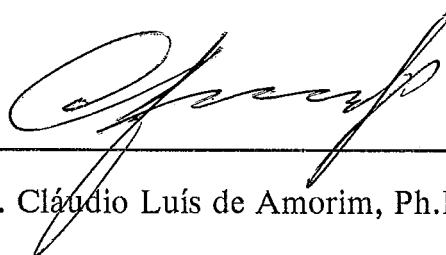


# INVESTIGAÇÃO DE PARALELISMO NA MIGRAÇÃO $\omega$ -X

Silvio Sinedino Pinheiro

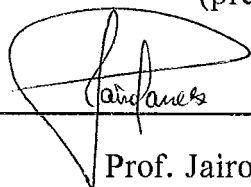
TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO EM ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

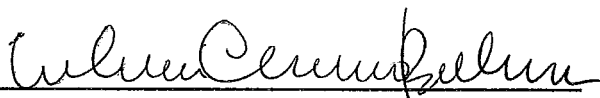


Prof. Cláudio Luís de Amorim, Ph.D.

(presidente)



Prof. Jairo Panetta, Ph.D.



Prof. Valmir Carneiro Barbosa, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

OUTUBRO DE 1993

PINHEIRO, SILVIO SINEDINO

Investigação de paralelismo na migração  $\omega$ -x [Rio de Janeiro] 1993,

XI, 91 p. 29,7 cm (COPPE/UFRJ, M.Sc.,

Engenharia de Sistemas e Computação, 1993)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Processamento Paralelo 2. Processamento Sísmico

3. Migração (sísmica)

I. COPPE/UFRJ II. Título (série)

"Vivendo se aprende; mas o que se aprende, mais, é só a fazer outras maiores perguntas."

Guimarães Rosa

Para

Ana e meus filhotes Raul, Letícia e Bárbara.

## AGRADECIMENTOS

À PETROBRÁS por ter possibilitado este mestrado e pelo suporte operacional a este trabalho.

A todos os companheiros de trabalho pelo apoio recebido, entre os quais destaco: Amaral, Anselmo, Beth, Bia, Celso, Chaia, Credilson, Francis, Gilberto, Haroldo, Ismael, Ivan, Joaquim, José Eduardo, Luiz Alberto, Márcia, Marcos, Mônica, Paulo Osório, Valdirene e Vandemir.

Aos amigos da minha turma da COPPE/Sistemas pela camaradagem, especialmente Evande, Nalvo, Nahri, Paulo, Raquel, Valério e Vitória.

Ao pessoal administrativo da COPPE/Sistemas pelo apoio, especialmente à Ana Paula.

À Profa. Lúcia Drummond (UFF) pela eficiência e simpatia.

Ao Prof. Jairo Panetta pela dedicação na orientação.

À minha mãe pelo esforço e obstinação.

À minha querida Ana pela força, carinho e compreensão, sempre.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para obtenção do grau de Mestre em Ciências (M.Sc.)

## INVESTIGAÇÃO DE PARALELISMO NA MIGRAÇÃO $\omega$ -X

Silvio Sinedino Pinheiro

OUTUBRO DE 1993

Orientadores: Cláudio Luís de Amorim

Jairo Panetta

Programa: Engenharia de Sistemas e Computação

A prospecção de petróleo através da sísmica de reflexão exige um processamento computacional muito intenso. Com o surgimento das primeiras máquinas paralelas comerciais é de grande interesse saber da viabilidade do uso de paralelismo no processamento sísmico.

Este trabalho tem como motivação básica ajudar a responder esta dúvida. Para isto escolheu-se um processo representativo da complexidade computacional do processamento sísmico que é a Migração Sísmica.

Foram desenvolvidas duas formas de paralelização para a migração  $\omega$ -x que foram experimentadas em máquinas paralelas de memória distribuída, de memória central e em rede de estações trabalhando como uma máquina paralela.

Após a análise dos resultados obtidos nos experimentos conclui-se pela viabilidade do paralelismo no processamento sísmico e ressalta-se a coerência destes resultados com trabalhos correlatos recentes.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfilment of the requirements for the degree of Master of Science (M.Sc.)

## INVESTIGATION OF PARALLELISM IN $\omega$ -X MIGRATION

Silvio Sinedino Pinheiro

OCTOBER, 1993

Thesis Supervisors: Cláudio Luís de Amorim

Jairo Panetta

Department: Computing and Systems Engineering

Oil prospection using the seismic reflection method demands intense computational effort. With the appearance of commercial parallel machines it is important to know about the viability of using parallelism in seismic processing.

The basic motivation of this work is to help clarify this doubt. To that end the seismic migration algorithm was chosen as representative of the computational complexity of the seismic processing.

Two forms of parallelism were developed for the  $\omega$ -x migration and tested in parallel machines with distributed memory, central memory and a cluster of workstations working as a parallel machine.

The analysis of the results obtained in the experiments lead to the conclusion of the viability of the use of parallelism in seismic processing. This conclusion is in accordance with recent related works.

# Índice

|  |           |
|--|-----------|
| <b>Lista das Ilustrações</b> .....   | <b>ix</b> |
| <b>Lista das Tabelas</b> .....   | <b>x</b>  |
| <b>Capítulo 1 - Introdução</b> .....   | <b>1</b>  |
| <b>Capítulo 2 - Sumário do Método Sísmico de Reflexão</b> .....              | <b>4</b>  |
| 2.1 - A Aquisição Sísmica .....  | 4         |
| 2.2 - O Processamento Sísmico .....  | 9         |
| 2.2.1- Demultiplexação .....   | 10        |
| 2.2.2- Correção de Decaimento de Amplitude .....                             | 10        |
| 2.2.3- Correção dinâmica .....   | 12        |
| 2.2.4- Análise de Velocidades .....  | 13        |
| 2.2.5- Correção Estática .....   | 13        |
| 2.2.6- Empilhamento .....  | 15        |
| 2.2.7- Migração Sísmica .....  | 15        |
| <b>Capítulo 3 - A Migração Sísmica</b> .....                                 | <b>16</b> |
| 3.1 - O que é a Migração .....   | 16        |
| 3.2 - O Impacto da Migração .....  | 16        |
| 3.3 - Métodos de Migração .....  | 23        |
| 3.3.1 - Formulação Matemática .....  | 23        |
| 3.3.2 - Migração Phase-Shift .....   | 25        |
| 3.3.3 - Migração W-X .....   | 27        |
| <b>Capítulo 4 - Paralelismo na Migração W-X</b> .....                        | <b>35</b> |
| 4.1 - Alguns Fatos sobre a Implementação .....                               | 35        |
| 4.2 - Análise da Complexidade do Algoritmo .....                             | 37        |
| 4.3 - Dependências nos Laços Centrais .....                                  | 38        |
| 4.4 - Formas de Paralelismo .....  | 40        |
| 4.5 - Análise da Complexidade das Formas de Paralelismo .....                | 42        |
| 4.5.1 - Análise da Complexidade da Primeira Forma .....                      | 42        |
| 4.5.2 - Análise da Complexidade da Segunda Forma .....                       | 43        |
| <b>Capítulo 5 - Experimentos e Resultados em Máquinas Hipercúbicas</b> ..... | <b>46</b> |
| 5.1 - Primeira Forma de Paralelização .....                                  | 46        |

|  |           |
|--|-----------|
| 5.1.1 - Descrição da Implementação   | 46        |
| 5.1.2 - Descrição dos Experimentos   | 47        |
| 5.1.3 - Apresentação e Análise de Resultados no INTEL/iPSC                   | 47        |
| 5.1.4 - Apresentação e Análise de Resultados no NCP I/COPPE                  | 51        |
| 5.2 - Segunda Forma de Paralelização   | 55        |
| 5.2.1 - Descrição da Implementação   | 55        |
| 5.2.2 - Descrição dos Experimentos   | 56        |
| 5.2.3 - Apresentação e Análise de Resultados no INTEL/iPSC                   | 56        |
| 5.3 - Conclusões   | 57        |
| <b>Capítulo 6 - Experimentos e Resultados em Máquinas de Memória Central</b> | <b>59</b> |
| 6.1 - Primeira Forma de Paralelização  | 59        |
| 6.1.1 - Descrição da Implementação   | 59        |
| 6.1.2 - Descrição dos Experimentos   | 60        |
| 6.1.3 - Apresentação e Análise dos Resultados para o IBM 3090                | 60        |
| 6.1.4 - Apresentação e Análise dos Resultados para o IBM 9021                | 63        |
| 6.2 - Segunda Forma de Paralelização   | 66        |
| 6.2.1 - Descrição da Implementação   | 66        |
| 6.2.2 - Apresentação e Análise dos Resultados para o IBM 9021                | 67        |
| 6.3 - Terceira Forma de Paralelização  | 68        |
| 6.3.1 - Descrição da Implementação   | 68        |
| 6.3.2 - Apresentação e Análise dos Resultados para o IBM 9021                | 68        |
| 6.4 - Conclusões   | 70        |
| <b>Capítulo 7 - Experimentos e Resultados em Rede de Estações</b>            | <b>71</b> |
| 7.1 - Primeira Forma de Paralelização  | 73        |
| 7.1.1 - Descrição da Implementação   | 73        |
| 7.1.2 - Descrição dos Experimentos   | 73        |
| 7.1.3 - Apresentação e Análise de Resultados na Rede RS-6000/PVM             | 74        |
| 7.1.4 - Expressão Analítica para o Tempo de Execução                         | 76        |
| 7.1.5 - Calibração e Validação   | 79        |
| 7.1.6 - Extremos   | 80        |
| 7.2 - Segunda Forma de Paralelização   | 82        |
| 7.2.1 - Descrição da Implementação   | 82        |
| 7.2.2 - Descrição dos Experimentos   | 82        |
| 7.2.3 - Apresentação e Análise de Resultados na Rede RS-6000/PVM             | 82        |
| 7.3 - Conclusões   | 85        |
| <b>Capítulo 8 - Conclusões e Futuros Trabalhos</b>                           | <b>87</b> |
| <b>Referências Bibliográficas</b>  | <b>90</b> |



## Lista das Ilustrações

|  |    |
|--|----|
| Figura 1. Disposição das fontes e receptores             | 5  |
| Figura 2. Cobertura CMP                                  | 6  |
| Figura 3. Reflexões em camadas horizontais               | 7  |
| Figura 4. Reflexões em camadas mergulhantes              | 7  |
| Figura 5. Reflexões sísmicas em lanço split              | 8  |
| Figura 6. Sismograma antes do ganho AGC                  | 11 |
| Figura 7. Sismograma após ganho AGC                      | 11 |
| Figura 8. Sismograma antes da correção dinâmica          | 12 |
| Figura 9. Sismograma após a correção dinâmica            | 13 |
| Figura 10. Efeito da correção estática                   | 14 |
| Figura 11. Efeito da migração sísmica                    | 15 |
| Figura 12. Frente de onda no espaço                      | 17 |
| Figura 13. Campo de onda no modelo do refletor explosivo | 18 |
| Figura 14. Relação entre mergulhos                       | 19 |
| Figura 15. Princípios de Migração                        | 20 |
| Figura 16. Variação de posição com a migração            | 22 |

## Lista das Tabelas

|             |   |    |
|-------------|---|----|
| Tabela 3.1  | Deslocamento horizontal e vertical .....            | 23 |
| Tabela 4.1  | Tempos de execução seqüencial .....                 | 36 |
| Tabela 4.2  | Tempos percentuais de execução .....                | 36 |
| Tabela 4.3  | Comportamento dos tempos de execução .....          | 38 |
| Tabela 5.1  | INTEL/iPSC - Tempos de execução c/ 1 proc. ....     | 48 |
| Tabela 5.2  | INTEL/iPSC - Tempos de execução c/ 4 procs. ....    | 49 |
| Tabela 5.3  | INTEL/iPSC - Ganhos com 4 procs. ....               | 49 |
| Tabela 5.4  | INTEL/iPSC - Tempos de execução c/ 8 procs. ....    | 50 |
| Tabela 5.5  | INTEL/iPSC - Ganhos com 8 procs. ....               | 50 |
| Tabela 5.6  | NCP I/COPPE - Tempos de execução c/ 1 proc. ....    | 51 |
| Tabela 5.7  | NCP I/COPPE - Tempos de execução c/ 4 procs. ....   | 52 |
| Tabela 5.8  | NCP I/COPPE - Ganhos com 4 procs. ....              | 53 |
| Tabela 5.9  | NCP I/COPPE - Tempos de execução c/ 8 procs. ....   | 54 |
| Tabela 5.10 | NCP I/COPPE - Ganhos com 8 procs. ....              | 54 |
| Tabela 5.11 | INTEL/iPSC - Tempos de execução c/ 8 procs. ....    | 56 |
| Tabela 5.12 | INTEL/iPSC - Ganhos com 8 procs. (forma 2) .....    | 57 |
| Tabela 6.1  | IBM 3090 - Tempos de execução (forma 1) .....       | 61 |
| Tabela 6.2  | IBM 3090 - Ganhos com a forma 1 .....               | 61 |
| Tabela 6.3  | IBM 3090 - Ganhos com a forma 1a .....              | 62 |
| Tabela 6.4  | IBM 3090 - Ganhos com a forma 1b .....              | 62 |
| Tabela 6.5  | IBM 9021 - Tempos de execução (forma 1) .....       | 63 |
| Tabela 6.6  | IBM 9021 - Ganhos com a forma 1 .....               | 64 |
| Tabela 6.7  | IBM 9021 - Ganhos com a forma 1a .....              | 65 |
| Tabela 6.8  | IBM 9021 - Ganhos com a forma 1b .....              | 65 |
| Tabela 6.9  | IBM 9021 - Tempos de execução (forma 2) .....       | 67 |
| Tabela 6.10 | IBM 9021 - Ganhos com a forma 2 .....               | 67 |
| Tabela 6.11 | IBM 9021 - Tempos de execução (forma 3) .....       | 69 |
| Tabela 6.12 | IBM 9021 - Ganhos com a forma 3 .....               | 69 |
| Tabela 7.1  | RS-6000 - Tempos de execução (exceções) .....       | 72 |
| Tabela 7.2  | RS-6000 - Tempos de execução seqüencial .....       | 72 |
| Tabela 7.3  | RS-6000 - Tempos de execução c/ 4 ests. ....        | 74 |
| Tabela 7.4  | RS-6000 - Ganhos com 4 ests. ....                   | 74 |
| Tabela 7.5  | RS-6000 - Tempos de execução c/ 6 ests. ....        | 75 |
| Tabela 7.6  | RS-6000 - Ganhos com 6 ests. ....                   | 75 |
| Tabela 7.7  | RS-6000 - Composição percentual instrumentada ..... | 76 |
| Tabela 7.8  | RS-6000 - Tempos medidos X previstos .....          | 79 |
| Tabela 7.9  | RS-6000 - Distribuição tempos processamento .....   | 80 |
| Tabela 7.10 | RS-6000 - Número procs X ganho ótimo .....          | 81 |
| Tabela 7.11 | RS-6000 - Tempos de execução c/ 4 ests. ....        | 83 |
| Tabela 7.12 | RS-6000 - Ganhos com 4 ests. (forma2) .....         | 83 |

|             |   |    |
|-------------|---|----|
| Tabela 7.13 | RS-6000 - Tempos de execução c/ 6 ests. ....        | 84 |
| Tabela 7.14 | RS-6000 - Ganhos com 6 ests. (forma2) .....         | 84 |
| Tabela 7.15 | RS-6000 - Composição percentual execução .....      | 85 |
| Tabela 8.1  | RS-6000 - Hierarquia capacidade computacional ..... | 88 |

## Capítulo 1 - Introdução

Na fase inicial da prospecção do petróleo procura-se definir os melhores locais para a perfuração dos poços. Isto exige um conhecimento profundo da disposição das camadas geológicas em subsuperfície, na área de interesse.

O conhecimento preciso da geologia de uma bacia sedimentar só é possível através da perfuração de poços. Entretanto isto é extremamente caro. Assim, torna-se necessária a utilização de outros métodos mais baratos, sem comprometer o objetivo final de indicar o posicionamento correto das estruturas mais favoráveis à ocorrência de óleo.

Para isto são utilizados os chamados métodos indiretos de prospecção geofísica como a gravimetria, a magnetometria e a sísmica de reflexão. Dentre eles destaca-se o Método Sísmico de Reflexão. Este método compreende três etapas: Aquisição, Processamento e Interpretação.

Na Aquisição Sísmica são geradas, na superfície, ondas acústicas que se propagam nas camadas em subsuperfície. Ao encontrar um meio com características petrofísicas diferentes (litologia, densidade, velocidade de propagação) parte da energia é refletida e o restante é refratada. A energia refletida é captada na superfície por dispositivos especiais e é gravada digitalmente em meio magnético.

O processo de construção de imagens representativas da geologia em subsuperfície a partir destes dados se dá através da aplicação de diversos processos computacionais, constituindo o que é conhecido como Processamento Sísmico.

O Processamento Sísmico, apesar de largamente utilizado, é limitado pela capacidade computacional existente. O volume de dados a processar num levantamento típico em duas dimensões, 2-D, é grande (da ordem de centenas de

Mb) e alguns processos são computacionalmente muito intensos ( $O(n^3)$ ). Assim, muitas vezes, estes processos tornam-se praticamente inviáveis, sendo postergados para fases avançadas do processamento, quando o volume de dados já se encontra bastante reduzido. Nestes casos, não é todo o potencial do processo que pode ser aproveitado pela sua aplicação tardia.

A Interpretação dos Dados Sísmicos é a análise dos dados coletados e processados, com base no conhecimento geológico da região, procurando identificar nas camadas geológicas em subsuperfície a localização de situações mais favoráveis à ocorrência de jazidas de óleo e gás.

São exatamente os intérpretes, com suas cabidas exigências de uma maior resolução e quantidade de informações nos dados processados, que requerem a utilização de técnicas mais complexas e computacionalmente intensivas que demandam o uso de computadores progressivamente mais poderosos.

Assim, nem os atuais super-computadores onde é executado o processamento sísmico tornam factível a aplicação de tais processos.

O surgimento das primeiras máquinas paralelas comerciais traz a indagação do seu desempenho no Processamento Sísmico. Conhecidas as restrições destas máquinas com relação à entrada/saída de dados e à comunicação entre processadores, vem a pergunta: "Será viável a utilização das máquinas paralelas no Processamento Sísmico ? "

Ajudar a responder esta pergunta é a motivação básica desta tese de mestrado.

Para isto, escolheu-se um processo representativo da intensidade computacional do Processamento Sísmico que é a Migração Sísmica. Esta tese então, objetiva avaliar diversas formas de paralelização da Migração Sísmica em algumas máquinas paralelas: INTEL/iPSC, NCP-1/COPPE, IBM 3090/600VF, IBM 9021/820 e uma rede de estações IBM RS-6000 funcionando como uma máquina paralela através do uso do PVM (Parallel Virtual Machine).

Após grande quantidade de experimentos onde os ganhos mostraram-se muito bons em todas as máquinas testadas, conclui-se pela viabilidade do uso de paralelismo no Processamento Sísmico.

Este trabalho está organizado do seguinte modo: no capítulo 2, apresenta-se um sumário do método sísmico de reflexão para situar o leitor no ambiente da sísmica. No capítulo 3, há uma explicação intuitiva da migração e a apresentação de dois métodos de migração, o Phase-Shift e o  $\omega$ -x. No capítulo 4, é discutido o paralelismo na migração  $\omega$ -x. Nos capítulos de 5 a 7 são descritos os experimentos e discutidos os resultados obtidos nos diversos equipamentos testados. Finalmente, no capítulo 8 são apresentadas as conclusões e delineados os futuros trabalhos.

## Capítulo 2 - Sumário do Método Sísmico de Reflexão

Este capítulo introduz o Método Sísmico de Reflexão e um de seus componentes, a Migração Sísmica. Descreve-se a aquisição e alguns métodos de processamento.

### 2.1 - A Aquisição Sísmica

No método de reflexão sísmica são geradas, na superfície, ondas elásticas que se propagam nas camadas em subsuperfície.

As fontes sísmicas podem ser vibracionais ou explosivas para aquisição terrestre e ar comprimido (air-gun) ou canhão hidráulico (water-gun), entre outras, para aquisição marítima.

Uma parte da energia sísmica gerada reflete-se nas interfaces das camadas retornando à superfície, onde é coletada por *geofones* (no caso terrestre) ou *hidrofonos* (no caso marítimo), que convertem vibrações em sinais elétricos.

Os sinais convertidos são transmitidos para um sismógrafo que os recebe por varredura, isto é, a cada intervalo determinado de tempo, chamado *intervalo de amostragem de aquisição*, o aparelho registra os sinais recebidos, gravando-os digitalmente em fita magnética.

A cada disparo da fonte sísmica, que é chamado de *tiro*, um conjunto determinado de geofones (ou hidrofonos) é ativado, recebendo as reflexões durante um período de tempo que é definido em função da profundidade da área de interesse.

A idéia é que conhecido o tempo decorrido entre o disparo da fonte e o registro da reflexão, pode-se determinar a profundidade do refletor (isto é, a interface das camadas que ocasionou a reflexão), desde que se conheça a velocidade do meio.

A partir daqui, serão usados termos de aquisição terrestre. A aquisição marítima é, em geral, análoga.

Para uma aquisição 2-D, em geral, define-se uma linha, denominada *linha sísmica*, sobre a qual são marcados os pontos de disparo da fonte sísmica, *pontos de tiro*, e a posição do conjunto de geofones. Após cada tiro, o ponto de tiro e o conjunto de geofones são deslocados sobre a linha.

A configuração geométrica dos tiros e geofones, chamada *lanço*, é determinada em função da profundidade, das características da área de interesse e dos tipos de ruído existentes no local. Uma ilustração da geometria de aquisição descrita acima pode ser vista na figura 1.

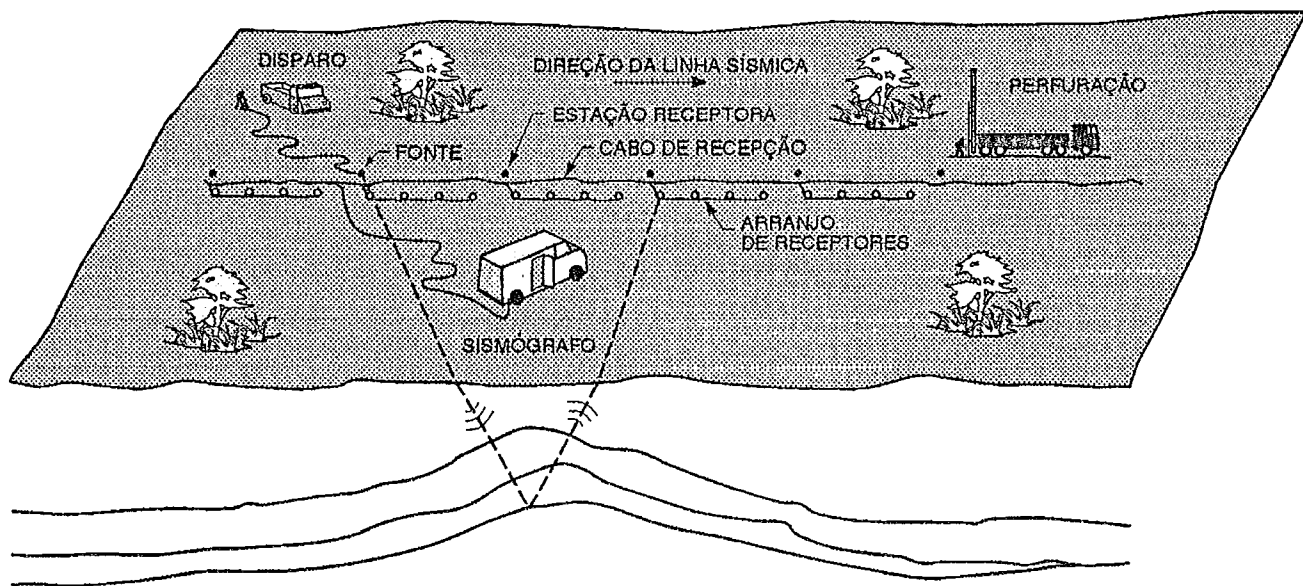


Figura 1. Disposição das fontes e receptores: (Adaptação de [Robinson80]).



