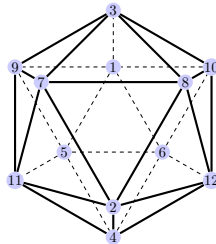


# Metaheurísticas - 6 - Busca local (*local search*)

Alexandre Checoli Choueiri

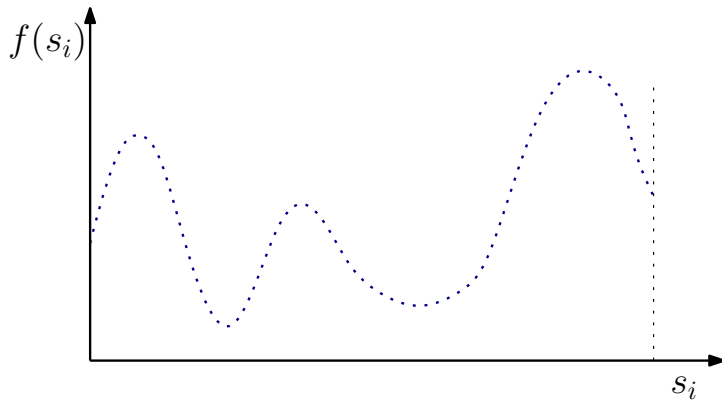
29/03/2023



## ① Introdução

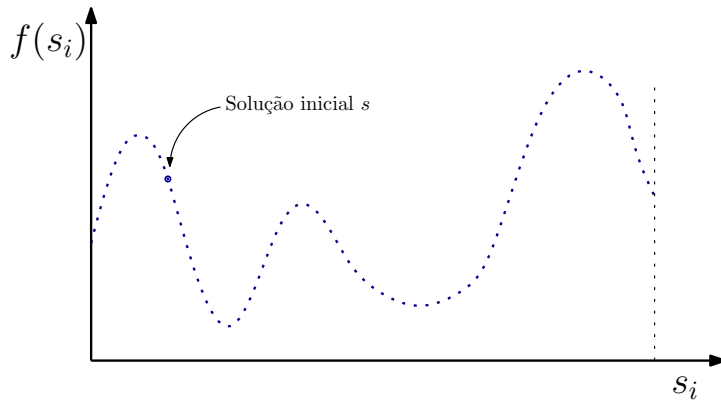
A **Busca Local** (*Local Search*) é provavelmente a mais simples e mais antiga metaheurística. Ela começa com uma solução inicial, e a cada iteração faz a substituição da solução por uma solução vizinha com melhor valor da função objetivo. A busca é interrompida quando todos os vizinhos da solução atual tem valores piores da função objetivo, ou seja, atinge-se um **ótimo local**.

## Representação visual



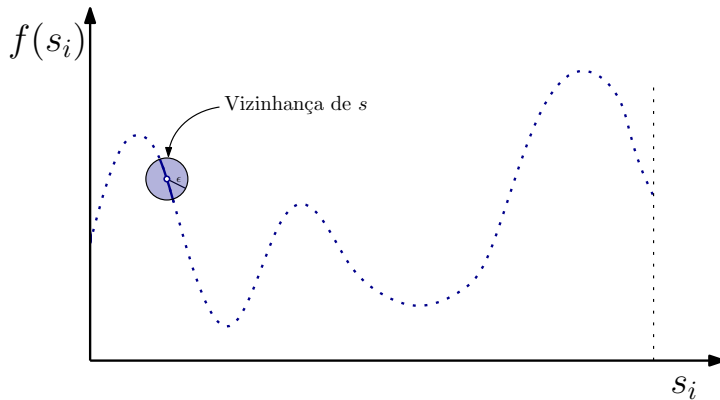
Podemos entender o que a busca local faz usando a nossa representação do *fitness landscape* (considerando um problema de minimização).

