3

BUSCA EM DESCIDA DE VIZINHANÇA VARIÁVEL

As heurísticas **Variable Neighborhood Descent (VND)** e **Variable Neighborhood Search (VNS)** são métodos muito populares em otimização combinatória.

VARIABLE NEIGHBORHOOD DESCENT (VND)

O **VND**, veja Algoritmo 3.0.1, é uma heurística de busca local avançada que utiliza um conjunto predefinido de estruturas de vizinhança $\{N_1, N_2, \ldots, N_n\}$ de forma determinística (sequencial). O **VND** explora de forma sistemática várias vizinhanças diferentes, uma de cada vez. A esperança é que, ao mudar de vizinhança, se consiga escapar de ótimos locais que são ruins em relação a uma vizinhança, mas não em outras. Partindo da primeira vizinhança (N_1), toda vez que um ótimo local é obtido, muda-se para a próxima vizinhança. Porém, toda vez que uma solução melhor do que a incumbente for encontrada, atualiza-se esta solução e retorna-se a primeira vizinhança (N_1). Se todas as vizinhanças tenham sido exploradas sem sucesso, o método para.

VARIABLE NEIGHBORHOOD SEARCH (VNS)

O **VNS**, veja Algoritmo 3.0.2, é uma meta-heurística que utiliza as mesmas múltiplas estruturas de vizinhança do **VND**, mas de forma mais ampla, combinando a diversificação com a intensificação. O **VNS** muda sistematicamente a vizinhança para escapar de ótimos locais, utilizando um processo chamado *shaking* (agitação ou perturbação) para mover a solução atual para longe de um ótimo local, antes de aplicar uma busca local, geralmente o próprio **VND**, para intensificar a busca na nova região do espaço de soluções.

A agitação é feita escolhendo-se uma vizinhança N_k, geralmente aleatoriamente, para perturbar a solução corrente. Aplica-se um número aleatório de movimentos dessa vizinhança escolhida para modificar a solução corrente. Isso diversifica a busca. Depois, faz-se uma busca local, frequentemente o próprio **VND**, ou seja, **VNS** com **VND** é aplicada para encontrar um ótimo local. Isso intensifica a busca.

PSEUDOCÓDIGOS DAS HEURÍSTICAS VND E VNS

Algorithm 3.0.1: Variable Neighborhood Descent (VND)

Entrada: S : solução inicial
Entrada: $N = \{N_1, \ldots, N_n\}$: conjunto de vizinhanças
Saída : S : melhor solução

```
1  k ← 1;
2  while k ≤ n do
3  |  S' ← BuscaLocal(S, N_k);
4  |  if f(S') > f(S) then
5  |  |  S ← S';
6  |  |  k ← 1 ; // reinicia com a primeira vizinhança
7  |  end
8  |  else
9  |  |  k ← k + 1 ; // passa para a próxima vizinhança
10 |  end
11 end
12 return S
```

O QUE DEVE SER FEITO

1. Com o código em anexo, você deve montar um conjunto de experimentos que mostrem qual combinação de vizinhaças do **VND**, 1-Opt seguido da 2-Op ou 2-Opt seguido da 1-Opt, é melhor para as instâncias resolvidas. Note que o **VND** implementado têm duas estratégias de implantação da busca local, o primeiro movimento melhorante e o melhor movimento entre todos. Além disso, há a ideia do **Random VND** implementado.

2. Implemente o **Smart VNS** ou o Algoritmo 3.0.3:

3. Depois de testar o **VND**, use o **VNS** com todos os **VNDs** acima descritos e descubra qual combinação de **VNS** foi melhor.

4. Explique qual combinação apresentou o melhor desempenho. Por que ela obteve esse resultado? Manteve-se a tendência do trabalho prático anterior? Quem ganhou lá ganhou aqui também?

   PS: Use o melhor método construtivo do primeiro trabalho prático para fazer estes testes

Algorithm 3.0.2: Variable Neighborhood Search (VNS) Básico

Entrada: S : solução inicial
Entrada: $h_{\max}$: número máximo de iterações
Entrada: $N = \{N_1, \ldots, N_n\}$: conjunto de vizinhanças
Saída : S^* : melhor solução

$S^* \leftarrow S$
$h \leftarrow 0$
while $h \leq h_{\max}$ **do**
    $h \leftarrow h + 1$
    $k \leftarrow 1$
    while $k \leq n$ **do**
        $S' \leftarrow$ `Agitação`(S, N_k) `// Perturba` S `na vizinhança` N_k
        $S' \leftarrow$ `VND(`S', N`) // Busca local usando VND`
        if $f(S') > f(S^*)$ **then**
            $S^* \leftarrow S'$
            $k \leftarrow 1$ `// reinicia a perturbação com a primeira vizinhança`
        else
            $k \leftarrow k + 1$ `// Aumenta o nível de perturbação (próxima vizinhança)`
        end
    end
end
return S^*

Algorithm 3.0.3: Smart Variable Neighborhood Search (VNS) Básico

Entrada: S : solução inicial
Entrada: $h_{\max}$: número máximo de iterações
Entrada: $p_{\max}$: número máximo de perturbações/agitações na mesma vizinhança
Entrada: $N = \{N_1, \ldots, N_n\}$: conjunto de vizinhanças
Saída : S^* : melhor solução

```
1  S* ← S
2  h ← 0
3  while h ≤ h_max do
4      h ← h + 1
5      k ← 1
6      p ← 1
7      while k ≤ n do
8          S' ← Agitação(S, N_k, p) // Perturba S na vizinhança N_k com
             intensidade p
9          S' ← SVND(S', N) // Busca local usando Smart VND
10         if f(S') > f(S*) then
11             S* ← S'
12             k ← 1 // reinicia a perturbação com a primeira
                 vizinhança
13             p ← 1 // reinicia a intensidade da perturbação
14         else
15             if p = p_max then
16                 k ← k + 1 p ← 1 // reinicia a intensidade da
                     perturbação
17             else
18                 p ← p + 1 // aumenta a intensidade da perturbação
19             end
20         end
21     end
22 end
23 return S*
```