O Modelo PRAM

Jones Albuquerque
DFM-UFRPE

2004, Recife - PE.

PRAM

PRAM - *Parallel Random Access Mahine*

Trata poder de processamento como recurso ilimitado

O modelo não é realista: ignora a complexidade para comunicação entre processadores

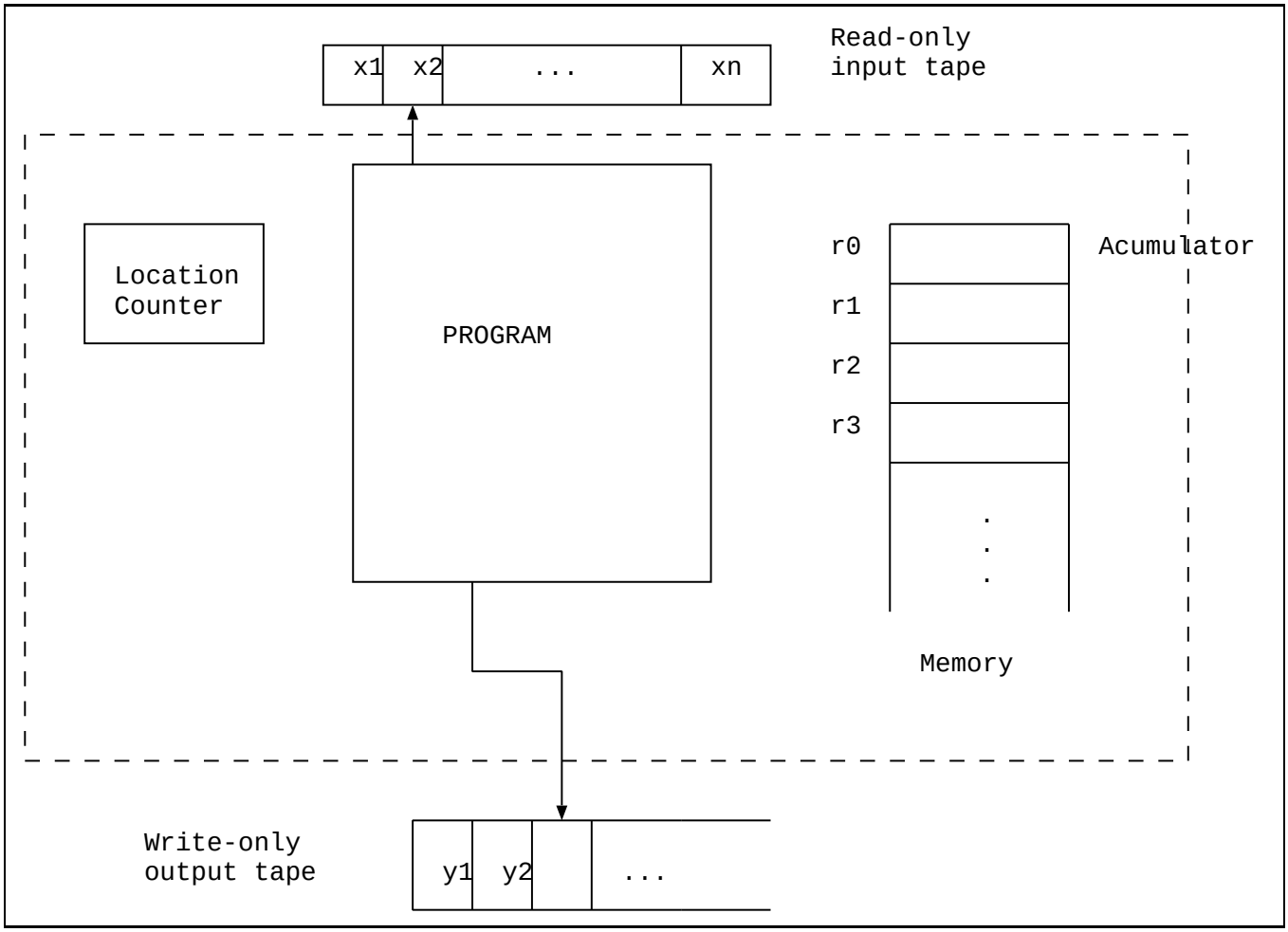
Preocupação com o paralelismo inerente da computação

Otimalidade em PRAM significa que o número total de operações realizadas por um algoritmo numa PRAM é da mesma classe de complexidade do algoritmo seqüencial ótimo

RAM - Um modelo Seqüencial

RAM - *Random Access Machine*

RAM consiste de uma memória, uma fita de entrada *read-only*, uma fita de saída *write-only* e um programa



RAM - Funcionamento

O programa não é armazenado na memória e não pode ser modificado

A fita de entrada contém uma seqüência de inteiros.

