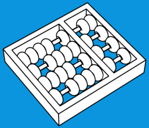
Ordenação topológica

MC558 - Projeto e Análise de Algoritmos II

06

08/24



Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP

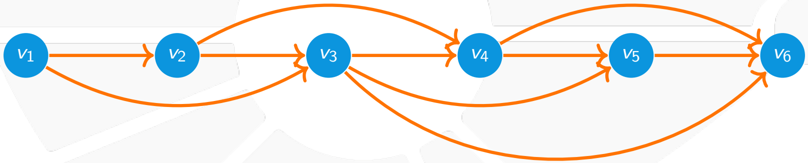
"Um ciclo foi detectado, portanto, não é possível uma ordenação topológica."

Mensagem de erro do ORM

Ordenação Topológica

Definição

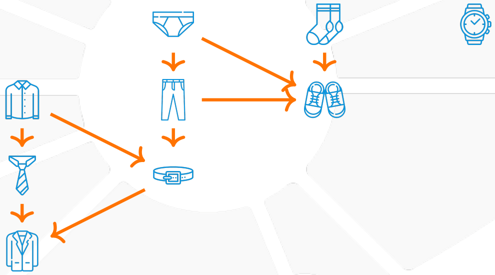
Uma **ORDENAÇÃO TOPOLÓGICA** de um grafo direcionado é um arranjo dos vértices $\mathbf{v_1, v_2, \ldots, v_n}$ tal que se $\mathbf{(v_i, v_j)}$ é uma aresta do grafo, então $i < j$.



Exemplo de aplicação

Representando dependências:

- Um grafo pode representar precedências entre tarefas.
- Queremos um ordem que respeita as precedências.

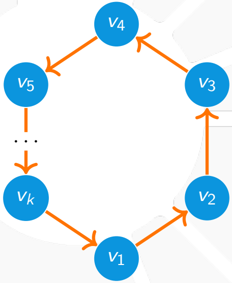


Exemplo de ordenação topológica

Condições de existência

Todo grafo direcionado possui ordenação topológica?

- **NÃO**, um ciclo direcionado não possui!
- Assim, nenhum grafo que contém um ciclo possui!



Um grafo direcionado é **ACÍCLICO** se não contiver um ciclo direcionado.

Condições de existência

Teorema

*Um grafo direcionado é **ACÍCLICO** se e somente se possui uma **ORDENAÇÃO TOPOLÓGICA**.*

Demonstração:

- Se G tem uma ordenação topológica, então ele é **ACÍCLICO**.
- Em seguida, mostraremos a recíproca.

Um lema auxiliar

- Uma **FONTE** é um vértice com grau de entrada zero.
- Um **SORVEDOURO** é um vértice com grau de saída zero.

Lema

*Todo grafo direcionado acíclico G com pelo menos um vértice possui uma **FONTE** e um **SORVEDOURO**.*

Demonstração:

- Tome um caminho maximal $P = (\mathbf{v_0} \ldots \mathbf{v_k})$ em G.
- Então, $\mathbf{v_0}$ é uma fonte e $\mathbf{v_k}$ é um sorvedouro.

Demonstração do teorema

Considere um grafo acíclico $G = (\mathbf{V}, \mathbf{E})$.

Mostraremos que G possui uma ordenação topológica por indução em $|\mathbf{V}|$:

- Se $|\mathbf{V}| = 1$, então a afirmação é clara.
- Considere um grafo com pelo menos dois vértices:
  - Pelo lema anterior, G possui uma fonte $\mathbf{u}$.
  - Pela hipótese de indução, o grafo $G - \mathbf{u}$ possui uma ordenação topológica $\mathbf{v_1}, \ldots, \mathbf{v_{n-1}}$.
  - Logo, $\mathbf{u}, \mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_{n-1}}$ é uma ordenação topológica de G.

Encontrando uma ordenação topológica

A demonstração anterior é construtiva:

- É baseada em exibir uma ordenação topológica.
- Sugere um **ALGORITMO RECURSIVO**.

Algoritmo para ordenação topológica:

1. Encontre uma fonte $\mathbf{u}$ de G.
2. Recursivamente, obtenha ordenação $\mathbf{v_1}, \ldots, \mathbf{v_{n-1}}$ de $G - \mathbf{u}$.
3. Devolva $\mathbf{u}, \mathbf{v_1}, \ldots, \mathbf{v_{n-1}}$.

