ALGORITMOS AVANÇADOS

Prof. Michael da Costa Móra

Algoritmos Gulosos

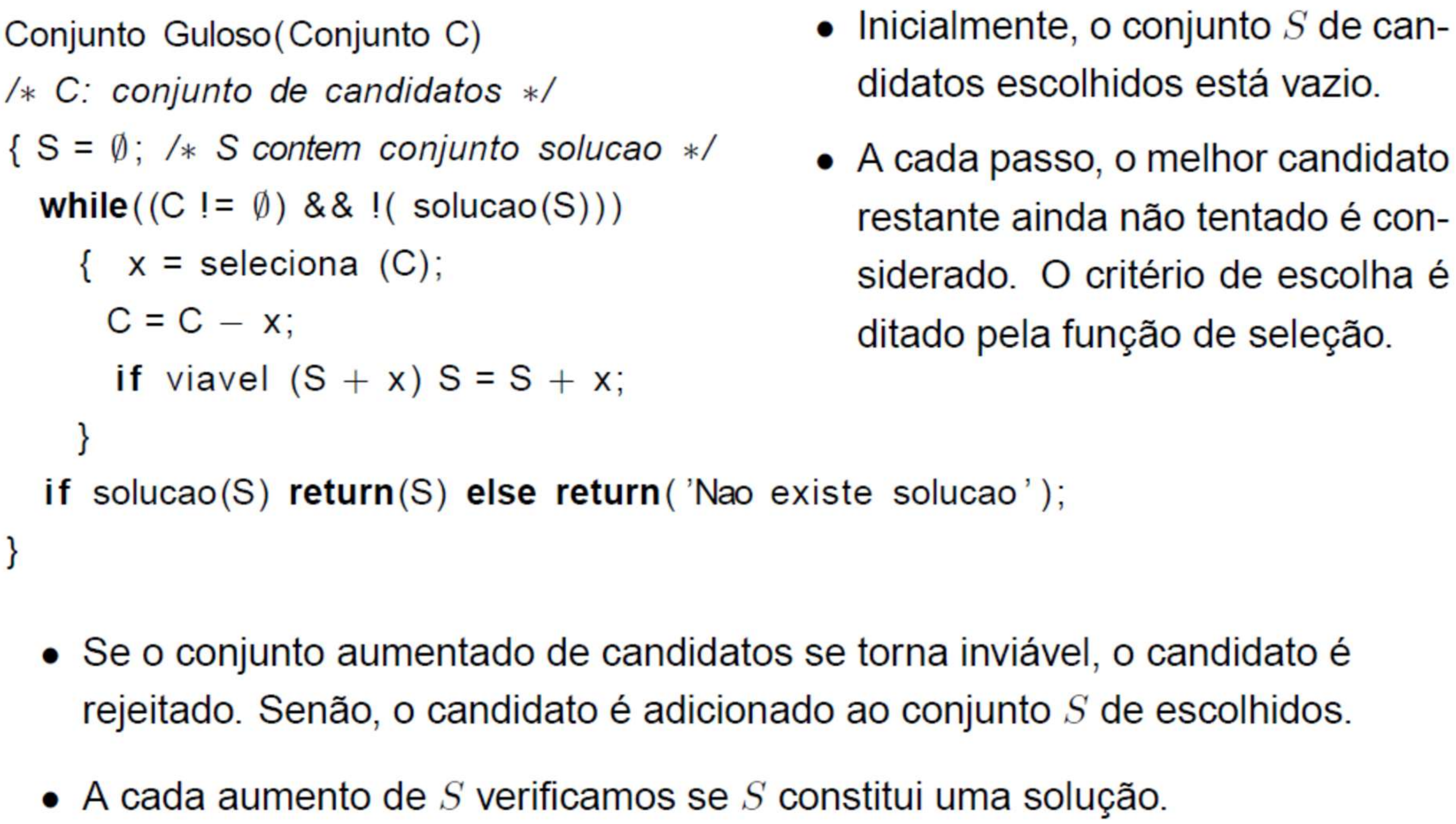
Algoritmos Gulosos

- Resolve problemas de otimização.
- Ex: encontrar o menor caminho entre dois vértices de um grafo.
  - Escolhe a aresta que parece mais promissora em qualquer instante;
  - Independente do que possa acontecer, nunca reconsidera a decisão.
- Não necessita avaliar alternativas, ou usar procedimentos sofisticados para desfazer decisões tomadas previamente.
- Problema geral: dado um conjunto C, determine um subconjunto $S \subseteq C$ tal que:
  - S satisfaz uma dada propriedade P, e
  - S é mínimo (ou máximo) em relação a algum critério α.
- O **algoritmo guloso** consiste em um processo iterativo em que S é construído adicionando-se ao mesmo elementos de C um a um.

Algoritmos Gulosos

- Para construir a solução ótima existe um conjunto ou lista de candidatos.