A cada "vez", um valor é lido, e a cabeça de leitura avança um campo de leitura

A cabeça de escrita funciona de maneira semelhante: avança um campo a cada escrita

A memória consiste de uma seqüencia ilimitada de registradores, chamados $r_0, r_1, r_2, \ldots$

Cada registrador consegue armazenar um único inteiro

O registrador r_0 é o acumulador, onde computações são realizadas

RAM - Observações

Complexidade de tempo no pior caso é a função $f(n)$, o máximo tempo gasto por um programa para executar sobre todas as entradas de tamanho n, isto é $x_1, x_2, \ldots, x_n$

Complexidade de tempo no caso médio é a média, sobre todas as entradas de tamanho n, dos tempos de execução

As definições de complexidade de espaço no pior caso e no caso médio são análogas as de tempo, exceto pela substituição da palavra "tempo" por "espaço" nas definições anteriores

RAM - Observações

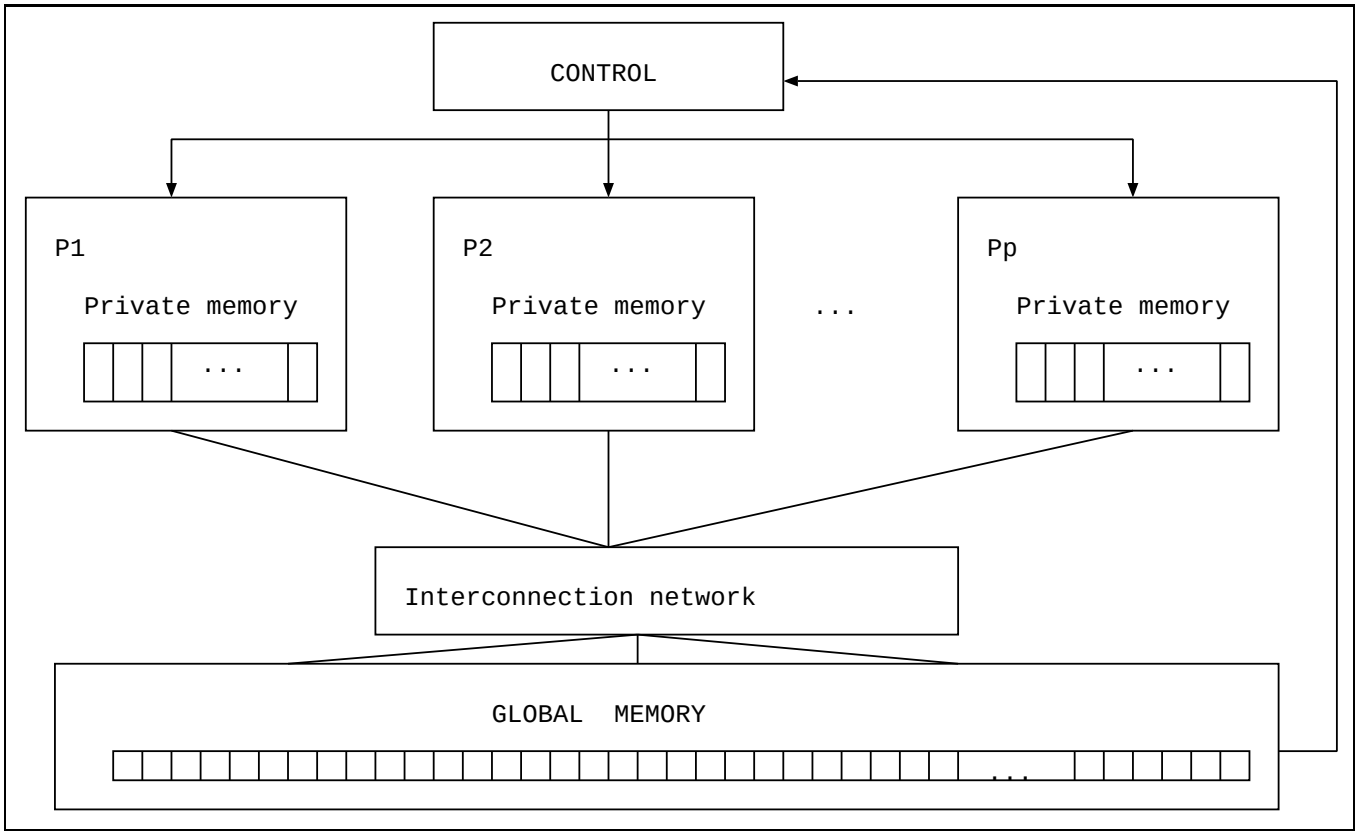
Existem duas maneiras de medir tempo e espaço no modelo RAM:

1. critério de **custo uniforme**: cada instrução executa em uma unidade de tempo e todo registrador ocupa uma unidade de espaço

2. critério de **custo logaritmo** considera que uma palavra real de memória tem uma capacidade limitada de armazenamento. Este é o adotado pelo livro-texto

PRAM - Um Modelo Paralelo

PRAM consiste de uma unidade de controle, memória global e uma conjunto ilimitado de processadores, cada um com sua própria memória local



PRAM - Funcionamento

Embora os processadores ativos executem instruções idênticas, cada processador tem um único índice. Este índice pode ser utilizado para habilitar ou desabilitar o processador ou influenciar no acesso à memória

Uma computação começa com a entrada armazenada em memória global e um único elemento de processamento

A cada passo de computação um elemento ativo (processador habilitado) pode ler um valor de sua memória local ou da global, executar uma operação simples (tipo RAM) e escrever o resultado ou na memória local ou na global

PRAM - Funcionamento

Durante a computação, um processador pode ativar (habilitar) um outro processador

Todos os processadores ativos devem executar a mesma instrução, embora em posições de memória diferentes

A computação termina quando o último processador termina (*halt*)

PRAM - Definições

O **custo** de uma computação PRAM é o produto da complexidade de tempo paralela e o número de processadores usados

Por exemplo, um algoritmo PRAM que tenha complexidade $\theta(log\ p)$ usando p processadores tem custo $\theta(p\ log\ p)$

PRAM - Modelos

Os modelos diferem em como eles tratam conflitos de leitura e escrita, ou seja, quando dois ou mais processadores tentam ler de ou escrever para a mesma posição memória global

Os modelos PRAM segundo Li e Yesha em 1989

EREW *Exclusive Read Exclusive Write*
conflitos de leituras e escritas não são permitidos

CREW *Cocurrent Read Exclusive Write*
leitura concorrente permitida (múltiplos processadores podem ler a mesma posição da memória global no mesmo passo de instrução). Conflitos de escrita não são permitidos (PRAM Default)

PRAM - Modelo CRCW

CRCW *Concurrent Read Concurrent Write*
leitura e escrita concorrente permitidas. Há uma variedade de modelos para os CRCW de acordo com as pilíticas para tratar escritas à mesma posição de memória global

1. **COMMON** - todos os processadores que escrevem na mesma posição de memória devem escrever o mesmo valor

2. **ARBITRARY** - dos vários processadores que desejam escrever numa mesma posição de memória, apenas um (1) é escolhido arbitrariamente

3. **PRIORITY** - dos vários processadores que desejam escrever numa mesma posição de memória, o processador com menor índice é escolhido

Modelos PRAM - Observações

O modelo EREW é o mais fraco.

É fácil ver que um CREW executa qualquer algoritmo EREW na mesma quantidade de tempo, basta não utilizar a facilidade de leitura concorrente. Similarmente, um CRCW pode executar qualquer algoritmo CREW na mesma quantidade de tempo

PRIORITY é o modelo mais forte

Qualquer algoritmo projetado para um COMMON executará com a mesma complexidade de em um ARBITRARY e PRIORITY. Como todos os processadores escrevem o mesmo valor quando concorrem para uma escrita, a escolha de um deles é indiferente

Modelos PRAM - Observações

Como PRIORITY é mais forte que EREW, um algoritmo para resolver um problema num EREW pode ter tempo de complexidade maior que o mesmo problema em um PRIORITY

Lema. (Cole 1988) Um EREW com p-processadores consegue ordenar um vetor de p-elementos armazenado em memória global em $\theta(log\ p)$ time

PRAM - Teorema

Teorema. (Eckstein 1979 e Vishkin 1983) Um PRIORITY com p-processadores pode ser simulado por um EREW com p-processadores em uma complexidade de tempo acrescida por um fator de $\theta(log\ p)$

Prova: