# Exploração do paralelismo em um algoritmo do processamento digital de sinais: critério para sua avaliação.

Dr. Darcy Domingues Novo
Chefe do Depto. Eletrônica Aplicada
ITA - Instituto Tecnológico de Aeronáutica, SJC

Engº Mateus de Azevedo Faria
Chefe do Depto. de Processamento Digital de Sinais
TECNASA Eletrônica Profissional S/A, SJC

Resumo

*Os algoritmos utilizados no processamento digital são considerados como bem apropriados à execução por meio de múltiplos processadores operando em paralelo. Objetivando uma verificação prática, um algoritmo de filtragem será tratado por uma abordagem de realização paralela.*

*Será apresentado em seguida um critério para avaliação de desempenho.*

## 1. Enunciado do problema

Segundo Gajski e Peir [1], existem basicamente quatro escolas de pensamento relacionando o que é mais importante para se obter um desempenho de ordem de grandeza superior em um sistema.

A primeira acredita que a tecnologia cada vez mais rápida será suficiente para aumentar o desempenho, o que implica em manter a atual arquitetura dos sistemas, possivelmente aumentada com um mecanismo de sincronização utilizando-se processadores em paralelo.

A segunda escola coloca a prioridade na otimização ou vetorização de compiladores, possivelmente de forma interativa, que detetará paralelismos e ajudará os usuários a escreverem melhores programas.

Uma terceira escola acredita que um dramático aumento em desempenho virá principalmente de novos algoritmos paralelos de forma a poder suportar desenvolvimentos de novas linguagens que permitirão a fácil conversão desses algoritmos em programas.

A quarta escola suporta novos modelos de computação tais como máquinas data flow, que devem permitir dramáticos aumentos em paralelismo e pode facilmente ser explorado pela arquitetura de processadores no sentido de aumentar o desempenho.

Entretanto, na opinião desses dois autores, nenhuma destas escolas mostrou como combinar as necessidades computacionais atuais com a capacidade realizadora de novas tecnologias de VLSI que são por eles consideradas, não um degrau na evolução mas sim uma verdadeira revolução na concepção de novas máquinas.

Por isso propõem uma quinta escola de pensamento dizendo que o tratamento a ser dado a multiprocessadores necessita introduzir três novas exigências não colocadas até então:

- primeiro: cada problema deve ser separado em tarefas ( *partição*);
- segundo : cada tarefa deve ser planejada para ser executada por um ou mais processadores ( *planejamento ou atribuição* );
- terceiro : sincronização do contrôle e fluxo de dados precisam ser realizados durante a execução.

Adotando-se tais idéias, observa-se que, aplicando-as ao tratamento do problema de processamento digital de sinais, é fundamental que a divisão do processo de tratamento em tarefas e sua conveniente partição entre processadores que as executarão, podem levar a resultados mais convenientes.

Assim um campo de pesquisas se abre no sentido de procurar dentro do processamento digital de sinais paralelismos especiais ao mesmo tempo que se procura critérios para determinar sua eficiência.

Neste campo, dois grandes ramos existem: a filtragem digital e a determinação da transformada de Fourier. Em qualquer um deles o problema consiste na implementação de tal forma que a relação entrada/saída seja convertida num algorítmo computacional conveniente, tal como já foi realizado pela FFT - Transformada Rápida ( ou Finita ) de Fourier. Estamos particularmente interessados em pesquisar a existência de paralelismo em algoritmos para tratamento de filtros digitais que possam eventualmente ser implementados em dispositivos que utilizem as técnicas de VLSI.

## 2. Granularidades fina e grossa. Partição.

Os algorítmos para o processamento digital de sinais são especificados em termos de um conjunto de computações elementares ordenadas de adição, atraso e multiplicação por uma constante. É possível representá-los por uma estrutura composta de conexões e nós, dispostos em forma de rede orientada, de tal forma que as conexões representem as operações e os nós, os resultados obtidos. Designaremos tal rede por diagrama de fluxo.

