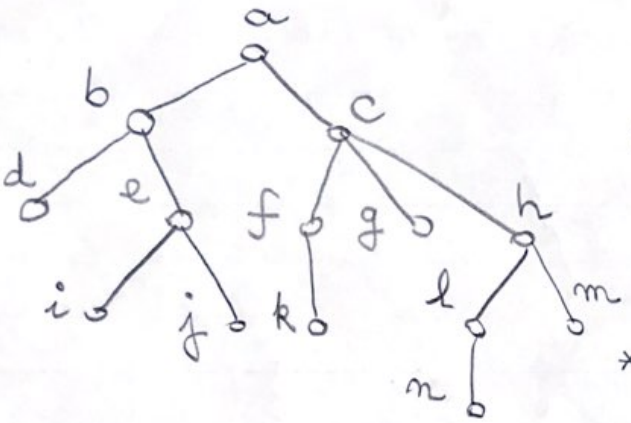
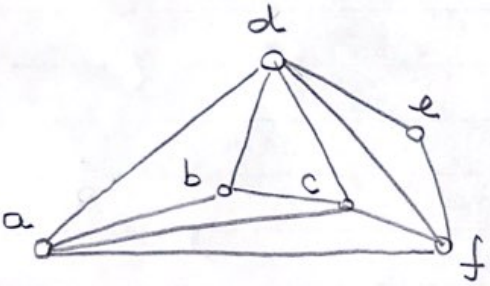
4 Buscas em Grafos

Explorar um grafo, caminhar pelos vértices e arestas.



árvore com raiz a

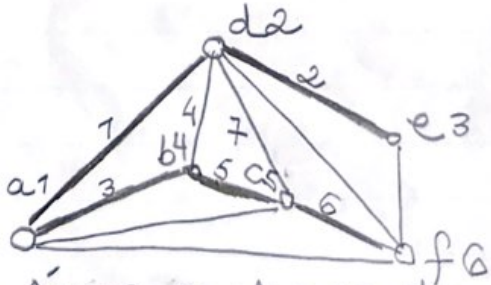
* preordem
  a b d e i j c f k g h l n m
  desce em profundidade, da esquerda para direita
* ordem de nível
  a b c d e f g h i j k l m n
  desce em largura, por nível.



Algoritmo 4.1: Busca geral

<u>Dados</u>: grafo $G(V,E)$
  desmarcar todos os vértices
  escolher e marcar vértice inicial
  <u>enquanto</u> existe vértice v marcado e incidente a aresta (v,w) não explorada <u>efetuar</u>
    escolher v e explorar (v,w)
    <u>se</u> w é não marcado <u>então</u> marcar w.

a marcado
escolhe a explora (a,d)
d marcado d alcançado
escolhe d explora (d,e)
e marcado
escolhe a explora (a,b)
b marcado
escolhe b explora (b,d)
escolhe b explora (b,c)
c marcado b explorado
escolhe c explora (c,f)
f marcado
escolhe c explora (c,d)



Árvore geradora: arestas alcançam vértice pela primeira vez.

