RECIT
Revista Eletrônica Científica Inovação e Tecnologia

Revista Eletrônica Científica Inovação e Tecnologia
ISSN: 2175-1846

https://periodicos.utfpr.edu.br/recitv

# ALGORÍTMO GOTAS D'ÁGUA INTELIGENTES APLICADO NA SOLUÇÃO DO PROBLEMA DA MOCHILA MULTIDIMENSIONAL

**Eduardo Attuy Carvalho**
engedu.carvalho@hotmail.com
http://orcid.org/0000-0002-4557-4169

**Aline Chaves**
alinechavesrs@yahoo.com.br
http://orcid.org/0000-0001-7739-1757

**Carlos H. Farias dos Santos**
chf.santos@uol.com.br
http://orcid.org/0000-0001-5226-0355

**Edgar M. Carreno Franco**
emfra.unioeste@gmail.com
http://orcid.org/0000-0003-0700-4701

## RESUMO

O algoritmo das gotas d'água inteligentes é uma técnica de otimização baseada no fluxo de água em rios, lagos e oceanos, a qual pode ser utilizada para solucionar o problema da mochila multidimensional. A principal característica deste método de otimização é a utilização de uma abordagem construtiva, por meio de uma população, que visa encontrar a solução ótima de um determinado problema. Baseando-se nas gotas de água que fluem na natureza, de modo que cada gota de água constrói uma solução atravessando no espaço de busca do problema e modificando seu ambiente, desta forma, levou-se em conta os parâmetros de velocidade e solo da gota alterados conforme o caminho de respostas que a mesma atravessa. Fez-se os testes e a validação do algoritmo desenvolvido mediante o software Matlab®, analisando o número mínimo de iterações para encontrar as respostas ótimas dos problemas e o tempo de simulação.

**PALAVRAS-CHAVE:** Otimização; metaheuristicas; mochila multidimensional.

## INTRODUÇÃO

No problema da mochila multidimensional trabalha-se com uma ou mais mochilas e vários itens, sendo que, cada item possui um custo e um peso. Portanto, o objetivo do algoritmo é incluir os itens na(s) mochila(s) obtendo o maior custo possível sem exceder o valor máximo de peso suportado pela mesma.

Uma forma de resolver este problema é utilizando o algoritmo Gotas d'Água Inteligentes (GAI), que é um algoritmo de otimização baseado nos fenômenos de um enxame de gotas de água que flui com o solo ao longo de um leito de rio. Esse algoritmo foi introduzido por (Shah-Hosseini, 2007). Como um método construtivo, o algoritmo GAI constrói uma solução otimizada através da cooperação entre um grupo de agentes chamados gotas de água.

Conforme dito por (Singla, Sharma, 2014), as gotas de água cooperam juntas para alcançar uma solução melhor para um determinado problema. Segundo (Bonabeau, Dorigo e Theraultz, 1999), esse algoritmo baseia-se na dinâmica dos sistemas fluviais, ações e reações que ocorrem entre as gotas de água nos rios. Pode ser usado para resolver problemas de maximização ou minimização, no qual suas soluções são construídas incrementalmente pelo algoritmo em estudo, logo, esse algoritmo é um algoritmo de otimização construtiva baseado em populações de respostas.

Para o projeto de algoritmos GAI deve-se definir as propriedades principais para as gotas: velocidade e solo, ambas as propriedades podem mudar durante o tempo de vida desse algoritmo. Conforme (Shah-Hosseini, 2009), a força gravitacional da terra proporciona a tendência para fluir em direção ao destino, então a gota de água tende a retirar solo do ambiente em que atravessa tomando para si, portanto, alterando o seu tamanho e, consequentemente, a sua velocidade ( $vel^{GAI}$), como pode-se observar na Figura 1.

