

Busca Tabu

Gustavo Post Sabin

Introdução

- ▶ 1986 – Fred W. Glover
- ▶ Ideia
 - cruzar a fronteira da otimalidade local

Introdução

▶ Características

- Memória com mecanismo de controle
- Estratégia para evitar ciclos

▶ Objetivos

- Escapar de ótimos locais
 - Identificar regiões promissoras
- 

Memória

► Dimensões

- proximidade no tempo
- frequência
- qualidade
- impacto

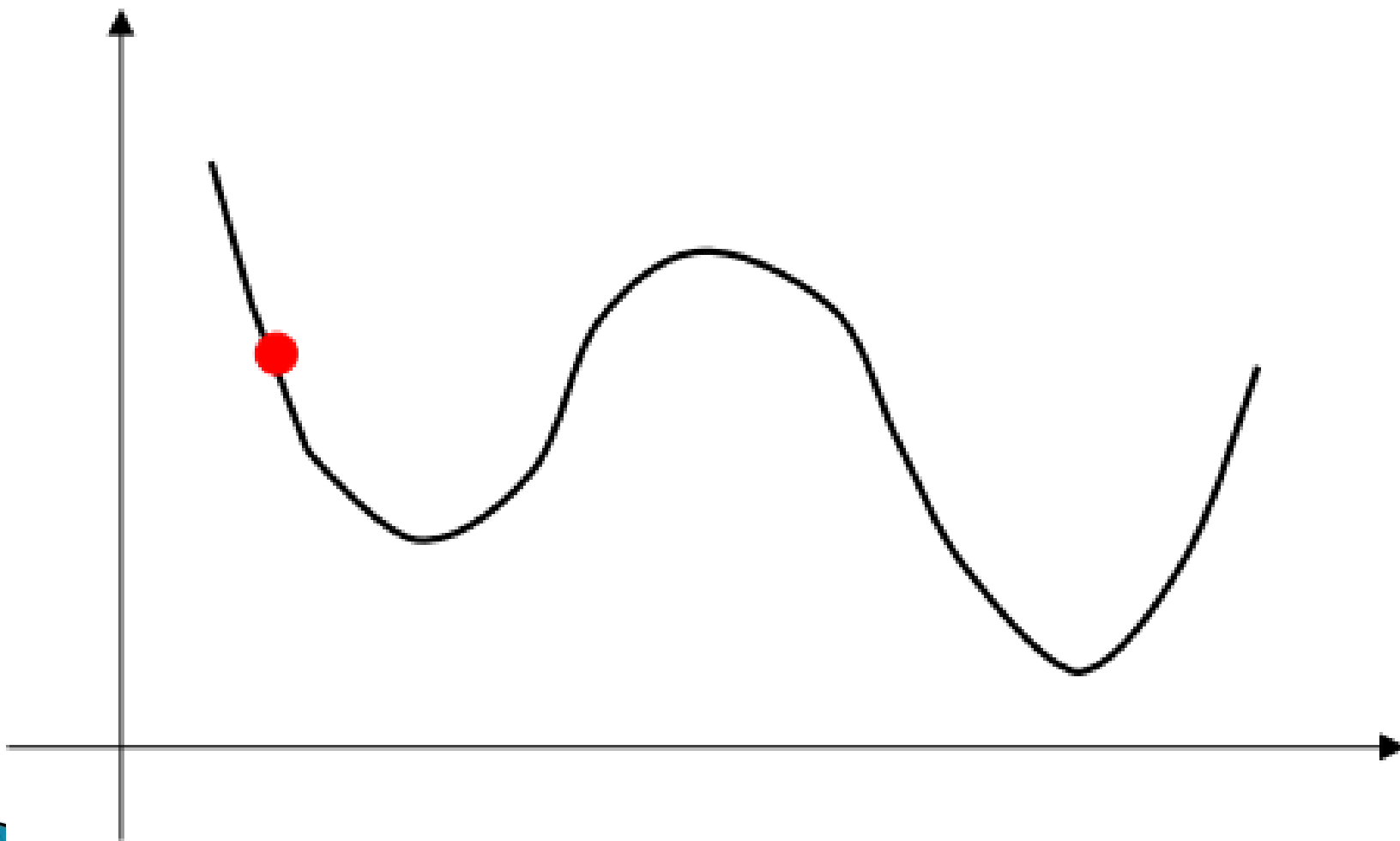
► Tipos

- explícita: soluções–elite, vizinhos promissores não–explorados
- atributiva

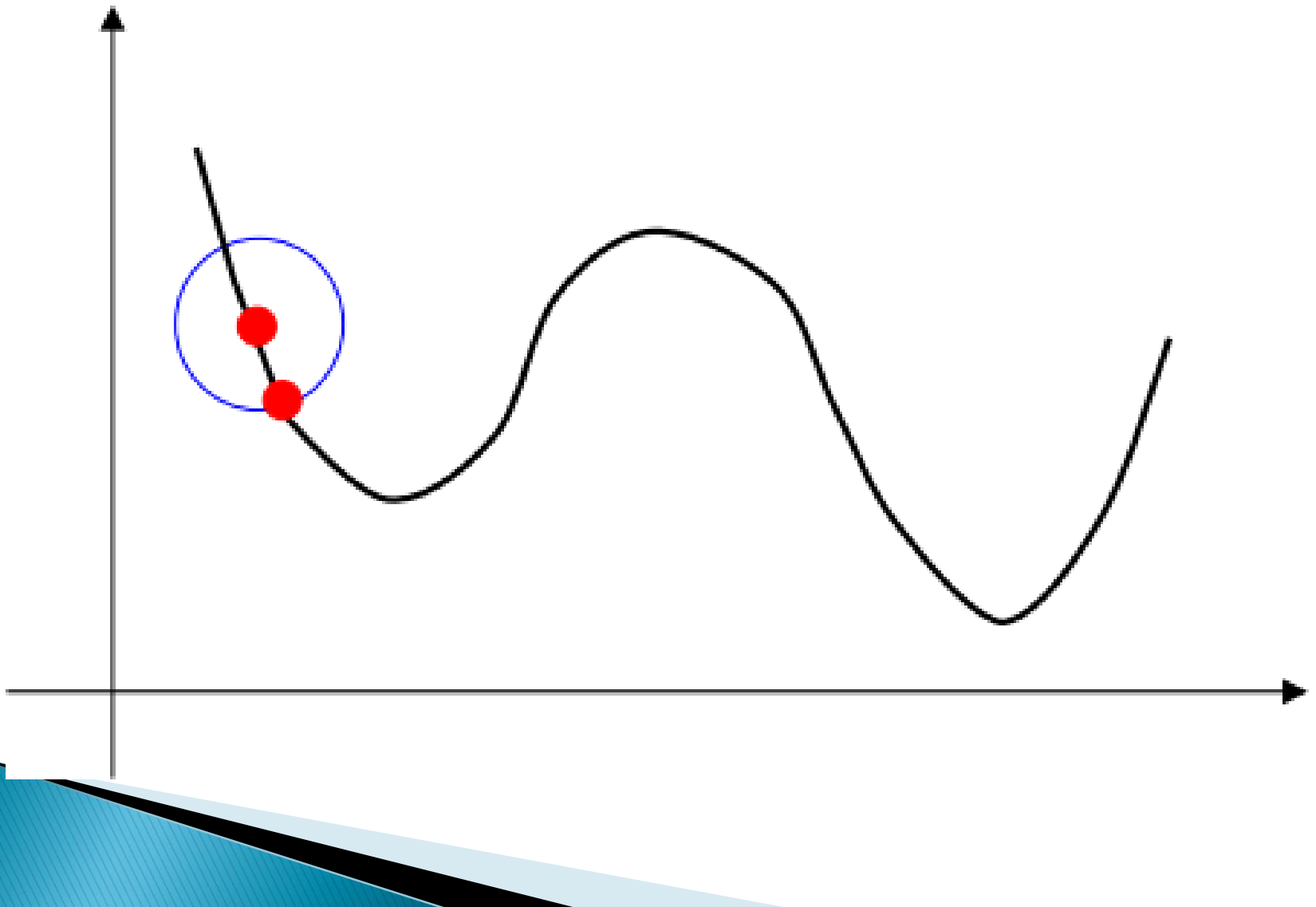
Memória

- ▶ Inviável armazenar todas as soluções visitadas
 - Limitar tamanho
 - Adotar estratégia de renovação
- Lista de movimentos reversos

