

# Relatório de IA sobre ant clustering

Paulo H. Gonçalves<sup>1</sup>

<sup>1</sup> Departamento de Ciência da Computação  
Universidade do Estado de Santa Catarina (UDESC) – Joinville, SC – Brazil

Paulohgon01@gmail.com

**Abstract.** *This article aims to conduct an analysis of algorithms based on ant and anthill behavior for data clustering, utilizing existing information in a random manner. The methodology of application, mathematical functions, and demonstration of results in each specific case will be explained, along with their conclusions.*

**Resumo.** *Este artigo preocupa-se em fazer uma análise sobre algoritmo que simulam o resfriamento de metais utilizando essa ideia para otimizar soluções de funções problema, fazendo uma busca global dos resultados quando a temperatura esta alta e busca local quando a temperatura esta baixa, e a temperatura abaixa em função de uma equação que leva em conta a quantidade de iterações. Será feita a comparação desse algoritmo random search para ter uma base de eficiência.*

## 1. Introdução

Simulated Annealing é um algoritmo de otimização probabilístico que é inspirado pelo processo de resfriamento controlado de um material para diminuir seu estado de energia até alcançar um estado de mínimo global. Ele é comumente utilizado para resolver problemas de otimização global, onde o objetivo é encontrar a melhor solução em um espaço de busca grande e complexo.

O algoritmo utiliza uma variável chamada "temperatura", que controla a probabilidade de aceitar soluções piores do que a solução atual que for gerada. No início, a temperatura é alta, permitindo que o algoritmo aceite soluções piores com uma alta probabilidade. Isso ajuda o algoritmo a evitar ficar preso em vales sem solução e explorar o espaço de busca de forma mais global.

À medida que o algoritmo progride, a temperatura é gradualmente reduzida de acordo com uma função de resfriamento específica. À medida que a temperatura diminui, a probabilidade de aceitar soluções piores também diminui. Isso significa que o algoritmo se torna cada vez mais focado em buscar soluções melhores e converge para um mínimo global ou uma solução ótima, dependendo do problema em questão.

A capacidade do Simulated Annealing para escapar de mínimos locais e explorar o espaço de busca de maneira eficaz faz dele uma técnica poderosa para uma variedade de problemas de otimização, como o problema do 3 sat, que será abordado nesse artigo.

## 2. Metodologia de desenvolvimento

O primeiro passo do desenvolvimento é definir a fórmula de decaimento de temperatura. Por conta da facilidade de desenvolvimento e não levar em conta a temperatura atual ou

temperatura inicial. Foi escolhido a formula

$$T = \left(1 - \frac{iteracao}{maximodeiteracao}\right)^t$$

Onde *iteracao* é a iteração atual do algoritmo e *maximodeiteracao* é o número de iterações máximas do algoritmo, aqui escolho 250.000. Além disso, *t* é a taxa de resfriamento do algoritmo, escolhido aqui arbitrariamente como 10.

Outros pontos que foram escolhidos é a frequência que a queda de temperatura chamado SAMAX, que nesse algoritmo é 5, ou seja, a cada 5 iterações a temperatura será recalculada e diminuída.

Além disso, também é escolhido a quantidade de variáveis que será trocada a cada geração de vizinho, nesse algoritmo ela é dinâmica, ou seja, ela começa com 5, e vai diminuindo até 2 no fim do algoritmo.

Agora, levando em conta essa fórmula de resfriamento, será aplicado o método simulated annealing em problemas 3-SAT com 20, 100 e 250 variáveis junto com uma análise em comparação com random search.

### 3. O problema

O problema de 3-sat é basicamente um conjunto de variáveis booleanas ligadas por operadores booleanos and, not, or. É conhecido por ser um problema NP-completo que se baseia em descobrir se existe um conjunto de valores booleanos que, quando aplicados a fórmula no modelo 3-sat resultam em verdadeiro. Um exemplo de uma fórmula 3sat é:

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$$

Por ser NP completo achar uma solução em tempo polinomial para grandes instâncias desse problema ainda é um mistério, para isso então usamos a inteligência artificial, como será demonstrado em seguinte.

### 4. Random search

O random search é um algoritmo de otimização que busca de forma completamente aleatória escolher valores que resultem em uma solução, uma maximização ou minimização de uma função problema.

