MAC323 EXERCÍCIO-PROGRAMA 1

LIMIAR DE CONEXIDADE PARA CERTOS GRAFOS GEOMÉTRICOS

Y. KOHAYAKAWA

Data de entrega: 8/4/2013 (23:55)

1. Introdução

Inicialmente, lembre-se do problema geométrico discutido em sala no dia 28/2/2013 (veja `prog3.18.c`). Estamos aqui interessados em encontrar o 'limiar de conexidade' para uma coleção 'grande' e 'típica' de pontos no plano. (Este EP poderia ter como motivação o estudo de redes de sensores (*wireless sensor networks*, WSNs).)

2. Descrição do problema

Sejam dados N e $0 \le d \le 1$. Sejam ainda $p_1, \dots, p_N \in [0,1]^2$ pontos no quadrado unitário. Definimos um grafo $G = (V, E)$ com $V = [N] = \{1, \dots, N\}$ e

$$E = \big\{\{i,j\} \colon \|p_i - p_j\| \le d\big\}. \tag{1}$$

Para explicitar a dependência de G da coleção de pontos p_i ($1 \le i \le N$) e de d, podemos escrever $G(\mathbf{p}, d)$ para G acima, onde $\mathbf{p} = (p_1, \dots, p_N)$. O *limiar de conexidade* $d_*(\mathbf{p})$ de $\mathbf{p}$ é o menor d tal que $G(\mathbf{p}, d)$ é conexo. (Para conceitos básicos da teoria dos grafos, veja, por exemplo, o material em `http://www.ime.usp.br/~pf/grafos-exercicios/` Basta você ler a definição de grafo e então ler a definição de conexidade de grafos.)

Seu EP deve implementar um algoritmo que, dado $\mathbf{p}$ como acima, determina $d_*(\mathbf{p})$. Este algoritmo deve ser 'eficiente' para instâncias aleatórias de $\mathbf{p}$ (como definido mais precisamente a seguir) com N 'grande'. Seu programa deverá estimar o que podemos chamar de 'densidade normalizada crítica para conexidade' para N pontos uniformemente escolhidos em $[0,1]^2$.

2.1. **Instâncias aleatórias.** Dado N, para gerar uma instância aleatória $\mathbf{p} = (p_1, \dots, p_N)$, você deve gerar os p_i ($1 \le i \le N$) uniformente ao acaso em $[0,1]^2$, independentemente.

2.2. **Densidade normalizada crítica.** Definimos (informalmente) a *densidade normalizada crítica para conexidade* para N pontos aleatórios em $[0,1]^2$ como sendo o valor 'típico' $c_*(N)$ de

$$(d_*(\mathbf{p}))^2 \frac{N}{\log N}, \tag{2}$$

onde $\mathbf{p}$ é como definido em §2.1. Prova-se que, quando $N \to \infty$, esses valores típicos convergem para um valor c_* ($0 < c_* < \infty$).

3. Seu programa

Seu programa deve ter dois modos de operação.

Versão de 19 de março de 2013, 09:09.

3.1. **Verificação de conexidade.** No modo de *verificação de conexidade*, a tarefa principal do programa é verificar se um dado $\mathbf{p} = (p_1, \ldots, p_N)$ e d definem um grafo $G(\mathbf{p}, d)$ conexo. Neste modo, seu programa deve receber como entrada N e d e ele deve gerar N pontos aleatórios como em §2.1. A saída do programa deve ser uma mensagem, dizendo se $G(\mathbf{p}, d)$ é conexo ou não. Seu programa deve também receber, opcionalmente, um inteiro não-negativo s (uma *semente*) para inicializar o gerador de números aleatórios (você deve usar `rand()` e `srand()` do `stdlib` em seu programa). Caso o usuário não dê uma semente, seu programa deve usar um valor padrão fixo de semente (um valor *default*). Finalmente, o usuário também deverá poder executar seu programa de modo que ele tenha também como saída uma listagem dos pontos p_i gerados (isto será útil na depuração de seu programa).

