Por que quebrar um grafo?

Por que quebrar um grafo?

Tadinho do grafo!

Por que quebrar um grafo?

A idéia de quebrar um grafo (decompor é um termo mais técnico) é reduzir alguns problemas a grafos menores.

Por que quebrar um grafo?

A idéia de quebrar um grafo (decompor é um termo mais técnico) é reduzir alguns problemas a grafos menores.

Uma primeira decomposição é em componentes conexas. Praticamente tudo interessante sobre grafos acontece em grafos conexos.

Por que quebrar um grafo?

A idéia de quebrar um grafo (decompor é um termo mais técnico) é reduzir alguns problemas a grafos menores.

Uma primeira decomposição é em componentes conexas. Praticamente tudo interessante sobre grafos acontece em grafos conexos.

Vamos ver um tipo de decomposição menos óbvio, mas muito útil.

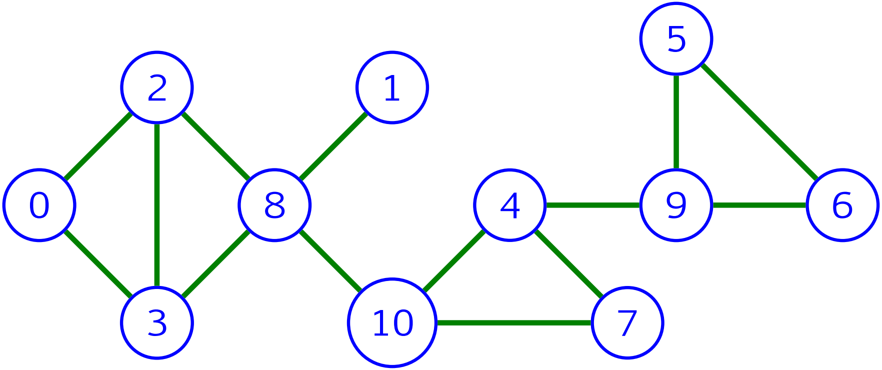
Prelúdio: pontes

Prelúdio: pontes

Uma aresta de um grafo conexo é uma ponte se sua remoção desconecta o grafo.

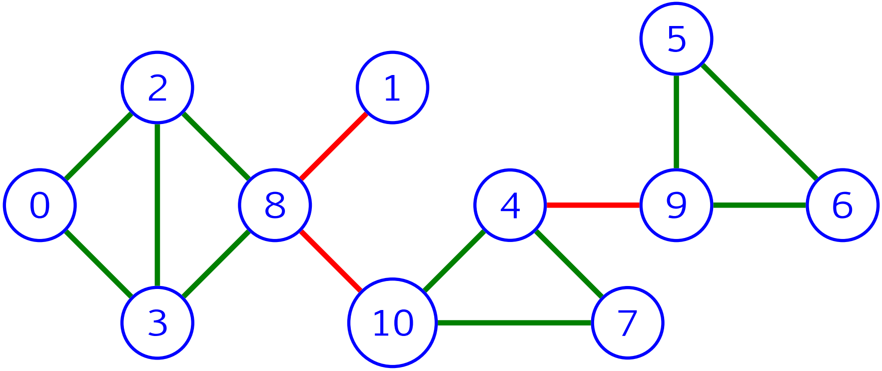
Prelúdio: pontes

Uma aresta de um grafo conexo é uma ponte se sua remoção desconecta o grafo.



Prelúdio: pontes

Uma aresta de um grafo conexo é uma ponte se sua remoção desconecta o grafo.



Caracterização de pontes

Proposição

Seja G um grafo conexo, e e uma aresta. Então, são equivalentes:

Caracterização de pontes

Proposição

Seja G um grafo conexo, e e uma aresta. Então, são equivalentes:

1. *e é uma ponte.*

Caracterização de pontes

Proposição

Seja G um grafo conexo, e e uma aresta. Então, são equivalentes:

1. *e é uma ponte.*
2. *e não está em nenhum ciclo.*

Caracterização de pontes

Proposição

Seja G um grafo conexo, e e uma aresta. Então, são equivalentes:

1. *e é uma ponte.*
2. *e não está em nenhum ciclo.*
3. *Toda árvore geradora de G contém e.*

Caracterização de pontes

Proposição

Seja G um grafo conexo, e e uma aresta. Então, são equivalentes:

1. *e é uma ponte.*
2. *e não está em nenhum ciclo.*
3. *Toda árvore geradora de G contém e.*

Corolário

Uma árvore é um grafo conexo em que todas as arestas são pontes.