Figura 1 – Travessia de uma GAI por uma borda

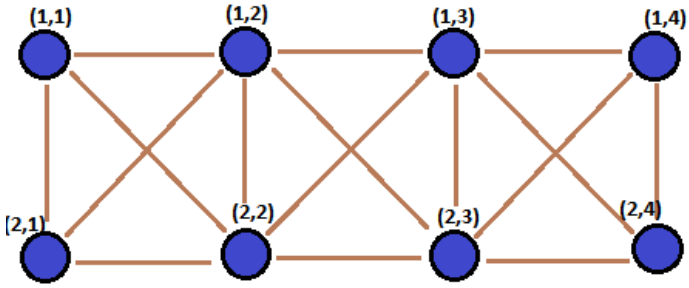

Fonte: Shah-Hosseini (2008)

Considera-se que cada gota do algoritmo GAI caminha discretamente pelas possíveis soluções do problema, partindo da sua localização atual para a sua localização seguinte, e sua velocidade é aumentada pela quantidade não linearmente proporcional ao inverso do solo entre os dois locais. Assim, um caminho com menos solo permite que esse algoritmo fique mais rápido do que um caminho com mais solo, ou seja, a velocidade de uma gota d'água inteligente que atravessa um caminho determina a quantidade de solo que foi retirado desse caminho. Em contrapartida, a velocidade da gota d'água inteligente também é alterada pelo caminho, tal que um caminho com baixa quantidade de solo aumenta a velocidade da gota d'água inteligente mais do que um caminho com uma quantidade considerável de solo (Bonabeau, Dorigo e Theraultz, 1999).

Uma gota d'água inteligente precisa de um mecanismo para escolher o caminho para seu próximo local ou etapa. Neste mecanismo, a gota d'água inteligente prefere "escolher" os caminhos que têm solos baixos do que os caminhos que têm solos altos. Esse comportamento de escolha de caminho é implementado impondo uma distribuição aleatória uniforme nos solos dos caminhos disponíveis, para este trabalho utilizou-se como mecanismo a ferramenta da roleta, onde a probabilidade de um caminho ser escolhido é diretamente proporcional à qualidade do caminho em relação aos outros (Haupt;, 2004). Então, a probabilidade do próximo caminho a ser escolhido é inversamente proporcional aos solos dos caminhos disponíveis. Portanto, caminhos com solos mais baixos têm maior chance de serem escolhidos pela gota d'água inteligente.

O algoritmo GAI obtém uma representação do problema na forma de um gráfico com o conjunto de nós $(N)$ e conjunto de bordas $(E)$. Na Figura 2 mostra-se a distribuição gráfica dos nós e caminhos para uma mochila com três itens e, como pode-se observar, as gotas caminham horizontal, vertical e diagonalmente.

Figura 2 – Distribuição gráfica das GAI




Fonte: Próprio Autor, 2016

Então, cada gota d'água inteligente começa a construir sua solução gradativamente, viajando nos nós do gráfico ao longo de suas bordas até que a gota d'água inteligente finalmente complete sua solução. Uma iteração do algoritmo está completa quando todas as gotas d'água inteligentes completarem suas soluções. O algoritmo gota d'água inteligente possui dois tipos de parâmetros. Um tipo é aquele que permanece constante durante a vida útil do algoritmo e são chamados de "parâmetros estáticos". E o outro tipo são chamados de "parâmetros dinâmicos" e são reinicializados após cada iteração do algoritmo.

Segundo (Shah-Hosseini, 2008) o algoritmo gota d'água inteligente é composto pelas seguintes etapas:

Inicialização de parâmetros estáticos. O gráfico $(N, E)$ do problema é dado ao algoritmo. A qualidade da melhor solução total $T^{TB}$ é inicialmente definido para o pior valor: $q(T^{TB}) = -\infty$. O número máximo de iterações $iter_{max}$ é especificado pelo usuário. A contagem de iteração $iter_{count}$ é definido como zero. O número de gotas de água $N_{GAI}$ é definido como um valor inteiro positivo, que geralmente é definido para o número de nós $N_c$ do gráfico. Para a atualização da velocidade, os parâmetros são $a_v = 1$, $b_v = 0{,}01$ e $c_v = 1$. Para a atualização do solo, $a_s = 1$, $b_s = 0{,}01$ e $c_s = 1$. O parâmetro local de atualização do solo $\rho_n$, que é um pequeno número positivo menor que 1, é definido como $\rho_n = 0{,}9$. O parâmetro global de atualização do solo $\rho_{GAI}$,, que é escolhido entre $[0, 1]$ é definido como $\rho_{GAI} = 0{,}9$. Além disso, o solo inicial em cada caminho (borda) é denotado pela constante $InitSoil$ tal que o solo do caminho entre cada dois nós $i$ e $j$ é definido por $soil(i, j) = InitSoil$. A velocidade inicial de cada gota d'água inteligente é

chamado de $InitVel$. Ambos os parâmetros $InitSoil$ e $InitVel$ são selecionados pelo usuário e devem ser sintonizados experimentalmente para a aplicação. Aqui, $InitSoil = 10000$ e $InitVel = 200$. Para o problema da mochila multidimensional, $InitVel = 4$ é usado, que é o mesmo valor usado em (Shah-Hosseini, 2007).