Seu programa deve receber a entrada na linha de comando. Assim, seu programa poderia ser executado, por exemplo, das seguintes formas:

```
prompt$ ep1 -N100 -d0.1
prompt$ ep1 -N1000 -d0.01 -s323
prompt$ ep1 -N10000 -d0.001 -s323 -v
```

Você deve também implementar a seguinte forma de execução útil para depuração: com a chamada

```
prompt$ ep1 -C -d0.3
```

seu programa deve ler as coordenadas x e y de pontos em $[0, 1]^2$ do `stdin`, e deve determinar se o d dado na linha de comando define um grafo conexo. Sua entrada para a chamada acima poderia ser, por exemplo,

```
0.1 0.6
0.7 0.1
0.6 0.5
0.3 0.4
```

Esta entrada tem 4 pontos, a saber, $(0.1, 0.6)$, $(0.7, 0.1)$, $(0.6, 0.5)$ e $(0.3, 0.4)$.

3.2. **Estimação da densidade normalizada crítica.** Seu programa também deve ser capaz de estimar a densidade normalizada crítica $c_*(N)$ (veja §2.2). Para tanto, o usuário fornecerá N e também um inteiro M. Seu programa deve gerar M instâncias aleatórias $\mathbf{p}_1, \ldots, \mathbf{p}_M$ e deve determinar $d_*(\mathbf{p}_j)$ para todo $1 \leq j \leq M$. Seu programa deve então devolver como sua estimativa para $c_*(N)$ a média dos M números da forma (2) (um para cada $\mathbf{p}_j$).

Novamente, o usário deve ter a opção de fornecer uma semente para o gerador de números aleatórios, através da opção de linha de comando `-s`. Ademais, se o usuário der a opção `-v`, seu programa deve imprimir os M valores da expressão (2) obtidos. Ademais, implemente a opção `-V`, que faz com que seu programa imprima não só estes M valores, mas lista os $\mathbf{p}_j$ correspondentes.

Seu programa poderia ser executado, por exemplo, das seguintes formas:

```
prompt$ ep1 -N100 -M1
prompt$ ep1 -N1000 -M100 -s323
prompt$ ep1 -N10000 -M100 -s323 -v
prompt$ ep1 -N20 -M5 -s323 -V
```

Como em §3.1, você deve também implementar a seguinte forma de execução útil para depuração: com a chamada

```
prompt$ ep1 -D
```

seu programa deve ler uma seqüência de pontos em $[0,1]^2$ do `stdin` e deve determinar $d_*(\mathbf{p})$, onde $\mathbf{p}$ é a seqüência de pontos dada pelo usuário; no exemplo em §3.1, temos

$$\mathbf{p} = ((0.1, 0.6), (0.7, 0.1), (0.6, 0.5), (0.3, 0.4)).$$

Observações

1. *Este EP é estritamente individual.* Programas semelhantes receberão nota 0.
2. Seja cuidadoso com sua programação (correção, documentação, apresentação, clareza do código, etc), dando especial atenção a suas estruturas de dados. A correção será feita levando isso em conta.
3. Comparem entre vocês o desempenho de seus programas.
4. Entregue seu EP no Paca.
5. Não deixe de incluir em seu código um *relatório* para discutir seu EP: discuta as estruturas de dados usadas, os algoritmos usados, etc. *Se você escrever claramente como funciona seu EP, o monitor terá pouca dificuldade em corrigi-lo, e assim você terá uma nota mais alta.* (Se o monitor sofrer para entender seu código, sua nota será baixa.)

Observação final. Enviem dúvidas para a lista de discussão da disciplina.

Instituto de Matemática e Estatística, Universidade de São Paulo, Rua do Matão 1010, 05508–090 São Paulo, SP

Endereço eletrônico: `yoshi@ime.usp.br`