Análise de metaheurísticas para o problema de agrupamento multidimencional

Wanderson Ralph Silva Vita[1]

Abstract

Este artigo começa apresenta o resultado do treinamento das metaheurísticas Grasp, Simulated Annealing e Algoritimos Genéticos para o problema de agrupamento. Com o resultado dessas análises é escolhidos os melhores para etapa de teste.

Keywords: Metaheurísticas, Grasp, Simulated Annealing, Algoritimos Genéticos, Problema de Agupamento, Cluster, K-means

1. Introdução

Neste trabalho vamos avaliar a eficácia das metaheurísticas Grasp, Simulated Annealing e Algoritimos Genetico, para resolver o problema do agrupamento.

Primeiro será implementada uma estrutura para representar esse problema, e modelada para usar nos algorítimos citados. Logo após terá a etapa de treinamento, que com o método de grid search, será considerado diferentes combinações de hiperparâmetros para cada heurística. Depois é feito a normalização com zscore e o ranqueamento de cada configuração, afim de escolher os melhores para etapa de teste.

Na estapa de teste, as melhores configurações serão comparadas com algoritimo K-means, que é um algoritmo especializado nesse problema.

[1]Aluno de Engenharia de Computação da Universidade Federal do Espirito Santo

2. Descrição do Problema de K Médias

O problema de K médias tem o objetivo de agrupar diferentes pontos, a depender de suas características.

Dado n de pontos, como agrupa-los em K grupos distindo, de forma que a soma das distâncias euclidianas quadradas (SSE) entre os pontos de cada grupo, seja mínima?

3. Descrição dos Métodos Utilizados

3.1. Arquitetura

A implementação dos algoritimos, é baseada na arquitetura mostrada no diagrama de classes da Figura 1, onde cada classe de metaheurística deve implementar os métodos da classe abstrata 'Metaheuristica'. Esta por sua vez tem um atributo 'state' do tipo 'IState', uma interface que força que cada problema modelado para executar esses algoritimos tenha implementado os métodos básicos de um estado. Como o 'Value' que retorna o valor do estado, o 'Equal' que verifica se dois estados são iguais, o 'Compare' que compara dois estados a fim de classificalos, 'NextState' que forne um próximo estado basedo no atual e o 'Change' que permite modificar um indicice especifico de um estado. Esse conseito permite que a metaheurisitica implementada não precise ser alterada pra cada problema diferente que for aplicado, e sim a classe que representa o problema, que deve ser modelada para tal.

3.2. Representação do Descritor de Espaço de Estados

O problema modelado, que no caso é representado pela classe 'Cluster', possui uma matrix X_{Kxn} que guarda cada ponto do espaço n-dimencional dado como entrada, um array $grupos_{Kx1}$ para amarzenar o grupo de cada ponto, e os métodos 'CalcularCentroides' e 'SSE' que calcula respectivamente o centroide a a soma das distancia euclidiana quadrada para o estado atual do cluster.

Além desses, o 'Cluster' deve implementar a interface 'IState' para utilizar das metaheurísticas , e a 'IMutable' para ser utilizada no algoritimo genético.

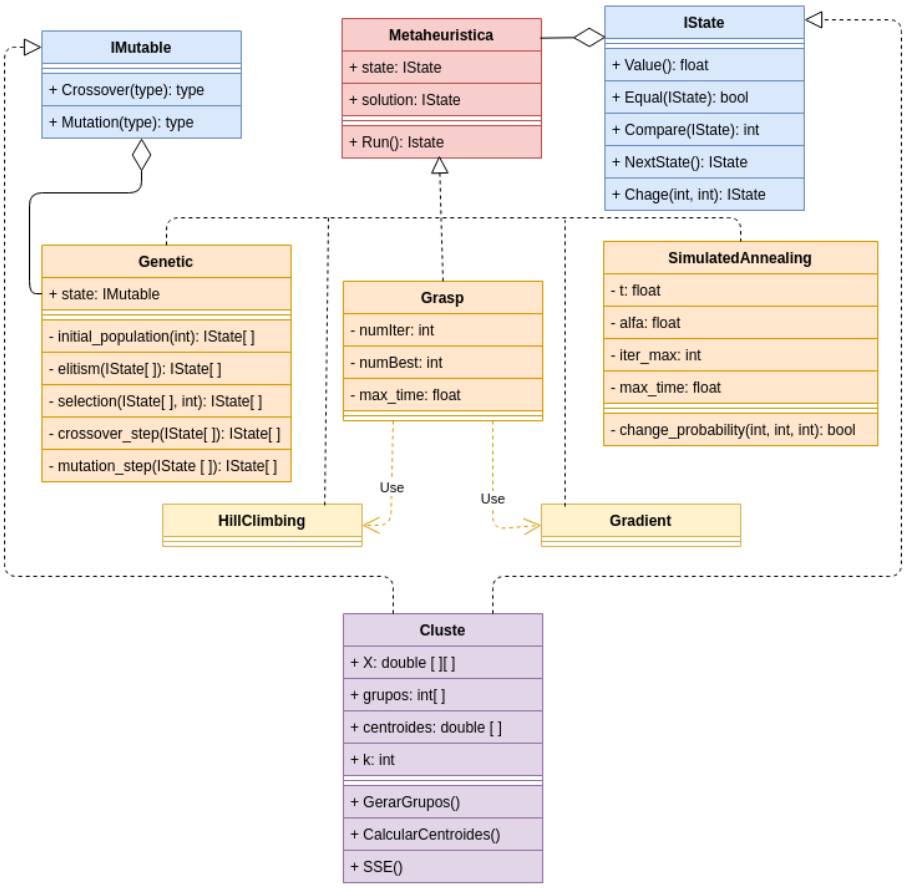

Figure 1: Diagrama de classes

3.3. GRASP

Este algoritimo consiste em combinar duas outras metaheurísticas, uma não deterministica a fim de fazer uma busca global, e outra deterministica para fazer uma busca local em torno do melhor resultado do passo anterior.

Para esse trabalho foram usados o Hill Climbin não deterministico e o Gradiente descentente, esses só depedem de um estado inicial para fazer a busca. O Grasp encerra quando termina o número máximo de iterações passado como parâmetro ou o tempo limite.

3.4. Simulated Annealing

Este algotimo tenta vencer o desafio dos algoritimos de busca, que podem ficar presos em um minimo local, não conseguindo achar o global. Quando para chegar neste, envolveria primeiro passar para um estado pior que o atual.

O procedimento começa com uma temperatura t que vai diminuindo pela taxa a $alfa$ cada iteração. Essa temperatura t é utlizada para calcular a probabilidade de escolher um estado pior para proseguir para as próximas iterações. Esta probabilidade tende a ser menor a medida que t diminui, fazendo com que as ultimas iterações tenda a não escolher um estado pior.

Esse efeito tem a vantagem de permitir sair de um minimo local, e seguir a busca pelo global.

3.5. Genetic Algorithm

É uma metaheuristica que se baseia na natureza para simular atificialmente um conceito de computação evolutiva. Onde o problema começa em um estado inicial donde é gerada uma população inicial. Dessa população é feita a seleção dos melhores, para o cruzamento entres os individuos e uma posterior possivel mutação destes novos a depender da taxa *mut_ratio*. Este procedimento é repetido a cada nova geração de população até que haja a convergengia e todos sejam iguais, ou atinja um tempo limite.

Na implementação deste também foi utilizado o conceito de eletismo, onde uma porcentagem dos melhores são escolhidos para pular as etapas de seleção, cruzamento e mutação e ir direto para próxima geração. Num primeiro momento foi adotada uma taxa de 20% de elitismo, mas causava a convergência muito rápido, então usei uma taxa de 5%.

4. Descrição dos Resultados dos Experimentos

4.1. Treino

4.1.1. Apresentação de tabela com os valores dos hiperparâmetros das cinco melhores configuração de cada método

Meta-Heurística	Configuração
Grasp	numIter: 20, numBest: 5
	numIter: 20, numBest: 15
	numIter: 20, numBest: 10
	numIter: 50, numBest: 5
	numIter: 100, numBest: 15
Simulated Annealing	t: 100, alfa: 0.85, iter_max: 500
	t: 50, alfa: 0.95, iter_max: 500
	t: 100, alfa: 0.85, iter_max: 350
	t: 500, alfa: 0.85, iter_max: 500
	t: 500, alfa: 0.95, iter_max: 500
Genetic Algorithm	pop_size: 10, cross_ratio: 0.75, mut_ratio: 0.2
	pop_size: 10, cross_ratio: 0.85, mut_ratio: 0.2
	pop_size: 10, cross_ratio: 0.95, mut_ratio: 0.2
	pop_size: 10, cross_ratio: 0.75, mut_ratio: 0.1
	pop_size: 10, cross_ratio: 0.85, mut_ratio: 0.1

4.1.2. Apresentação dos bloxplots de média e de tempo de cada método

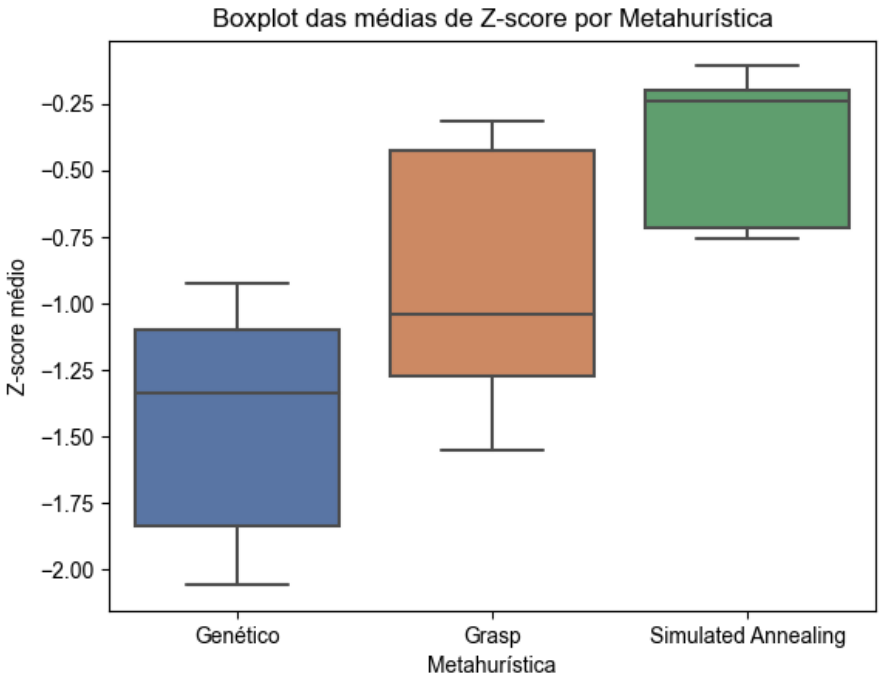

Figure 2: Boxplot das médias de de Z-score por Metahurística

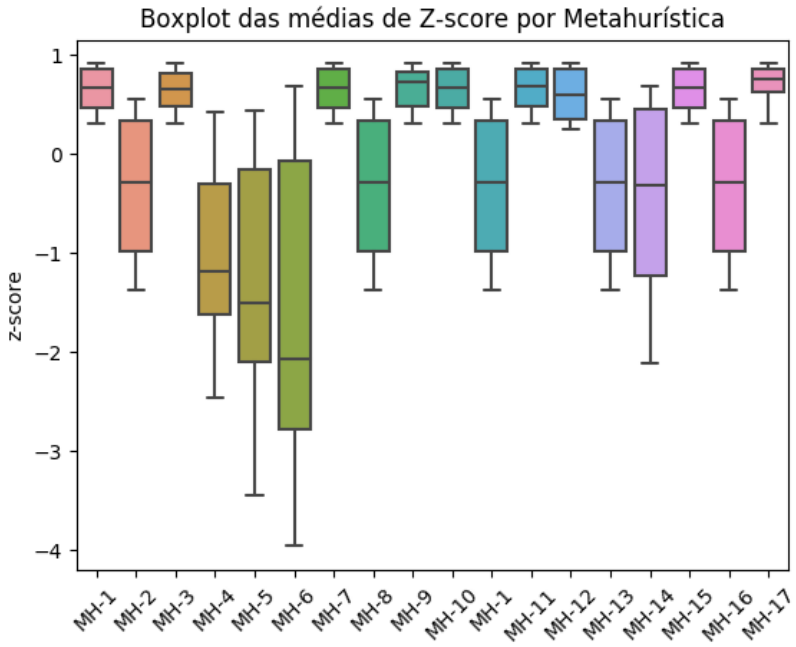

Figure 3: Genético - Boxplot das médias de de Z-score por Configuração

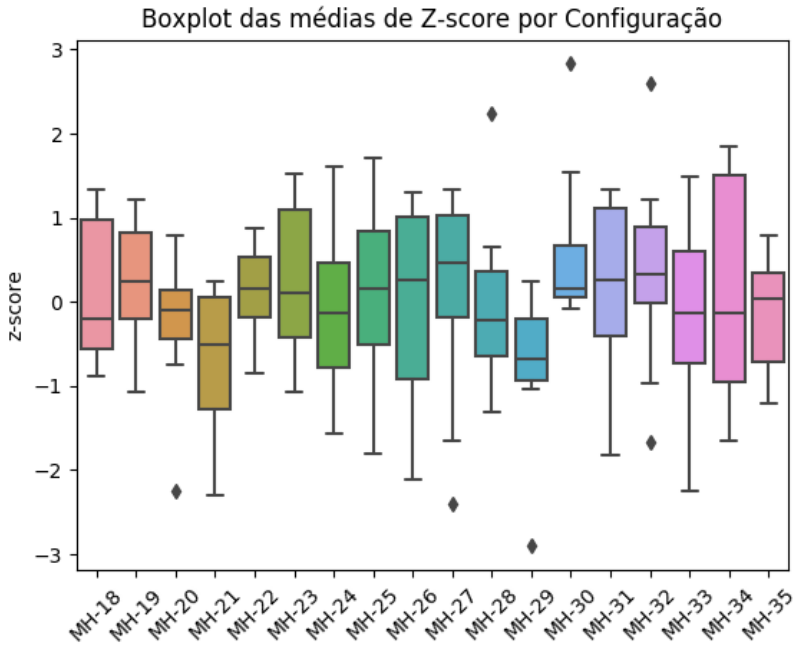

Figure 4: Grasp - Boxplot das médias de de Z-score por Configuração

A Figura 4 nos mostra que o Algoritimo Genético em média obteve resultados melhores que Grasp, que por sua vez se saiu melhor que o Simulated Annealing.