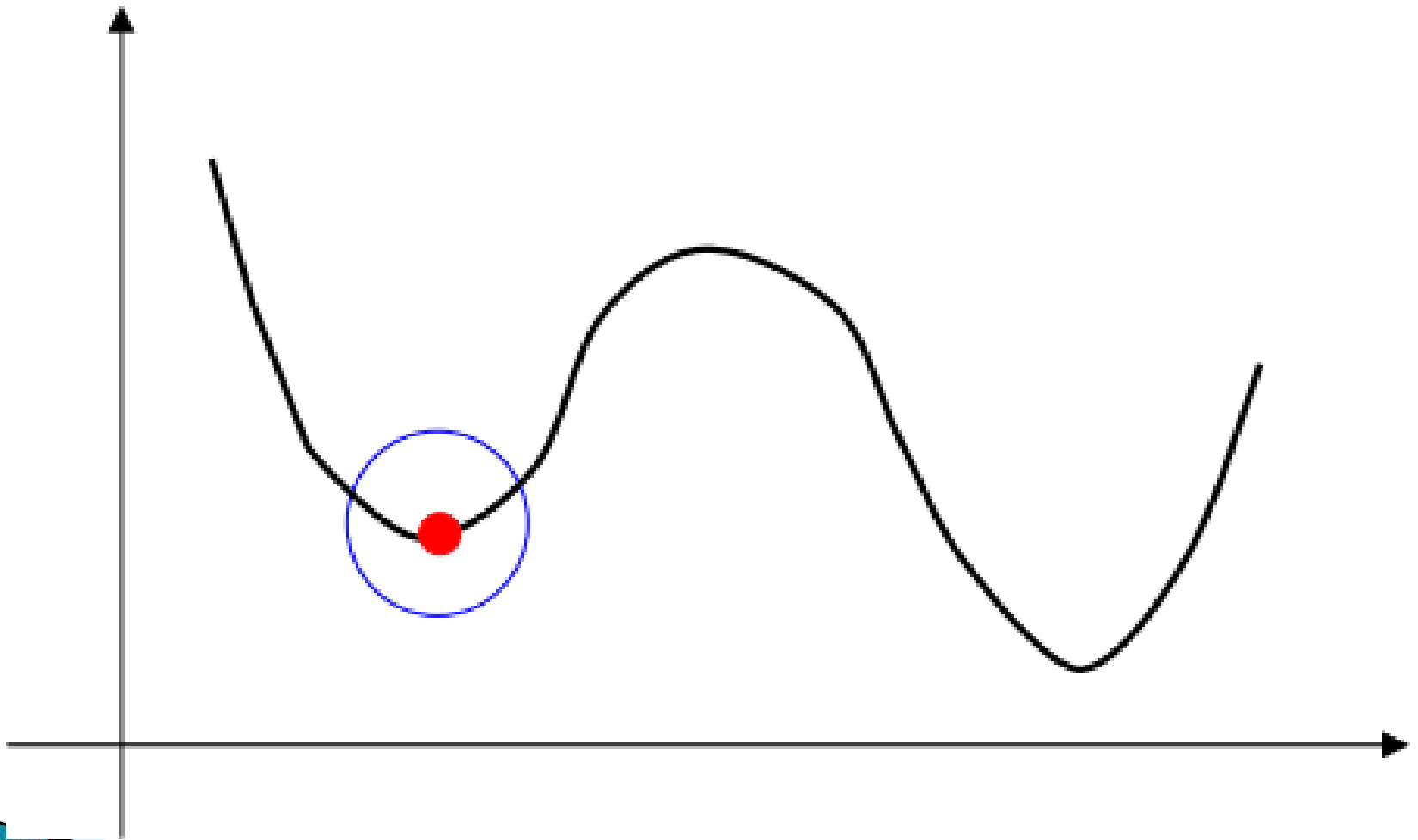
Busca



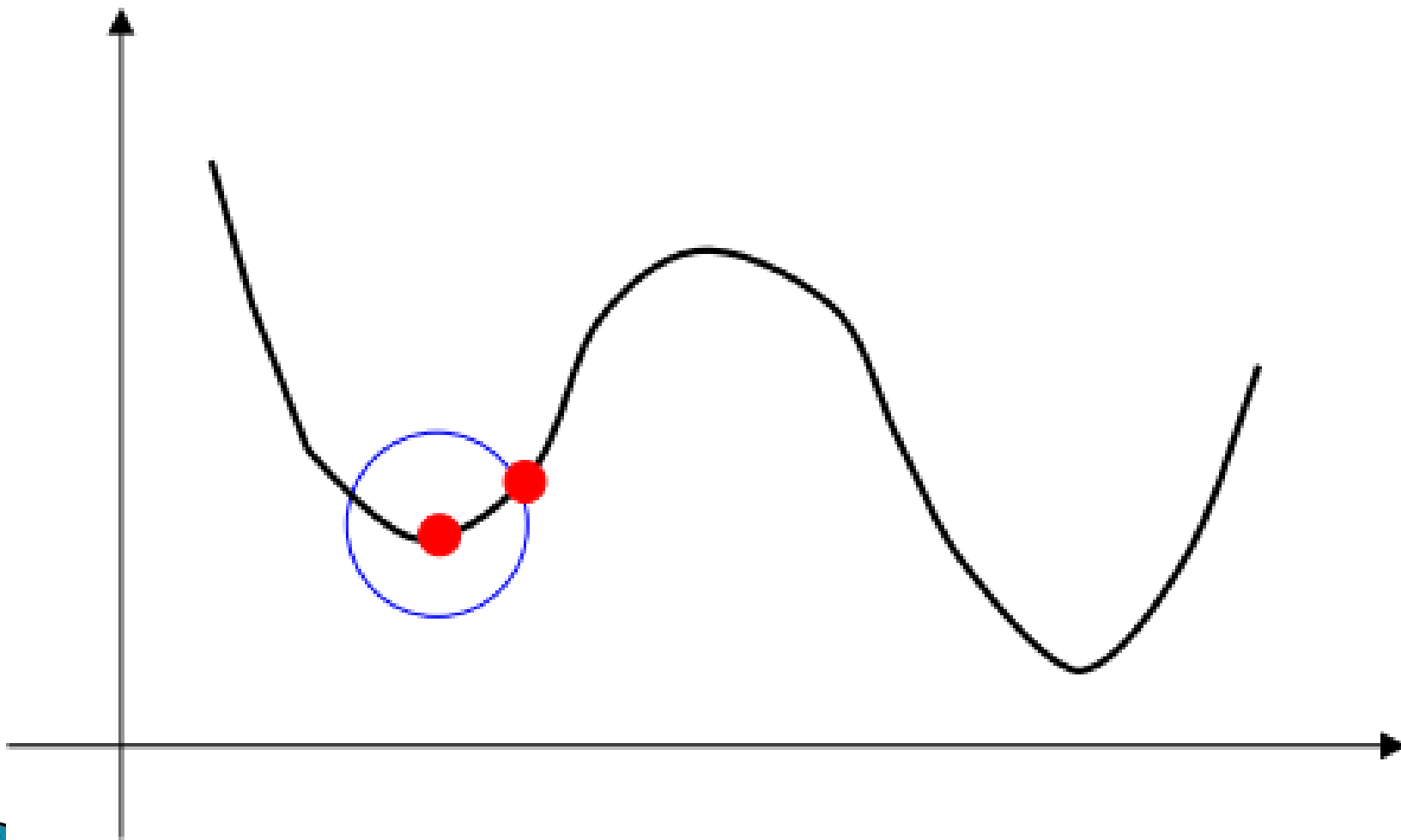
Busca



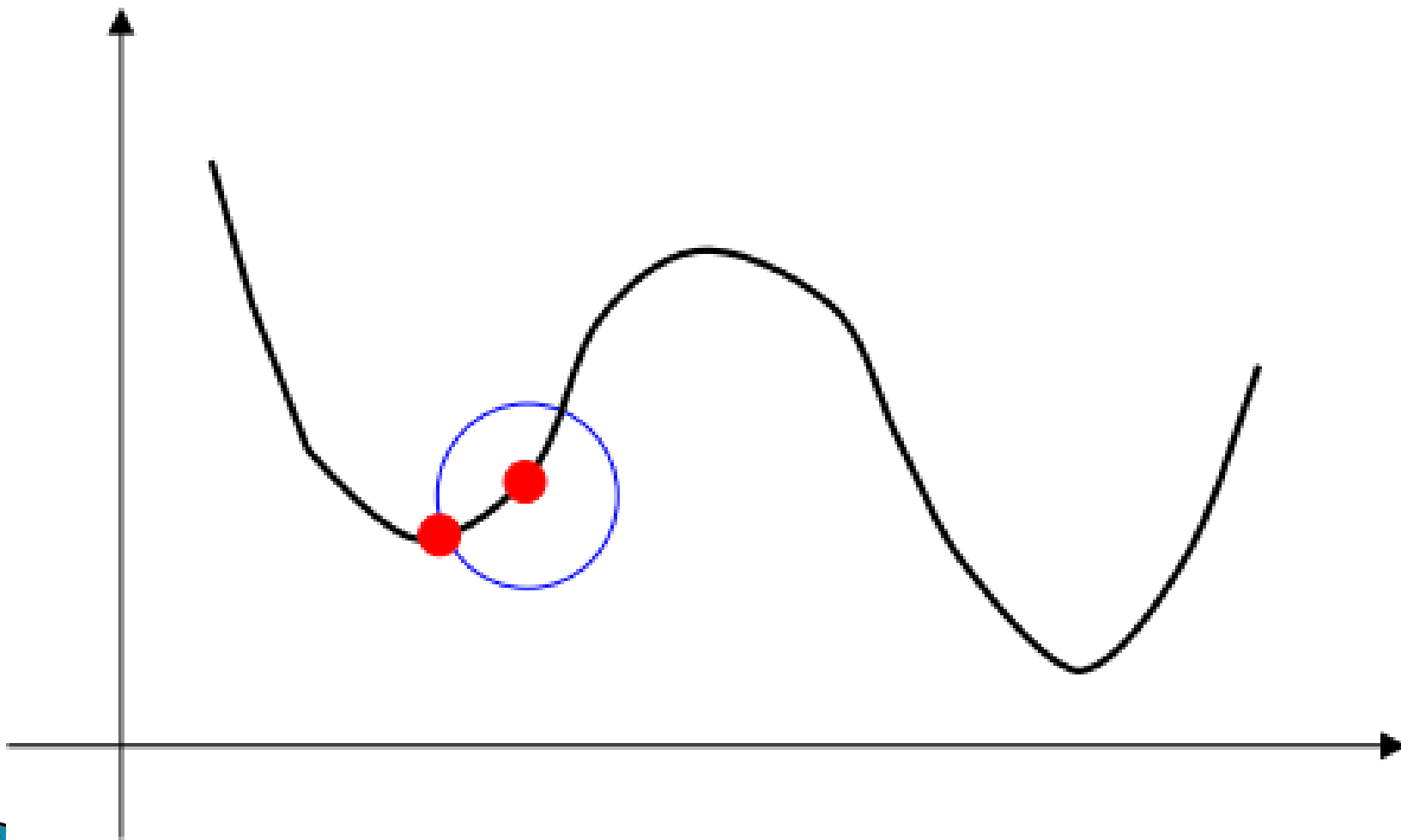
Busca



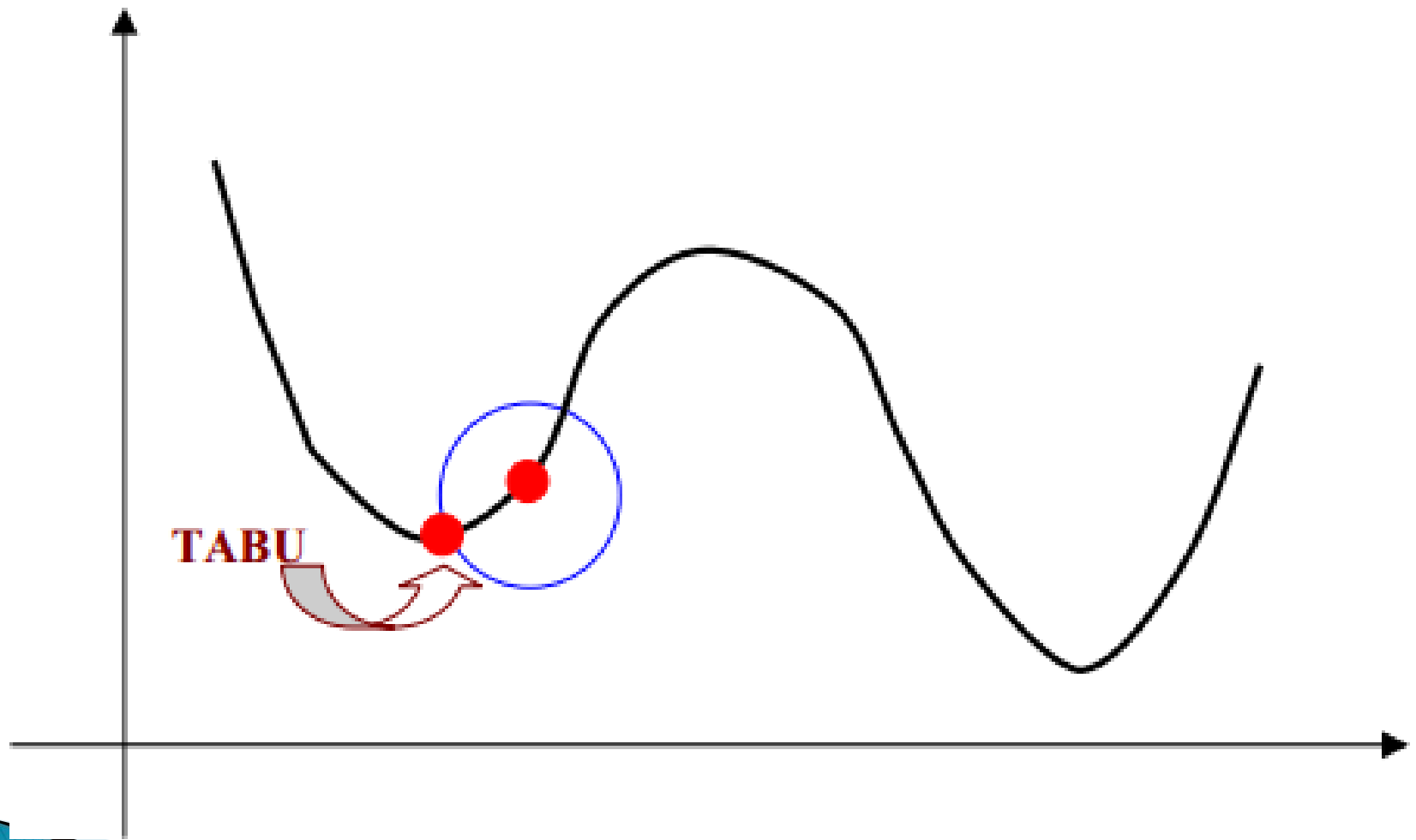
Busca



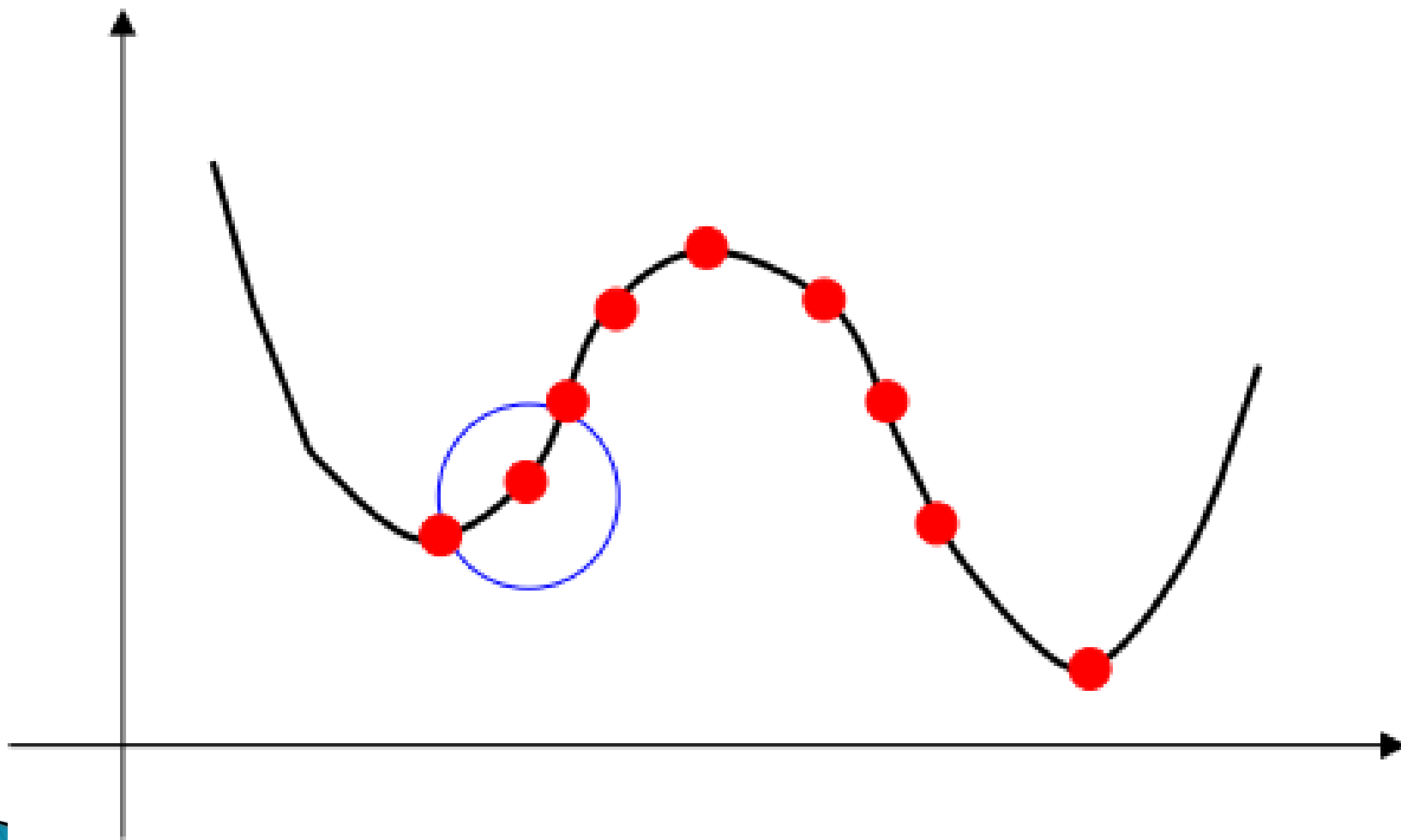
Busca



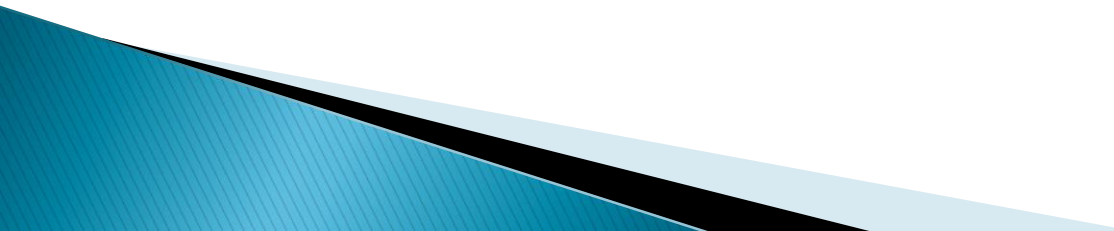
Busca



Busca