A aquisição geralmente utiliza a técnica dos pontos médios comuns, *CMP*<sup>1</sup>, que consiste na sobreposição de informações oriundas de pares tiro-geofone que possuem o mesmo ponto médio.

A técnica CMP gera uma multi-cobertura de dados com benefícios na melhoria da qualidade dos dados, na atenuação dos ruídos e na estimativa das velocidades em subsuperfície. A figura 2 mostra a cobertura múltipla gerada pela técnica CMP.

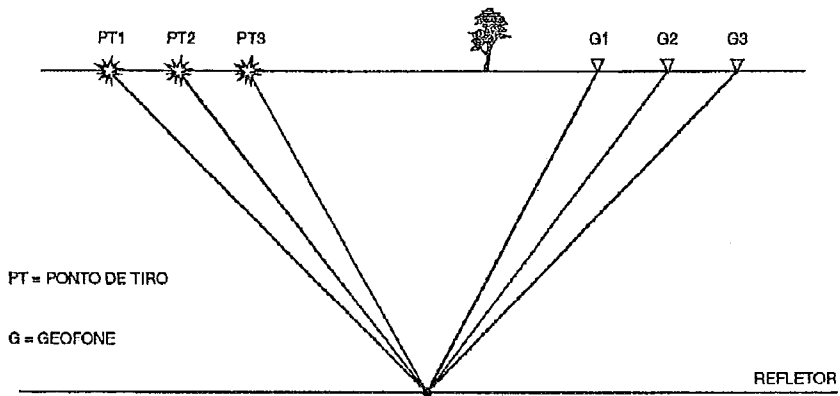


Figura 2. Cobertura CMP: Princípio básico da cobertura múltipla (Adaptação de [Hatton86]).

Como pode ser visto na figura 3, para refletores horizontais, um geofone está amostrando reflexões de pontos situados no plano dos pontos médios entre o tiro e o geofone. O afastamento entre o ponto de tiro e o geofone é denominado *offset*.

<sup>1</sup>*Common Midpoint*

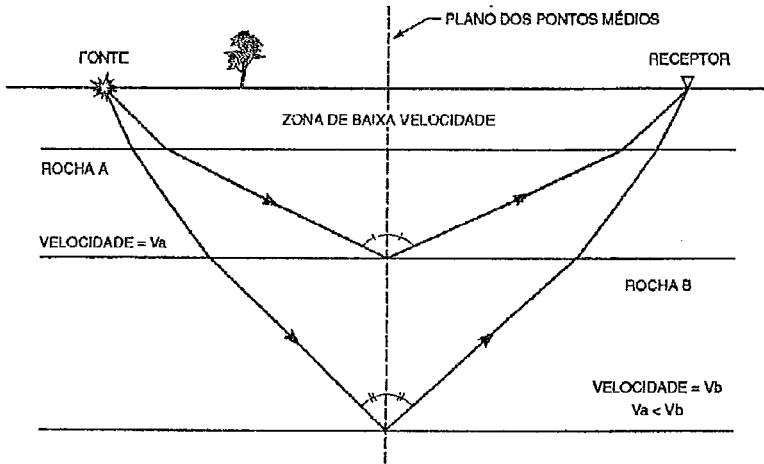


Figura 3. Reflexões em camadas horizontais: Caminhos dos raios aos pontos médios, para um meio de camadas horizontais (Adaptação de [Hatton86]).

Entretanto, se a subsuperfície contém camadas não horizontais, *mergulhantes*, (que é o caso usual) a técnica CMP induz distorções, amostrando pontos laterais como pertencentes ao plano dos pontos médios, como mostra a figura 4.

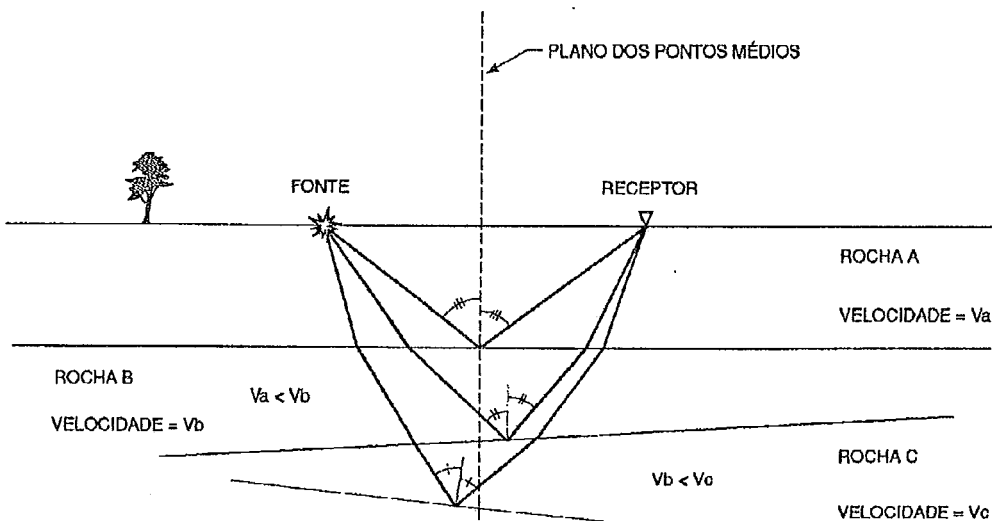


Figura 4. Reflexões em camadas mergulhantes: Caminho dos raios aos "pontos médios", considerando o efeito das camadas mergulhantes. (Adaptação de [Hatton86]).

O conjunto de dados recebidos por um determinado geofone durante um tiro é denominado *traço sísmico* (computacionalmente um vetor de amostras correspondentes a instantes sucessivos de tempo).

O tempo durante o qual as reflexões são recebidas é chamado de *comprimento do traço sísmico*. O número de amostras que compõem o traço sísmico é igual ao comprimento do traço sísmico dividido pelo intervalo de amostragem de aquisição.

A figura 5 retrata os traços sísmicos recebidos pelos geofones durante o registro de um tiro, para um lanço *split* (o ponto de tiro fica entre dois conjuntos de geofones).

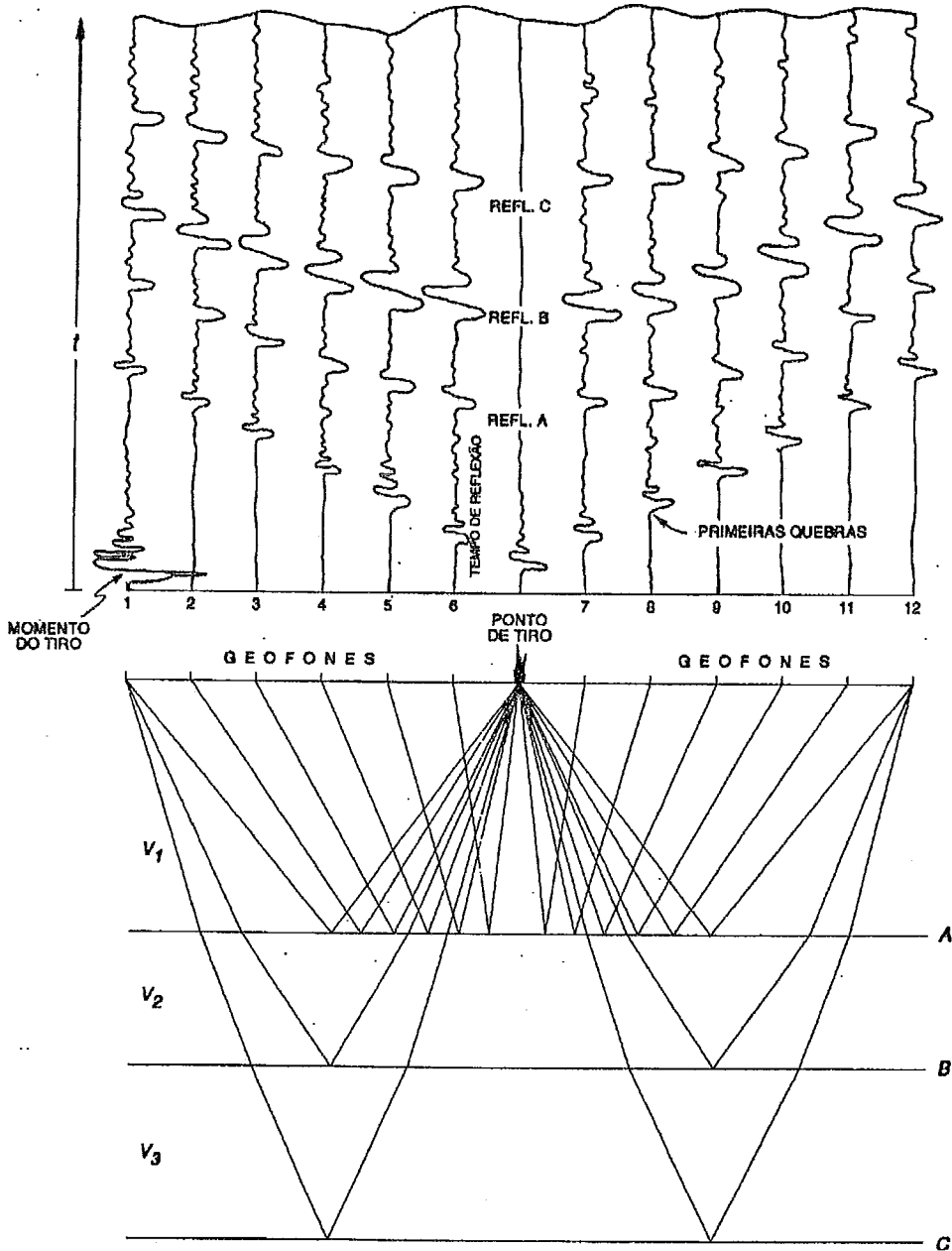


Figura 5. Reflexões sísmicas em lanço split: (Adaptação de [Bentz61]).

Os eventos indicados na figura 5 como *primeiras quebras*<sup>2</sup> devem-se principalmente à energia refratada na interface da primeira camada abaixo da superfície que é usualmente denominada *zona de baixa velocidade*.

Nos casos de camadas horizontais, as reflexões atingem primeiramente os geofones mais próximos do ponto de tiro. Como também pode ser observado na figura 5, as curvas dos tempos de chegada das reflexões aos geofones, em função da distância destes ao ponto de tiro, são aproximadamente hiperbólicas.

## 2.2 - O Processamento Sísmico

O objetivo desta etapa da prospecção é construir imagens geológicas interpretáveis da subsuperfície a partir dos dados sísmicos. Alguns dos processos aplicados visam eliminar os ruídos incorporados aos dados pela aquisição, recuperar as amplitudes relativas nos traços sísmicos e posicionar corretamente as estruturas em subsuperfície.

Para isto são necessários computadores de alto desempenho, técnicas matemáticas de processamento de sinais e a habilidade subjetiva do geofísico de processamento.

Um resultado do processamento é uma aproximação da seção sísmica de offset zero (fonte e receptor na mesma posição), isto é, todos os traços gerados pelos pares tiro-geofone que têm o mesmo CMP são colapsados em um único traço.

Seria o equivalente a um experimento hipotético em que a fonte e o receptor estivessem na mesma posição. Esta posição é o ponto médio das posições da fonte e do receptor no experimento real, isto é, o próprio CMP. O processo de colapsar todos estes traços em um único traço é chamado *empilhamento*.

A apresentação dos traços sísmicos será chamada de *sismograma* e de *seção sísmica* antes e após o empilhamento, respectivamente.

A seguir são descritas e exemplificadas as principais fases do processamento:

---

<sup>2</sup>*first breaks*

### **2.2.1- Demultiplexação**

Não é um processo geofísico e sim uma conversão de formato de gravação de dados. Existem dois modos de aquisição de dados: um é o modo multiplexado, em que o sismógrafo grava seqüencialmente todas as primeiras amostras do primeiro ao último geofone, todas as segundas amostras do primeiro ao último geofone e assim sucessivamente até a gravação das últimas amostras do primeiro ao último geofone.

O outro é o modo demultiplexado, em que o sismógrafo grava seqüencialmente da primeira à última amostra do primeiro geofone, da primeira à última amostra do segundo geofone e assim sucessivamente até da primeira à última amostra do último geofone.

Apesar do modo demultiplexado ser o mais moderno, ainda se encontram muitos dados adquiridos no modo multiplexado. Nestes casos, é necessária a aplicação de um conversor que coloque os dados no formato demultiplexado que é a forma em que os dados são processados.

### **2.2.2- Correção de Decaimento de Amplitude**

Uma análise das amplitudes de um traço sísmico mostra que há uma diminuição destas amplitudes à medida que o tempo aumenta. Isto se deve principalmente a perdas por divergência esférica (quando a frente de onda esférica avança, seu raio aumenta, diminuindo assim a quantidade de energia por unidade de superfície), perdas por absorção de energia (enquanto a onda se propaga pela terra, parte da sua energia é transformada em calor), perdas de transmissão (a energia da onda vai sendo reduzida pelas parcelas refletidas nas interfaces) e perdas por espalhamento (irregularidades e heterogeneidades no meio de propagação espalham a energia em direções aleatórias).

Há duas formas de correção: independente dos dados (RMS) e dependente dos dados (AGC). A análise de cada caso pelo geofísico de processamento vai indicar a melhor forma de correção a ser aplicada. As figuras 6 e 7 mostram um sismograma antes e após a aplicação de um ganho tipo AGC. Observe que antes do ganho as amplitudes nos tempos superiores a 0.7 s estão muito atenuadas e após a aplicação do ganho houve uma recuperação destas amplitudes.

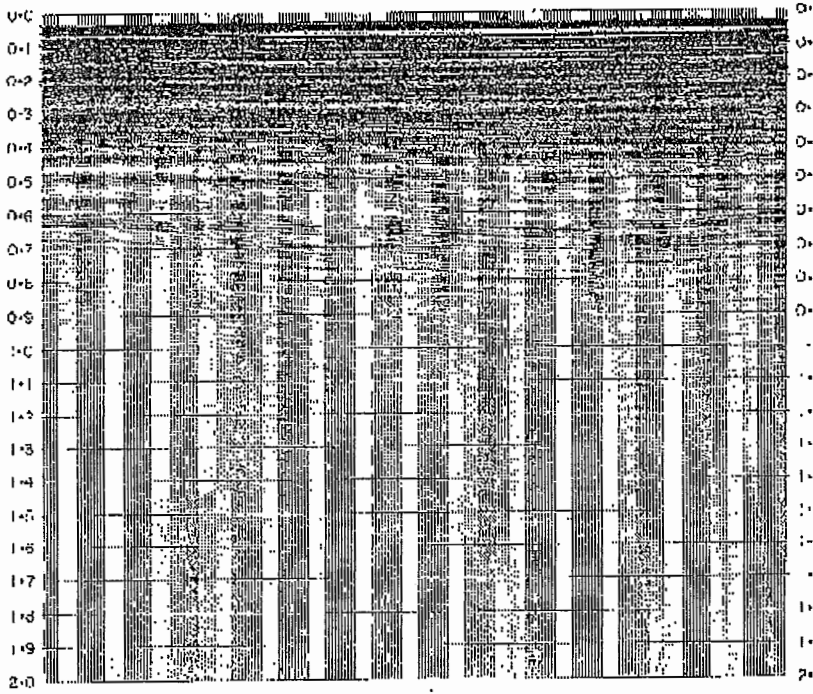


Figura 6. Sismograma antes do ganho AGC: (Adaptação de [Hatton86]).

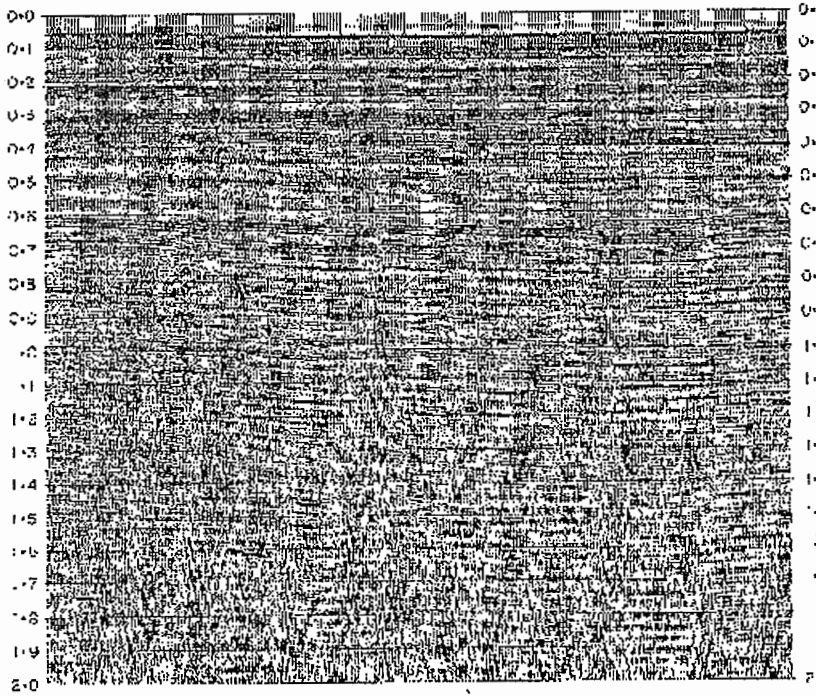


Figura 7. Sismograma após ganho AGC: (Adaptação de [Hatton86]).

### 2.2.3- Correção dinâmica

A correção dinâmica é também conhecida como *NMO*<sup>3</sup>. Na técnica CMP há uma redundância de dados pela múltipla amostragem de um mesmo ponto em subsuperfície. Para que os diversos traços relativos a um certo CMP possam ser empilhados, é necessária a aplicação de uma correção dinâmica (dependente do tempo) aos traços.

Esta correção é devida ao atraso com que as reflexões atingem os receptores de maior offset, como mostra a figura 5. A figura 8 apresenta um conjunto de CMPs antes da aplicação da correção de NMO e a figura 9 apresenta o mesmo conjunto de CMPs após esta correção. Observe que as reflexões que se apresentavam como curvas hiperbólicas antes da correção aparecem alinhadas após a aplicação do NMO.

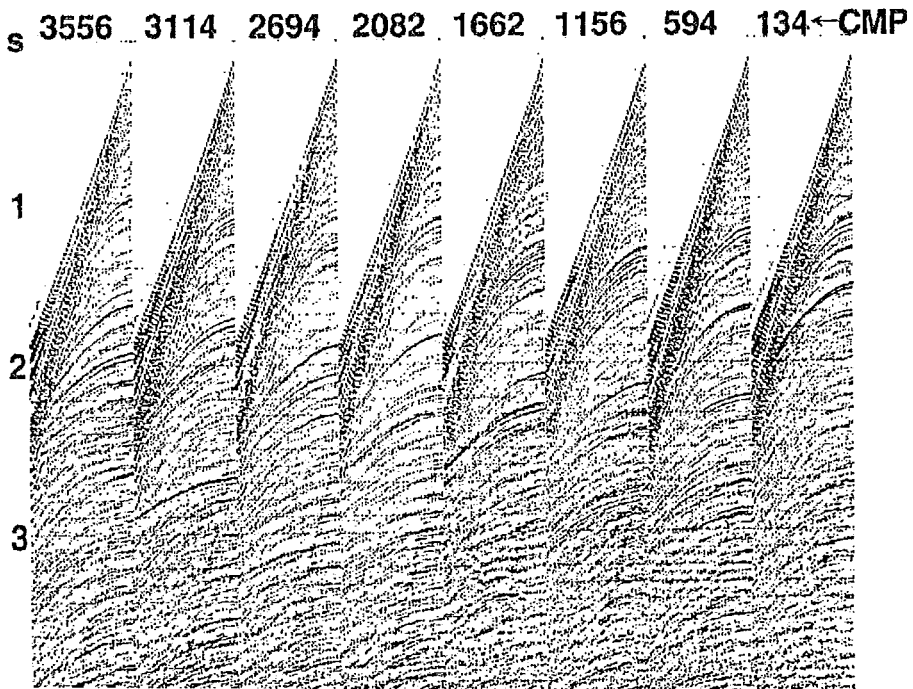


Figura 8. Sismograma antes da correção dinâmica: (Adaptação de [Yilmaz87]).

<sup>3</sup>Normal Move Out

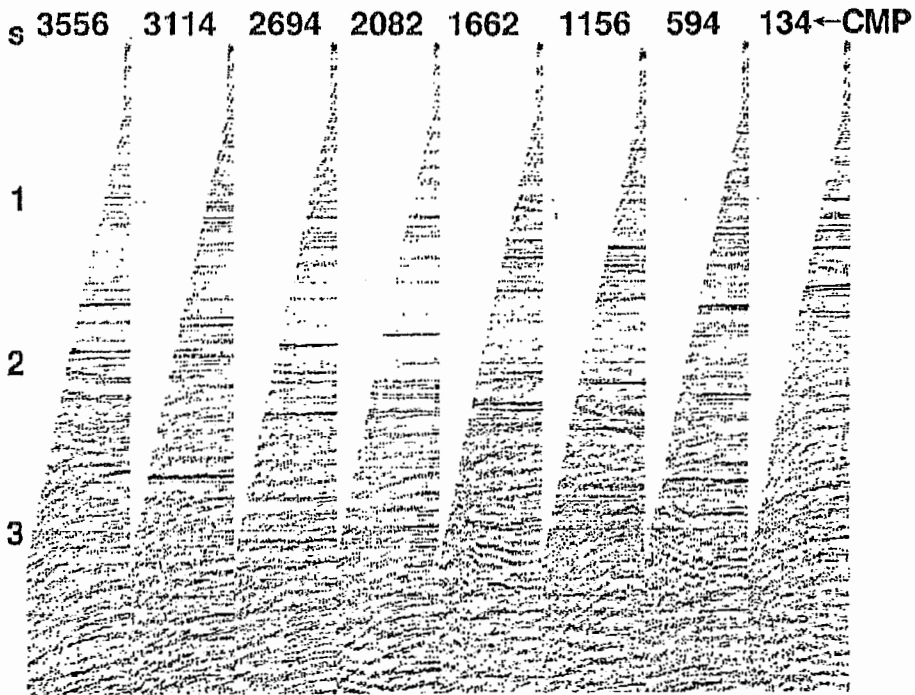


Figura 9. Sismograma após a correção dinâmica: (Adaptação de [Yilmaz87]).

