população* é inicializado com k soluções s_j ($j = 1, 2, \ldots, k$), cada uma gerada com valores aleatórios para as n variáveis s_j^i ($i = 1, 2, \ldots, n$) que compõem a solução do problema. Estes valores aleatórios estão entre os valores mínimo e máximo permitidos para cada variável (restrição de domínio). Durante a execução, as soluções no arquivo são usadas para gerar novas soluções na etapa de construção e podem ser substituídas por soluções melhores construídas pelas formigas durante as iterações seguintes. A ideia desse *arquivo população* é ter uma distribuição de probabilidade diferente para cada variável do problema, como representado na Figura 1, que mostra as probabilidades individuais de cinco soluções e uma outra que é a combinação delas (núcleo) e que representa o conhecimento da colônia.

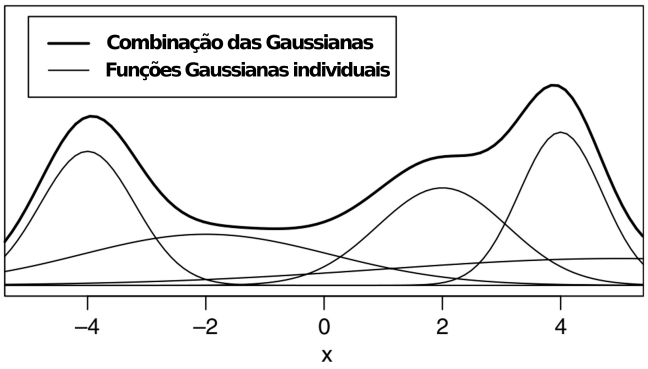

Figura 1. Exemplo de cinco funções Gaussianas e a distribuição combinada delas. Adaptado de [Socha and Dorigo 2006].

O processo de construção de novas soluções pode ser dividido em duas etapas: a primeira em que a formiga escolhe uma solução s_l ($l = 1, 2, \ldots, k$) que servirá de base para todo o processo de amostragem de uma nova solução X construída por essa formiga; e a segunda etapa que é a utilização dos valores das variáveis que compõem s_l como médias na amostragem de cada variável de X. Assim, a cada iteração, cada formiga escolhe uma solução s_l do *arquivo população*. Na Equação 2, p_l é a probabilidade da solução s_l ser escolhida e os valores ω_l e ω_j, que são definidos na Equação 3, representam a quantidade de feromônio no caminho (solução) s_l e s_j respectivamente[5]. A variável q é um parâmetro do algoritmo onde baixos valores de q implicam que as melhores soluções serão preferencialmente escolhidas pois as primeiras posições do vetor terão altos valores de ω_j, por outro lado, altos valores para q implicam que as soluções serão escolhidas com uma probabilidade mais uniforme, porque os valores de cada ω_j terão uma menor diferença entre eles.

Da solução s_l, escolhida para ser a solução base para essa formiga nessa iteração, pode-se obter um valor de média e um de desvio padrão para cada uma das n dimensões do problema, onde a média é o valor s_l^i ($i = 1, 2, \ldots, n$), que é o valor da i-ésima variável

[5] l e j são os índices das soluções no *arquivo população* e dois valores são necessários pelo fato do feromônio de cada solução ser comparado com todas as outras no arquivo.

da solução s_l e o desvio padrão é definido pela Equação 4, onde σ_l^i é o valor do desvio padrão da i-ésima variável da solução s_l e ξ é um parâmetro do algoritmo, onde um alto valor de ξ implica em uma baixa velocidade de convergência da população no algoritmo.

$$p_l = \frac{\omega_l}{\sum_{j=1}^{k} \omega_j}. \quad (2) \qquad \omega_j = \frac{1}{qk\sqrt{2\pi}} e^{-\frac{(j-1)^2}{2q^2k^2}}. \quad (3) \qquad \sigma_l^i = \xi \sum_{j=1}^{k} \frac{\left|s_j^i - s_l^i\right|}{k-1}. \quad (4)$$

Na construção de uma nova solução X, para cada variável $x_i \in X$ $(i = 1, 2, \ldots, n)$, o processo de amostragem usa, como descrito acima, o valor s_l^i como média de uma função Gaussiana (função de distribuição normal) juntamente com o valor de desvio padrão da Equação 4, para amostrar o valor de x_i. Quando todos os valores desta nova solução $X = x_1, x_2, \ldots, x_n$ forem amostrados, X é avaliado e mantido em um *arquivo população* auxiliar/temporário. Só no final da iteração, ou seja, quando todas as formigas tiverem construído suas soluções, estas novas soluções serão movidas do arquivo auxiliar para o *arquivo população*.

