UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
FACOM - FACULDADE DE COMPUTAÇÃO
ALGORITMOS E PROGRAMAÇÃO II

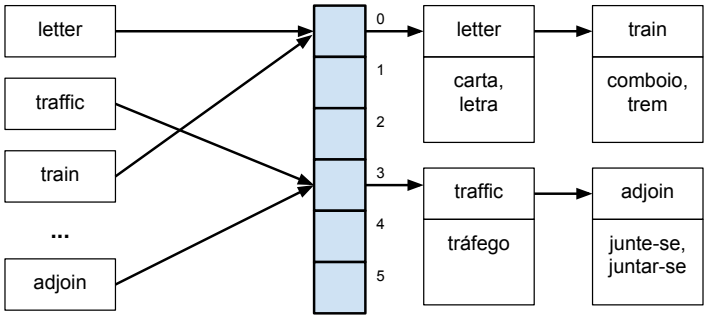
Terceiro Trabalho

1 O Problema

Escreva um programa para implementar um dicionário Inglês-Português utilizando uma Tabela de Espalhamento (ou Dispersão) com tratamento de colisões por encadeamento externo.

A função de hashing utilizada deve garantir uma espalhamento suficiente dos dados pela Tabela e a definição de uma boa função de hashing faz parte dos critérios de avaliação do trabalho.

O fator de carga da tabela deve estar entre 50 e 60; ou seja, espera-se que, em média, 50 a 60 diferentes entradas sejam mapeadas para a mesma entrada na tabela. Assim, a escolha do tamanho da tabela deve ser feita dinamicamente, após conhecida a quantidade de entradas a serem armazenadas. Uma ilustração da tabela de espalhamento é o seguinte:



h(letter) = h(train) = 0

h(traffic) = h(adjoin) = 3

Seu programa deve conter a implementação completa de HashTable, contendo operações de criação, destruição, busca, inserção e remoção de elementos. O tratamento de colisões deve utilizar istas duplamente encadeadas. O código da estrutura “List"implementado em sala de aula pode ser utilizado sem muitas alterações.

2 Entradas e Saídas

O conjunto de dados que deve ser carregado no dicionário é composto por palavras organizadas em uma arquivo de entrada chamado `dicionario.txt` no seguinte formato:

```
a: um, uma
a lot of: muito
abacus: ábaco
aback: atrás, detrás
abaft: pôpa, atrás
abalone: molusco da califórnia
abandon: abandone, abandonar
abandoned: abandonado
abandonment: abandono
abase: rebaixe, rebaixar
...
```

Cada entrada ocupa uma linha do arquivo. A palavra em inglês é seguida de ":"(dois pontos), um espaço, e o(s) seu(s) significado(s) em português. Uma palavra pode ter mais de um significado e um significado pode conter mais do que uma única palavra. Para simplificar, considere que o significado de uma palavra em inglês é toda a string após o símbolo de "dois pontos". Exemplo: o significado de abandon é "abandone, abandonar"

O arquivo dicionario.txt fornecido como exemplo contém 5900 entradas.

A função *hashing* deve ser aplicada sobre a palavra em inglês e devolver um inteiro representando a posição da palavra na tabela de *hashing*. Se houver colisões, deve ser feita uma inserção no final da lista de tratamento de colisões.

Os casos de teste serão definidos em um arquivo chamado `in.txt` contendo uma sequência de palavras em inglês, uma palavra por linha, até o final do arquivo.

Exemplo: `in.txt`

```
adrenaline
adulation
tolerate
satisfied
bechmarck
rudder
```

A saída esperada é o(s) significado(s) em português de cada palavra da entrada.

Exemplo: `out.txt`

```
adrenalina
adulação
tolere
satisfeito
-
leme
```

Se uma determinada palavra não for encontrado, deve imprimir o caractere "-"na linha correspondente ao seu significado no arquivo de saída (No exemplo, a palavra benchmark não foi encontrada no dicionário).

Em seguida, seu programa deve informar algumas estatísticas sobre a sua HashTable. Siga o modelo abaixo:

```
Quantidade de entradas da tabela (m):
Quantidade de entradas do arquivo texto.txt (n):
Fator de carga esperado (n/m):
Função de hashing aplicada:
```

3 A Avaliação

Um arquivo .tar.gz contendo TODOS os arquivos de cabeçalho e implementação deve ser entregue pelo Modle-EAD (ead.facom.ufms.br) até o final do dia 06 de julho. O programa pode ser feito em duplas e apenas um dos integrantes deve fazer a entrega; porém, lembrem-se de deixar um arquivo README.txt com o nome dos dois integrantes da dupla no pacote.

A correção será feita mediante análise e execução do programa. A qualidade da função de hashing escolhida será um critério importante da avaliação. A compilação e execução serão realizadas utilizando a sequencia de comandos abaixo:

```
$ gcc *.c -o prog
$ ./prog
```

É necessário que seu programa leia os arquivos de entrada `texto.txt` e `in.txt` e gere um arquivo de saída `out.txt` com a saída esperada.