Inicialização dos parâmetros dinâmicos. Toda gota d'água inteligente tem uma lista de nós visitados $V_c(GAI)$, que é inicialmente vazia: $V_c(GAI) = \{\}$. A velocidade de cada gota d'água inteligente é denotada por $InitVel$. Todas as gotas d'água inteligentes são definidas com uma quantidade inicial de solo nula.

Espalhe as gotas d'água inteligentes aleatoriamente pelos nós do gráfico como se fossem os primeiros nós visitados.

Atualize a lista de nós visitados de cada gota d'água inteligente para incluir os nós que acabamos de visitar.

Repita os passos 5.1 a 5.4 para aquelas gotas d'água inteligentes com soluções parciais:

Para a gota d'água inteligente que residente no nó $i$, escolha o próximo nó $j$, que não viola quaisquer restrições do problema e não está na lista de nó visitada $vc(GAI)$ da gota d'água inteligente, usando a seguinte probabilidade $p_i^{GAI}(j)$:

$$p_i^{GAI}(j) = \frac{f\big(soil(i,j)\big)}{\sum_{k \notin vc(GAI)} f\big(soil(i,k)\big)} \quad (1)$$

De tal modo que

$$f\big(soil(i,j)\big) = \frac{1}{\varepsilon_s + g(soil(i,j))} \text{ e}$$

$$g\big(soil(i,j)\big) = \begin{cases} soil(i,j) \ \ if \ \ min_{l \notin vc(GAI)}\,(soil(i,l)) \geq 0 \\ soil(i,j) - \ min_{l \notin vc(GAI)}\,\big(soil(i,l)\big) \ else \end{cases}$$

Em seguida, adicione o nó recentemente visitado $j$ à lista $vc(GAI)$.

Para cada gota d'água inteligente movendo-se do nó $i$ para o nó $j$, atualize sua velocidade $vel^{GAI}(t)$ para

$$vel^{GAI}(t+1) = vel^{GAI}(t) + \frac{a_v}{b_v + c_v . soil^2(i,j)} \quad (2)$$

Onde $vel^{GAI}(t+1)$ é a velocidade atualizada da gota d'água inteligente.

Para a gota d'água inteligente movendo-se no caminho do nó $i$ para $j$, calcular o solo $\Delta soil(i,j)$ que a gota d'água inteligente carrega do caminho por

$$\Delta soil(i,j) = \frac{a_s}{b_s + c_s . time^2(i,j; vel^{GAI}(t+1))} \quad (3)$$

De tal modo que

$$time\big(i,j; vel^{GAI}(t+1)\big) = \frac{HUD(j)}{vel^{GAI}(t+1)}$$

Onde a indesejabilidade heurística $HUD\ (j)$ é definida adequadamente para o problema dado.

Atualizar o $soil\ (i,j)$ do caminho do nó $i$ para $j$ percorrido por essa gota d'água inteligente e também atualizar o solo que a GAI transporta $soil^{GAI}$ de

$$soil(i,j) = (1-\rho_n).soil(i,j) - \rho_n.\Delta soil(i,j)$$
$$soil^{GAI} = soil^{GAI} + .\Delta soil(i,j) \quad (4)$$

Encontre a melhor solução de iteração $T^{IB}$ de todas as soluções $T^{GAI}$ encontradas pelas gotas d'água inteligentes

$$T^{IB} = \arg max_{\forall T^{GAI}} q(T^{GAI}) \quad (5)$$

Onde a função $q(.)$ dá a qualidade da solução.

Atualize a melhor solução total $T^{TB}$ pela melhor solução de iteração atual $T^{IB}$ usando

$$T^{TB} = \begin{cases} T^{TB}\ if\ q(T^{TB}) \geq q(T^{IB}) \\ T^{IB} de\ outra\ forma \end{cases} \quad (7)$$

Incremente o número da iteração

$$Iter_{count} = Iter_{count} + 1$$

Em seguida, vá para a Etapa 2 se

$$Iter_{count} < Iter_{max}.$$

O algoritmo para aqui com a melhor solução total $T^{TB}$.

