

# Conceitos de PAD

## Processamento de Alto Desempenho

### *Taxonomia Básica*



Programa Interdisciplinar de Pós-Graduação em Computação Aplicada  
Universidade do Vale do Rio dos Sinos

# Conceitos básicos

- Existem vários níveis de concorrência...

<b>Granularidade</b>	<b>Nível</b>	<b>Exemplo</b>
Muita fina	Intra-instrução	Proc Superescalares
Fina	Entre-Instruções	Proc Vetoriais
Fina/Média	Blocos	UMA
Média	Procedimentos	UMA/NUMA
Grossa	Processos	SC-NUMA/NORMA
Muito Grossa	Aplicações	Comp em grade ?

# Conceitos básicos

- Existem vários níveis de concorrência...

<b>Granularidade</b>	<b>Nível</b>	<b>Exemplo</b>
Muita fina	Intra-instrução	Proc Superescalares
Fina	Entre-Instruções	Proc Vetoriais
Fina/Média	Blocos	UMA
Média	Procedimentos	UMA/NUMA
Grossa	Processos	SC-NUMA/NORMA
Muito Grossa	Aplicações	Comp em grade ?

# Taxonomia

- **Programa concorrente**
  - ◊ Decomposto em atividades independentes
- **Programa paralelo**
  - ◊ Atividades são executadas em processadores independentes, potencialmente ao mesmo tempo
- **Programa distribuído**
  - ◊ As atividades executam em processadores independentes, cada um com sua própria memória

# Taxonomia

- **Programação concorrente**
  - ♦ Técnica de programação que explora a **independência** temporal de atividades definidas por uma aplicação
  - ♦ **A resolver:**
    - **Compartilhamento** de dados
    - **Cooperação** entre atividades

# Taxonomia

- **Programação concorrente**
  - ♦ **Desenvolver um programa concorrente possui o mesmo grau de dificuldade do desenvolvimento de um programa seqüencial ?**
    - **Sim**: O problema básico da programação continua o mesmo: dado um problema, definir as instruções a serem executadas pelo computador
    - **Não**: Novos itens devem ser tratados, como detecção das atividades potencialmente concorrentes, o compartilhamento dos recursos (dados) e a estrutura da colaboração entre tarefas

# Taxonomia

- **Dado**
  - ♦ Informação manipulada no programa
  - ♦ Sofre as ações de transformação
- **Tarefa**
  - ♦ Atividade de cálculo para transformação de um dado
- **Sincronizaç o**
  - ♦ Mecanismo de controle de acesso aos dados
  - ♦ Forte:
    - Prevê uma relação de produção e consumo de dados
  - ♦ Fraca:
    - Não define uma ordem no acesso aos dados

# Taxonomia

- Exemplo de sincronização:

- ♦ Forte:

```
void Fibor( int n, int* res ) {  
    if( n <= 2 )  
        return 1;  
    else {  
        Fibor( n-1, r1 );  
        Fibor( n-2, r2 );  
  
        return r1+r2;  
    }  
}
```

# Taxonomia

- Exemplo de sincronização:

- ♦ Forte:

```
void Fibor( int n, int* res ) {
    if( n <= 2 )
        return 1;
    else {
        Fibor( n-1, &r1 );
        Fibor( n-2, &r2 );
        SYNC;
        return r1+r2;
    }
}
```

# Taxonomia

- Exemplo de sincronização:
  - ♦ Fraca:

Global:

```
int x;  
sync m;
```

```
void Add( int v ) {  
    Obtem(m)  
    x = x + v;  
    Libera(m)  
}
```

```
void Dec( int v ) {  
    Obtem(m)  
    x = x - v;  
    Libera(m)  
}
```

# Taxonomia

- Exemplo de sincronização:

- ♦ Fraca:

Global:

```
int x;  
sync m;
```

```
void Add( int v ) {  
    Obtem(m)  
    x = x + v;  
    Libera(m)  
}
```

```
void Dec( int v ) {  
    Obtem(m)  
    x = x - v;  
    Libera(m)  
}
```

# Processo de Desenvolvimento

- 1) Escolha de um modelo para a máquina;
- 2) Construa uma representação para o algoritmo;
- 3) Escolha de um modelo de execução;
- 4) Procura por um escalonamento;
- 5) Identifique a arquitetura;
- 6) Procure uma ferramenta de programação;
- 7) Implemente.