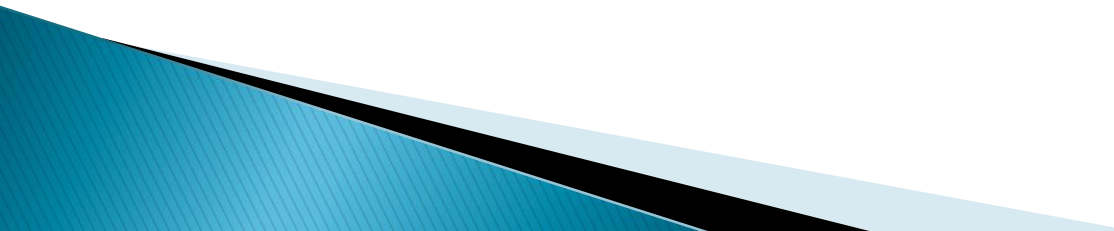
Aspiração

- ▶ Default
 - ▶ Por objetivo
 - ▶ Direção de busca
- 

Duração

- ▶ Fixa
 - Número de iterações
- ▶ Dinâmica
 - Número de iterações sem melhoria
 - Solução aceitável

Funcionamento

- ▶ Solução inicial
 - Aleatória
 - ▶ Explora subconjunto da vizinhança da solução corrente
