Exercício Programa 1

Fábio Nakano, *Professor,EACH-USP*

***Abstract*—Instrui como implementar uma árvore de decisão usando uma lista generalizada. Entrega da implementação até 09.10.2016.**

***Index Terms*—listas generalizadas, árvores de decisão, exercício programa, IEEEtran paper template, LaTeX.**

I. Introdução

UMA rede de venda de telefones celulares quer construir um app que, entre outras utilidades como lista de tarefas, agenda, ... sugira ao usuário que está na hora de trocar de telefone e oferece desconto para isso. Sua tarefa é construir a parte do código que computa a decisão a tomar usando árvores de decisão implentadas com listas generalizadas.

Listas generalizadas são listas em que um nó pode ser um elemento ou uma sub-lista. É uma estrutura de dados muito versátil e com muitas variações. A escolha da regra de encadeamento dos nós/listas depende da aplicação específica. Com esta estrutura é possível representar tabelas diversas, como as usadas em *hashing* e tabelas de símbolos em compiladores e interpretadores, matrizes densas ou esparsas, grafos quaisquer (árvores são um subconjunto do conjunto de todos os grafos),...

Árvores, na definição em teoria de grafos, são grafos acíclicos e conexos. Uma árvore com raíz é uma árvore em que um nó foi escolhido como raíz. O(s) pai(s) de um nó são os nós diretamente conectados a este e mais próximos da raiz. O(s) filho(s) de um nó são os nós diretamente conectados a ele e mais distantes da raiz. Nós sem filhos podem ser chamados folhas ou nós terminais. Sub-árvore é um subgrafo de uma árvore que tem as características de árvore. Nota: grafo é uma trinca $\{V, A, f\}$ onde V é o conjunto de vértices ou nós do grafo, A é o conjunto de arestas ou arcos do grafo e f é a função de incidência, que associa vértices a arestas.

Árvores de classificação/decisão são estruturas usadas para classificar ou decidir baseada na avaliação sequencial de atributos. Os dois grandes processos sobre árvores de decisão são **treinamento** e **uso**. **Nota**: o objetivo da técnica que emprega esta estrutura é classificar ou, se preferir, discriminar.

A. Treinamento

No treinamento são definidos topologia (formato) da árvore e os parâmetros dos nós **a partir dos dados**. Estes provêm de bancos de dados, ou são dados mineirados, ou são resultado de pesquisa de opinião. Para cada unidade amostral são coletados atributos: cada unidade tem um vetor de atributos e o conjunto de todos os vetores de atributos usados para o treinamento constituem o **conjunto de treinamento**. Há várias formas de fazer o treinamento e podem enviesar a árvore resultante.

Publicado em 05 de Setembro de 2016.

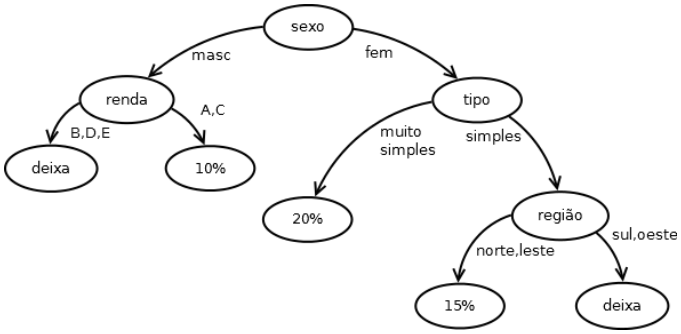

Fig. 1. O primeiro atributo testado é sexo, se masculino, o segundo atributo é a renda, se for A ou C a decisão é oferecer com 10% de desconto, senão a decisão é "deixar quieto". Se feminino o segundo atributo é o tipo de celular que possuía, se for muito simples a decisão é oferecer com 20% de desconto, se simples, testa em que região mora, se for leste ou norte então a decisão é oferecer com 15% de desconto. **Notas**: é comum que nem todos os atributos sejam utilizados pois alguns são mais discriminantes que outros; algumas decisões "deixa quieto" foram omitidas e categorias foram agrupadas para reduzir o tamanho da figura.