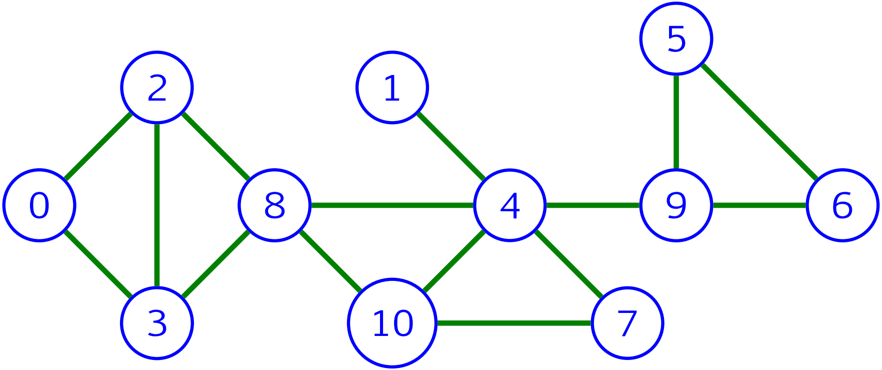
Articulações

Articulações

Uma articulação (=*articulation point*) num grafo conexo (também vértice de corte = *cut vertex*) é um vértice cuja remoção desconecta o grafo.

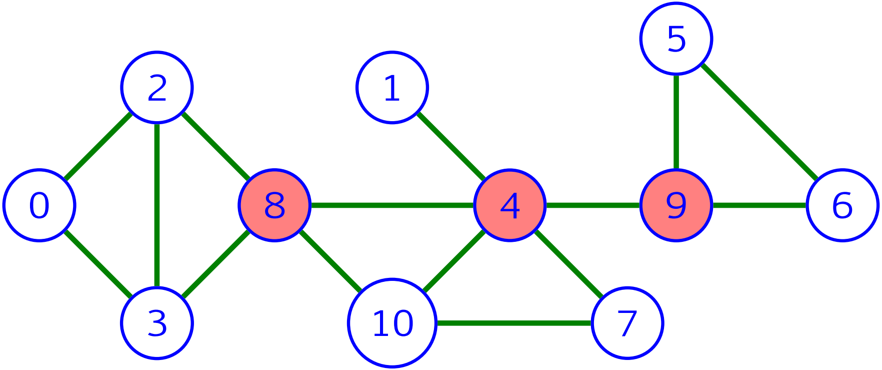
Articulações

Uma articulação (=*articulation point*) num grafo conexo (também vértice de corte = *cut vertex*) é um vértice cuja remoção desconecta o grafo.



Articulações

Uma articulação (=*articulation point*) num grafo conexo (também vértice de corte = *cut vertex*) é um vértice cuja remoção desconecta o grafo.



Grafos biconexos

Teorema

Seja G um grafo conexo. As propriedades seguintes são equivalentes:

Grafos biconexos

Teorema

Seja G um grafo conexo. As propriedades seguintes são equivalentes:

1. *A remoção de um vértice não desconecta o grafo (ou seja, G não tem vértices de corte).*

Grafos biconexos

Teorema

Seja G um grafo conexo. As propriedades seguintes são equivalentes:

1. *A remoção de um vértice não desconecta o grafo (ou seja, G não tem vértices de corte).*
2. *G tem só uma aresta ou para cada dois vertices de G existem dois caminhos simples entre eles sem vértice em comum além das pontas.*

Grafos biconexos

Teorema

Seja G um grafo conexo. As propriedades seguintes são equivalentes:

1. *A remoção de um vértice não desconecta o grafo (ou seja, G não tem vértices de corte).*
2. *G tem só uma aresta ou para cada dois vertices de G existem dois caminhos simples entre eles sem vértice em comum além das pontas.*
3. *Cada duas arestas estão em um ciclo.*

Grafos biconexos

Teorema

Seja G um grafo conexo. As propriedades seguintes são equivalentes:

1. *A remoção de um vértice não desconecta o grafo (ou seja, G não tem vértices de corte).*
2. *G tem só uma aresta ou para cada dois vertices de G existem dois caminhos simples entre eles sem vértice em comum além das pontas.*
3. *Cada duas arestas estão em um ciclo.*

Grafos biconexos

Teorema

Seja G um grafo conexo. As propriedades seguintes são equivalentes:

1. *A remoção de um vértice não desconecta o grafo (ou seja, G não tem vértices de corte).*
2. *G tem só uma aresta ou para cada dois vertices de G existem dois caminhos simples entre eles sem vértice em comum além das pontas.*
3. *Cada duas arestas estão em um ciclo.*

É conveniente decretar que $K_2 = \bullet\!\!-\!\!\!-\!\!\!-\!\!\!-\!\!\bullet$ é biconexo.

Blocos

Blocos

Um bloco de um grafo é um subgrafo biconexo maximal.

Blocos

Um bloco de um grafo é um subgrafo biconexo maximal.