- São acumulados um conjunto de candidatos considerados e escolhidos, e o outro de candidatos considerados e rejeitados.
- Existe função que verifica se um conjunto particular de candidatos produz uma *solução* (sem considerar otimalidade no momento).
- Outra função verifica se um conjunto de candidatos é *viável* (também sem preocupar com a otimalidade).
- Uma *função de seleção* indica a qualquer momento quais dos candidatos restantes é o mais promissor.
- Uma *função objetivo* fornece o valor da solução encontrada, como o comprimento do caminho construído (não aparece de forma explicita no algoritmo guloso).

Algoritmos Gulosos



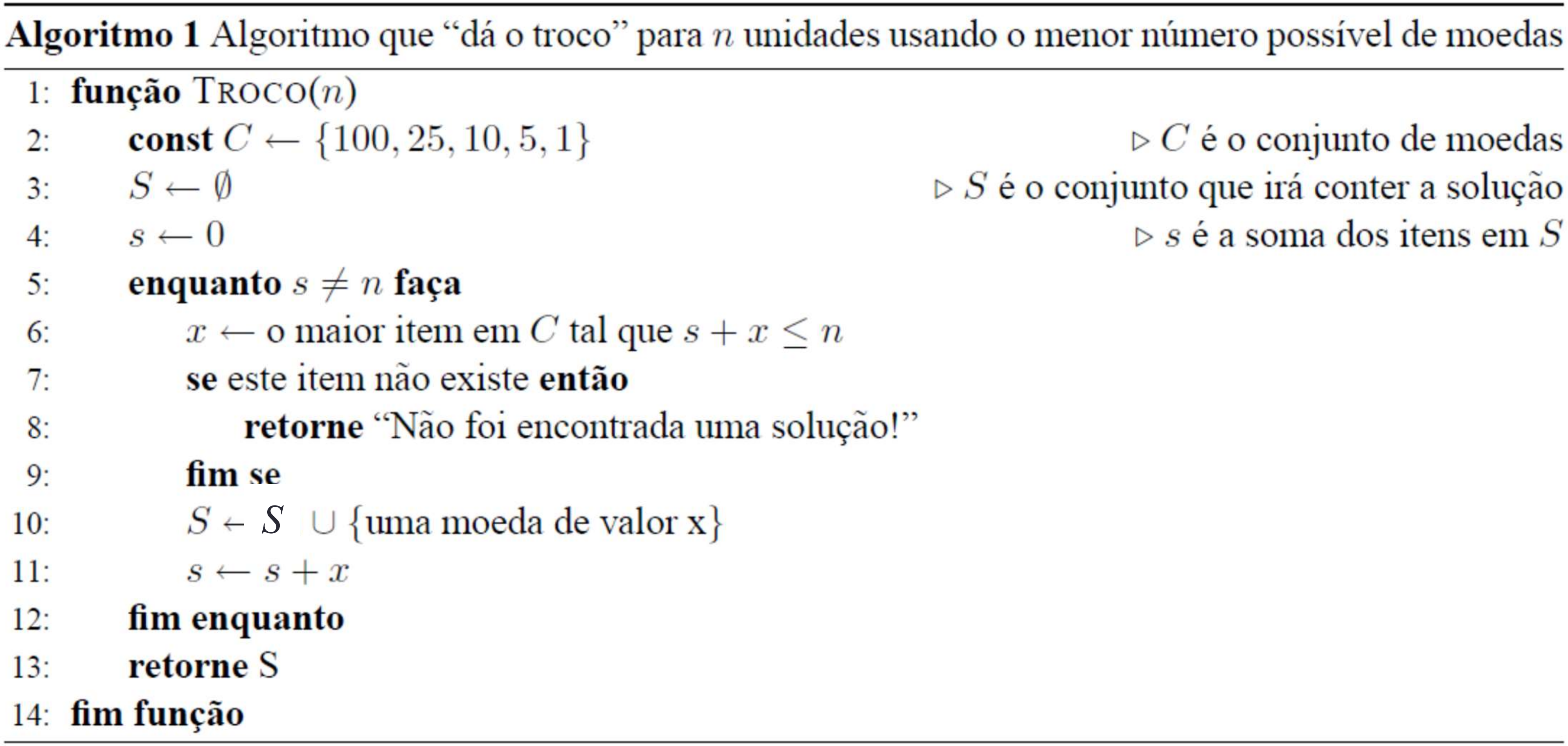
```
Conjunto Guloso(Conjunto C)
/* C: conjunto de candidatos */
{ S = ∅; /* S contem conjunto solucao */
  while((C != ∅) && !( solucao(S)))
    {  x = seleciona (C);
      C = C − x;
      if viavel (S + x) S = S + x;
    }
  if solucao(S) return(S) else return('Nao existe solucao');
}
```

