Discussão preliminar sobre Conectividade

- *A **conectividade** está relacionada a passagem de um vértice a outro em um grafo através de ligações existentes.*

- *Está passagem diz respeito a **atingibilidade**.*

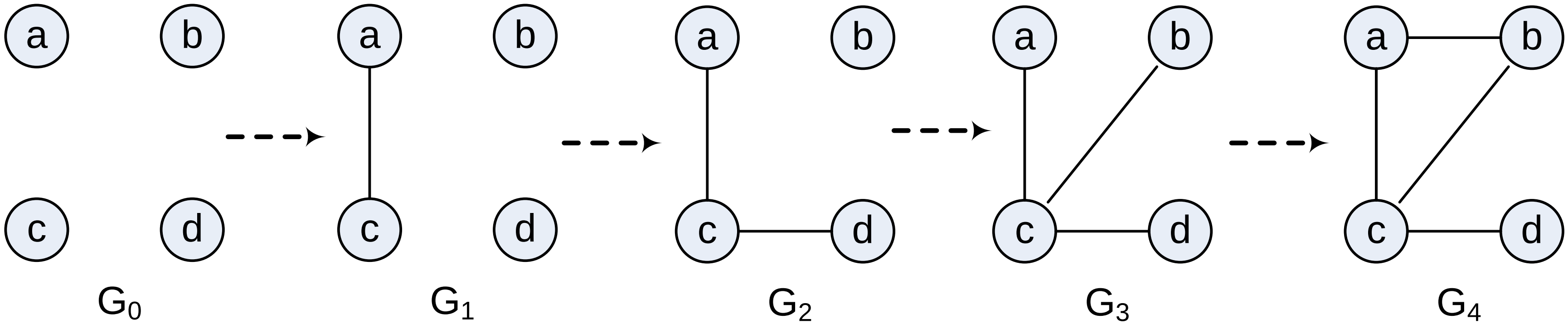
- *Exemplos na prática:*

    - *Um vértice servidor pode enviar mensagens de dados para um determinado cliente?*

    - *Você consegue ir de carro da cidade X para a cidade Y?*

Discussão preliminar sobre Conectividade

Conectividade em grafos não orientados



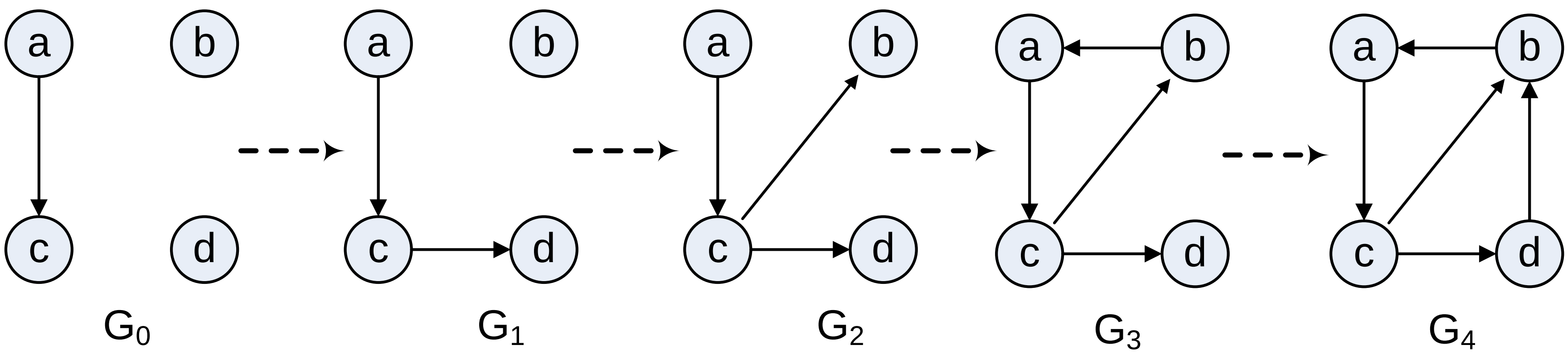
- *Podemos ilustrar a atingibilidade na seqüência acima.*

- *Dado um gravo trivial G=(V, ∅), adicionamos sucessivas ligações ao conjunto de arestas, para aumentarmos a atingibilidade entre vértices.*

Discussão preliminar sobre Conexidade

Conexidade em grafos orientados

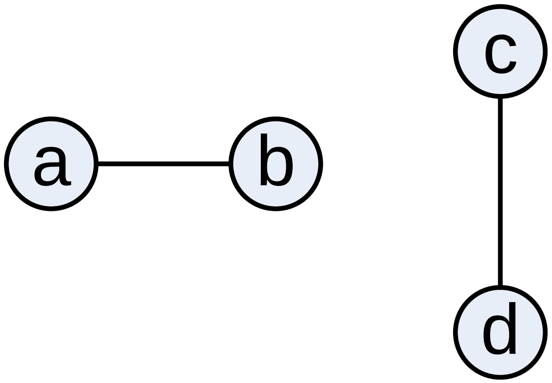
- *A atingibilidade entre vértices também pode ser observada em grafos orientados.*
- *Vejamos a seqüência abaixo:*



Tipos de Conectividade

Para grafos orientados ou não

- Conexo ou não conexo (*definição*)
  - *"Um grafo é não conexo (desconexo) se nele existir ao menos um par de vértices não unidos por uma cadeira"*



G_0

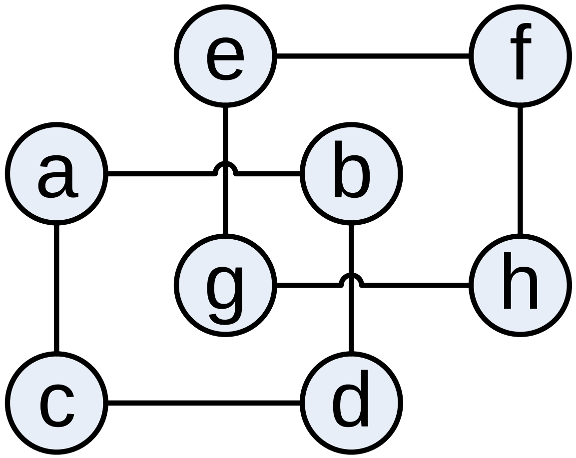
Pares não unidos por uma cadeira:

- (a,c)
- (a,d)
- (b,c)
- (b,d)

Tipos de Conexidade

Para grafos orientados ou não

- Conexo ou não conexo (*definição*)
  - *"Um grafo é não conexo (desconexo) se nele existir ao menos um par de vértices não unidos por uma cadeira"*



G_1

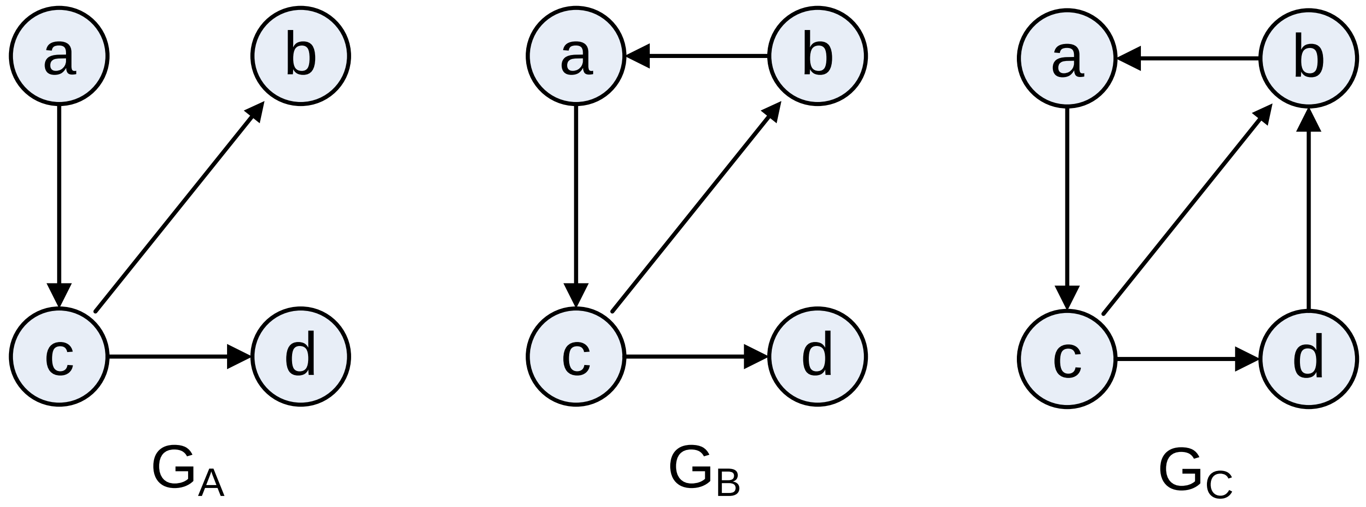
Pares não unidos por uma cadeira:

- (a,e), (a,f), (a,g), (a,h),
- (b,e), (b,f), (b,g), (b,h),
- (c,e), (c,f), (c,g), (c,h),
- (d,e), (d,f), (d,g), (d,h);

Tipos de Conectividade

Em grafos orientados

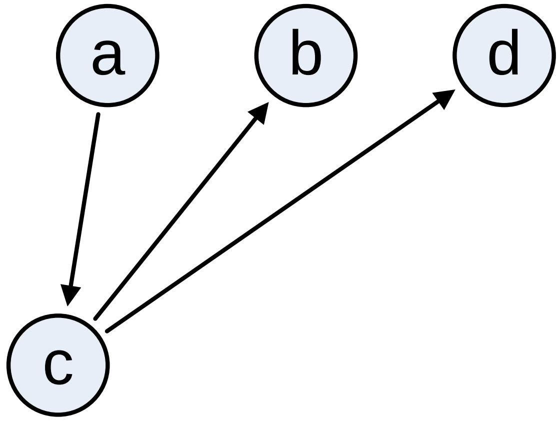
- *Os grafos G_A, G_B e G_C são conexos, mas possuem diferenças fundamentais de atingibilidade...*



Tipos de Conexidade

Simplesmente conexo (s-conexo)

- *s-conexo (simplesmente conexo):*
  - *todo par de vértices é unido por ao menos uma cadeia*



G s-conexo

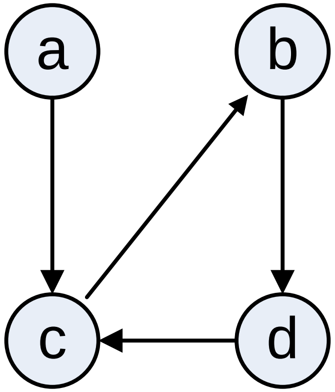
Não existe percurso de *b* para *d* e nem de *d* para *b*, mas estes vértices estão ligados por uma cadeia de arcos:

- (c,b); (c,d)

Tipos de Conexidade

Semi-fortemente conexo (sf-conexo)

- *sf-conexo (semi-fortemente conexo):*
  - *Em todo par de vértices, ao menos um dos vértices é atingível a partir do outro*



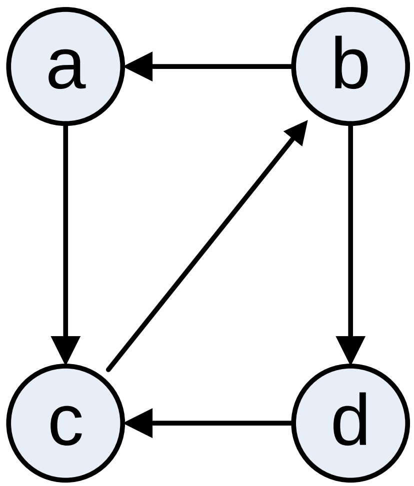
G sf-conexo

Não existe percurso de *b* para a, mas existe percurso de *a* para *b*.

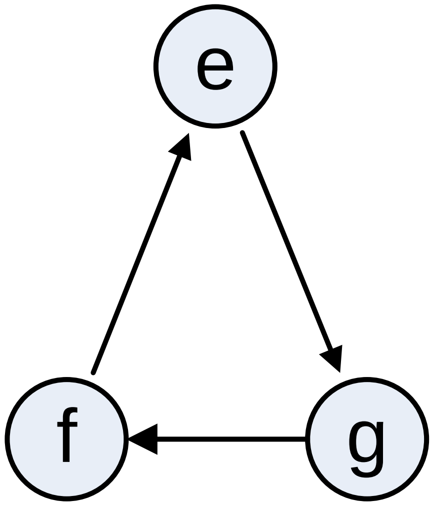
Tipos de Conexidade

Fortemente conexo (f-conexo)

- *f-conexo (fortemente conexo):*
  - *Em todo par de vértices, os vértices são mutuamente atingíveis.*



