Disciplina: SCC0530 - Inteligência Artificial
Trabalho: Algoritmos de busca

---

1. Sobre o trabalho

Este trabalho tem como objetivo a implementação dos algoritmos de busca vistos em aula e aplicá-los em um problema real de busca.

○ Implementação

Os seguintes algoritmos de busca serão utilizados nos experimentos:

- Busca em profundidade
- Busca em largura
- Algoritmo Best-First
- Algoritmo A*

As seguintes redes serão utilizadas para os experimentos, onde cada grupo deve selecionar apenas um tipo de rede para o trabalho (a descrição de como gerar as redes encontra-se na Apêndice).

- Rede KNN
- Rede Aleatória
- Rede Pequeno Mundo
- Rede Geográfica

Os algoritmos de busca e a geração das redes do tipo selecionado devem ser implementados na linguagem de programação que o grupo escolher.

○ Experimentos

Os seguintes experimentos são propostos para este projeto:

i. Dado o tipo de rede selecionado pelo grupo, para cada rede gerada, selecionar um vértice origem e um vértice destino e aplicar todos algoritmos de busca. Para cada algoritmo, mostrar o caminho final percorrido (cada vértice que faz parte do caminho solução) e a distância final percorrida. No caso de visualização, do experimento, gere uma rede com poucos vértices (e.g. 100 vértices).

ii. Para o tipo de rede selecionado pelo grupo gere as redes com as seguintes configurações:

➢ **Rede KNN**

a. (n=2000, k=3)
b. (n=2000, k=7)

c. (n=2000, k=11)

➢ **Rede Aleatória**

a. (n=2000, p=5%)
b. (n=2000, p=2,5%)
c. (n=2000, p=1%)

➢ **Rede Pequeno Mundo**

a. (n=2000, k=7, p=10%)
b. (n=2000, k=7, p=5%)
c. (n=2000, k=7, p=1%)

➢ **Rede Geográfica**

a. (n=2000, λ=0,01)
b. (n=2000, λ=0,02)
c. (n=2000, λ=0,03)

Assim, selecionar, no mínimo, 10 pares distintos de vértices (origem, destino) e aplicar os algoritmos de busca com objetivo de reportar a distância média percorrida, além do tempo médio gasto por cada algoritmo.

iii. Realizar uma comparação, utilizando as redes do experimento ii, entre o algoritmo A* e o algoritmo de Dijkstra.

Ao fim, um relatório, de no máximo 10 páginas, deve ser entregue. Neste relatório deve conter: introdução, descrição da implementação dos algoritmos de busca e da rede selecionada, resultados dos experimentos propostos e uma discussão sobre os resultados obtidos. O trabalho deve ser apresentado em sala de aula, por meio de slides, e deve conter os resultados obtidos nos experimentos e conclusões.

2. Informações importantes

- O grupo deve ter **4 integrantes**. São permitidos apenas 4 grupos por tipo de rede. O preenchimento dos grupos e do tipo de rede selecionado será feito no calendário de apresentações, pelo seguinte link:

  https://docs.google.com/spreadsheets/d/1EQ1-1GU8yRp7OGGp46VFG0IWFJkbZaLAenJmVMEuQaU/edit?usp=sharing

- O projeto pode ser desenvolvido em qualquer linguagem de programação;

- Relatório (em .pdf) e código-fonte do projeto devem ser entregues no Tidia (escaninho) por apenas um integrante do grupo;
- O **tempo de apresentação** é de **12 minutos + 3 de arguição**;
- **Trabalhos plagiados receberão nota 0.**

3. Datas

- **Entrega do relatório e código-fonte** até o final do dia **23 de Maio**;
- **As apresentações serão nos dias 23, 26 e 30 de Maio**. O dia e a ordem das apresentações devem ser preenchidos na seguintes planilha, assim como a rede selecionada:

4. Apêndice

Nesta seção é descrito como implementar o gerador das redes apresentadas na Seção 1.

➢ Rede KNN

O pseudo-código abaixo elucida a geração de uma rede KNN, sendo *n* o número de vértices na rede e *k* o número de vizinhos (arestas) conectadas a cada vértice, temos que:

```
função gera_rede_knn(n, k):
       lista_vértices <- gera_vértices(n)
       lista_arestas <- gera_arestas(lista_vertices, k)
retorna lista_vertices, lista_arestas
```

Onde:

- A função ***gera_vertices*** retorna uma lista de *n* vértices, onde para cada vértice *v* é associada as coordenadas x e y. As coordenadas x e y são dadas aleatoriamente, e devem respeitar os limites do eixo-x e eixo-y que devem ser limitados ao valor de n. Por exemplo, tendo *n = 1000*, as coordenadas de x e y devem estar entre os valores de 0 e 1000. Portanto, quanto maior o valor de *n*, maior o tamanho do plano xy gerado, o que aumenta as possibilidades de distribuição dos vértices de acordo com o tamanho do grafo.