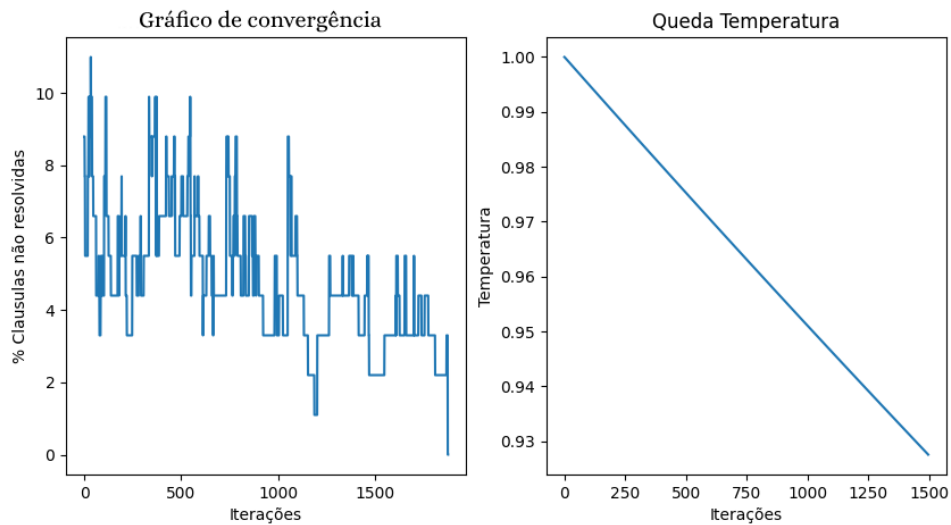
Ele é uma primeira abordagem para solução e é a base de comparação da maioria dos algoritmos de otimização. Então será o algoritmo usado para comparação nesse artigo. Utilizando as mesmas instâncias, mesmo hardware, mesma linguagem e mesmo ambiente.

### 5. Resultados

Agora será feita a análise de todas as instâncias e seus resultados, acompanhados de gráficos para simulated annealing e random search e no fim análise dos resultados com base na média e desvio padrão.

#### 5.1. 20 variáveis

Nessa instância um algoritmo qualquer já resolveria e não demoraria muito, o próprio random search resolveu a maioria das instâncias. Enquanto o simulated annealing(SA)



**Figure 1. Simulated annealing 20 variaveis**

resolveu todas. Para ter essa noção iremos analisar o gráfico de convergência de ambos os algoritmos, enquanto o simulated annealing tem também a queda temperatura. Além disso terá por fim o box plot do simulated annealing.

Analisando os gráficos podemos perceber que o simulated annealing resolveu o problema 3-sat bem antes do random search. Uma vez que o random search atingiu o 0 pouco depois das 100.000 interações enquanto o simulated annealing resolveu pouco depois das 1500 iterações. Mas, como já dito, uma instância desse problema com 20 variáveis é facilmente resolvida, vemos isso já que o simulated annealing resolveu as 10 vezes que foi executado e o random search resolveu 8. Também vemos isso ao analisar o box plot dos valores de clausulas não resolvidas do SA.

