BCC204 - Teoria dos Grafos

Marco Antonio M. Carvalho

Departamento de Computação
Instituto de Ciências Exatas e Biológicas
Universidade Federal de Ouro Preto

Conteúdo

Fonte

Este material é baseado no livro

- Goldbarg, M., & Goldbarg, E. (2012). *Grafos: conceitos, algoritmos e aplicações*. Elsevier.

Licença

Ordenação Topológica

Definição

Uma Ordenação Topológica de um grafo acíclico direcionado (GAD), é uma ordenação linear de seus vértices, na qual cada vértice aparece antes de seus descendentes.

Em outras palavras, é uma ordenação linear de vértices na qual cada vértice precede os vértices que formam seu fecho transitivo direto.

Cada GAD possui uma ou mais ordenações topológicas.

Caso um grafo possua ciclos ou seja não direcionado, não é possível estabelecer uma relação de precedência entre os vértices, e portanto, é impossível estabelecer uma ordenação topológica.

Histórico

Algoritmos de ordenação topológica começaram a ser estudados na década de 60, no contexto da técnica PERT/CPM.

O PERT prevê o cálculo da duração de atividades complexas a partir da média ponderada de três durações possíveis dessas atividades.

O CPM é um método de identificação do caminho crítico dada uma sequência de atividades, isto é, quais atividades de uma sequência não podem sofrer alteração de duração sem que isso reflita na duração final de um projeto.

Exemplo

Um exemplo simples do encadeamento de atividades é relacionado na tabela abaixo, para a fabricação de uma estante.

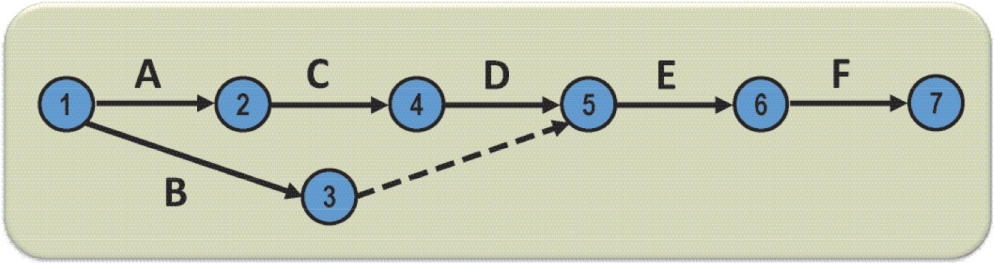
Atividade	Discrição	Duração	Anterior	Posterior
A	Comprar tábuas	1 dia	–	3
B	Comprar parafusos	1 dia	–	5
C	Cortar as tábuas	2 dias	1	4
D	Pintar as tábuas	1 dia	2	5
E	Montar as tábuas com parafusos	1 dia	4,2	6
F	Transportar a estante	1 dia	5	–

Exemplo

Considerando que a coluna *Duração* tenha sido calculada pelo método PERT, é possível, com base na tabela, organizar um grafo do sequenciamento lógico das atividades de fabricação da estante:

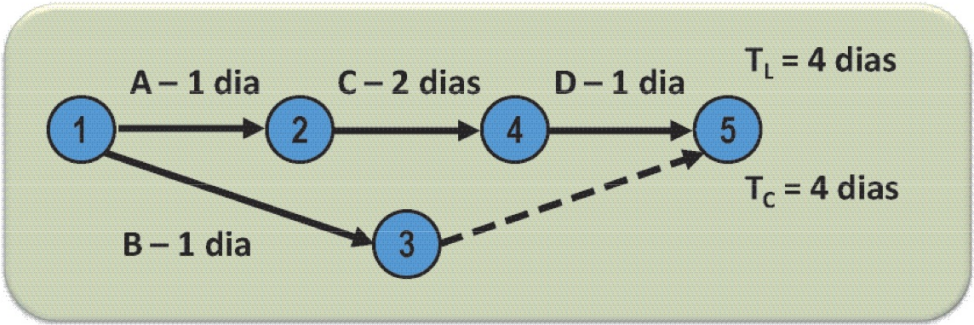
- Os vértices representam eventos instantâneos que caracterizam o início e o término das atividades especificadas nos arcos;
- A sequência das atividades é o principal ponto destacado;
- O arco pontilhado representa a dependência lógica da atividade de montagem das tábuas em relação à atividade de compra dos parafusos.
- Essa "atividade" é denominada atividade fantasma, e sua duração é igual a zero (instantânea).