Neste específico problema os atributos sobre cada pessoa que de fato comprou telefone na loja são: {ID, ano de nascimento, sexo, relacionamento, renda, região onde mora, tipo de celular que possuía, tipo de transporte que utiliza, quantas vezes por semana vai a restaurante}. O conjunto de atributos dos compradores foi enriquecido com o resultado de uma pesquisa de intenção de compra, compondo o conjunto de treinamento.

O treinamento (não é foco deste EP explicar detalhadamente como foi feito) resultou na árvore de decisão da Fig. 1.

B. Uso

Com a árvore e um vetor de atributos coletado de alguma forma, a decisão é computada percorrendo a árvore testando atributos até chegar a uma decisão. Por exemplo, se os atributos coletados são: {ID=1234, sexo=feminino, relacionamento=namorando, renda=C, região=Leste, tipo de celular=simples, tipo de transporte=predominantemente coletivo, restaurante=2}, a árvore é percorrida a partir da raiz, que testa o sexo, como é feminino vai para o filho da direita, que testa tipo do celular, como é simples vai para o filho da direita, que testa a região onde mora, como é Leste, então a decisão é oferecer com 15% de desconto.

C. O app

O app, com as devidas permissões, coleta ou estima a informação do dono do celular para preencher o vetor de atributos, computa a decisão percorrendo a árvore previamente programada e executa a decisão.

TABLE I
ATRIBUTOS E SEUS ÍNDICES, CATEGORIAS E SEUS ÍNDICES.

I	Atributo	Categorias
0	ID	não usado como atributo
1	sexo	0-fem, 1-masc
2	relacionamento	0-solteiro 1-namorando/comprometido 2-casado 3-separado
3	renda	0-A 1-B 2-C 3-D 4-E
4	região onde mora	0-Norte 1-Sul 2-Leste 3-Oeste
5	tipo de celular que possuía	0-muito simples 1- simples 2-midrange 3-top
6	tipo de transporte que utiliza	0-somente coletivo 1-predominantemente coletivo, 2-predominantemente individual 4-somente individual
7	quantas vezes por semana vai a restaurante	0,1,2,3,4,5,6,7

TABLE II
DECISÕES E SEUS ÍNDICES

I	Decisão
0	"deixa quieto"
1	oferece com desconto de 10%
2	oferece com desconto de 15%
3	oferece com desconto de 20%

II. CÓDIGO

```
typedef struct node {
int categoria;
int atributoOuDecisao;
struct node *prox;
struct node *lista;
} No;
```

O nó lembra o de uma lista duplamente encadeada, mas a semântica é diferente. Quando o nó é raiz da árvore, categoria não é usado. decisaoOuCaracteristica contém o índice da decisão ou da (próxima) característica a testar: caso lista==NULL atributoOuDecisao é uma decisão e o nó não tem filhos caso contrário é um atributo e a lista contém nós, cada um corresponde a uma categoria do atributo testado no pai, armazenada em categoria e indica a ação seguinte, seja decisão ou o teste de outra característica. Caso a categoria buscada não seja a contida no nó, vai para o próximo nó e assim por diante até encontrar a categoria ou não haver mais próximo.

O tipo de dados de categoria, atributoOuDecisao é inteiro - um índice. O mapeamento entre a descrição é o inteiro relacionado deve ser feito por tabela auxiliar que não precisa ser implementada para este EP. A relação entre descrição e índice é dada na tabela I. A relação entre decisão e índice é dada na tabela II.

A árvore de decisão do exemplo em termos da estrutura de dados é apresentada na Fig. 2.

O vetor de atributos é um *array* de inteiros, onde o índice do elemento é dado pela coluna I da tabela I e os valores possíveis para o elemento são listados na mesma tabela, na lista de Categorias. O exemplo usado na seção Tomada de Decisão corresponde ao *array* {1234, 0, 1, 2, 2, 1, 1, 2}.

III. TAREFA

Codifique em C os métodos com os seguintes funcionamentos:

```
No *criaArvore();
```

Cria uma árvore com um único nó que corresponde a uma decisão, inicializa os campos com zero ou NULL, conforme o tipo do campo. Retorna o ponteiro para esse nó.

```
No *criaFilho (No *pai, int atributoDoPai, ...
```

Cria um nó filho do nó pai, ajusta o atributo do pai, categoria e decisao do filho. Retorna o ponteiro para o filho.

```
No *buscaFilho (No *n, int atributo, ...
```

Busca, entre os filhos de n aquele com atributo e categoria dados como parâmetro. Retorna os ponteiros para o filho e seu antecessor (se houver, como **conteúdo** do parâmetro). Caso não haja antecessor atribui NULL ao **conteúdo** do parâmetro. Caso não encontre filho, não há antecessor e retorna NULL.

```
int decide (No *arvore, int *atributos);
```

Recebe uma árvore de decisão e um vetor de atributos e retorna a decisão computada pela árvore.