- A função ***gera_arestas*** recebe a *lista_vértices* e o valor de *k*. Para cada vértice são conectadas arestas nos *k* vértices mais próximos do vértice em questão. O valor (peso) da aresta é dado pela distância geométrica entre o par de vértices da aresta.

Assim, ao final da função ***gera_rede_knn*** é gerado um grafo não-direcionado com *n* vértices e grau médio igual a *k*.

➢ Rede Aleatória

O pseudo-código abaixo elucida a geração de uma rede aleatória, sendo *n* o número de vértices na rede e *p* a probabilidade de existir uma aresta entre um par de vértices.

```
função gera_rede_aleatoria(n, p):
       lista_vértices <- gera_vértices(n)
       lista_arestas <- gera_arestas(lista_vertices, p)
retorna lista_vertices, lista_arestas
```

Onde:

- A função ***gera_vertices*** retorna uma lista de *n* vértices, onde para cada vértice *v* é associada as coordenadas x e y. As coordenadas x e y são dadas aleatoriamente, e devem respeitar os limites do eixo-x e eixo-y que devem ser limitados ao valor de n. Por exemplo, tendo *n = 1000*, as coordenadas de x e y devem estar entre os valores de 0 e 1000. Portanto, quanto maior o valor de *n*, maior o tamanho do plano xy gerado, o que aumenta as possibilidades de distribuição dos vértices de acordo com o tamanho do grafo.

- A função ***gera_arestas*** recebe a *lista_vértices* e a probabilidade *p*. Para cada par de vértices gerar uma valor aleatório (entre 0 e 1), se esse valor for menor do que *p* é gerada uma aresta entre o par de vértice em questão, caso contrário não gera uma nova aresta. O valor (peso) da aresta é dado pela distância geométrica entre o par de vértices da aresta.

Assim, ao final da função ***gera_rede_aleatoria*** é gerado um grafo não-direcionado com *n* vértices e grau médio esperado igual a *(n - 1)*p*.

➢ Rede Pequeno Mundo

O pseudo-código abaixo elucida a geração de uma rede pequeno mundo, sendo *n* o número de vértices na rede, *k* número de vizinhos (arestas) conectadas a cada vértice, e *p* a probabilidade de reconectar uma das arestas de um vértice existente.

```
função gera_rede_pequeno_mundo(n, k, p):
    lista_vértices, lista_arestas <- gera_grafo_knn(n,k)
    lista_arestas <- reconecta_arestas(lista_arestas, p)
retorna lista_vertices, lista_arestas
```

Onde:

- A função ***reconecta_arestas*** recebe a lista de vértices gerada pela função gera grafo-knn e a probabilidade *p* de reconectar uma aresta. Para cada aresta gerar um valor aleatório (entre 0 e 1), se o valor for menor do que *p* um dos vértices da aresta será trocado por um vértice qualquer (sorteado aleatoriamente), caso contrário a aresta não é modificada. O valor (peso) da aresta é dado pela distância geométrica entre o par de vértices da aresta.

Assim, ao final da função ***gera_rede_pequeno_mundo*** é gerado um grafo não-direcionado com *n* vértices e grau médio igual a *k*.

➢ Rede Geográfica

O pseudo-código abaixo elucida a geração de uma rede geográfica, sendo *n* o número de vértices na rede e um valor λ.

```
função gera_rede_geográfica(n,λ):
       lista_vértices <- gera_vértices(n)
       lista_arestas <- gera_arestas(lista_vertices, λ)
retorna lista_vertices, lista_arestas
```

Onde:

- A função ***gera_vertices*** retorna uma lista de *n* vértices, onde para cada vértice *v* é associada as coordenadas x e y. As coordenadas x e y são dadas aleatoriamente, e devem respeitar os limites do eixo-x e eixo-y que devem ser limitados ao valor de n. Por exemplo, tendo *n = 1000*, as coordenadas de x e y devem estar entre os valores de 0 e 1000. Portanto, quanto maior o valor de *n*, maior o tamanho do plano xy gerado, o que aumenta as possibilidades de distribuição dos vértices de acordo com o tamanho do grafo.

- A função ***gera_arestas*** recebe a *lista_vértices* e o valor λ. Para cada par de vértices gerar uma valor aleatório (entre 0 e 1), se esse valor for menor do que *p* é gerada uma aresta entre o par de vértice em questão. O valor de *p* varia de acordo com cada par de vértices e é dado por $P(i \ -> \ j) \ = \ e^{-\lambda s_{ij}}$, sendo S_{ij} a distância geométrica entre os vértices i e j. O valor (peso) da aresta é dado pela distância geométrica entre o par de vértices da aresta.