Em um diagrama de fluxo representativo de um algoritmo, poderemos identificar a existência de ramos sequenciais e paralelos.

Serão considerados sequenciais aqueles onde a execução de cada elemento computacional tem um único precedente e um consequente de tal forma que somente poderemos chegar a um resultado se passarmos pelo elemento precedente. Observe-se que para se obter um resultado parcial será necessário um certo intervalo de tempo, dependente das características do processador utilizado. Os ramos paralelos, por outro lado, podem ser computados por processadores independentes.

A própria computação da adição e da multiplicação poderia ser explorada visando detetar nos seus algoritmos possíveis paralelismos. Quando isto é feito, dizemos que se está procurando uma granularidade fina. Por outro lado, quando se deseja pesquisar outros paralelismos possivelmente existentes no algorítmo do tratamento digital do sinal, supondo-se que os algorítmos da adição e da multiplicação já foram considerados otimizados, dizemos que estamos tratando de uma granularidade grossa.

A escolha de uma particular granularidade visa estabelecer um compromisso entre velocidades de execução de cada tarefa elementar e as possíveis penalidades ( *overhead* ) provenientes da sincronização necessária entre os processadores para executar o algoritmo.

É claro que uma granularidade grossa não explora a totalidade do paralelismo possível e se coloca assim o problema da conveniência ou não de fazê-lo. No que segue suporemos que as únicas operações que serão executadas pelo algoritmo são as indicadas e estaremos, portanto, adotando o tratamento do problema do ponto de vista da granularidade grossa.

Vejamos agora o conceito de partição. Ela consiste na divisão de um algoritmo em tarefas, módulos e processos. Tarefas ( ou procedimentos ) são as grandes atividades que compõem o algoritmo. Módulos ( ou sub-tarefas) são partes do programa que executam uma função definida e completa. Já os processos correspondem a um menor nível computacional relacionadas com atividades de menor porte tais como as operações aritméticas.

A granularidade do processo está diretamente relacionada com a partição das atividades entre os processadores que executarão o algoritmo.

## 3. Algoritmo de filtragem IIR

Consideremos um sistema que executa a amostragem de sinais analógicos seguido de um filtro digital, um conversor digital/análogo e um conveniente filtro passa baixo, tal como ilustrado na figura 1.


Fig. 1: Sistema de filtragem digital

As variáveis envolvidas neste sistema são:

$x(t)$ : valor analógico do sinal de entrada ( contínuo )
$y(t)$ : valor analógico do sinal de saída ( contínuo )
$x(n)$ : valor digitalizado do sinal de entrada no instante n
$y(n)$ : valor digitalizado do sinal de saída no instante n

A chave indicada por A simboliza o amostrador.

Entende-se por filtragem digital não recursiva aquela que relaciona o valor atual da amostra de saída $y(n)$ com o valor atual e anteriores das amostras digitalizadas da entrada $x(n)$. Por outro lado, a filtragem recursiva considera além destes valores, certos valores da própria saída, anteriores ao instante considerado.

Tais filtros obedecem portanto as seguintes relações matriciais :

$$y(n) = \mathbf{x}^T(n)\, \mathbf{a}_N \qquad \text{( filtragem não recursiva)}$$

e

$$y(n) = \mathbf{x}^T(n)\, \mathbf{a}_N + \mathbf{y}^T(n-1)\, \mathbf{b}_M \qquad \text{( filtragem recursiva)}$$

onde

$y(n)$ : valor da saída no instante n

e as matrizes $\mathbf{x}(n)$, $\mathbf{y}(n-1)$, $\mathbf{a}_N$ e $\mathbf{b}_M$ são:

$$\mathbf{x}(n) = \begin{bmatrix} x(n) \\ x(n-1) \\ x(n-2) \\ \vdots \\ x(n-N) \end{bmatrix} \qquad \mathbf{y}(n-1) = \begin{bmatrix} y(n-1) \\ y(n-2) \\ \\ \\ y(n-M) \end{bmatrix}$$

$$\mathbf{a}_N = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix} \qquad \mathbf{b}_M = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{bmatrix}$$

e $\mathbf{x}^T(n)$ e $\mathbf{y}^T(n)$ as respectivas transpostas.