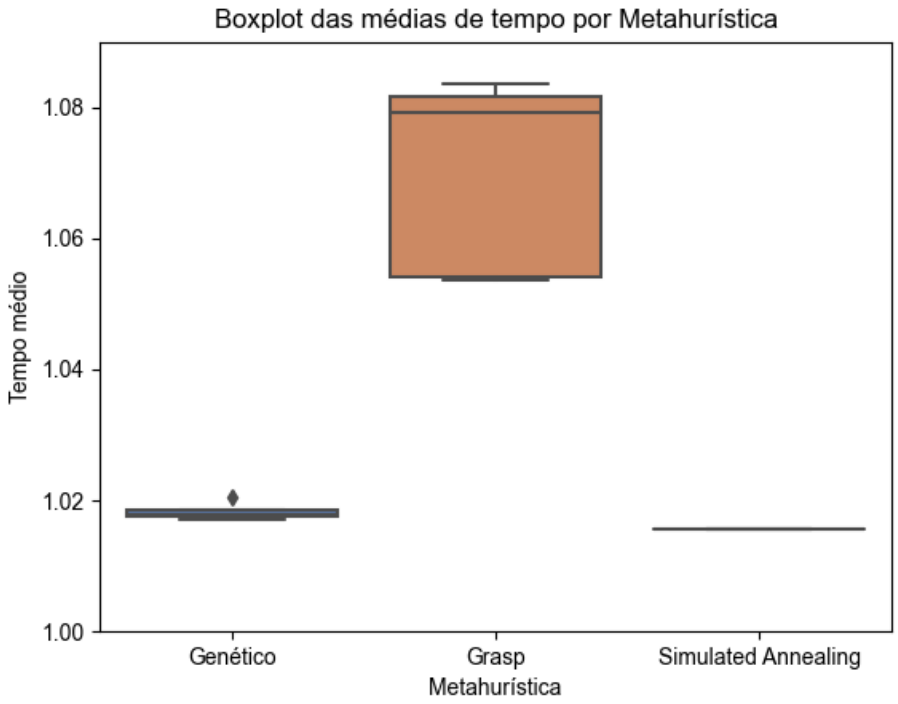

Figure 5: Boxplot das médias de tempo por Metahurística

Podemos ver nos boxplots da Figura 7, que o Grasp ultrapassar o tempo maximo bem mais que os outros métodos, ele teve uma grande variação de tempo. Isso é devido ele chamar outros dois algoritimos de busca. Se o tempo vencer quando estiver dentro de algum desses, e até sair dos dois para testar o tempo total, vai passar levemente do tempo máximo.

