Universidade Federal da Paraíba
Centro de Informática
Análise e Projeto de Algoritmos

Projeto Final

Relatório dos Resultados

Grupo:
André Iarley Soares da Cruz — Mat. 20230145258
Diogo Cantuaria Da Silva Hiebert — Mat. 20230144813
Luiz Paulo de Souza Fontes Junior — Mat. 20230146291

Professor:
Bruno Petrato Bruck

17 de setembro de 2025

Tabela dos resultados

Instância	ótimo	Heurística construtiva			VND		
		valor da solução	tempo (ms)	GAP	valor da solução	tempo	GAP
n12_q20	15700	44100	0.081858	181%	28400	0.2391	80.9%
n12_q30	14600	44100	0.066941	202%	28400	0.189297	94.5%
n13_q30	16900	47600	0.097488	182%	42200	0.219304	149.7%
n14_q12	13500	17100	0.061816	26.6%	15600	0.322652	15.5%
n14_q30	12600	14900	0.070072	18.2%	13700	0.152879	8.7%
n17_q10	31443	69022	0.100315	119.5%	38823	0.710559	23.5%
n17_q20	29259	69022	0.089735	135.9%	38823	0.448789	32.7%
n20_q20	91619	116843	0.139644	27.5%	107739	1.20405	17.6%
n20_q30	76999	87670	0.102941	13.8%	79239	3.42771	2.9%
n26_q20	31100	67700	0.09872	117.6%	53700	0.593925	72.6%
n26_q30	30300	67700	0.078737	123.4%	48400	0.742623	59.7%
n40_q20	59493	175332	0.1681	194.7%	132526	1.00145	122.8%
n40_q30	57476	175332	0.137753	205.0%	132062	1.0881	129.8%
n54_q30	120277	550598	0.195839	357.7%	405528	3.38977	237.162%
n58_q30	65669	123550	0.483725	88.1%	95847	4.9898	45.9%
n74_q20	48829	105207	0.579534	115.4%	66482	9.92584	36.1%
n79_q30	39979	54718	0.458655	36.8%	52484	4.66258	31.2%
n81_q10	388680	663158	0.609469	70.6%	628189	9.40112	61.6%
n115_q20	157115	692103	0.93524	340.5%	381962	36.1842	143.1%

Instruções de Compilação

Na raiz do projeto execute o comando no terminal:

make

Instruções de Execução

Para executar uma instância basta passar o caminho como argumento no terminal:

./build/prog <caminho_do_arquivo>

Para executar todas as instâncias e calcular o gap bas executar sem argumentos:

./build/prog

Organização do projeto

./input - Instâncias de teste.

./output - saída das instâncias de teste.

./include - Arquivos de cabeçalho.

./src - Implementações do código.

Principais Arquivos:

Algoritmo guloso:

LowerCost

A classe *LowerCost* implementa um algoritmo guloso do **vizinho mais próximo**, utilizando uma **matriz de adjacência** ordenada para selecionar a próxima estação com o menor custo que ainda não tenha sido visitada, garantindo que a capacidade do caminhão não seja ultrapassada e que a carga não se torne negativa.

RouteStep

A classe *RouteStep* representa um movimento na rota construída, armazenando as informações relevantes de um trecho entre duas estações, como o ID da estação de origem, o ID da estação de destino, a carga, o custo e o custo acumulado até aquele ponto.

VND:

./src/VND.cpp - A lógica de execução do VND utilizando os algoritmos de vizinhança descritos a seguir.

Algoritmos de vizinhança:

2-Opt - ./src/TwoOpt.cpp - Também implementado conforme demonstrado na vídeo aula. Realiza as trocas em cada rota.

ReInsertion - ./src/ReInsertion.cpp - Implementado como mostrado ao final da vídeo aula. Atua a partir do reposicionamento do nó na rota encontrada.

Swap2 - *./src/swap2.cpp* - Implementado conforme demonstrado na vídeo aula, com a modificação que permite não apenas trocar nós dentro de uma mesma rota, mas também realizar permutações entre diferentes rotas.