Ordenação Topológica



Exemplo

É possível ainda representar a duração de cada atividade representada no grafo e calcular o tempo mais curto (T_C) e o tempo mais longo de conclusão das tarefas (T_L).



Alternativa

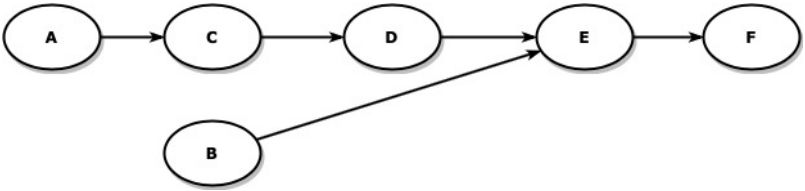
Alternativamente, o grafo resultante pode ser modelado de outra maneira:

- Tarefas a serem desempenhadas em um projeto são representadas por vértices;
- Existe um arco entre as tarefas v e w caso a tarefa v obrigatoriamente deva ser terminada antes que a tarefa w seja começada;
- Uma ordenação topológica dos vértices nos fornece uma maneira de realizar todas as tarefas sem violar as precedências entre tarefas.

Algoritmos

Existem diferentes algoritmos para obtenção de ordenações topológicas em grafos, os de melhor desempenho possuem complexidade linear.

Dois dos mais utilizados são o baseado na Busca em Profundidade (ou DFS) e o Algoritmo de *Kahn*.



Princípio

Atenção, este algoritmo considera o sentido oposto dos arcos em relação ao original apresentado nos diagramas!

O algoritmo considera que, caso um vértice x dependa de um vértice y, então existe um arco de x para y.

Terminologia

- L: Lista que conterá os elementos da ordenação topológica;
- Um vértice pode ser não marcado, temporariamente marcado ou definitivamente marcado;
- Inicialmente, todos os vértices são não marcados;
- Ao serem atingidos pela primeira vez, os vértices são temporariamente marcados;
- Após terem todas as suas dependências examinadas, os vértices são definitivamente marcados;
- Caso um vértice temporariamente marcado seja examinado novamente, o grafo possui pelo menos um ciclo.

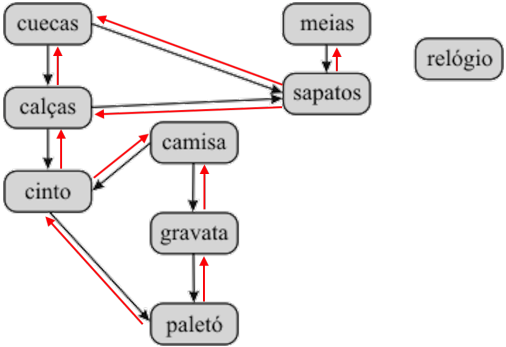
Busca em Profundidade - DFS

Entrada: Grafo $G = (V, A)$

```
1  L ← ∅;
2  enquanto existir vértice não marcado e sem arcos de entrada faça
3  |   selecione um vértice v não marcado;
4  |   visite(G, v, L);
5  fim
```

```
1  função visite(G,v,L)
2  se v é temporariamente marcado então  retorna Erro;//detecção de ciclo ;
3  se v é não marcado então
4  |   marque temporariamente v;
5  |   para cada arco (vw) faça
6  |   |   visite(G, w, L);//chamada recursiva da função
7  |   fim
8  |   marque definitivamente v;
9  |   adicione v ao final de L;
10 fim
```

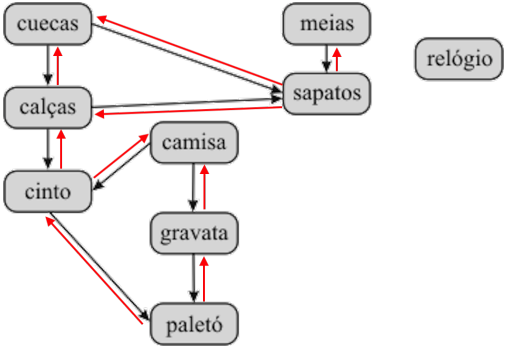
Exemplo



Lembre-se de inverter o sentido dos arcos!

