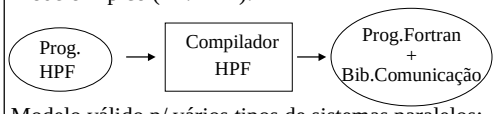
# Formas de Programação Paralela

**Tópicos**:
- Compilação de Linguagens de Alto Nível
- Padrão OpenMP
- Comparação entre Abordagens



# Compilação de Linguagens de Alto Nível

Modelo Típico (Ex: HPF):




Modelo válido p/ vários tipos de sistemas paralelos:
- Mem. Distribuída: Comunicação = troca de mensagens
- Mem. Compartilhada: Comunic. = variáveis comuns

Na prática:
- Mesmo prog. HPF pode executar em vários sistemas



# Compilação de Linguagens de Alto Nível (cont.)

- O mesmo vale para programas em MPI:
  - Sist. Memória Distribuída: *Send/Recv* = mensagem
  - Sist. Memória Compartilhada: *Send/Recv* = cópia/mem.
- Logo, para Sist. Memória Compartilhada, há várias formas possíveis de programação:
  - Usando múltiplos *threads* (ex: pthreads)
  - Usando troca de mensagens (ex: MPI)
  - Usando linguagens de alto nível (ex: HPF)

⇒ Qual a forma mais conveniente? Outras formas ???



# Padrão OpenMP

- Idéia Básica:
  - Simplificar o porte de programas seriais p/ ambientes paralelos de memória compartilhada
- Implementação:
  - Uso de diretivas de compilação nas linguagens normais
- Situação até 1997:
  - Cada fabricante tinha suas próprias diretivas
- Em 1997: Criação do padrão OpenMP
  - Busca de portabilidade entre sistemas
  - Alternativa ao uso de MPI



# Padrão OpenMP (cont.)

- Construção básica em OpenMP:
  - Região paralela, com múltiplos *threads*

Exemplo:

```
        print *,'init'
!$omp parallel private(iam,iglob)
        iglob = omp_get_num_threads( )
        iam=omp_get_thread_num( )
        print *,'I am',iam,' of', iglob
!$omp end parallel
        print *,'all done'
```



# Padrão OpenMP (cont.)

- Uso efetivo: loops paralelizados

Exemplo:

```
        real a( ), b( ), c( ), w, x
        . . .
        w = entrada( )
!$omp parallel do private(x), shared(a,b,c,w)
        do i = 1 , n
            x = w * ( a(i) + b(i) )
            c(i) = sqrt ( x )
        end do
```

## Padrão OpenMP (cont.)

- Divisão das iterações do loop pelos *threads*:
  - Estática: Cada thread pega um conjunto fixo (e pré-determinado) de iterações
  - Dinâmica: À medida que termina uma iteração, cada *thread* vai recebendo uma nova iteração

    Vantagens: melhor equilíbrio de trabalho pelos processadores, nos casos em que o tempo por iteração for variável (melhor balanceamento de carga)
  - Variações: Caso dinâmico, com *k* iterações sendo distribuídas a um certo processador, ao invés de 1



## Comparação Entre Abordagens

| | X3H5 | MPI | Pthreads | HPF | OpenMP |
|---|---|---|---|---|---|
| Scalable | no | yes | sometimes | yes | yes |
| Incremental parallelization | yes | no | no | no | yes |
| Portable | yes | yes | yes | yes | yes |
| Fortran binding | yes | yes | no | yes | yes |
| High level | yes | no | no | yes | yes |
| Supports data parallelism | yes | (no) ? | (no) | yes | yes |
| Performance oriented | no | yes | no | tries | yes |