Documentação JavaGraphPajek

Emanuel Costa Pedro Henrique Ribeiro
Henrique Resende Carlos Eduardo Meints
Thiago Guimarães

Novembro 2023

1 Introdução

A biblioteca "JavaGraphPajek" é uma ferramenta em Java para criar, ler e exportar grafos, utilizando o formato Pajek para representação de grafos simples/intermediários.

2 Como Começar

2.1 Requisitos

Certifique-se de que você tenha o Java instalado na sua máquina.

2.2 Instalação

Faça o download da biblioteca "JavaGraphPajek". Adicione a biblioteca ao seu projeto Java.

3 Funcionalidades Principais

3.1 Criar um Grafo

Para cirar um grafo direcionado, passe como parâmetro 'true', caso não queira direcionado, 'false'.

```
Grafo grafoDirecionado = new Grafo(true);
```

```
Grafo grafoNaoDirecionado = new Grafo(direcionado:false);
```

3.2 Criar Vértices

Para criar um vértice, basta passar como parâmetro um rótulo(String) e um peso(Double).

```
Vertice vertice1 = new Vertice(rotulo:"a", peso:10);
Vertice vertice2 = new Vertice(rotulo:"b", peso:20);
Vertice vertice3 = new Vertice(rotulo:"c", peso:30);
```

3.3 Adicionar Vértices

Para adicionar um vértice ao grafo já criado, basta usar a função 'adicionarVertice' passando como parâmetro o vértice que deseja adicionar.

```
grafoDirecionado.adicionarVertice(vertice1);
grafoDirecionado.adicionarVertice(vertice2);
grafoDirecionado.adicionarVertice(vertice3);
```

3.4 Criar Arestas

Para criar uma aresta, basta passar como parâmetro os dois vértices(Vertice), um peso(Double) e um rótulo(String).

```
Aresta aresta1 = new Aresta(vertice1, vertice2, peso:1, rotulo:"a");
Aresta aresta2 = new Aresta(vertice2, vertice3, peso:2, rotulo:"b");
Aresta aresta3 = new Aresta(vertice3, vertice1, peso:3, rotulo:"c");
```

3.5 Adicionar Arestas

Para adicionar uma aresta ao grafo já criado, basta usar a função 'adicionarAresta' passando como parâmetro a aresta(Aresta) que deseja adicionar.

```
grafoDirecionado.adicionarAresta(aresta1);
grafoDirecionado.adicionarAresta(aresta2);
grafoDirecionado.adicionarAresta(aresta3);
```

3.6 Remover aresta

Para remover uma aresta basta utilizar a função 'removerAresta' no grafo desejado passando como parâmetro a aresta(Aresta) que deseja remover, funcionando tanto em um grafo direcionado como um não direcionado.

```
grafoDirecionado.removerAresta(aresta3);
grafoNaoDirecionado.removerAresta(arestaNaoDirecionada3);
```

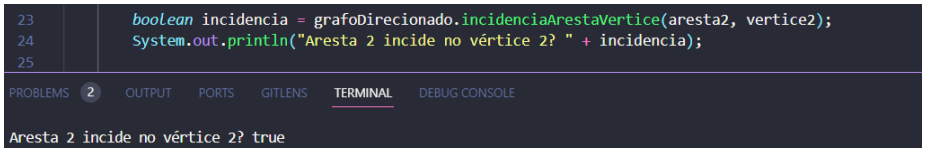
3.7 Verificar existência de aresta entre dois vértices

Para verificar se uma aresta existe basta utilizar a função 'arestaExiste' no grafo desejado passando como parâmetro os dois vértices(Vertice) que deseja verificar, funcionando tanto em um grafo direcionado como um não direcionado.

```
grafoDirecionado.arestaExiste(vertice1, vertice2);
grafoNaoDirecionado.arestaExiste(vertice1, vertice2);
```

3.8 Conferindo incidências em um grafo direcionado

Para conferir a incidência entre uma aresta e um vértice, basta utilizar a função 'incidenciaArestaVertice', passando como parâmetro a aresta(Aresta) e o vértice(Vertice) que deseja conferir.



3.9 Conferindo se grafo é completo

Para conferir se o grafo é completo, basta utilizar a função 'verificaGrafoCompleto' no grafo desejado, funcionando tanto em um grafo direcionado como um não direcionado, retornando 'true' se for completo e 'false' se não.

```
grafoDirecionado.verificaGrafoCompleto()
```

```
grafoNaoDirecionado.verificaGrafoCompleto()
```

3.10 Conferindo se grafo é vazio

Para conferir se o grafo é vazio, basta utilizar a função 'verificaGrafoVazio' no grafo desejado, funcionando tanto em um grafo direcionado como um não direcionado, retornando 'true' se for vazio e 'false' se não.

```
grafoDirecionado.verificaGrafoVazio()
```

```
grafoNaoDirecionado.verificaGrafoVazio()
```

3.11 Matriz de adjacência

Para obter a matriz de adjacência, basta utilizar a função 'getMatrizAdjacencia' no grafo desejado, funcionando tanto em um grafo direcionado como um não direcionado, retorna uma matriz de inteiros.

```
grafoDirecionado.getMatrizAdjacencia();
```

```
grafoNaoDirecionado.getMatrizAdjacencia();
```

3.12 Ler um grafo a partir de um arquivo pajek

Para ler um grafo de um arquivo pajek, basta utilizar a classe 'GrafoPAJEK', criar uma instância dela e utilizar a função 'lerArquivoPAJEK' nessa instância, passando como parâmetro o caminho do arquivo(String) e se o grafo é direcionado ou não(Boolean). A função retorna o grafo gerado a partir do arquivo.

```
String caminhoArquivo = "grafoTeste.net";
Grafo grafo = GrafoPAJEK.lerArquivoPAJEK(caminhoArquivo, direcionado:true);
```

3.13 Gerar um arquivo pajek a partir de um grafo

Para gerar um arquivo pajek a partir de um grafo, basta utilizar a classe 'GrafoPAJEK', criar uma instância dela e utilizar a função 'gerarArquivoPAJEK' nessa instância, passando como parâmetro o caminho do arquivo(String) e o grafo que deseja exportar(Grafo).

```
GrafoPAJEK.gerarArquivoPAJEK(nomeArquivo, grafoTeste);
```

Referências

Link para a especificação do formato Pajek: https://pt.slideshare.net/AndrMarques18/tutorial-pajek