Considerando-se o algorítmo da filtragem não recursiva como particularização da forma recursiva quando $b_M = 0$, somente esta última será analisada. Admitiremos além disso que M, definido como a ordem do filtro, será sempre maior ou igual a N. Os filtros recursivos são também denominados de **Infinite Impulse Response**, ou **IIR**. A figura 2 mostra um diagrama de fluxo representativo do filtro **IIR**.

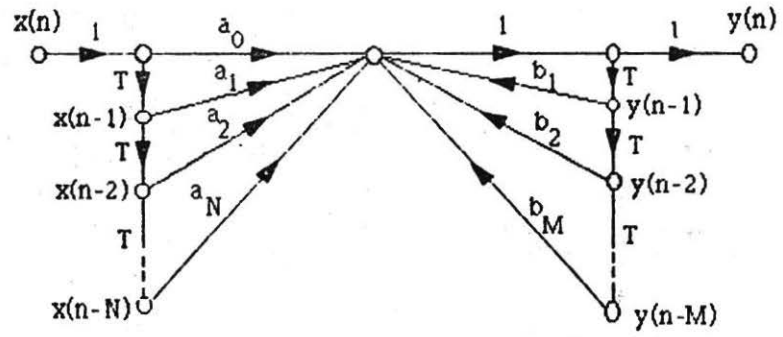


Fig. 2: Diagrama representativo do Filtro IIR

É bom lembrar que existem outras formas possíveis para o diagrama de fluxo da filtragem IIR além da forma canônica acima. Outra prática comum é representar o filtro por meio de secções de segunda ordem em cascata, semelhante às implementações de filtros analógicos. Das diferentes formas de representação podem-se obter diversas abordagens de quebra e granularidade.

## 4. Quebra do algoritmo proposto

A fim de explorar o paralelismo inerente ao algorítmo do filtro IIR será proposta uma abordagem de quebra.

Admitindo-se que a execução das adições será feita com dois operandos de cada vez, o algorítmo do filtro IIR pode ser representado na forma indicada pela figura 3.

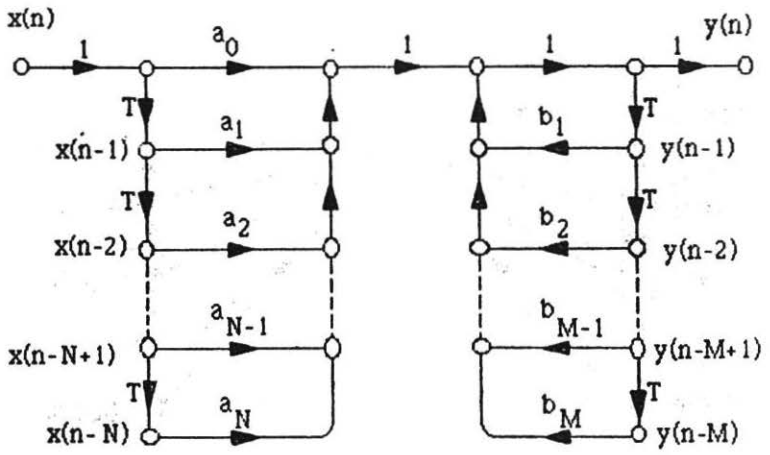


Fig. 3: Algorítmo do Filtro IIR

Este diagrama de fluxo pode ser visualizado numa escala de tempo onde os valores atrasados de entrada e de saída são tomados como os valores passados ( ou anteriores se preferirem ) em relação ao instante atual. Desta forma procura-se uma representação dinâmica para as sequências a serem executadas pelo algoritmo tal como indicado na figura 4.