G_1 f-conexo

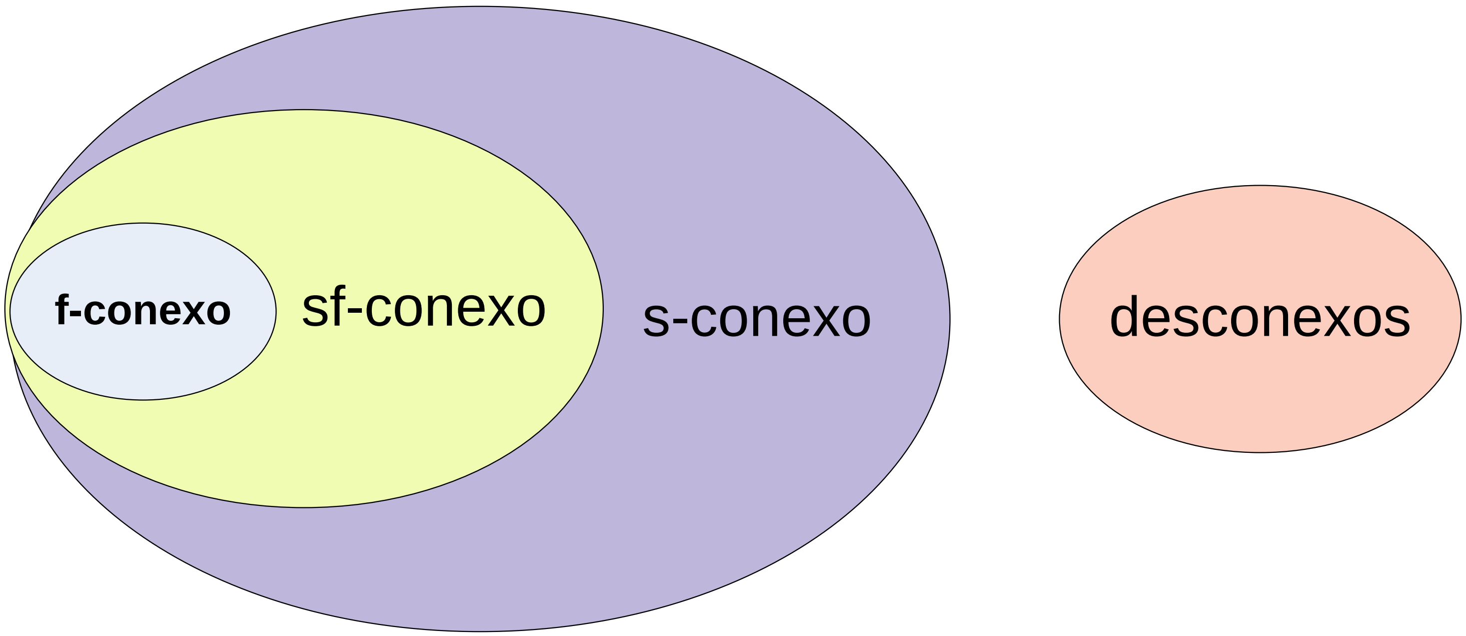


G_2 f-conexo

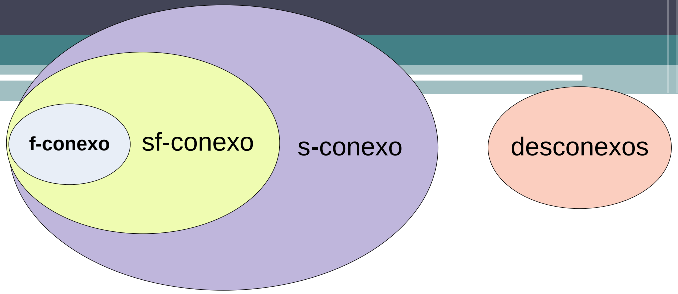
Tipos de Conexidade

Observações

- *Nos grafos orientados, podemos notar que:*
  - *Todo grafo f-conexo é também um grafo sf-conexo;*
  - *Todo grafo sf-conexo é também um grafo s-conexo;*



Tipos de Conexidade
Aplicações



Atenção especial para os conjuntos

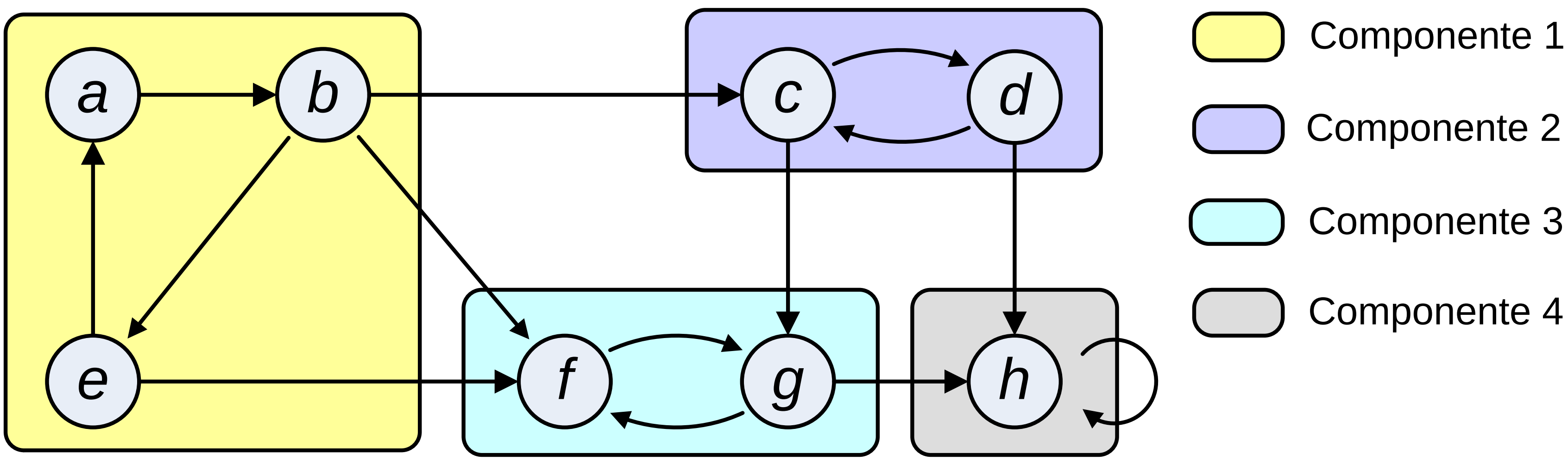
- C_1 = (*s-conexo*) – (*sf-conexo*)
- C_2 = (*sf-conexo*) – (*f-conexo*)

- Os conjuntos C_1 e C_2 contem grafos cujos alguns vértices são *privilegiados*, em relação a outros.

- Muitas aplicações em redes, por exemplo, precisam desta representação

Componentes fortemente conectados

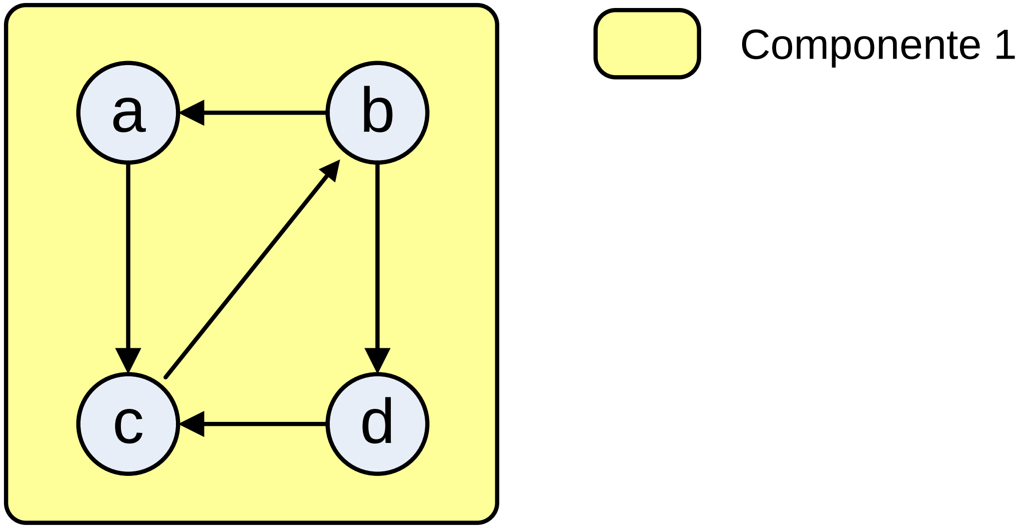
- Um componente fortemente conectado de um grafo orientado $G = (V,A)$ é um conjunto máximo de vértices $C \subseteq V$ tal que, para todo par de vértices u e v em C, temos que os vértices u e v são acessíveis um a partir do outro.



Componentes fortemente conexos
Observação

- Todo grafo que possui apenas um componente conexo é fortemente conexo *(f-conexo)*

- Exemplo já visto anteriormente:

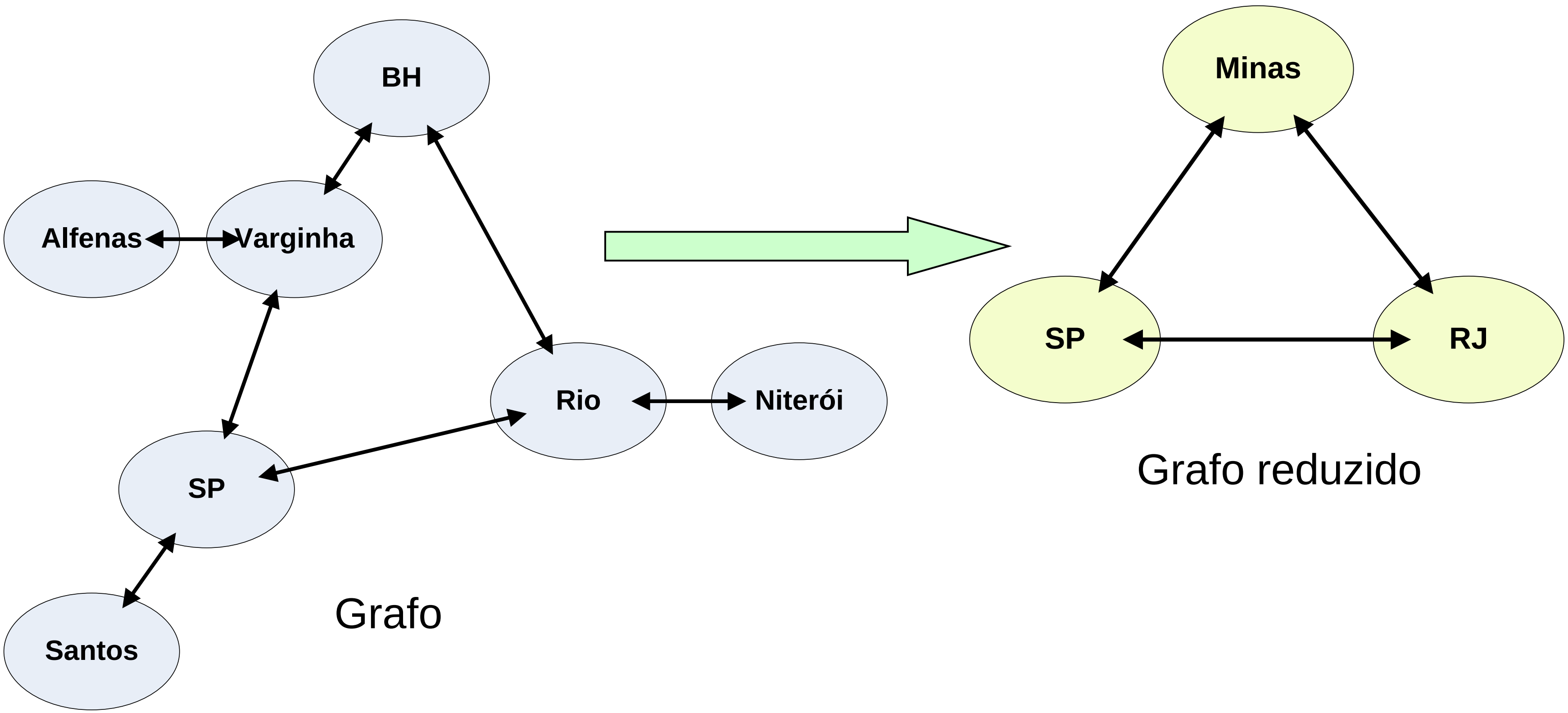


Grafos Reduzidos

Grafos Reduzidos

Definição

- *"Define-se um grafo G_r, obtido de G, através de uma seqüência de contrações de vértices, feitas de um critério predefinido."*

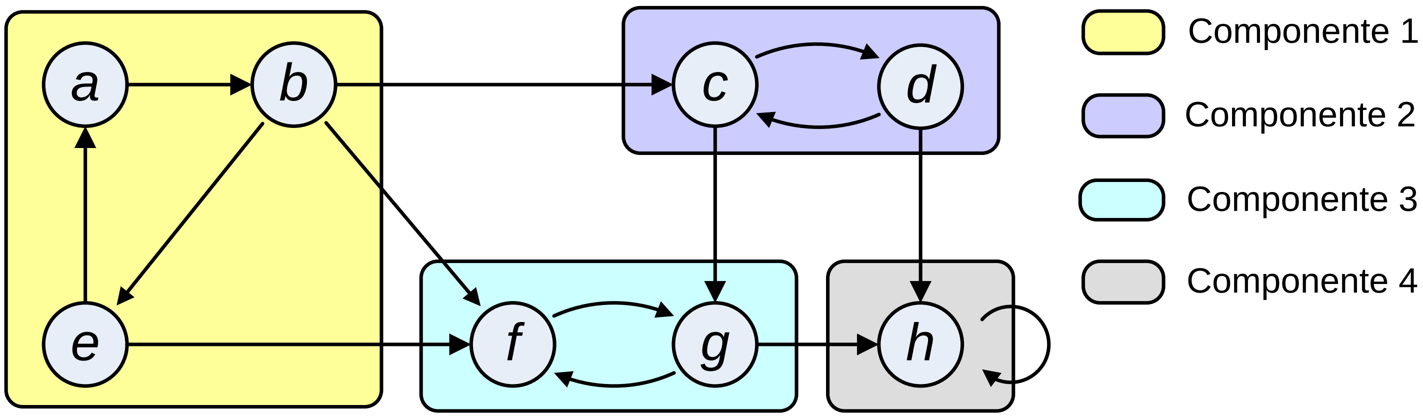


Critério: Agrupar por estado da federação

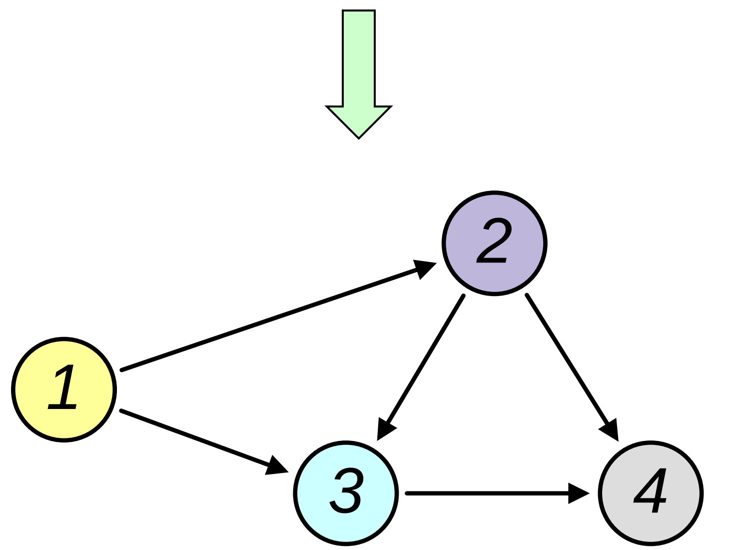
Grafos Reduzidos

Redução utilizando o critério de componentes fortemente conexos

G



G_r



Aplicações em redes!!!