Conforme (Shah-Hosseini, 2009) a gota d'água inteligente mostrou ter a propriedade de convergência em valor. Isso significa que o algoritmo gota d'água inteligente é capaz de encontrar a solução ideal se o número de iterações for suficientemente grande.

Conforme (Bonabeau, Dorigo e Theraultz, 1999), a otimização por GAI pode ser comparada aos algoritmos de otimização baseados em formigas, pois considerando que estas exalam feromônios nos caminhos em que se movem pode-se buscar as soluções mais atrativas para um determinado problema. Da mesma forma, as GAI alteram o solo nos caminhos por onde passam, no entanto, em contraste com as formigas, estas mudanças não são constantes e dependem da velocidade e do solo da gota d'água inteligente que atravessa o caminho. Além disso, as gotas d'água inteligentes podem variar de velocidades ao longo de cada iteração do algoritmo, enquanto que em algoritmos baseados em formigas as velocidades das formigas são irrelevantes.

O algoritmo GAI tem sido utilizado para o problema do caixeiro viajante (PCV) (Shah-Hosseini, 2008) e o problema de mochila múltipla (ou multidimensional) (PMM) (Shah-Hosseini, 2009), com resultados promissores. Tanto o problema do caixeiro viajante quanto o da mochila multidimensional são frequentemente usados para testar algoritmos de otimização. No artigo estudado, o algoritmo GAI-PCV (Shah-Hosseini, 2008) é alterado para obter melhores resultados para os problemas do caixeiro viajante. Além disso, o algoritmo GAI-PMM (Shah-Hosseini, 2009) é usado por alguns outros PMMs para analisar sua potência em obter soluções ótimas ou quase ótimas.

## O PROBLEMA

O problema da mochila multidimensional consiste em:

### 3.1) CODIFICAÇÃO

Fez-se o mapeamento do problema conforme o número de itens n e mochilas m existentes, onde o número de pontos possíveis é determinado por:

$$P = (m+1)^n$$

Caso a quantidade de itens seja par tem-se uma matriz quadrada $\boldsymbol{M_{rxr}}$, sendo que

$$r = (m+1)^{(n/2)}$$

**contudo, caso o número de itens seja ímpar a matriz será $\boldsymbol{M_{pxq}}$, onde:**

$$p = (m+1)^{(n+1)/2}$$

$$q = (m+1)^{(n-1)/2}$$

Sendo assim, quando uma GAI passa por um determinado nó (i,j) da matriz M, tem-se que os itens a serem depositados são determinados por:

*Itens = número de colunas de* $M * (i-1) + j$

Ao converter este valor para números binários, utilizando *n* como o número de casas, tem-se quais itens serão depositados na mochila. Portanto, caso o número seja dado por *n* = 3 itens para o problema de m = 1 mochila, tem-se que:

p = 4 e q = 2

Para $(i,j) = (1,1) \rightarrow$ Itens = 0, logo $[0\ 0\ 0]$

**Para $(\boldsymbol{i,j}) = (\mathbf{4,2}) \rightarrow$ Itens = 7, logo $[\mathbf{1\ 1\ 1}]$**

**Aplicou-se um contador global para caso o número de mochilas aumente.**

3.2) Cálculo da função de indesejabilidade

A função objetivo é calculada com a soma dos pesos dos itens escolhidos, que pode ser feito multiplicando o vetor Itens pelos vetores custo e peso dos itens. Dessa forma, tem-se que a função de indesejabilidade:

$$HUD = \frac{1}{\sum_{i=1}^{m} Item(i) * custo'}$$

Caso o peso dos itens (*Peso*) tenha superado o peso máximo (*pmax*) da mochila tenha sido superado tem-se um aumento da indesejabilidade da função pela seguinte operação:

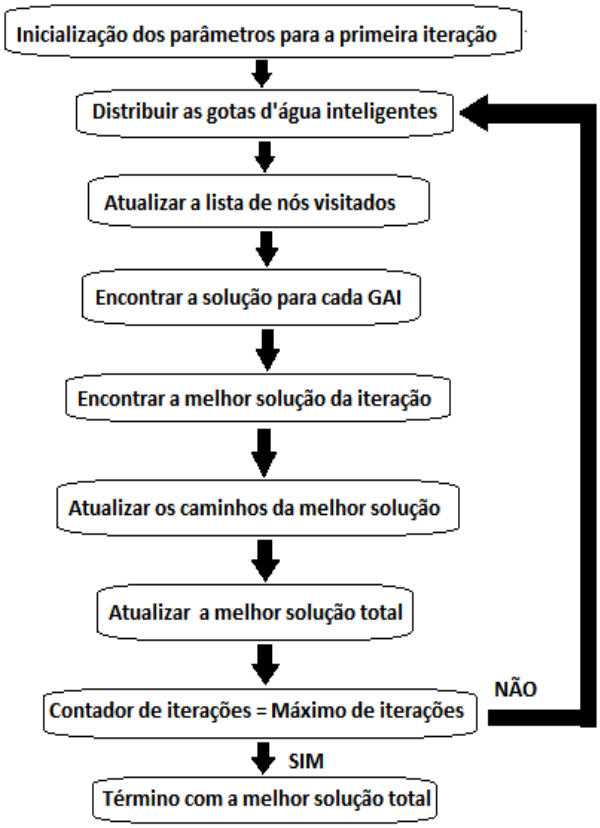
$$HUD = HUD + \sum_{i=1}^{m} (pmáx(i) - Peso(i))$$

Além disso, caso HUD seja superior à melhor solução global, essa não é aceita como a melhor solução. Este procedimento foi feito pois, apesar de uma solução superar o peso máximo suportado por uma da(s) mochila(s), pode-se ter a solução ótima próxima a uma solução inválida. Além disso, caso *Peso(i) << pmáx(i)*, tem-se que *HUD* →0.

### 3.3) TÉCNICA DE SOLUÇÃO

Com a distribuição dos nós e caminhos pode-se exemplificar o procedimento da solução do problema em um fluxograma conforme a Figura 3 (Shah-Hosseini, 2008).

Figura 3 - Fluxograma para solução por meio do algoritmo das GAI




Fonte: os autores

## TESTES E RESULTADOS

Fez-se validação do algoritmo desenvolvido mediante os casos de uma, duas e três mochilas, sendo que, para todos os casos, os pesos e custos dos itens foram os mesmos, vide a Tabela 1.

Tabela 1 – Pesos e custos dos itens adicionados na mochila

| | |
|---|---|
| Peso dos itens | 23 31 29 44 53 38 63 85 89 82 |
| Custo dos itens | 92 57 49 68 60 43 67 84 87 72 |

Fonte: Próprio Autor, 2016

Trabalhou-se com quatro casos

Tabela 2 – Configuração do problema para o primeiro caso

| | |
|---|---|
| Número de itens | 10 |
| Número de possíveis soluções | 1024 |
| Número de mochilas | 1 |
| Capacidade máxima da(s) mochila(s) | 165 |

Fonte: Próprio Autor, 2016

Tabela 3 – Configuração do problema para o segundo caso

| | |
|---|---|
| Número de itens | 10 |
| Número de possíveis soluções | 59049 |
| Número de mochilas | 2 |
| Capacidade máxima da(s) mochila(s) | 70 127 |

Fonte: Próprio Autor, 2016

Tabela 4 – Configuração do problema para o terceiro caso

| | |
|---|---|
| Número de itens | 10 |
| Número de possíveis soluções | 59049 |
| Número de mochilas | 2 |
| Capacidade máxima da(s) mochila(s) | 103 156 |

Fonte: Próprio Autor, 2016

Tabela 5 – Configuração do problema para o quarto caso

| | |
|---|---|
| Número de itens | 10 |
| Número de possíveis soluções | 9765625 |
| Número de mochilas | 3 |
| Capacidade máxima da(s) mochila(s) | 50 81 120 |

Fonte: Próprio Autor, 2016

As especificações técnicas do equipamento utilizado nas simulações, via *software* Matlab®, podem ser vistas na tabela 6:

Tabela 6 – Especificações técnicas do *Hardware* utilizado

| | |
|---|---|
| Sistema Operacional | Windows 7 |
| Memória RAM | 5 Gb |
| Processador | Intel Core i5-2410M 2.1 GHz |

