Iterated Local Search (ILS)

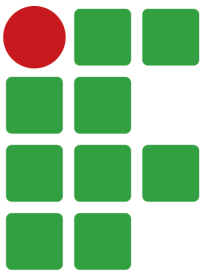
Pesquisa Operacional T2

Caike Lobo

Ivaniso da Silva

Pedro Ferraiuoli

Matheus André



INSTITUTO
FEDERAL
Fluminense

Código do Problema da Mochila Binária

Recapitulando o problema

O problema da mochila binária é um desafio clássico de otimização que envolve a seleção de itens para maximizar o valor total transportado por uma mochila, sem ultrapassar sua capacidade de peso. Este problema é amplamente estudado em algoritmos de programação e possui aplicações práticas em diversas áreas, como logística, finanças e planejamento de recursos.

Definição do problema

No contexto do problema da mochila binária, temos um conjunto de itens, cada um com um peso e um valor. A tarefa consiste em determinar quais itens incluir em uma mochila com uma capacidade máxima definida, de modo a maximizar o valor total dos itens selecionados. Cada item pode ser incluído ou não, o que caracteriza a natureza binária do problema.

Neste cenário, a metaheurística Iterated Local Search (ILS) é utilizada para abordar a complexidade combinatória do problema. O ILS é uma técnica que combina busca local com perturbações controladas, permitindo escapar de ótimos locais e explorar soluções em um espaço de busca mais amplo. Essa abordagem é especialmente útil em problemas como o da mochila binária, onde a solução ótima pode não ser facilmente alcançada por métodos tradicionais.

Pseudocódigo base do ILS

algoritmo ILS

gere uma solução inicial s_0

$\hat{s} \leftarrow$ BuscaLocal (s_0)

enquanto (critério de parada não for satisfeito) **faça**

$s' \leftarrow$ Perturbação ($\hat{s}$, histórico)

$\hat{s}' \leftarrow$ BuscaLocal (s')

$\hat{s} \leftarrow$ CritériodeAceitação ($\hat{s}$, $\hat{s}'$, histórico)

fim-enquanto

fim-algoritmo

Classes do problema

Item.class

Representa um objeto que você pode escolher para colocar na mochila. Cada item tem dois atributos principais: Nome (Nome do item. Ex: Borracha, Lápis, etc), seu peso ou espaço que ocupa na mochila e seu valor ou ganho para o problema.

Mochila.class

Representa um espaço de capacidade limitada onde você pode colocar uma seleção de itens. A mochila tem uma capacidade máxima, que limita o quanto ela pode carregar em termos de peso. O objetivo do problema é maximizar o valor total dos itens que você coloca na mochila sem exceder essa capacidade.

Todas as variáveis de Mochila são preenchidas em sua construção, o construtor lê o .txt indicado e com isso ele sabe o tamanho que o vetor itens deve ter, o peso maximo da mochila e por fim ele preenche o vetor Itens com os Itens do .txt

MetodosILS.class

Representa o ILS Sendo Aplicado no problema da mochila binária, com variáveis como:

CriterioDeParadaDoILS - diz quantas vezes o ciclo do ILS de Perturbação e Busca Local Ocorre
CriterioDeParadaBuscaLocal - diz quantas vezes a busca local pode ser feita por Ciclo ILS

TamanhoPerturbação - dita a quantidade de variáveis que a perturbação vai atingir.

O Método **BuscaLocal**, ele pega a solução que foi perturbada e tenta ajustar valores para tentar melhorar a solução, ele pode tanto adicionar 1 item a mochila, quanto ele pode retirar um item e colocar outro, o número de tentativas dele é determinado pela variável CriterioDeParadaBuscaLocal.

A cada tentativa de melhorar a solução, a BuscaLocal compara com a melhor solução encontrada por ele até aquele momento, se a tentativa de solução não for melhor, ele adiciona +1 ao *numeroDeErros,* A BuscaLocal será encerrada (Assim prosseguindo para uma perturbação) se o *numeroDeErros chegar* até o numero determinado pelo **CriterioDeParadaBuscaLocal,** mas caso a tentativa de solução for melhor, ele irá salvar aquela solução como a melhor encontrada pela BuscaLocal até o momento e irá zerar o *numeroDeErros,* resumindo, a busca local irá tentar melhorar a solução até o ponto em que irá falhar (*numeroDeErros)* seguidas.

Por Final, a Busca Local Retorna a melhor solução encontrada por ela.

O método **Perturbação**, pega a variável TamanhoPerturbação e percorre a Melhor Solução Atual, Ele irá retirar aleatoriamente da mochila X números de itens, X é um aleatório entre 1 e uma porcentagem máxima de itens da mochila, a porcentagem máxima é determinada pelo *TamanhoPerturbação.*

O Método **EncherVetor**, é basicamente uma Busca Local personalizada, feita para encher o vetor rapidamente e obter um vetor com uma solução adequada logo de início, facilitando o trabalho dos Ciclos de Busca e Perturbação que virão a seguir.

E o Método **EncontrarSolucao** é aquele responsável por realizar os ciclos de BuscaLocal e Perturbacao, além de manter guardado qual é a melhor solução que o código encontrou até o momento.

Ele primeiro cria um vetor de tamanho X, sendo X o número de Itens no Total, após isso ele chama o método **EncherVetor,** que por sua vez, enche o vetor com uma solução viável, Após isso ele começa um ciclo for, que vai acontecer Y número de vezes, sendo Y o CriterioDeParadaILS, a partir daí ele irá realizar uma perturbação e uma buscaLocal por

ciclo, sempre recebendo a solução que o buscaLocal está enviando e comparando com a melhor solução encontrada até agora, caso a solução que o busca local enviou seja melhor do que a melhorSolucao armazenada no EncontrarSolução, a solução recebida passa a ser a melhorSolucao.

Existem outros métodos auxiliares, como **verificarNumeroDeItens,** que verifica quantos itens estão dentro da mochila, **verificarPesoMochila e verificarValorMochila**, que São bem auto explicativos e **verificarValidadeMochila,** que verifica se a solução passou do peso máximo da mochila ou não

Main.class

Esta classe tem a responsabilidade de indicar o caminho e nome do arquivo txt e iniciar os valores dos seguintes parâmetros: Número de perturbações, tamanho de cada perturbação e a quantidade de buscas locais por cada perturbação.

ArquivoUtils.class

É apenas uma classe que lida com o .txt de Saída, o Output.txt

Valores de entrada

1. input0.text(1000 itens, 5000 de Peso Máximo)
2. input1.txt (5000 itens, 30000 de Peso Máximo)
3. input2.txt (10000 items, 20000 de Peso Máximo)

Como executar o algoritmo

1. Baixar o código do GIT
2. Importe para o Eclipse
3. Indique o .txt preenchido adequadamente
4. Indique um local para o Output.txt (Na classe ArquivoUtils)
5. Coloque os parâmetros Adequados para o nível de Complexidade da Lista
6. Execute a classe Main.class