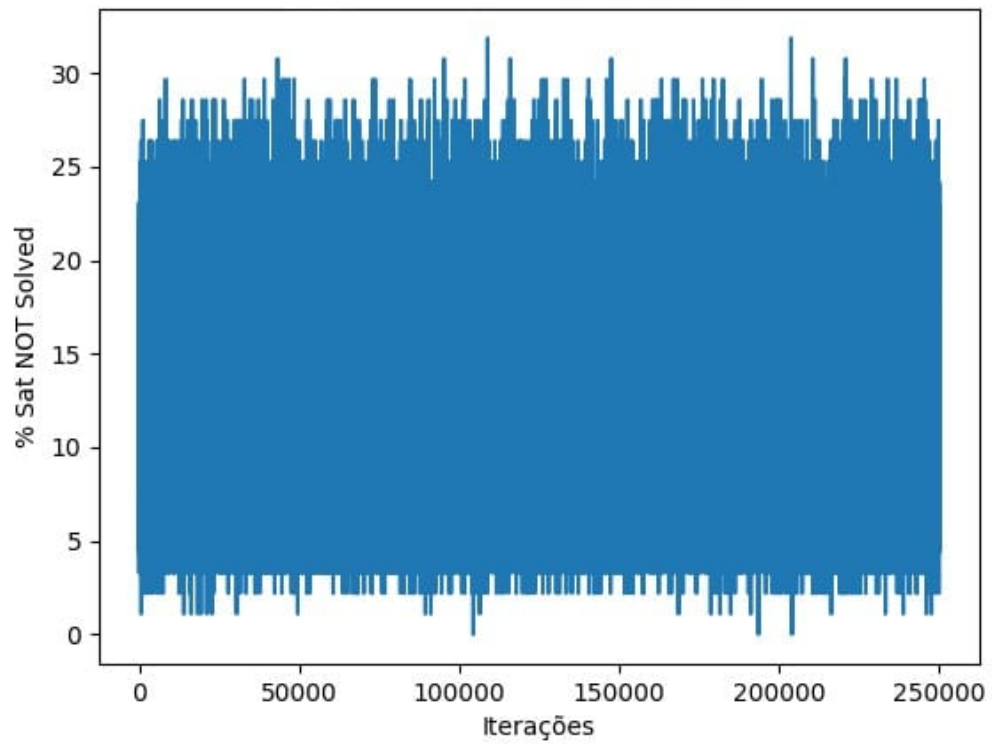
## 5.2. 100 variaveis

Já nessa quantidade de variáveis, o random search não foi capaz de resolver nenhuma das iterações, e o simulated annealing algumas. Porém nessa instância já se torna inviável a resolução não utilizando algoritmos de IA.

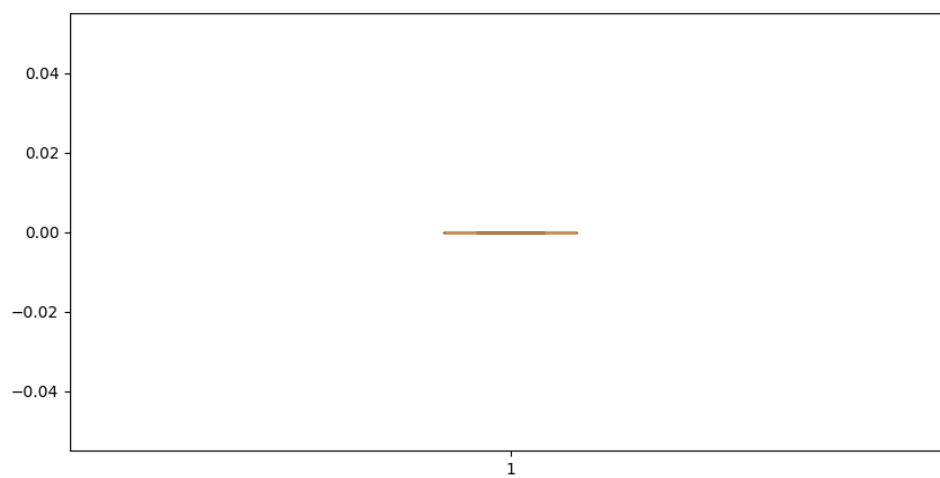
Analisar melhor esse gráfico do simulated annealing, na figura 4, para compreender o seu funcionamento. quando comparamos o grafico de convergencia com a queda de temperatura, é notável que, com a temperatura mais alta, a variância de resultados é muito maior, ou seja o algoritmo está fazendo uma busca global, e conforme a temperatura vai caindo, ele muda para uma busca mais local e com menos variância de resultados, convergindo por fim ao resultado de resolver todas as clausulas, nesse ponto o algoritmo para a execução.

Enquanto o random search na figura 5 não resolveu nenhuma das 10 vezes que foi executado, ficando geralmente acima de 90% de resolução. Mas já é perceptível por que usar um algoritmo mais inteligente como o SA.

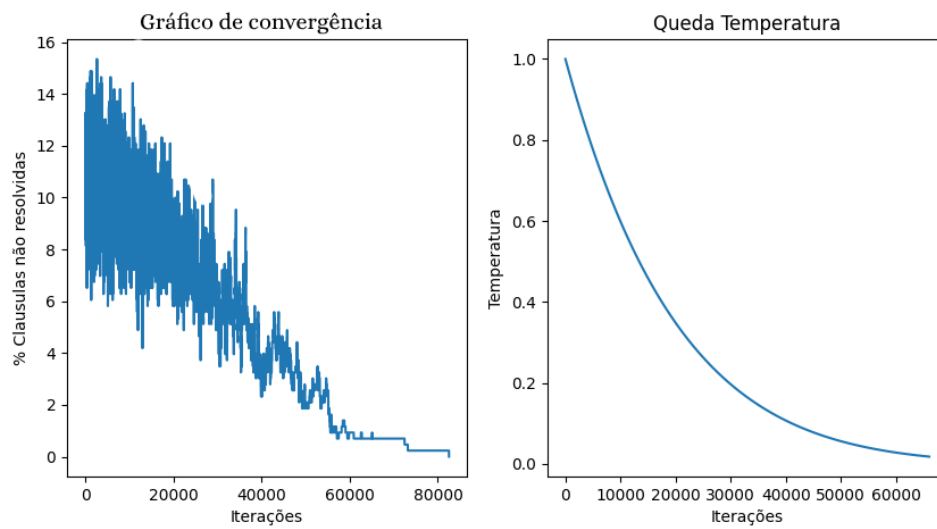
Por fim temos também o boxplot na figura 6 dos retornos de todas as 10 execuções do simulated annealing, onde podemos analisar melhor as respostas. Nessa instância,



