HEURÍSTICAS 1

Marcus Ritt

INF 05010 – Otimização combinatória — *<2020-06-17 qua>*

Outline

1. Introdução

2. Heurísticas construtivas

3. Heurísticas de busca local

$\in P \rightarrow \exists$ alg. eficiente

$\in NP\text{-C.} \rightarrow P \neq NP$;
$\Rightarrow$ alg. exp.

→ Simplex (PL)
Dijkstra (Cam. mais curto)

- Algoritmos exatos
- Algoritmos de aproximação
- Heurísticas e meta-heurísticas
  - Avaliação experimental

Algoritmo eficiente (polinomial) com garantia de qualidade.

Sem garantia.

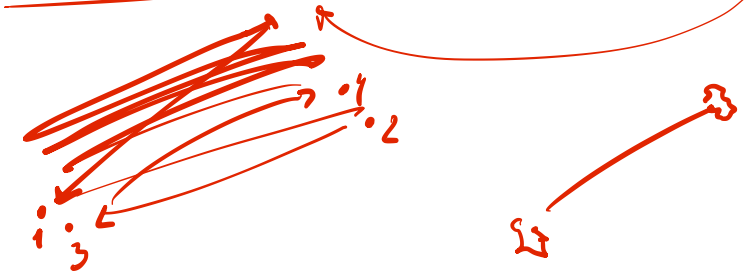
PCV $\in$ NP-C.

n

H_0 : return (1, 2, 3, 4, ..., n-1, n)

AA : retorne em tempo pol. uma solução em 5% da otimalidade

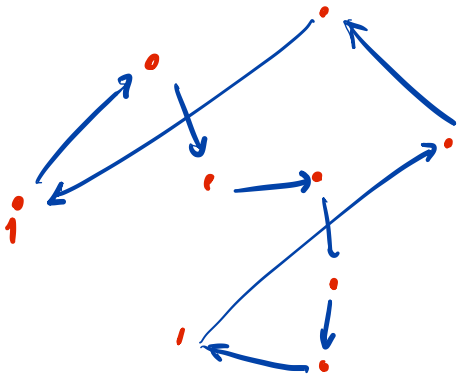
100 → 105, 104, 103 ..., 100



- Construção de soluções — Gulosos, Multi-start, GRASP
- Modificação de solução (busca local) — Simulated Annealing, Busca Tabu, VNS
- Recombinação de soluções — Algoritmo Genético

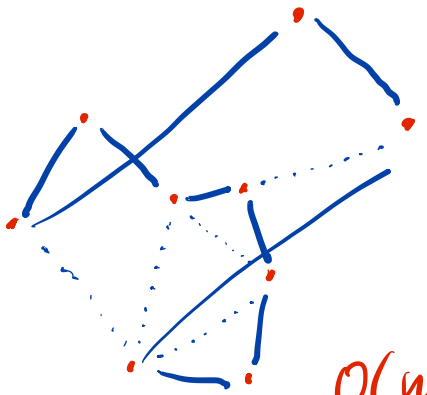
- Geralmente algoritmos gulosos
- Exemplo PCV: vizinho mais próximo, selecão gulosa no PCV

