MAC 2301 LABORATÓRIO DE ESTRUTURAS DE DADOS

DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO – USP

TERCEIRO EXERCÍCIO-PROGRAMA PRAZO DE ENTREGA: ATÉ 17/06/02

Troca de mensagens em um hipercubo

1. Introdução

Em computação paralela, uma das estruturas de mais sucesso no início dos anos 90 foi o hipercubo. Através desta estrutura pode-se interconectar vários nós de um computador para formar-se um computador paralelo. Alguns exemplos de hipercubos podem ser vistos abaixo:

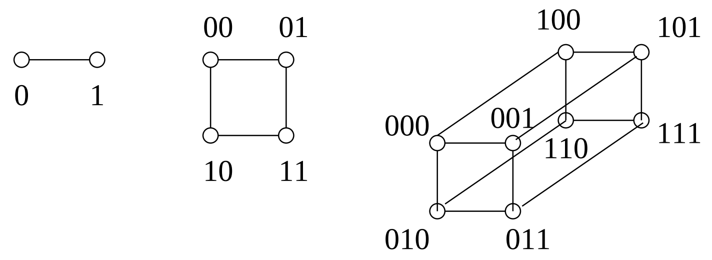

Figura 1. Hipercubos com dois, quatro e oito nós.

A definição do hipercubo pode ser feita da seguinte forma: Dado $n = 2^k$, k inteiro. Os vértices do hipercubo são os nós da forma $V = \{b_1, \dots, b_k\}$. Dois nós v_1 e v_2 são adjacentes se eles diferem em apenas uma posição, ou bit.

Além de poder ser definida de forma recursiva, o hipercubo também tem outras propriedades interessantes. Dado um hipercubo com n (potência de 2), nós cada nó está ligado a $\log n$ nós e a distância máxima entre dois nós é $\log n$.

Naturalmente, uma das operações mais usadas em computação paralela é a troca de mensagens, operação que pode ser feita de maneira muito simples, usando apenas $\log n$ passos. Basicamente o algoritmo utilizado é o seguinte.

```
para i de 1 até k faça
  troque mensagens entre os nós adjacentes na dimensão i
```

Date: 26 de maio de 2002.

Podemos ver na figura 2 as diversas etapas de troca de mensagem para um hipercubo de 8 nós.

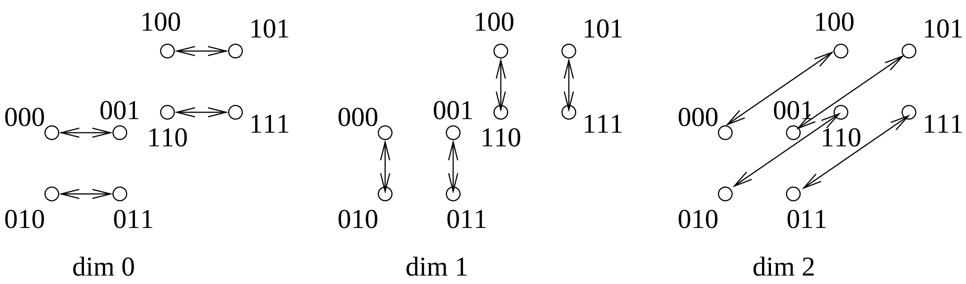

FIGURA 2. Etapas de comunicação em um hipercubos com oito nós.

Observando-se a figura vemos que uma mensagem do nó 000 com destino ao nó 111 passa pelos nós 001, 011 até chegar ao seu destino final.

2. O Exercício

2.1. **Primeira parte.** Dados k, e um nó do hipercubo $v = \{b_1, \dots, b_k\}$ faça uma função que retorna uma árvore com os nós atingíveis a partir do nó v.

A partir do nó v mudando a primeira dimensão temos $v_{\bar{1}} = \{\bar{b_1}, b_2, \dots, b_k\}$. A partir do nó v mudando a segunda dimensão temos: $v_{\bar{2}} = \{b, \bar{b_2}, \dots, b_k\}$, e a partir de $v_{\bar{1}}$, $v_{\bar{1}\bar{2}} = \{\bar{b_1}, \bar{b_2}, \dots, b_k\}$. Desta forma teremos em cada nível da árvore os nós a uma distância dada do nó v.

O seu programa também deve ser capaz de imprimir esta árvore de forma conveniente.

2.2. **Segunda parte.** Em um cenários mais complexo, podemos fornecer uma matriz A, $n \times n$, que representa mensagens a serem transmitidas no hipercubo. Podemos supor que as mensagens na coluna 1 correspondem às mensagens que o nó $0 \dots 0$ deve enviar. Na linha 1 teremos as mensagens destinadas a ele mesmo, na 2 as mensagens ao nó $0 \dots 01$, e assim por diante.

Usando o esquema de comunicação por dimensões, dados k, uma matriz de comunicação A e uma aresta do hipercubo imprima a soma de todas as mensagens que passarão por ela.

3. Especificações

O programa deve ser capaz de ler uma entrada da forma:

```
3
010
 0  2  3  4  5  6  7  8
 9  0 11 12 13 14 15 16
17 18  0 20 21 22 23 24
25 26 27  0 29 30 31 32
33 34 35 36  0 38 39 40
41 42 43 44 45  0 47 48
49 50 51 52 53 54  0 56
57 58 59 60 61 62 63  0
111 110
```

Onde a primeira linha indica o tamanho do hipercubo (no caso do exemplo com $2^3 = 8$ nós) e a segunda linha o nó a partir do qual a árvore deve ser desenhada. A partir da terceira linha são dados os tamanhos das mensagens a serem enviadas pelos processadores, e finalmente, na última linha são dados dois vértices que determinam uma aresta do hipercubo. Para o cálculo da soma das mensagens você deve **obrigatoriamente** usar a primeira parte do EP.

Para este exemplo a saída do seu programa deve ser algo como:

```
010
    011
        001
            101
        111
    000
        100
    110
```

Que representa a seguinte árvore:

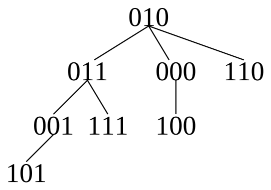

Figura 3. Árvore a partir do nó 010.

Além disto o programa também deve imprimir (ops ainda não calculei..) (que corresponde a $63 + 56 + \dots$).