UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

DEPARTAMENTO DE FÍSICA E MATEMÁTICA

CURSO: LICENCIATURA EM COMPUTAÇÃO

DISCIPLINA: PROGRAMAÇÃO PARALELA E DISTRIBUÍDA

PROFESSOR: JONES OLIVEIRA

ALUNO: JONAS FRANCISCO

PPD

Junho/2004

Programação Paralela e Distribuída

Resolução das Questões de Lista

Questão 1

Solução: Uma das vantagens do paralelismo de dados é que o processamento é independente, isto é, não se faz necessário esperar pela saída do estágio anterior para dar continuidade ao processamento. Como desvantagens podemos citar a quebrar da computação que não é um processo muito fácil, muito menos os controles da divisão e junção das unidades computadas. Aqui cada unidade produtiva (operário, processador, etc.) é responsável por toda unidade produzida. Uma outra vantagem é o fato de que a cada unidade de tempo tem-se k unidades produzidas.

Das vantagens do paralelismo de controle é válido ressaltar a baixa complexidade de processamento na produção de uma unidade, pois a tarefa é dividida em estágio. Em cada estágio é executada uma parte do processamento, cuja saída poderá ser a entrada do estágio seguinte e assim sucessivamente até que a unidade esteja completamente pronta. Uma das grandes desvantagens deste tipo de processamento é sua baixa produtividade em relação ao paralelismo de dados, pois apenas tem-se uma unidade produzida a cada t unidades de tempo.

Questão 2

Solução: Para que m1 fique pronta leva-se mit de tempo, no entanto, após este tempo tem-se apenas 1t (unidade de tempo). Daí, o tempo total para que se processe n tarefas todas com m estágios é T = (n-1)t + mt, pondo t em evidência, temos T= (n-1 + m)t.

Exemplo 1: tarefa: montar 3 carros em pipeline com 3 estágios. O primeiro carro sai após tempo 3t, do segundo carro em diante leva-se tempo 1t. Como são três carros temos 2t correspondentes a (n-1)t. Os três t acima dizem respeito aos três estágios e chamemos de mt. Os (n-1) referem-se a quantidade de carros excluindo o primeiro. Logo a expressão do tempo total é verdadeira.

Exemplo 2: tarefa : confeccionar 5 bolsas em uma linha de produção com 3 estágios, cada um consumindo 10 minutos cada. Para sair a primeira bolsa leva-se 3 vezes 10 minutos, ou seja, 30 minutos. Da segunda bolsa em diante leva-se apenas 10 minutos. Como temos apenas agora 4 bolsas o tempo restante é 4 x10 = 40 minutos, conclui-se que o tempo total para se aprontar as 5 bolsas é 70 minutos. Aplicando a equação do tempo total chegaremos ao mesmo resultado, ou seja, T=(5+3-1)t = 7x10=70 minutos.

Portanto, a fórmula é **T= (n-1 + m)t**.

Questão 3

Solução: Seja f a fração de operações numa computação que deve ser executada de forma seqüencial, em que 0=f=1. O Speedup alcançado por um computador paralelo composto de p processadores executando a computação é dado por

$$S = \frac{1}{f + \frac{(1- f)}{p}}$$

Substituindo f por 1/k, temos:

$$S = \frac{1}{1/k + \frac{(1- 1/k)}{p}}$$

$$= \frac{1}{\frac{1}{k} + \frac{\frac{k-1}{k}}{p}}$$

$$= \frac{1}{\frac{1}{k} + \frac{(k-1)}{k} \times \frac{1}{p}}$$

$$= \frac{1}{\frac{1}{k} + \frac{k-1}{kp}}$$

$$= \frac{1}{\frac{p + k - 1}{kp}}$$

$$= \frac{1}{1} \times \frac{kp}{p + k - 1}$$

Assim, $S = \frac{kp}{p + k - 1}$ fazendo kp ser f(p) e p+k-1 ser g(p), chegamos a

$S = \frac{f(p)}{g(p)}$ aplicando limite sobre S, temos: $\lim_{p \to \infty} \frac{f(p)}{g(p)} = k$.

Ou seja, o limite do Speedup quando o número de processadores tende ao infinito é uma constante k.

Demonstração:

$$\lim_{p \to \infty} \frac{f(p)}{g(p)} = \frac{\lim f(p)}{\lim g(p)} = \frac{\infty}{\infty}$$

Vemos que chegamos a uma das formas de indeterminação, ou seja, 0/0 ou 8/8 e que devemos aplicar a regra de L´Hôpital que diz: $\lim_{x \to 8} \frac{f(x)}{g(x)} = \lim \frac{f'(x)}{g'(x)}$

onde f´(x) e g´(x) são as derivadas de x.

Obs.: Não devemos derivar o quociente e sim cada parte individualmente.

Voltando às variáveis reais do problema, ou seja, k e p, temos:

f(p) = kp e g(p) = p + k –1 e $\lim_{p \to 8} \frac{f(p)}{g(p)} = \frac{f'(p)}{g'(p)}$

sabendo que $f'(p) = \frac{d(kp)}{dp} = k$ e que $g'(p) = \frac{d(p + k - 1)}{dp} = 1$, temos que

$\frac{f'(p)}{g'(p)} = k = \mathbf{k}$, conforme queríamos provar.

Questão 5

Sejam as matrizes $A_{\mathbf{lxm}}$ e $B_{\mathbf{mxn}}$ pretende-se gerar uma matriz $C_{\mathbf{lxn}}$ como produto de $A_{\mathbf{lxm}}$ por $B_{\mathbf{mxn}}$ usando o algorítmo:

```
for(i = 0; i <  l; i++){
  for(j = 0;  j < n; j++){
      t = 0;
      for(k = 0; k < m;k++){
          t = t + A[i][k] * B[k][j];
      }
     C[i][j] = t;
  }
}
```

Partindo do loop mais interno vemos que o número de operações é **m.**
Subindo um nível, ou seja, o segundo loop, seu número de iterações é **n**.
Chegando ao primeiro, vê-se que o mesmo ocorre **l** (L minúsculo) vezes.
Então, o nível de complexidade do algoritmo é: **l x m x n**

Questão 5.1

Baseado na complexidade do algoritmo acima, podemos dizer que para gerar uma matriz produto a partir de duas de dimensões **n x n** tal complexidade vai para **n x n x n**, ou seja $\mathbf{n^3}$
Logo, a complexidade do algoritmo é Theta($\mathbf{n^3}$).

Questão 5.2

O algoritmo é ótimo porque gera uma complexidade logarítmica em vez de poligarítmica.

Questão 5.3 – Ver teorema de Brent.