#### 2.2.4- Análise de Velocidades

Para que possa ser aplicada a Correção Dinâmica, é necessário que se tenha informações sobre as velocidades de propagação nos diversos meios em subsuperfície. Como já foi comentado na seção 2.1, a multicobertura gerada pela técnica CMP permite estimativas das velocidades em subsuperfície.

A análise de velocidades consiste em apresentar medidas de coerência de sinal ao longo das hipérbolas de reflexão, em CMPs previamente escolhidos, permitindo ao geofísico de processamento a escolha das velocidades mais convenientes em cada CMP e a cada profundidade.

#### 2.2.5- Correção Estática

Os tempos de reflexão são freqüentemente afetados por irregularidades próximas à superfície. Para corrigir estes efeitos, especialmente o da variação da elevação na superfície, é feita uma correção no tempo dos traços, como se eles tivessem sido adquiridos não na superfície mas em uma linha, em geral plana, abaixo da superfície, denominada *datum*.



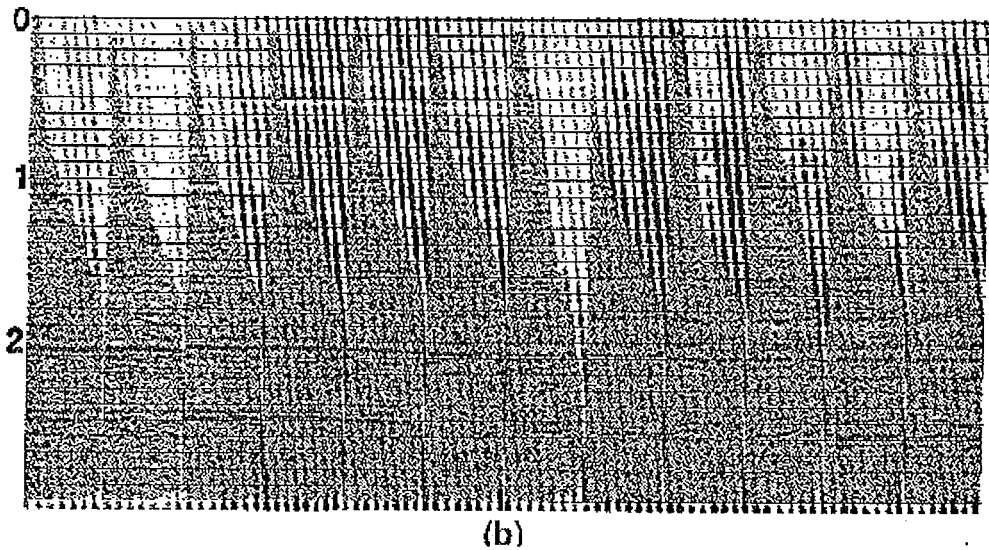
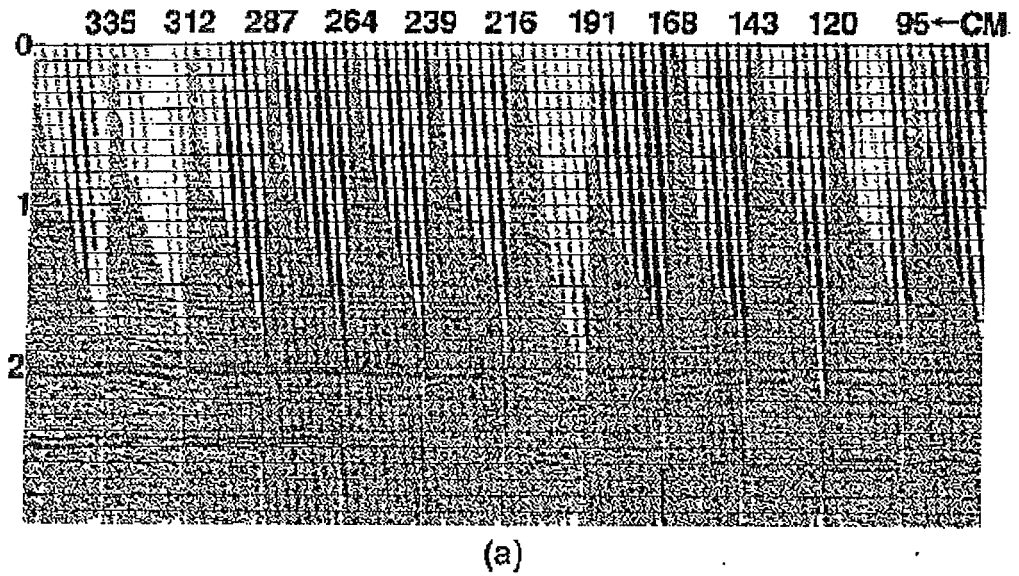


Figura 10. Efeito da correção estática: Sismograma antes (a) e após (b) a correção estática (Adaptação de [Yilmaz87]).

A figura 10 mostra um sismograma antes e após a aplicação da correção estática. Observe que o refletor situado no tempo de 2 s, na posição do CMP 216 alinhou-se após a correção estática.

## 2.2.6- Empilhamento

Uma vez que as correções dinâmicas e estáticas tenham sido aplicadas, os dados estão prontos para o empilhamento. Assim todos os traços relativos a um mesmo CMP são somados amostra a amostra, reforçando os sinais da reflexão que são coerentes e ao mesmo tempo cancelando os ruídos aleatórios. A saída deste processo simula uma seção sísmica de offset zero, onde o traço resultante está posicionado no CMP.

## 2.2.7- Migração Sísmica

Uma seção empilhada não garante que as estruturas mostradas estejam na sua posição correta em subsuperfície. A migração é o processo sísmico usado para posicionar corretamente os eventos em subsuperfície. Como é o objeto de análise desta tese, está descrito detalhadamente no próximo capítulo. A figura 11 mostra uma seção empilhada antes e após a aplicação da migração sísmica. Observe que a migração moveu o evento B para a sua posição dada como correta em subsuperfície A e colapsou a difração D para o seu ápice P.

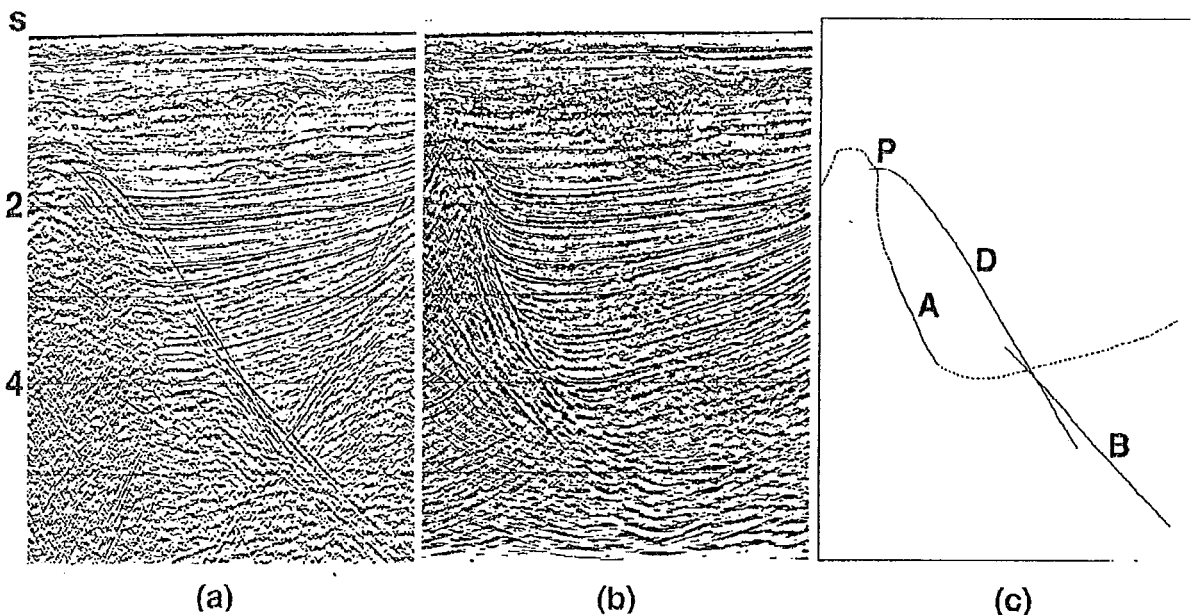


Figura 11. Efeito da migração sísmica: Seção sísmica antes (a) e após (b) a migração sísmica. Em (c) vê-se o desenho esquemático de uma difração proeminente D migrada para a posição P e um evento mergulhante antes (B) e após (A) a migração. (Adaptação de [Yilmaz87]).

## Capítulo 3 - A Migração Sísmica

Neste capítulo é feita uma descrição simples da Migração Sísmica mostrando a sua importância na prospecção. Após isto são apresentados uma formulação matemática e dois métodos de solução, representando os métodos Phase-Shift e  $\omega$ - $x$ . Finalmente é mostrado um algoritmo para o método  $\omega$ - $x$ .

### *3.1 - O que é a Migração*

O que está disponível para o geofísico antes da migração é uma seção sísmica em tempo, representando a imagem da subsuperfície obtida na superfície, através da Aquisição Sísmica.

Como já foi mostrado no capítulo anterior, os eventos representados na seção empilhada (aproximação da seção de offset zero), se apresentam mergulho, não estão nas posições corretas em subsuperfície.

A Migração Sísmica é o processo que se propõe a transformar a imagem recebida na superfície na imagem real do que existe em subsuperfície. A Migração, além disso, também corrige as distorções introduzidas pelo tratamento simplificado (NMO-empilhamento) dos dados.

### *3.2 - O Impacto da Migração*

Para entender a migração 2-D, vai-se analisar uma estrutura geológica extremamente simples: um único ponto difrator num meio homogêneo.

Considere um sistema cartesiano de coordenadas  $(x,z)$ , onde  $x$  acompanha a linha sísmica na superfície e  $z$  é o eixo da profundidade. Suponha, como na figura 12, que a fonte e o receptor estejam no mesmo ponto  $P$  da superfície (offset zero),

que  $B$  represente o ponto difrator e que o tempo de trânsito do sinal originado em  $P$ , refletido em  $B$  e retornado a  $P$  seja  $t$ . Suponha, ainda, que a velocidade de propagação no meio seja  $v$ , constante.

A frente de onda gerada pela fonte sísmica em  $P$ , no espaço tri-dimensional  $(x,y,z)$  é esférica. Assim, no plano  $(x,z)$  em que se está trabalhando, esta frente de onda será um círculo.

Como a reflexão será desenhada verticalmente abaixo da posição comum da fonte-receptor, o ponto  $B$  aparecerá no sismograma na posição  $A$ , como mostra a figura 12.

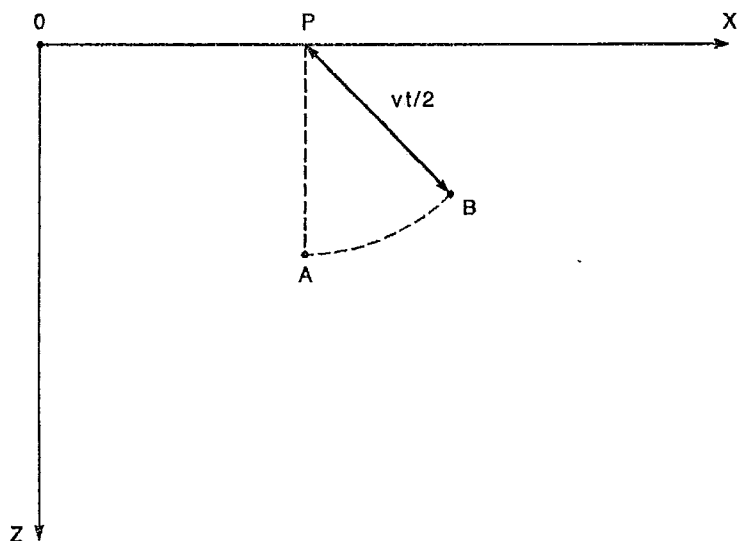


Figura 12. Frente de onda no espaço  $(x,z)$ : Esta frente de onda com origem em  $P$ , encontra o ponto difrator  $B$  na distância  $\frac{vt}{2}$ . (Adaptação de [Hatton86]).

Um outro modo de analisar a propagação da frente de onda é uma abstração conhecida como "Modelo do Refletor Explosivo" [Loewenthal76]. A idéia deste modelo vem da observação de que o tempo que a frente de onda leva entre a fonte e o refletor é igual ao tempo que esta mesma frente leva entre o refletor e o receptor. Assim, neste modelo, supõe-se que quem explode é o refletor e a frente de onda gerada é gravada pelos receptores na superfície no tempo  $t/2$ .

Como os dados estão sendo recebidos na superfície, necessita-se de uma dimensão que relacione-se com a profundidade  $z$ , para que se possa estimá-la. Esta dimensão é o tempo, representado pelo eixo  $t$ . Assim, como a frente de onda

esférica no espaço  $(x,y,z)$  se expande com o tempo, os círculos que a representam no espaço  $(x,z)$  serão círculos de raios crescentes com o tempo, o que faz com que esta frente de onda no novo espaço de estudo  $(x,z,t)$  seja um cone.

A relação entre os eixos  $t$  e  $z$  é dada pela equação do movimento uniforme  $z = vt$ . Observe também que pelo Modelo do Refletor Explosivo, a cada tempo  $t$  o plano  $(x,z)$  contém a frente de onda após  $t$  segundos.

Deste modo, a frente de onda cônica no espaço  $(x,z,t)$  ao interceptar o plano  $z = 0$  forma uma hipérbole de difração idêntica à gerada no caso real quando a explosão e a recepção se dão na superfície. Um exemplo análogo ao da figura 12 é mostrado na figura 13 para o modelo do refletor explosivo (compare o caráter hiperbólico da reflexão com a figura 5).

Note ainda que se no modelo real a frente de onda gastava  $t$  segundos para ir de  $P$  a  $B$  e retornar a  $P$ , agora o tempo de percurso reduziu-se a  $t/2$  já que neste modelo a explosão parte de  $B$ .

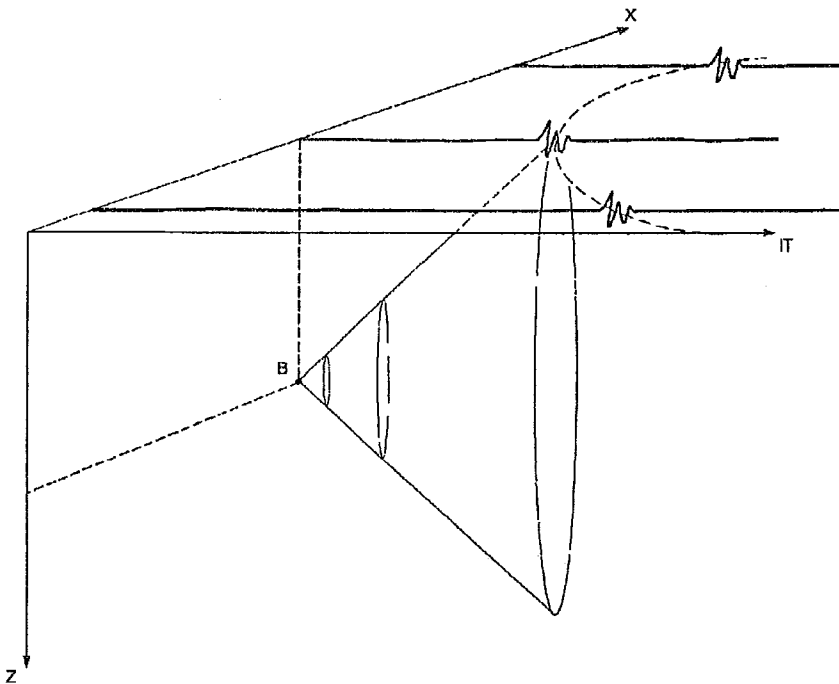


Figura 13. Campo de onda no modelo do refletor explosivo: Campo no espaço  $(x,z,t)$  para um ponto difrator  $B$  num meio homogêneo. (Adaptação de [Hatton86]).

Supondo que o refletor seja uma interface, ele deve ser tangente à frente de onda no ponto  $B$ . O mergulho aparente na seção em tempo deve ser tangente à

curva de difração no ponto  $A$ . A figura 14 é uma composição das figuras 12 e 13, onde a profundidade  $z$  e o tempo escalonado  $\frac{vt}{2}$  estão representados no mesmo eixo. Note também que  $PA = PB = \frac{vt}{2}$ .

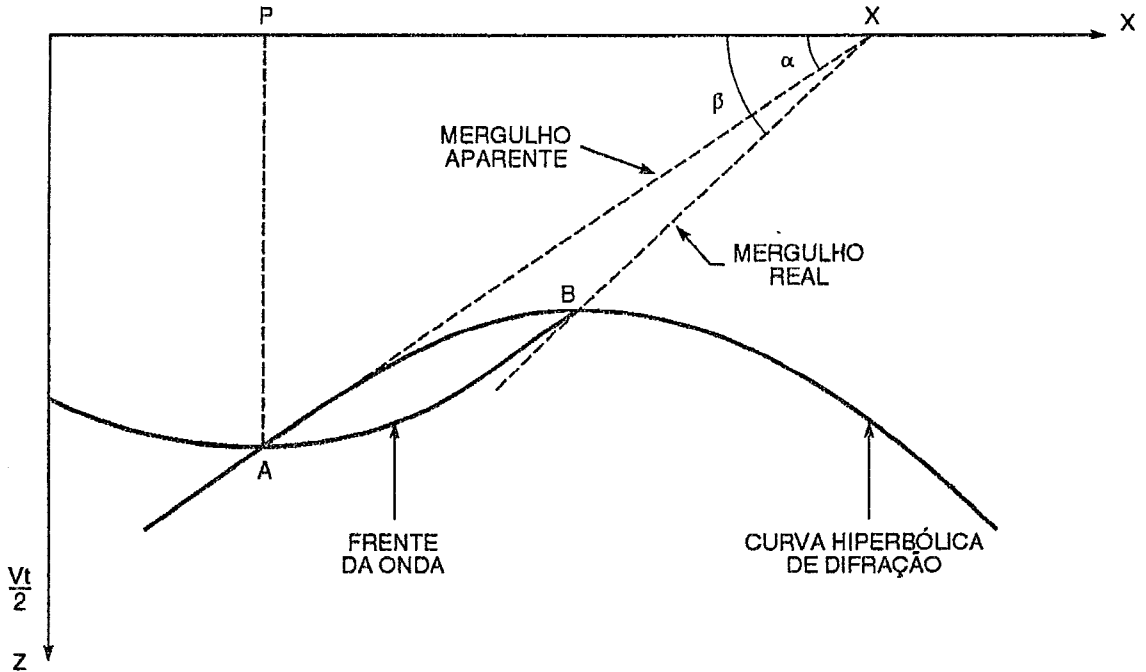


Figura 14. Relação entre mergulhos: Relação entre o mergulho aparente e real de um refletor. (Adaptação de [Hatton86]).

Pela figura 14:

$$\sin \beta = \frac{PB}{PX}$$

$$\tan \alpha = \frac{PA}{PX}$$

Como  $PA = PB$ ,

$$\sin \beta = \tan \alpha$$

Esta relação entre o mergulho real e o mergulho aparente é conhecida como "Equação da Migração". Para compreender de forma intuitiva o processo da

migração vai-se analisar geometricamente a relação entre uma seção geológica (posição real dos refletores em subsuperfície) e uma seção em tempo obtida a partir de dois pares fonte-receptor (offset zero) colocados nas posições  $A$  e  $B$ , como mostra a figura 15. Considere  $v/2 = 1$  de tal forma que os eixos  $z$  e  $t$  sejam diretamente intercambiáveis.

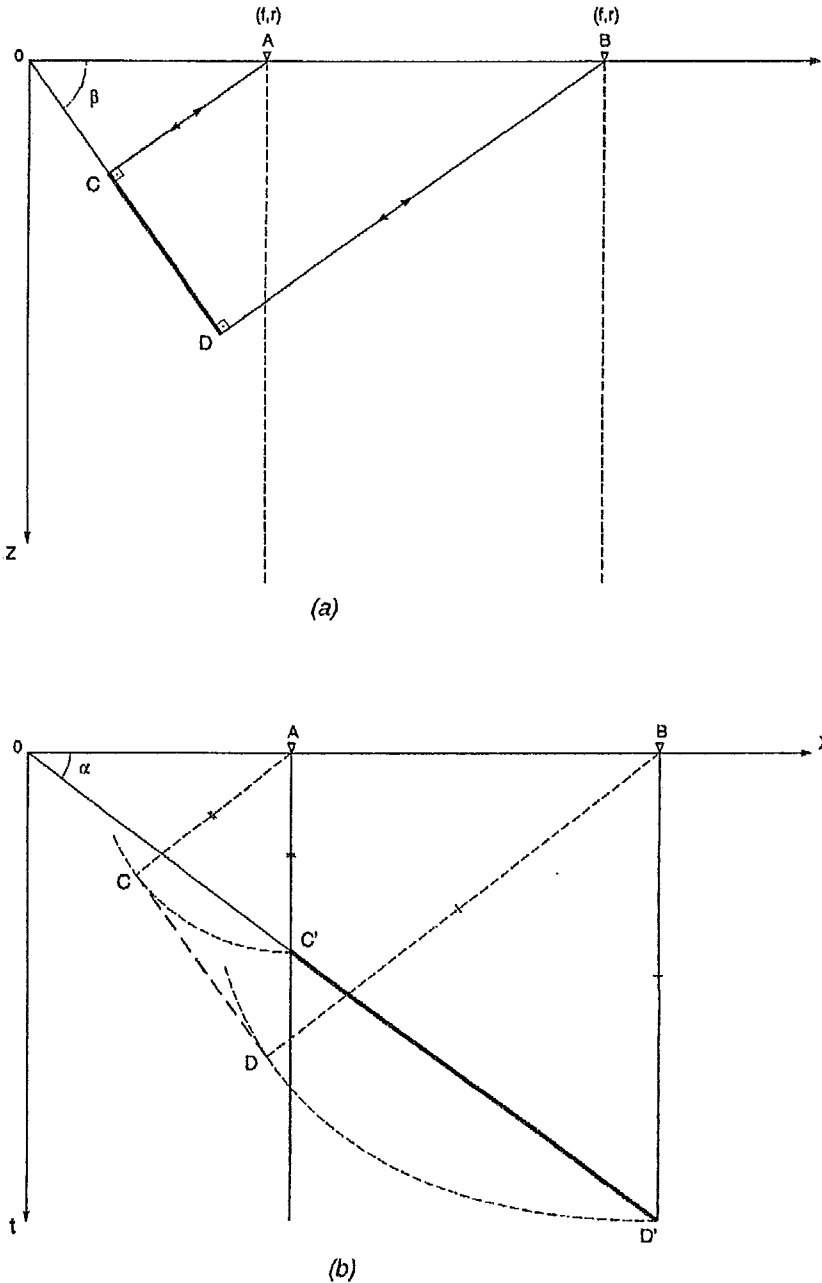


Figura 15. Princípios de Migração: O segmento de reflexão  $C'D'$  na seção em tempo (b), quando migrado é movido mergulho acima, mais inclinado e encurtado ao ser colocado na sua real posição em subsuperfície  $CD$  (a). (Adaptação de [Chun81]).

Note que a seção geológica deve ser o resultado da migração da seção em tempo, já que por definição a migração coloca os refletores nas suas posições corretas em subsuperfície.

Da geometria das seções mostradas na figura 15, pode-se fazer as seguintes observações sobre a migração:

1. O ângulo de mergulho do refletor na seção geológica é maior do que na seção empilhada; logo, a migração aumenta o ângulo de mergulho dos refletores.
2. O tamanho do refletor na seção geológica é menor do que na seção empilhada; logo, a migração encurta os refletores.
3. A migração move os refletores mergulho acima.

Então, já que a migração altera a posição dos refletores vai-se procurar ter um avaliação quantitativa desta variação de posição. Para isto, vai-se calcular o deslocamento de posição de um ponto  $P$  de uma seção empilhada em profundidade para a posição  $\bar{P}$  migrada, também em profundidade. Considere que  $P$  está no tempo  $t$  na seção em tempo correspondente à seção em profundidade mostrada na figura 16.

Sejam  $h = \overline{B\bar{P}}$  e  $z = BP$ . Pelo próprio conceito de migração (veja fig. 16)  $BP = B\bar{P}$ . Sejam  $d_x$  e  $d_z$  os deslocamentos causados pela migração em superfície e em profundidade, respectivamente. Então:

$$h = B\bar{P} \sin(90 - \beta)$$

$$h = z \cos \beta$$

$$d_z = z - h$$

$$d_z = z(1 - \cos \beta).$$

Mas,

$$z = Vt$$



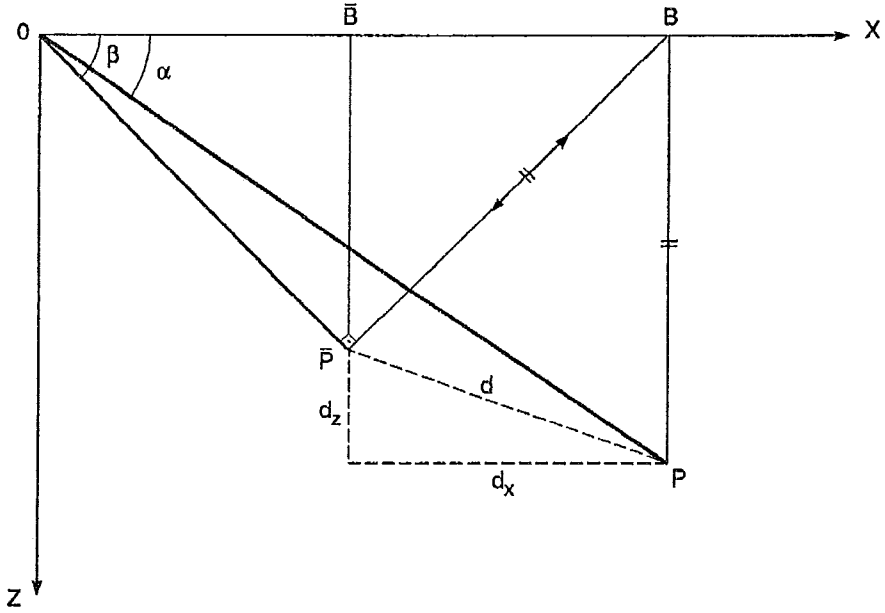


Figura 16. Variação de posição com a migração: Variação de posição de um ponto  $P$  da seção empilhada em profundidade para  $\bar{P}$  na seção migrada em profundidade. (Adaptação de [Chun81]).

logo

$$d_z = Vt(1 - \cos \beta).$$

Ainda pela figura 16:

$$d_x = B\bar{P} \cos(90 - \beta)$$

$$d_x = z \sin \beta$$

$$d_x = Vt \sin \beta$$

Pelos resultados obtidos fica evidenciada a importância da velocidade no deslocamento dos eventos e por consequência na correta migração. Exatamente no cálculo destas velocidades reside o maior problema e fonte de erros no

processo de migração. Para mostrar dados reais deste deslocamento, reproduz-se a tabela a seguir:

| t(s) | v<br>(m/s) | dx<br>(m) | dt<br>(s) |
|------|------------|-----------|-----------|
| 1,00 | 2.500,00   | 250,00    | 0,13      |
| 2,00 | 3.000,00   | 720,00    | 0,40      |
| 3,00 | 3.500,00   | 1.470,00  | 0,86      |
| 4,00 | 4.000,00   | 2.560,00  | 1,60      |
| 5,00 | 4.500,00   | 4.060,00  | 2,82      |

Tabela 3.1 - Deslocamento horizontal e vertical de pontos em refletores mergulhantes em várias profundidades. (Adaptação de [Yilmaz87]).

Então, pela tabela 3.1, supondo que um horizonte de interesse a 4 s de profundidade contenha óleo e a perfuração do poço tenha sido realizada com base na seção não migrada, haveria um erro no local de perfuração de aproximadamente 2,5 km !!!

Em vista disto, não são precisos outros argumentos para justificar a necessidade e a importância da Migração no Processamento Sísmico.

### 3.3 - Métodos de Migração

Uma vez colocado intuitiva e geometricamente o problema da migração, apresenta-se a seguir sua formulação matemática e solução numérica.

#### 3.3.1 - Formulação Matemática

A propagação de ondas em um meio acústico, homogêneo (com densidade constante), bidimensional é descrito pela equação da onda:

$$\frac{1}{V^2} \frac{\partial^2 P}{\partial t^2} = \frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial z^2} \quad (1)$$

onde  $P(x,z,t)$  representa o campo de onda no espaço  $(x,z,t)$  e  $V$  é a velocidade de propagação no meio. No sistema de coordenadas cartesianas adotado,  $x$  (representa o deslocamento ao longo da linha sísmica na superfície),  $z$  (representa a profundidade em subsuperfície) e  $t$  (representa o tempo).

Seja  $f$  um candidato a solução de (1) na forma:

$$f_{\omega, k_x, k_z}(x, z, t) = e^{(-i\omega t + ik_x x + ik_z z)},$$

onde  $i^2 = -1$ ,  $\omega$  é a frequência temporal,  $k_x$  é a frequência espacial na direção  $x$  e  $k_z$  é a frequência espacial na direção  $z$ .

Substituindo  $f$  como  $P$  em (1), vem

$$\frac{-\omega^2}{V^2} f = -k_x^2 f - k_z^2 f$$

Logo,  $f$  é solução de (1) se:

$$\frac{\omega^2}{V^2} = k_x^2 + k_z^2$$

Tal relação é conhecida como *relação de dispersão*.

Como (1) é uma equação diferencial linear:

- i) Se  $f$  é solução e  $\alpha$  é um escalar não nulo, então  $\alpha f$  é solução.
- ii) Se  $g_i$  representa um conjunto de soluções então  $\sum g_i$  é solução.

Logo, pode-se dizer que:

$$P(x, z, t) = \sum_{\omega, k_x, k_z} \alpha_{\omega, k_x, k_z} e^{(-i\omega t + ik_x x + ik_z z)}$$

é solução de (1) se

$$k_x^2 + k_z^2 = \frac{\omega^2}{V^2}$$

A migração de dados empilhados pode ser definida da seguinte forma: dado um campo de onda registrado na superfície  $P(x, z = 0, t)$ , obter  $P(x, z, t = 0)$  conhecendo  $V = \text{constante}$ , onde

$$P(x, z, t) = \sum_{\omega, k_x, k_z} \alpha_{\omega, k_x, k_z} e^{(-i\omega t + ik_x x + ik_z z)}$$

é solução se

$$k_x^2 + k_z^2 = \frac{\omega^2}{V^2}$$

Serão apresentadas duas soluções desta equação, cada uma correspondendo a um método de migração sísmica.

### 3.3.2 - Migração Phase-Shift

O método Phase-Shift foi escolhido para ser apresentado pela sua simplicidade.

Explicitando o somatório em  $k_z$ :

$$P(x, z, t) = \sum_{\omega, k_x} \left( \sum_{k_z} \alpha_{\omega, k_x, k_z} e^{ik_z z} \right) e^{(-i\omega t + ik_x x)}$$

Definindo:

$$B_{\omega, k_x}(z) = \sum_{k_z} \alpha_{\omega, k_x, k_z} e^{ik_z z}$$

então,

$$P(x,z,t) = \sum_{\omega, k_x} B_{\omega, k_x}(z) e^{(-i\omega t + ik_x x)}$$

A equação (1), como equação da onda, contempla tanto as ondas que sobem quanto as ondas que descem. Como está sendo usado o modelo do refletor explosivo, só há interesse nas ondas que sobem do refletor para a superfície. O que permite a separação destas duas soluções (ascendente e descendente) é a escolha do  $k_z$  adequado.

Assim, dados  $k_x$ ,  $\omega$  e  $V$ , existe no máximo um  $k_z$  real que satisfaz à relação de dispersão e representa as ondas ascendentes. Seja  $k_z = f(\omega, k_x, V)$  este valor.

Logo, o somatório

$$B_{\omega, k_x}(z) = \sum_{k_z} \alpha_{\omega, k_x, k_z} e^{ik_z z}$$

se reduz a no máximo um termo:

$$B_{\omega, k_x}(z) = \alpha_{\omega, k_x, f(\omega, k_x, V)} e^{if(\omega, k_x, V)z}$$

Admitindo  $V$  constante, pode-se escrever:

$$B_{\omega, k_x}(z + \Delta z) = \alpha_{\omega, k_x, f(\omega, k_x, V)} e^{if(\omega, k_x, V)(z + \Delta z)}$$

$$B_{\omega, k_x}(z + \Delta z) = \alpha_{\omega, k_x, f(\omega, k_x, V)} e^{if(\omega, k_x, V)z} e^{if(\omega, k_x, V)\Delta z}$$

Então, identificando  $B_{\omega, k_x}(z)$  na expressão anterior, vem:

$$B_{\omega, k_x}(z + \Delta z) = B_{\omega, k_x}(z) e^{if(\omega, k_x, V)\Delta z} \quad (2)$$

com  $k_z = f(\omega, k_x, V)$  dado pela relação de dispersão.

## Algoritmo Phase-Shift

Como

$$P(x,z,t) = \sum_{\omega, k_x} B_{\omega, k_x}(z) e^{(-i\omega t + ik_x x)}, \quad (3)$$

então,

$$P(x,0,t) = \sum_{\omega, k_x} B_{\omega, k_x}(0) e^{(-i\omega t + ik_x x)}.$$

Esta expressão coincide com a expansão de  $P(x,0,t)$  em série de Fourier. Conseqüentemente,  $B_{\omega, k_x}$  são os coeficientes da transformada de Fourier, em duas dimensões, de  $P(x,0,t)$ . Desta forma,  $B_{\omega, k_x}$  pode ser obtido pelo algoritmo clássico de FFT 2-D aplicado à  $P(x,0,t)$  e a partir daí, aplicando (2) sucessivamente, tem-se:

$$B_{\omega, k_x}(\Delta z), B_{\omega, k_x}(2\Delta z), B_{\omega, k_x}(3\Delta z), \dots, B_{\omega, k_x}(z)$$

finalmente, a solução desejada é:

$$P(x,z,t=0) = \sum_{\omega, k_x} B_{\omega, k_x}(z) e^{ik_x x}$$

O método Phase-Shift apesar da sua simplicidade não se configura como um bom método por não permitir variação horizontal de velocidade. Isto serve como motivação para o próximo método que, apesar de partir também da premissa de velocidade constante, admite variação horizontal de velocidade.

### 3.3.3 - Migração $\omega$ - $x$

Pela formulação matemática já apresentada, resolvendo a equação da onda (1), explicitando  $k_z$  na relação de dispersão e escolhendo ondas ascendentes ( $k_z$  negativo):

$$k_z = -\frac{\omega}{V} \sqrt{1 - \frac{V^2}{\omega^2} k_x^2} \quad (4)$$

Derivando P, vem:

$$\frac{\partial P}{\partial z} = ik_z P \quad (5)$$

Usando (4) em (5):

$$\frac{\partial P}{\partial z} = -i \frac{\omega}{V} \sqrt{1 - \frac{V^2}{\omega^2} k_x^2} P$$

Para passar ao domínio do espaço, não basta substituir  $k_x^2$  pela expressão da segunda derivada em  $x$  porque surge uma raiz quadrada de uma derivada parcial. Para contornar este problema usa-se a expansão de (4) por frações contínuas [Young72]:

Definindo:

$$X^2 = \frac{V^2}{\omega^2} k_x^2$$

vem:

$$k_z = -\frac{\omega}{V} \sqrt{1 - X^2}$$

Definindo:

$$R = \sqrt{1 - X^2}$$

A recursão de ordem  $n + 1$  para  $R$  é dada por:

$$R_{n+1} = 1 - \frac{X^2}{1 + R_n}$$

com

$$R_0 = 1.$$

Tomando a aproximação  $R_1 = 1 - \frac{X^2}{2}$ , vem:

$$k_z = -\frac{\omega}{V} \left( 1 - \frac{V^2 k_x^2}{2\omega^2} \right)$$

$$k_z = -\frac{\omega}{V} + \frac{V}{2\omega} k_x^2$$

Re-escrevendo (5):

$$\frac{\partial P}{\partial z} = i \left( -\frac{\omega}{V} + \frac{V}{2\omega} k_x^2 \right) P$$

$$\frac{\partial P}{\partial z} = -i \frac{\omega}{V} P + i \frac{V}{2\omega} k_x^2 P$$

Mas:

$$\frac{\partial^2 P}{\partial x^2} = -k_x^2 P$$

Logo:

$$\frac{\partial P}{\partial z} = -i \frac{\omega}{V} P - i \frac{V}{2\omega} \frac{\partial^2 P}{\partial x^2} \quad (6)$$

Esta é a equação a ser resolvida nesta nova formulação do problema.

Seja

$$P(x, z, t) = \sum_{\omega} Q(x, z, \omega) e^{(-i\omega t - i \frac{\omega}{V} z)} \quad (7)$$

uma possível de solução para (6). Então, substituindo (7) em (6), obtém-se para cada  $\omega$ :

$$\frac{\partial Q}{\partial z} e^{(-i\omega t - i \frac{\omega}{V} z)} - i \frac{\omega}{V} P = -i \frac{\omega}{V} P - i \frac{V}{2\omega} \frac{\partial^2 Q}{\partial x^2} e^{(-i\omega t - i \frac{\omega}{V} z)}$$

ou seja,



$$\frac{\partial Q}{\partial z} = \frac{V}{2i\omega} \frac{\partial^2 Q}{\partial x^2} \quad (8)$$

Então, para que (7) seja solução de (6), é preciso que (8) seja verificada.

### Algoritmo $\omega$ - $x$

O algoritmo baseia-se na extrapolação em profundidade a partir da seção de entrada. O passo básico do algoritmo consiste em extrapolar a função  $Q(x, z, \omega)$  para uma próxima profundidade resultando em  $Q(x, z + \Delta z, \omega)$ , que multiplicada por uma exponencial e somada em  $\omega$  como em (7), resulta em  $P(x, z + \Delta z, t = 0)$ .

Para aplicar o algoritmo: a partir da seção de entrada que é  $P(x, z = 0, t)$ , aplica-se a FFT em uma dimensão a todos os traços, passando para o domínio da freqüência, obtendo-se  $Q(x, 0, \omega)$ , a partir daí:

$$\begin{aligned} Q(x, 0, \omega) &\xrightarrow{\text{passo básico}} Q(x, \Delta z, \omega) \Rightarrow P(x, \Delta z, 0) \\ Q(x, \Delta z, \omega) &\xrightarrow{\text{passo básico}} Q(x, 2\Delta z, \omega) \Rightarrow P(x, 2\Delta z, 0) \\ Q(x, 3\Delta z, \omega) &\xrightarrow{\text{passo básico}} Q(x, 3\Delta z, \omega) \Rightarrow P(x, 3\Delta z, 0) \end{aligned}$$

e assim sucessivamente até o final da seção.

A resolução de (8) é feita pelo método de Crank-Nicolson [Carnahan69] de diferenças finitas. Para a aplicação de diferenças finitas, é preciso discretizar o espaço de trabalho. Seja a discretização do eixo  $x$  onde a faixa de interesse é de  $X_{\min}$  a  $X_{\max}$ , e o número de pontos de discretização,  $n_x$ , então (usando letras maiúsculas para o contínuo e minúsculas para o discreto):

$$\Delta x = \frac{X_{\max} - X_{\min}}{n_x - 1}$$

$$x_i = (i - 1)\Delta x + x_1 \quad i = 1, \dots, n_x$$

$$x_1 = X_{\min}$$

$$x_{n_x} = X_{\max}$$

Os eixos  $z$  e  $\omega$  são discretizados analogamente. A função  $Q$  para um certo  $\omega$  é discretizada pela matriz  $q_z^x$  que, por abuso de notação, será representada por  $q_z^x$ . Desta forma,  $q_z^{x \pm 1}$  corresponde à  $q_{z \pm 1}^x$  que é  $Q_w(x + \Delta x, z - \Delta z)$ .

Aproximando as derivadas por:

$$\frac{\partial Q}{\partial z} = \frac{q_z^x - q_z^{x-1}}{\Delta z}$$

$$\frac{\partial^2 Q}{\partial z^2} = \frac{1}{2} \left( \frac{q_z^{x+1} - 2q_z^x + q_z^{x-1}}{(\Delta x)^2} \right) + \frac{1}{2} \left( \frac{q_{z+1}^{x+1} - 2q_{z+1}^x + q_{z+1}^{x-1}}{(\Delta x)^2} \right)$$

Substituindo em (8), vem:

$$\frac{q_{z+1}^x - q_z^x}{\Delta z} = \frac{V}{2i\omega} \left( \frac{q_z^{x+1} - 2q_z^x + q_z^{x-1}}{2(\Delta x)^2} + \frac{q_{z+1}^{x+1} - 2q_{z+1}^x + q_{z+1}^{x-1}}{2(\Delta x)^2} \right) \quad (9)$$

Fazendo:

$$\alpha = \frac{V\Delta z}{4i\omega(\Delta x)^2}$$

e substituindo em (9):

$$q_{z+1}^x - q_z^x = \alpha \left[ (q_z^{x+1} - 2q_z^x + q_z^{x-1}) + (q_{z+1}^{x+1} - 2q_{z+1}^x + q_{z+1}^{x-1}) \right]$$

Agrupando as incógnitas no lado esquerdo:

$$-\alpha q_{z+1}^{x+1} + (1 + 2\alpha)q_{z+1}^x - \alpha q_{z+1}^{x-1} = \alpha q_z^{x+1} + (1 - 2\alpha)q_z^x + \alpha q_z^{x-1}$$

Observe-se que ao resolver a equação (8), está-se achando  $Q$  e não  $P$  que é a solução de (6) desejada. Para achar  $P$  é preciso multiplicar  $Q$  por  $e^{(-i\omega t - i\frac{\omega}{V}z)}$ .

Como na solução  $t = 0$ , basta multiplicar por  $e^{-i\frac{\omega}{V}z}$ , o que é conhecido por *deslocamento de fase*<sup>4</sup>.

Este produto poderia ser feito uma única vez no final, já que  $V$  é constante. Entretanto, a cada passo da solução está sendo feito o produto por  $e^{-i\frac{\omega}{V}\Delta z}$ , o que acomoda variações de velocidade, contrariamente ao estabelecido para a equação diferencial (1) onde  $V$  é constante.

Como a entrada para o problema é uma seção em tempo, é conveniente que a saída também o seja, para isto faz-se uma mudança de variável:

$$\tau = \frac{z}{V}$$

$$\frac{\partial \tau}{\partial z} = \frac{1}{V}$$

$$\frac{\partial P}{\partial z} = \frac{\partial \tau}{\partial z} \frac{\partial P}{\partial \tau} = \frac{1}{V} \frac{\partial P}{\partial \tau}$$

Aplicando em (6):

$$\frac{\partial P}{\partial \tau} = -i\omega P - \frac{iV^2}{2\omega} \frac{\partial^2 P}{\partial x^2}$$

e, novamente, aplicando em (8):

$$\frac{\partial Q}{\partial \tau} = \frac{V^2}{2i\omega} \frac{\partial^2 Q}{\partial x^2}$$

Então:

$$\alpha = \frac{V^2 \Delta \tau}{4i\omega \Delta z}$$

A implementação do algoritmo  $\omega$ - $x$  utilizada segue estritamente a sugerida por Claerbout na página 111 do seu livro [Claerbout85]. O algoritmo inicializa algumas variáveis e converte a seção sísmica de entrada do domínio do tempo para o domínio da frequência, em cujo domínio será feita a extrapolação em profundidade. A conversão é feita através da aplicação de FFTs unidimensionais,

---

<sup>4</sup>*phase – shift*

uma para cada traço, obtendo uma matriz de  $n_x$  traços por  $n_w$  amostras. Esta matriz é denominada *seção em frequência*.

Inicialmente, a seção em frequência retrata os dados coletados na superfície. O algoritmo prossegue extrapolando estes dados para profundidades progressivamente crescentes, até a profundidade máxima. Este processo de extrapolação implementa a solução das duas equações diferenciais, (8) e (6), na forma proposta por Claerbout. A primeira equação, (8), é resolvida pelo método de Crank-Nicolson. Isto requer a solução de um sistema tridiagonal de equações lineares para cada frequência e cada profundidade. A segunda equação é resolvida multiplicando-se o resultado da primeira por uma exponencial complexa e fazendo a acumulação em frequência, como em (7). O cálculo necessário para a solução da primeira equação será doravante denominado *solução da tridiagonal*, enquanto a solução da segunda equação será denominada *deslocamento de fase*.

A extrapolação é recursiva, pois a seção em frequência resultante do deslocamento de fase é utilizada na montagem do sistema tridiagonal para a próxima profundidade. Em seguida, da seção em frequência deve-se extrair os eventos em subsuperfície. Para tanto, é necessário o conhecimento de dois fatos.

O primeiro é que, conceitualmente, a seção em frequência para uma dada profundidade representa os dados amostrados naquela profundidade, ou seja, o processo de extrapolação move os receptores para a profundidade desejada. O segundo é que, em uma seção em tempo, as amostras em  $t = 0$  representam as interfaces da subsuperfície na profundidade que a seção representa.

Como consequência destes dois fatos, para obter os eventos basta converter a seção em frequência em uma seção no tempo, para em seguida obter, no tempo nulo, a representação das interfaces naquela profundidade.

Entretanto, não é necessário converter toda a seção da frequência para o tempo, em cada profundidade. Basta selecionar as amostras da seção em frequência que comporão as amostras de tempo nulo na seção em tempo e efetuar a transformação. Matematicamente, trata-se de realizar a FFT inversa para  $t = 0$ , o que resulta em uma simples soma das amostras em frequência. Desta forma, a matriz de saída é construída passo a passo, pela acumulação das componentes das frequências responsáveis pela imagem em  $t = 0$ .

Uma descrição sucinta da estrutura do algoritmo acima discutido é fornecida a seguir.