 - ▶ Compara com melhor solução encontrada
 - ▶ Utilizar lista tabu
 - Evitar ciclos
 - escapar de mínimos locais
- 

Intensificação x Diversificação

▶ Diversificação

- Gerar soluções significativamente diferentes das melhores soluções encontradas
- Quando não existem movimentos de melhora

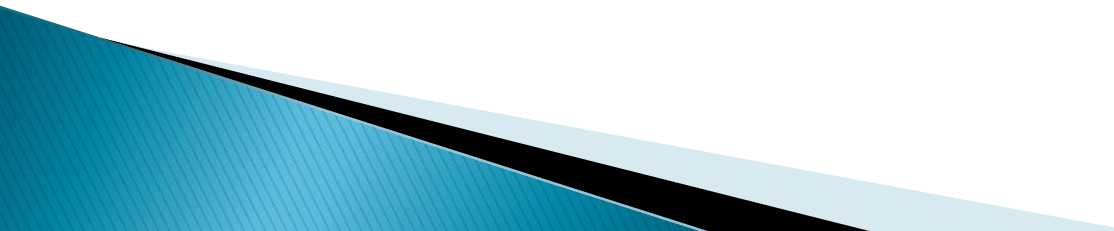
▶ Intensificação

- Explorar melhor a vizinhança de boas soluções
- Incorporar atributos das melhores solução encontradas

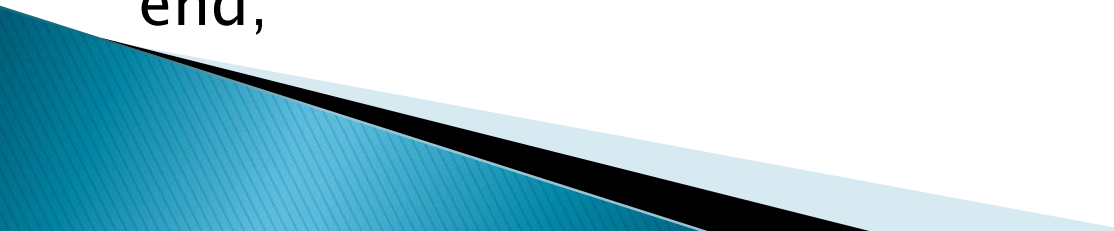
Path relinking

- ▶ Estratégia de intensificação
- ▶ Incorpora atributos de soluções elite
 - Favorece movimentos que levem a soluções–elite