L = []

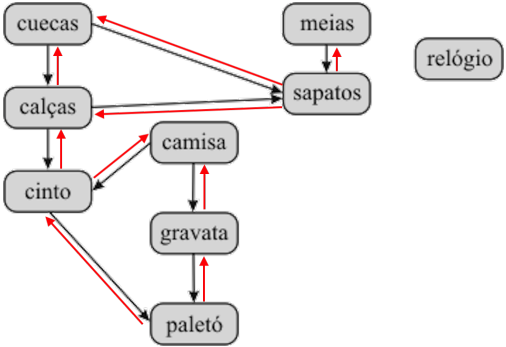
Exemplo



Lembre-se de inverter o sentido dos arcos!

L = [cuecas]

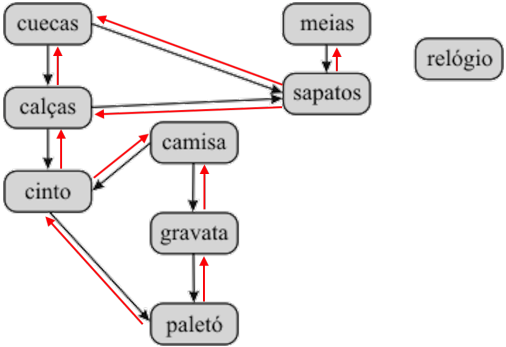
Exemplo



Lembre-se de inverter o sentido dos arcos!

L = [cuecas, calças]

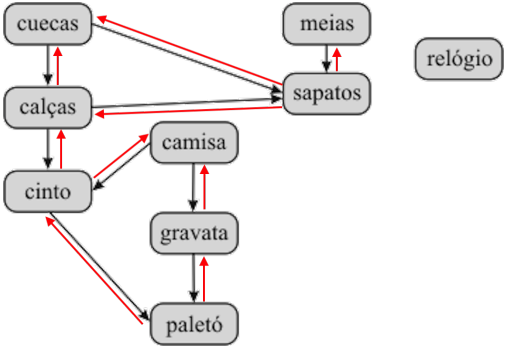
Exemplo



Lembre-se de inverter o sentido dos arcos!

L = [cuecas, calças, camisa]

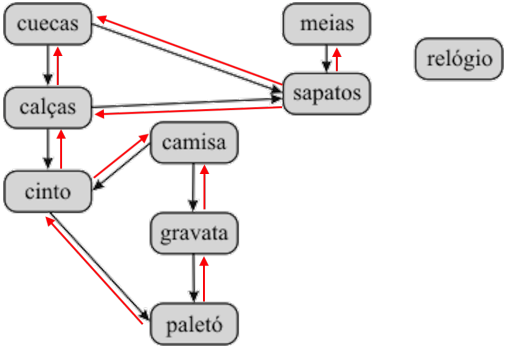
Exemplo



Lembre-se de inverter o sentido dos arcos!

L = [cuecas, calças, camisa, cinto]

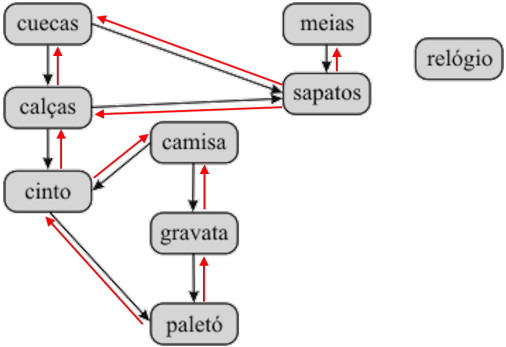
Exemplo



Lembre-se de inverter o sentido dos arcos!

L = [cuecas, calças, camisa, cinto, gravata]

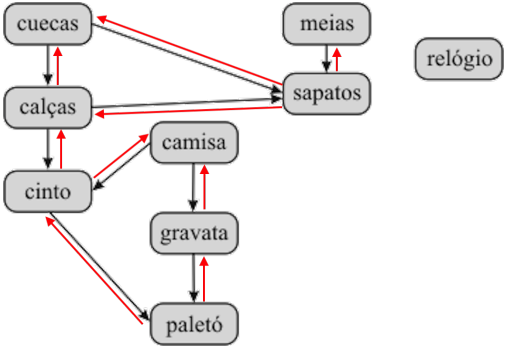
Exemplo



Lembre-se de inverter o sentido dos arcos!