Inicialização;

Aplique FFT para cada traço;

**Para  $i_z = 1$  até  $n_z$  faça** (Laço em profundidade)

**Para  $i_\omega = 2$  até  $n_\omega$  faça** (Laço em freqüência)

Gere o Sistema Tridiagonal

Resolva o Sistema Tridiagonal

Aplique Deslocamento de Fase

Acumule Resultados

**fim faça** (Fim do laço em freqüência)

**fim faça** (Fim do laço em profundidade)

A inicialização anula a matriz de resultados, obtém os dados de entrada e os converte de reais para complexos. A FFT é aplicada sobre a seção de entrada. O sistema tridiagonal possui  $n_x$  equações e igual número de incógnitas. Armazenam-se apenas as três diagonais não nulas da matriz dos coeficientes. A solução do sistema utiliza o algoritmo da Eliminação de Gauss. O deslocamento de fase constitui-se na multiplicação das  $n_x$  amostras de cada freqüência por uma exponencial. A acumulação coleta, em cada ponto  $(i_x, i_z)$  da seção resultante a contribuição de cada freqüência  $i_\omega$ .

## Capítulo 4 - Paralelismo na Migração $\omega$ -X

Este capítulo, inicialmente, apresenta algumas características da implementação da Migração  $\omega$ -x. Em seguida analisa sua complexidade seqüencial, as fontes de paralelismo no algoritmo e como podem ser exploradas. Este processo conduz à seleção das duas formas de paralelismo que serão utilizadas no decorrer deste trabalho. O capítulo termina com a análise da complexidade das formas de paralelismo escolhidas.

### *4.1 - Alguns Fatos sobre a Implementação*

Como citado no capítulo anterior, a referência [Claerbout85] contém uma implementação completa desse algoritmo, descrita em RATFOR. Esta descrição foi transcrita para FORTRAN-77 padrão e executada, sem modificações, em um grande elenco de máquinas, desde computadores pessoais até máquinas da classe dos supercomputadores.

A implementação adota os dados de entrada sugeridos na referência. Trata-se de uma seção sintética gerada automaticamente pelo programa. Para verificar a correção dos resultados, foi adotado um esquema simples de soma das amostras em cada linha e em cada coluna da matriz de saída. Estes dados podem ser impressos, a critério do usuário. Esta é a única operação de entrada e saída existente no programa.

A tabela 4.1 contém os tempos de execução (CPU) desta implementação e de seus principais passos na estação IBM RS-6000/320H, em função do tamanho do problema. Os tempos de execução são expressos em centésimos de segundo.

|                          |       |         |         |         |
|--------------------------|-------|---------|---------|---------|
| <i>traços/amostras</i>   | 64/64 | 128/128 | 256/256 | 512/512 |
| <i>tempo total</i>       | 343   | 2757    | 21519   | 172046  |
| <i>inicialização</i>     | 1     | 3       | 10      | 40      |
| <i>fft</i>               | 7     | 32      | 132     | 635     |
| <i>gera sistema</i>      | 64    | 456     | 3539    | 29644   |
| <i>resolve sistema</i>   | 191   | 1663    | 12915   | 102707  |
| <i>deslocamento fase</i> | 47    | 324     | 2631    | 20894   |
| <i>acumulação</i>        | 33    | 279     | 2231    | 18127   |

Tabela 4.1 - Tempos de execução da Migração  $\omega$ -x na versão seqüencial na RS-6000/320H e suas distribuições entre os passos do algoritmo.

A tabela 4.2 apresenta os tempos da tabela 4.1 em percentuais do tempo total de execução, para cada tamanho do problema. Esta outra visão dos dados permite determinar facilmente os passos de maior custo computacional do programa.

|                          |        |         |         |         |
|--------------------------|--------|---------|---------|---------|
| <i>traços/amostras</i>   | 64/64  | 128/128 | 256/256 | 512/512 |
| <i>tempo total</i>       | 100,00 | 100,00  | 100,00  | 100,00  |
| <i>inicialização</i>     | 0,29   | 0,10    | 0,04    | 0,02    |
| <i>fft</i>               | 2,04   | 1,16    | 0,64    | 0,36    |
| <i>gera sistema</i>      | 18,65  | 16,53   | 16,69   | 17,23   |
| <i>resolve sistema</i>   | 55,68  | 60,31   | 60,01   | 59,69   |
| <i>deslocamento fase</i> | 13,70  | 11,75   | 12,22   | 12,14   |
| <i>acumulação</i>        | 9,62   | 10,11   | 10,36   | 10,53   |

Tabela 4.2 - Tempos de execução da tabela 4.1 expressos como percentuais do tempo total de execução para cada tamanho de problema.

As tabelas demonstram que o tempo de execução é completamente dominado pelo aninhamento dos dois laços centrais. A acumulação dos percentuais de execução sobre os passos internos aos laços centrais e para cada tamanho de

problema, resulta em 97,67%, 98,74%, 99,32% e 99,69% do tempo total. Embora o trecho dominante do aninhamento seja a solução do sistema tridiagonal (55,68%, 60,31%, 60,01% e 59,69%), os demais trechos do aninhamento têm importância similar (41,97%, 38,39%, 39,27% 39,90%).

Estes dados permitem concluir que o esforço da paralelização deve ser concentrado no aninhamento dos laços centrais. Entretanto, não basta paralelizar a solução dos sistemas lineares, visto o peso computacional dos demais passos do laço central.

## 4.2 - *Análise da Complexidade do Algoritmo*

Os resultados experimentais anteriores permitem concentrar a análise de complexidade do algoritmo apenas no aninhamento central.

O problema é definido em um espaço de três dimensões. A primeira dimensão, representada por  $x$ , acompanha a linha sísmica na superfície, identificando as posições de tiro e de coleta dos sinais. A segunda dimensão é a profundidade, denotada por  $z$ . A terceira dimensão, denotada por  $t$ , é o tempo de trânsito do sinal, do tiro ao receptor. Este espaço é discretizado uniformemente em cada dimensão, com  $n_x$  pontos na direção  $x$ ,  $n_z$  pontos na direção  $z$  e com  $n_t$  pontos na direção  $t$ . Como contém apenas duas dimensões espaciais (a terceira é temporal) o problema é conhecido como migração 2-D.

A extrapolação do campo de ondas é realizada no domínio da frequência. Logo, há um quarto eixo, o da frequência, representado por  $\omega$  e discretizado por meio de  $n_\omega$  pontos. Quando utilizado, este eixo substitui o eixo dos tempos.

O laço central é executado exatamente  $n_z(n_\omega - 1)$  vezes. Como  $n_z$  e  $n_\omega$  são parâmetros derivados de  $n_t$  ( $n_z = n_t$  e  $n_\omega = n_t/2$ ), conclui-se que o número de iterações do laço central é  $n_t(n_t/2 - 1)$ .

A complexidade de cada um dos passos do aninhamento é de  $O(n_x)$ . Conseqüentemente, a complexidade do aninhamento é  $O(n_t^2 n_x)$ . Logo, o tempo de execução cresce linearmente com o número de traços e quadraticamente com o número de amostras por traço. A tabela 4.3, a seguir, também obtida experimentalmente, permite distinguir os comportamentos linear e quadrático do tempo de execução em função de cada um dos dois parâmetros.



| <i>traços/amostras</i>   | 128/128 | 128/256 | 256/128 |
|--------------------------|---------|---------|---------|
| <i>tempo total</i>       | 2757    | 10817   | 5310    |
| <i>inicialização</i>     | 3       | 4       | 4       |
| <i>fft</i>               | 32      | 69      | 61      |
| <i>gera sistema</i>      | 456     | 1814    | 858     |
| <i>resolve sistema</i>   | 1663    | 6493    | 3139    |
| <i>deslocamento fase</i> | 324     | 1287    | 643     |
| <i>acumulação</i>        | 279     | 1150    | 605     |

Tabela 4.3 - Comportamentos linear e quadrático dos tempos de execução com relação a  $n_t$  e  $n_x$ .

A tabela 4.3 está de acordo com o cálculo da complexidade desenvolvido. Observe que o tempo de execução cresce linearmente com o número de traços (compare a segunda e a quarta colunas) e quadraticamente com o número de amostras (compare a segunda e a terceira colunas). Observe ainda que a hierarquia entre os tempos de execução dos passos é mantida.

### ***4.3 - Dependências nos Laços Centrais***

A exploração do paralelismo existente no algoritmo requer o conhecimento das dependências em seus passos. Esta análise será conduzida em nível conceitual, evitando que detalhes de programação possam dificultar o raciocínio. Inicialmente serão analisadas as dependências entre os passos de uma única iteração do laço central. Posteriormente serão analisadas as dependências internas a cada passo de uma única iteração do laço central. Finalmente, as dependências entre iterações distintas do laço central.

É importante recordar que cada iteração do corpo do aninhamento de laços utiliza dados em uma dada frequência, registrados na profundidade imediatamente anterior à desejada. O resultado da iteração são os dados na mesma frequência, registrados na profundidade desejada, além da acumulação das componentes devidas na seção de saída.

Os passos de cada iteração deste corpo são absolutamente dependentes, pois os dados da saída de cada passo são os dados de entrada do próximo passo. A saída da montagem do sistema tridiagonal é a entrada da solução deste mesmo sistema. Por sua vez, a solução do sistema tridiagonal é a entrada para o processo de deslocamento de fase, cujo resultado é utilizado na acumulação. Conseqüentemente, não há paralelismo entre os passos de uma iteração do laço.

Entretanto, cada passo é executado sobre um vetor de  $n_x$  dados, o que pode possibilitar independências internas em cada passo. Estas independências ocorrem em todos os passos, exceto na solução de sistemas tridiagonais. A montagem do sistema tridiagonal, a aplicação do deslocamento de fase e a acumulação podem ser realizadas para cada amostra isoladamente. Já a solução do sistema tridiagonal utiliza todas as amostras de uma mesma profundidade e freqüência. Conseqüentemente, há paralelismo interno a alguns passos de cada iteração do laço.

Determinadas as dependências internas a cada passo e as dependências internas a cada execução do laço, resta avaliar as dependências entre iterações do laço. Para tanto, é fundamental observar que os laços na freqüência e na profundidade são intercambiáveis. Fixando-se uma freqüência, pode-se extrapolar os dados nesta freqüência para todas as profundidades. Por outro lado, fixando-se uma profundidade, é possível extrapolar toda a seção em freqüência para a próxima profundidade.

Este fato ocorre porque os dados para cada freqüência são independentes dos dados para outras freqüências e porque o processo de extrapolação de uma freqüência depende exclusivamente desta freqüência. Em outras palavras, uma iteração do laço em freqüência não necessita dados de outras freqüências.

O mesmo não ocorre com as iterações do laço em profundidade, pela própria natureza do processo de extrapolação. Para obter dados em uma certa profundidade, são necessários os dados na profundidade imediatamente anterior.

Conseqüentemente, se o laço em freqüência for o laço externo, as iterações do laço interno (em profundidade) serão dependentes. Entretanto, se o laço externo for em profundidade, as iterações do laço interno (em freqüência) serão independentes.

A rigor, há alguma independência entre passos das iterações mesmo quando o laço em frequências for externo. Isto ocorre porque a acumulação em uma iteração é independente da montagem do sistema tridiagonal de outra iteração. Entretanto, trata-se de independência de pequena monta, quando contrastada com a independência existente quando o laço em profundidade for externo. Neste caso, todos os passos de cada iteração são completamente independentes dos de outras.

#### ***4.4 - Formas de Paralelismo***

A análise de dependências efetuada indica possibilidade de paralelismo em dois locais. O primeiro é interno a cada passo de cada iteração do aninhamento de laços centrais. O segundo é entre iterações distintas desse aninhamento, quando o laço mais externo é em profundidade.

A primeira possibilidade é típica do processamento vetorial. Trata-se de vetorizar os laços internos a cada passo do aninhamento central. Esta forma de paralelismo foi abandonada por dois motivos. Primeiro, não é possível vetorizar a Eliminação de Gauss para sistemas tridiagonais. Isto limita bruscamente a efetividade da paralelização, pois este passo representa aproximadamente 60% do tempo de execução total. Segundo, porque a troca do algoritmo da Eliminação de Gauss por outro algoritmo paralelo (como a Redução Cíclica), apesar de aumentar a quantidade de paralelismo, aumenta substancialmente a complexidade do problema (torna-o  $O(n_t^2 n_x \log(n_x))$ ).

Experimentos conduzidos no IBM 3090 da Petrobrás indicam a correção desta decisão. Utilizando a Eliminação de Gauss, o grau de vetorização de um programa (definido como a razão entre o tempo gasto em instruções vetoriais sobre o tempo total do programa) é inferior a 30%, para problemas de 512 amostras por 512 traços. Quando a Redução Cíclica é utilizada, o grau de vetorização sobe a 80%, mas o tempo de execução é acrescido de 50%, quando comparado com a Eliminação de Gauss.

Já a segunda fonte de paralelismo permite múltiplas formulações, propícias a máquinas MIMD. Destas, selecionamos duas para análise, implementação e experimentação.

A primeira forma particiona o espaço de freqüências pelos processadores. Cada processador recebe os dados relativos a um conjunto de freqüências e extrapola estes dados da superfície até o fundo da seção. A acumulação é realizada em dois passos. Localmente, cada processador acumula as amostras das freqüências sob seu controle. Globalmente, um processador utiliza as acumulações parciais de todos os outros processadores para realizar a acumulação final.

A segunda forma particiona o espaço de profundidades pelos processadores. Cada processador utiliza os dados de todas as freqüências na profundidade atual e extrapola-os um passo em profundidade. Os dados extrapolados tornam-se disponíveis para o processador responsável pela próxima profundidade e assim sucessivamente.

A quantidade de paralelismo desta segunda forma depende do momento em que os dados tornam-se disponíveis aos processadores. Se toda a seção (isto é, todas as freqüências) for processada antes das transferências, a computação é seqüencial. Entretanto, se algumas freqüências tornam-se disponíveis enquanto outras são processadas, há paralelismo entre os processadores. Se uma freqüência for processada e em seguida disponibilizada, o paralelismo é na forma de um *pipeline* entre os processadores, onde o elemento que transita pelo *pipeline* é o conjunto de dados de uma determinada profundidade.

É interessante comparar estas duas formas de paralelismo. Enquanto na primeira os dados relativos a cada freqüência estão estacionados em um processador, na segunda estes dados transitam entre processadores. Enquanto na primeira os dados relativos a cada profundidade estão distribuídos pelos processadores, na segunda estes dados serão computados por um único processador. Como as extrapolações são independentes nas freqüências, o tráfego de dados durante a extrapolação é desnecessário na primeira forma mas necessário na segunda. Como as acumulações são independentes nas profundidades, o tráfego de dados durante a acumulação é necessário na primeira forma mas desnecessário na segunda.

## 4.5 - Análise da Complexidade das Formas de Paralelismo

A análise da complexidade será efetuada apenas para máquinas paralelas de memória distribuída. Análise similar pode ser desenvolvida para máquinas paralelas de memória central.

### 4.5.1 - Análise da Complexidade da Primeira Forma

O tempo de execução seqüencial, derivado da análise da complexidade, é aproximadamente  $k_0 n_t^2 n_x$ , onde  $k_0$  representa o tempo de execução de uma iteração do aninhamento central.

Já o tempo de execução da primeira forma de paralelismo é composto por três parcelas: o custo da transmissão inicial da seção em frequência, o custo da computação local a cada processador e o custo da acumulação final.

Admitindo-se que as FFTs são computadas por um único processador, o custo da transmissão inicial é de  $k_1 n_\omega n_x$ , onde  $k_1$  indica o tempo necessário para transmitir um dado. Isto ocorre porque toda a seção em frequência deve ser transmitida de um único processador para os demais (na realidade, o custo é ligeiramente inferior, visto que o processador que calcula a seção não precisa transmiti-la para si mesmo). Substituindo  $n_\omega = n_t/2$  e incorporando a divisão por 2 a  $k_1$ , atinge-se  $k_1 n_t n_x$ .

A computação local é idêntica à computação seqüencial, apenas realizada sobre um número menor de frequências. Indicando por  $p$  o número de processadores e admitindo que  $p$  divide exatamente  $n_\omega$ , o número de frequências por processador será  $n_\omega/p$ . Como estas frequências serão tratadas simultaneamente pelos processadores, o custo da computação local será  $(k_0 n_t^2 n_x)/p$ .

O custo da acumulação final abrange tanto a transmissão das  $p$  acumulações parciais quanto a acumulação final propriamente dita. Cada transmissão movimenta  $n_z n_x$  dados, que também é o número de somas necessárias à incorporação destes dados na acumulação final. Como há  $p$  transmissões (na realidade,  $p - 1$ , que serão aproximadas por  $p$ ) e armazenando em  $k_2$  o custo da transmissão e da soma de cada dado, obtém-se o custo de acumulação final de  $k_2 p n_z n_x$ . Como  $n_z = n_t$ , atinge-se  $k_2 p n_t n_x$ .

Conseqüentemente, o custo da execução paralela será  $k_1 n_t n_x + (k_0 n_t^2 n_x)/p + k_2 p n_t n_x$ , e o ganho será

$$S(n_t, n_x, p) = \frac{k_0 n_t^2 n_x}{k_1 n_t n_x + \frac{k_0 n_t^2 n_x}{p} + k_2 p n_t n_x}$$

Simplificando-se esta fórmula e utilizando  $k_3$  para representar o quociente de  $k_1$  por  $k_0$  e  $k_4$  para representar o quociente de  $k_2$  por  $k_0$  atinge-se

$$S(n_t, n_x, p) = \frac{p n_t}{k_3 p + n_t + k_4 p^2}$$

Esta formulação permite três conclusões interessantes. Primeiro, o ganho independe de  $n_x$ , ou seja, do número de traços por seção. Segundo, para um valor fixo de  $p$ , o ganho tende a  $p$  quando  $n_t$  tende a infinito. Isto significa que, para qualquer número de processadores, haverá um problema de tamanho suficiente para obter ganho ótimo. Terceiro, para um valor fixo de  $n_t$ , o ganho tende a 0 quando  $p$  tende a infinito. Conseqüentemente, para um problema de tamanho fixo, não adianta acrescentar processadores à máquina acima de um certo valor, pois o ganho cairá.

#### 4.5.2 - Análise da Complexidade da Segunda Forma

Na segunda forma de paralelismo cada profundidade é tratada por um único processador, que recebe as amostras da profundidade anterior correspondentes a uma freqüência, extrapola estas amostras para a profundidade sob sua responsabilidade e envia-as para o próximo processador. Após processar todas as profundidades sob sua responsabilidade, o processador envia os resultados acumulados para um processador central que os armazenará.

Quando observada globalmente, esta computação é organizada como em um *pipeline*. Inicialmente, o *pipeline* deve ser preenchido, o que acarreta a ociosidade de alguns processadores. Em um segundo momento, todos os processadores estão ocupados e o *pipeline* encontra-se completamente preenchido. Finalmente, o *pipeline* vai se esvaziando enquanto os resultados são comunicados ao processador centralizador.

Para efeitos desta análise, a computação paralela será dividida em duas fases. A primeira fase ocorre do início do processamento até que o momento em que o *pipeline* começa a se esvaziar. A segunda fase contém o esvaziamento do *pipeline* e a comunicação de resultados.

A primeira fase contém  $n_\omega n_z/p$  iterações do laço central. Para concluir este fato, observe inicialmente que cada processador é responsável por  $n_z/p$  profundidades e que o processamento de cada profundidade requer o trato de  $n_\omega - 1$  frequências, aproximadas por  $n_\omega$ . Em seguida, observe que a primeira fase corresponde exatamente aos instantes nos quais o primeiro processador trata todas as suas profundidades. Estas duas observações conduzem à conclusão acima.

Cada iteração do laço central contém os passos usuais do laço acrescidos da comunicação das amostras extrapoladas para o próximo processador. Como estas duas computações demandam da ordem de  $n_x$  operações, o número de operações da primeira fase é de  $n_x n_\omega n_z/p$ . Denotando por  $k_1$  o tempo de execução destas operações, obtém-se  $k_1 n_x n_\omega n_z/p$  para o tempo de execução da primeira fase.

A segunda fase possui  $p$  passos, um para cada processador. O passo consiste em enviar o resultado acumulado nas profundidades tratadas pelo processador  $i$  para armazenagem. Enquanto isto ocorre, os processadores  $i+1$  até  $p$  continuam trabalhando nas iterações restantes da fase anterior. Destes dois processamentos, arbitramos que o primeiro demanda maior tempo de execução, visto requerer a transmissão de  $n_z/p$  vetores de tamanho  $n_x$ , enquanto o segundo depende apenas de  $n_x$ . Desta forma, o tempo de execução da segunda fase será  $k_2 n_z n_x$ .

Conseqüentemente, o tempo de execução da segunda forma será  $k_1 n_x n_\omega n_z/p + k_2 n_z n_x$ . Substituindo  $n_\omega$  e  $n_z$  por suas relações com  $n_t$  e incorporando eventuais constantes a  $k_1$  e  $k_2$ , atinge-se o tempo de execução de  $k_1 n_x n_t^2/p + k_2 n_t n_x$  e o ganho de

$$S(n_t, n_x, p) = \frac{k_0 n_t^2 n_x}{\frac{k_1 n_x n_t^2}{p} + k_2 n_t n_x}$$

que pode ser simplificado para

$$S(n_t, n_x, p) = \frac{pn_t}{k_3n_t + k_4p}$$

Este resultado permite três conclusões. Primeiro, o ganho independe de  $n_x$ . Segundo, fixo o número de processadores, o ganho tende a  $p/k_3$  quando  $n_t$  tende a infinito, ou seja, o ganho tende a uma fração (desconhecida) do número de processadores quando o tamanho do problema é arbitrariamente aumentado. Terceiro, para um problema de tamanho fixo, o ganho tende a  $n_t/k_4$  quando o número de processadores tende a infinito. Ou seja, aumentar arbitrariamente o número de processadores não aumenta o ganho de um problema de tamanho fixo além de um certo valor.



## Capítulo 5 - Experimentos e Resultados em Máquinas Hipercúbicas

Este capítulo descreve a implementação e analisa os resultados das duas formas de paralelização da migração  $\omega$ -x no INTEL/iPSC e no NCP I/COPPE, as duas máquinas hipercúbicas disponíveis na COPPE.

As implementações nas duas máquinas distinguem-se apenas por detalhes das primitivas de comunicação e sincronização. Por esta razão, as implementações serão descritas uma única vez. Já os resultados serão tratados individualmente.

### 5.1 - Primeira Forma de Paralelização

#### 5.1.1 - Descrição da Implementação

A primeira forma de paralelização permite múltiplas alocações de freqüências aos processadores. Por exemplo, freqüências sucessivas podem ser alocadas a um único processador, ou então distribuídas pelos processadores. Implementou-se a segunda possibilidade. Representando-se por  $p$  o número de processadores, cada processador  $i$  trata amostras das freqüências  $i + 1, i + 1 + p, i + 1 + 2p, \dots$  para  $i = 1$  até  $p$ .

A composição da seção de entrada e a execução da FFT são realizadas unicamente pelo primeiro processador, que em seguida envia os trechos da seção em freqüência para os demais processadores. Estes só iniciam sua computação após receberem as amostras relativas a todas as freqüências sob sua responsabilidade e o primeiro processador continua sua computação após transmitir todos os dados. Admitindo que o número de processadores divide exatamente o número de freqüências a processar (balanceamento de carga

perfeito), este processo compreende a transmissão de  $(p - 1)(n_\omega - 1)/p$  mensagens, cada uma com  $n_x$  amostras.