Implementação

- ▶ Solução inicial: aleatória
 - ▶ Duração fixa: número de iterações
 - ▶ Aspiração: default
 - ▶ Vizinhaça: flip (1 bit aleatório)
- 

```
s = solução inicial;
sBest = s;
tabuList = [];
for i=1:nIteracoes
    melhorV = gerarVizinho(s);
    for j=2:nVizinhos
        v = gerarVizinho(s);
        if(!tabuList.verificaLista(v) and v.apt < melhorV.apt)
            melhorV = v;
        end;
    s = melhorV;
    if (melhorV.apt < sBest.apt)
        sBest = melhorV;
    tabuList.insert(melhorV);
    if (tabuList.size > tamMax)
        tabuList.remove();
    end;
end;
```



Problema

- ▶ Encontrar modelo para a determinação de celulose microcristalina no medicamento Escitalopram.
- ▶ Dados
 - Espectro de absorção de infra-vermelho próximo
 - 372 variáveis (comprimentos de onda)
 - 89 amostras de calibração
 - 67 amostras de validação
 - 72 amostras de predição

Comando para execução da implementação

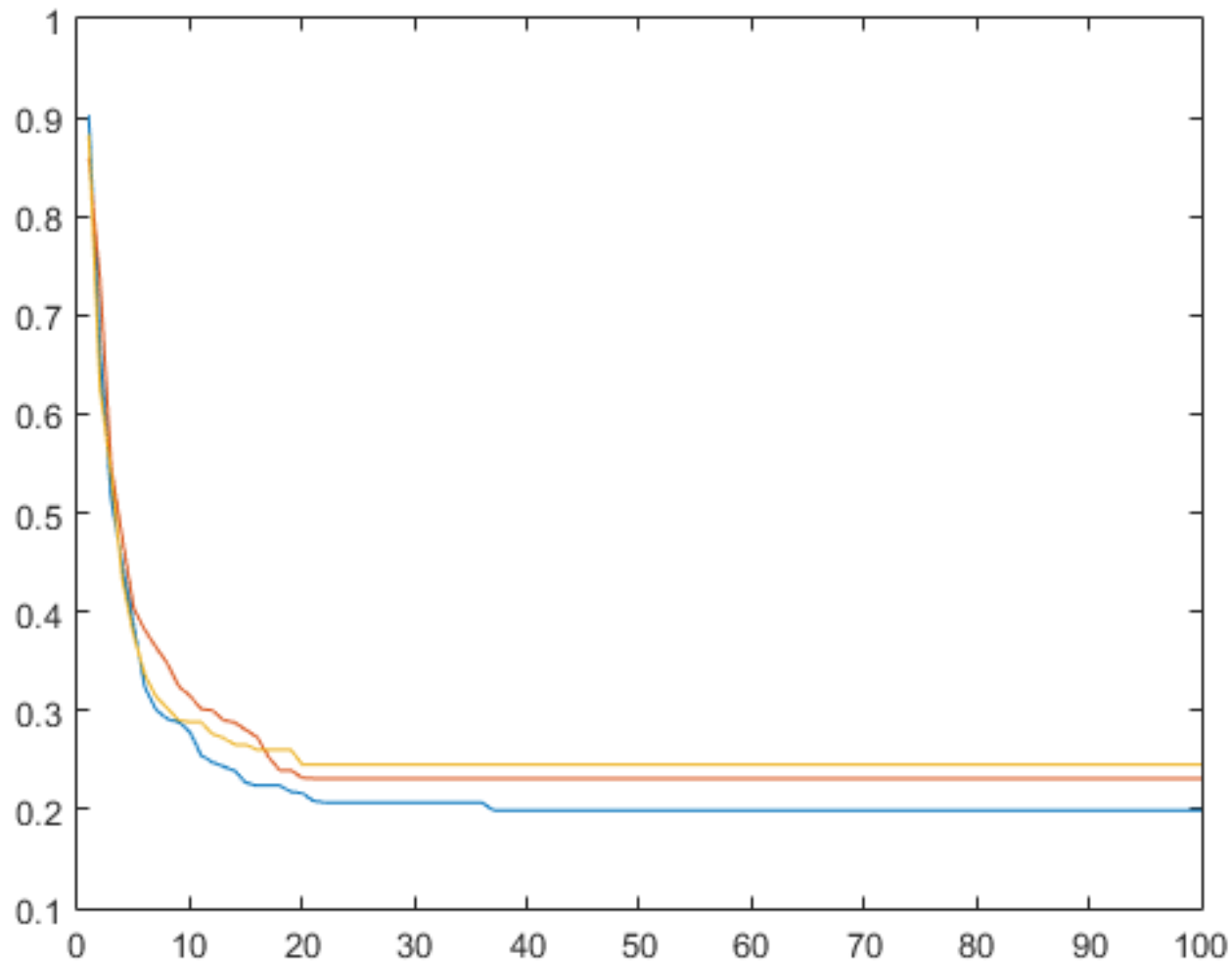
- ▶ `[sBest, vetFit] = tabu(nIteracoes, nVizinhos, tamTabu, dados)`
- ▶ Onde:
 - `sBest`: melhor solução encontrada
 - `vetFit`: matriz contendo 1 linha para cada iteração do algoritmo, armazenando a solução corrente na coluna 1 e a melhor solução na coluna 2 (para gerar os gráficos)
 - `nIteracoes`: nº máximo de iterações do algoritmo
 - `nVizinhos`: nº de vizinhos gerados em cada iteração
 - `tamTabu`: tamanho da lista tabu (implementado como uma fila limitada)
 - `Dados`: dados espectrais do problema (arquivo `dadosTabu.mat`)