## Representação visual



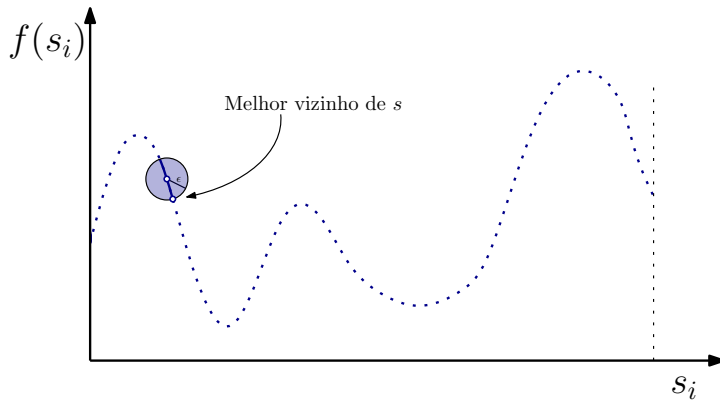
Precisamos de uma solução inicial  $s$  para dar início à busca.

## Representação visual



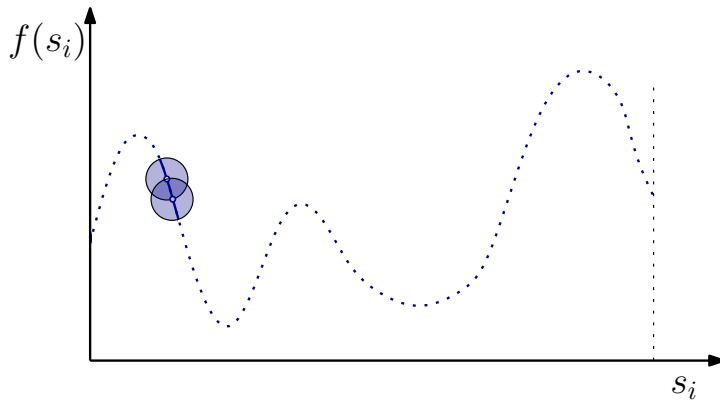
Em seguida, considerando alguma estrutura de vizinhança  $N$  geramos os vizinhos de  $s$ .

## Representação visual



Verificamos então se existe alguma solução vizinha  $s'$  com valor da função objetivo melhor do que a solução atual.

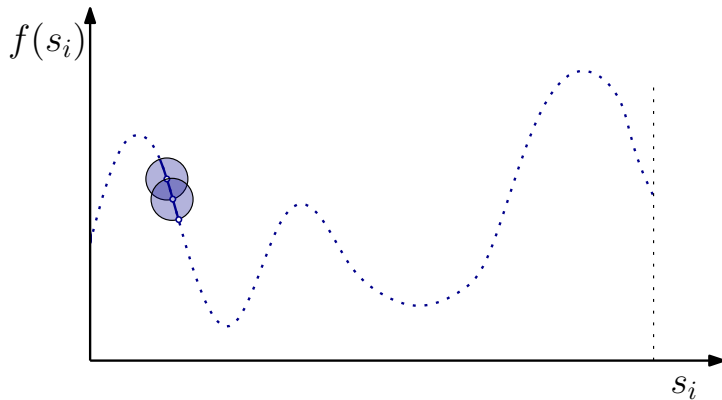
## Representação visual



Se existe, substituímos a solução inicial pelo vizinho encontrado  $s'$ , e novamente geramos os seus vizinhos.

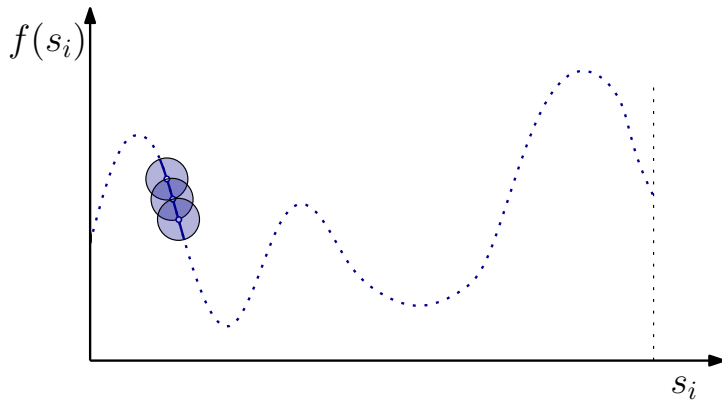


## Representação visual



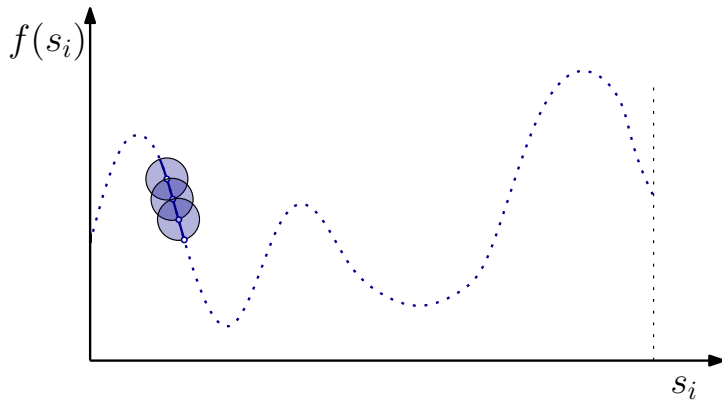
O processo é então repetido até que nenhuma solução vizinha tenha um valor de função objetivo melhor do que a solução atual.

## Representação visual



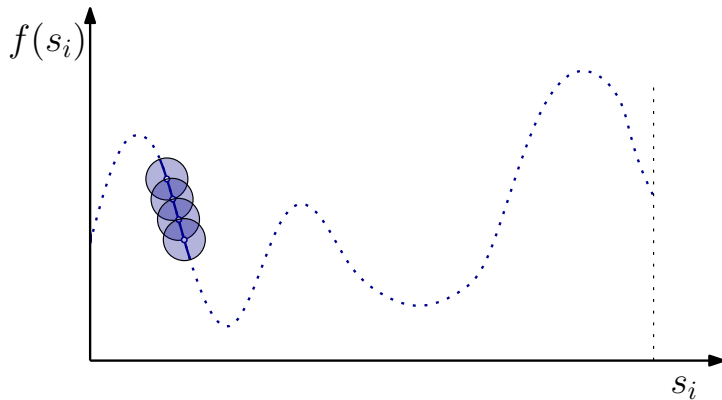
O processo é então repetido até que nenhuma solução vizinha tenha um valor de função objetivo melhor do que a solução atual.

## Representação visual



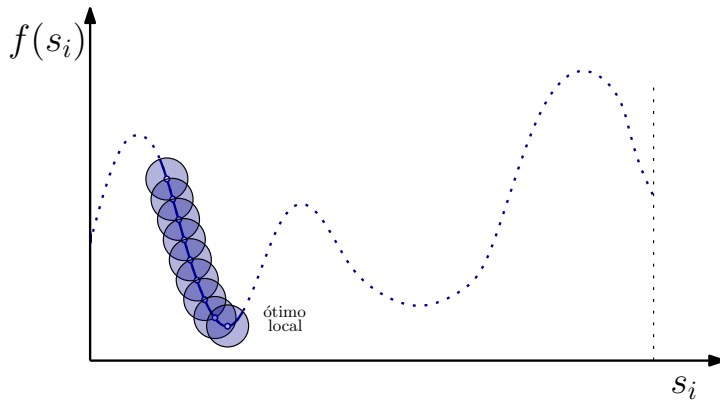
O processo é então repetido até que nenhuma solução vizinha tenha um valor de função objetivo melhor do que a solução atual.

## Representação visual



O processo é então repetido até que nenhuma solução vizinha tenha um valor de função objetivo melhor do que a solução atual.

## Representação visual



No fim do algoritmo a busca local **encontra o ótimo local** em relação à vizinhança  $N$ .

# Pseudocódigo

Abaixo segue um pseudocódigo template da busca local.

---

**Algorithm 1** Template genérico busca local

---

```
 $s$  = GeraSolInicial()                                ▷ Gera uma solução inicial
while Critério de parada não for satisfeito do
    GeraVizinhos( $\mathbb{N}(s)$ )                                ▷ Gera vizinhos de  $s$ 
     $s'$  = Selecciona( $\mathbb{N}(s)$ )                                ▷ Selecciona um vizinho de  $s$ ,  $s'$ 
    if  $s'$  for melhor do que  $s$  then
         $s = s'$                                             ▷ Substitui  $s$  por  $s'$ 
    else
        Pare
    end if
end while
return  $s$ .
```

---

# Pseudocódigo

## Componentes no design do algoritmo

Em que:

1. `GeraSolInicial()`: Gera uma solução inicial para o problema ( *Greedy*, por exemplo).
2. `GeraVizinhos( $\mathbb{N}(s)$ )`: Gera os vizinhos de  $s$  pela estrutura de vizinhança  $\mathbb{N}$ .
3. `Seleciona( $\mathbb{N}(s)$ )`: Seleciona algum vizinho de  $s \in \mathbb{N}(s)$ .

# Pseudocódigo

## Componentes no design do algoritmo

Em que:

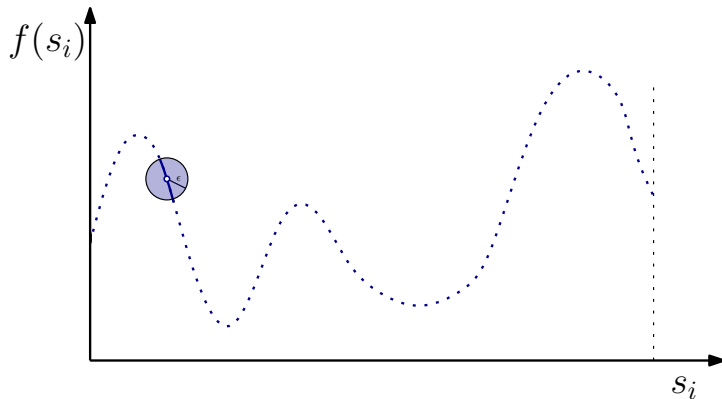
1. GeraSolInicial(): Gera uma solução inicial para o problema ( *Greedy*, por exemplo).
2. GeraVizinhos( $\mathbb{N}(s)$ ): Gera os vizinhos de  $s$  pela estrutura de vizinhança  $\mathbb{N}$ .
3. Seleciona( $\mathbb{N}(s)$ ): Seleciona algum vizinho de  $s \in \mathbb{N}(s)$ .

Dessa forma, **existem 3 decisões importantes** ao se fazer o design da Busca Local. Já vimos métodos para gerar uma solução inicial e os conceitos de geração de vizinhança, falta definir a **forma de seleção do vizinho**.



# Busca Local

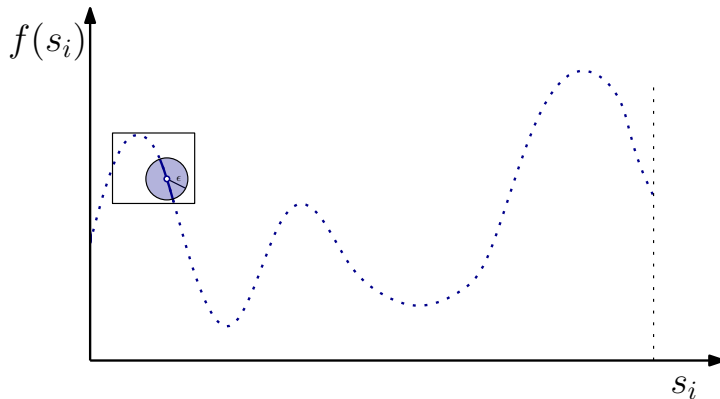
Seleção do vizinho



Verificando o *fitness landscape* para a busca local

# Busca Local

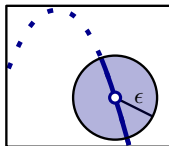
## Seleção do vizinho



Mais especificamente na etapa de geração dos vizinhos.

# Busca Local

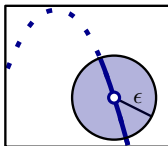
## Seleção do vizinho



A busca local **nunca** aceita vizinhos com valores piores da função objetivo.

# Busca Local

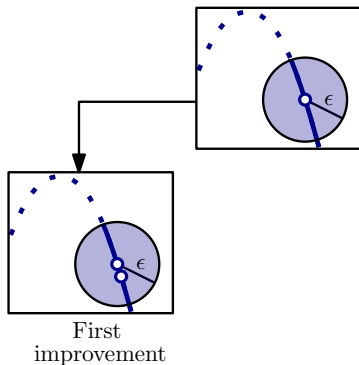
## Seleção do vizinho



Mas podem existir **diversos vizinhos** com valores melhores, qual escolher então?

# Busca Local

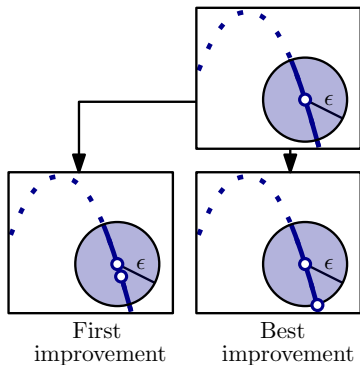
## Seleção do vizinho



Uma opção é escolher o primeiro vizinho com função melhor que a solução atual, esse método de seleção é chamado de *first improvement* (primeira melhoria).

# Busca Local

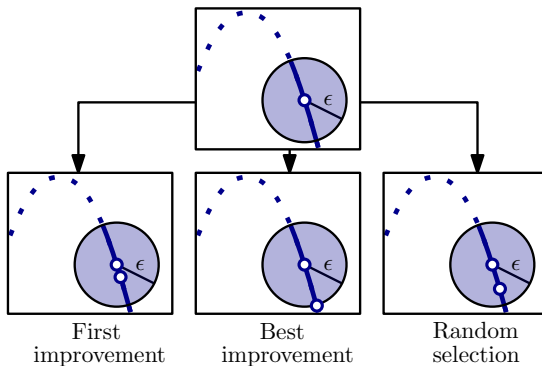
## Seleção do vizinho



Podemos ainda gerar toda a vizinhança e escolher o vizinho com a melhor função objetivo, esse método de seleção é chamado de *best improvement* (melhor "melhoria").

# Busca Local

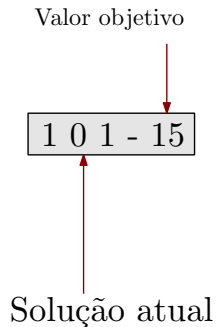
## Seleção do vizinho



Um terceiro método consiste em, dentre todos os vizinhos que são melhores que a solução atual, selecionar um aleatoriamente, esse método é chamado de *random selection* (seleção aleatória).

# Busca Local

## Seleção do vizinho - exemplo

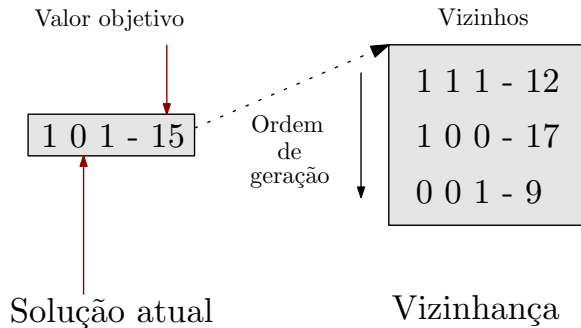


Por exemplo, considere um problema com representação binária e o operador *flip*.