A complexidade desse algoritmo é $O(V^2)$:

- Encontrar uma fonte leva tempo $O(V)$.
- Há $|V|$ chamadas recursivas.
- Pode-se fazer em tempo $O(V + E)$. (exercício)

Algoritmo baseado em DFS

Considere um grafo direcionado acíclico:

- Como não há ciclo, não existe aresta de retorno.
- Considere o instante em que **v** fica preto.
- Nesse instante **TODOS** seus vizinhos são pretos.
- Isso sugere considerar os vértices na ordem de término.

Ideia para o algoritmo:

- O primeiro vértice a ficar preto não tem arestas saindo.
- O segundo só pode ter arestas para o primeiro.
- O terceiro só pode ter arestas para os dois primeiros.
- etc.

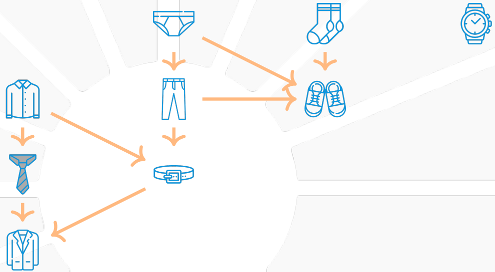
Algoritmo $\textsc{Topological-Sort}$

Algoritmo: $\textsc{Topological-Sort}(u)$

1 execute $\mathrm{DFS}(G)$ e calcule $f[v]$ para cada vértice v
2 quando um vértice finalizar, insira-o no **INÍCIO** de uma lista
3 **devolva** a lista resultante

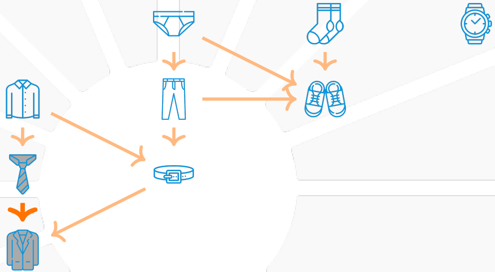
- Inserir cada um dos $|V|$ vértices leva tempo $O(1)$.
- Executamos DFS uma vez.
- Portanto, a complexidade de tempo é $O(V + E)$.

Exemplo



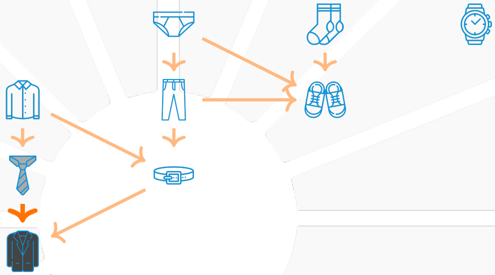
Lista:

Exemplo



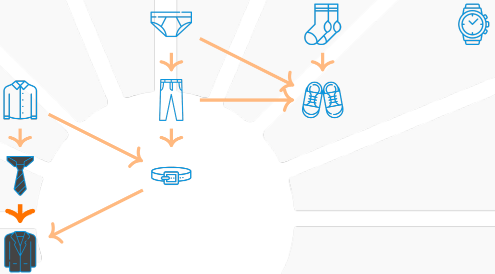
Lista:

Exemplo



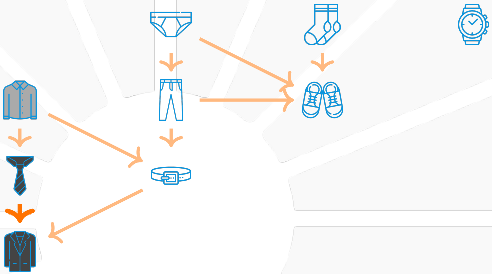
Lista:

Exemplo



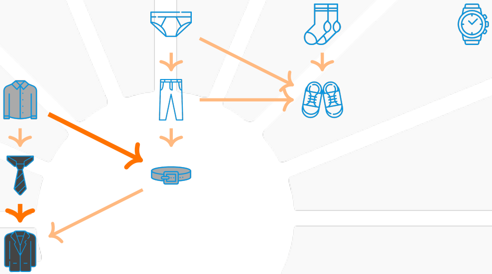
Lista:

Exemplo



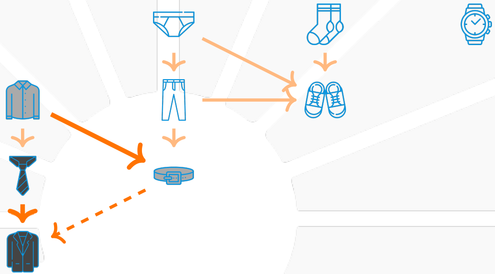
Lista:

Exemplo



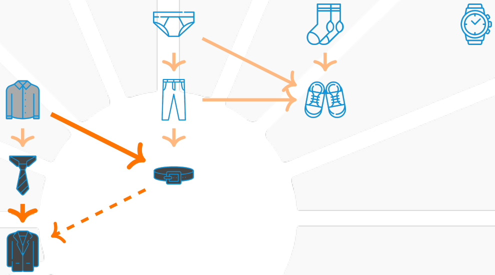
Lista:

Exemplo



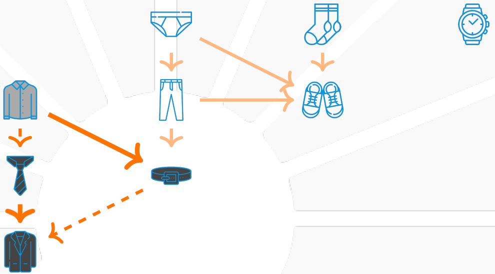
Lista:

Exemplo



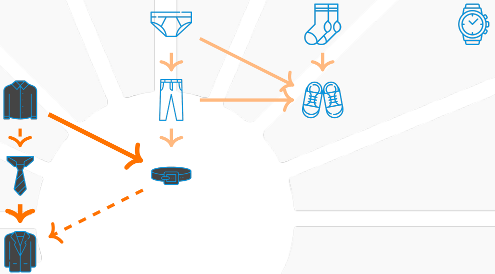
Lista:

Exemplo



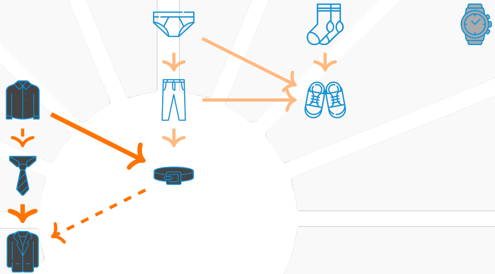
Lista:

Exemplo



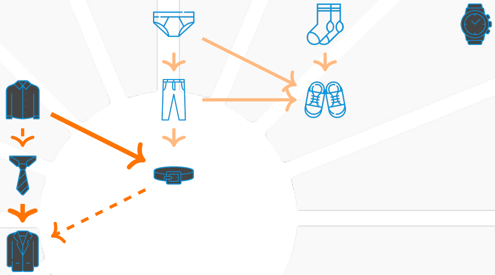
Lista:

Exemplo



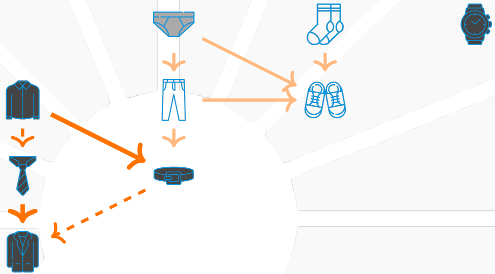
Lista:

Exemplo



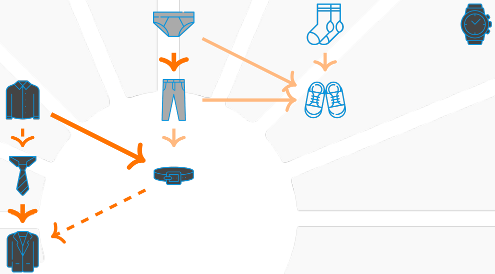
Lista:

Exemplo



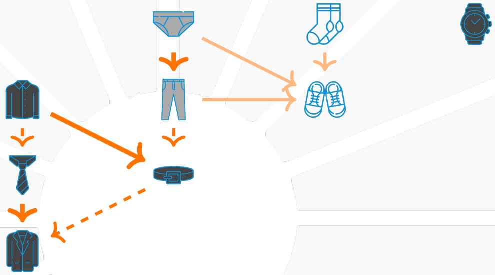
Lista:

Exemplo



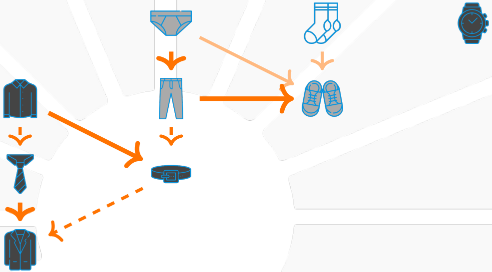
Lista:

Exemplo



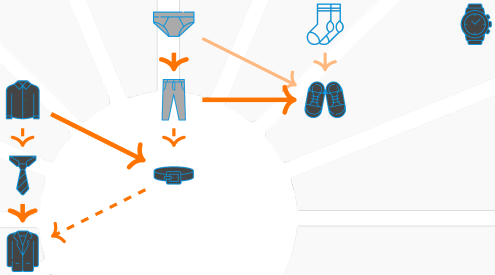
Lista:

Exemplo



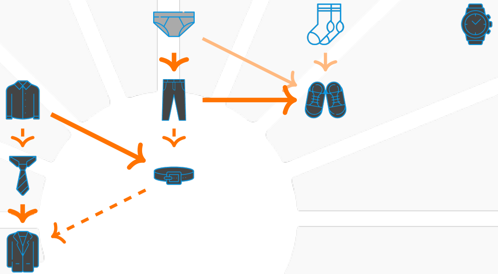
Lista:

Exemplo



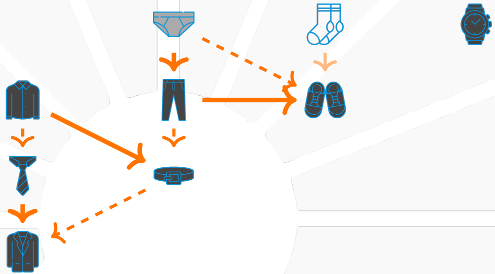
Lista:

Exemplo



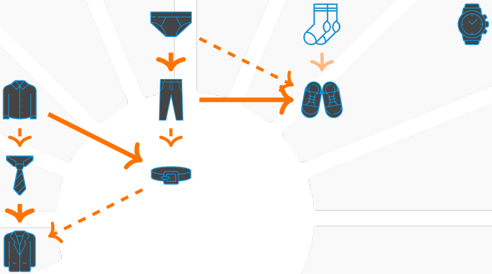
Lista:

Exemplo



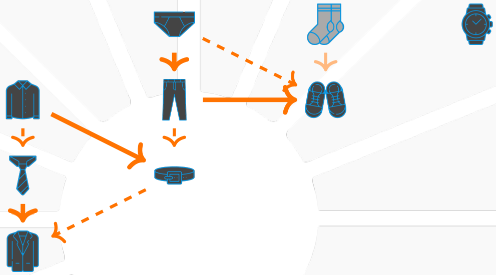
Lista:

Exemplo



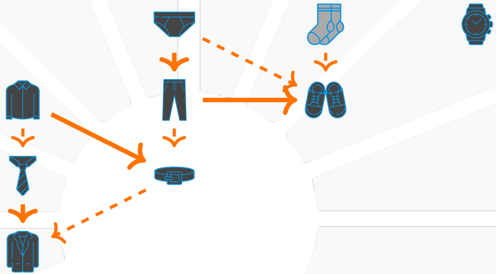
Lista:

Exemplo



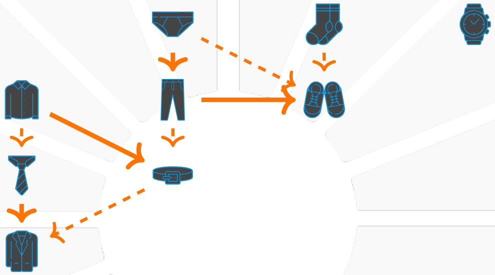
Lista:

Exemplo



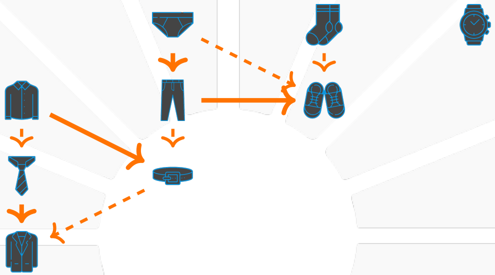
Lista:

Exemplo



Lista:

Exemplo



Lista:

Correção

Teorema

TOPOLOGICAL-SORT(G) *devolve uma ordenação topológica de* G.

