DCC206 – Algoritmos 1

Aula 04 – Aplicações de caminhamento em grafos – Parte 2

Professor Renato Vimieiro

DCC/ICEx/UFMG

Introdução

- Nessa aula, vamos discutir uma outra aplicação dos algoritmos de caminhamento para verificar propriedades de grafos
- Seguimos com aplicações em grafos direcionados
- Vamos discutir como encontrar componentes fortemente conexos em grafos direcionados

Introdução

- Anteriormente, vimos que cada árvore iniciada por uma DFS indicava um novo componente conexo em um grafo não-direcionado
- No caso dos grafos direcionados, o problema se torna um pouco mais complexo, já que dois vértices ligados por uma arestas não são necessariamente mutuamente alcançáveis
- Assim, precisamos refletir um pouco mais sobre como esses algoritmos podem ser usados com o propósito de encontrar os componentes fortemente conexos (CFC)

Componentes fortemente conexos

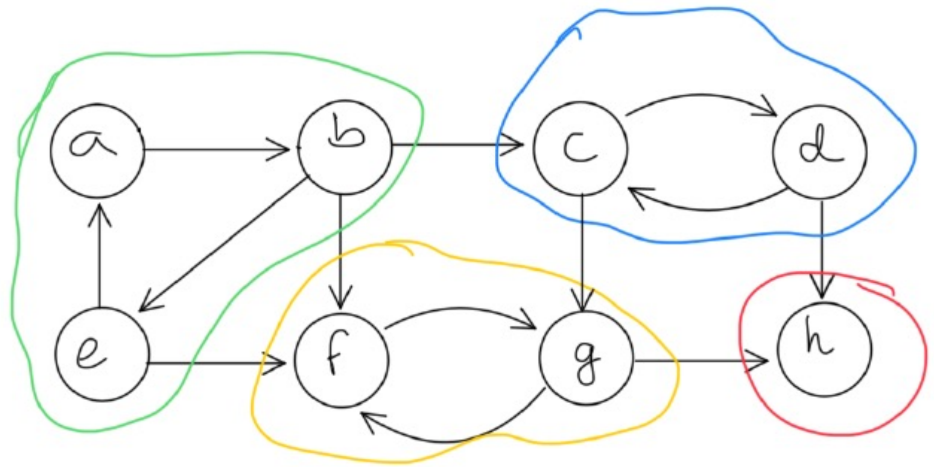
- Lembrando, um componente fortemente conexo é um subconjunto maximal de vértices $C \subseteq V$, tal que todo par de vértices $u, v \in C$ é mutuamente alcançável
- Essa definição de CFC induz uma relação de equivalência entre os vértices de G
  - Reflexividade: todo vértice alcança a si mesmo
  - Transitividade: se v é mutuamente alcançável a partir de u, e u é mutuamente alcançável de z, então v é mutuamente alcançável de z, basta usar u como intermediário
  - Simetria: u e v devem ser mutuamente alcançáveis por definição
- Dessa forma, os componentes fortemente conexos formam uma partição do conjunto de vértices

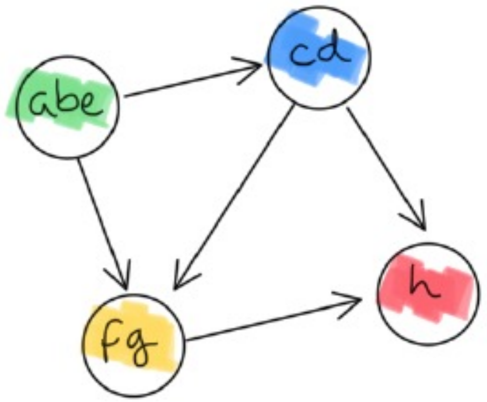
Componentes fortemente conexos

- Agora vamos tentar desenhar um algoritmo que encontre os CFCs de um grafo direcionado
- Pela definição, a DFS poderia ser usada para encontrar os alcançáveis a partir de um vértice v
- Intuitivamente, poderíamos rodar a busca no grafo reverso para avaliar aqueles que alcançam v e, assim, determinar os componentes
- No entanto, qual deve ser a escolha do vértice inicial? Há garantia para tal escolha?
- Veremos que essa decisão é promissora, bastando alguns pequenos ajustes que nos garantem a correção do algoritmo

Grafo de componentes

- Antes de iniciarmos a discussão do algoritmo, vamos analisar algumas propriedades do problema
- Um grafo de componentes de um grafo direcionado G tem como vértices os CFCs de G e as arestas indicam se um componente alcança outro
  - O grafo é denotado por $G^{CFC} = (V^{CFC}, E^{CFC})$