Fonte: Próprio Autor, 2016

Adotando-se os valores propostos por (Shah-Hosseini, 2008), além de uma população com 10 GAI, o algoritmo gerou os resultados, apresentados nas tabelas 7, 8, 9 e 10:

Caso 1:

Tabela 7 – Resultados para o primeiro caso

| | |
|---|---|
| Número mínimo de iterações para encontrar a solução ótima | 30 |
| Solução ótima | 1 1 1 1 0 1 0 0 0 0 |
| Tempo de simulação[s] | 0,192356 |

Fonte: Próprio Autor, 2016

Caso 2:

Tabela 8 – Resultados para o segundo caso

| | |
|---|---|
| Número mínimo de iterações para encontrar a solução ótima | 5000 |
| Solução ótima | 0 0 0 0 0 0 1 0 0 0<br>1 1 1 1 0 0 0 0 0 0 |
| Tempo de simulação[s] | 0,469699 |

Fonte: Próprio Autor, 2016

Caso 3:

Tabela 9 – Resultados para o terceiro caso

| | |
|---|---|
| Número mínimo de iterações para encontrar a solução ótima | 3000 |
| Solução ótima | 0 1 1 0 0 1 0 0 0 0<br>1 0 0 1 0 0 0 0 1 0 |
| Tempo de simulação[s] | 0,733682 |

Fonte: Próprio Autor, 2016

Caso 4:

Tabela 10 – Resultados para o quarto caso

| | |
|---|---|
| Número mínimo de iterações para encontrar a solução ótima | 3000000 |
| Solução ótima | 0 0 0 1 0 0 0 0 0 0<br>0 1 0 0 0 1 0 0 0 0<br>1 0 1 0 0 0 1 0 0 0 |
| Tempo de simulação[s] | 720,311729 |

Fonte: Próprio Autor, 2016

Alterou-se o número de GAI para 100 nos casos 2, 3 e 4, visando, desta forma, reduzir o número mínimo de iterações para encontrar a solução ótima e o tempo de simulação. Os resultados após essa alteração são exibidos nas Tabelas 10, 11 e 12.

Tabela 11 – Resultados para o segundo caso após a alteração do número de GAI

| | |
|---|---|
| Número mínimo de iterações para encontrar a solução ótima | 200 |
| Solução ótima | 0 0 0 0 0 0 1 0 0 0<br>1 1 1 1 0 0 0 0 0 0 |
| Tempo de simulação[s] | 0,372996 |

Fonte: Próprio Autor, 2016

Tabela 12 – Resultados para o terceiro caso após a alteração do número de GAI

| | |
|---|---|
| Número mínimo de iterações para encontrar a solução ótima | 3000 |
| Solução ótima | 0 1 1 0 0 1 0 0 0 0<br>1 0 0 1 0 0 0 0 1 0 |
| Tempo de simulação[s] | 0,733682 |

Fonte: Próprio Autor, 2016

Tabela 13 – Resultados para o quarto caso após a alteração do número de GAI

| | |
|---|---|
| Número mínimo de iterações para encontrar a solução ótima | 5000 |
| Solução ótima | 0 0 0 1 0 0 0 0 0 0<br>0 1 0 0 0 1 0 0 0 0<br>1 0 1 0 0 0 1 0 0 0 |
| Tempo de simulação[s] | 12,710758 |