Após o término do processamento de sua quota de frequências, cada nó envia sua acumulação parcial para o primeiro processador, para a totalização final. Esta estratégia de concentrar as mensagens ao fim do processamento pode, potencialmente, congestionar o tráfego entre os processadores. Uma variação interessante desta estratégia é cada processador enviar sua acumulação parcial após  $q$  profundidades. Esta foi a variação implementada.

Foram realizados diversos experimentos para determinar a influência de  $q$  no desempenho. A influência observada foi pequena. Mesmo assim, optou-se por utilizar  $q = 32$ , que foi o valor correspondente aos melhores resultados nas duas máquinas.

Foi testada também uma outra forma de distribuição de trabalho em que as frequências atribuídas aos nós são contíguas. Os resultados foram ligeiramente inferiores aos apresentados com as frequências esparsas.

### 5.1.2 - Descrição dos Experimentos

Cada experimento realizado é reportado por duas tabelas. A primeira contém os tempos de execução (CPU), em segundos, medidos no primeiro processador. Este foi escolhido por ser, por força da implementação, o primeiro nó a iniciar a computação e o último nó a terminá-la. A segunda tabela contém os ganhos, medidos pelo quociente entre o tempo de execução do programa seqüencial executado em um processador e o tempo de execução da versão paralela com  $p$  processadores.

As lacunas existentes nas tabelas correspondem a dimensões do problema que não puderam ser experimentadas por insuficiência de memória nos nós (8Mb no INTEL/iPSC e 2Mb NCP I/COPPE). Foram coletados os tempos de execução para 1, 4 e 8 processadores, com todas as combinações possíveis de 64, 128, 256, 512 e 1024 amostras e traços.

### 5.1.3 - Apresentação e Análise de Resultados no INTEL/iPSC

A tabela a seguir contém os tempos de execução da versão seqüencial do algoritmo em um processador:

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|----------------|
| Numero de Amostras |              |               |               |               |                |
| 64                 | 3,73         | 7,37          | 14,65         | 29,50         | 58,80          |
| 128                | 14,63        | 29,38         | 58,88         | 119,29        | 238,03         |
| 256                | 58,43        | 117,09        | 236,42        | 479,96        | 957,13         |
| 512                | 234,46       | 466,88        | 944,90        | 1.924,29      | 3.840,75       |
| 1024               | 935,08       | 1.864,50      | 3.760,41      | 7.702,57      | .              |

Tabela 5.1 - INTEL/iPSC - Tempos de execução (s) com 1 processador.

Observa-se nitidamente o comportamento linear do tempo de execução com o aumento de traços, além do comportamento quadrático com o aumento de amostras. Note-se que a seção com 512 traços por 1024 amostras exigiu mais de duas horas de processamento. A dificuldade de experimentar seções maiores é patente; uma seção real típica contém da ordem de 3000 traços por 1500 amostras, com tempo de execução seqüencial da ordem de 27 horas.

Apresenta-se, a seguir, o tempo de execução e o ganho para quatro processadores. Observe-se o aumento do número de experimentos que não puderam ser realizados por falta de memória. Isto deve-se à carga do ambiente paralelo em cada processador, bem como ao aumento (modesto) dos requisitos de memória da versão paralela quando comparada com a seqüencial.

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|----------------|
| Numero de Amostras |              |               |               |               |                |
| 64                 | 1,58         | 2,20          | 4,13          | 8,15          | 15,96          |
| 128                | 4,10         | 7,83          | 15,34         | 31,16         | 62,12          |
| 256                | 14,96        | 29,85         | 60,25         | 122,90        | 245,23         |
| 512                | 59,32        | 118,19        | 238,57        | 488,17        | .              |
| 1024               | 233,46       | 468,49        | 944,18        | .             | .              |

Tabela 5.2 - INTEL/iPSC - Tempos de execução (s) com 4 processadores

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|----------------|
| Numero de Amostras |              |               |               |               |                |
| 64                 | 2,36         | 3,35          | 3,55          | 3,62          | 3,68           |
| 128                | 3,57         | 3,75          | 3,84          | 3,83          | 3,83           |
| 256                | 3,91         | 3,92          | 3,92          | 3,91          | 3,90           |
| 512                | 3,95         | 3,95          | 3,96          | 3,94          | .              |
| 1024               | 4,01         | 3,98          | 3,98          | .             | .              |

Tabela 5.3 - INTEL/iPSC - Ganhos com 4 processadores

Observe que o ganho é praticamente independente do número de traços, o que é coerente com a análise da complexidade do algoritmo paralelizado anteriormente efetuada. Esta independência não ocorre nos casos 64 e 128 traços por 64 e 128 amostras. Nesta região, a dimensão do problema *em cada processador* é suficientemente pequena (8 a 16 frequências por processador) para desacoplar o tempo de execução da análise da complexidade efetuada. Fora desta região, a variação máxima entre os ganhos para um número de amostras fixo é de 3%.

O ganho é crescente com o número de amostras, como previsto pela análise da complexidade. Observe ainda que ganhos muito próximos do ótimo foram atingidos nos problemas de maior tamanho. A tabela apresenta um ganho de 4,01, superior ao ótimo para este número de processadores, que pode ser atribuído a imprecisões de medida.

Apresenta-se, a seguir, o tempo de execução e o ganho para oito processadores.

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|----------------|
| Numero de Amostras |              |               |               |               |                |
| 64                 | 0,69         | 1,21          | 2,24          | 4,72          | 9,44           |
| 128                | 2,14         | 4,40          | 8,39          | 16,41         | 32,97          |
| 256                | 7,95         | 15,56         | 30,94         | 63,10         | 125,77         |
| 512                | 30,18        | 60,00         | 120,96        | 247,37        | .              |
| 1024               | 117,49       | 234,73        | 475,20        | .             | .              |

Tabela 5.4 - INTEL/iPSC - Tempos de execução (s) com 8 processadores

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|----------------|
| Numero de Amostras |              |               |               |               |                |
| 64                 | 5,40         | 6,09          | 6,54          | 6,25          | 6,23           |
| 128                | 6,84         | 6,68          | 7,02          | 7,27          | 7,22           |
| 256                | 7,35         | 7,52          | 7,64          | 7,61          | 7,61           |
| 512                | 7,77         | 7,78          | 7,81          | 7,78          | .              |
| 1024               | 7,96         | 7,94          | 7,91          | .             | .              |

Tabela 5.5 - INTEL/iPSC - Ganhos com 8 processadores

Novamente, não há dependência do ganho com o número de traços, exceto na região entre 64 e 256 amostras por 64 e 128 traços. O ganho é crescente com o número de amostras e apresenta valores muito próximos ao ótimo.

Note-se que houve um aumento da região em que o ganho se apresenta como dependente do aumento de traços. Isto se deve ao aumento do número de processadores (de 4 para 8) que fez com que dobrasse a dimensão do problema que gerava tarefas muito pequenas por processador (8 a 16 frequências por processador).

#### 5.1.4 - Apresentação e Análise de Resultados no NCP I/COPPE

A tabela a seguir contém os tempos de execução da versão seqüencial do algoritmo em um processador:

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|----------------|
| Numero de Amostras |              |               |               |               |                |
| 64                 | 35,95        | 71,56         | 141,78        | 281,71        | 561,57         |
| 128                | 145,17       | 289,70        | 574,57        | 1.140,17      | .              |
| 256                | 583,70       | 1.165,80      | 2.318,44      | .             | .              |
| 512                | 2.342,07     | 4.680,29      | .             | .             | .              |
| 1024               | 9.385,05     | .             | .             | .             | .              |

Tabela 5.6 - NCP I/COPPE - Tempos de execução (s) com 1 processador

A tabela de execução apresenta mais lacunas do que a tabela do INTEL/iPSC porque a memória dos processadores do NCP I/COPPE é menor (2Mb versus 8Mb).

Esta tabela também mostra a linearidade do crescimento do tempo de execução com o aumento de traços e o crescimento quadrático com o aumento das amostras, conforme já previsto pela análise da complexidade do algoritmo.

Note que a maior seção testada, com 64 traços e 1024 amostras, apresentou um tempo de execução superior a 2,5 horas. Fazendo-se uma projeção do tempo de execução para uma seção real típica que contém 3000 traços por 1500 amostras, chega-se a uma estimativa de tempo de execução superior a 263 horas. Comparando com a projeção análoga feita para o INTEL/iPSC vê-se que a capacidade computacional deste é quase 10 vezes maior do que a do NCP I/COPPE.

Apresenta-se a seguir o tempo de execução e o ganho para quatro processadores. Observe-se que o número de experimentos que não pôde ser realizado não aumentou, indicando que o ambiente de paralelismo no NCP I/COPPE ocupa muito pouca memória, ao contrário do INTEL/iPSC.

As tabelas a seguir contém os tempos de execução e os ganhos para quatro processadores.

|                           | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|---------------------------|--------------|---------------|---------------|---------------|----------------|
| <b>Numero de Amostras</b> |              |               |               |               |                |
| 64                        | 10,11        | 20,03         | 39,62         | 78,73         | 156,99         |
| 128                       | 38,70        | 77,15         | 152,98        | 303,61        | .              |
| 256                       | 151,40       | 302,44        | 601,51        | .             | .              |
| 512                       | 598,74       | 1.197,38      | .             | .             | .              |
| 1024                      | 2.382,56     | .             | .             | .             | .              |

Tabela 5.7 - NCP I/COPPE - Tempos de execução (s) com 4 processadores

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|----------------|
| Numero de Amostras |              |               |               |               |                |
| 64                 | 3,56         | 3,57          | 3,58          | 3,58          | 3,58           |
| 128                | 3,75         | 3,76          | 3,76          | 3,76          | .              |
| 256                | 3,86         | 3,85          | 3,85          | .             | .              |
| 512                | 3,91         | 3,91          | .             | .             | .              |
| 1024               | 3,94         | .             | .             | .             | .              |

Tabela 5.8 - NCP I/COPPE - Ganhos com 4 processadores

Observa-se que o ganho independe do número de traços e é crescente com o número de amostras, confirmando a previsão da análise da complexidade.

Vê-se também que a tabela de ganho não apresenta a região de exceção à independência do número de traços, como ocorre no INTEL/iPSC. Isto pode ser explicado pela menor capacidade computacional do NCP I/COPPE, que faz com que mesmo as menores dimensões do problema testadas (que correspondem de 8 a 16 frequências por processador) já sejam dominadas pela complexidade do algoritmo.

Ganhos muito próximos do ótimo foram obtidos para as maiores dimensões testadas.

As tabelas a seguir contêm os tempos de execução e os ganhos para oito processadores.



|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|----------------|
| Numero de Amostras |              |               |               |               |                |
| 64                 | 5,92         | 11,50         | 22,54         | 44,69         | 89,14          |
| 128                | 20,96        | 41,49         | 82,07         | 162,77        | .              |
| 256                | 78,86        | 157,17        | 312,38        | .             | .              |
| 512                | 305,60       | 610,88        | .             | .             | .              |
| 1024               | 1.204,50     | .             | .             | .             | .              |

Tabela 5.9 - NCP I/COPPE - Tempos de execução (s) com 8 processadores

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos | 1024<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|----------------|
| Numero de Amostras |              |               |               |               |                |
| 64                 | 6,07         | 6,22          | 6,29          | 6,30          | 6,30           |
| 128                | 6,93         | 6,98          | 7,00          | 7,00          | .              |
| 256                | 7,40         | 7,42          | 7,42          | .             | .              |
| 512                | 7,66         | 7,66          | .             | .             | .              |
| 1024               | 7,79         | .             | .             | .             | .              |

Tabela 5.10 - NCP I/COPPE - Ganhos com 8 processadores

Novamente confirma-se a expressão da complexidade, com o ganho se mostrando independente do aumento de traços e crescente com o aumento de amostras. Observa-se porém o reaparecimento da região (64 a 128 traços por 64 a 128 amostras) em que o ganho varia com o aumento de traços. Isto pode ser explicado pelo aumento do número de processadores (de 4 para 8) que fez com que as tarefas alocadas a cada processador (4 a 8 frequências) fossem suficientemente pequenas para desacoplar os tempos de execução da complexidade analisada.

Para as maiores dimensões testadas o ganho apresentado aproxima-se do ótimo.

## 5.2 - Segunda Forma de Paralelização

### 5.2.1 - Descrição da Implementação

Na primeira forma de paralelização, as frequências eram distribuídas entre os processadores. Nesta segunda forma, as profundidades é que serão divididas entre os processadores. Assim, para  $p$  processadores, o processador  $i$  será responsável pelas profundidades  $i, i + p, i + 2p, i + 3p, \dots$ , para  $i = 1$  até  $p$ .

A geração da seção de entrada e a execução da FFT são tarefas do primeiro processador. A partir daí, então, vai se formar um *pipeline* em anel, onde o primeiro nó processa todas as frequências na primeira profundidade de sua responsabilidade e transmite os dados para o próximo nó do anel, que os processa em todas as frequências na profundidade seguinte e assim sucessivamente.

Uma característica marcante desta segunda forma é que não há mais acumulações parciais dos resultados em cada nó. Como cada profundidade é processada por um único nó, a acumulação que é feita nos nós já é a acumulação global, para aquelas profundidades. Assim, ao final da sua computação, cada nó transmite suas acumulações para o primeiro nó apenas para centralização e apresentação dos resultados.

Note-se que se a transmissão dos dados processados por um nó se der só ao final do processamento de todas as frequências numa dada profundidade, haverá um processamento seqüencial, onde apenas um nó trabalha por vez. Para evitar este comportamento seqüencial no processamento, a cada  $q$  frequências processadas é feita a comunicação para o próximo nó do anel.

Supondo-se que o número de processadores divide exatamente o número de profundidades (balanceamento perfeito de carga), haverá a transmissão de  $n_z(n_\omega/q)$  mensagens cada uma com  $qn_x$  amostras, para a distribuição de trabalho entre os nós.

Medidas preliminares mostraram uma grande influência de  $q$  no tempo de execução. O valor que apresentou os melhores resultados foi  $q = 8$ , que foi adotado.

### 5.2.2 - Descrição dos Experimentos

Os experimentos com esta segunda forma de paralelização só foram realizados no INTEL/iPSC, devido ao grande tempo necessário à condução dos experimentos. Pelo mesmo motivo, o conjunto de dimensões testadas foi reduzido, relativamente àquele usado na primeira forma.

Cada experimento é reportado por duas tabelas. A primeira, contém os tempos de execução (CPU), em segundos, e a segunda contém os ganhos, medidos de maneira análoga à primeira forma de paralelização.

### 5.2.3 - Apresentação e Análise de Resultados no INTEL/iPSC

As tabelas a seguir apresentam os resultados obtidos.

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|
| Numero de Amostras |              |               |               |               |
| 64                 | 1,40         | 2,71          | 5,25          | 10,49         |
| 128                | 2,81         | 5,53          | 10,66         | 21,28         |
| 256                | 8,49         | 16,51         | 32,87         | 66,37         |
| 512                | 33,13        | 64,35         | 128,58        | .             |

Tabela 5.11 - INTEL/iPSC - Tempos de execução (s) com 8 processadores

|                    | 64<br>tracos | 128<br>tracos | 256<br>tracos | 512<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|
| Numero de Amostras |              |               |               |               |
| 64                 | 2,66         | 2,72          | 2,79          | 2,81          |
| 128                | 5,21         | 5,31          | 5,52          | 5,61          |
| 256                | 6,88         | 7,09          | 7,19          | 7,23          |
| 512                | 7,08         | 7,26          | 7,35          | .             |

Tabela 5.12 - INTEL/iPSC - Ganhos com 8 processadores

A observação da tabela de ganhos mostra um aumento no patamar dos ganhos quando se passa de 64 para 128 amostras muito superior ao esperado. Isto pode ser explicado pela observação de que com 64 traços e com  $q = 8$ , formam-se *bolhas* no *pipeline*, i.e., os processadores ficam ociosos entre o fim do processamento de uma profundidade e o início do processamento da próxima.

Ressalvadas as linhas da tabela correspondentes a 64 e 128 amostras que apresentam o efeito de *bolha* (64 em maior escala), a tabela de ganhos mostra a sua independência com relação ao aumento dos traços, com diferenças inferiores a 5% que não foram analisadas.

Observa-se também um crescimento do ganho com o aumento do tamanho do problema, o que confirma a análise da complexidade de modo restrito (já que a faixa de crescimento do problema é pequena).

Os maiores problemas testados apresentaram ganho próximo ao ótimo.

### 5.3 - Conclusões

Os ganhos obtidos foram próximos do ótimo nas duas máquinas, o que atesta a viabilidade da migração sísmica nesta classe de máquinas paralelas.

Comprovou-se, com alguma restrição, a independência do ganho em relação a  $n_x$  e o crescimento do ganho com o aumento de traços. A terceira previsão da

análise da complexidade não pôde ser comprovada pela limitação do número dos processadores (máximo de 8).

Ficou demonstrado também, o maior custo (em uso de memória) do paralelismo no INTEL/iPSC comparado com o NCP I/COPPE. A projeção do tempo para o processamento de uma seção real permitiu a medida comparativa do poder computacional mostrando que a CPU do INTEL/iPSC é cerca de 10 vezes mais rápida que a do NCP I/COPPE.

Na segunda forma de paralelização, a menor quantidade de experimentos não possibilitou uma análise tão rica quanto a anterior. Uma observação interessante foi a verificação experimental das *bolhas no pipeline*, que prejudicaram o desempenho esperado.

Apesar das duas formas mostrarem boas possibilidades de desempenho na solução do problema, a primeira forma apresentou uma maior eficiência em todas as medidas comuns.

## Capítulo 6 - Experimentos e Resultados em Máquinas de Memória Central

Este capítulo descreve a implementação e analisa os resultados de três formas de paralelização da migração  $\omega$ -x no IBM 3090/600VF e no IBM 9021/820.

Os experimentos foram iniciados no IBM 3090/600VF que era a máquina científica de produção da PETROBRÁS, com 6 CPU's e 6 *Vector Facilities*. Posteriormente esta máquina foi substituída pelo IBM 9021/820, com 4 CPU's e 4 *Vector Facilities*, onde os experimentos foram repetidos. As duas máquinas são de memória central e também *binary compatible* (de modo restrito, já que o IBM 9021 executa o código binário do IBM 3090 mas a recíproca não é verdadeira).

As segunda e terceira formas de paralelização só foram experimentadas no IBM 9021/820 devido à desativação do IBM 3090/600VF.

O número de experimentos realizados nestas máquinas de memória central foi muito menor do que nas máquinas hipercúbicas, por serem máquinas de produção e os experimentos exigirem exclusividade no uso do equipamento. Mesmo assim, foram usadas aproximadamente 75 horas de CPU dos IBM 3090 e 9021 com exclusividade.

### 6.1 - Primeira Forma de Paralelização

#### 6.1.1 - Descrição da Implementação

Esta forma de paralelização é a mesma primeira forma das máquinas hipercúbicas, onde as frequências são distribuídas entre os processadores, que as processam em todas as profundidades.

Como as máquinas agora em teste são de memória central, não é mais necessário que os dados sejam transmitidos à memória dos processadores. Do mesmo modo, não há mais necessidade de comunicação dos resultados. Entretanto, como todos os processadores vão acumular os seus resultados numa estrutura comum, é necessário evitar *corridas*<sup>5</sup> entre os processadores. Para isto, basta garantir exclusividade de acesso utilizando regiões críticas.

Foram feitas implementações desta primeira forma, atribuindo frequências contíguas (forma 1) e frequências esparsas (formas 1a e 1b) aos processadores. Com relação à região crítica para acumulação dos resultados, foram experimentadas duas formas de controle de acesso: um único semáforo para a acumulação de todas as profundidades (formas 1 e 1a) e um semáforo para cada profundidade a acumular (forma 1b).

### 6.1.2 - Descrição dos Experimentos

Por causa da redução do número de experimentos optou-se por variar o número de traços,  $n_x$ , e o número de amostras,  $n_t$ , simultaneamente. Foram testados problemas com dimensões ( $n_x \times n_t$ ) de 64X64, 128X128, 256X256 e 512X512, para 1, 2, 3, 4, 5 e 6 processadores no caso do IBM 3090/600VF e 1, 2, 3 e 4 processadores no IBM 9021/820.

Como para as máquinas hipercúbicas, cada experimento é reportado por duas tabelas. A primeira contém os tempos de execução (tempo de parede) em segundos. A segunda contém os ganhos, calculados pelo quociente entre o tempo de execução de cada forma de implementação com um processador e o tempo de execução desta mesma forma com  $p$  processadores em paralelo.

A redução do número de experimentos exigiu uma nova organização das tabelas, que apresentam de uma só vez todos os resultados de cada forma de implementação, para os  $p$  processadores testados.

### 6.1.3 - Apresentação e Análise dos Resultados para o IBM 3090

As tabelas a seguir apresentam os tempos de execução e os ganhos para a forma 1:

---

<sup>5</sup>*race conditions*

| Dimensao | numero de processadores |         |         |         |         |        |
|----------|-------------------------|---------|---------|---------|---------|--------|
|          | 1                       | 2       | 3       | 4       | 5       | 6      |
| 64X64    | 1,230                   | 0,923   | 1,044   | 1,208   | 1,168   | 1,383  |
| 128X128  | 8,900                   | 6,550   | 3,497   | 3,330   | 3,135   | 2,789  |
| 256X256  | 67,675                  | 57,091  | 23,559  | 18,883  | 15,339  | 12,918 |
| 512X512  | 520,707                 | 424,081 | 179,114 | 134,094 | 108,386 | 92,047 |

Tabela 6.1 - IBM 3090/600VF - Tempos de execução (s) com a forma 1.

| Dimensao | numero de processadores |       |       |       |       |       |
|----------|-------------------------|-------|-------|-------|-------|-------|
|          | 1                       | 2     | 3     | 4     | 5     | 6     |
| 64X64    | 1,000                   | 1,333 | 1,178 | 1,018 | 1,053 | 0,889 |
| 128X128  | 1,000                   | 1,359 | 2,545 | 2,673 | 2,839 | 3,191 |
| 256X256  | 1,000                   | 1,185 | 2,873 | 3,584 | 4,412 | 5,239 |
| 512X512  | 1,000                   | 1,228 | 2,907 | 3,883 | 4,804 | 5,657 |

Tabela 6.2 - IBM 3090/600VF - Ganhos obtidos com a forma 1.

Os ganhos mostram-se crescentes com o aumento do problema, com exceção das medidas com dois processadores. Não se procurou determinar as razões desta falha. Ganhos próximos ao ótimo foram obtidos com os maiores problemas testados.

A observação da tabela de ganhos mostra que o IBM 3090 apresenta um alto custo de paralelismo. Só para as maiores dimensões do problema obtiveram-se bons ganhos. Fica patente que se está tratando com uma máquina preparada para grandes problemas. Seções de tamanho real que não cabiam nas máquinas hipercúbicas são processadas nesta máquina rotineiramente. Estas seções não



foram testadas para manter coerência com as medidas possíveis nos outros equipamentos usados.