NN



$\Omega(n)$ $O(n^2)$

GR

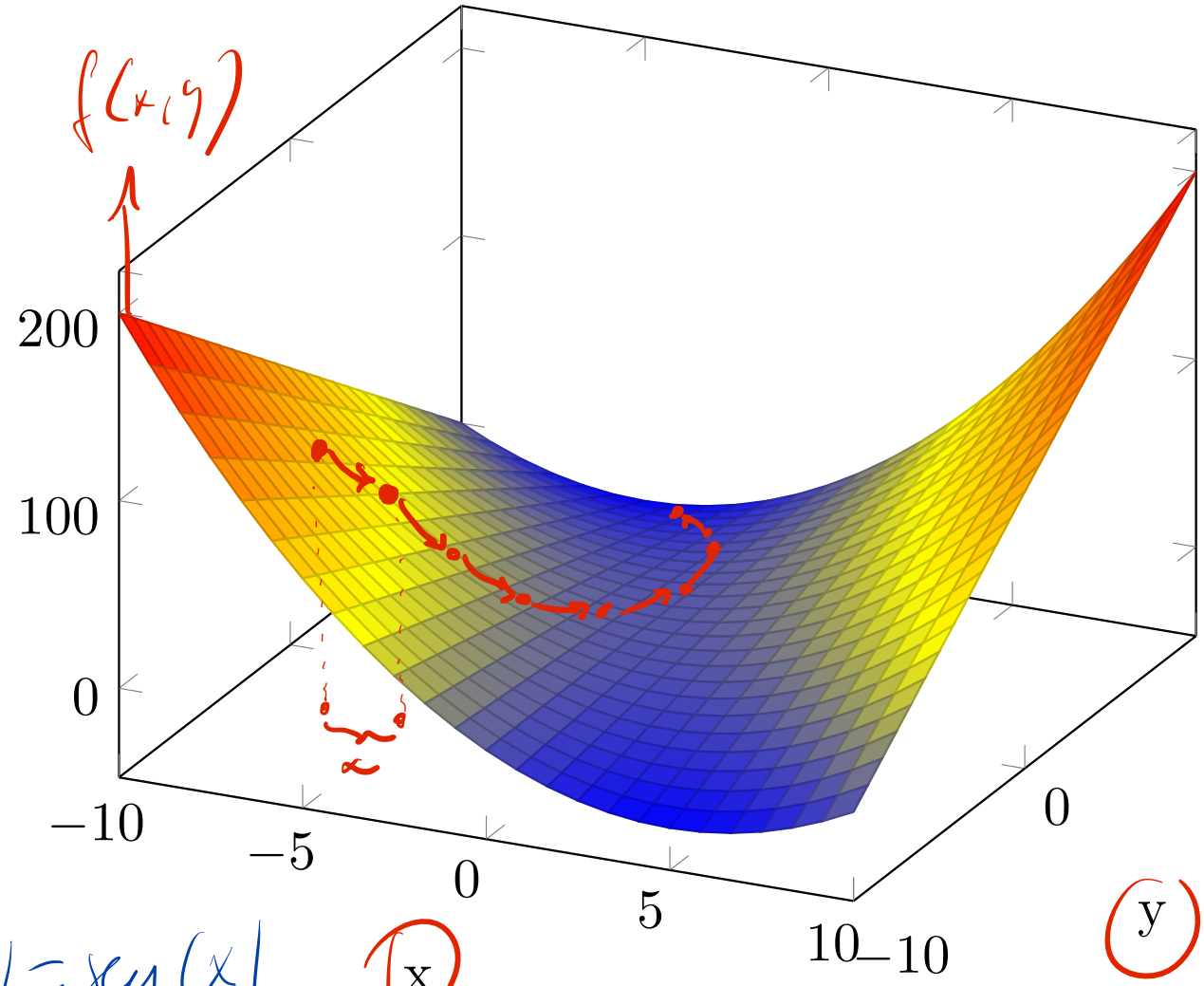


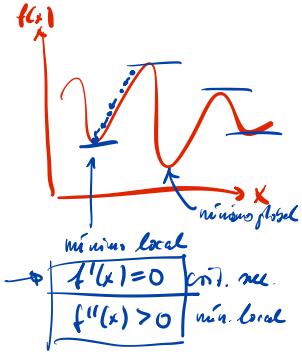
Sort; analise em ordem crescente; verificar lacos.