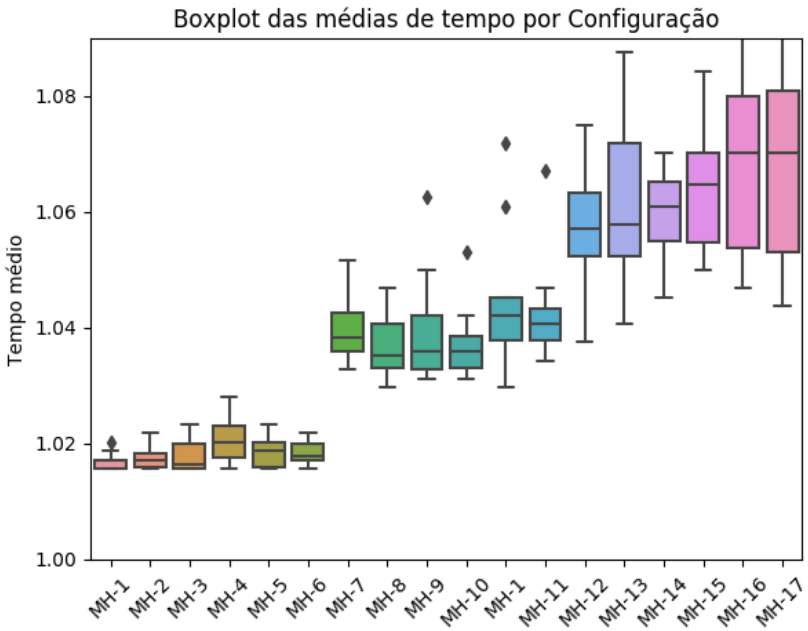

Figure 6: Genético Boxplot das médias de tempo por Configuração

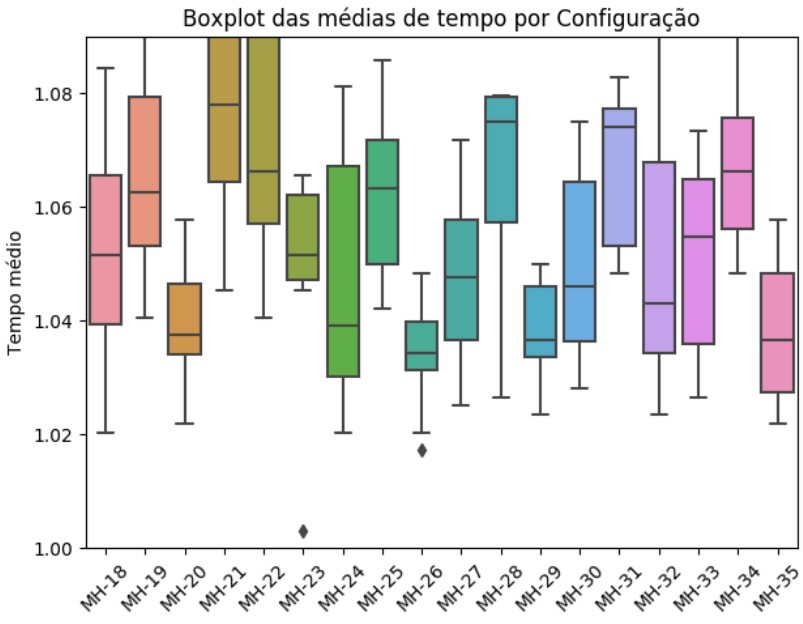

Figure 7: Grasp - Boxplot das médias de tempo por Configuração

4.2. Apresentação de tabela com ranqueamento em cada problema e ranqueamento médio de cada método

Configuração			K									
			Iris					Wine				
pop_size	cross_ratio	mut$_r atio$	3	7	10	13	22	2	6	9	11	33
10	0.75	0.1	5.0	3.0	5.0	4.0	6.0	3.0	4.0	2.0	4.0	3.0
	0.75	0.2	2.0	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	2.0
	0.85	0.1	4.0	5.0	4.0	3.0	4.0	5.0	5.0	5.0	5.0	6.0
	0.85	0.2	1.0	2.0	2.0	2.0	2.0	1.0	2.0	4.0	2.0	1.0
	0.95	0.1	6.0	6.0	6.0	6.0	5.0	6.0	6.0	6.0	6.0	5.0
	0.95	0.2	3.0	4.0	3.0	5.0	3.0	4.0	3.0	3.0	3.0	4.0
30	0.75	0.1	12.0	12.0	18.0	16.0	18.0	12.0	9.0	10.0	15.0	10.0
	0.75	0.2	10.0	10.0	8.0	8.0	7.0	7.0	8.0	7.0	8.0	7.0
	0.85	0.1	15.0	8.0	16.0	7.0	9.0	9.0	11.0	14.0	12.0	12.0
	0.85	0.2	7.0	11.0	10.0	11.0	16.0	10.0	7.0	8.0	14.0	8.0
	0.95	0.1	13.0	16.0	7.0	14.0	11.0	13.0	16.0	15.0	17.0	9.0
	0.95	0.2	9.0	7.0	13.0	10.0	8.0	8.0	10.0	9.0	7.0	15.0
50	0.75	0.1	14.0	15.0	12.0	9.0	14.0	16.0	12.0	13.0	9.0	17.0
	0.75	0.2	8.0	14.0	9.0	12.0	17.0	18.0	15.0	12.0	10.0	16.0
	0.85	0.1	17.0	18.0	17.0	18.0	15.0	14.0	17.0	17.0	18.0	11.0
	0.85	0.2	11.0	13.0	15.0	17.0	10.0	15.0	13.0	18.0	11.0	13.0
	0.95	0.1	18.0	17.0	14.0	13.0	13.0	17.0	18.0	11.0	16.0	14.0
	0.95	0.2	16.0	9.0	11.0	15.0	12.0	11.0	14.0	16.0	13.0	18.0

Table 1: Ranqueamento de cada configuração do **Algoritimo Gnético** por Base e Número de grupos K

