Trabalho 2

Aluno 1 — cc24310@g.unicamp.br

Aluno 2 — cc24337@g.unicamp.br

Introdução

Este relatório explora a aplicação de **metaheurísticas** para resolver o desafiador **Problema do Caixeiro Viajante com Coleta e Entrega (TSPPD)**. Nosso objetivo é determinar a rota mais eficiente que visita múltiplos pontos, garantindo que cada item seja coletado antes de sua respectiva entrega e que cada local seja visitado apenas uma vez. A complexidade inerente do TSPPD, especialmente para instâncias de maior escala, exige a utilização de métodos que encontrem soluções de alta qualidade em tempo computacional razoável.

Para enfrentar essa complexidade, foram implementadas e analisadas duas abordagens distintas: o **Simulated Annealing (SA)** e o **Biased Random-Key Genetic Algorithm (BRKGA)**. O SA, inspirado no processo de recozimento de metais, explora o espaço de soluções aceitando, probabilisticamente, movimentos para estados de maior custo para evitar ótimos locais. Por outro lado, o BRKGA, baseado em princípios da evolução natural, utiliza vetores de chaves aleatórias para representar as soluções, que são então decodificadas em rotas válidas, evoluindo a população através de

seleção, recombinação e mutação.

Este documento apresentará as implementações detalhadas de ambos os algoritmos e sua capacidade de lidar com as restrições de precedência do TSPPD. A performance e a qualidade das soluções geradas serão comparadas e discutidas ao longo de **nove instâncias distintas**, oferecendo uma análise abrangente da eficácia de cada meta-heurística frente a diferentes cenários do problema.

Algoritmos

0.1 Simulated Annealing

A função pegará os pontos do documento em questão em que eles estão, e em seguida, gerar uma solução inicial. Em seguida, verificará essa solução, se ela cumpre com o requisito de ter suas coletas antes das entregas, e se sim, calculará o custo dessa solução e iniciará a variável temperatura. Depois, iniciará uma repetição que continuamente gerará soluções vizinhas calculando seu custo e se for melhor que o atual, esta se tornará a atual. Caso não seja melhor, haverá uma chance de aceitá-la com base em uma probabilidade calculada pela equação de decaimento exponencial, dependente da diferença de custo entre as soluções e da temperatura atual. Essa etapa é fundamental para permitir que o algoritmo escape de ótimos locais e explore melhor o espaço de soluções. A cada iteração, a temperatura é reduzida multiplicativamente, controlando gradualmente a aceitação de piores soluções. O processo continua até que a temperatura atinja um valor mínimo predefinido, momento em que a melhor solução encontrada até então é retornada como resultado final do

algoritmo.

0.2 BRKGA

O **Biased Random-Key Genetic Algorithm (BRKGA)** foi implementado como uma meta-heurística baseada em população, utilizando uma abordagem de **codificação indireta**. Diferente do SA, que opera diretamente nas soluções, o BRKGA manipula **cromossomos** — vetores de números reais (chaves aleatórias) entre 0 e 1. A inteligência do problema, incluindo as coordenadas dos pontos e as restrições de coleta e entrega do TSPPD, é encapsulada em um **Decoder (Decodificador)**.

O ciclo evolutivo do BRKGA consiste nas seguintes etapas:

1. **Inicialização da População:** Uma população inicial de cromossomos é gerada aleatoriamente. Cada cromossomo tem um tamanho fixo, correspondente ao número de pontos a serem permutados na rota (excluindo o depósito).

2. **Avaliação e Decodificação:** Cada cromossomo da população é passado para o Decoder. O Decoder é responsável por:

   - Mapear as chaves aleatórias do cromossomo para uma permutação inicial dos pontos.
   - Aplicar uma função de **reparo (`_fix_solution`)** para garantir que todas as restrições de precedência (coleta antes da entrega) sejam satisfeitas, rearranjando os pontos conforme necessário. Essa função é crítica para a validade e qualidade da solução.
   - Calcular o custo total da rota resultante (`_calculate_total_distance`),

que serve como a aptidão do cromossomo.

3. **Seleção (Elite e Não-Elite):** Os cromossomos são ordenados pela sua aptidão (custo). Uma porcentagem dos melhores indivíduos forma a **elite**, enquanto os restantes compõem o grupo **não-elite**.

4. **Recombinação (Crossover):** Novos descendentes são gerados através da combinação de genes de um pai da elite e um pai não-elite. A probabilidade de um gene ser herdado do pai elite é controlada pelo parâmetro `rho` (viés genético), promovendo a propagação de características de alta qualidade.

5. **Mutação:** Uma pequena parcela da nova população é composta por **mutantes**, que são cromossomos gerados completamente aleatoriamente. Isso injeta diversidade na população e ajuda o algoritmo a explorar novas regiões do espaço de busca, evitando a convergência prematura em ótimos locais.

6. **Evolução da População:** A próxima geração é formada pela elite atual, pelos descendentes gerados e pelos mutantes, repetindo o ciclo por um número predefinido de gerações. O melhor cromossomo encontrado ao longo de todas as gerações é mantido e sua solução decodificada representa o resultado final.

