CI1238 - Otimização

Aula 11 - Backtracking (Parte 1)

Professor Murilo V. G. da Silva

Departamento de Informática

Universidade Federal do Paraná

2026 / Primeiro Semestre

Backtracking

Objetivo: Explorar recursivamente o espaço de soluções de um problema

- Ideia chave: Árvore de chamadas recursivas do algoritmo
  - Soluções nas folhas ou em todos os nós (depende do problema)
  - Nós da árvore correspondem a elementos do espaço de soluções
  - A árvore completa contém contém todo espaço de soluções
  - Importante: Algoritmo de backtracking procura "podar" ramos da árvore de recursão caso identifique que eles não contenham a solução.
- Livro para este tópico: [KS99]

Antes de entrarmos a fundo no Backtracking:

- "Aquecimento": vamos resolver o problema da MOCHILA de maneira recursiva.

Mochila: solução recursiva

Instância: (p, w, n, M), onde p é um vetor de n valores reais, w é um vetor de n pesos reais e M um número real (capacidade da mochila).

Resposta: lista $[x_0, ..., x_{n-1}]$ tal que $x_i \in \{0, 1\}$
Sujeito a $\sum_{i=0}^{n-1} x_i w_i \leq M$ e tal que $\sum_{i=0}^{n-1} x_i v_i$ seja máximo.

Algorithm 4.1: KNAPSACK1 (ℓ)

global $X, OptP, OptX$
if $\ell = n$

then
- **if** $\sum_{i=0}^{n-1} w_i x_i \leq M$
  - **then**
    - $CurP \leftarrow \sum_{i=0}^{n-1} p_i x_i$
    - **if** $CurP > OptP$
      - **then**
        - $OptP \leftarrow CurP$
        - $OptX \leftarrow [x_0, \ldots, x_{n-1}]$

else
- $x_\ell \leftarrow 1$
- KNAPSACK1$(\ell + 1)$
- $x_\ell \leftarrow 0$
- KNAPSACK1$(\ell + 1)$

Variáveis globais no início: $OptP = 0$ $\quad OptX = [\ \ ]$
$X = [x_0, ..., x_{n-1}]$ onde cada x_i começa "não instanciado"

Chamada inicial: $l = 0$ $\quad$ A instância (p, w, n, M) do problema também é global

Mochila: solução recursiva

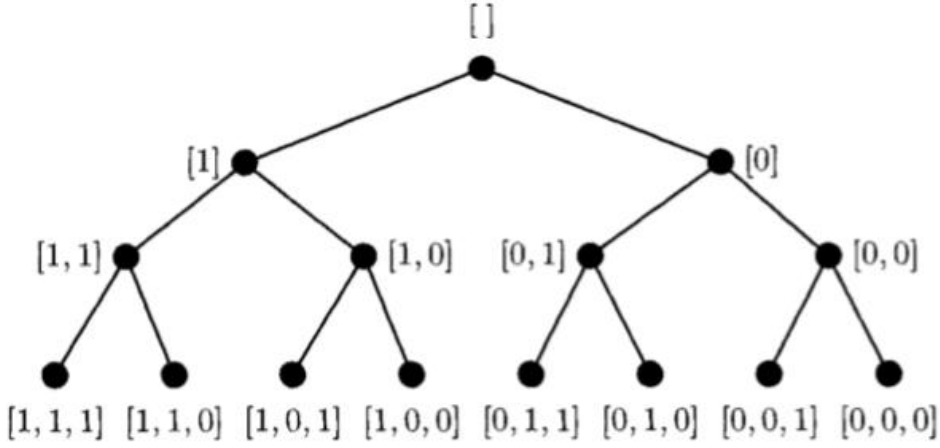
Algorithm 4.1: KNAPSACK1 (ℓ)

global X, $OptP$, $OptX$

if $\ell = n$

then
- **if** $\sum_{i=0}^{n-1} w_i x_i \leq M$
  - **then**
    - $CurP \leftarrow \sum_{i=0}^{n-1} p_i x_i$
    - **if** $CurP > OptP$
      - **then**
        - $OptP \leftarrow CurP$
        - $OptX \leftarrow [x_0, \ldots, x_{n-1}]$

else
- $x_\ell \leftarrow 1$
- KNAPSACK1$(\ell + 1)$
- $x_\ell \leftarrow 0$
- KNAPSACK1$(\ell + 1)$