3. Algoritmo Proposto

Na Subsecção 2.2.1 foi mostrado que o $ACO_{\mathbb{R}}$ escolhe a solução no processo de construção utilizando apenas a informação do feromônio (Equação 2) e como foi mostrado na Subseção 2.1, o uso da visibilidade juntamente com o feromônio (Tabela 1) implica em uma significativa melhora da técnica. Nesta seção é proposta uma possível interpretação para visibilidade em domínio contínuo, com base na premissa de que a informação heurística tem grande importância na meta-heurística de ACO para acelerar a velocidade de convergência em boas soluções. A técnica apresentada nesta seção para a visibilidade utiliza a representação de feromônio proposta por [Socha 2004] para o feromônio e as implementações são comparadas na Seção 4.

3.1. Visibilidade para ACO em Domínio Contínuo

A visibilidade proposta é baseada na ideia de que, para cada solução construída pelas formigas, existe uma solução vizinha que parece ser mais promissora do que a original. Essa direção de pesquisa promissora é obtida por uma busca local na vizinhança da solução construída por cada formiga, como exemplificado na Figura 2 que apresenta uma possível evolução do algoritmo. A Figura 2a, mostra um exemplo em que as três soluções s_i (i = 1,2,3) são soluções geradas pelas formigas. A Figura 2b mostra uma possível distribuição de probabilidade de cada solução s_i referente a chance de que ela seja escolhida em uma próxima iteração para que seus valores sirvam de média em uma nova amostragem (construção de uma nova solução) no algoritmo $ACO_{\mathbb{R}}$ básico. Note que s_3 tem a maior probabilidade de ser escolhida apesar de que essa solução não se encontra em uma área promissora, pois já está muito próxima do ponto de máximo local. Voltando na Figura 2a, existem as soluções s_i' que são vizinhas geradas pela busca local na região das soluções s_i. A Figura 2c mostra a nova distribuição de probabilidade após essa informação heurística, baseada em quão promissora uma solução parece ser, ser adicionada no *arquivo população*. Agora, as chances de s_1' e s_2' serem escolhidas como valores de média para novas amostragens pelo algoritmo se tornam maiores. A ideia

dessa proposta é inserir essa informação heurística (visibilidade), apresentada no exemplo da Figura 2, no *arquivo população* a fim de acelerar a velocidade de convergência em boas soluções.

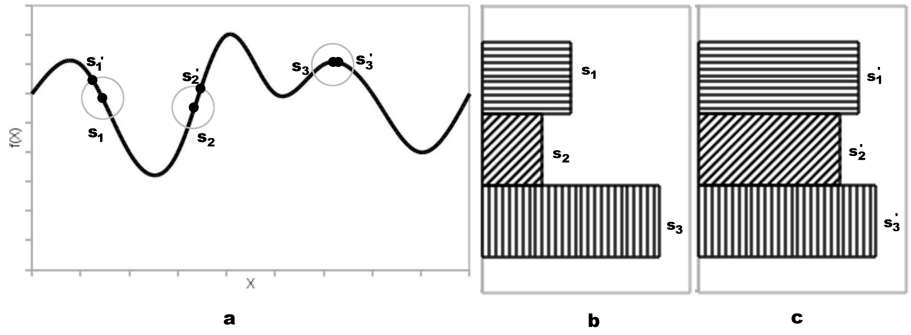

Figura 2. Probabilidade da solução ser escolhida antes e depois da influência da visibilidade.

Para aplicar ao algoritmo o comportamento descrito acima, ao final do processo de construção da solução X realizado pela formiga, uma cópia de X é feita e a busca local é realizada na vizinhança, sendo armazenada nessa cópia (X'). Esta nova solução X' é tratada como se fosse uma solução regular, gerada pelas formigas, ou seja, ela é adicionada a um arquivo auxiliar/temporário juntamente com a solução X de onde será movida para o *arquivo população* somente no final da iteração.