Configuração			K									
			Iris					Wine				
t	alfa	iter_max	3	7	10	13	22	2	6	9	11	33
100	0.7,	350,	6.0	12.0	17.0	3.0	16.0	6.0	7.0	7.0	5.0	16.0
	0.7,	500,	16.0	3.0	15.0	16.0	7.0	5.0	11.0	10.0	14.0	5.0
	0.85	350	4.0	9.0	11.0	14.0	9.0	8.0	5.0	5.0	1.0	12.0
	0.85	500	7.0	1.0	1.0	1.0	4.0	12.0	10.0	3.0	8.0	10.0
	0.95	350	3.0	15.0	9.0	5.0	10.0	13.0	9.0	14.0	11.0	8.0
	0.95	500	8.0	18.0	14.0	8.0	6.0	16.0	2.0	4.0	18.0	14.0
50	0.7	350	9.0	7.0	8.0	17.0	3.0	18.0	13.0	8.0	2.0	6.0
	0.7	500	14.0	14.0	2.0	10.0	18.0	2.0	8.0	6.0	13.0	15.0
	0.85	350	15.0	11.0	16.0	13.0	5.0	17.0	3.0	1.0	16.0	3.0
	0.85	500	11.0	16.0	6.0	15.0	12.0	1.0	17.0	16.0	12.0	1.0
	0.95	350	5.0	13.0	7.0	12.0	14.0	11.0	4.0	18.0	7.0	2.0
	0.95	500	1.0	4.0	3.0	6.0	11.0	7.0	6.0	11.0	4.0	7.0
500	0.7	350	18.0	10.0	13.0	18.0	8.0	10.0	15.0	9.0	9.0	9.0
	0.7	500	12.0	6.0	10.0	4.0	1.0	15.0	16.0	17.0	17.0	11.0
	0.85	350	10.0	8.0	18.0	2.0	2.0	9.0	12.0	15.0	15.0	17.0
	0.85	500	17.0	2.0	5.0	7.0	15.0	4.0	1.0	13.0	10.0	13.0
	0.95	350	2.0	17.0	12.0	9.0	17.0	3.0	18.0	2.0	3.0	18.0
	0.95	500	13.0	5.0	4.0	11.0	13.0	14.0	14.0	12.0	6.0	4.0

Table 2: Ranqueamento de cada configuração do **Simulated Annealing** por Base e Número de grupos K

Configuração		K									
		Iris					Wine				
numIter	numBest	3	7	10	13	22	2	6	9	11	33
100	10	15.0	13.0	9.5	15.0	11.5	13.5	14.0	14.0	14.0	10.5
	15	4.0	13.0	9.5	7.0	11.5	5.0	4.0	6.5	3.5	10.5
	5	11.0	8.0	17.0	10.0	11.5	13.5	14.0	14.0	14.0	10.5
20	10	7.0	3.0	2.0	3.5	3.0	2.0	8.0	2.0	9.0	2.0
	15	1.0	2.0	4.0	2.0	4.0	1.0	1.0	6.5	3.5	10.5
	5	8.5	1.0	1.0	1.0	1.0	13.5	7.0	1.0	7.0	1.0
200	10	15.0	13.0	9.5	15.0	11.5	13.5	14.0	14.0	14.0	10.5
	15	4.0	13.0	9.5	7.0	11.5	5.0	4.0	6.5	3.5	10.5
	5	15.0	6.0	17.0	11.0	11.5	13.5	14.0	14.0	14.0	10.5
350	10	15.0	13.0	9.5	15.0	11.5	13.5	14.0	14.0	14.0	10.5
	15	4.0	13.0	9.5	7.0	11.5	5.0	4.0	6.5	3.5	10.5
	5	10.0	7.0	15.0	15.0	11.5	13.5	14.0	14.0	14.0	10.5
50	10	15.0	5.0	9.5	15.0	11.5	8.0	14.0	14.0	14.0	10.5
	15	4.0	13.0	9.5	7.0	11.5	5.0	4.0	6.5	3.5	10.5
	5	8.5	4.0	3.0	3.5	2.0	13.5	9.0	3.0	8.0	10.5
500	10	15.0	13.0	9.5	15.0	11.5	13.5	14.0	14.0	14.0	10.5
	15	4.0	13.0	9.5	7.0	11.5	5.0	4.0	6.5	3.5	10.5
	5	15.0	18.0	17.0	15.0	11.5	13.5	14.0	14.0	14.0	10.5

Table 3: Ranqueamento de cada configuração do **Grasp** por Base e Número de grupos K

4.3. Apresentação de tabela com melhor configuração de cada método por média e por ranqueamento médio

Metahurística	**Configuração**	**Z-score médio**	**Rank médio**
Genético	pop_size: 10, cross_ratio: 0.75 mut_ratio: 0.2	-2.058675	1.3
Grasp	numIter: 20, numBest: 5	-1.551241	4.2
	numIter: 20, numBest: 15	-1.272055	3.55
Simulated Annealing	t: 100, alfa: 0.85, iter_max: 500	-0.758096	5.7

Table 4: Melhor configuração por método

As melhores configurações tanto do Algoritimos genético quanto Simulated Annealing tiveram a melhor média de z-score e o melhor ranqueamento. Já o Grasp teve duas configurações elegíveis.