Para montar tabelas de encaminhamento

Redução por componentes fortemente conexos
Algoritmo

- *Existem diferentes algoritmos para decomposição por conectividade.*

- *Uma implementação eficiente faz uso do algoritmo de busca em profundidade para identificar as componentes conexos.*

Redução por componentes fortemente conexos
Algoritmo: Componentes fortemente conexos

1. Faça a pesquisa em profundidade em G e calcule o tempo de finalização em cada vértice u;

2. Gere o grafo transposto G^T (grafo dual) do grafo G.

3. Faça a pesquisa em profundidade em G^T, mas considerando os vértices acessíveis na ordem decrescente ao seu tempo de finalização encontrado no passo 1.

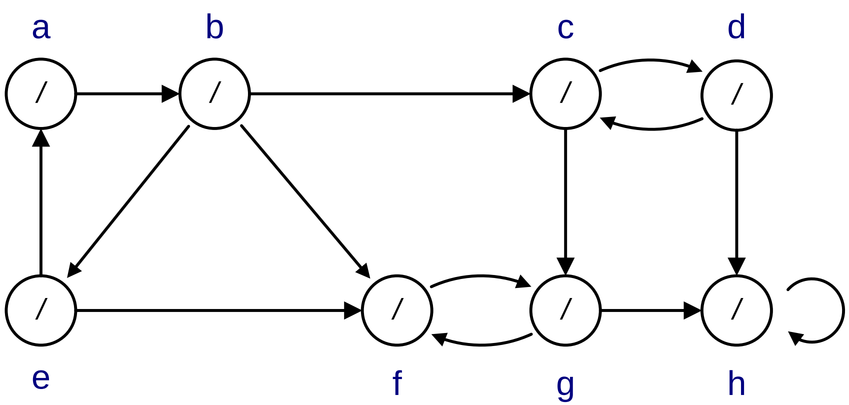
Cada árvore da floresta primeiro em profundidade encontrada no passo 3, corresponde a um componente fortemente conexo de G.

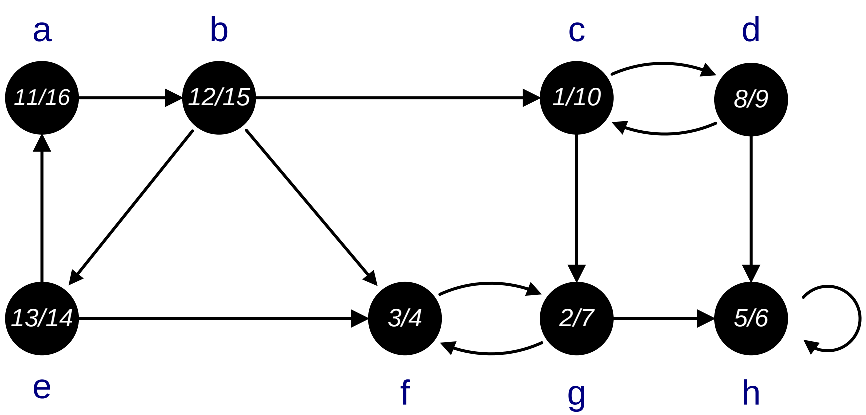
Vamos fazer um acompanhamento...

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

*Retomando ao final do do **passo 1***





Passo 1: Aplique a DFS no grafo original G=(V,A)

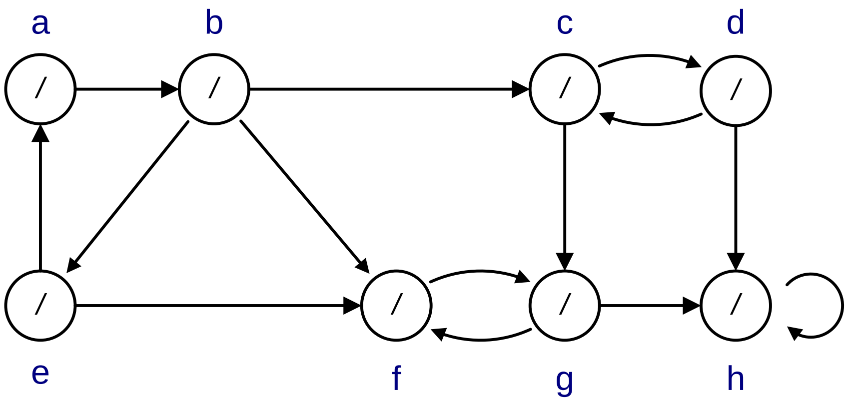
Lista em ordem decrescente ao tempo de finalização:
[a, b, e, c, d, g, h, f]

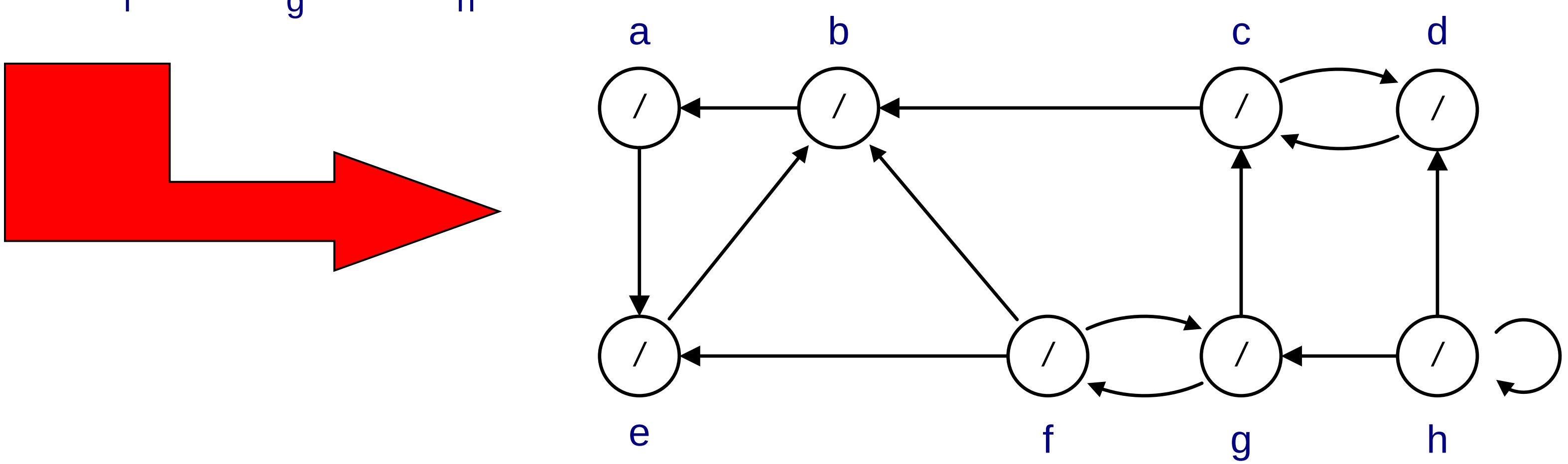
Armazenar para usar no passo 3.

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 2: Gere G^T





Lista em ordem decrescente ao tempo de finalização:
[a, b, e, c, d, g, h, f]

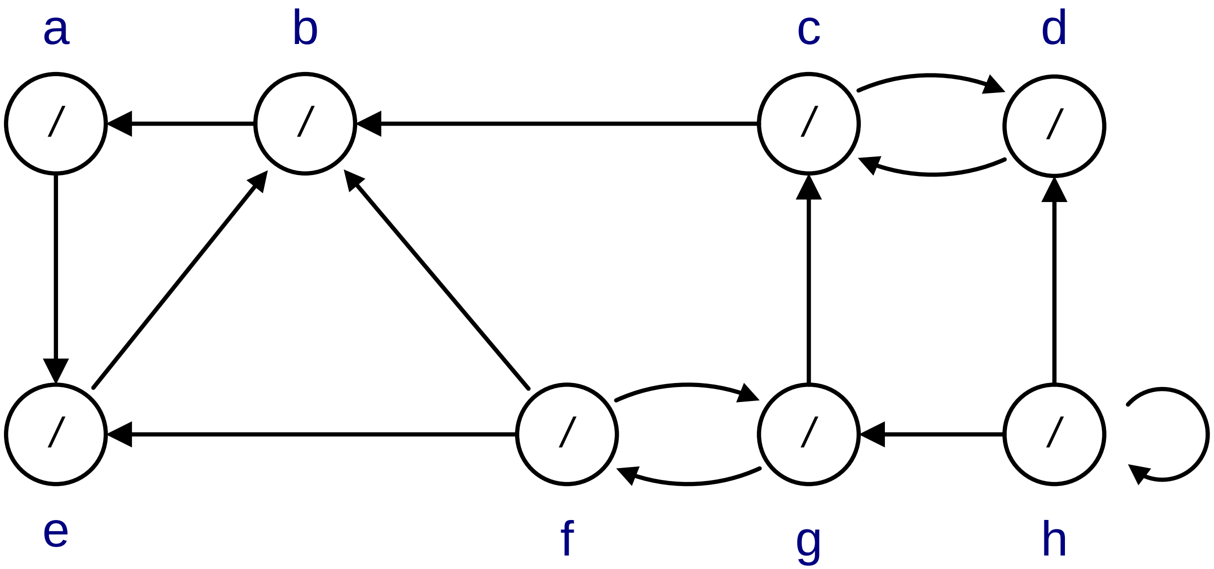
Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Capturando primeiro da lista de vértices disponíveis a ser explorado*

G^T



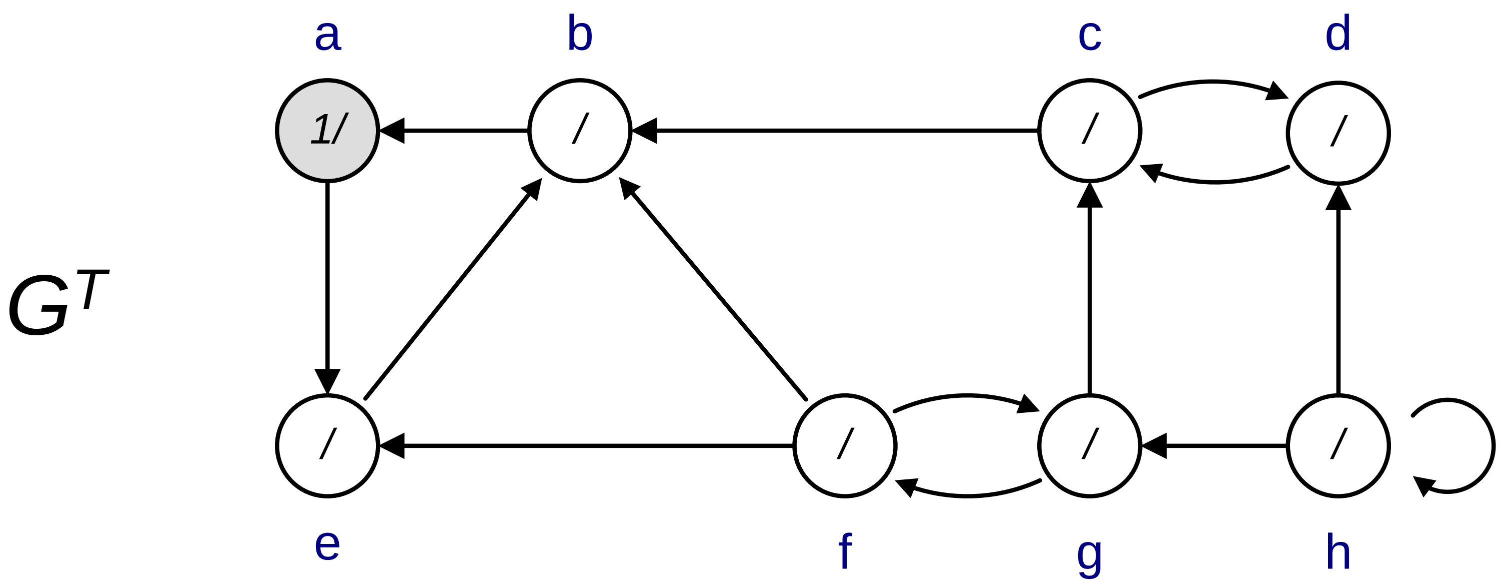
Lista em ordem decrescente ao tempo de finalização:
[a, b, e, c, d, g, h, f]

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice a é encontrado.*



Lista em ordem decrescente ao tempo de finalização:
[b, e, c, d, g, h, f]