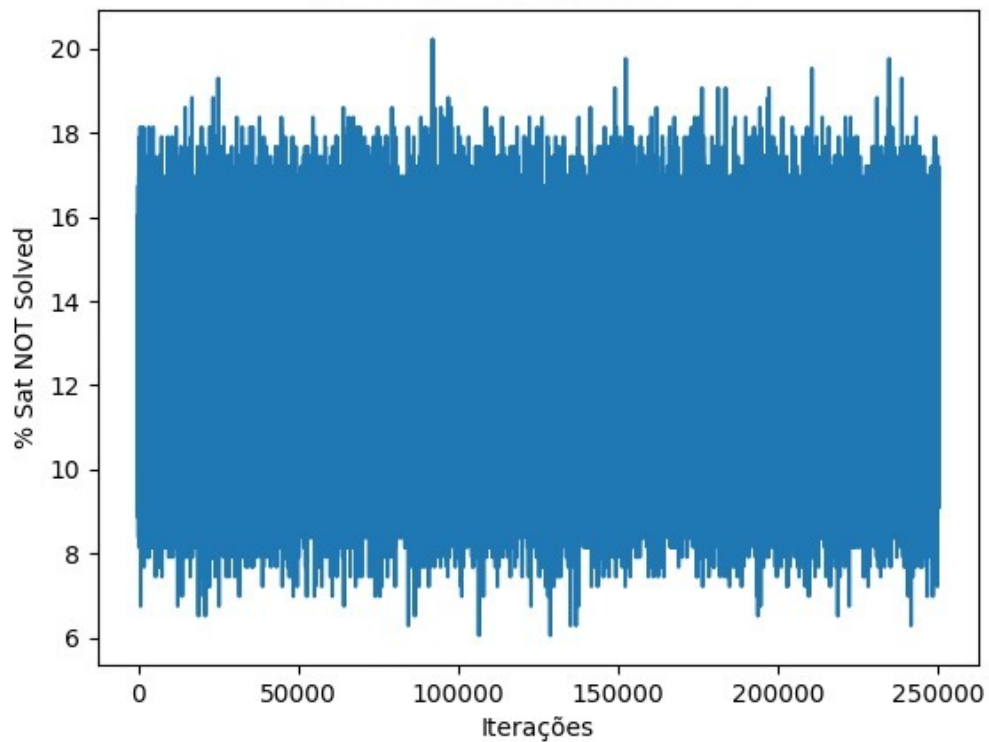
**Figure 2. Random search 20 variaveis**