4.4. Análise dos resultados alcançados

4.5. Teste

4.5.1. Apresentação da tabela contendo média padronizada, desvio padrão, média e desvio padrão dos tempos de execução de todas os métodos testados

Metahurística	**Configuração**	μ**(Z-score)**	σ**(Z-score)**	μ**(Tempo)**	σ**(Tempo)**
Genético	pop_size: 10, cross_ratio: 0.75 mut_ratio: 0.2	0.393045	0.374154	1.023802	0.006660
Grasp	numIter: 20, numBest: 5	0.532376	0.037040	1.119635	0.094369
	numIter: 20, numBest: 15	0.551031	0.126695	1.153099	0.118255
Simulated A.	t: 100, alfa: 0.85, iter_max: 500	0.454563	0.096084	1.015625	0.000000
K-means		-1.997601	0.002598	0.15713	0.136363

Table 5: Médias e desvios padrão por método

4.5.2. Apresentação dos bloxplots de média e de tempo de cada método

Podemos ver pelos boxplot da Figura 8 que o K-means tem a média de z-score bem abaixo dos outros métodos. Mas o interessante é que na etapa de teste o Simulate Annealing melhorou em relação ao Grasp, se comparado com a etapa de treinamento.

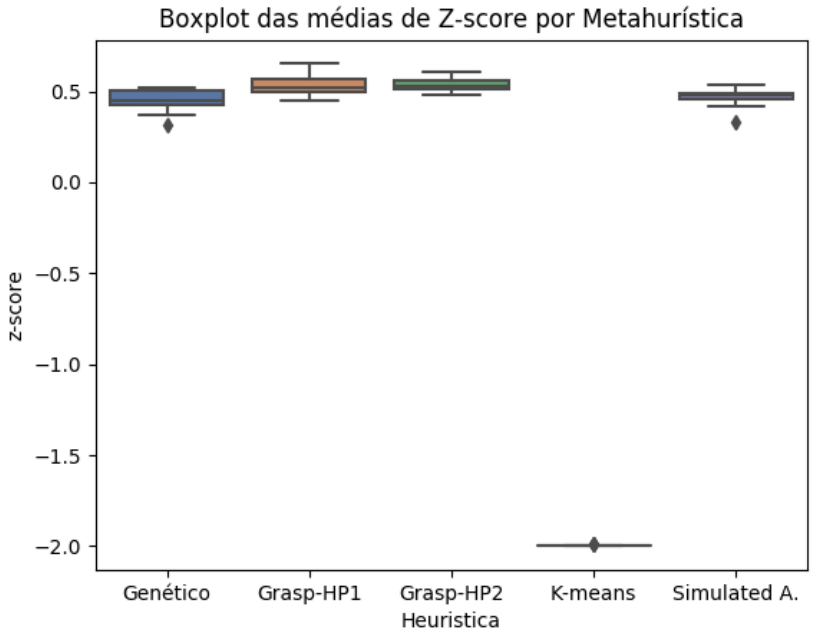

Figure 8: Boxplot das médias de Z-score por Metahurística

A média de tempo que o K-means gasta é 5 vezes menor que dos outros métodos, como visto na Figura 9, e ainda encontra soluções melhores. Mas isso se deve ao fato do K-means ser um algoritimo especifico para esse tipo de problema, enquanto os outros métodos podem ser usados em vários problemas.