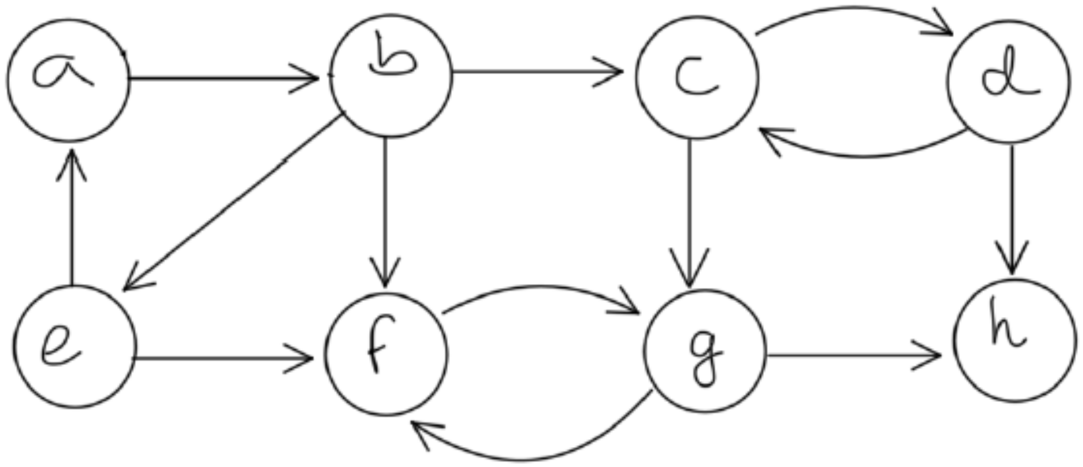


Grafo de componentes

- **Lema 1**: O grafo de componentes é um DAG.
- **Prova** (intuição): se existisse um ciclo, os vértices dos diferentes componentes envolvidos no ciclo seriam mutuamente alcançável, contradizendo a hipótese de que os CFCs são maximais.

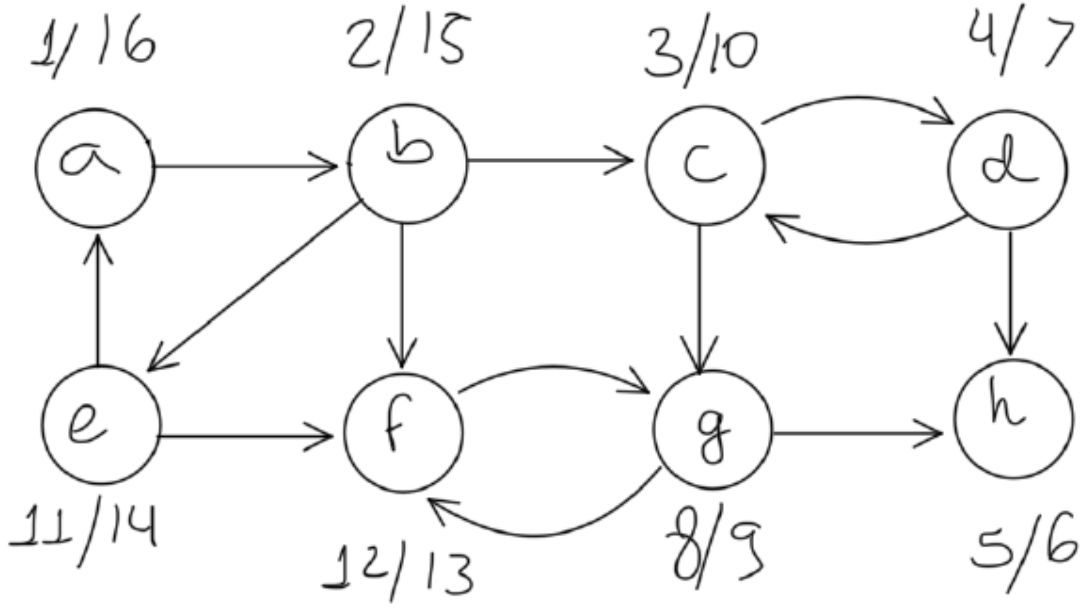
Algoritmo de Kosaraju-Sharir

- Antes de tentarmos efetivamente desenhar um algoritmo para encontrar os CFCs, vamos verificar o que acontece com um raciocínio análogo ao de grafos não-direcionados
- Vamos executar uma DFS no grafo abaixo e analisar o resultado