**Figure 3. Boxplot dos valores de clausulas não resolvidas do SA**

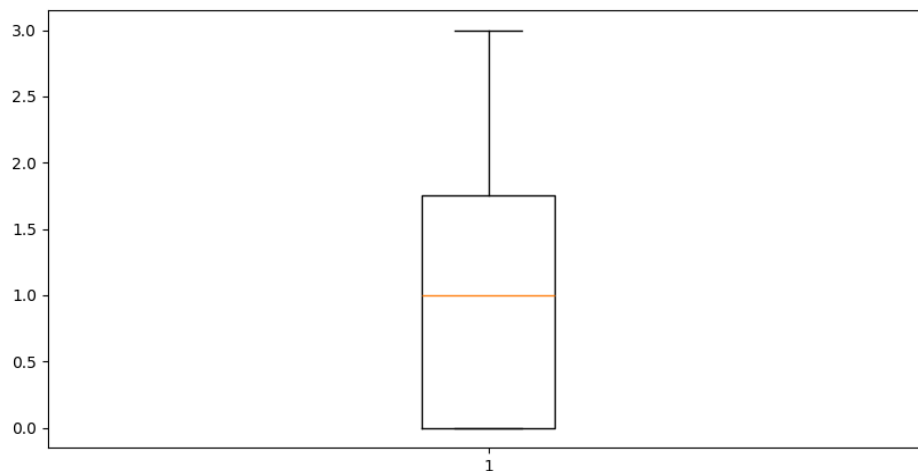


**Figure 4. Simulated annealing para 100 variaveis**



**Figure 5. Enter Caption**

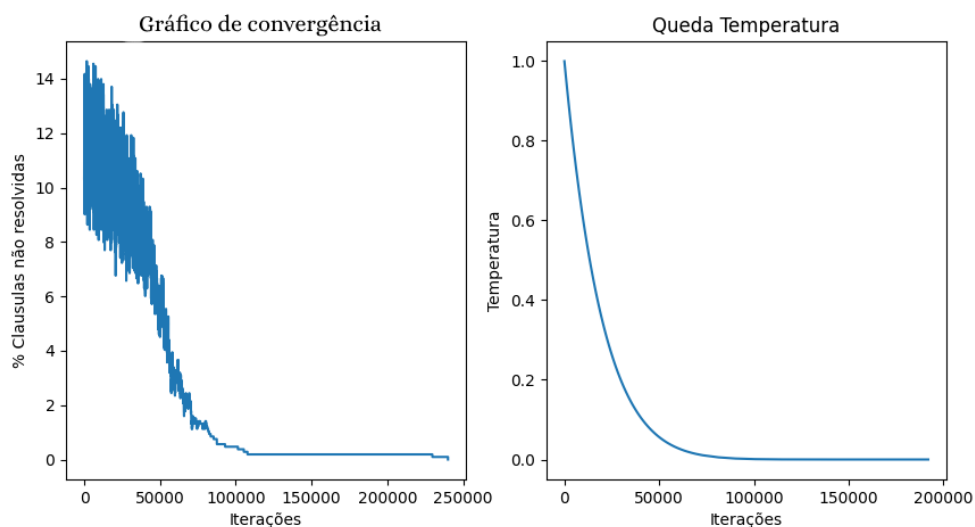
foram resolvidas 4 das 10 execuções, enquanto as outras resolveram cerca de 99% das clausulas



**Figure 6. BoxPlox das quantidades da clausulas nao resolvida de cada execucao SA**

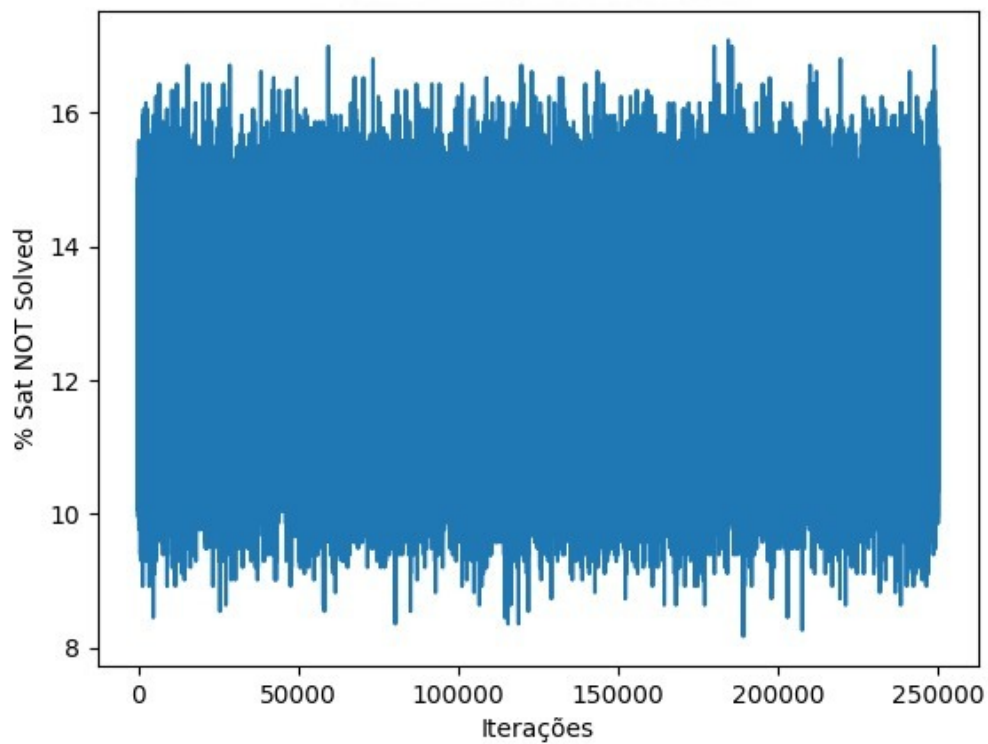
### 5.3. 250 variaveis

Por fim temos a instância de 250 variaveis, essas que foram um desafio maior até para o algoritmo SA, ja para o random search, novamente, nenhuma execucao teve um retorno positivo.