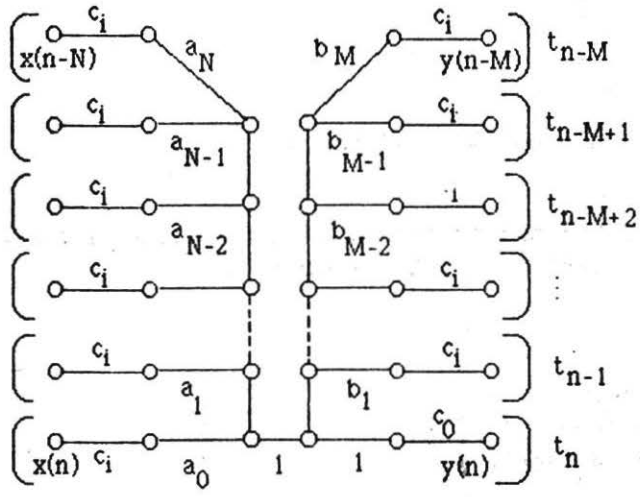


Fig. 4 : Diagrama de fluxo temporal da filtragem IIR

onde

$t_n$ : instante de tempo atual

$t_{n-1}$ a $t_{n-M}$ : instantes anteriores

$c_i$ : comunicação do processador para obtenção das amostras de entrada ou saída necessárias para o cálculo

$c_o$ : comunicação do processador para fornecimento de uma amostra de saída, filtrada pelo algorítmo IIR.

A figura 4 representa a sequência temporal de atividades a ser realizada em um processador, a fim de obter uma amostra filtrada da saída y(n). Observa-se assim que são necessárias as seguintes atividades:

| | |
|---|---|
| N+M+1 | comunicações de entrada ($c_i$) |
| N+M+1 | produtos ( p ) |
| N+M | adições ( a ) |
| 1 | comunicação de saída ( $c_o$) |

Para se proceder à análise de um caso particular, admitiremos que os tempos de execução necessários para que as atividades sejam efetivamente realizadas, obedeçam as especificações que seguem:

| | | | |
|---|---|---|---|
| $c = c_i = c_o$ | : 1 | unidade de tempo | ( 1 u.t. ) |
| a | : 2 | unidades de tempo | ( 2 u.t.) |
| p | : 16 | unidades de tempo | ( 16 u.t.) |

A unidade de tempo ( u.t. ), obviamente, depende do processador específico que for escolhido para a implementação real.

Nestas condições, o tempo requerido para o cálculo de uma amostra de saída, por um processador único é

$$t_1 = 19\,N + 19\,M + 18 \quad \text{unidades de tempo.}$$

Analisemos agora a possibilidade de execução do mesmo algorítmo utilizando múltiplos processadores. Inicialmente deveremos atribuir um número mínimo de operações que devam ser executadas em cada processador. Tal conjunto de operações será denominado **processo** e, por hipótese, é indivisível.

Um processo conterá, no máximo, 2 comunicações, 2 produtos e 2 adições, o que implica que o tempo máximo de execução de um processo ($t_\theta$) será igual a 38 u.t..

Para execução do algorítmo utilizando P processadores, é necessário que se faça uma partição do algorítmo entre eles. Por exemplo, um filtro de ordem M terá M + 1 processos a serem executados por P processadores. Consideremos o caso em que P = M + 1. A figura 5 mostra a atribuição dos processos ao longo do tempo

| Tempo / $P_i$ | T | 2 T | 3 T | 4 T | | M T | (M+1)T | (M+2)T | (M+3)T |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | 1 | 2 | 3 | 4 | | M | [M+1] | 1 | 2 |
| $P_2$ | [M+1] | 1 | 2 | 3 | | M-1 | M | [M+1] | 1 |
| $P_3$ | M | [M+1] | 1 | 2 | | M-2 | M-1 | M | [M+1] |
| $P_4$ | M-1 | M | [M+1] | 1 | | M-3 | M-2 | M-1 | M |
| $P_5$ | M-2 | M-1 | M | [M+1] | | M-4 | M-3 | M-2 | M-1 |
| | | | | | | | | | |
| $P_{M+1}$ | 2 | 3 | 4 | 5 | | [M+1] | 1 | 2 | 3 |

Fig. 5 : Execução do filtro IIR de ordem M por M+1 processadores.