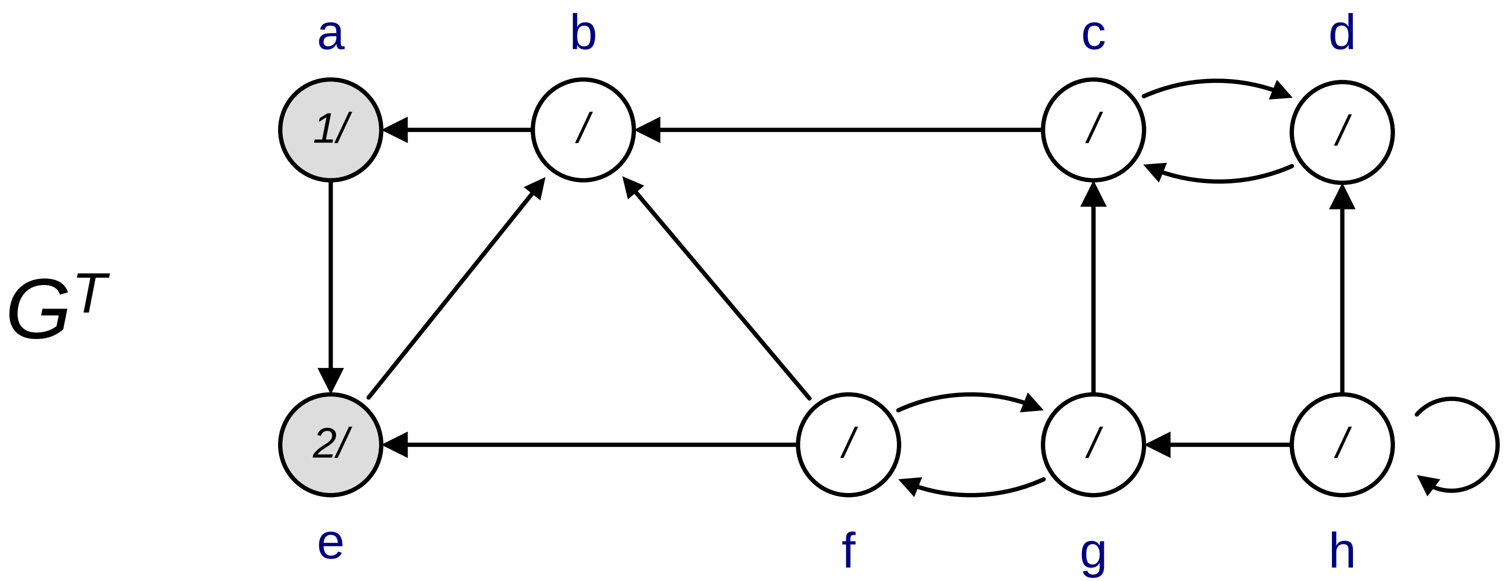
Contador = 1

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice e é encontrado.*



Lista em ordem decrescente ao tempo de finalização:
[b, e, c, d, g, h, f]

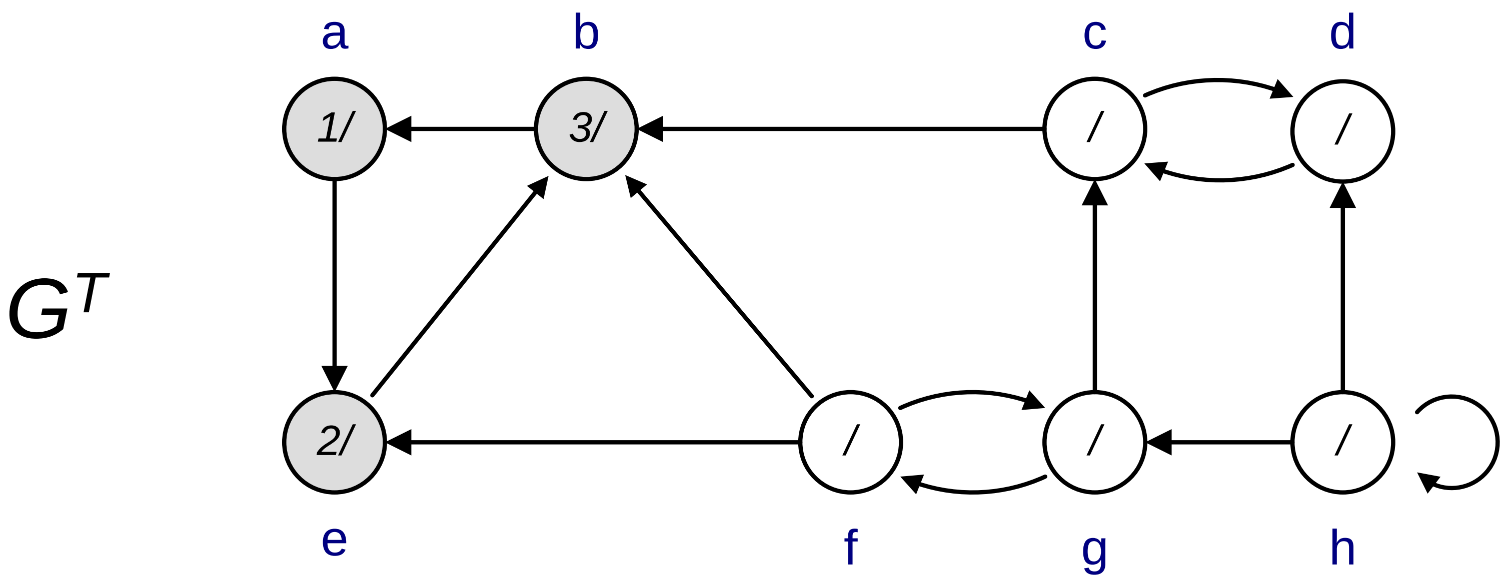
Contador = 2

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice b é encontrado.*



Lista em ordem decrescente ao tempo de finalização:
[b, e, c, d, g, h, f]

Contador = 3

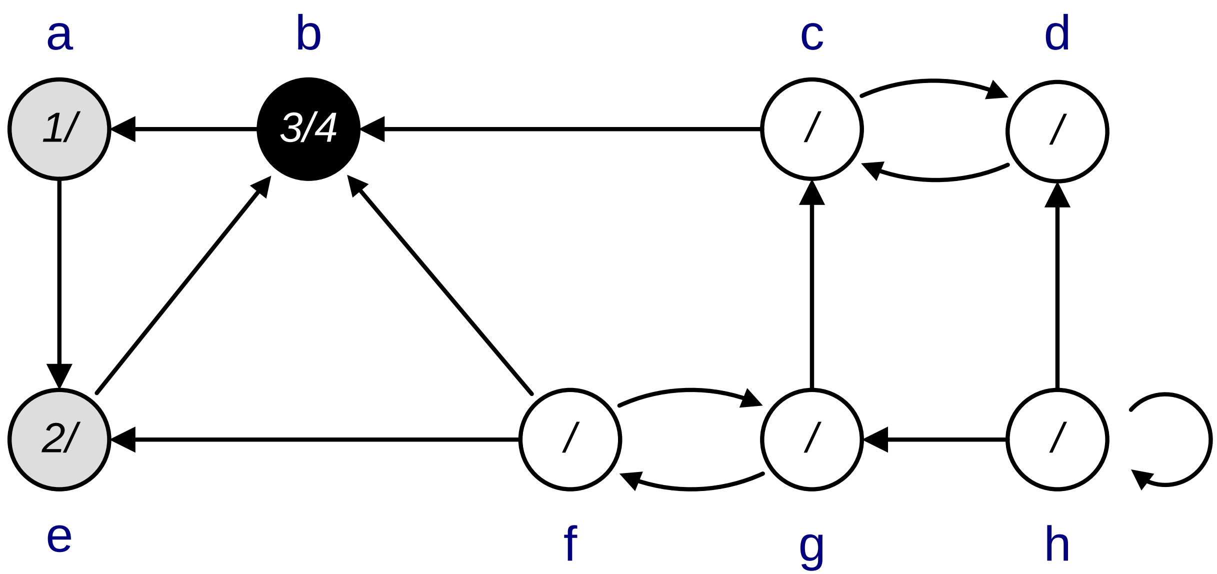
Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice b é finalizado.*

G^T



Lista em ordem decrescente ao tempo de finalização:
[b, e, c, d, g, h, f]

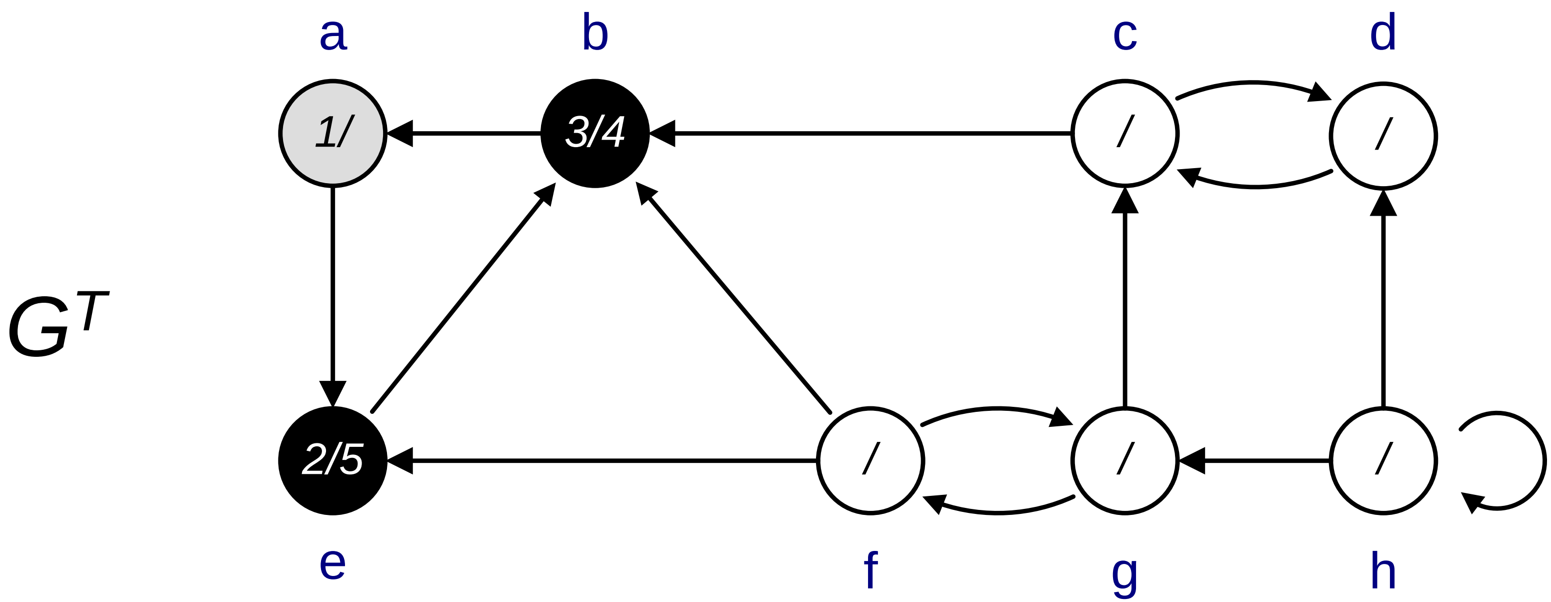
Contador = 4

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice e é finalizado.*



Lista em ordem decrescente ao tempo de finalização:
[b, e, c, d, g, h, f]

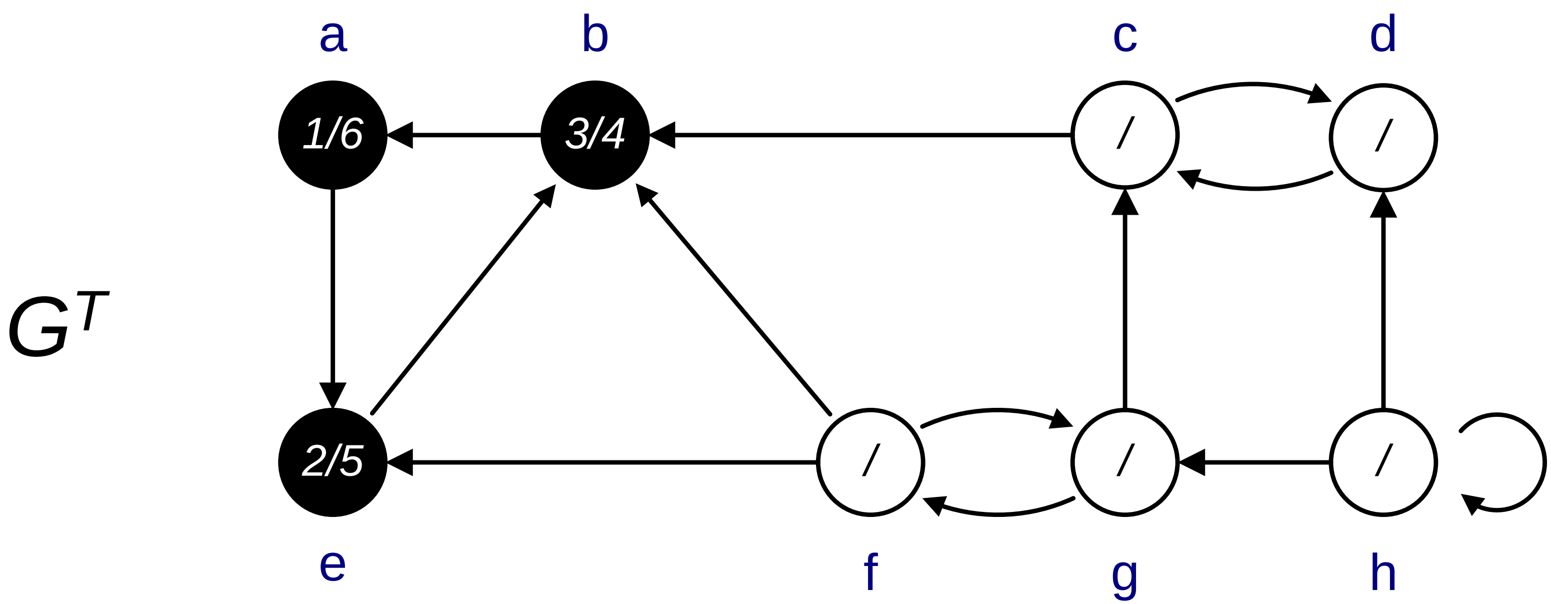
Contador = 5

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice a é finalizado.*



Lista em ordem decrescente ao tempo de finalização:
[b, e, c, d, g, h, f]