# Busca Local

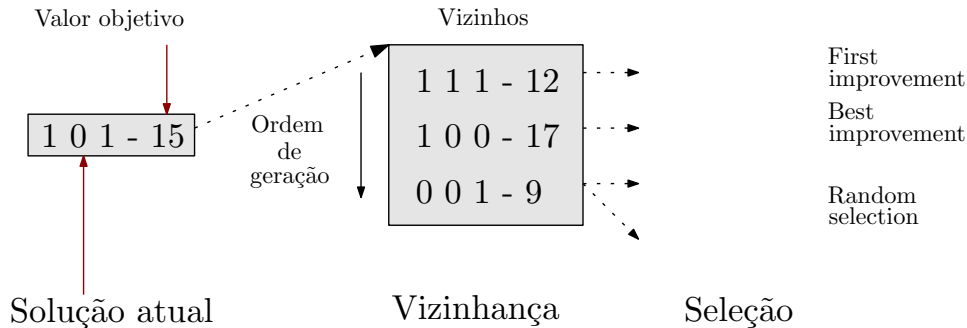
## Seleção do vizinho - exemplo



Os vizinhos da solução inicial, bem como seus valores de função objetivo são mostrados acima.

# Busca Local

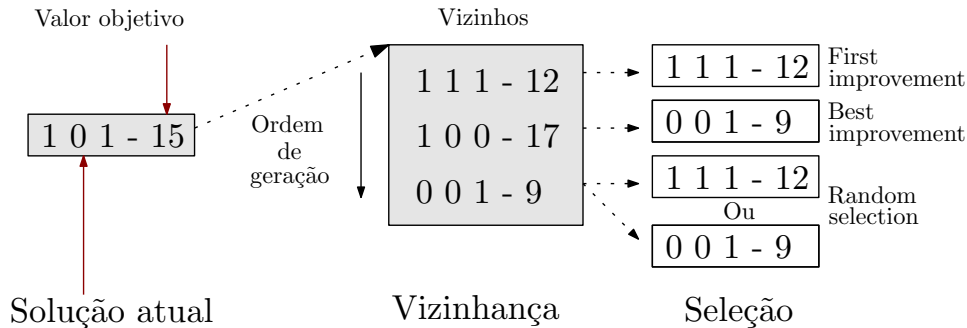
## Seleção do vizinho - exemplo



Quais soluções seriam escolhidas, de acordo com os tipos de seleção possíveis?

# Busca Local

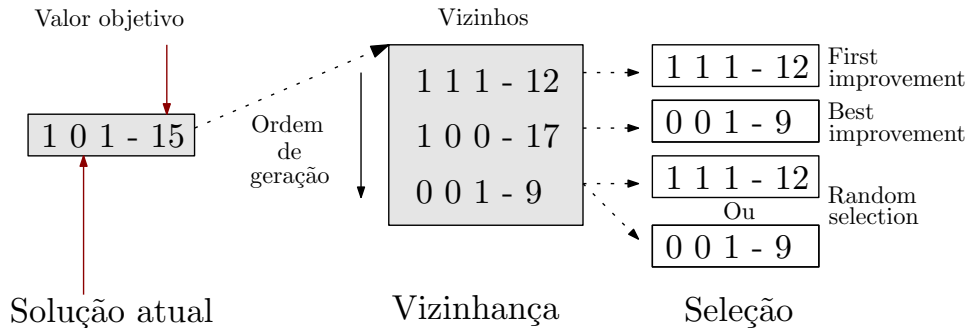
## Seleção do vizinho - exemplo



Quais soluções seriam escolhidas, de acordo com os tipos de seleção possíveis?

# Busca Local

## Seleção do vizinho - conclusões



Quais soluções seriam escolhidas, de acordo com os tipos de seleção possíveis?

### Conclusões

Nota-se que o método *Best Improvement* gera **toda** a vizinhança da solução  $s$ , o que pode ser **computacionalmente caro**. Já o *First Improvement* não seleciona o melhor vizinho, porém o tempo de geração é muito menor. Em muitas aplicações na prática têm-se observado que estratégia *First Improvement* leva a soluções de **mesma qualidade** da *Best Improvement* em menos tempo computacional (**mas isso não é uma regra!**).

# Atividade 1

1. Considerando o problema de sua escolha, busque pelo menos 2 artigos científicos em que o algoritmo de busca local é utilizado (geralmente a etapa de busca local é usada como uma parte de outros algoritmos). Nesses artigos identifique:
  - 1.1 As estruturas de vizinhança utilizadas.
  - 1.2 Qual o método de seleção do vizinho utilizado.