Em cada intervalo de tempo entre amostras todos os processadores executam apenas um processo. A sequência de execução é a mesma para cada processador porém os processos são defasados para serem por eles executados.

Num dado intervalo de tempo n, o processador que executa o ( M + 1)-ésimo processo é quem fornece a saída y(n). Assim a ordem de obtenção das saídas do filtro para a figura 5 é

$$P_2, P_3, P_4, \ldots\ldots, P_M, P_{M+1}, P_1, P_2, \ldots\ldots$$

E este é o caso ideal, onde o número de processadores se iguala ao número de processos. Durante todo o tempo são executados M + 1 processos em paralelo.

O intervalo de tempo entre as amostras de saída é dado pelo tempo de execução de um processo apenas; logo

$$(t_p)_{P=M+1} = t_e = 38 \text{ u.t.}$$

O tempo $t_1$ requerido no caso de um único processador, para o filtro de ordem M , ( M = N ), é

$$t_1 = 38\,M + 18 \text{ u.t.}$$

Comparando-se então os intervalos de tempo obtidos nos dois casos obtemos a relação

$$(t_1/t_p)_{P=M+1} = P - 10/19$$

Nota-se assim que a redução de tempo depende linearmente do número de processadores P. É interessante portanto procurar um critério que permita avaliar o desempenho do sistema. É o que faremos a seguir.

## 5. Critério de desempenho

Tomaremos como critério para verificação do desempenho do paralelismo o aproveitamento no tempo dos vários processadores utilizados. Consideremos então, por definição,

$$E_P = (t_1/t_P)/ P = t_1/( P . t_P )$$

onde

$E_P$ : coeficiente de desempenho com P processadores;

$t_1$ : intervalo de tempo entre amostras de saída para a solução com um processador;

$t_P$ : intervalo de tempo entre amostras de saída para a solução com P processadores.

Para um aproveitamento máximo, $E_P$ deve ser igual a 1. Isto ocorre na solução com um processador, onde não há perda de tempo em atividades redundantes ou então por falta de atividade.

$$E_1 = 1$$

O desempenho com M + 1 processadores implementando um filtro de ordem M, utilizando-se os valores adotados no item 4 é

$$E_{M+1} = (M + 9/19) / ( M + 1)$$

O gráfico da figura 6 a seguir mostra o desempenho em função da ordem do filtro M, supondo-se P= M + 1.

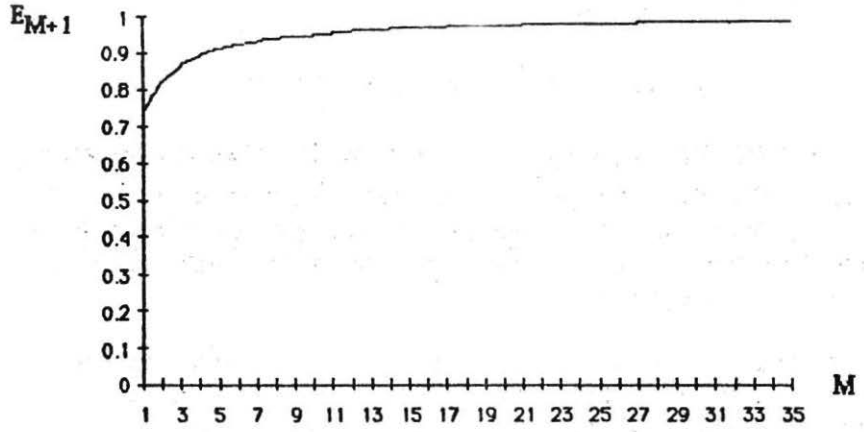


Fig. 6 : Desempenho com M+1 processadores

A implementação do filtro poderá ser feita com número menor de processadores, desde que cada um execute mais de um processo por intervalo de tempo. Consideremos, por exemplo, 4 processadores executando um filtro de ordem M = 5. A figura 7 mostra uma possível atribuição dos 6 processos para os quatro processadores.