Para efeito de comparação, seguem-se as tabelas de ganhos com as outras variantes da primeira forma de paralelização.

| Dimensao | numero de processadores |       |       |       |       |       |
|----------|-------------------------|-------|-------|-------|-------|-------|
|          | 1                       | 2     | 3     | 4     | 5     | 6     |
| 64X64    | 1,000                   | 1,335 | 1,178 | 1,074 | 1,019 | 0,940 |
| 128X128  | 1,000                   | 1,807 | 2,514 | 2,617 | 2,730 | 3,077 |
| 256X256  | 1,000                   | 1,954 | 2,876 | 3,598 | 4,455 | 5,258 |
| 512X512  | 1,000                   | 1,976 | 2,925 | 3,906 | 4,806 | 5,668 |

Tabela 6.3 - IBM 3090/600VF - Ganhos obtidos com a forma 1a.

Os ganhos obtidos na forma 1a são similares aos da forma 1, com variação máxima de 5%, exceto para dois processadores. Neste caso, os ganhos não apresentam a anomalia observada na forma 1.

| Dimensao | numero de processadores |       |       |       |       |       |
|----------|-------------------------|-------|-------|-------|-------|-------|
|          | 1                       | 2     | 3     | 4     | 5     | 6     |
| 64X64    | 1,000                   | 1,363 | 1,219 | 1,195 | 1,057 | 0,923 |
| 128X128  | 1,000                   | 1,826 | 2,518 | 3,036 | 3,279 | 3,651 |
| 256X256  | 1,000                   | 1,973 | 2,877 | 3,776 | 4,561 | 5,250 |
| 512X512  | 1,000                   | 1,997 | 3,005 | 3,986 | 4,954 | 5,687 |

Tabela 6.4 - IBM 3090/600VF - Ganhos obtidos com a forma 1b.

Os ganhos na forma 1b são muito similares aos ganhos na forma 1a, exceto no caso de 128X128 com 4, 5 e 6 processadores, onde a forma 1b apresenta resultados até 20% superiores.

A comparação dos ganhos nas três variantes mostra uma diferença inferior a 5%, ressalvadas as regiões de exceção já notadas, demonstrando uma independência do ganho em relação às variantes da primeira forma de paralelização.

A explicação das regiões de exceção apontadas acima exigiria mais experimentos, que pelas razões já citadas foram inviáveis.

#### 6.1.4 - Apresentação e Análise dos Resultados para o IBM 9021

A seguir estão as tabelas de tempos de execução e ganhos com a forma 1:

| Dimensao | numero de processadores |        |        |        |
|----------|-------------------------|--------|--------|--------|
|          | 1                       | 2      | 3      | 4      |
| 64X64    | 0,526                   | 0,550  | 0,682  | 0,815  |
| 128X128  | 3,318                   | 2,028  | 1,923  | 1,972  |
| 256X256  | 25,598                  | 13,060 | 9,585  | 7,994  |
| 512X512  | 188,709                 | 95,538 | 65,040 | 50,424 |

Tabela 6.5 - IBM 9021/820 - Tempos de execução (s) com a forma 1.

## numero de processadores

1            2            3            4

| Dimensao |       |       |       |       |
|----------|-------|-------|-------|-------|
| 64X64    | 1,000 | 0,956 | 0,771 | 0,645 |
| 128X128  | 1,000 | 1,636 | 1,725 | 1,683 |
| 256X256  | 1,000 | 1,960 | 2,671 | 3,202 |
| 512X512  | 1,000 | 1,975 | 2,901 | 3,742 |

Tabela 6.6 - IBM 9021/820 - Ganhos obtidos com a forma 1.

A análise destas tabelas mostra que o custo do paralelismo que já era alto no IBM 3090, é ainda maior no IBM 9021. Observe-se que ganhos bons (acima de 80% do ótimo) para o número máximo de processadores, que eram atingidos com a dimensão 256X256 no IBM 3090, só são atingidos no IBM 9021 com a dimensão de 512X512. Uma comparação entre os tempos de execução dos maiores problemas nos IBM 3090 e 9021 mostra que a CPU deste último é quase três vezes mais rápida que a do 3090, para este tipo de problema.

Os ganhos mostram-se crescentes com o tamanho do problema, com exceção de pequenos problemas ( $< 256 \times 256$ ) quando o número de processadores passa de 3 para 4. Ganhos próximos do ótimo foram obtidos para os maiores problemas experimentados.

Note-se que as anomalias apresentadas pela forma 1 no IBM 3090 com dois processadores não se repetiram no IBM 9021.

Para efeito de comparação, são apresentados a seguir os ganhos com as variantes 1a e 1b.

numero de processadores

1            2            3            4

| Dimensao |       |       |       |       |
|----------|-------|-------|-------|-------|
| 64X64    | 1,000 | 1,004 | 0,612 | 0,678 |
| 128X128  | 1,000 | 1,601 | 2,052 | 1,996 |
| 256X256  | 1,000 | 1,917 | 2,710 | 3,244 |
| 512X512  | 1,000 | 1,984 | 2,936 | 3,734 |

Tabela 6.7 - IBM 9021/820 - Ganhos obtidos com a forma 1a.

numero de processadores

1            2            3            4

| Dimensao |       |       |       |       |
|----------|-------|-------|-------|-------|
| 64X64    | 1,000 | 0,967 | 0,560 | 0,671 |
| 128X128  | 1,000 | 1,636 | 2,052 | 2,143 |
| 256X256  | 1,000 | 1,829 | 2,760 | 3,423 |
| 512X512  | 1,000 | 1,976 | 2,919 | 3,706 |

Tabela 6.8 - IBM 9021/820 - Ganhos obtidos com a forma 1b.

Observa-se que a forma 1a também apresenta ganhos decrescentes (128X128 de 3 para 4 processadores) como os apresentados na forma 1 já comentada. A explicação destas discrepâncias exigiria mais experimentos que não puderam ser realizados.

A variação máxima dos ganhos entre as três variantes é menor que 5%, ressalvados os casos de 128X128 e 256X256 com 3 e 4 processadores, mostrando a independência do ganho com relação às diversas implementações da primeira forma de paralelização.

## 6.2 - Segunda Forma de Paralelização

### 6.2.1 - Descrição da Implementação

Enquanto na primeira forma de paralelização havia uma divisão das freqüências entre os processadores, nesta segunda forma, como na segunda forma para as máquinas hipercúbicas, as profundidades é que são divididas entre os processadores. Assim, considerando  $p$  processadores, cada processador  $i$  será responsável pelo processamento de todas as freqüências nas profundidades  $i, i + p, i + 2p, \dots$ , para  $i = 1$  até  $p$ .

A geração da seção de entrada e a execução da FFT são realizadas pelo primeiro processador. A partir daí forma-se um *pipeline* em anel, onde o primeiro nó processa todas as freqüências da primeira profundidade, o próximo nó do anel executa o processamento de todas as freqüências na profundidade seguinte e assim sucessivamente.

O *pipeline* é implementado através de um vetor que para cada freqüência informa a última profundidade em que ela foi processada e assim, libera o seu processamento na próxima profundidade.

Uma diferença fundamental em relação à implementação feita nos hipercubos é que pelo fato da memória ser central, não é mais necessária a transmissão dos dados de um processador para o outro no anel.

Outra característica interessante desta implementação é que como cada processador é o responsável exclusivo por uma dada profundidade, até mesmo a acumulação final dos resultados pode ser feita simultaneamente por todos os processadores sem necessidade de criar-se uma região crítica para isto. Entretanto, o acesso ao vetor que controla o *pipeline* constitui uma região crítica que é controlada por um semáforo para cada freqüência.

Infelizmente, a implementação apresentou comportamento anômalo para problemas de tamanho superior a 128X128. Sob circunstâncias não determinadas a execução do programa não termina. O problema é intermitente; execuções consecutivas podem terminar ou não. Apesar de múltiplos esforços, não se determinou a causa desta anomalia.

### 6.2.2 - Apresentação e Análise dos Resultados para o IBM 9021

As observações feitas para a primeira forma quanto aos tempos medidos, forma de cálculo dos ganhos e tabelas de apresentação dos resultados, são válidas também para esta segunda forma de paralelização. O número muito reduzido de experimentos deve-se ao comportamento anômalo já comentado.

A seguir estão as tabelas de tempos de execução e ganhos medidos nos experimentos.

numero de processadores  
1            2            3            4

| Dimensao |       |       |       |       |
|----------|-------|-------|-------|-------|
| 64X64    | 0,555 | 0,500 | 0,587 | 0,847 |
| 128X128  | 3,320 | 1,926 | 1,657 | 1,424 |

Tabela 6.9 - IBM 9021/820 - Tempos de execução (s) com a forma 2.

numero de processadores  
1            2            3            4

| Dimensao |       |       |       |       |
|----------|-------|-------|-------|-------|
| 64X64    | 1,000 | 1,110 | 0,945 | 0,655 |
| 128X128  | 1,000 | 1,724 | 2,004 | 2,331 |

Tabela 6.10 - IBM 9021/820 - Ganhos obtidos com a forma 2.

A observação das tabelas mostra um ganho crescente com o aumento do tamanho do problema. Os poucos resultados conseguidos indicam uma tendência de crescimento de ganhos melhor do que a apresentada pela forma 1. Lamentavelmente, por impossibilidade de maiores experimentos, esta tendência não pôde ser confirmada.

## ***6.3 - Terceira Forma de Paralelização***

### **6.3.1 - Descrição da Implementação**

Esta terceira forma de paralelização é uma generalização da segunda forma, explorando alocação dinâmica de processadores. Nesta forma, qualquer nó pode processar qualquer freqüência na profundidade disponível para esta freqüência. O controle dos pares (freqüência, profundidade) disponíveis é feito através de uma lista circular cujo acesso é uma região crítica controlada por um único semáforo.

Um inconveniente trazido por este escalonamento dinâmico é o retorno da região crítica para acumular os resultados, já que vários nós podem processar simultaneamente a mesma profundidade em freqüências distintas. O acesso a esta região crítica é controlado por um semáforo para cada profundidade.

O escalonamento dinâmico funciona do seguinte modo: o primeiro processador gera a seção de entrada, executa a FFT e inicializa a lista circular, colocando todas as freqüências como disponíveis para processamento na primeira profundidade. A partir daí, cada nó tenta o acesso à lista circular, obtém um par (freqüência, profundidade) da lista, processa-o e ao fim do processamento, se não for a última profundidade, insere o par (freqüência, profundidade + 1) no final da lista, liberando aquela freqüência para o processamento na próxima profundidade, e assim sucessivamente.

### **6.3.2 - Apresentação e Análise dos Resultados para o IBM 9021**

As observações feitas para a primeira forma quanto aos tempos medidos, forma de cálculo dos ganhos, tabelas de apresentação dos resultados e quantidade de experimentos, são válidas também para esta terceira forma de paralelização.

A seguir estão as tabelas de tempos de execução e ganhos medidos nos experimentos.

| Dimensao | numero de processadores |         |        |        |
|----------|-------------------------|---------|--------|--------|
|          | 1                       | 2       | 3      | 4      |
| 64X64    | 0,564                   | 0,534   | 0,650  | 0,830  |
| 128X128  | 3,466                   | 2,055   | 1,739  | 1,760  |
| 256X256  | 25,861                  | 13,416  | 9,175  | 7,570  |
| 512X512  | 189,402                 | 163,792 | 66,129 | 51,411 |

Tabela 6.11 - IBM 9021/820 - Tempos de execução (s) com a forma 3.

| Dimensao | numero de processadores |       |       |       |
|----------|-------------------------|-------|-------|-------|
|          | 1                       | 2     | 3     | 4     |
| 64X64    | 1,000                   | 1,056 | 0,868 | 0,680 |
| 128X128  | 1,000                   | 1,687 | 1,993 | 1,969 |
| 256X256  | 1,000                   | 1,928 | 2,819 | 3,416 |
| 512X512  | 1,000                   | 1,156 | 2,864 | 3,684 |

Tabela 6.12 - IBM 9021/820 - Ganhos obtidos com a forma 3.

A observação das tabelas indica ganhos crescentes com o tamanho do problema e próximos do ótimo para os maiores problemas experimentados, apesar de ligeiramente inferiores aos das formas 1 e 2, nas medidas comuns. Nota-se um valor discrepante para o ganho com dois processadores em problemas de dimensão 512X512, cuja explicação exigiria mais experimentos.



## 6.4 - *Conclusões*

Os ganhos próximos do ótimo observados comprovam a viabilidade do uso desta classe de máquinas na migração sísmica, em ambiente paralelo. Um problema de implementação não permitiu maiores experimentos com a forma 2, que apresentava os ganhos mais promissores.

As duas máquinas apresentaram tempos de execução inexplicavelmente altos para algumas combinações de tamanho do problema com o número de processadores (quase sempre 2). Teria sido muito interessante executar uma maior variedade de experimentos para explicar tal comportamento.

Os ganhos mostraram-se independentes da forma de paralelização, ressalvadas algumas regiões bem determinadas. Ficou claro também o alto custo do paralelismo nas duas máquinas, o que sugere o uso de paralelismo só para grandes problemas, que é o caso dos problemas reais de migração sísmica.

A CPU do IBM 9021 mostrou-se quase três vezes mais rápida do que a do IBM 3090, nos maiores problemas experimentados.

## Capítulo 7 - Experimentos e Resultados em Rede de Estações

Este capítulo descreve a implementação e analisa os resultados das duas formas de paralelização da migração  $\omega$ -x em uma rede de estações IBM RS-6000, funcionando como uma máquina paralela através do PVM (Parallel Virtual Machine) [Sunderam90].

A grande disponibilidade destas estações no período noturno permitiu a realização de grande quantidade de experimentos e, conseqüentemente, a análise detalhada dos resultados.

Há 19 estações RS-6000 disponíveis: 2 modelo 530H, 2 modelo 530, 8 modelo 320H e 7 modelo 320. Estas estações estão ligadas em redes Ethernet e Token-Ring, formando 3 sub-redes. As máquinas de uma mesma sub-rede comunicam-se num tempo  $t_s$ , enquanto a comunicação entre máquinas de sub-redes distintas é feita em  $2.5t_s$  ou  $4.5t_s$ , dependendo de quais sub-redes estão envolvidas. Para evitar heterogeneidade, optou-se por trabalhar com uma sub-rede de 6 estações 320H com tempo de comunicação  $t_s$ .

As formas de paralelização experimentadas foram a de frequências esparsas e a de *pipeline*, as mesmas já testadas em outros equipamentos. Ao se executar as medidas seqüenciais para a base de cálculo dos ganhos, identificou-se que os tempos de execução para problemas com 128 traços divergem da análise de complexidade, conforme apresentado na tabela a seguir:

|                           | 127<br>tracos | 128<br>tracos | 129<br>tracos |
|---------------------------|---------------|---------------|---------------|
| <b>Numero de Amostras</b> |               |               |               |
| 64                        | 2,89          | 3,62          | 3,04          |
| 128                       | 11,69         | 14,66         | 12,16         |
| 256                       | 46,97         | 59,47         | 48,88         |
| 512                       | 187,37        | 236,75        | 204,21        |

Tabela 7.1 - RS-6000 320H - Tempos de execução (s) seqüencial com 127, 128 e 129 traços.

A observação da tabela 7.1 confirma tempos de execução para 128 traços não compatíveis com os tempos medidos para 127 e 129 traços. Esta discrepância provavelmente deve-se à arquitetura de memória do equipamento. Para não contaminar as análises que serão feitas, em vez de medidas com 128 traços serão utilizadas medidas com 127 traços.

Segue abaixo a tabela dos tempos seqüenciais que serão usados no cálculo dos ganhos.

|                           | 64<br>tracos | 127<br>tracos | 256<br>tracos | 512<br>tracos |
|---------------------------|--------------|---------------|---------------|---------------|
| <b>Numero de Amostras</b> |              |               |               |               |
| 64                        | 1,45         | 2,89          | 5,64          | 11,17         |
| 128                       | 5,88         | 11,69         | 23,00         | 44,95         |
| 256                       | 23,64        | 46,97         | 93,00         | 181,26        |
| 512                       | 94,99        | 187,37        | 376,01        | 736,49        |

Tabela 7.2 - RS-6000 320H - Tempos de execução (s) seqüencial

Observa-se claramente o comportamento linear do tempo de execução com o aumento de traços, além do comportamento quadrático com o aumento de amostras. Note-se que tanto esta tabela quanto as subseqüentes neste capítulo, foram geradas com programas compilados com opção de otimização. Já as tabelas apresentadas no capítulo 4 (análise de complexidade) foram geradas com programas não otimizados. Ao confrontar estas tabelas nota-se a grande importância da opção de otimização.

## ***7.1 - Primeira Forma de Paralelização***

### **7.1.1 - Descrição da Implementação**

Esta forma de paralelização é a mesma primeira forma já descrita para as máquinas hipercúbicas. Fundamentalmente, o que muda são as primitivas de comunicação entre os processadores. Lembre-se que nesta forma as freqüências são divididas entre os processadores que as trabalham em todas as profundidades. Optou-se pela divisão das freqüências de modo esparso.

### **7.1.2 - Descrição dos Experimentos**

Foram executadas medidas variando-se independentemente o número de traços e o de amostras. Pôde-se ainda experimentar vários tamanhos para as mensagens entre os processadores, adotando-se o tamanho de 4096 bytes que apresentou o melhor resultado.

Cada experimento realizado é reportado por duas tabelas. A primeira contém os tempos de execução (tempo de parede), em segundos, medidos no primeiro processador. Este foi escolhido por ser, por força da implementação, o primeiro nó a iniciar a computação e o último nó a terminá-la. A segunda tabela contém os ganhos, medidos pelo quociente entre o tempo de execução do programa seqüencial executado em uma estação e o tempo de execução da versão paralela com  $p$  estações.

Foram coletados os tempos de execução para 1, 4 e 6 estações, com todas as combinações possíveis de 64, 127, 256 e 512 traços com 64, 128, 256 e 512 amostras.

### 7.1.3 - Apresentação e Análise de Resultados na Rede RS-6000/PVM

Apresenta-se, a seguir, o tempo de execução e o ganho para quatro estações.

|                    | 64<br>tracos | 127<br>tracos | 256<br>tracos | 512<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|
| Numero de Amostras |              |               |               |               |
| 64                 | 0,75         | 1,22          | 2,23          | 4,23          |
| 128                | 2,24         | 3,82          | 7,26          | 13,69         |
| 256                | 7,10         | 13,35         | 26,08         | 50,25         |
| 512                | 26,19        | 50,24         | 99,70         | 194,16        |

Tabela 7.3 - RS-6000/PVM - Tempos de execução (s) com 4 estações.

|                    | 64<br>tracos | 127<br>tracos | 256<br>tracos | 512<br>tracos |
|--------------------|--------------|---------------|---------------|---------------|
| Numero de Amostras |              |               |               |               |
| 64                 | 1,93         | 2,37          | 2,53          | 2,64          |
| 128                | 2,63         | 3,06          | 3,17          | 3,28          |
| 256                | 3,33         | 3,52          | 3,57          | 3,61          |
| 512                | 3,63         | 3,73          | 3,77          | 3,79          |

Tabela 7.4 - RS-6000/PVM - Ganhos com 4 estações.

Observam-se ganhos crescentes com o tamanho do problema e próximos ao ótimo para as maiores dimensões testadas. Nota-se também uma variação dos ganhos com a variação de  $n_x$  não prevista na análise de complexidade realizada no capítulo 4.

Apresenta-se, a seguir, o tempo de execução e o ganho para seis estações.

|                           | 64<br>tracos | 127<br>tracos | 256<br>tracos | 512<br>tracos |
|---------------------------|--------------|---------------|---------------|---------------|
| <b>Numero de Amostras</b> |              |               |               |               |
| 64                        | 0,73         | 1,19          | 2,14          | 3,96          |
| 128                       | 1,92         | 3,27          | 6,00          | 11,37         |
| 256                       | 5,63         | 10,22         | 20,09         | 38,68         |
| 512                       | 19,09        | 36,17         | 70,97         | 138,00        |

Tabela 7.5 - RS-6000/PVM - Tempos de execução (s) com 6 estações.

|                           | 64<br>tracos | 127<br>tracos | 256<br>tracos | 512<br>tracos |
|---------------------------|--------------|---------------|---------------|---------------|
| <b>Numero de Amostras</b> |              |               |               |               |
| 64                        | 1,99         | 2,43          | 2,64          | 2,82          |
| 128                       | 3,06         | 3,57          | 3,83          | 3,95          |
| 256                       | 4,20         | 4,60          | 4,63          | 4,69          |
| 512                       | 4,98         | 5,18          | 5,30          | 5,34          |

Tabela 7.6 - RS-6000/PVM - Ganhos com 6 estações.

Do mesmo modo que para 4 estações, as tabelas mostram ganhos crescentes com o tamanho do problema e variação dos ganhos com  $n_x$ . Note-se, ainda, que a eficiência com 6 estações é inferior à obtida para 4 estações, por exemplo, 89% versus 94% para problemas 512X512.

Para entender o que ocorre durante a execução, foi criada uma versão instrumentada do programa, medindo separadamente os tempos de comunicação inicial, os tempos de processamento e os tempos de comunicação de resultados, para cada uma das estações envolvidas na computação. A tabela a seguir

expressa, percentualmente, a importância relativa de cada um destes tempos na computação.

|         | TEMPOS       | 4 PROC | 6 PROC |
|---------|--------------|--------|--------|
| 64X64   | Com. Inicial | 21,13  | 23,40  |
|         | Process.     | 72,08  | 59,44  |
|         | Com. Final   | 6,77   | 17,14  |
| 128X128 | Com. Inicial | 7,16   | 9,08   |
|         | Process.     | 87,62  | 75,71  |
|         | Com. Final   | 5,21   | 15,20  |
| 256X256 | Com. Inicial | 2,21   | 3,26   |
|         | Process.     | 94,59  | 89,31  |
|         | Com. Final   | 3,19   | 7,41   |
| 512X512 | Com. Inicial | 0,72   | 1,15   |
|         | Process.     | 97,74  | 95,08  |
|         | Com. Final   | 1,53   | 3,75   |

Tabela 7.7 - RS-6000/PVM - Composição percentual do tempo de execução na versão instrumentada com 4 e 6 estações.

Pela observação da tabela fica clara a importância da comunicação nos problemas de pequena dimensão. Esta importância diminui com o crescimento do problema e aumenta com o número de processadores. Este comportamento justifica o aumento do ganho com o tamanho dos problemas (a comunicação perde seu peso relativo no tempo de execução) e a perda da eficiência com o aumento do número de processadores (a eficiência é definida como o ganho dividido pelo número de processadores).

#### 7.1.4 - Expressão Analítica para o Tempo de Execução

Para detalhar as conclusões sobre os comportamentos destacados é necessário analisar a expressão do tempo de execução. Esta expressão é obtida por um refinamento da expressão usada na análise da complexidade, com ênfase no custo de comunicação. O tempo de execução pode ser escrito como:

$$t_{exec} = t_{com\_ini} + t_{proc} + t_{com\_fin}$$

onde  $t_{com\_ini}$  representa o tempo da comunicação inicial,  $t_{proc}$  representa o tempo de processamento e  $t_{com\_fin}$  representa o tempo da comunicação final.