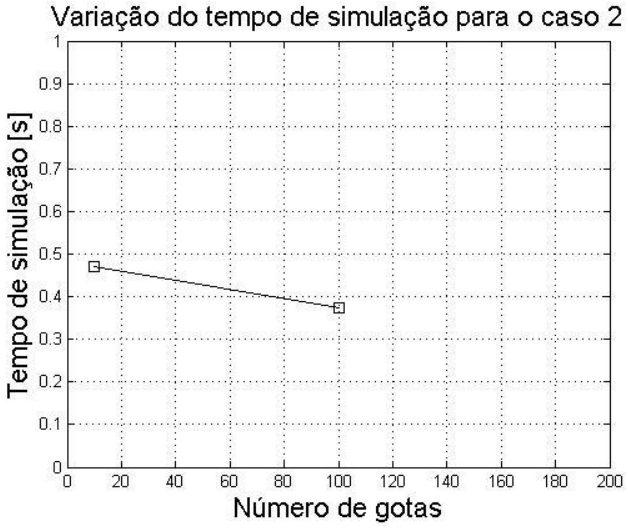
Fonte: Próprio Autor, 2016

A comparação entre os quatro casos com a variação do parâmetro número de gotas é feita nas Figuras 4,5, 6, 7, 8 e 9. Observou-se que houve uma melhora no tempo de simulação e no número mínimo de iterações para encontrar a resposta ótima, excetuando-se apenas o caso três que teve um aumento do tempo de simulação. Isto se deve ao fato de mais gotas gerarem mais cálculos, tais como de aumento e decremento de solo.

Figura 4 – Comparação do número mínimo de iterações para 10 e 100 gotas no caso 2


Fonte: Próprio Autor, 2016

Figura 5 – Comparação do tempo de simulação para 10 e 100 gotas no caso 2

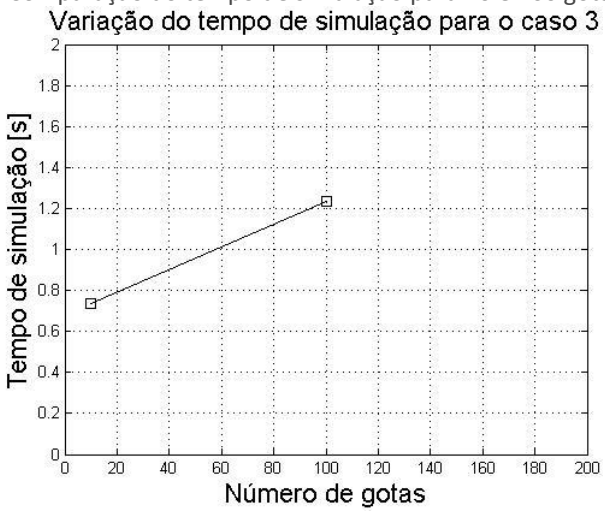


Fonte: Próprio Autor, 2016

Figura 6 – Comparação do número mínimo de iterações para 10 e 100 gotas no caso 3

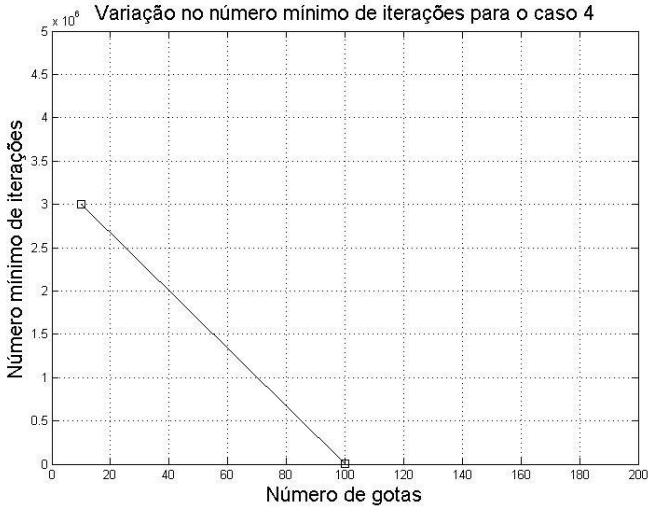

Fonte: Próprio Autor, 2016

Figura 7 – Comparação do tempo de simulação para 10 e 100 gotas no caso 3




Fonte: Próprio Autor, 2016

Figura 8 – Comparação do número mínimo de iterações para 10 e 100 gotas no caso 4




Fonte: Próprio Autor, 2016

Figura 9 – Comparação do tempo de simulação para 10 e 100 gotas no caso 3


Fonte: Próprio Autor, 2016

## COMENTÁRIOS FINAIS

Foi possível desenvolver um algoritmo de otimização para o problema da mochila multidimensional utilizando o algoritmo gotas d'água inteligentes. Mediante testes utilizando o *software* Matlab® provou-se a eficácia do algoritmo, sendo que os principais fatores levados em consideração foram o número de GAI e o número de iterações do algoritmo. Constatou-se que o caso de 3 mochilas teve um maior custo computacional que as demais, sendo justificável pelo elevado número de possíveis soluções. Concluiu-se também que em vista o número de operações existentes neste tipo de algoritmo, este deve ser aplicado em casos de elevado número de mochilas e itens.