Teste

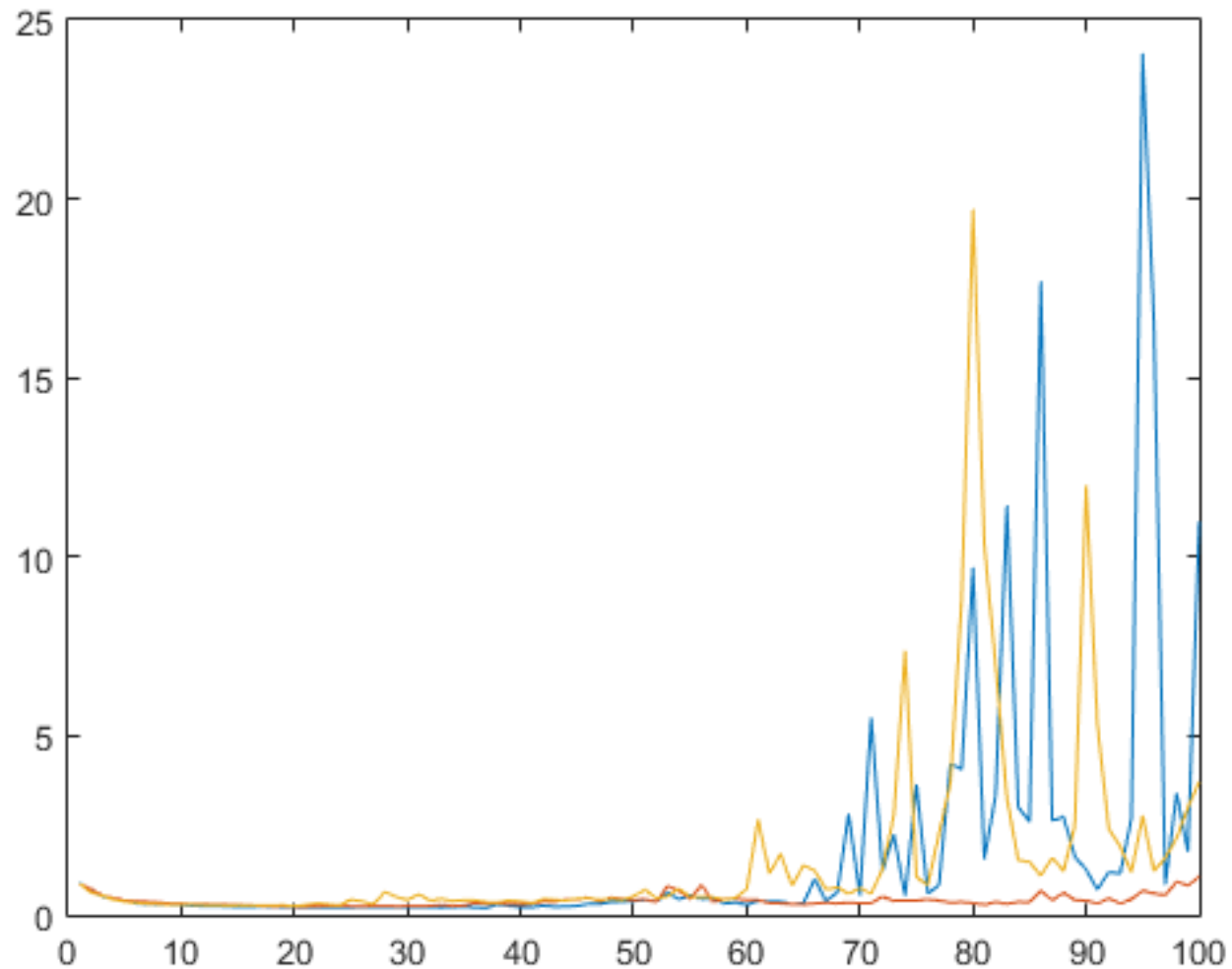
▶ Parâmetros utilizados

- nIterações: 100
- nVizinhos: 100
- tamTabu: 100

Resultados – evolução



Resultados – busca



Resultados

▶ Erros

- Validação: 0.1718
- Predição: 1.7464

Referências

- ▶ Furtado, J. C., Lorena, L. A. N. Otimização de leiaute utilizando busca tabu. GESTÃO & PRODUÇÃO v.4, n.1, p. 88–108, abr. 1997.
- ▶ AGUIAR, F. N., et al. Metaheurística busca tabu para o problema de coloração de grafos. XXXVII SBPO, p. 2497–2504, 2005.
- ▶ COELHO, L. S. Busca tabu aplicada à otimização de banco de capacitores em sistemas secundários de distribuição de energia elétrica. SBAI 2007.