| Tempo / $P_i$ | T | 2 T | 3 T | 4 T | 5 T | 6 T | 7 T | 8 T | 9 T | 10 T | 11 T |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | 1<br>3 | 2<br>4 | 3<br>5 | 4<br>[6] | 5<br>1 | [6]<br>2 | 1<br>3 | 2<br>4 | 3<br>5 | 4<br>[6] | 5<br>1 |
| $P_2$ | [6]<br>2 | 1<br>3 | 2<br>4 | 3<br>5 | 4<br>[6] | 5<br>1 | [6]<br>2 | 1<br>3 | 2<br>4 | 3<br>5 | 4<br>[6] |
| $P_3$ | 5 | [6] | 1 | 2 | 3 | 4 | 5 | [6] | 1 | 2 | 3 |
| $P_4$ | 4 | 5 | [6] | 1 | 2 | 3 | 4 | 5 | [6] | 1 | 2 |

Fig 7 : Execução de um filtro IIR de ordem 5 por 4 processadores

A ordem de obtenção das saídas é

$$P_2,\ P_3,\ P_4,\ P_1,\ P_2,\ P_1,\ P_2,\ P_3,\ P_4,\ P_1,\ P_2,\ \ldots\ldots$$

Os processadores $P_1$ e $P_2$ tem carga maior de trabalho: 2 processos durante o intervalo de tempo entre amostras. Os processadores P3 e P4 têm menor carga, ficando sem atividade durante metade desse intervalo de tempo. Isto representa, no nosso ponto de vista, uma queda no aproveitamento dos processadores.

Adotando-se ainda as unidades de tempo atribuídas anteriormente no exemplo do item 4, podemos estabelecer uma expressão para valores arbitrários de M e P ( P maior ou igual a 2) dada por

$$E_P = \frac{38 \cdot M + 18}{38 \cdot \{\frac{M+1}{P}\}^* \cdot P}$$

onde $E_P$ : coeficiente de desempenho com P processadores;

M : ordem do filtro;

P : número de processadores;

38 : unidades de tempo necessárias para o cálculo de um processo com dados adotados no item 4;

$\{\ \}^*$ : **menor inteiro** maior ou igual ao valor contido.

A figura 8 mostra o coeficiente de desempenho de P processadores realizando um filtro de ordem M = 11, com os tempos adotados no item 4 e supondo, para simplificar, que N = M.

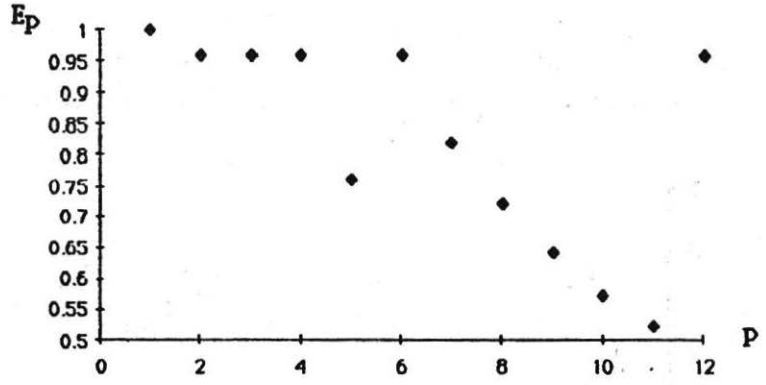


Fig. 8 : Desempenho versus número de processadores num filtro IIR de ordem 11

A figura 8 mostra que o coeficiente de desempenho ótimo, obtido no caso em que P = M+1 = 12, é também atingido com 2, 3, 4 ou 6 processadores, significando que, para estes casos, há máxima utilização dos mesmos. Como exemplo, numa implementação com 11 processadores, pode-se dizer que 5 deles estarão inativos durante quase todo o tempo, pois aproximadamente a metade dos recursos disponíveis não é utilizada. Obviamente, o desempenho é máximo quando apenas um processador executa todas as tarefas, sendo utilizado na totalidade do tempo.