Em um algoritmo de ACO discreto, a visibilidade influencia principalmente nas primeiras iterações, quando o feromônio ainda não acumulou em uma determinada região do espaço de busca. A visibilidade leva os agentes a explorarem uma boa região no início e, a partir dali, as formigas acabam encontrando excelentes soluções devido ao acumulo de feromônio. De forma análoga, a visibilidade contínua dessa proposta influencia principalmente nas primeiras iterações e, após atingir um determinado limiar, ela é desligada. Desligar a visibilidade é necessário devido ao método demandar mais avaliações que o $ACO_{\mathbb{R}}$ original e também pelo fato de que, a partir de um certo ponto, a influência não é mais significativa. Deixar a visibilidade ligada só implicaria em um maior número de avaliações, consequentemente um maior tempo de processamento/execução. O desligamento da visibilidade é descrito na Subseção 3.2.

O comportamento descrito acima influenciará as soluções do algoritmo para que caminhem para boas áreas do domínio mais rapidamente e garante que essa área será explorada. Essa influência implica em um comportamento similar a influência da visibilidade no ACO clássico e, apesar de parecer simples, essa interpretação para a visibilidade apresenta resultados significativamente melhores que serão apresentados na Seção 4.

3.2. Algoritmo para a Visibilidade

A busca local na vizinhança, aplicada nas soluções geradas pelas formigas na proposta de visibilidade para que boas regiões sejam encontradas mais rapidamente, é definida no Algoritmo 1. X é a solução gerada pela formiga, X' é o melhor vizinho de X, X'' é uma solução temporária/auxiliar, X'_i ($i = 1, 2, \ldots, n$) é a i-ésima variável da solução X', Δx é o valor absoluto de variação no valor original da variável que será verificado se é melhor, $random()$ retorna um número real aleatório entre 0 e 1 e o método $melhor\text{-}entre\ ()$ testa

se a saída da solução modificada $f(X'')$ é melhor que a já conhecida $f(X')$. Ao final do algoritmo, X' será o melhor vizinho de X.

Algoritmo 1 Visibilidade - Busca local na vizinhança.

$X' \leftarrow X$
$i \leftarrow 1$
enquanto $i \leq n$ **faça**
  $X'' \leftarrow X'$
  $\Delta x \leftarrow \sigma_l^i$
  se $random() < 0{,}5$ **então**
    $X_i'' \leftarrow X_i' + \Delta x$
  se não
    $X_i'' \leftarrow X_i' - \Delta x$
  fim se
  $X' \leftarrow$ *melhor-entre*(X', X'')
  $i \leftarrow i + 1$
fim enquanto

A utilização do método $random()$ no Algoritmo 1 implica que, para cada variável, o valor de Δx será ou adicionado ou subtraído, sendo feita apenas uma alteração por variável cada vez que a visibilidade é chamada. Esta nova solução X' (melhor vizinho de X), retornada no final do algoritmo, será considerada uma solução comum, ou seja, receberá o mesmo tratamento que uma solução construída pelas formigas e competirá com todas as outras soluções do *arquivo população* para ser a solução base e seus valores servirem de média em novas amostragens.

Como citado na Subseção 3.1, a visibilidade é desligada em determinado ponto. Esse ponto ocorre quando todos os valores de desvio padrão são menores que um $limiar$ (parâmetro do algoritmo). Durante a construção da solução X pela formiga, os n valores de desvio padrão referentes as variáveis da solução base escolhida s_l são calculados e se todos eles forem menores que o $limiar$, o método para a busca local não é chamado. Além da economia no número de avaliações, consequentemente no tempo de processamento, refletida por essa atitude, o fato de que um valor de desvio padrão muito baixo implica que a busca local explorará a vizinhança com baixos valores de Δx, sendo que a própria amostragem, por ser uma distribuição normal, acabará explorando essa pequena vizinhança.

Ao final da iteração, quando todas as m formigas tiverem construído suas soluções e outras m soluções resultantes da busca local nas soluções das formigas também tiverem sido geradas, todas essas $2m$ soluções são movidas do arquivo auxiliar para o *arquivo população*. Estas soluções são então ordenadas e as $2m$ piores são descartadas da população. Dessa forma, o algoritmo sempre mantem a quantia de k soluções no *arquivo população*.

4. Testes e Resultados

Como mostrado na Subseção 2.1, o termo visibilidade é muito importante para algoritmos de ACO no domínio discreto. Para provar que este comportamento se estende a domínios

contínuos, é proposto um conjunto de experimentos para mostrar as melhoras no algoritmo usando a visibilidade. Nestes experimentos foram comparadas a proposta de visibilidade (apresentada na Subseção 3.1) com o $ACO_{\mathbb{R}}$ original (introduzido na Subseção 2.2.1).