Algoritmo 4.2: Busca em Profundidade

<u>Dados</u>: $G(V,E)$, conexo

<u>Procedimento</u> $P(v)$

marcar v

colocar v na pilha Q

<u>para</u> $w \in A(v)$ <u>efetuar</u>

<u>se</u> w é não marcado <u>então</u>

visitar $(v,w) \rightsquigarrow$ I

$P(w)$

caso contrário

<u>se</u> $w \in Q$ e v, w não são consecutivos em Q <u>então</u>

visitar $(v,w) \rightsquigarrow$ II

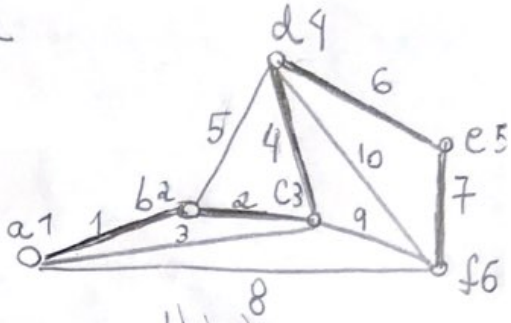
retirar v de Q

desmarcar todos os vértices

definir uma pilha Q

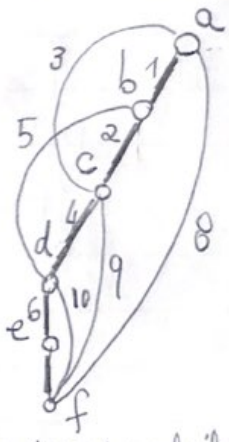
escolher uma raiz s

$P(s)$



a	b c f
b	a c d
c	a b d f
d	b c e f
e	d f
f	a c d e

1 ← f
2 ← e
3 ← d
4 ← c
5 ← b
6 ← a
Q



Árvore de profundidade

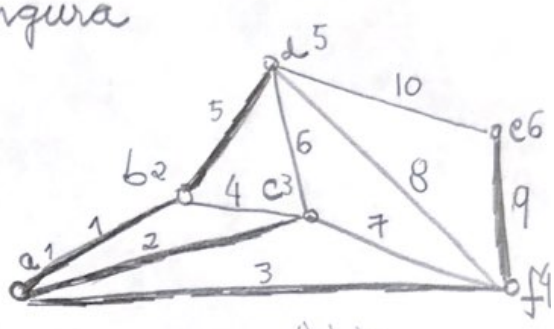
Algoritmo 4.2 particiona $E = I \cup II =$ árvore $\cup$ retorno

Teorema 41 As arestas de árvore definem uma árvore geradora chamada árvore de profundidade.

Teorema 4.2 Se T é árvore de profundidade, então toda aresta de G é entre ancestral e descendente em T.

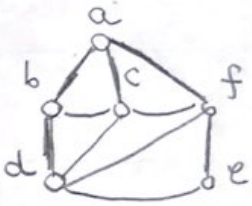
Algoritmo 4.3 : Busca em Largura

<u>Dados</u> : $G(V, E)$, conexo



desmarcar todos os vértices
escolher uma raiz $s \in V$
definir uma fila Q, vazia
marcar s
inserir s em Q
<u>enquanto</u> $Q \neq \emptyset$ <u>efetuar</u>
  seja v o primeiro elemento de Q
  <u>para</u> $w \in A(v)$ <u>efetuar</u>
    <u>se</u> w é não marcado <u>então</u>
      visitar (v, w) $\rightsquigarrow$ I
      marcar w
      inserir w em Q
    <u>caso contrário</u>
      <u>se</u> $w \in Q$ <u>então</u>
        visitar (v, w) $\rightsquigarrow$ II
  retirar v de Q

a	b c f
b	a c d
c	a b d f
d	b c e f
e	d f
f	a c d e



1	2	3	4	5
a	b	c	f	d	e

Árvore de Largura

Algoritmo 4.3 particiona $E = I \cup II$ = árvore $\cup$ II

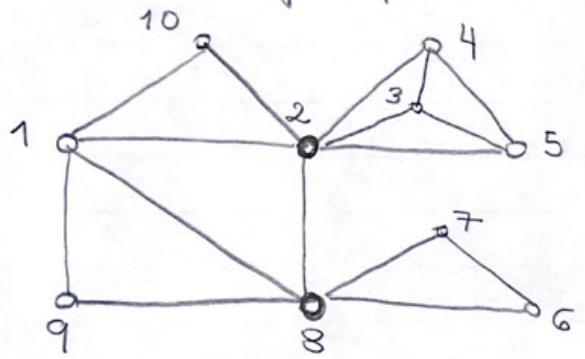
Teorema 4.5 As arestas de árvore definem uma árvore geradora chamada árvore de largura.

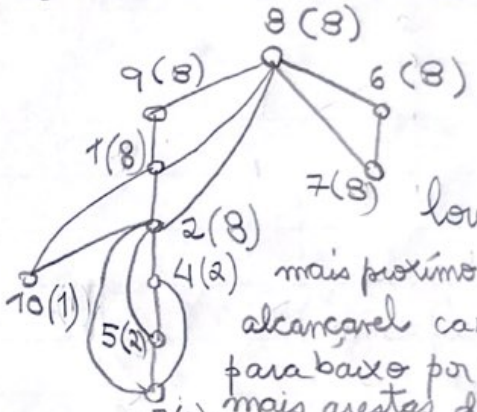
Teorema 4.6 se T é árvore de largura, então os extremos de uma aresta de G têm níveis diferindo no máximo de 1.