Os protótipos completos são dados no arquivo de header *decisor.h*. O arquivo que contém as implementações chama-se *decisor.c* e o arquivo que contém o(s) teste(s) chama-se *teste.c*.

A. Compilação

Compila-se os arquivos-fonte e cria-se o executável com a linha de comando:

gcc decisor.c teste.c -o decisor

IV. CONDIÇÕES DE ENTREGA

- O EP deve ser feito individualmente.
- Entrega através do Tidia até 23h do dia 09.10.2016. Já está considerada tolerância por dessincronia dos relógios envolvidos.
- Deve ser entregue um arquivo COMPACTADO. O nome do arquivo deve ser <seunusp>.zip contendo apenas o arquivo *decisor.c*, com cabeçalho similar ao de *teste.c*, sem subdiretórios. Modificações em outros arquivos não terão efeito.
- Erros no formato de entrega acarretarão descontos.
- Plágios resultarão em zero.

REFERENCES

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

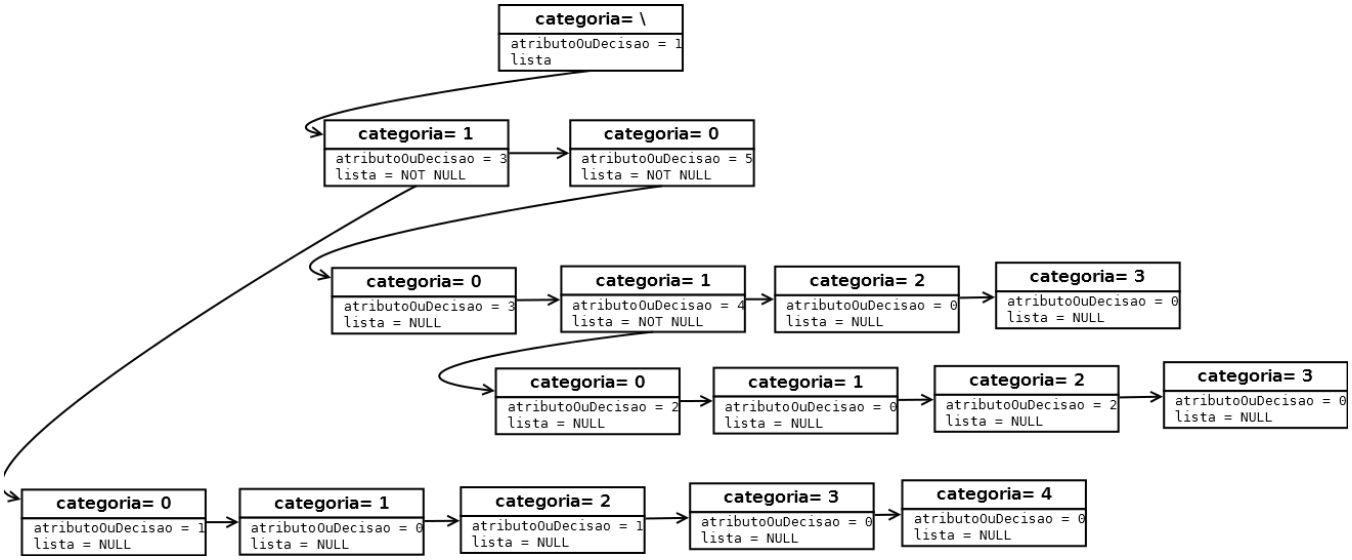

Fig. 2. Estrutura de dados da árvore de decisão do exemplo. O campo prox estrutura a lista ligada e corresponde às conexões retas.

Fábio Nakano Docente turma 03 da disciplina ACH2023 - Algoritmos e Estruturas de Dados I

Este documento (também) é um teste com o formato de revistas do IEEE.