Os problemas usados para comparar o $ACO_{\mathbb{R}}$ básico com a versão com visibilidade são todos problemas conhecidos na literatura por serem geralmente usados por autores de novas técnicas para mostrar a qualidade e permitir a comparação entre diferentes propostas. Além disso, estas mesmas funções/problemas foram utilizadas por [Socha and Dorigo 2006] e, devido a limitação de espaço, suas fórmulas são omitidas neste artigo mas podem ser encontradas no trabalho [Socha and Dorigo 2006]. Os nomes dos problemas usados, juntamente com suas respectivas dimensões e domínios são apresentados junto com as tabelas dos resultados, Tabelas 2 e 3.

Para que fosse possível a comparação com os valores divulgados por [Socha and Dorigo 2006], utilizou-se os mesmos critérios de parada, que são os definidos pelas Equações 5 e 6. A Equação 5, utilizada nos problemas *Cigar*, *Ellipsoid* e *Tablet*, têm como parâmetros f que é a saída da função objetivo para uma solução construída pelas formigas, f^* que é a melhor saída da função (valor ótimo[6]) e ϵ_1 que é o erro admitido e foi utilizado com o valor $\epsilon_1 = 10^{-10}$. A Equação 6, utilizada nos demais problemas, têm ϵ_2 e ϵ_3 que foram utilizados com o mesmo valor $\epsilon_2 = \epsilon_3 = 10^{-4}$. Todos esses valores vieram de [Socha and Dorigo 2006] e os critérios de parada implicam que o algoritmo pare quando certa precisão do valor encontrado pela solução das formigas em relação ao valor ótimo da função seja atingido, assim as soluções encontradas nas simulações realizadas neste trabalho são equivalentes as realizadas em [Socha and Dorigo 2006].

$$|f - f^*| < \epsilon_1. \qquad (5) \qquad\qquad |f - f^*| < \epsilon_2 \cdot f + \epsilon_3. \qquad (6)$$

Os valores para os parâmetros utilizados no algoritmo são os mesmos usados por [Socha and Dorigo 2006] em suas simulações: $m = 2$, $k = 50$, $\xi = 0{,}85$ e $q = 0{,}0001$. Além disso, os autores escolheram estes valores porque outras técnicas de otimização na literatura os usam, assim, seria possível uma comparação de desempenho com outras propostas sem que houvesse a necessidade de replicar os algoritmos e experimentos. Sobre os parâmetros, m é o número de formigas, k é o número de soluções no *arquivo população*, ξ influência na velocidade de convergência da população (Equação 4) e q reflete nas melhores soluções sendo preferencialmente escolhidas como solução base ou se essa probabilidade será mais uniforme (Equação 3). O parâmetro adicional desta proposta é o uso de $limiar = 1{,}5$, onde $limiar$ é um parâmetro da visibilidade que controla quando ela deve ser desligada, como explicado na Subseção 3.1, e seu valor foi definido após uma série de simulações de teste com o objetivo de encontrar um valor que retornasse o comportamento mais desejado possível para as saídas dos problemas. O valor não é o melhor para todos os problemas, mas, ao definir um único valor para todos, esse foi o que apresentou os melhores resultados. A escolha desse parâmetro pode ser relacionada a escolha de valores para α e β para definir a importância do feromônio e da visibilidade em um ACO discreto.

[6]O valor é conhecido por se tratarem de problemas muito explorados na literatura.

As Tabelas 2 e 3 mostram, respectivamente, o número de iterações e número de avaliações necessárias até o critério de parada ser atingido. Os valores referentes ao $ACO_{\mathbb{R}}$ original foram retirados de [Socha and Dorigo 2006]. Já os valores da versão com visibilidade são todos médias de 50 simulações independentes para a implementação proposta utilizando os parâmetros definidos para o experimento. É importante ressaltar que os valores do número de iterações para o $ACO_{\mathbb{R}}$ original não aparecem de forma explícita no trabalho de [Socha and Dorigo 2006], entretanto, a partir do número de agentes m e do número de soluções geradas aleatoriamente no inicio do algoritmo k, esse valor é deduzível[7]. Os valores entre parenteses são o desvio padrão para o $ACO_{\mathbb{R}}$ + visibilidade para o número de iterações na Tabela 2 e para o número de avaliações na Tabela 3. Os valores de desvio padrão referentes ao $ACO_{\mathbb{R}}$ original não são apresentados em [Socha and Dorigo 2006].

Tabela 2. Comparação do número de iterações sem e com a visibilidade.

