Introdução à Inteligência Artificial

Trabalho Prático 2 - Problema de Otimização

Frederico Quelhas – 2022135081

Luccas Galvan – 2022100410

1 Introdução

Este trabalho consiste em conceber, implementar e testar métodos de otimização que encontrem soluções de boa qualidade para diferentes instâncias do problema a seguir descrito. Neste problema pretendemos obter soluções que minimizem o custo. Estas soluções consistem em maneiras diferentes de encontrar, implementar e testar métodos de otimização que encontrem soluções de boa qualidade para diferentes instâncias do problema Minimum Cost Subset Problem.

Temos como dados de entrada uma variável 'k', que representa o tamanho do subconjunto de vértices, uma variável 'p' que representa o número de vértices e uma variável 'e' que contém as ligações e o seu custo.

2 Descrição da função de avaliação e objetivo da otimização

O Minimum Cost Subset Problem consiste em encontrar um subconjunto de k-vértices tal que o custo das arestas dentro do subconjunto seja mínimo, com todos os vértices do subconjunto com pelo menos uma ligação.

Pretende-se que implemente e avalie a capacidade de diferentes algoritmos de otimização para encontrar soluções de boa qualidade para o problema descrito.

Descrição dos algoritmos e/ou das heurísticas utilizadas

3 Algoritmo de Pesquisa Local (Trepa-Colinas)

O algoritmo de pesquisa local "trepa-colinas" é uma técnica heurística que visa encontrar uma solução ótima local para um problema. A estratégia do trepa-colinas é bastante simples e dividida nos seguintes passos: Inicialização, que começa (neste caso) com uma solução pré-determinada; Geração de vizinhança, que examina as soluções à vizinhança da solução atual; Avaliação, que calcula o valor da função objetivo para cada solução na vizinhança; Movimento, que move-se para a solução vizinha com o valor (neste caso, como o problema é de minimização) mais baixo; E, por último, a Convergência, que repete os passos anteriores até que (se existir) o número de repetições seja excedido ou uma solução satisfatória seja encontrada.

Para além disso, escolhemos utilizar o algoritmo trepa-colinas não apenas por ser relativamente mais simples de implementar e compreender, mas também pela sua eficiência que nos permite analisar em uma profundeza maior o problema para conseguir as melhores soluções. O algoritmo de recristalização simulada não foi considerado pelo nosso grupo, por ser uma técnica mais global, explorando soluções fora da vizinhança imediata. Em contraste, o trepa-colinas é mais local, concentrando-se na melhoria incremental.

4 Algoritmo Evolutivo

É uma busca por uma solução ótima para o problema em questão. O processo de busca é levado pela codificação das soluções do problema como um cromossoma, função para avaliar a capacidade (fitness), inicialização da população inicial, operadores de seleção e operadores de reprodução.

5 Algoritmo Híbrido

É uma combinação do algoritmo de pesquisa local e o evolutivo.

O objetivo é o algoritmo evolutivo tratar de encontrar o melhor caminho com a melhor qualidade, e a seguir o algoritmo local desenvolve o caminho e acaba por encontrar a melhor solução.

6 Métodos de seleção e operadores genéticos implementados

6.1 crossover(parents, d, offspring)

6.2 recombinacao_uniforme(parents, d, offspring);

6.3 mutation(offspring, d);

6.4 mutacao_por_troca(offspring, d);

São algoritmos de otimização global, baseados nos mecanismos de seleção natural e da genética. Eles empregam uma estratégia de busca paralela e estruturada, mas aleatória, que é voltada em direção ao reforço da busca de pontos de "alta capacidade", ou seja, pontos nos quais a função a ser minimizada (ou maximizada) tem valores relativamente baixos (ou altos).

Apesar de aleatórios, eles não dependem caminhadas aleatórias não direcionadas, pois exploram informações para encontrar novos pontos de busca onde são esperados melhores desempenhos. Isto é feito através de processos iterativos, onde cada iteração é representada como uma geração.

Em cada geração, os princípios de seleção e reprodução são aplicados a uma população de candidatos que pode variar, dependendo da complexidade do problema e dos recursos computacionais disponíveis. Através da seleção, se determina quais indivíduos conseguirão se reproduzir, gerando um número de descendentes para a próxima geração, com uma probabilidade determinada pelo seu índice de aptidão. Ou seja, os indivíduos com maior adaptação relativa têm maiores chances de se reproduzir.