A expressão do coeficiente de desempenho pode ser generalizada para unidades de tempo arbitrárias. Indiquemos por a, p e c os tempos necessários para execução da adição, produto e comunicação, respectivamente, de um dado processador.

Nestas condições poderemos escrever

$$E_p = \frac{(N+M)\cdot a + (N+M+1)\cdot p + (N+M+2)\cdot c}{2\cdot\left\{\frac{M+1}{p}\right\}^{*}\cdot(a+p+c)\cdot P}$$

A figura 9 mostra a variação do desempenho em função da relação entre a ordem do filtro e o número de processadores, onde os valores $E_{max}$ e $E_{min}$ dependem dos parâmetros N, M, a, p e c de cada caso particular.

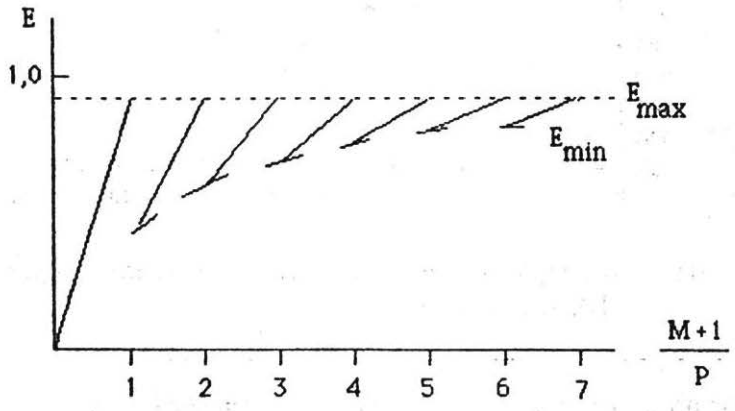


Fig. 9 : Coeficiente de desempenho em função da ordem do filtro e do número de processadores.

Na abordagem de quebra do algorítmo estudado, para se obter o máximo desempenho do sistema, é necessário que a ordem do filtro mais um seja um múltiplo inteiro do número de processadores utilizados. Como parâmetro de projeto, um determinado número de processadores P poderá implementar, com máximo desempenho, uma família de filtros IIR de ordens $M = P.k - 1$, onde k representa um número inteiro.

## 6. Conclusão

Os algorítmos do processamento digital de sinais apresentam, em geral, grande regularidade e raramente possuem desvios no fluxo de controle em função dos dados. Estas características têm se mostrado fundamentais para a execução paralela com alto aproveitamento dos recursos computacionais.

Este artigo mostrou um exemplo de aplicação do processamento paralelo, indicando o potencial dessa técnica na área do processamento digital de sinais. Entretanto, há ainda um vasto campo de possibilidades a serem exploradas.

## 7. Agradecimentos

Estudos de aspectos de tratamento digital de sinais e sua interrelação com processamento paralelo foram originalmente propostos pelo Dr. E. Davis, da Universidade da Carolina do Norte, EUA, quando de sua estada no ITA em convênio patrocinado pela IBM. Os autores agradecem pelas suas sugestões. Os autores agradecem também o considerável apoio da Tecnasa permitindo que esta pesquisa fosse levada adiante. O assunto e outros aspectos correlatos serão ainda mais desenvolvidos em tese de Mestrado.

## BIBLIOGRAFIA

1. D. D. Gajsli & J. K. Peir - Essential Issues in Multiprocessor Systems, Computer Magazine, June 1985.
2. E. Davis - Anotações de curso de pós graduação : Processamento Digital de sinais: algorítmos e arquiteturas, ITA ,1986
3. J. P. Brafman, J. Szczupak & S. K. Mitra - An Approach to the Implementation of Digital Filters Using Microprocessors.
   IEEE Trans. on Acoustics, Speech and Digital Signal Processing, Vol ASSP 26, Nº 05, Oct. 78
   A. V. Oppenheim R. W. Schafer - Digital Signal Processing
   Prentice Hall, 1975