Função, dimensão, intervalo	$ACO_{\mathbb{R}}$*	$ACO_{\mathbb{R}}$ + visibilidade
B2, $n = 2$, $[-100, 100]$	247	103 (11,92)
Branin RCOS, $n = 2$, $[-5, 15]$	403,75	100,3 (31,93)
Cigar, $n = 10$, $[-3, 7]$	2663	872 (90,51)
De Jong, $n = 3$, $[-5{,}12, 5{,}12]$	171	73,02 (10,19)
Easom, $n = 2$, $[-100, 100]$	361	115,22 (13,98)
Ellipsoid, $n = 10$, $[-3, 7]$	5760	660,26 (77,08)
Goldstein and Price, $n = 2$, $[-2, 2]$	167	78,64 (10,22)
Martin and Gaddy, $n = 2$, $[-20, 20]$	147,5	73,98 (12,52)
Sphere model, $n = 6$, $[-5{,}12, 5{,}12]$	365,5	142,92 (19)
Tablet, $n = 10$, $[-3, 7]$	1258,5	674,46 (75,63)
Zakharov, $n = 2$, $[-5, 10]$	121,25	59,88 (10,19)

* Valores desta coluna foram divulgados por [Socha and Dorigo 2006].

Tabela 3. Comparação do número de avaliações sem e com a visibilidade.

Função, dimensão, intervalo	$ACO_{\mathbb{R}}$*	$ACO_{\mathbb{R}}$ + visibilidade
B2, $n = 2$, $[-100, 100]$	544	394,88 (29,57)
Branin RCOS, $n = 2$, $[-5, 15]$	857,5	391,88 (132,43)
Cigar, $n = 10$, $[-3, 7]$	5376	2168 (222,81)
De Jong, $n = 3$, $[-5{,}12, 5{,}12]$	392	240,32 (23,72)
Easom, $n = 2$, $[-100, 100]$	772	573,8 (67,39)
Ellipsoid, $n = 10$, $[-3, 7]$	11570	1983,72 (188,64)
Goldstein and Price, $n = 2$, $[-2, 2]$	384	207,28 (20,44)
Martin and Gaddy, $n = 2$, $[-20, 20]$	345	298,28 (26,54)
Sphere model, $n = 6$, $[-5{,}12, 5{,}12]$	781	467,36 (49,08)
Tablet, $n = 10$, $[-3, 7]$	2567	2028,52 (172,87)
Zakharov, $n = 2$, $[-5, 10]$	292,5	226,56 (23,87)

* Valores desta coluna foram divulgados por [Socha and Dorigo 2006].

Analisando os resultados das Tabelas 2 e 3 é possível perceber que, apesar do número de avaliações ter diminuído para todos os problemas quando a visibilidade foi

[7]Cálculo: $iterações = \frac{avaliações - k}{2}$

usada, alguns alcançaram uma redução maior que outros. Esse comportamento pode ser relacionado ao ACO discreto, onde foi visto que dependendo da complexidade do problema, havia diferente variação da influência da visibilidade no resultado. Além disso, o número de iterações teve significante redução para todos os problemas testados com reduções de, no mínimo, 46,4%.

Embora a visibilidade proposta demande algumas avaliações a mais nas primeiras iterações em relação ao $ACO_{\mathbb{R}}$ original devido a busca local, como a visibilidade é desligada em determinado ponto durante a execução e como número de iterações é reduzido, o numero total de avaliações também é reduzido. A consequência disso é um menor tempo de execução do algoritmo. Outro fator que é importante citar é que, para alguns problemas é mais rápido realizar algumas avaliações extras do que executar uma nova iteração que amostra valores para cada variável e todos os demais passos já explicados. Esse é o caso dos problemas que foram testados, pois apesar de alguns problemas não apresentarem significativa redução do número de avaliações, o tempo de execução foi reduzido significativamente devido ao menor número de iterações.

A partir dos resultados, também pode-se concluir que a visibilidade, apesar de acelerar a convergência do algoritmo, demanda avaliações extras que, utilizando a comparação pelo número de avaliações, pode não parecer tão significativa. Entretanto, combinando essa redução com a redução do número de iterações reflete-se a importância do uso da visibilidade em algoritmos ACO de domínio contínuo, assim como acontece no domínio discreto, que é guiar o algoritmo mais rapidamente para as melhores soluções do domínio. Em todos os problemas testados a visibilidade apresentou melhores resultados que a versão original, tanto no número de avaliações, quanto no número de iterações.