L = [cuecas, calças, camisa, cinto, gravata, paletó]

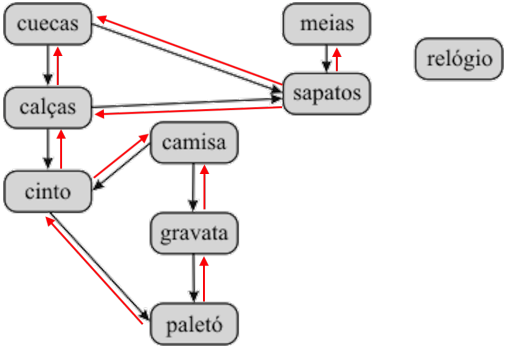
Exemplo



Lembre-se de inverter o sentido dos arcos!

L = [cuecas, calças, camisa, cinto, gravata, paletó, relógio]

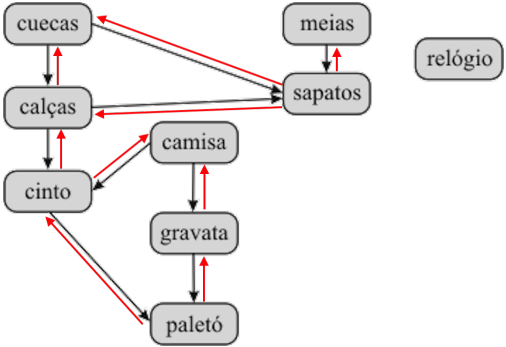
Exemplo



Lembre-se de inverter o sentido dos arcos!

L = [cuecas, calças, camisa, cinto, gravata, paletó, relógio, meias]

Exemplo



Lembre-se de inverter o sentido dos arcos!

L = [cuecas, calças, camisa, cinto, gravata, paletó, relógio, meias, sapatos]

Algoritmo de *Kahn*

Aplicação

O algoritmo de *Kahn* data de 1962 e possui como princípio determinar a cada instante os vértices que não possuam arcos de entrada e inserir na solução.

A cada vértice inserido na solução, todos seus arcos correspondentes são removidos do grafo.

Também detecta a existência de ciclos no grafo.

Terminologia

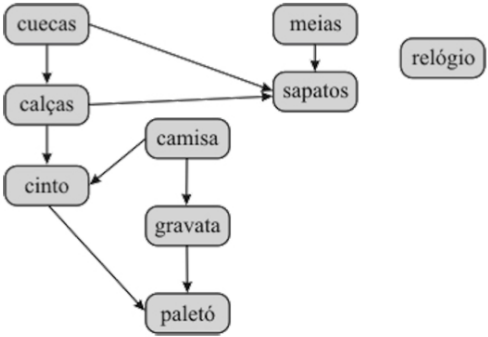
- L: Lista que conterá os elementos da ordenação topológica;
- S: Conjunto de vértices que não possuem arcos de entrada.

Algoritmo de *Kahn*

Entrada: Grafo $G=(V, A)$

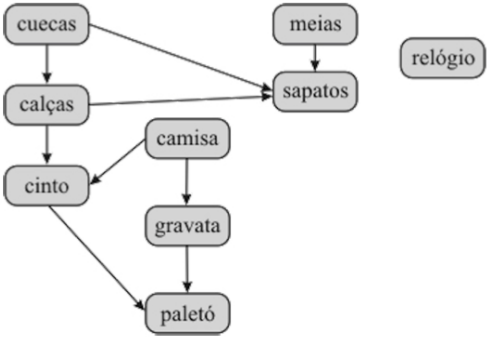
```
1  L ← ∅;
2  S ← vértices sem arcos de entrada;
3  enquanto S ≠ ∅ faça
4  |   Remova um vértice v de S;
5  |   Insira o vértice v em L;
6  |   para cada arco (vw) faça
7  |   |   Remova o arco vw de A;
8  |   |   se w não possui mais arcos de entrada então
9  |   |   |   Insira o vértice w em S;
10 |   |   fim
11 |   fim
12 fim
13 se A ≠ ∅ então  retorna Erro;//o grafo possui pelo menos um ciclo ;
14 senão  retorna L;//a ordenação topológica ;
```

Exemplo



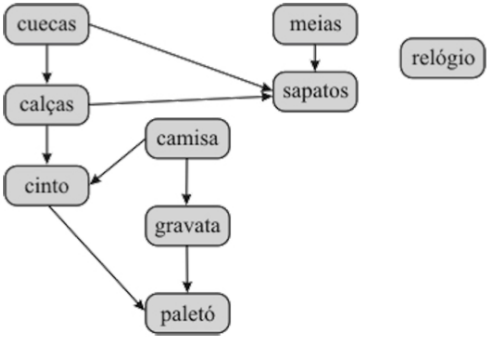
S = {cuecas, camisa, meias, relógio}
L = []