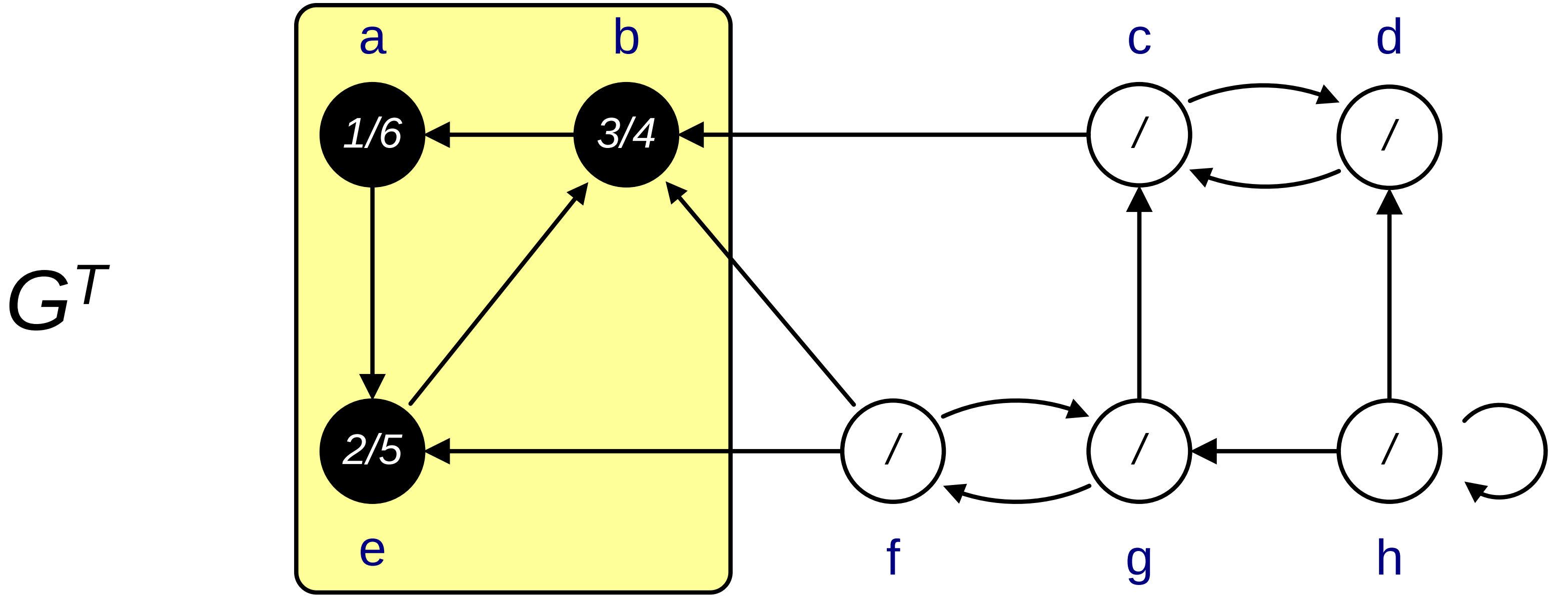
Contador = 6

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Neste momento a busca em G^T não possui mais caminhamento, então os vértices encontrados com raiz no vértice a formam um componente fortemente conexo em G.*



Lista em ordem decrescente ao tempo de finalização:
[b, e, c, d, g, h, f]

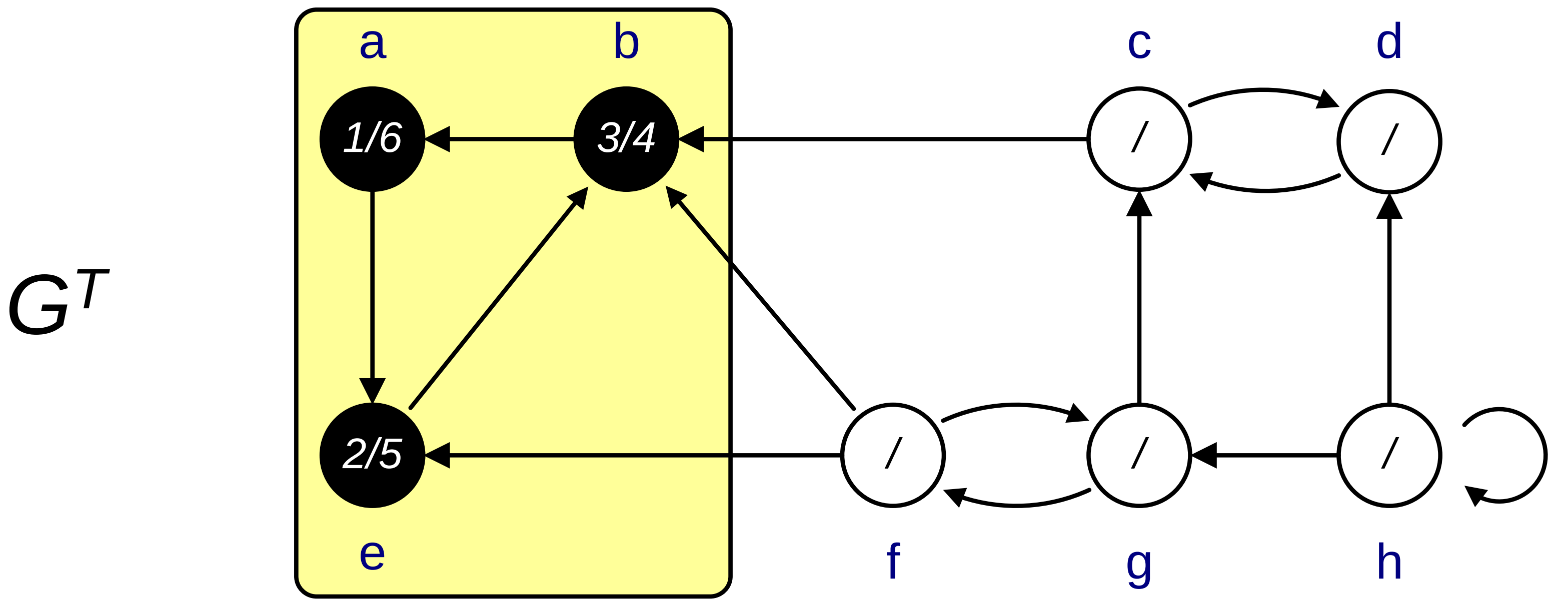
Contador = 6

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação: capturando o próximo vértice não visitado da lista. Vértice c.*



Lista em ordem decrescente ao tempo de finalização:
[b, e, c, d, g, h, f]

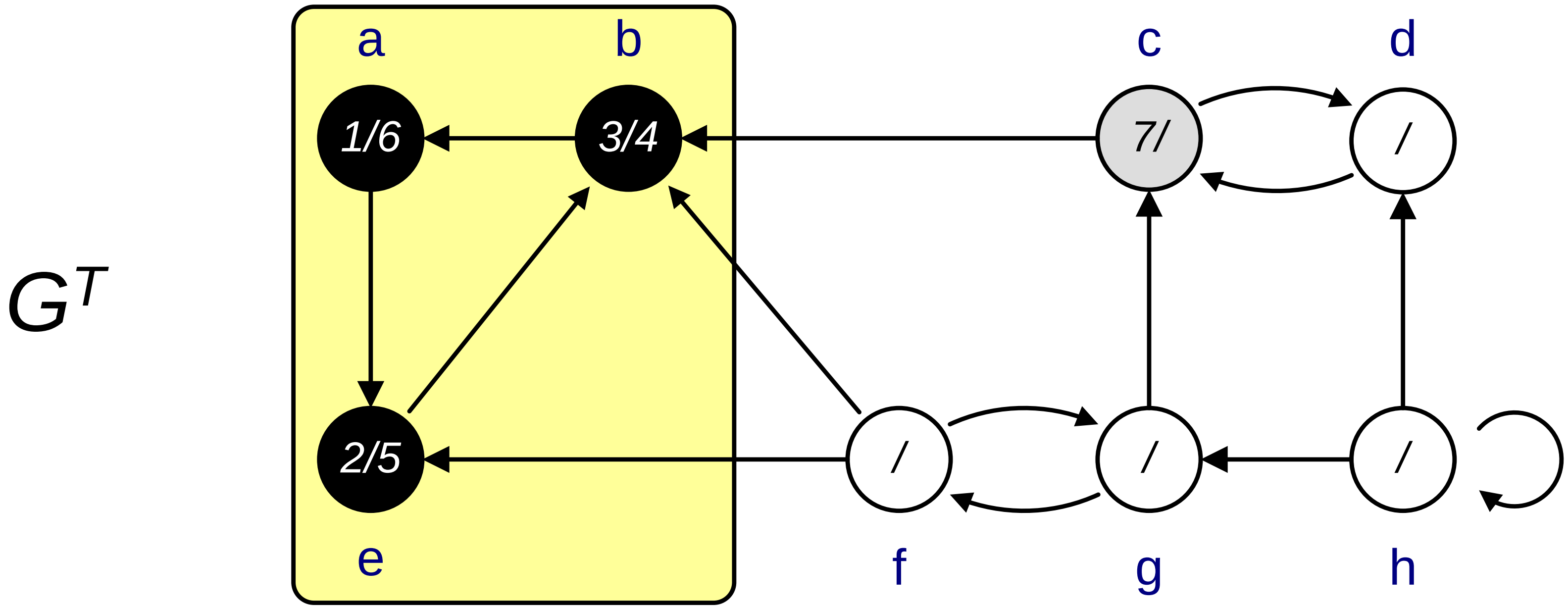
Contador = 6

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice c foi encontrado.*



Lista em ordem decrescente ao tempo de finalização:
[d, g, h, f]

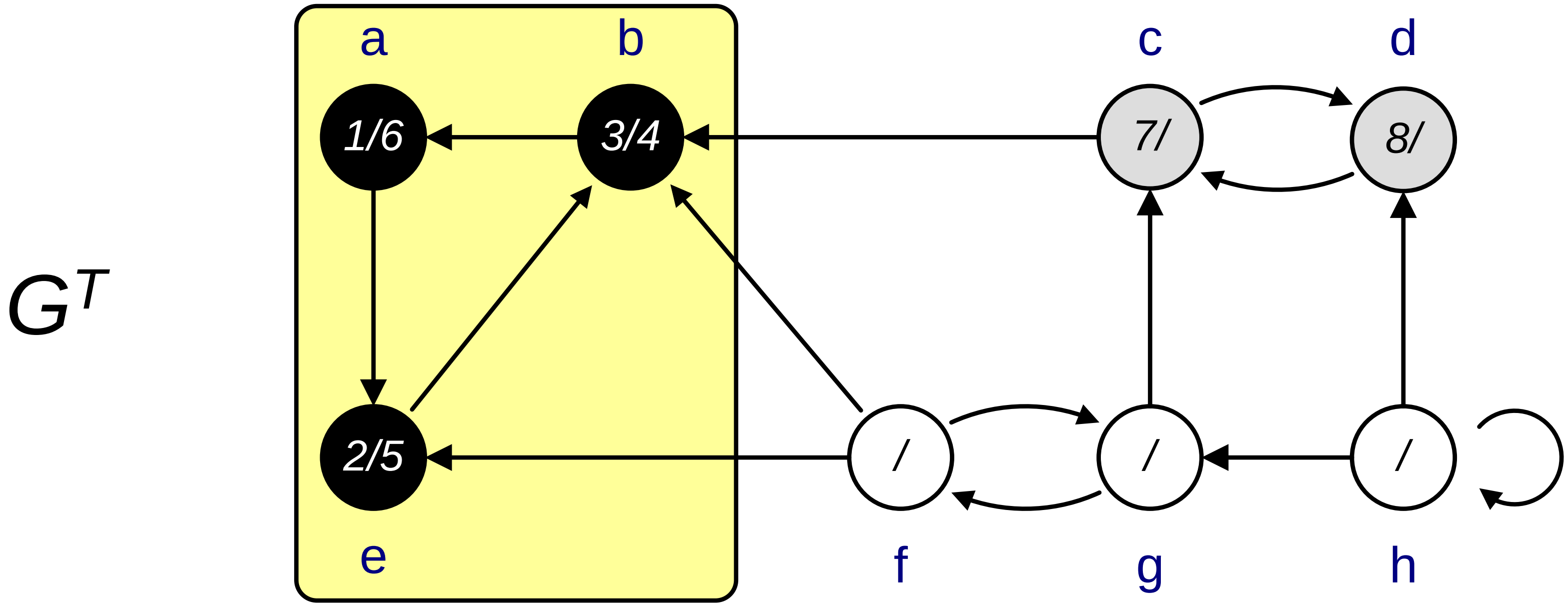
Contador = 7

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice d foi encontrado.*



Lista em ordem decrescente ao tempo de finalização:
[d, g, h, f]