O tempo de processamento,  $t_{proc}$ , é descrito pela expressão utilizada na análise de complexidade:

$$t_{proc} = \frac{k_0 n_t^2 n_x}{p},$$

onde  $n_t$  e  $n_x$  são as dimensões do problema e  $p$  é o número de estações.

A instrumentação demonstra que a expressão do tempo de comunicação inicial utilizada na análise de complexidade não modela adequadamente problemas de tamanho pequeno, por desprezar a latência da comunicação. Introduzindo este fator, modelamos o tempo de uma transmissão de  $n$  bytes por

$$t(n) = c_0 + c_1 n$$

onde  $c_0$  é a latência da transmissão e  $c_1$  é o custo da transmissão por byte. Como a transmissão inicial é composta por  $n_t/2$  mensagens, cada uma com  $8n_x$  bytes, o tempo de comunicação inicial passa a ser expresso por

$$t_{com\_ini} = \frac{n_t}{2} t(8n_x) = \frac{c_0}{2} n_t + 4c_1 n_t n_x$$

O tempo da comunicação final mantém expressão equivalente à utilizada na análise de complexidade:

$$t_{com\_fim} = k_3(p - 1)n_t n_x$$

Assim, a expressão do tempo de execução passa a ser:

$$t_{exec} = 0.5c_0 n_t + 4c_1 n_t n_x + \frac{k_0 n_t^2 n_x}{p} + k_3(p - 1)n_t n_x$$



Conseqüentemente, o ganho será

$$S = \frac{k_0 n_t^2 n_x}{0.5c_0 n_t + 4c_1 n_t n_x + \frac{k_0 n_t^2 n_x}{p} + k_3(p-1)n_t n_x}$$

Observe que a latência é a única parcela do tempo de execução que independe de  $n_x$ . Se o valor numérico de  $c_0$  for considerável, a latência não poderá ser desprezada, fazendo com que o ganho dependa de  $n_x$ . Observe ainda que, com o aumento do problema, esta parcela tende a diminuir de importância, por variar linearmente com o tamanho do problema, enquanto as demais parcelas variam quadraticamente e cubicamente. Logo, a dependência do ganho com  $n_x$  é maior nos pequenos problemas e se dilui à medida que o problema aumenta.

Pode-se ver também que, fixo o número  $p$  de processadores, o ganho cresce com o tamanho do problema, pois a parcela do processamento cresce cubicamente enquanto as parcelas de comunicação crescem quadraticamente.

A partir do ganho, obtém-se a eficiência da computação, dada por

$$E = \frac{S}{p} = \frac{k_0 n_t^2 n_x}{0.5c_0 n_t p + 4c_1 n_t n_x p + k_0 n_t^2 n_x + k_3 p(p-1)n_t n_x}$$

Observe que, para um problema de tamanho fixo, a eficiência diminui com o aumento de  $p$ . Este comportamento deve-se à dependência das comunicações com relação a  $p$ .

Desta forma, as três observações realizadas sobre os dados experimentais são explicáveis pela análise da expressão do tempo de execução. Tanto a dependência do ganho com  $n_x$ , quanto o decréscimo da eficiência com o aumento de  $p$  devem-se aos custos de comunicação. Já o ganho cresce com o tamanho do problema porque o custo de processamento cresce cubicamente e os custos de comunicação crescem quadraticamente.

### 7.1.5 - Calibração e Validação

Visando avaliar ainda mais detalhadamente a importância dos tempos de comunicação no processamento, realizaram-se uma série de experimentos para a determinação numérica de todas as constantes envolvidas.

A constante  $k_0$  foi estimada a partir dos tempos de execução da versão seqüencial do programa, resultando em  $k_0 = 5.54 \times 10^{-6}$

As constantes  $c_0$  e  $c_1$  foram estimadas por experimentos de transmissão e recepção de dados pela rede, independentes do programa de migração. Obtiveram-se os valores  $c_0 = 3.90 \times 10^{-3}$  e  $c_1 = 5.86 \times 10^{-7}$ . Note-se a acentuada importância da latência no custo de transmissão.

A constante  $k_3$  foi estimada medindo-se o trecho da migração correspondente, pois ela engloba a recepção de dados e a acumulação final. Obteve-se  $k_3 = 3.83 \times 10^{-6}$ . Substituindo os valores das constantes na expressão analítica, obtém-se o tempo de execução, em microssegundos:

$$t_{exec} = 1.95 \times 10^3 n_t + 2.34 n_t n_x + \frac{5.54 \times n_t^2 n_x}{p} + 3.83(p - 1)n_t n_x$$

A tabela a seguir contrasta os tempos de comunicação e de computação medidos com os previstos pela expressão acima.

|         | TEMPOS       | 4<br>processadores |          | 6<br>processadores |          |
|---------|--------------|--------------------|----------|--------------------|----------|
|         |              | MEDIDO             | PREVISTO | MEDIDO             | PREVISTO |
| 64X64   | Com. Inicial | 0,13               | 0,13     | 0,15               | 0,13     |
|         | Process.     | 0,45               | 0,36     | 0,38               | 0,24     |
|         | Com. Final   | 0,04               | 0,04     | 0,11               | 0,07     |
| 128X128 | Com. Inicial | 0,26               | 0,28     | 0,28               | 0,28     |
|         | Process.     | 3,19               | 2,88     | 2,35               | 1,92     |
|         | Com. Final   | 0,19               | 0,18     | 0,47               | 0,31     |
| 256X256 | Com. Inicial | 0,57               | 0,65     | 0,64               | 0,65     |
|         | Process.     | 24,43              | 23,85    | 17,59              | 15,50    |
|         | Com. Final   | 0,82               | 0,75     | 1,46               | 1,25     |
| 512X512 | Com. Inicial | 1,40               | 1,61     | 1,57               | 1,61     |
|         | Process.     | 188,00             | 186,02   | 129,71             | 124,01   |
|         | Com. Final   | 2,94               | 3,01     | 5,12               | 5,03     |

Tabela 7.8 - Tempos de execução (s) medidos e previstos para 4 e 6 estações.

Há duas observações a fazer. Primeiro, a boa aderência dos tempos de comunicação. Segundo, a discrepância entre os tempos de processamento. Os gráficos a seguir mostram o comportamento dos tempos de processamento de uma profundidade no processador 0, medidos como tempo de parede e tempo de CPU, em experimento de tamanho 512X512, com 4 estações. O tempo de processamento médio é de 0,367198 s em tempo de parede e de 0,362871 s em tempo de cpu.

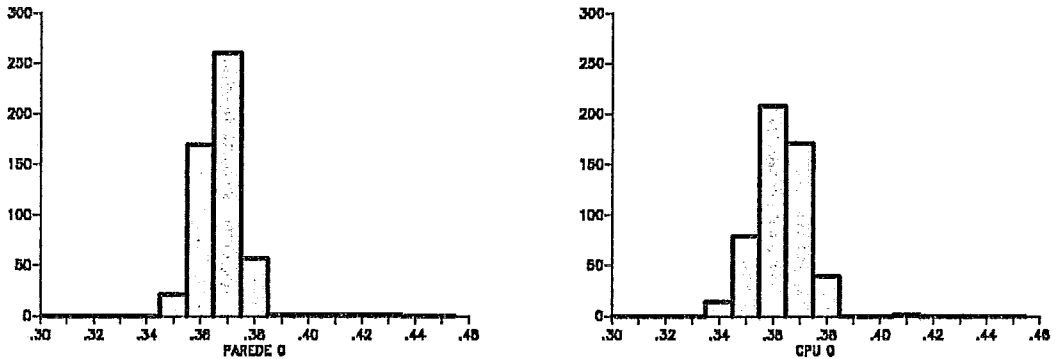


Tabela 7.9 - RS-6000/PVM - Distribuição dos tempos de processamento de uma profundidade, no processador 0, problema 512X512, com 4 estações (tempo de parede e de CPU, em segundos).

Observa-se que o tempo de processamento medido como tempo de parede apresenta uma média superior à apresentada pelo tempo de CPU. Esta diferença deve ser creditada a interrupções sofridas pelo processador durante o processamento, seja pelo PVM e/ou pelo próprio UNIX, não deterministicamente. O produto do tempo médio de CPU mostrado, multiplicado pelo número de profundidades processadas (512) dá exatamente o tempo de execução previsto. As interrupções explicam assim as diferenças entre os tempos de processamento medidos e previstos na tabela 7.8.

### 7.1.6 - Extremos

O fato da expressão analítica modelar bem o problema permite que se façam projeções do comportamento do ganho com a variação das dimensões do problema e do número de processadores na rede. Supondo  $n_x = n_t = n$ , pode-se escrever o ganho  $S$  como uma função de  $n$  e  $p$ :

$$S(n,p) = \frac{pn^2}{p\left(\frac{c_0}{2k_0} + \frac{4c_1}{k_0}n\right) + n^2 + \frac{k_3}{k_0}p(p-1)n}$$

Assim, fixando-se o número de processadores em  $p$  e fazendo  $n \rightarrow \infty$ , tem-se que  $S(n,p) \rightarrow p$ . Então, fixo um número de processadores existe sempre um problema suficientemente grande para o qual o ganho será ótimo.

Por outro lado, fixando-se o tamanho  $n$  do problema e fazendo o número de processadores  $p \rightarrow \infty$ , tem-se que  $S(n,p) \rightarrow 0$ . Logo, para um problema de tamanho fixo, aumentar o número de processadores acima de um certo valor *aumenta* o tempo de processamento. Para se investigar a existência de um ponto de máximo, faz-se:

$$\frac{\partial S}{\partial p} = 0 \Rightarrow p = \sqrt{\frac{k_0}{k_3}n}$$

Então, para um determinado problema de tamanho  $n$  sempre existe um número de processadores  $p$  suficientemente grande para o qual o ganho é máximo. Notar que o ganho máximo não significa ganho ótimo. Utilizando os valores numéricos das constantes, obtém-se o número de processadores e o ganho máximo para cada tamanho de problema. A tabela a seguir contém estes valores.

| n     | p  | S  |
|-------|----|----|
| 64    | 9  | 3  |
| 128   | 13 | 6  |
| 256   | 19 | 9  |
| 512   | 27 | 13 |
| 1.500 | 46 | 23 |

Tabela 7.10 - Número de processadores ( $p$ ) que dá ganho máximo ( $S$ ) em função do tamanho do problema ( $n$ )

Observa-se, portanto, a limitação desta forma de paralelização na rede de estações. Deve-se ter claro que, fundamentalmente, o fator responsável por esta limitação é a comunicação entre os processadores. O custo de comunicação apresentado pela rede de estações sob o PVM é que determina qual será este limite. Desse modo, uma diminuição dos custos de comunicação na rede tem reflexos diretos no aumento do valor limite do ganho.

## ***7.2 - Segunda Forma de Paralelização***

### **7.2.1 - Descrição da Implementação**

Esta forma de paralelização é a mesma segunda forma já descrita para as máquinas hipercúbicas. Fundamentalmente, o que muda são as primitivas de comunicação entre os processadores. Lembre-se que nesta forma as profundidades são divididas entre os processadores que as trabalham em todas as freqüências, pela montagem de um *pipeline* em forma de anel.

### **7.2.2 - Descrição dos Experimentos**

A descrição feita para a primeira forma de paralelização (7.1.2), também é válida para esta segunda forma.

### **7.2.3 - Apresentação e Análise de Resultados na Rede RS-6000/PVM**

Apresenta-se, a seguir, o tempo de execução e o ganho para quatro estações.

|                           | 64<br>tracos | 127<br>tracos | 256<br>tracos | 512<br>tracos |
|---------------------------|--------------|---------------|---------------|---------------|
| <b>Numero de Amostras</b> |              |               |               |               |
| 64                        | 1,63         | 3,02          | 5,70          | 11,46         |
| 128                       | 5,82         | 11,43         | 22,52         | 43,75         |
| 256                       | 22,80        | 43,71         | 87,82         | 171,84        |
| 512                       | 87,23        | 173,27        | 347,05        | 698,62        |

Tabela 7.11 - RS-6000/PVM - Tempos de execução (s) com 4 estações.

|                           | 64<br>tracos | 127<br>tracos | 256<br>tracos | 512<br>tracos |
|---------------------------|--------------|---------------|---------------|---------------|
| <b>Numero de Amostras</b> |              |               |               |               |
| 64                        | 1,06         | 0,99          | 1,01          | 1,03          |
| 128                       | 1,02         | 1,04          | 1,03          | 1,08          |
| 256                       | 1,07         | 1,11          | 1,07          | 1,05          |
| 512                       | 1,09         | 1,17          | 1,09          | 1,05          |

Tabela 7.12 - RS-6000/PVM - Ganhos com 4 estações.

Os ganhos apresentados nesta segunda forma com 4 processadores são pífios.

Apresenta-se, a seguir, o tempo e o ganho para seis estações.

|                           | 64<br>tracos | 127<br>tracos | 256<br>tracos | 512<br>tracos |
|---------------------------|--------------|---------------|---------------|---------------|
| <b>Numero de Amostras</b> |              |               |               |               |
| 64                        | 1,61         | 2,33          | 4,43          | 8,81          |
| 128                       | 4,34         | 8,36          | 16,58         | 32,24         |
| 256                       | 15,98        | 31,82         | 61,75         | 122,74        |
| 512                       | 62,06        | 121,83        | 244,61        | 496,42        |

Tabela 7.13 - RS-6000/PVM - Tempos de execução (s) com 6 estações.

|                           | 64<br>tracos | 127<br>tracos | 256<br>tracos | 512<br>tracos |
|---------------------------|--------------|---------------|---------------|---------------|
| <b>Numero de Amostras</b> |              |               |               |               |
| 64                        | 1,07         | 1,28          | 1,30          | 1,35          |
| 128                       | 1,36         | 1,42          | 1,40          | 1,47          |
| 256                       | 1,53         | 1,52          | 1,52          | 1,48          |
| 512                       | 1,54         | 1,67          | 1,54          | 1,48          |

Tabela 7.14 - RS-6000/PVM - Ganhos com 6 estações.

Apesar de apresentarem um aumento, os ganhos obtidos com 6 processadores nesta segunda forma de paralelização ainda estão longe do que vinha sendo obtido. Para explicar estes resultados criou-se uma versão instrumentada do programa, para medir separadamente os tempos de envio, de espera e de recebimento de mensagens intermediárias, de processamento e de comunicação final.

| Dimensao | Env   | Esp   | Rec   | Proc  | C.Fin |
|----------|-------|-------|-------|-------|-------|
| 64X64    | 26,94 | 20,39 | 18,29 | 30,20 | 4,14  |
| 127X128  | 23,59 | 20,75 | 1,90  | 48,06 | 5,60  |
| 256X256  | 24,99 | 22,45 | 2,03  | 49,05 | 1,45  |
| 512X512  | 25,35 | 24,43 | 2,83  | 46,91 | 0,45  |

Tabela 7.15 - RS-6000/PVM - Composição percentual do tempo de execução da versão *pipeline* com 4 estações.

A tabela mostra que há um custo total de comunicação muito alto, nunca inferior a 50%. Já se sabia que a versão *pipeline* tem uma quantidade de comunicação superior à da primeira forma, mas o que se esperava é que houvesse uma superposição da comunicação com o processamento, não afetando desfavoravelmente o ganho. Entretanto, não é isto o que ocorre. O processador gasta mais que 20% do seu tempo só *esperando* comunicação. Assim, a comunicação provoca o efeito de bolhas no *pipeline*, comprometendo inapelavelmente o desempenho. A versão então, mostra-se inviável nesta rede de estações sob o PVM.

### 7.3 - Conclusões

A primeira forma de paralelização apresentou ganhos crescentes com o tamanho do problema e próximos ao ótimo para as maiores dimensões testadas. A ocorrência de variações de ganho com  $n_x$  não previstas na análise de complexidade, motivou uma reanálise que demonstrou a importância da latência de comunicação nesta rede.

A segunda forma mostrou-se inviável na rede sob o PVM, pela inexpressividade dos ganhos apresentados.



A expressão analítica obtida para o tempo de execução, na primeira forma de paralelização, permitiu projeções para ganhos máximos e o mapeamento das restrições desta forma de paralelismo.

Finalmente, ficou patente a importância do custo de comunicação no desempenho da migração  $\omega$ -x na rede de estações sob o PVM, nas formas testadas.

## Capítulo 8 - Conclusões e Futuros Trabalhos

Ao final deste trabalho, depois de muitos experimentos e análises, a principal conclusão, indicada em vários capítulos anteriores, é a viabilidade da utilização do processamento paralelo na Migração Sísmica. Assim, a pergunta da introdução que é a motivação básica desta tese, sobre a viabilidade do paralelismo no Processamento Sísmico tem uma resposta afirmativa respaldada na grande quantidade de experimentos realizados em máquinas hipercúbicas de memória distribuída, em máquinas de memória central e em rede de estações sob o PVM.

A Migração Sísmica mostrou-se de paralelismo fácil, em várias formas e em vários equipamentos. O desempenho conseguido foi muito bom, com poucas exceções. A faixa de tamanho de problemas e números de processadores testados não foi suficiente para o aparecimento dos valores limites apontados pela análise projetiva do capítulo 7. O problema imposto pelo custo da comunicação na rede de estações, apesar de só ter sido identificado na rede, é um problema latente nos outros equipamentos, dependendo apenas da combinação do tamanho do problema com o número de processadores.

Pelas análises feitas, concluiu-se que:

- Fixado o número de processadores, existe sempre um problema suficientemente grande para o qual o ganho será ótimo.
- Fixado o tamanho de um problema existe um número de processadores para o qual o ganho é máximo, i.e., a partir de um certo número de processadores o ganho diminui.

O conhecimento dos limites da paralelização é um dado fundamental para uma utilização eficiente desta técnica.

Nas máquinas hipercúbicas com 8 processadores obtiveram-se ganhos de até 7,78 para problemas de tamanho 512X512. Nas máquinas de memória central com 4 processadores para o mesmo tamanho de problema os ganhos foram de até 3,74 e na rede com 6 estações, ainda com o mesmo tamanho de problema, os ganhos foram de até 5,34.

O impacto do custo de comunicação no desempenho das estações foi marcante e deve ser o primeiro alvo na busca de melhoria de ganhos.

Apesar dos experimentos não considerarem a entrada de dados, acredita-se que suas conclusões têm generalidade, sob o argumento de que a quantidade de entrada de dados na migração é muito pequena comparada ao volume de processamento executado.

Pôde-se criar uma hierarquia de capacidade computacional dos diversos processadores testados, para a migração  $\omega$ -x, como mostra a tabela 8.1 (atribuindo o valor 1,0 ao processador menos potente):

| maquina      | potencia CPU |
|--------------|--------------|
| NCP I/COPPE  | 1,00         |
| INTEL/iPSC   | 9,70         |
| RS-6000/320H | 25,40        |
| IBM 3090     | 35,90        |
| IBM 9021     | 99,20        |

Tabela 8.1 - Hierarquia da capacidade computacional na migração  $\omega$ -x.

A atualidade do tema desta tese é confirmada pelos trabalhos correlatos que podem ser vistos em [Almasi92], também em 2-D, e em [Black92a], [Black92b] e [Lynn92] em aplicações 3-D, que são análogas. Os resultados que foram obtidos bem como as conclusões a que se chegou nesta tese são coerentes com os obtidos nos trabalhos citados, onde utilizou-se um maior número de processadores, outros softwares para conexão de estações e redes de comunicação com várias taxas de

transmissão, mas há grande similaridade com o que se apresentou neste trabalho, especialmente nas formas de paralelização.

Como futuros trabalhos em continuação a este, pode-se indicar:

- Implementação de versão de produção da Migração  $\omega$ -x em rede de estações usando o PVM.
- Investigação da viabilidade de uso de paralelismo em outras etapas do Processamento Sísmico.
- O fato de cada máquina testada ter exigido uma codificação diferente para a paralelização inspira a procura de uma linguagem de alto nível, onde o usuário codifique este paralelismo de um modo único e o compilador se encarregue de expressá-lo na melhor forma para cada equipamento. Esta é a proposta do HIGH PERFORMANCE FORTRAN, baseado no FORTRAN 90 padrão, cujo desenvolvimento deve ser acompanhado com interesse.

## Referências Bibliográficas

- [Almasi92] Almasi, G.S., McLuckie, T., Bell, J., Gordon, A., and Hale, D. Parallel distributed seismic migration. Future Generation Computer Systems, V. 8, p 9-26, 1992.
- [Black92a] Black, J.L., Su, C.B. Networked parallel seismic computing. In: Offshore Technology Conference, 24, 1992. Houston. Proceedings ... Richardson, TX, Offshore Technology Conference, 1992. p. 169-176.
- [Black92b] Black, J.L., Su, C.B. Performance of parallel downward continuation. In: Annual SEG International Meeting, 62, 1992. New Orleans. Expanded Technical Program Abstracts. p. 326-329.
- [Bentz61] Bentz A. Lehrbuch der angewandten Geologie. Stuttgart(Enke), 1961.
- [Carnahan69] Carnahan, B., Luther, H.A., and Wilkes, J.O., 1969, Applied numerical methods. New York, John Wiley & Sons, 1969. 604 p.
- [Claerbout85] Claerbout J.F. Imaging the earth's interior. Oxford, Blackwell Scientific Publications, 1985. 398 p. il.
- [Chun81] Chun, J.H, and Jacewitz, C. Fundamentals of frequency-domain migration. Geophysics, V. 46, p. 717-732, 1981.
- [Duarte85] Duarte, O.O. Processamento de reflexão sísmica Rio de Janeiro, Petrobrás, 1985. (apostila).

- [Duarte88] Duarte, O.O. Dicionário inglês-português de termos técnicos usados na prospecção sísmica. Rio de Janeiro, Petrobrás. CENPES. SINTEP. 1988. 89 p.
- [Hatton86] Hatton, L., Worthington, M.H., and Makin, J. Seismic data processing: theory and practice. Oxford, Blackwell Scientific Publications, 1986. 177p. il.
- [Loewenthal76] Loewenthal, D., Lu, L., Roberson, R., and Sherwood, J. The wave equation applied to migration. Geophysical Prospecting, V. 24, p. 380-399, 1976.
- [Lynn92] Lynn, W.S., Perkins, W., Cabrera J., and French, W.S. 3-D prestack imaging on massively parallel computers. In: Offshore Technology Conference, 24, 1992. Houston. Proceedings ... Richardson, TX, Offshore Technology Conference, 1992. p. 177-179.
- [Robinson80] Robinson, E.A., and Treitel, S. Geophysical signal analysis. Englewoods Cliffs, Prentice-Hall, 1980. 466 p. il.
- [Rosa93] Rosa, A.L.R. Migração de dados sísmicos. Rio de Janeiro, Petrobrás, 1993. (apostila).
- [Sheriff91] Sheriff, R.E. Encyclopedic dictionary of exploration geophysics. 3. ed. Tulsa, Society of Exploration Geophysicists, 1991. 376 p. il.
- [Sunderam90] Sunderam, V.S. PVM: a framework for parallel distributed computing. Concurrency: practice and experience, V. 2, n. 4, p. 315-339, 1990.
- [Yilmaz87] Yilmaz, O. Seismic data processing. Tulsa, Society of Exploration Geophysicists, 1987. 526 p. il.
- [Young72] Young, D.M., and Gregory, R.T. A Survey of numerical mathematics, Dover Publications Inc., 1972.