5. Conclusão

Neste artigo foi apresentado como o ACO pode ser usado em ambos domínios, discreto e contínuo, e como a visibilidade pode melhorar a velocidade de convergência e os resultados encontrados pelos algoritmos.

Entre as propostas de implementações para o ACO em domínio contínuo, o $ACO_{\mathbb{R}}$ foi escolhido pela similaridade com os conceitos e aspectos originais da meta-heurística. Mesmo assim, esta técnica não inclui a visibilidade no algoritmo. Foi então proposta uma interpretação de como a visibilidade poderia ser em um domínio contínuo.

A proposta foi implementada e foram feitos testes com os mesmos problemas de otimização já apresentados e aplicados no trabalho original do $ACO_{\mathbb{R}}$. Os resultados mostraram melhora na velocidade de convergência refletida em um menor número de iterações e de avaliações necessárias para alcançar bons resultados.

Pesquisas sobre outras possíveis interpretações da visibilidade e do método de busca local que encontra o melhor vizinho estão em desenvolvimento e serão o assunto de futuras publicações.

Referências

[Bilchev and Parmee 1995] Bilchev, G. and Parmee, I. C. (1995). The ant colony metaphor for searching continuous design spaces. *AISB Workshop on Evolutionary Computing*, 993:25–39.

[Cormen et al. 2009] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Greedy Algorithms*, chapter 16. MIT Press, 3rd edition.

[Dorigo 1992] Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms*. PhD thesis, Dip. Elettronica e Informazione, Politecnico di Milano, Italy.

[Dorigo et al. 1991] Dorigo, M., Maniezzo, V., and Colorni, A. (1991). Positive feedback as a search strategy. Technical report, Dipartimento di Elettronica - Politecnico di Milano - Italy.

[Dorigo et al. 1996] Dorigo, M., Maniezzo, V., and Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 26(1):29–41.

[Dréo and Siarry 2002] Dréo, J. and Siarry, P. (2002). A new ant colony algorithm using the heterarchical concept aimed at optimization of multiminima continuous functions. In Dorigo, M., Di Caro, G., and Sampels, M., editors, *Ant Algorithms*, volume 2463 of *Lecture Notes in Computer Science*, pages 216–221. Springer Berlin / Heidelberg.

[Huang and Hao 2006] Huang, H. and Hao, Z. (2006). Aco for continuous optimization based on discrete encoding. *Ant Colony Optimization and Swarm Intelligence*, pages 504–505.

[Kong and Tian 2005] Kong, M. and Tian, P. (2005). A binary ant colony optimization for the unconstrained function optimization problem. In Hao, Y., Liu, J., Wang, Y.-P., Cheung, Y.-m., Yin, H., Jiao, L., Ma, J., and Jiao, Y.-C., editors, *Computational Intelligence and Security*, volume 3801 of *Lecture Notes in Computer Science*, pages 682–687. Springer Berlin / Heidelberg.

[Kong and Tian 2006] Kong, M. and Tian, P. (2006). A direct application of ant colony optimization to function optimization problem in continuous domain. *Ant Colony Optimization and Swarm Intelligence*, pages 324–331.

[Madadgar and Afshar 2008] Madadgar, S. and Afshar, A. (2008). An Improved Continuous Ant Algorithm for Optimization of Water Resources Problems. *Water Resources Management*, 23(10):2119–2139.

[Mathur et al. 2000] Mathur, M., Karale, S. B., Priye, S., Jayaraman, V. K., and Kulkarni, B. D. (2000). Ant colony approach to continuous function optimization. *Industrial & Engineering Chemistry Research*, 39(10):3814–3822.

[Monmarchè et al. 2000] Monmarchè, N., Venturini, G., and Slimane, M. (2000). On how pachycondyla apicalis ants suggest a new search algorithm. *Future Generation Computer Systems*, 16(9):937–946.

[Socha 2004] Socha, K. (2004). Aco for continuous and mixed-variable optimization. *Ant Colony Optimization and Swarm Intelligence*, pages 25–36.

[Socha and Dorigo 2006] Socha, K. and Dorigo, M. (2006). Ant colony optimization for continuous domains. *European Journal of Operational Research*, 185:1155–1173.

[Vannimenus and Mézard 1984] Vannimenus, J. and Mézard, M. (1984). On the statistical mechanics of optimization problems of the travelling salesman type. *Journal de Physique Lettres*, 45(24):1145–1153.