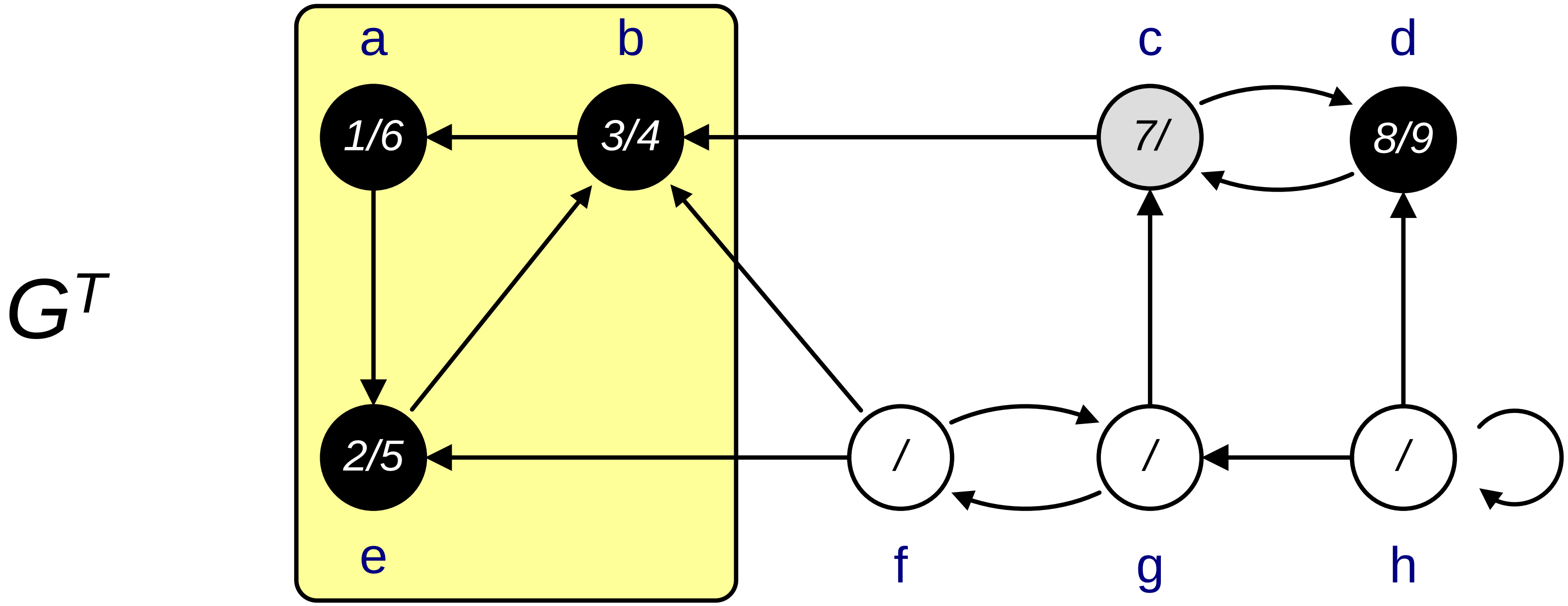
Contador = 8

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice d foi finalizado.*



Lista em ordem decrescente ao tempo de finalização:
[d, g, h, f]

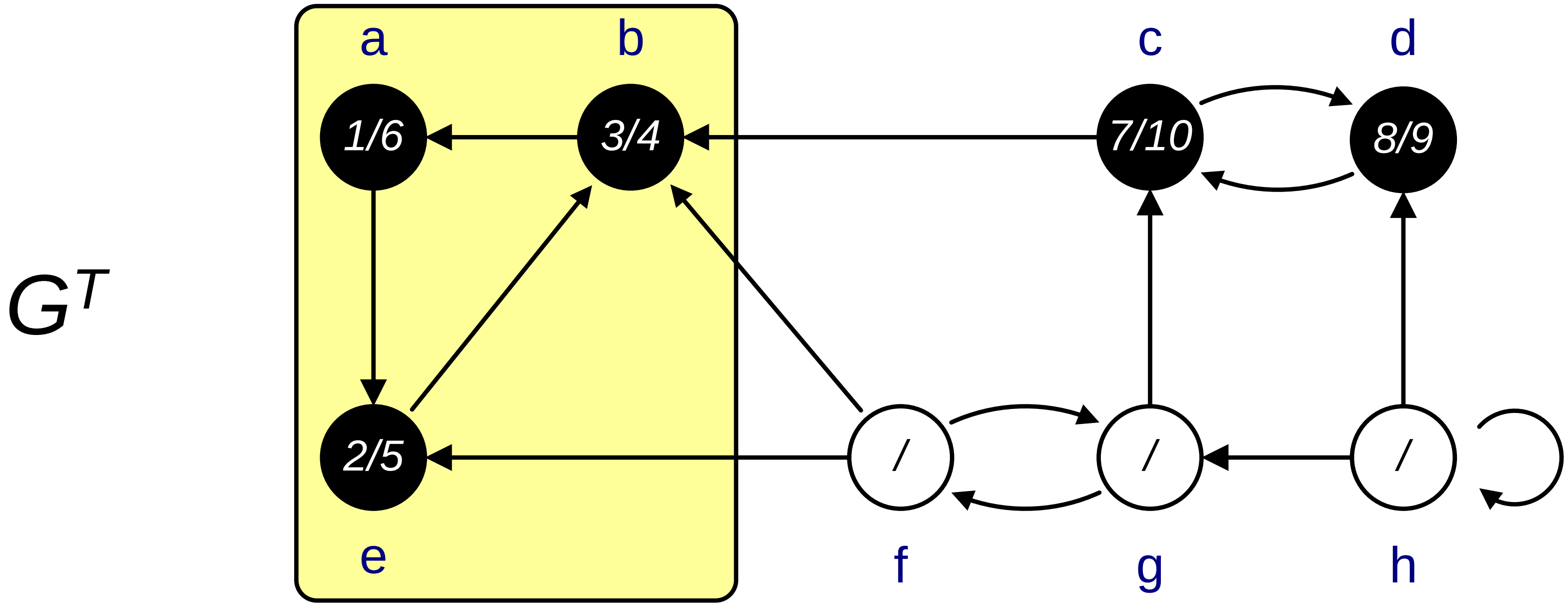
Contador = 9

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- Última ação: Vértice c foi finalizado.



Lista em ordem decrescente ao tempo de finalização:
[d, g, h, f]

Contador = 10

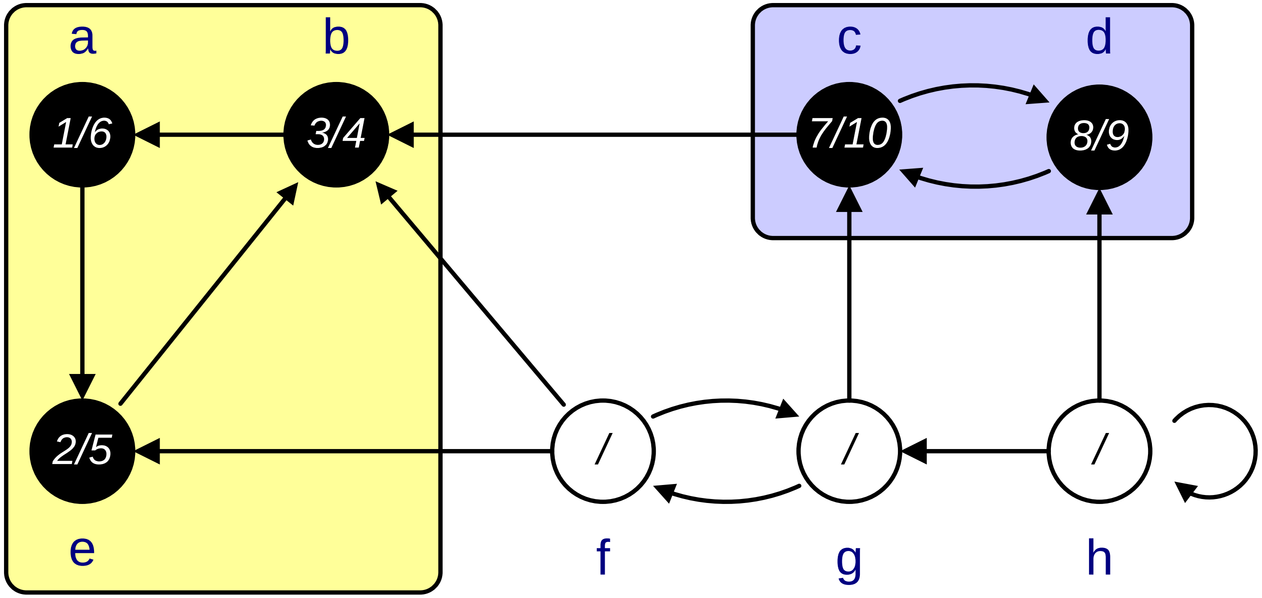
Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Neste momento a busca em G^T não possui mais caminhamento, então os vértices encontrados com raiz no vértice c formam um componente fortemente conexo em G.*

G^T



Lista em ordem decrescente ao tempo de finalização:
[b, e, c, d, g, h, f]

Contador = 10

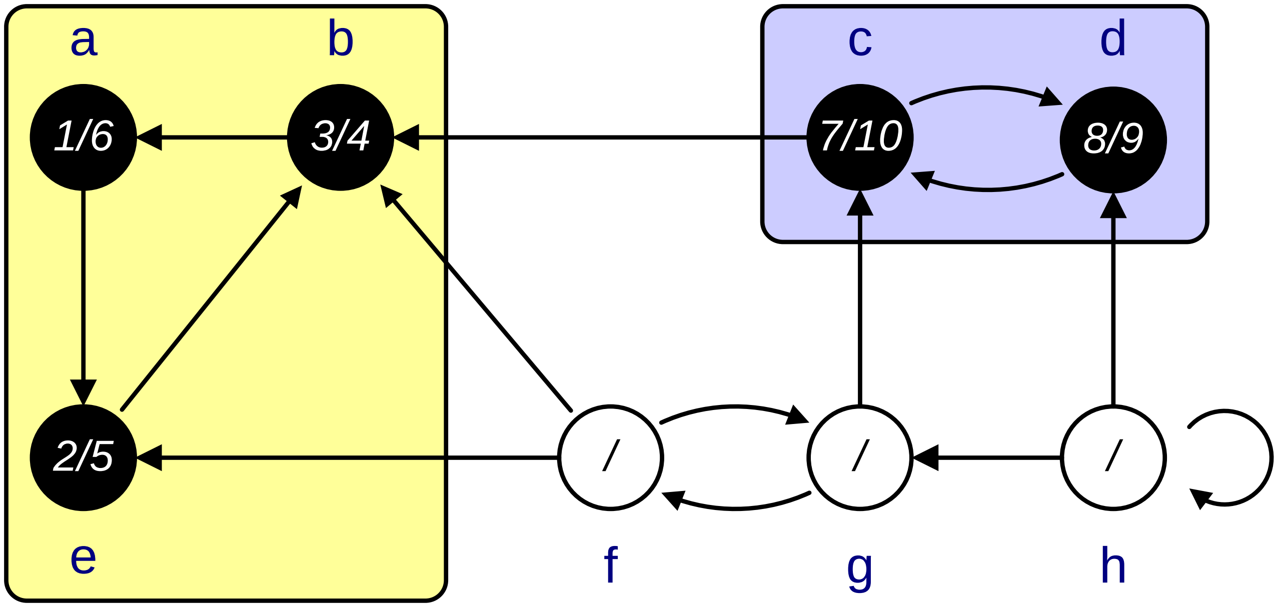
Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação: capturando o próximo vértice não visitado da lista. Vértice g.*

G^T



Lista em ordem decrescente ao tempo de finalização:
[d, g, h, f]

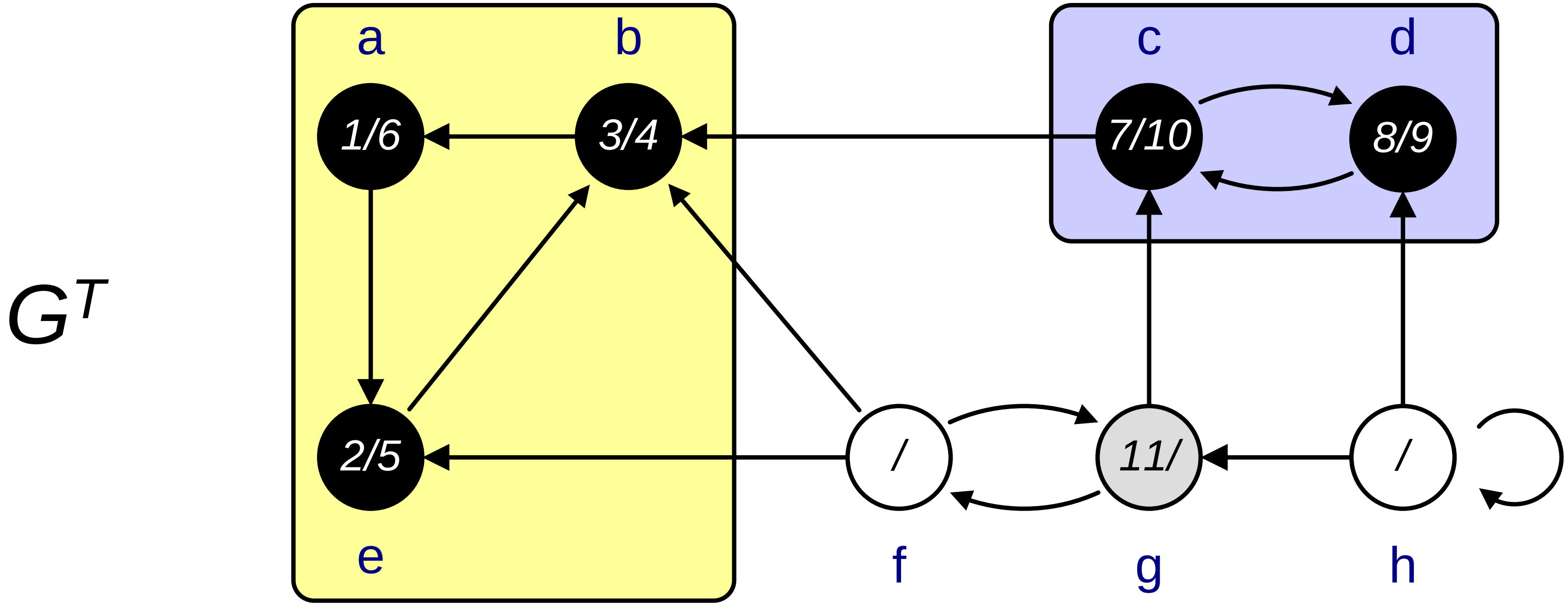
Contador = 10

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice g foi encontrado.*



Lista em ordem decrescente ao tempo de finalização:
[h, f]

