CI1238 - Otimização

Aula 15 - Programação Dinâmica (Parte 2)

Professor Murilo V. G. da Silva

Departamento de Informática

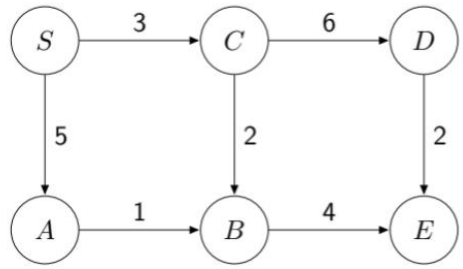
Universidade Federal do Paraná

19/11/2024

PD em Grafos Direcionados Acíclicos Ponderados

Grafo direcionado acíclico (GDA): Não contém *ciclos direcionados*.

Exemplo: Um GDA G ponderado com $V(G) = \{S, A, B, C, D, E\}$.



Caminhos mínimos em GDA (caminhos-gda)

Instância: (G, v), sendo G um GDA ponderado com n vértices e $v \in V(G)$.
Solução: Para cada $u \in V(G)$, um caminho direcionado de v até u de custo mínimo

No exemplo acima, os caminhos mínimos para a instância (G, S) são:

$\rightarrow$ $[S]$, $[S, A]$, $[S, C]$, $[S, C, B]$, $[S, C, D]$, $[S, C, B, E]$

Distâncias em GDA (distâncias-gda)

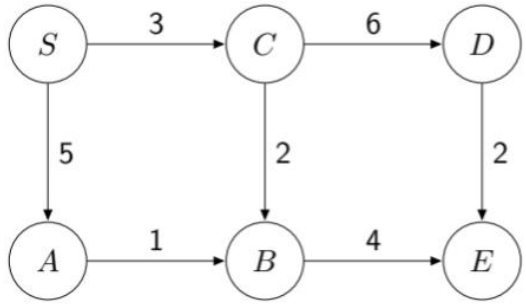
Instância: (G, v), sendo G um GDA ponderado com n vértices e $v \in V(G)$.
Solução: Um vetor d tal que $\forall u \in V(G)$, $d[u]$ a distância de v à u.

Solução para o exemplo (G, S) é um vetor d tal que:

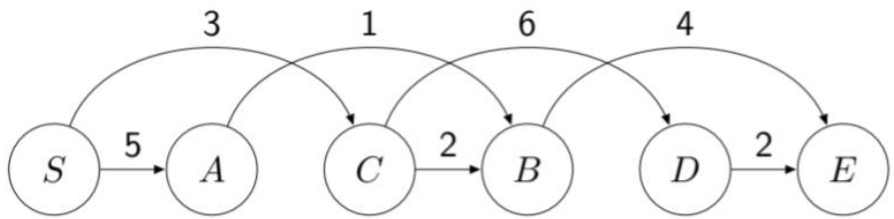
$\rightarrow$ $d[S] = 0 \quad d[A] = 5 \quad d[C] = 3 \quad d[B] = 5 \quad d[D] = 9 \quad d[E] = 9$

PD em Grafos Direcionados Acíclicos Ponderados

- Considere o GDA G do slide anterior.



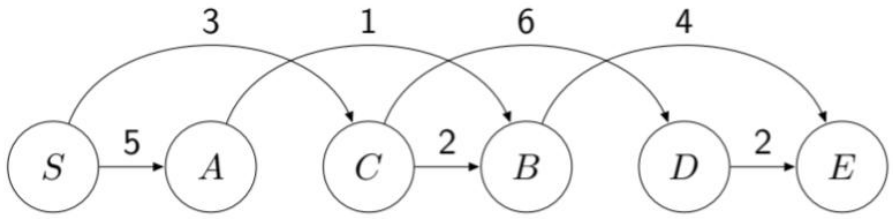
- Propriedade importante de GDAs: Possuem uma Ordenação Topológica:



  - Uma Ordenação $[v_1, v_2, ..., v_n]$ de $V(G)$ tal que $\forall v_i v_j \in E(G)$, $i < j$
    (obs: existe mais de uma ordenação topológica possível)
  - Algoritmo de ordenação usando uma Busca em Profundidade (DFS):
    - Sempre que a DFS "retorna" da recursão, insira vértice numa pilha
    - Ao final, a pilha contém os vértices ordenados do topo para o fundo
      i.e., o topo da pilha contém v_1 e o fundo v_n

PD em Grafos Direcionados Acíclicos Ponderados

Dado (G, v), considere uma ordenação topológica de G:



Distâncias em GDA (distâncias-gda)

Instância: (G, v), sendo G um GDA ponderado com n vértices e $v \in V(G)$.
Solução: Um vetor d tal que $\forall u \in V(G)$, $d[u]$ a distância de v à u.