Árvore de soluções para $n = 3$ (árvore do espaço de busca)

Precisamos expandir toda árvore? $\rightarrow$ Algoritmo de backtracking: poda ramos inviáveis

Algoritmo de Backtracking para MOCHILA

Algorithm 4.3: KNAPSACK2 $(\ell, CurW)$

global $X, OptX, OptP, C_\ell \quad (\ell = 0, 1, \ldots)$

if $\ell = n$

then $\begin{cases} \textbf{if } \sum_{i=0}^{n-1} p_i x_i > OptP \\ \quad \textbf{then} \begin{cases} OptP \leftarrow \sum_{i=0}^{n-1} p_i x_i \\ OptX \leftarrow [x_0, \ldots, x_{n-1}] \end{cases} \end{cases}$

if $\ell = n$

then $C_\ell \leftarrow \emptyset$

else $\begin{cases} \textbf{if } CurW + w_\ell \leq M \\ \quad \textbf{then } C_\ell \leftarrow \{1, 0\} \\ \quad \textbf{else } C_\ell \leftarrow \{0\} \end{cases}$

for each $x \in C_\ell$

do $\begin{cases} x_\ell \leftarrow x \\ \text{KNAPSACK2}(\ell + 1, CurW + w_\ell x_\ell) \end{cases}$

A primeira chamada do algoritmo é feita com $l = CurW = 0$.

Var. global C_l não instanciada p/ $l = 0, 1, 2, \ldots$

Ponto importante: Soluções viáveis parciais são escolhas parciais de itens que ainda estão dentro da capacidade da mochila.

Soluções Parciais e Conjunto de escolhas

Soluções para o caso de MOCHILA: n-tuplas $[x_0, x_1, ..., x_n]$, com $x_i \in \{0, 1\}$

Generalizando: Espaço de soluções para problemas em geral:

k-tupla $[x_0, x_1, ..., x_k]$, $k \leq n$. $\quad x_i \in P_i \quad$ (P_i são as possibilidades p/ x_i)

x_i não precisam ser bits como em MOCHILA

Seja $X = [x_0, x_1, ..., x_k]$ uma solução viável para um problema de otimização.

- Algoritmo vai construir X passo a passo
- Solução viável parcial: uma lista $[x_0, x_1, ..., x_l]$, $l \leq k$

  Notação simplificada: $[x_0, x_1, ...]$
  - $[x_0, x_1, ..., x_l]$ não necessariamente é viável
  - *viável parcial* significa que de ser *estendida* p/ alguma sol. viável X

- A partir de $[x_0, ..., x_l]$, $l < k$ e das restrições do problema em particular:
  - x_{l+1} pode assumir apenas um conjunto restrito de valores $C_{l+1} \subseteq P_{l+1}$

    O conjunto C_{l+1} é o conjunto de escolhas para x_{l+1},

    (também conhecido como "possibilidades viáveis")

Algoritmo Genérico de Backtracking

Algorithm 4.2: BACKTRACK (ℓ)

global $X, C_\ell \quad (\ell = 0, 1, \ldots)$
comment: $X = [x_0, x_1, \ldots]$
if $[x_0, x_1, \ldots, x_{\ell-1}]$ is a feasible solution
  then process it
Compute C_ℓ
for each $x \in C_\ell$
  do $\begin{cases} x_\ell \leftarrow x \\ \text{BACKTRACK}(\ell + 1) \end{cases}$

No algoritmo acima, o significado de "processa" vai depender do problema específico.

- A computação de C_l é chamado de poda da árvore
  Se $y \in P_l \setminus C_l$, nós da árvore **NÃO** tem $[x_0, ..., x_{l-1}, y]$ como sol. parcial
  i.e. y, não é escolha viável