Contador = 11

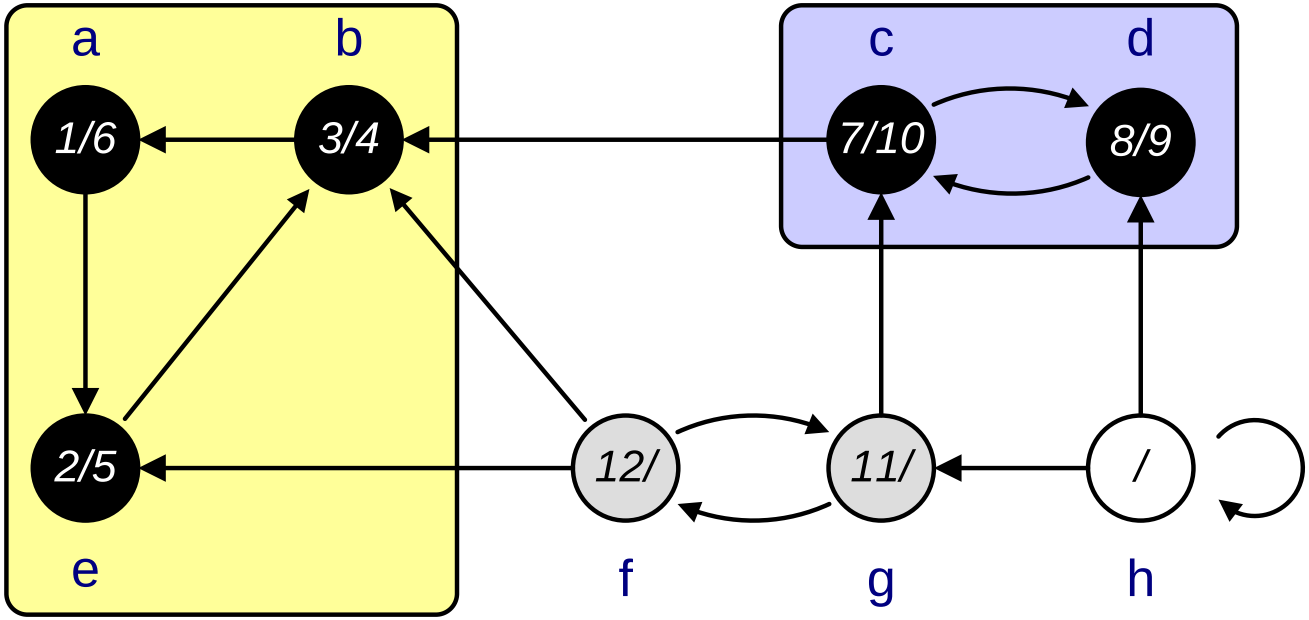
Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice f foi encontrado.*

G^T



Lista em ordem decrescente ao tempo de finalização:
[h, f]

Contador = 12

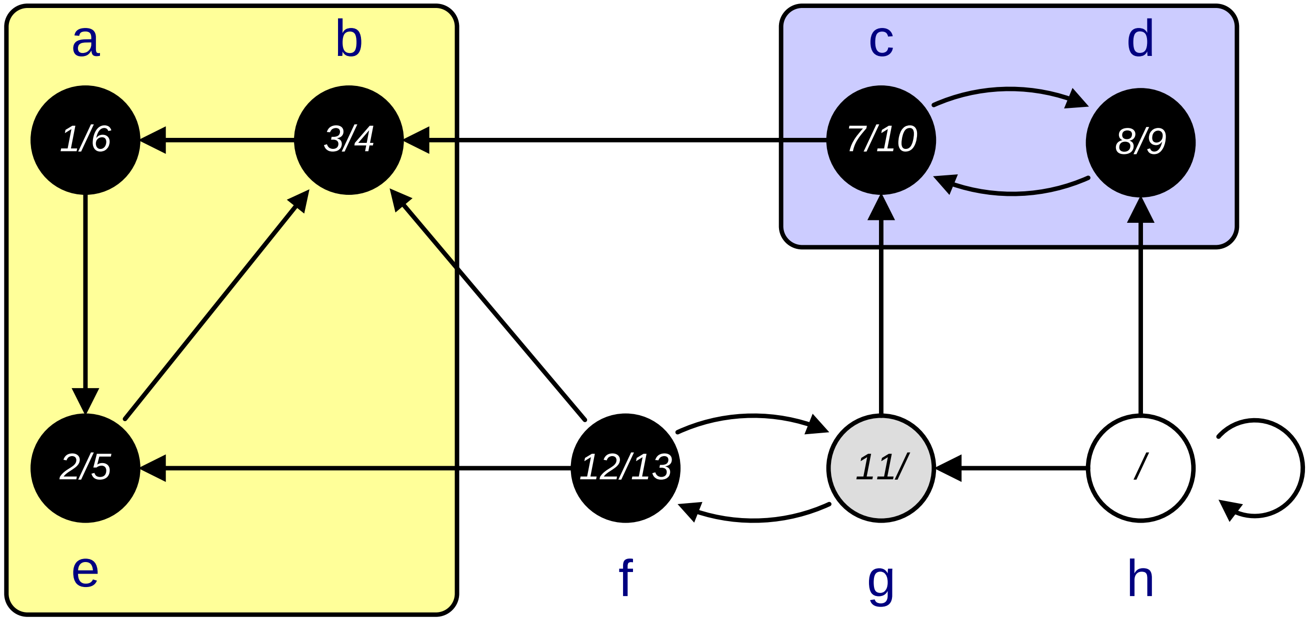
Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice f foi finalizado.*

G^T



Lista em ordem decrescente ao tempo de finalização:
[h, f]

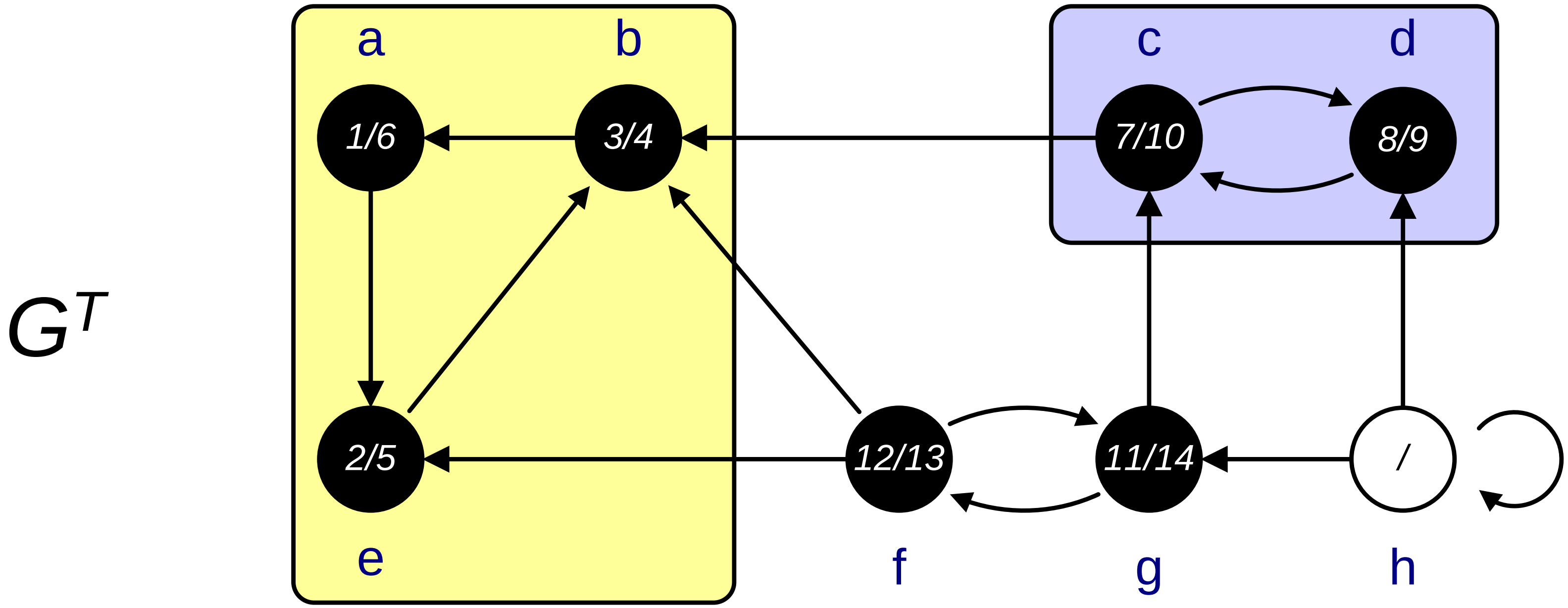
Contador = 13

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice g foi finalizado.*



Lista em ordem decrescente ao tempo de finalização:
[h, f]

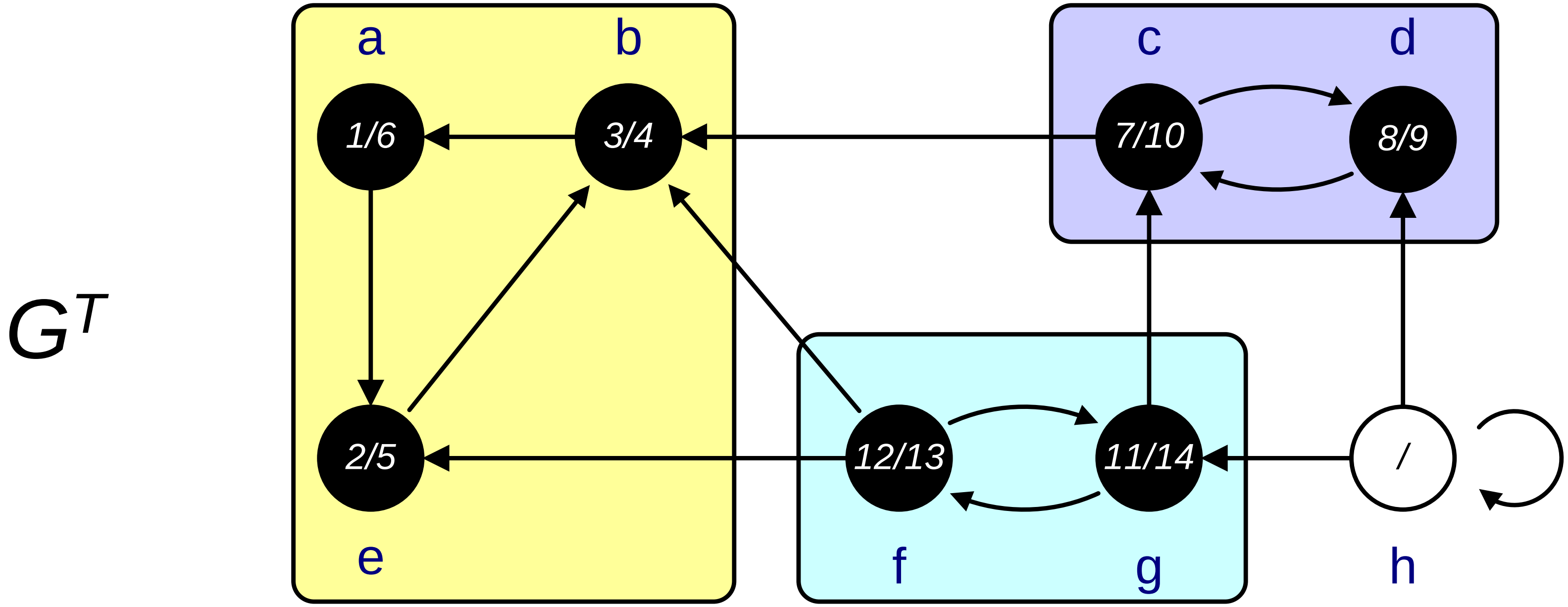
Contador = 14

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Neste momento a busca em G^T não possui mais caminhamento, então os vértices encontrados com raiz no vértice g formam um componente fortemente conexo em G.*



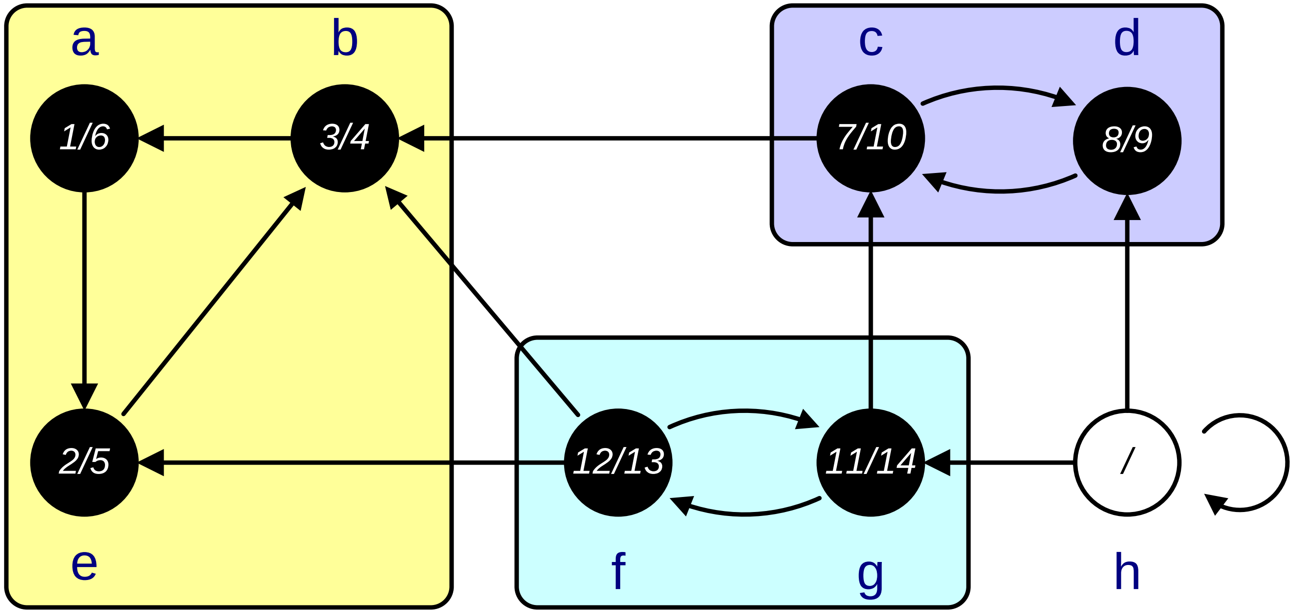
Lista em ordem decrescente ao tempo de finalização:
[h, f]