Exemplo



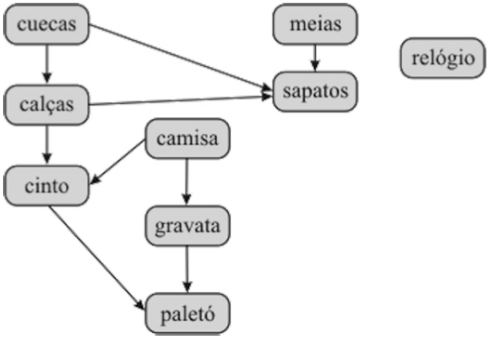
S = {camisa, meias, relógio, calças}
L = [cuecas]

Exemplo



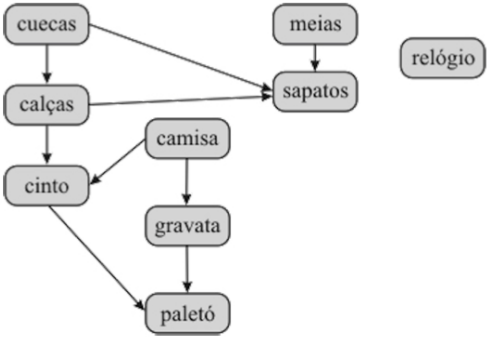
S = {meias, relógio, calças, gravata}
L = [cuecas, camisa]

Exemplo



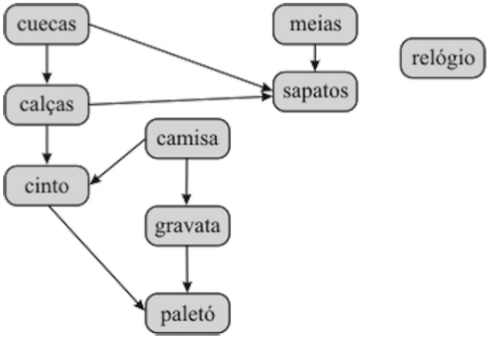
S = {relógio, calças, gravata}
L = [cuecas, camisa, meias]

Exemplo



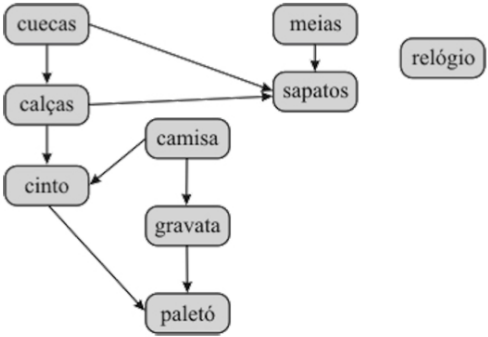
S = {calças, gravata}
L = [cuecas, camisa, meias, relógio]

Exemplo



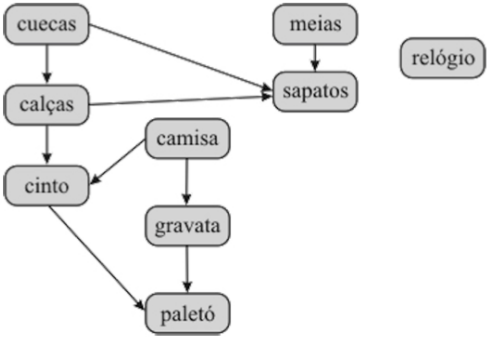
S = {gravata, sapatos, cinto}
L = [cuecas, camisa, meias, relógio, calças]

Exemplo



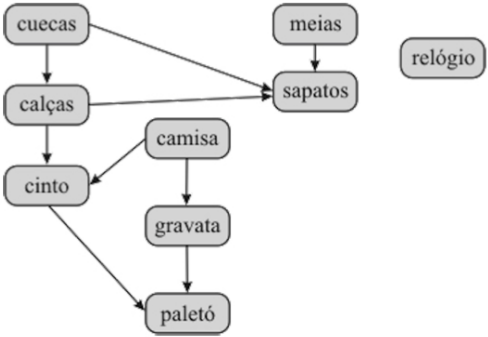
S = {sapatos, cinto}
L = [cuecas, camisa, meias, relógio, calças, gravata]