7 Resultados dos testes efetuados

Algoritmo Trepa Colinas

Trepa-Colinas com Vizinhança 1 e aceitando soluções de custo igual

NUM VERT		1000 it	10000 it	25000it	50000it
file1.txt	Melhor	45	45	46	45
	MBF	57.23	56.87	58.67	56.34
file2.txt	Melhor	14	11	9	10
	MBF	25.6	23.3	27.23	22.6
file3.txt	Melhor	340	336	340	338
	MBF	382.03	371.37	383.34	397.37
file4.txt	Melhor	16	7	10	4999
	MBF	1688.13	10.6	1782.87	4999
file5.txt	Melhor	119	4999	4999	121
	MBF	4836.33	4999	4999	4933.33

Algoritmo Trepa Colinas Probabilístico

Trepa-Colinas probabilístico, com vizinhança 1 e aceitando soluções de custo igual

NUM VERT		100000 iterações Prob = 0.01	200000 iterações Prob = 0.01	100000 iterações Prob = 0.0005	200000 iterações Prob = 0.0005
file1.txt	Melhor	352	363	390	408.27
	MBF	393.1	395.17	411.73	324
file2.txt	Melhor	543	360	5207	5218
	MBF	5138.766667	4975.7	5409.17	5398.33
file3.txt	Melhor	962.0	1008.0	1045.0	1037.0
	MBF	1047.6	1064.2	1097.6	1100.3
file4.txt	Melhor	5221.0	5011.0	5505.0	5532.0
	MBF	5470.5	5416.7	5748.8	5784.2
file5.txt	Melhor	5051	5232	6218	7219
	MBF	5887	5989.73	7022.13	7622.67

Algoritmo Evolutivo

	File1.txt	Algoritmo base (Recombinação de 1 ponto de corte + Mutação binária + Penalização cega)		Recombinação de 1 ponto de corte + Mutação por troca + Reparação2		com recombinação uniforme + mutação binária + reparação		com recombinação uniforme + mutação por troca + reparação	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF	Best	MBF	Best	MBF
pop = 100 (ger = 2500)	pr = 0.3	45.0	48.50	45.0	54.65	45.0	48.93	45.0	54.00
pm = 0.01	pr = 0.5	45.0	47.83	45.0	54.23	45.0	48.90	45.0	51.33
tsize = 2	pr = 0.7	45.0	49.93	45.0	51.57	45.0	48.30	45.0	51.93
pop = 100 (ger = 2500)	pm = 0.0	45.0	57.00	45.0	55.00	45.0	50.00	45.0	51.43
pop = 100	pm = 0.001	45.0	52.57	45.0	52.63	45.0	49.40	45.0	51.10
pr = 0.7	pm = 0.01	45.0	50.23	45.0	52.33	45.0	48.40	45.0	49.53
tsize = 2	pm = 0.05	45.0	45.70	45.0	52.83	45.0	45.40	45.0	51.23
pr = 0.7	pop = 10 (ger = 25K)	45.0	45.50	48.0	58.17	45.0	45.83	45.0	56.77
pm = melhor valor obtido	pop = 50 (ger = 5K)	45.0	45.50	45.0	55.60	45.0	45.50	45.0	52.87
tsize = 2	pop = 100 (ger = 2.5K)	45.0	45.57	45.0	52.93	45.0	45.10	45.0	48.80

		Algoritmo base		com mutação por troca + reparação2		uniforme + mutação		uniforme + mutação por	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF	Best	MBF	Best	MBF
pop = 100 (gen = 2500)	tsize = 3	45.0	52.53	45.0	53.33	45.0	45.50	45.0	50.60
pr = 0.7	tsize = 10	45.0	50.83	45.0	52.50	45.0	45.70	45.0	50.97
pm = 0.001	tsize = 50	45.0	50.23	45.0	50.20	45.0	45.40	45.0	49.77

Algoritmo Híbrido

	File1.txt	Algoritmo base híbrido i)		Algoritmo base híbrido ii)		Algoritmo base híbrido iii)	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF	Best	MBF
pop = 100 (gen = 2.5k)	PROBGERAVIZ = 1	45.0	46.83	45.0	46.303	45.0	46.50
pr = 0.7							
pm = 0.001	PROBGERAVIZ = 0.8	45.0	45.30	45.0	45.29	45.0	45.26
tsize = 2							

8 Conclusão

Os Algoritmos de Pesquisa Local conseguem eventualmente chegar a resultados ótimos em problemas simples, porém com o aumento do grau de complexidade, fica difícil definir com precisão qual direção tomar na escolha do melhor valor.

Os Algoritmos que envolvem abordagens Evolutivas já conseguem pegar em problemas mais complexos e dar resultados mais próximos, mesmo assim estando bastante longe de dar resultados confiantes. O Algoritmo Híbrido não foi tão útil como esperávamos, em teoria seria melhor mas muito frequentemente era inferior ao Algoritmo Evolutivo.