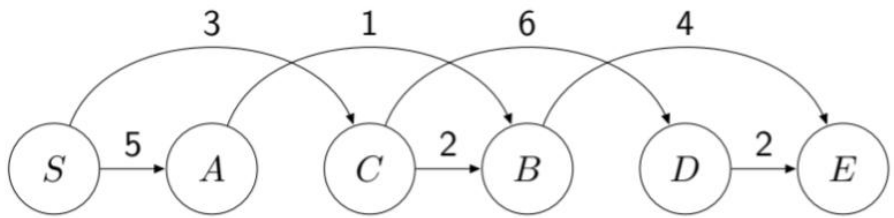
Distâncias em GDA (distâncias-topgda)

Instância: Um GDA G ordenado topologicamente com primeiro vértice s.
Solução: Um vetor d tal que $\forall u \in V(G)$, $d[u]$ a distância de v à u.

PD em grafo ordenado topologicamente

Vamos apresentar um algoritmo para DISTÂNCIAS-TOPGDA:

- Seja G com n vértices ordenado topologicamente com primeiro vértice s:



Def.: $pred(u) = \{x \mid xu \in E(G)\}$ exemplo acima: $pred(B) = \{A, C\}$

- Algoritmo de Programação Dinâmica abaixo:

CaminhoMinimoGDA(G, s)

1: $d[s] = 0$
2: **for** cada $u \in V(G)$ em ordem topológica **do**
3: $\quad d[u] = \min_{x \in pred(u)} \{d(x) + w(xv)\}$
4: **end for**

Quais são os subproblemas sendo resolvidos pelo algoritmo?

- Problema da distâncias de s até v_i, para $i = 1, 2, ..., n$
- **Exercício**: Adapte o algoritmo para resolver DISTÂNCIAS-GDA e CAMINHOS-GDA.
- E se quisermos os caminhos máximos? Basta trocar *min* por *max*.

PD - esboço de algoritmos para outros exemplos

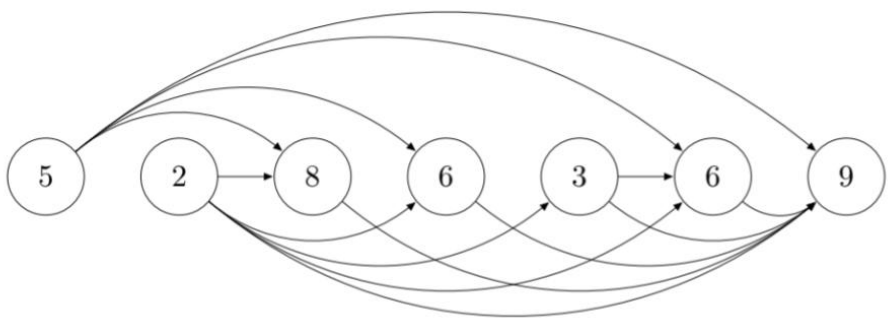
Problema da **Subsequência Crescente Máxima:**

Dada sequência de números inteiros $[a_1, a_2, ..., a_n]$,
Encontrar lista $[a_{i_1}, a_{i_2}, ..., a_{i_k}]$ tal que respeitando $i_j < i_{j+1}$, $j = 1, ..., k-1$

Exemplo: A solução da instância $[5, 2, 8, 6, 3, 6, 9]$ é $[2, 3, 6, 9]$

Algoritmo de PD: Dado $[5, 2, 8, 6, 3, 6, 9]$, crie um vértices para cada número:

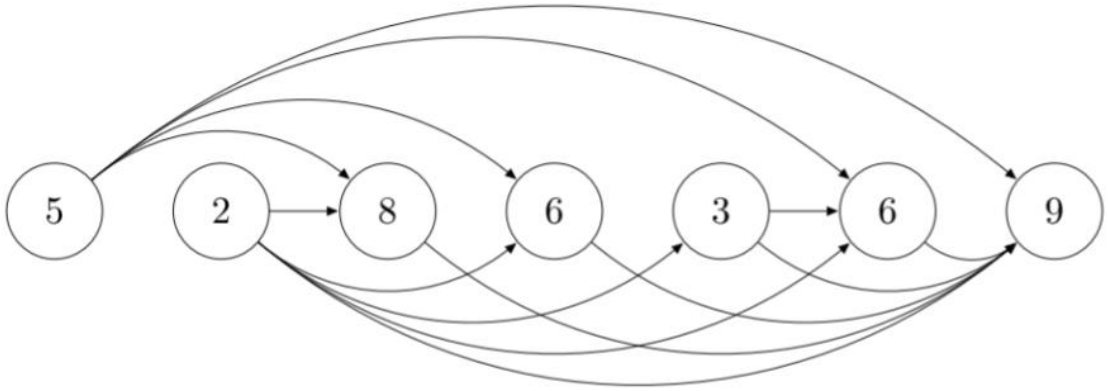
Crie um GDA com todos os arcos possíveis $a_i a_j$, $i < j$ e peso 1 nas arestas:



- Resolva o problema do caminho máximo no grafo para todos so vértices.

PD - esboço de algoritmos para outros exemplos

Problema da **Subsequência Crescente Máxima:**



LongestSubseq (G, L)

1: L$[s] = 1$ /* s é o primeiro vértice da ordenação */
2: **for all** $v \in V(G) \setminus \{s\}$ na ordem dada (topológica) **do**
3: L$[v] = \max_{u \in pred(v)} \{L(u) + 1\}$
4: **end for**

PD - esboço de algoritmos para outros exemplos

Distância de Edição (também chamado de "alinhamento de sequências"):

Qual a "distância de edição" entre as as strings SILVA e SILVER?
Resp: Distância 2.

Na segunda string faça:

- Remova R
- Troque E por A.

Ou na primeira string faça:

- Troque A por E.
- Insira R

Operações: remover, inserir, editar (trocar)

- Padrão que vamos seguir: Editar a primeira string para obter a segunda

PD - esboço de algoritmos para outros exemplos

Distância de Edição (também chamado de "alinhamento de sequências"):

Qual a "distância de edição" entre as as strings ACCTG e CACGTG?

Resp: Distância 2.

PD - esboço de algoritmos para outros exemplos

Distância de Edição (também chamado de "alinhamento de sequências"):

	␣	A	C	C	T	G
␣						
C						
A						
C						
G						
T						
G						

PD - esboço de algoritmos para outros exemplos

Distância de Edição (também chamado de "alinhamento de sequências"):

	⊔	A	C	C	T	G
⊔	0	1	2	3	4	5
C	1	1	1	2	3	4
A	2	1	2	2	3	4
C	3	2	1	2	3	4
G	4	3	2	2	3	3
T	5	4	3	3	2	3
G	6	5	4	4	3	2

$$\begin{aligned}\mathsf{ED}[i,j] = \min\{&\mathsf{ED}[i-1,j]+1,\\ &\mathsf{ED}[i,j-1]+1,\\ &\mathsf{ED}[i-1,j-1]+\ \mathsf{diff}(x[i],y[j])\}\end{aligned}$$

PD - esboço de algoritmos para outros exemplos

Mochila (com repetição e capacidades inteiras):

Item	Peso	Valor
1	6	\$30
2	3	\$14
3	4	\$16
4	2	\$19

Capacidade: $W = 9$

Ideia: *K*[*w*] =
"Maior valor p/ capacidade *w*"

MochilaR (K)

1: $K[0] = 0$
2: **for** $w = 1$ até W **do**
3: $\quad K[v] = \max_{\text{item } i}\{K[w - w_i] + v_i\}$
4: **end for**

obs: por questões de simplicidade, o código ignora testes para os limites do vetor

PD - esboço de algoritmos para outros exemplos

Mochila (capacidades inteiras):

Item	Peso	Valor
1	6	\$30
2	3	\$14
3	4	\$16
4	2	\$19

Capacidade: $W = 9$

Ideia: $K[w, i] =$
"Maior valor p/ capacidade w usando apenas itens 1,2,...,i"

PD - esboço de algoritmos para outros exemplos

Mochila (capacidades inteiras):

Ideia: $K[w, i] =$
"Maior valor pesando w usando itens 1,2,...,i"

Initialize all $K(0, j) = 0$ and all $K(w, 0) = 0$
for $j = 1$ to n:
    for $w = 1$ to W:
        if $w_j > w$: $K(w, j) = K(w, j - 1)$
        else: $K(w, j) = \max\{K(w, j - 1), K(w - w_j, j - 1) + v_j\}$
return $K(W, n)$