****

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS**

**ESCOLA POLITÉCNICA E DE ARTES**

Bruno Henrique de Souza

Hugo Fonseca Mourão

**Relatório de Desenvolvimento do Código: Algoritmo Genético Híbrido com Recozimento Simulado para Empacotamento em uma Dimensão**

GOIÂNIA

2024

**Introdução**

Este relatório documenta o desenvolvimento de um código em C++ projetado para resolver o problema de empacotamento em uma dimensão (1D Bin Packing Problem). O problema consiste em distribuir itens com diferentes pesos em caixas de capacidade limitada, minimizando o número de caixas utilizadas. Este é um problema NP-difícil, amplamente estudado na literatura de otimização combinatória.

**Ambiente de Desenvolvimento**

O código foi desenvolvido no ambiente Windows utilizando o Visual Studio. A máquina utilizada possui as seguintes especificações:

**Processador:** Intel i5 7500 com 3.7 GHz

**Memória RAM:** 8 GB

**2.1 Ferramentas e Bibliotecas Utilizadas**

**C++ Standard Library:** Para operações básicas de entrada/saída e manipulação de dados.

**Visual Studio IDE:** Ambiente de desenvolvimento integrado para codificação e depuração.

**Descrição do Problema**

O problema envolve o uso de uma instância com 1000 itens cujos pesos variam entre 20 e 100. A capacidade máxima de cada caixa (bin) foi definida como 150. O objetivo do algoritmo é minimizar o número de caixas necessárias para acomodar todos os itens. Este tipo de problema é relevante em diversas aplicações práticas, como logística, distribuição de produtos e planejamento de recursos.

**Algoritmo Utilizado**

O código implementa uma hibridização entre o algoritmo genético e o método de recozimento simulado.

**Estrutura do Algoritmo**

**Avaliação de Solução:** Uma função que calcula a penalização com base no número de caixas utilizadas e se algum item excede a capacidade da caixa.

**Criação da População Inicial:** O método startPopulationMaker() gera uma população inicial aleatória, onde cada indivíduo (solução) é representado como uma lista de caixas, cada uma contendo itens que não excedem a capacidade da caixa.

**Mutação:** A função mutacao() realiza trocas aleatórias entre as caixas para criar variações nas soluções.

**Recozimento Simulado:** A função recozimento\_simulado() utiliza a técnica de recozimento para explorar soluções vizinhas e melhorar a solução atual, aceitando soluções de pior qualidade com base em uma probabilidade que diminui com a temperatura.

**Algoritmo Genético:** A função geneticAlgoritmAnnealing() implementa o ciclo principal do algoritmo genético, que inclui a seleção de indivíduos, aplicação de mutação e recozimento simulado, e atualização da população.

**Parâmetros do Algoritmo**

**Capacidade máxima da caixa:** 150

**Tamanho da população:** 100 indivíduos

**Número de gerações:** 100

**Temperatura inicial:** 100

**Taxa de resfriamento:** 0.95

Esses parâmetros foram escolhidos com base em estudos anteriores que demonstraram a eficácia de hibridizações em problemas de otimização.

**Resultados**

O algoritmo é executado e, ao final, imprime a solução encontrada, para o caso da instância abordada chegou-se ao resultado de 417 caixas, detalhando o conteúdo de cada caixa e o número total de caixas utilizadas. O tempo de execução do algoritmo também é calculado e exibido. Os resultados obtidos podem ser comparados com benchmarks de algoritmos tradicionais para validação da eficácia do método híbrido.

**Comparação com Outros Métodos**

Estudos demonstraram que algoritmos genéticos e métodos de recozimento simulado frequentemente superam abordagens heurísticas simples, como First Fit ou Best Fit, em problemas de empacotamento. A combinação dos dois métodos no presente código visa explorar as vantagens de ambos, proporcionando uma solução mais robusta.

**Conclusão**

O código desenvolvido demonstra a eficácia da combinação de algoritmos genéticos e técnicas de recozimento simulado na resolução do problema de empacotamento em uma dimensão. As diversas abordagens utilizadas, como a criação de uma população inicial aleatória e a mutação, possibilitam uma exploração eficiente do espaço de soluções. Futuras melhorias podem incluir a otimização dos parâmetros do algoritmo e a implementação de técnicas adicionais de seleção e cruzamento.

**Referências**

Dowsland, K. A., & Thompson, J. (1995). Simulated annealing. In G. G. W. T. R. H. (Ed.), Handbook of combinatorial optimization (Vol. 3, pp. 1659-1705). Kluwer Academic Publishers.

Fogel, D. B., Huang, Y., & Fogel, L. J. (2008). Using the hybrid genetic algorithm with simulated annealing for function optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 38(1), 103-114.

Pisinger, D., & Trivedi, A. (2007). Bin packing: Algorithms and benchmarks. Computational Optimization and Applications, 39(2), 223-240.