Algoritmo de Kosaraju-Sharir

- O que podemos dizer sobre o último vértice finalizado na DFS?
  - Todos os vértices explorados na árvore desse vértice são alcançáveis a partir dele



Algoritmo de Kosaraju-Sharir

- Se invertermos as arestas do grafo e executarmos a DFS a partir do último finalizado, encontraremos os vértices que o alcançam no grafo original
- Assim, descobriremos o CFC desse vértice
- Se seguirmos o mesmo raciocínio, iniciando a busca a partir do último finalizado ainda não marcado, encontraremos os CFCs para cada nova árvore
- Logo, os vértices pertencentes a cada árvore são associados a um novo CFC como no caso não-direcionado
- Essa é a ideia implementada pelo algoritmo de Kosaraju-Sharir

Algoritmo de Kosaraju-Sharir

1. Rode a DFS a partir de um vértice arbitrário e compute os tempos de finalização dos vértices de G
2. Obtenha G^R
3. Rode a DFS em G^R escolhendo vértices em ordem decrescente do tempo de finalização obtido em 1
4. Retorne as árvores obtidas em 3 como CFCs separados

Algoritmo de Kosaraju-Sharir

- O algoritmo explora os componentes na ordem topológica do grafo de componentes
- O custo do algoritmo continua sendo O(|V| + |E|)
  - São executadas duas DFS
  - O tempo para computar o grafo reverso é linear no tamanho do grafo original
- Agora devemos provar a corretude do método

Algoritmo de Kosaraju-Sharir

- Vamos estender o conceito de tempo de descoberta e de finalização para componentes
- d(C) = min {u.d | u in C}
- f(C) = max {u.f | u in C}
- As funções retornam o menor tempo de descoberta e maior tempo de finalização entre os vértices de um CFC

Algoritmo de Kosaraju-Sharir

- **Lema 2**: sejam C e D dois CFC distintos de G. Se $(u,v) \in E$, $u \in C$ e $v \in D$, então $f(C) > f(D)$
- **Prova**: Suponha que (u,v) in E, $u \in C$ e $v \in D$. Devemos considerar dois casos:

1. $d(C) < d(D)$: seja x tal que $d(C) = x.d$. Nesse caso, todos os vértices de C estavam desmarcados em x.d e se tornaram descendentes de x. Como $(u,v) \in E$ e $d(C) < d(D)$, os vértices de D também estavam desmarcados em x.d e se tornaram descendentes de x. Portanto, para todo $w \in D$, $w.f < x.f$. Logo, $x.f = f(C) > f(D)$.
2. $d(C) > d(D)$: seja y tal que $d(D) = y.d$. Nesse caso, no momento y.d, todos os vértices de C e D estavam desmarcados. Como D é um CFC, todos os seus vértices são descendentes de y. Logo, $f(D) = y.f$. Como $(u,v) \in E$, os vértices de C não são alcançáveis a partir de y (lema 1). Portanto, $f(D) < f(C)$.

Algoritmo de Kosaraju-Sharir

- **Corolário**: sejam C e D CFC distintos. Suponha que $(v,u) \in E^R$, $v \in D$, u in C. Então, $f(D) < f(C)$.
- O corolário diz que só podem existir arestas entre componentes se ela conectar um CFC a outro que já foi explorado antes
- Dessa forma, ao explorar os vértices de G^R em ordem decrescente de tempo de finalização, quando encontrarmos uma aresta ligando dois componentes distintos, o destino já terá sido marcado

Algoritmo de Kosaraju-Sharir

- **Teorema**: O algoritmo de Kosaraju-Sharir encontra os CFCs de um grafo direcionado.
- **Prova**: A prova será por indução no número de árvores da segunda DFS.
- O caso base é trivial.
- Agora suponha como hipótese de indução que as k primeiras árvores encontradas formam CFCs distintos.
- Seja u a raiz da (k+1)-ésima árvore. Seja C o CFC tal que u ∈ C. Como u é o primeiro a ser explorado, todos os outros vértices de C ainda estão desmarcados em u.d. Logo, eles se tornam descendentes de u. Pelo corolário do Lema 2, se a busca iniciada em u encontrar uma aresta levando a outro CFC, esse tem que ter sido explorado antes. Assim, não serão descendentes de u nessa DFS. Portanto, a busca iniciada em u encontra todos e somente os vértices de C.

Leitura

- Seção 22.5 (CLRS)