Demonstração:

Considere uma aresta arbitrária **(u,v)**:

- Como a lista devolvida está em ordem **DECRESCENTE** de $f[\mathbf{v}]$, basta mostrar que $f[\mathbf{u}] > f[\mathbf{v}]$.
- Considere o instante em que **(u,v)** foi examinada:
- Como **(u,v)** não é aresta de retorno, **v** não pode ser cinza:
  1. Se **v** for branco, então ele será descendente de **u** e $f[\mathbf{u}] > f[\mathbf{v}]$.
  2. Se **v** for preto, então ele já foi finalizado e $f[\mathbf{u}] > f[\mathbf{v}]$.

Contagem de caminhos

Definição

Vamos resolver o seguinte problema:

Problema

Entrada. *Um grafo direcionado acíclico G e dois vértices* ***s, t****.*

Saída. *O número de caminhos de* ***s*** *a* ***t****.*

Observações:

- Queremos apenas **CONTAR** os caminhos, não exibi-los.
- Vamos supor que o grafo não possui arestas múltiplas.
- Esse caso pode ser tratado de modo similar.

Usando programação dinâmica

Considere o seguinte subproblema:

- Seja $\mathbf{v}$ um vértice **QUALQUER** de G.
- Denote por $p(\mathbf{v})$ o número de caminhos de $\mathbf{v}$ a $\mathbf{t}$.

Como calcular $p(\mathbf{s})$?

- É a soma do número de caminhos de cada vizinho $\mathbf{v}$ a $\mathbf{t}$:

$$p(\mathbf{s}) = \sum_{\mathbf{v} \in \mathrm{Adj}[\mathbf{s}]} p(\mathbf{v})$$

- Existe sobreposição de problemas.
- Precisamos de uma ordem para calculá-los.
- Usamos programação dinâmica e ordenação topológica.

Recorrência

Temos a seguinte recorrência:

$$p(\mathbf{u}) = \sum_{\mathbf{v} \in \mathrm{Adj}[\mathbf{u}]} p(\mathbf{v})$$

Esta recorrência vale se G contiver ciclos?

Algoritmo Conta-Caminhos

Algoritmo: CONTA-CAMINHOS(G, s, t)

para cada $u \in V[G] \setminus \{t\}$
    $p[u] \leftarrow 0$
$p[t] \leftarrow 1$
obtenha uma ordenação topológica $(v_1, v_2, \ldots, v_n)$ de G
para $i \leftarrow n$ **até** 1
    para cada $u \in \mathrm{Adj}[v_i]$
        $p[v_i] \leftarrow p[v_i] + p[v]$
devolva $p[s]$

A complexidade de tempo é $O(V + E)$. Para demonstrar a correção, basta provar a seguinte **INVARIANTE DE LAÇO** $p[v_i] = p(v_i)$.

Outra maneira

Considere um subproblema alternativo:

- Seja **v** um vértice **QUALQUER** de G.
- denote por $q(\mathbf{v})$ o número de caminhos de **s** a **v**.
- Temos a seguinte recorrência:

$$q(\mathbf{v}) = \sum_{\mathbf{u}:\mathbf{v}\in \mathrm{Adj}[\mathbf{u}]} q(\mathbf{u})$$

Algoritmo Conta-Caminhos alternativo

Algoritmo: Conta-Caminhos(G, s, t)

para cada $u \in V[G] \setminus \{s\}$
    $q[u] \leftarrow 0$
$q[s] \leftarrow 1$
obtenha uma ordenação topológica $(v_1, v_2, \ldots, v_n)$ de G
para $i \leftarrow 1$ **até** n
    para cada $u \in \mathrm{Adj}[v_i]$
        $q[u] \leftarrow q[u] + q[v_i]$
devolva $q[t]$

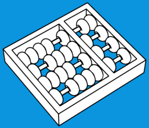
Note a diferença no modo em que $q[u]$ é calculado.

Ordenação topológica

MC558 - Projeto e Análise de Algoritmos II

06

08/24



Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br