$n(w) = n(v) + 1$: se v é pai de w, então (v, w) é aresta pai; senão (v, w) é aresta tio.
$n(w) = n(v)$: se v e w têm o mesmo pai, então (v, w) é aresta irmão, senão (v, w) é aresta primo.

4.4 Biconectividade

Um grafo é biconexo em vértices se e somente se não possui articulações. As componentes biconexas são os subgrafos maximais biconexos e as arestas que não estão em nenhum ciclo. As componentes biconexas definem uma partição do conjunto das arestas, mas não dos vértices. As articulações são os vértices que pertencem a mais de um bloco.





lowpt $(v) = w$ mais próximo da raiz alcançável caminhando para baixo por zero ou mais arestas de árvore seguido de no máximo uma de retorno.

Vértices 2 e 8 são articulações.
O grafo tem 3 blocos.

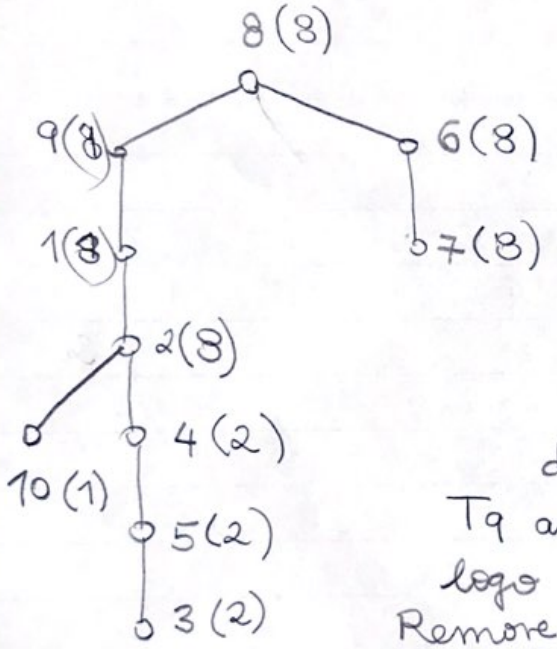
Lema 4.1 Seja G um grafo conexo e T uma árvore de profundidade. v é articulação se e somente se:
(i) v é raiz de T e possui mais de um filho; ou
(ii) v não é raiz de T e possui um filho w tal que lowpt $(w) = v$ ou w.

Uma articulação v é pai de um ou mais demarcadores w.

Lema 4.2 Seja G um grafo e T uma árvore de profundidade. Seja w demarcador de v tal que a subárvore T_w com raiz w não contém articulações. Então v junto com os vértices de T_w induzem um bloco.

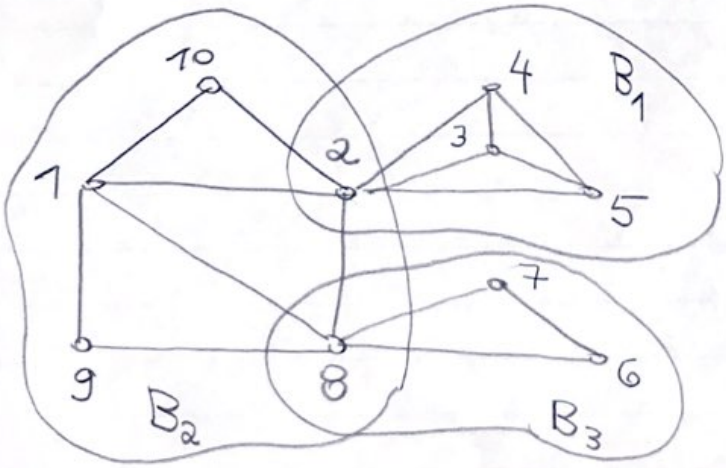
A11.5

O algoritmo primeiro realiza uma busca em profundidade, calcula lowpt e determina as articulações e os demarcadores.
Percorre a árvore de profundidade T de baixo para cima, escolhe demarcador w tal que T_w não possui articulação, o vértice pai v junto com T_w induzem um bloco, e retira T_w de T.



As articulações são 2 e 8.
6 e 9 demarcam a raiz 8.
4 demarcador de 2.

