



Figura 7.11 Classificação das arestas na busca em profundidade em um grafo direcionado.

## 7.4 Teste para Verificar se Grafo é Acíclico

Existem várias situações em que o teste para verificar se um grafo é acíclico é importante. As duas seções a seguir apresentam dois algoritmos bem distintos para verificar se um grafo é acíclico.

### 7.4.1 Usando Busca em Profundidade

O Programa 7.9 para realizar busca em profundidade pode ser usado para verificar se um grafo  $G = (V, A)$  é acíclico ou contém um ou mais ciclos. Se uma **aresta de retorno** é encontrada durante a busca em profundidade em  $G$ , então o grafo tem **ciclo**. Igualmente, se um grafo tem um ciclo, então uma aresta de retorno será sempre encontrada em qualquer busca em profundidade em  $G$ .

Assim, um grafo direcionado  $G$  é acíclico se e somente se a busca em profundidade em  $G$  não apresentar arestas de retorno. Como vimos na seção anterior, o algoritmo BuscaEmProfundidade pode ser alterado para descobrir arestas de retorno. Para isso, basta verificar se um vértice  $v$  adjacente a um vértice  $u$  apresenta a cor cinza na primeira vez que a aresta  $(u, v)$  é percorrida. Isso deve ser feito no momento em que a lista de adjacentes de  $u$  estiver sendo percorrida. Logo, a BuscaEmProfundidade é um algoritmo de custo linear no número de vértices e de arestas de um grafo  $G = (V, A)$  que pode ser utilizado para verificar se  $G$  é acíclico.

### 7.4.2 Usando o Tipo Abstrato de Dados Hipergrafo

Esta seção apresenta um algoritmo para verificar se um **hipergrafo** ou  $r$ -grafo  $G_r(V, A)$  é acíclico. Hipergrafos são apresentados na Seção 7.10. A forma mais adequada para representar um hipergrafo é por meio de estruturas de dados orientadas a arestas em que para cada vértice  $v$  do grafo é mantida uma lista das arestas que incidem sobre o vértice  $v$ .

Existem duas representações usuais para hipergrafos: **matrizes de incidência** e **listas de incidência**. Aqui utilizaremos a implementação de listas de incidência usando arranjos apresentada na Seção 7.10.2.

O Programa 7.10 utiliza a seguinte propriedade de  $r$ -grafos:

Um  $r$ -grafo é **acíclico** se e somente se a remoção repetida de arestas contendo apenas vértices de grau 1 (isto é, vértices sobre os quais incide apenas uma aresta) elimina todas as arestas do grafo.

O procedimento GrafoAciclico recebe o grafo e retorna no vetor  $L$  as arestas retiradas do grafo na ordem em foram retiradas. O procedimento primeiro procura os vértices de grau 1 e os coloca em uma fila. A seguir, enquanto a fila não estiver vazia, desenfileira um vértice e retira a aresta incidente ao vértice. Se a aresta retirada tinha algum outro vértice de grau 2, então esse vértice muda para grau 1 e é enfileirado. Se ao final não restar nenhuma aresta, então o grafo é acíclico. O custo do procedimento GrafoAciclico é  $O(|V| + |A|)$ .

Programa 7.10 Teste para verificar se um hipergrafo é acíclico usando arranjos

```

program GrafoAciclico;
{— Entram aqui os tipos do Programa 3.17 (ou do Programa 3.19) —}
{— Entram aqui tipos do Programa 7.25 —}
var
  i, j      : integer;
  Aresta    : TipoAresta;
  Grafo     : TipoGrafo;
  L          : TipoArranjoArestas;
  GAciclico : boolean;
{— Entram aqui os operadores FFVazia, Vazia, Enfileira e —}
{— Desenfileira do Programa 3.18 (ou do Programa 3.20) —}
{— Entram aqui os operadores ArestasIguais, FGVazio, —}
{— InsereAresta, RetiraAresta e ImprimeGrafo do Programa 7.26 —}

function VerticeGrauUm (V: TipoValorVertice;
                        var Grafo: TipoGrafo): Boolean;
begin
  VerticeGrauUm := (Grafo.Prim[V] >= 0) and
    (Grafo.Prox[Grafo.Prim[V]] = INDEFINIDO);
end;

procedure GrafoAciclico (var Grafo : TipoGrafo;
                        var L : TipoArranjoArestas;
                        var GAciclico: boolean);
var
  j      : TipoValorVertice;
  A1     : TipoValorAresta;
  x      : TipoItem;
  Fila   : TipoFila;
  NArestas: TipoValorAresta;
  Aresta : TipoAresta;

```