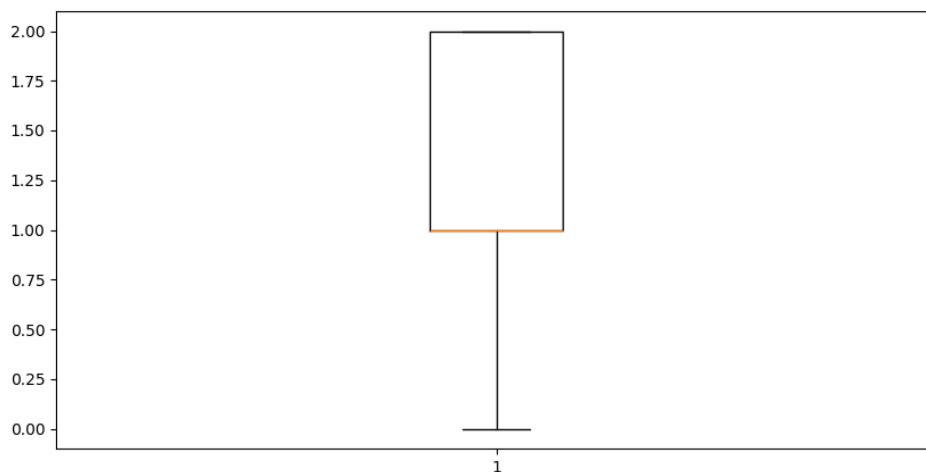


**Figure 7. Enter Caption**

Analisando o grafico de convergencia na figura7 podemos perceber como essa instância exigiu mais tempo do algoritmo SA, pois sua solução foi retornada somente proximo das 250000 iterações, que é o número maximo de iterações. É notável também o gráfico de queda de temperatura sendo executada com mais tempo, assim podemos ter mais noção do funcionamento da função de resfriamento, que leva em consideração a quantidade de iterações.



**Figure 8. Enter Caption**



**Figure 9. Enter Caption**

Já no algoritmo de random search na figura 8, percebemos que já não faz mais sentido utilizá-lo, pois a cada vez que aumentamos a quantidade de variáveis, a porcentagem que cláusulas não resolvidas aumenta, demonstrando a baixa eficiência do algoritmo nesses cenários e evidenciando o motivo de usarmos o SA.

Por fim, na figura 9 temos o boxplot da quantidade de clausulas nao resolvidas em todas as execuções do SA, para melhor analise.

#### 5.4. Média e desvio padrão

Para facilitar a análise dos resultados, foi gerado tabelas comparando o desvio padrão e média de cada um deles. Com isso, podemos analisar como nesses casos o algoritmo simulated annealing é superior em todas as instâncias.

Para o cálculo desses valores foi utilizado a quantidade de clausulas não resolvidas no final de cada execução, e foram 10 execuções para cada instância de problema.

| Número de Variáveis  | Média de clúsulas não resolvidas |                     |
|----------------------|----------------------------------|---------------------|
|                      | Random Search                    | Simulated Annealing |
| <b>20 Variáveis</b>  | 0.2                              | 0                   |
| <b>100 Variáveis</b> | 25.5                             | 1.0                 |
| <b>250 Variáveis</b> | 84.8                             | 1.2                 |

| Número de Variáveis  | Desvio padrão de clausulas não resolvidas |                     |
|----------------------|-------------------------------------------|---------------------|
|                      | Random Search                             | Simulated Annealing |
| <b>20 Variáveis</b>  | 0.42                                      | 0.0                 |
| <b>100 Variáveis</b> | 1.35                                      | 1.05                |
| <b>250 Variáveis</b> | 2.74                                      | 0.78                |

#### 6. Conclusão

Em síntese, considerando o desafio e objetivo proposto, podemos concluir que o algoritmo de SA é mais eficiente que o random search, pelo menos quando aplicado para a resolução do 3SAT. Porém, também é claro que existe a necessidade da execução do algoritmo diversas vezes para talvez conseguir uma solução completa. No caso das instâncias propostas todas tinham solução, assim como demonstrado, mas poderiam não possuir, e assim seria necessário executar diversas vezes e depois analisar a media e desvio padrão dela, tal como feito em ambas as tabelas feitas.

#### References