T_4 não possui articulação, logo {2, 4, 5, 3} induz bloco.
Remove {4, 5, 3} de T, e 2 deixa de ser articulação.
T_9 agora não possui articulação, logo {8, 9, 1, 2, 10} induz bloco.
Remove {9, 1, 2, 10} de T.
T_6 não possui articulação, logo {8, 6, 7} induz bloco.



Os três blocos.

4.5 Busca em profundidade - digrafos

Algoritmo 4.4: Busca em Profundidade em Digrafos

Dados: digrafo $D(V,E)$

Procedimento $P(v)$

marcar v

colocar v na pilha Q

para $w \in A(v)$ efetuar

visitar (v,w)

se w não marcado então

$P(w)$

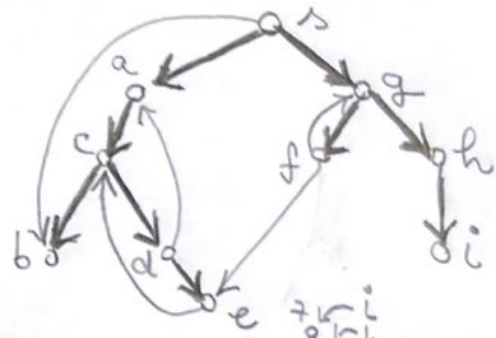
retirar v de Q

desmarcar todos os vértices

definir uma pilha Q

definir uma raiz $s \in V$

$P(s)$



s	a g b	7 ← i
a	c	8 ← h
b	∅	6 ← f
c	b d	9 ← g
d	a e	2 ← e
e	c	3 ← d
f	e g	1 ← b
g	f h	4 ← c
h	i d	5 ← a
i	∅	10 ← s
		Q

Algoritmo 4.4 particiona E em 4 subconjuntos de arestas (v,w):

v alcançado antes de w: w desmarcado $\Rightarrow (v,w)$ árvore;

w marcado $\Rightarrow (v,w)$ avanço $\quad PE(v) < PE(w)$

w alcançado antes de v: $w \in Q \Rightarrow (v,w)$ retorno;

$w \notin Q \Rightarrow (v,w)$ cruzamento, $PE(v) > PE(w)$ e $PS(v) > PS(w)$.

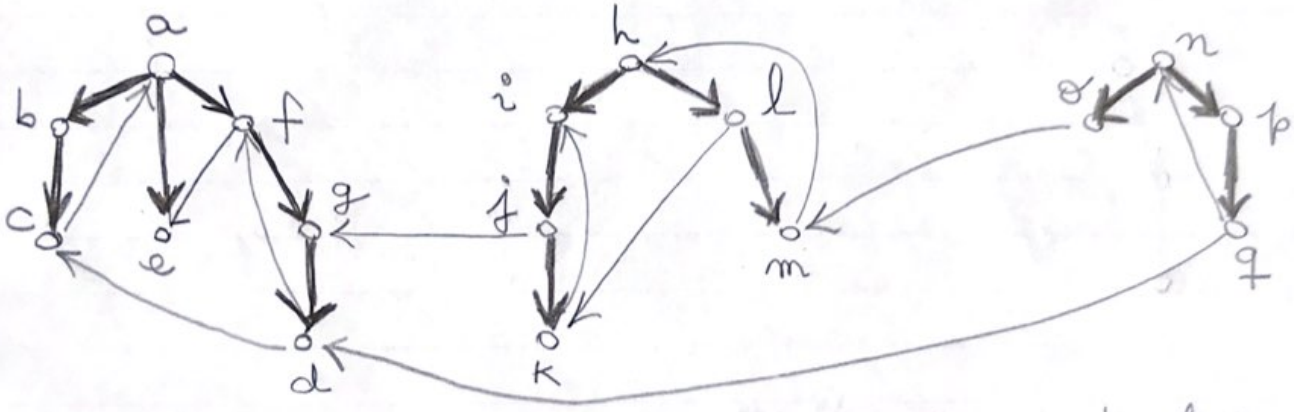
Teorema 4.4. Seja $D(V,E)$ um digrafo com raiz s. As arestas de árvore E_T definem $T(V,E_T)$ que é uma árvore de profundidade de raiz s. Toda aresta (v,w) de árvore é tal que v é pai de w; de avanço é tal que v é ancestral não pai de w; de retorno é tal que v é descendente de w; e de cruzamento é tal que v não é nem ancestral nem descendente de w em T.