## NOMENCLATURA

| | |
|---|---|
| $a_v, b_v, c_v$ | Parâmetros para atualização de velocidade; |
| $a_s, b_s, c_s$ | Parâmetros para atualização de solo; |
| $\rho_n$ | Parâmetro para atualização de solo local; |
| $p_i^{GAI}(j)$ | Probabilidade de uma gota ir para um nó *j;* |
| $vc(GAI)$ | Lista de nós visitados pelas gotas; |
| $T^{IB}$ | Melhor solução da iteração; |
| $T^{TB}$ | Melhor solução total; |
| $InitVel$ | Velocidade inicial das gotas; |
| $vel^{GAI}$ | Velocidade das gotas de água |
| $iter_{count}$ | Contagem de iterações |
| $soil(i,j)$ | Quantidade de solo entre os nós *i* e *j* |

# SMART WATER DROPS ALGORITHM APPLIED TO SOLVING THE MULTIDIMENSIONAL BACKPACK PROBLEM

## ABSTRACT

The algorithm of intelligent water droplets is an optimization technique based on the flow of water in rivers, lakes and oceans, which can be used to solve the problem of the multidimensional backpack. The main feature of this optimization method is the use of a constructive approach, through a population, which aims to find the optimal solution for a given problem. Based on the droplets of water that flow in nature, so that each droplet of water builds a solution crossing the space of search for the problem and modifying its environment, in this way, it took into account the speed and soil parameters of the droplet altered according to the response path that it crosses. The tests and validation of the algorithm developed using the Matlab® software were performed, analyzing the minimum number of iterations to find the optimal answers to the problems and the simulation time.

**KEYWORDS**: Optimization; metaheuristics; multidimensional backpack.

## REFERÊNCIAS

Alijla, B. O.; Wong, L. P.; Lim, C. P.; et.al. A modified Intelligent Water Drops algorithm and its application to optimization problems. Expert Systems with Applications, May, p.6555-6569, 2014.

Bonabeau, E.; Dorigo M.; Theraultz, G. Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, 1999.

Singla, N.; Sharma, S.; Study on Selection of Intelligent Water Drop Algorithm for Solving Multiple Problems: A Review. International Journal of Advanced Research in Computer Science & Technology, Vol.2, January-March, pp.60-63, 2014.

Shah-Hosseini, H. Problem solving by intelligent water drops. Proc. IEEE Congress on Evolutionary Computation, Swissotel The Stamford, Singapore, September, pp.3226–3231, 2007.

Shah-Hosseini, H. Intelligent water drops algorithm: a new optimization method for solving the multiple knapsack problem. Int. Journal of Intelligent Computing and Cybernetics, Vol. 1, No. 2, pp.193–212, 2008.

Shah-Hosseini, H. The intelligent water drops algorithm: a nature-inspired swarm-based optimisation algorithm. Int. J. Bio-Inspired Computation, Vol. 1, Nos. 1/2, pp.71–79, 2009.

Haupt R.; Haupt, S. Pratical Genetic Algorithms Second Edition, 2004

Florida State University, Data s ets, [Online]. Disponível em: <http://people.sc.fsu.edu/~jburkardt/datasets /datasets.html>. Acesso em 4 de dezembro de 2016.

VIGGIANO, C. E. O Produto Dietético no Brasil e sua Importância para Indivíduos Diabéticos, disponível em <http://www.bdbomdia.com/ed_68/terapeutica/dmnut/artigo_09.html>, acesso em 8/06/2005.

**Recebido:** 2017-12-11.
**Aprovado:** 2021-03-20.
**DOI:** 103895/recit. V11n28.7498
**Como citar:** CARVALHO. E. A. CHAVES, A SANTOS,C. H. F, FRANCO, E. F. Algoritmo gotas d'água inteligentes aplicado na solução do problema da mochila multidimensional . R. Eletr. Cient. Inov. Tecnol, Medianeira, v. 11. n. 28, p. 53- 69, set/dez, 2020 Disponível em: <https://periodicos.utfpr.edu.br/recit>. Acesso em: XXX.
**Correspondência:**
Universidade Estadual do Oeste do Paraná – Unioeste, Brasil