A modularidade do BRKGA, separando a lógica evolutiva da lógica do problema no Decoder, permite uma maior flexibilidade e reusabilidade.

—

Análise do Desvio Percentual (GAP)

O **Desvio Percentual (GAP)** é uma métrica fundamental em otimização para avaliar a qualidade de uma solução encontrada por um algoritmo heurístico ou meta-heurístico em relação a um valor de referência. Essa referência pode ser a solução ótima global para o problema (se conhecida) ou, como é o caso aqui, a **Melhor Solução Conhecida (BKS - Best Known Solution)**. O GAP é calculado da seguinte forma:

$$\text{GAP} = \left(\frac{\text{Custo da Solução Encontrada} - \text{BKS}}{\text{BKS}} \right) \times 100\%$$

Um valor de GAP igual a 0% indica que a solução encontrada pelo algoritmo é idêntica à BKS, ou seja, é a melhor solução conhecida. Valores positivos representam o percentual em que a solução do algoritmo se distancia da BKS, com valores menores indicando melhor desempenho. Quanto menor o GAP, mais próxima a solução encontrada está da referência.

Comparativo de Desempenho dos Algoritmos

A Tabela abaixo apresenta um comparativo detalhado do desempenho do SA e do BRKGA para as nove instâncias do problema do TSPPD. Para os algoritmos SA e BRKGA, o custo exibido na tabela representa o **melhor custo encontrado entre as diferentes execuções para aquela instância**, e o tempo é a **média do tempo de execução**. Os valores de BKS fornecidos pelo professor servem como a melhor referência conhecida para cada instância, permitindo uma análise precisa do quão próximas as soluções dos algoritmos

estão do ótimo.

Inst.	N	BKS	Custo SA	GAP SA	Custo BRKGA	GAP BRKGA	Tempo SA (s)	Tempo BRKGA (s)
pdtsp-n105	105	6019	12580.80	108.99%	8812.35	46.42%	0.127	23.73
pdtsp-n171	171	10203	30235.98	196.34%	19600.65	92.10%	0.216	50.44
pdtsp-n213	213	7314	20540.64	180.83%	15159.82	107.28%	0.30	70.89
pdtsp-n289	289	14447	53994.23	273.76%	41171.97	185.00%	0.42	120.44
pdtsp-n317	317	9901	50799.36	413.07%	32552.90	228.78%	0.49	145.28
pdtsp-n335	335	16002	71475.98	346.67%	48413.80	202.55%	0.51	152.50
pdtsp-n401	401	13399	73591.51	449.23%	47827.17	256.96%	0.612	217.08
pdtsp-n459	459	12350	64021.30	418.39%	45294.70	266.76%	0.81	264.52
pdtsp-n547	547	20126	124483.20	518.52%	84432.91	319.53%	1.02	384.35
pdtsp-n599	599	21017	148890.17	608.45%	93741.72	346.04%	1.27	449.92

As porcentagens de GAP são arredondadas para duas casas decimais para clareza na visualização.

Como pode ser observado na tabela, o **BRKGA consistentemente superou o SA em termos de qualidade da solução**, apresentando um desvio percentual (GAP) significativamente menor em relação ao BKS em todas as instâncias avaliadas. Isso sugere que o BRKGA foi mais eficaz em explorar a paisagem de busca e em lidar com as complexas restrições do problema, encontrando rotas de custo substancialmente mais baixo.

Em contrapartida, o **SA demonstrou uma vantagem considerável no tempo de execução**, sendo ordens de magnitude mais rápido que o BRKGA. No entanto, essa velocidade vem acompanhada de soluções de qualidade inferior, com GAPs percentuais muito mais elevados em comparação ao BKS. Este trade-off entre qualidade da solução e tempo computacional é uma característica comum no estudo de meta-heurísticas.

Conclusão

Neste trabalho, analisamos a aplicação de duas metaheurísticas — Simulated Annealing (SA) e Biased Random-Key Genetic Algorithm (BRKGA) — para a resolução do Problema do Caixeiro Viajante com Coleta e Entrega (TSPPD). Observamos que, embora o SA tenha apresentado um tempo de execução significativamente inferior, suas soluções mostraram-se consistentemente menos precisas quando comparadas ao BRKGA.

O BRKGA, por sua vez, destacou-se pela qualidade das soluções, apresentando GAPs percentuais muito menores em relação à Melhor Solução Conhecida (BKS) para todas as instâncias avaliadas. Isso demonstra sua maior eficácia na exploração do espaço de busca e no tratamento das restrições complexas do TSPPD.

Portanto, a escolha entre as duas abordagens depende diretamente do objetivo do problema: se o foco for obter soluções rapidamente com uma precisão aceitável, o SA pode ser suficiente; porém, para aplicações que demandam soluções de alta qualidade, mesmo com maior custo computacional, o BRKGA é claramente a escolha mais adequada.