Algumas propriedades:

- A interseção de dois blocos é vazia, ou consiste de um único vértice, que é de corte.

Blocos

Um bloco de um grafo é um subgrafo biconexo maximal.

Algumas propriedades:

- A interseção de dois blocos é vazia, ou consiste de um único vértice, que é de corte.
- Um bloco contem uma única aresta se e só se ela é uma ponte.

Blocos

Um bloco de um grafo é um subgrafo biconexo maximal.

Algumas propriedades:

- A interseção de dois blocos é vazia, ou consiste de um único vértice, que é de corte.
- Um bloco contem uma única aresta se e só se ela é uma ponte.
- Toda aresta está num único bloco.

Blocos

Um bloco de um grafo é um subgrafo biconexo maximal.

Algumas propriedades:

- A interseção de dois blocos é vazia, ou consiste de um único vértice, que é de corte.
- Um bloco contem uma única aresta se e só se ela é uma ponte.
- Toda aresta está num único bloco.

Blocos

Um bloco de um grafo é um subgrafo biconexo maximal.

Algumas propriedades:

- A interseção de dois blocos é vazia, ou consiste de um único vértice, que é de corte.
- Um bloco contem uma única aresta se e só se ela é uma ponte.
- Toda aresta está num único bloco. (tem uma relação de equivalência por trás disso)

Listagem de blocos

Problema

Dado um grafo conexo, encontrar seus blocos e suas articulações.

Blocos e DFS

Blocos e DFS

Suponha G conexo, que tenha sido feita uma DFS em G, com as anotações de tempos e da árvore.

Blocos e DFS

Suponha G conexo, que tenha sido feita uma DFS em G, com as anotações de tempos e da árvore.

A raiz da árvore é articulação sse

Blocos e DFS

Suponha G conexo, que tenha sido feita uma DFS em G, com as anotações de tempos e da árvore.

A raiz da árvore é articulação sse tem grau 1 na árvore.

Blocos e DFS

Suponha G conexo, que tenha sido feita uma DFS em G, com as anotações de tempos e da árvore.

A raiz da árvore é articulação sse tem grau 1 na árvore.

Como detectar outras articulações?

Blocos e DFS

Suponha G conexo, que tenha sido feita uma DFS em G, com as anotações de tempos e da árvore.

A raiz da árvore é articulação sse tem grau 1 na árvore.

Como detectar outras articulações? Idéias na lousa.

Blocos e DFS

Suponha G conexo, que tenha sido feita uma DFS em G, com as anotações de tempos e da árvore.

A raiz da árvore é articulação sse tem grau 1 na árvore.

Como detectar outras articulações? Idéias na lousa.

Defina

$$v.low = \min \begin{cases} v.d & , \\ w.d & u\text{—}w \text{ é aresta de retorno com } u \\ & \text{descendente próprio de } v \end{cases}$$

Blocos e DFS

Suponha G conexo, que tenha sido feita uma DFS em G, com as anotações de tempos e da árvore.

A raiz da árvore é articulação sse tem grau 1 na árvore.

Como detectar outras articulações? Idéias na lousa.

Defina

$$v.low = \min \begin{cases} v.d & , \\ w.d & u—w \text{ é aresta de retorno com } u \\ & \text{descendente próprio de } v \end{cases}$$

Como anotar a DFS para calcular isso?

Blocos e DFS

Suponha G conexo, que tenha sido feita uma DFS em G, com as anotações de tempos e da árvore.

A raiz da árvore é articulação sse tem grau 1 na árvore.

Como detectar outras articulações? Idéias na lousa.

Defina

$$v.low = \min \begin{cases} v.d & , \\ w.d & u\text{—}w \text{ é aresta de retorno com } u \\ & \text{descendente próprio de } v \end{cases}$$

Como anotar a DFS para calcular isso?

Algoritmo de Hopcroft

Blocos e DFS

Suponha G conexo, que tenha sido feita uma DFS em G, com as anotações de tempos e da árvore.

A raiz da árvore é articulação sse tem grau 1 na árvore.

Como detectar outras articulações? Idéias na lousa.

Defina

$$v.low = \min \begin{cases} v.d & , \\ w.d & u\text{—}w \text{ é aresta de retorno com } u \\ & \text{descendente próprio de } v \end{cases}$$

Como anotar a DFS para calcular isso?

Algoritmo de Hopcroft

CLRS Exercise 22-2.

Para ajudar

DFS-VISIT(G, u)

```
u.state = descoberto
u.d = time += 1
for each vertex v ∈ u.adj
    if v.state == inicial
        // achou v, u—v é da árvore
        v.sob = u
        DFS-VISIT(G, v)
    else
        // a aresta u—v é de retorno
v.state = finalizado
// faz alguma coisa?
u.f = time += 1
```