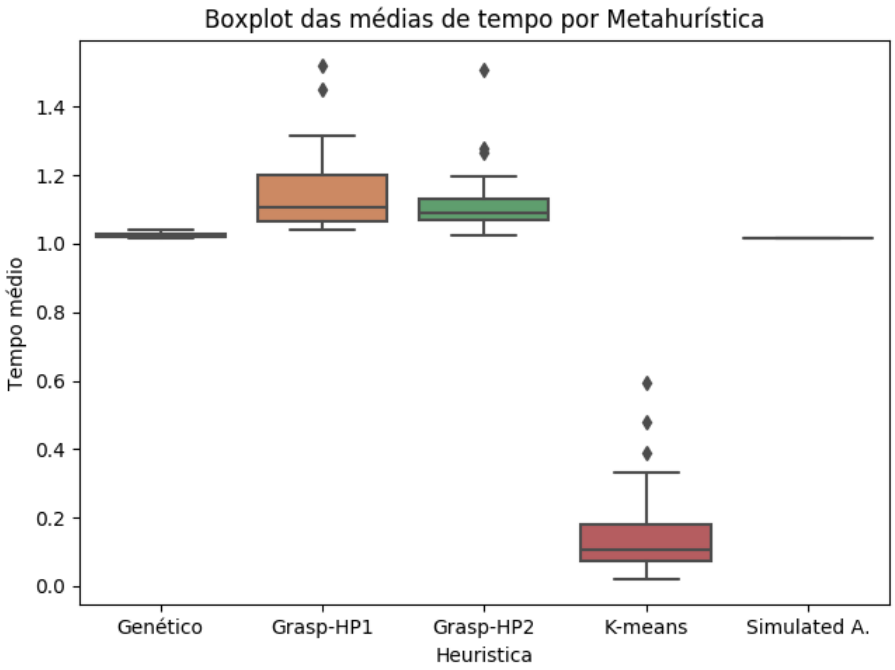

Figure 9: Boxplot das médias de tempo por Metahurística

4.5.3. Apresentação de tabela contendo os ranqueamentos das métodos para cada problema de teste e o ranqueamento médio

Sendo **Grasp-HP1** com os hiperparâmetros {'numIter': 20, 'numBest': 5} e **Grasp-HP2** {'numIter': 20, 'numBest': 15}, temos a seguinte tabela de ranqueamento por problema para cada metaheurística.

Problema/Heuristica	Simulated A.	Genético	Grasp-HP1	Grasp-HP2	K-means
Ionosphere-10	2.0	3.0	4.0	5.0	1.0
Ionosphere-15	3.0	4.0	5.0	2.0	1.0
Ionosphere-2	3.0	2.0	4.0	5.0	1.0
Ionosphere-20	2.0	4.0	5.0	3.0	1.0
Ionosphere-25	3.0	4.0	5.0	2.0	1.0
Ionosphere-3	2.0	4.0	5.0	3.0	1.0
Ionosphere-30	3.0	4.0	5.0	2.0	1.0
Ionosphere-40	2.0	4.0	5.0	3.0	1.0
Ionosphere-5	2.0	3.0	4.0	5.0	1.0
Ionosphere-50	2.0	4.0	5.0	3.0	1.0
Iris-11	2.0	3.0	5.0	4.0	1.0
Iris-15	2.0	4.0	5.0	3.0	1.0
Iris-17	3.0	2.0	4.0	5.0	1.0
Iris-2	3.0	2.0	5.0	4.0	1.0
Iris-23	4.0	2.0	5.0	3.0	1.0
Iris-28	4.0	2.0	3.0	5.0	1.0
Iris-32	3.0	2.0	5.0	4.0	1.0
Iris-4	3.0	2.0	4.0	5.0	1.0
Iris-50	2.0	3.0	4.0	5.0	1.0
Iris-8	4.0	2.0	5.0	3.0	1.0
Wine-13	3.0	2.0	4.0	5.0	1.0
Wine-15	3.0	2.0	4.0	5.0	1.0
Wine-20	4.0	3.0	5.0	2.0	1.0
Wine-23	3.0	2.0	4.0	5.0	1.0
Wine-25	2.0	3.0	4.0	5.0	1.0
Wine-3	3.0	2.0	4.0	5.0	1.0
Wine-30	3.0	2.0	4.0	5.0	1.0
Wine-41	2.0	3.0	5.0	4.0	1.0
Wine-45	2.0	3.0	5.0	4.0	1.0
Wine-5	3.0	2.0	5.0	4.0	1.0

Table 6: Ranqueamentos das métodos para cada problema

4.5.4. Apresentar tabela pareada de testes estatíticos com destaque nos pares de métodos onde houve diferença significativas

4.5.5. Apresentar melhor método geral por média e ranqueamento médio

Como visto na Tabela 7, o melhor método tanto pelo quesito da média de z-score, quando pelo ranqueamento médio, é o Kmeans.

Metahurística	**Configuração**	**Z-score médio**	**Rank médio**
K-means		-1.997601	1.000000
Genético	pop_size: 10, cross_ratio: 0.75 mut_ratio: 0.2	0.393045	2.800000
Simulated Annealing	t: 100, alfa: 0.85, iter_max: 500	0.454563	2.733333
Grasp	numIter: 20, numBest: 5	0.532376	4.533333
	numIter: 20, numBest: 15	0.551031	3.933333

Table 7: Z-score médio e Rank médio por método

4.5.6. Análise dos resultados alcançados

5. Conclusões

5.1. Análise geral dos resultados

5.2. Contribuições do Trabalho

5.3. Melhorias e trabalhos futuros

References