# MAC 328 — Algoritmos em Grafos

Tarefa 3 — 7/6/2012 — entrega 27/6/2012

v. 1.1 (12/6/2012)

ATENÇÃO: **Este trabalho pode ser feito em duplas**

## Descobrindo se um grafo é denso

Desde o começo do curso trabalhamos com duas representações de grafos, por listas e por matriz. Da análise assintótica fica claro que listas é melhor quando o grafo é esparso, e matriz pode ser melhor quando o grafo é denso. Só que não é muito claro como diferenciar um caso do outro. Além disso, a análise foi bem superficial, será que não faltou um pouco mais de detalhe?

Aqui vai ser o caso de comparar experimentalmente essas duas representações, com grafos de densidades diferentes.

## 1 O experimento

Para isso, cada experimento consiste em:

1. Gerar um grafo aleatório $G$, de acordo com o número `V` de vértices e uma probabilidade dada $p$ para cada arco. O grafo deve ser representado das duas formas, matriz e listas.
2. Para cada representação de $G$:
   (a) Faça 8 buscas em profundidade em $G$, registrando o tempo. (Obs.: talvez seja melhor fazer $8 * \texttt{V}$, para ficar comparável com (b))
   (b) Para cada vértice $v$, faça 8 buscas em largura a partir de $v$, registrando o tempo total.

Esse número mágico 8 não tem nenhuma base teórica ou empírica. É um simples chute para amortizar efeitos pontuais da carga do computador.

O *resultado* do experimento consiste de cinco números, os resultados de (2a) e (2b) para as duas representações e o número de arcos do grafo. (Um trabalho mais científico guardaria também informação suficiente para reproduzir o experimento, mas não vamos exagerar). Use `clock` ou `clock_gettime` para calcular os tempos.

Para uniformizar um pouco os resultados, vamos fixar o seguinte:

1. Utilize o código das transparências: `DIGRAPHdfs` (13/03) e `DIGRAPHbfs` (10/04). Não tenho o código em formato utilizável, alguém vai ter que digitar. Por outro lado, eu gostaria que todos usassem o mesmo código nesta parte, assim, encontrem uma forma de dividir o trabalho (entre todos) e compartilhar esse código (por exemplo, colocando um arquivo no fórum).

2. Altere o tipo `Digraph`, para conter a matriz de adjacência `Adj` e o vetor de listas de adjacência `adj`.

3. Para cada função que usar que se baseia na matriz de adjacência, edite o nome da função colocando um `M` no fim e substituindo cada `adj` por `Adj`. Para cada função que usa listas, altere seu nome colocando um `L` no fim.

4. Para gerar digrafos aleatórios, crie uma função que recebe como parâmetros o número de vértices `V` e uma probabilidade `p`, e devolve o digrafo. O algoritmo de geração começa com um grafo sem arcos e:

   > Para cada par ordenado `(i,j)` de vértices distintos,
   > se `rand() < p*RAND_MAX` insira o arco `(i,j)`.

5. Crie uma função `Experimento` que, dados `V` e `p`, realiza um experimento com esses parâmetros, devolvendo o resultado.

Até aí tudo deve ser programado em C.

## 2 O teste

Seu programa deve permitir rodar vários experimentos, registrando os resultados para processamento estatístico posterior. Para isso, é preciso construir um driver que produza vários `V` e `p` e chame `Experimento`.

*Não existe nenhuma restrição quanto à construção do driver,* exceto que ele deve poder funcionar num sistema Linux Debian ou Ubuntu, sem grandes instalações adicionais. Em particular, o driver pode ser em qualquer linguagem. Algumas arquiteturas possíveis (aceito outras):

- Um programa em C, C++ ou alguma outra linguagem que consiga ligar com as funções acima.

- Embutir essas funções em um programa que recebe `V` e `p` na linha de comando e fazer o driver chamar esse programa.
- Embutir essas funções em um servidor que recebe solicitações `V` e `p` por algum mecanismo de IPC, e devolve o resultado do experimento.

Execute experimentos com duas famílias de `p`: $\alpha$ e $\frac{\alpha}{\lg \mathtt{V}}$, onde $\alpha$ é uma constante no intervalo real $(0, 1)$. Para cada valor de `V` e `p`, execute $n$ experimentos para estimar o resultado médio (média componente a componente). O valor de $n$, para começar, é 30, mas posso variar depois de uns cálculos estatísticos e sugestões de vocês.

Tente, usando busca binária, encontrar em cada caso, que valor de $\alpha$ indica a transição entre "denso" e "esparso", no sentido que os algoritmos em uma representação sejam melhores que na outra.

Rode experimentos para vários valores de `V` começando em 100 e incluindo 1000 e 10000. Se você for usar algum software de análise estatística ou uma planilha para analisar os resultados, pode ser conveniente emitir os resultados na forma de um arquivo csv.

# 3 O que deve ser entregue

Além dos programa-fonte, um README instruindo como compilar e como usar seu programa. **E** um relatório (pdf) descrevendo os experimentos realizados, os resultados obtidos e tirando ou não conclusões sobre a eficácia relativa das duas representações.

Identifique claramente os autores em todos os arquivos que forem de sua autoria.

Entregue tudo num arquivo `tgz` da seguinte forma: crie um diretório cujo nome seja o nUSP de um dos autores e coloque todo o material lá dentro; arquive com

```
tar czf nusp.tgz nusp
```

onde, é claro, `nusp` deve ser substituído pelo respectivo número.

No caso de dois autores, um entrega o tgz, o outro, para deixar o paca feliz, entrega um arquivo chamado `veja-nusp` (`nusp` acima) que não precisa conter nada.

# 4 Aditivos

Este enunciado quase certamente terá aditivos, ou para simplificar os requerimentos, ou para trocar as constantes chutadas 8 e 30 por outras mais justificadas, ou para deixar mais claro uma parte ou outra.