- Inicialmente, o conjunto S de candidatos escolhidos está vazio.
- A cada passo, o melhor candidato restante ainda não tentado é considerado. O critério de escolha é ditado pela função de seleção.

- Se o conjunto aumentado de candidatos se torna inviável, o candidato é rejeitado. Senão, o candidato é adicionado ao conjunto S de escolhidos.
- A cada aumento de S verificamos se S constitui uma solução.

Algoritmos Gulosos

- Quando funciona corretamente, a primeira solução encontrada é sempre ótima.
- A função de seleção é geralmente relacionada com a função objetivo.
- Se o objetivo é:
  - maximizar ⇒ provavelmente escolherá o candidato restante que proporcione o maior ganho individual.
  - minimizar ⇒ então será escolhido o candidato restante de menor custo.
- O algoritmo nunca muda de idéia:
  - Uma vez que um candidato é escolhido e adicionado à solução ele lá permanece para sempre.
  - Uma vez que um candidato é excluído do conjunto solução, ele nunca mais é reconsiderado.



Algoritmo 1 Algoritmo que “dá o troco” para n unidades usando o menor número possível de moedas

```
função TROCO($n$)
    const $C \leftarrow \{100, 25, 10, 5, 1\}$                ▷ $C$ é o conjunto de moedas
    $S \leftarrow \emptyset$                ▷ $S$ é o conjunto que irá conter a solução
    $s \leftarrow 0$                ▷ $s$ é a soma dos itens em $S$
    enquanto $s \neq n$ faça
        $x \leftarrow$ o maior item em $C$ tal que $s + x \leq n$
        se este item não existe então
            retorne “Não foi encontrada uma solução!”
        fim se
        $S \leftarrow S \cup \{$uma moeda de valor x$\}$
        $s \leftarrow s + x$
    fim enquanto
    retorne S
fim função
```

Quando o Algortimo Guloso do Troco Funciona

- Se o conjunto de moedas não for vazio
- Se cada moeda for múltipla de cada uma das suas moedas menores.