4.6 Componentes Fortemente Conexas

Um digrafo $D(V,E)$ é fortemente conexo quando todo par de vértices $v, w \in V$ admite caminho em D de v para w e de w para v.

Uma componente fortemente conexa é um subdigrafo maximal de D que seja fortemente conexo.

As componentes fortemente conexas definem uma partição do conjunto de vértices. Se o digrafo possui mais de uma componente fortemente conexa, então, caso o digrafo seja fracamente conexo, algumas arestas não pertencem a nenhuma componente forte.



Floresta de profundidade, seis componentes fortes.

Lema 4.4 Seja $D(V,E)$ um digrafo, $T(V, E_T)$ uma floresta de profundidade de D e $S(V_S, E_S)$ uma componente forte de D. Então V_S define subárvore parcial de T.

subárvore parcial de T é uma subárvore que possivelmente perdeu vértices e ainda é conexa.

As componentes fortes particionam V em subárvores parciais com raízes nos vértices fortes: a, e, i, h, o, n.

Algoritmo 4.5: Componentes Fortes

Dados: digrafo $D(V,E)$

Procedimento $F(v)$

marcar v

$j := j+1$

$PE(v) := low(v) := j$

para $w \in A(v)$ **efetuar**

se w não marcado **então**

incluir (v,w) em T, sendo v pai de w

$F(w)$

$low(v) := \min\{low(v), low(w)\}$

caso contrário

se w está em T **então**

$low(v) := \min\{low(v), PE(w)\}$

se $low(v) = PE(v)$ **então**

retirar v e seus descendentes de T

$j := 0$

desmarcar todos os vértices

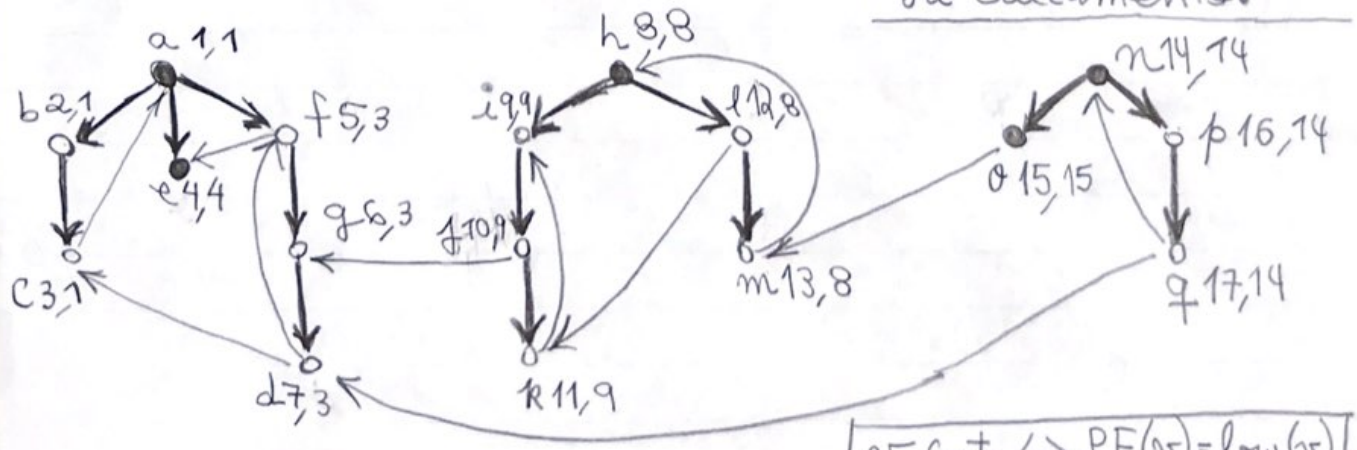
definir uma árvore T, vazia

para $s \in V$ **efetuar**

se s não marcado **então**

$F(s)$

$low(v) = PE(z)$, z menor PE na mesma componente forte que v, alcançável por zero ou mais de árvore, seguido por no máximo 1 retorno ou cruzamento.



$\boxed{v \text{ forte} \Leftrightarrow PE(v) = low(v)}$