Exemplo



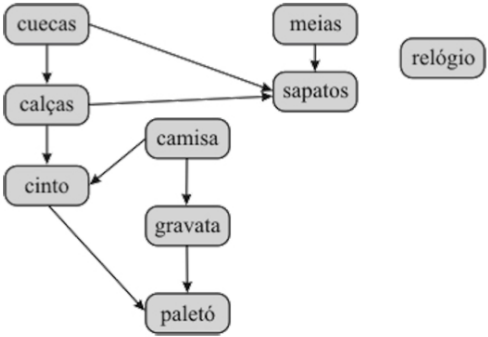
S = {cinto}
L = [cuecas, camisa, meias, relógio, calças, gravata, sapatos]

Exemplo



S = {paletó}
L = [cuecas, camisa, meias, relógio, calças, gravata, sapatos, cinto]

Exemplo



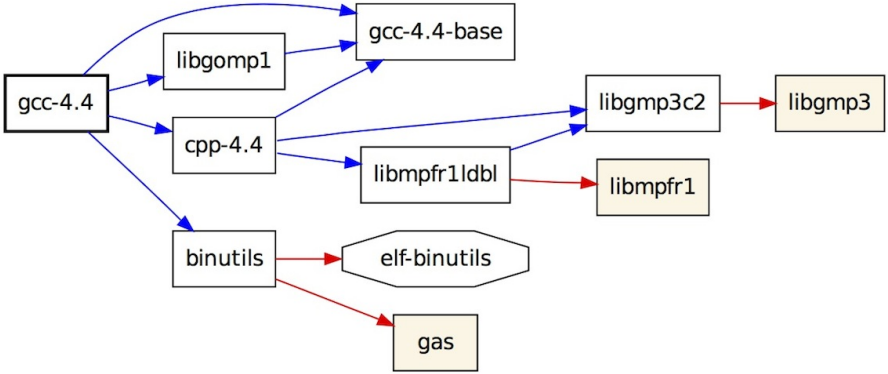
S = {}
L = [cuecas, camisa, meias, relógio, calças, gravata, sapatos, cinto, paletó]

Aplicações

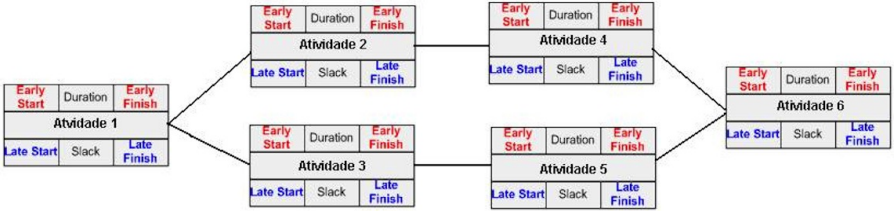
Exemplos

- Roteiros de instalação de pacotes;
- O *make* (do *makefile*);
- Confecção de dicionários;
- Organização de bancos de dados;
- Sistemas geográficos;
- Alocação de projetos (*Project Scheduling*).

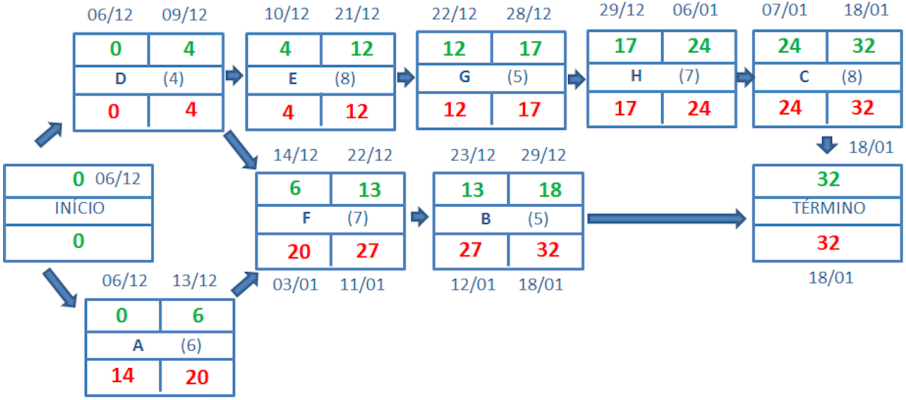
Aplicações: Dependências do pacote GCC



Aplicações: Caminho Crítico



Aplicações: Caminho Crítico (2)



Curiosidade

Em plataformas Linux, há o comando *tsort*, que realiza uma ordenação topológica em uma lista de adjacências informada pelo usuário.

Dúvidas?