Contador = 14

Redução por componentes fortemente conexos
Algoritmo: Componentes fortemente conexos
Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação: capturando o próximo vértice não visitado da lista. Vértice h.*

G^T



Lista em ordem decrescente ao tempo de finalização:
[h, f]

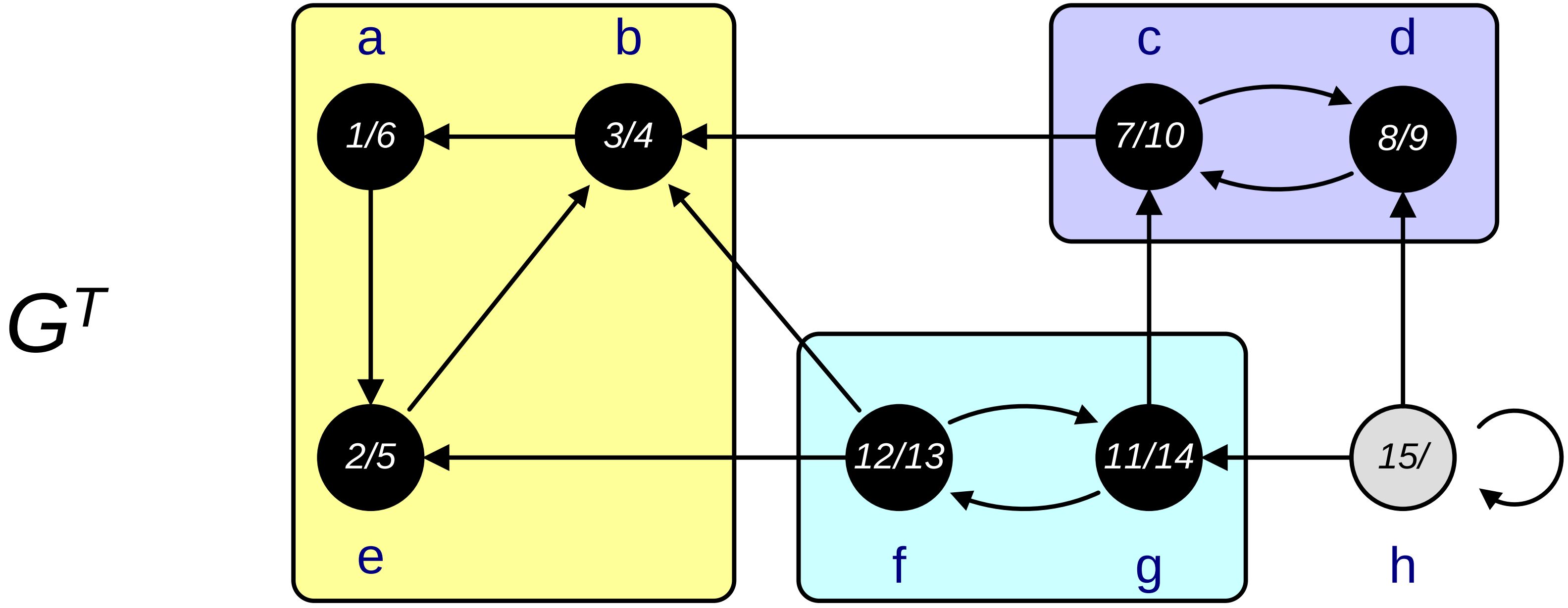
Contador = 14

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice h foi encontrado.*



Lista em ordem decrescente ao tempo de finalização:
[f]

Contador = 15

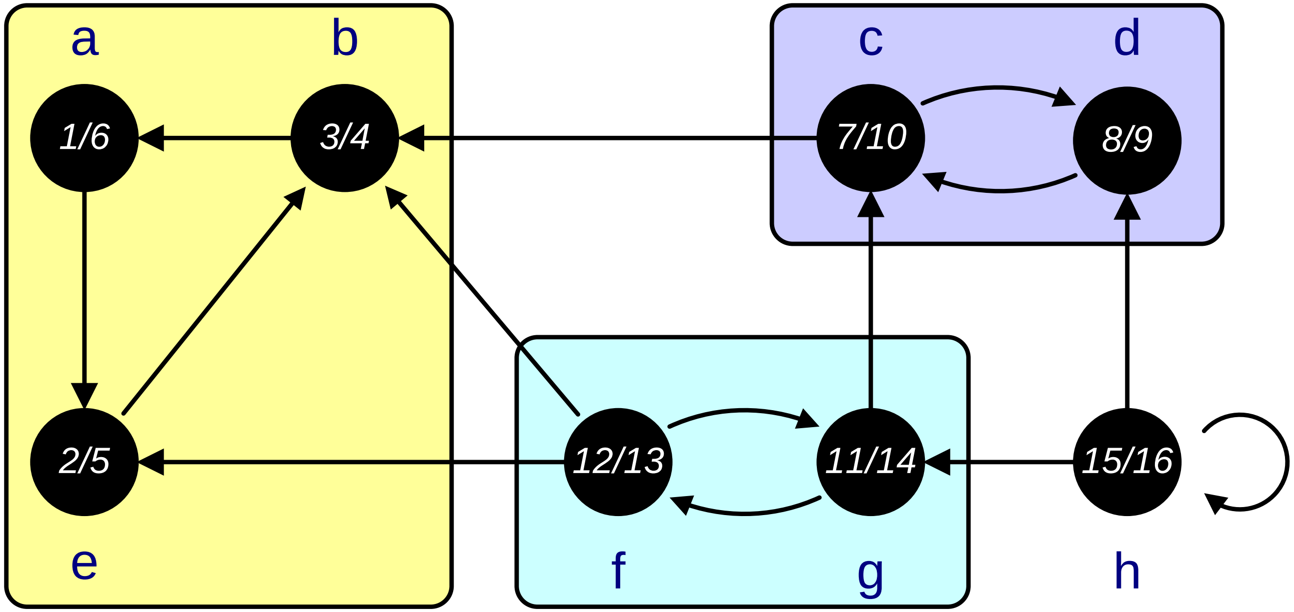
Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Última ação:* *Vértice h foi finalizado.*

G^T



Lista em ordem decrescente ao tempo de finalização:
[f]

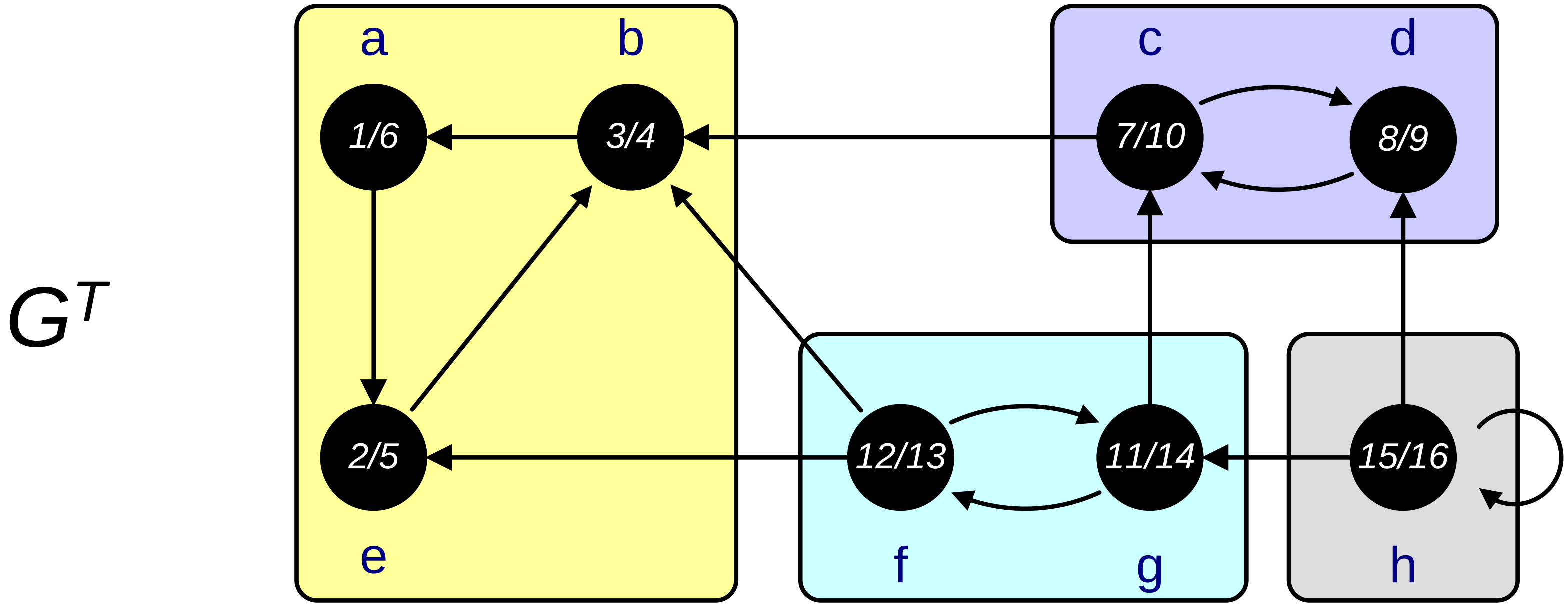
Contador = 16

Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Passo 3: Efetue a busca em profundidade em G^T ordem armazenada

- *Neste momento a busca em G^T não possui mais caminhamento, então os vértices encontrados com raiz no vértice h formam um componente fortemente conexo em G.*



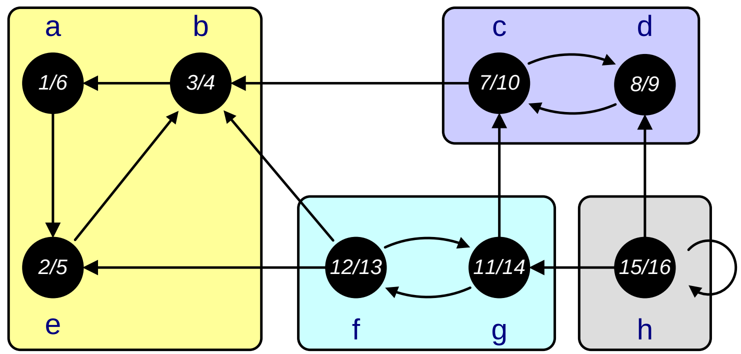
Lista em ordem decrescente ao tempo de finalização:
[f]

Contador = 16

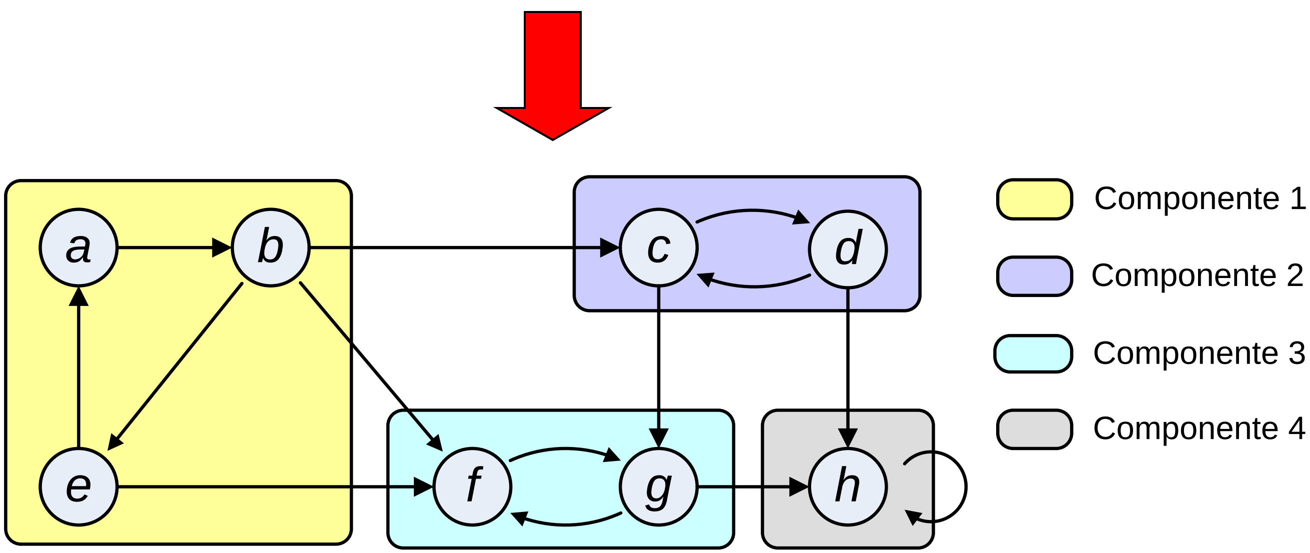
Redução por componentes fortemente conexos
Algoritmo: Componentes fortemente conexos
Lembrete

- *Os componentes encontrados são referentes a G, e não a G^T.*

G^T



G



Redução por componentes fortemente conexos

Algoritmo: Componentes fortemente conexos

Discussão da complexidade do algoritmo

- Busca em profundidade sobre G:
  $\Theta(|V|+|A|)$

- Cálculo de G^T:
  $\Theta(|V|+|A|)$

- Busca em profundidade sobre G^T:
  $\Theta(|V|+|A|)$

- Assim, a complexidade de tempo de todo o algoritmo é
  $\Theta(|V|+|A|)$