$O(n^2 \log n)$

$O(n \log n)$

$O(n \log n) + O(n \cdot \log n)$



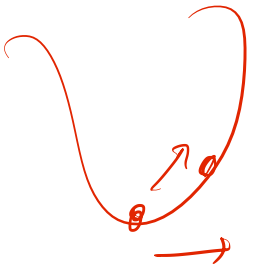


$f(x) = \sin(x)$

$f'(x) = \cos(x) = 0$

Busca local contínua

```
BuscaLocal(s_0, α) =
    s := s_0
    while ∇f(s) ≠ 0 do
        s' := s − α∇f(s)
        if f(s') < f(s) then
            s := s'
        else
            diminui α
        end if
    end while
    return s
```

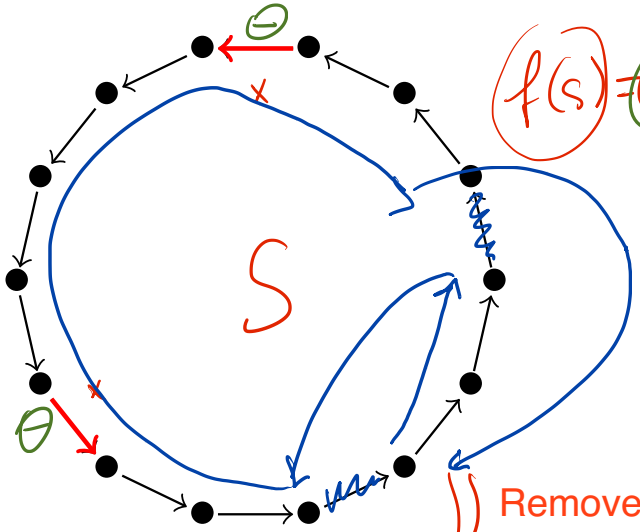


- Não existe vizinhança natural.
- Temos que *definir uma vizinhança*.
  - Ideia: solução s' obtida por uma pequena modificação na solução s.
  - Notação: O conjunto de soluções vizinhas de $x \in S$ é $\mathcal{N}(x)$.
  - Geralmente: simétrico e conexo.
- Importante: método eficiente para avaliar a função objetivo de vizinhos.

movimento

- ~~Qual~~ o vizinho de um ciclo Hamiltoniano?

n cidades ⇒ n arcos



$f(s) = 300$

Quantos vizinhos? $\binom{n}{2} - n = \Theta(n^2)$.

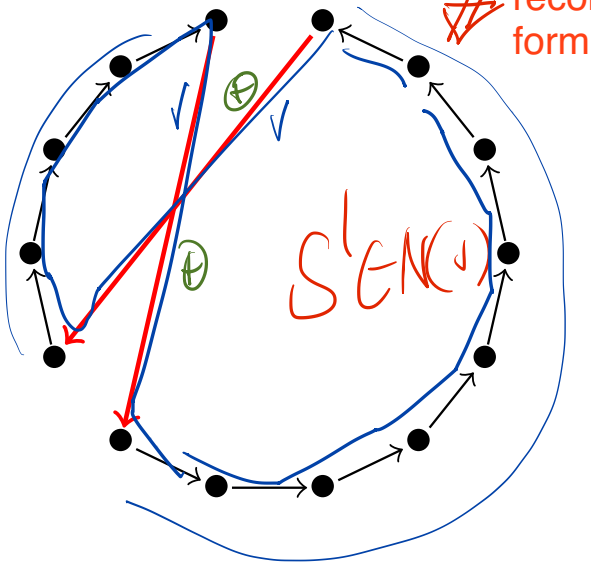
x custo de calcular o valor da solução: ~~$O(n)$~~ $O(1) = \Theta(n^2)$

↳ $\Theta(n^3)$

Remove 2 arcos não adjacentes; reconecta os segmentos de outra forma.

Movimento

2-exchange



$f(s') = ?$

Complexidade?

- Buscar um vizinho melhor na vizinhança: "first improvement"

primeira melhora

```
BuscaLocal(s_0) =
  s := s_0
  repeat
    seleciona s' ∈ N(s) não visto ainda
    if f(s') < f(s) then s := s'
  until todas soluções em N(s) vistas
  return s
```

melhor melhora

- Buscar o melhor vizinho na vizinhança: "best improvement"

```
BuscaLocal(s_0) =
  s := s_0
  while true
    s' := argmin_y{f(y) | y ∈ N(s)}
    if f(s') < f(s) then s := s' else break
  end while
  return s
```