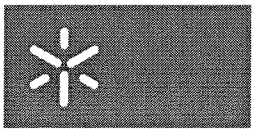


[CN-01]

Pais, J.C., Delgado, R.

"Análise de métodos de resolução de sistemas de equações lineares em processamento paralelo"

III Encontro de mecânica computacional, Coimbra, 1992

Análise de métodos de resolução de sistemas de equações lineares em processamento paralelo

Jorge Carvalho Pais
Raimundo Moreno Delgado
Faculdade de Engenharia da Universidade do Porto
4099 Porto Codex, Portugal

SUMÁRIO

É objectivo deste trabalho apresentar as possibilidades do processamento paralelo na resolução de sistemas de equações lineares, tendo-se para o efeito seleccionado três métodos de resolução de sistemas:

1 - Método da eliminação de Gauss
2 - Método da factorização de Choleski
3 - Método iterativo dos Gradientes Conjugados

Para os métodos utilizados são apresentados resultados do Speedup e respectivas eficiências em função do número de processadores utilizados.

1. INTRODUÇÃO

Os problemas de Engenharia de análise estrutural implicam, de um modo geral, a resolução de sistemas de equações lineares de grandes dimensões, a que corresponde uma grande parte do tempo total de análise. Por isso tem-se procurado desenvolver métodos que conduzam a tempos cada vez menores para a sua resolução. De entre os métodos conhecidos salienta-se o da eliminação de Gauss, por ser o de mais generalizada utilização, o método de factorização de Choleski, que conduz a ganhos de tempo em relação ao da eliminação de Gauss necessitando da mesma quantidade de memória, e o método iterativo dos gradientes conjugados que para sistemas bem condicionados se revela de grande eficiência.

O estudo dos métodos de resolução de sistemas de equações lineares em processamento paralelo, que constitui o objectivo deste trabalho, vai ter como base o estudo das eficiências destes três métodos sob as condições mais desfavoráveis, ou seja, a não existência de termos não nulos na matriz dos coeficientes.

2. SISTEMAS DE PROCESSAMENTO PARALELO

A necessidade de resolução de problemas que exigem grandes quantidades de cálculo impulsionou o aparecimento de computadores cada vez mais poderosos tanto em termos de velocidade de cálculo como em termos de capacidade de armazenamento de informação.

O primeiro passo para conseguir estes objectivos foi dado pela construção de computadores paralelos, máquinas que para além do processador central dispõem de um processador aritmético partilhando ambos uma memória central.

O passo seguinte foi dado pelos supercomputadores, máquinas com vários processadores mas apenas com uma memória que é acedida por todos eles. Estes supercomputadores, bastante rápidos, têm o inconveniente de, ao possuirem este tipo de memória, se tornarem pouco eficientes quando a ela chegam muitos acessos no mesmo instante.

O passo mais recente foi dado pelos multicomputadores, máquinas que têm vários processadores, dispondo cada um deles de uma memória local. Nestas máquinas, os processadores estão ligados uns aos outros por uma rede de comunicações que permite que os dados armazenados na memória dum processador sejam transferidos para outro, sendo necessário a existência de sincronismo entre eles para que esta comunicação se faça correctamente.

2.1. *Sistema existente na FEUP*

A máquina actualmente existente na FEUP para processamento paralelo é um multicomputador composto por 16 processadores T800, vulgarmente designados por "TRANSPUTERS", por serem compostos por um processador central, um processador aritmético, 2 Mbytes de memória local e quatro ligações para comunicação.

Um dos pontos fulcrais destes multicomputadores é o tipo e a forma das ligações entre processadores. Elas podem ter as mais variadas formas respeitando sempre as quatro ligações entrada/saída existentes em cada processador. A figura 1 ilustra alguns dos tipos mais comuns de redes de processadores [1].

Depois de definida a rede de comunicações entre processadores ficam determinados os percursos a percorrer de um processador até outro qualquer.

A linguagem de programação paralela para este tipo de máquinas é o OCCAM, a qual permite realizar todas as comunicações entre processadores seguindo as ligações previamente definidas. Esta linguagem é caracterizada por ser uma linguagem de baixo nível e trabalhar ao nível do hardware. Tal como todas as outras linguagens de baixo nível é bastante difícil e morosa a sua programação.

2.2. *Linguagem de distribuição de componentes (CDL)*

Para superar as dificuldades, que advêm do uso do OCCAM, esta máquina dispõe de um programa, o CDL, que permite definir novos tipos de comunicações entre processadores. Na figura 2 encontram-se representados os tipos de ligações suportados por esta linguagem.

Com este tipo de comunicação podem utilizar-se canais definidos na linguagem C e consequentemente utilizar esta linguagem que é mais fácil de ser manuseada. O único inconveniente que se nos depara tem a ver com a perda de certos canais de comunicação, como é o caso da estrutura Farm em que não é permitida a comunicação entre Slaves.

3. SPEEDUP E EFICIÊNCIA

A medida da qualidade dos algoritmos paralelos é usualmente quantificada através do seu Speedup e da eficiência [2].

O Speedup Sp dum algoritmo paralelo a correr em p processadores é dado pelo quociente entre o tempo $T1$ de resolução num só processador e o tempo Tp de resolução em p processadores em funcionamento paralelo.

$$Sp = T1/Tp \qquad (1)$$

Para $T1$ geralmente utiliza-se o tempo de resolução do algoritmo sequencial, apesar deste ser diferente do algoritmo paralelo a correr apenas num processador.

O valor óptimo para o Speedup Sp deve ser igual a p, pois isto quer dizer que o programa paralelo a correr em p processadores demorou p vezes menos tempo que o sequencial. Esta situação ideal , no entanto, não se verifica porque a comunicação entre processadores gasta muito mais tempo do que aquele que é necessário para uma quantidade significativa de operações matemáticas, pelo que Sp é sempre menor que p.

O Speedup pode assim ser interpretado como o número de processadores que estão a trabalhar a 100% comparativamente ao programa sequencial.

A eficiência Ep dum algoritmo paralelo é dada pelo quociente entre o Speedup Sp e o número de processadores .

$$Ep = Sp/p \qquad (2)$$

Esta quantidade assume sempre valores inferiores à unidade já que Sp é menor que p.

4. MÉTODO DE GAUSS

Como é sabido, o método da eliminação de Gauss consiste na eliminação de todos os elementos da matriz abaixo da diagonal principal. Isto é conseguido eliminando uma coluna de cada vez até obtermos só valores nulos abaixo da diagonal principal, após a qual a obtenção da solução do sistema se faz por retrosubstituição.

4.1. *Tempo de cálculo do algoritmo sequencial*

Para a medida da eficiência dos algoritmos paralelos para o método de Gauss em banda simétrica foi necessário, dadas as limitações de memória de um só processador, extrapolar os valores do tempo de resolução do algoritmo sequencial, baseando-se esta extrapolação no número de operações efectuadas por este algoritmo.

A eliminação de Gauss para uma matriz em banda simétrica conduz a:

$$S = 1/2*N*LSB\hat{}2 + 1/2*N*LSB - 1/3*LSB\hat{}3 + 1/3*LSB - N \qquad (3)$$

somas e subtracções e

$$P = 1/2*N*LSB\hat{}2 + 3/2*N*LSB - 1/3*LSB\hat{}3 - 1/2*LSB\hat{}2 + 5/6*LSB - 2*N \quad (4)$$

multiplicações e divisões, sendo N o número de equações do sistema e LSB a largura da sua semi-banda.

Para a obtenção dos valores extrapolados, admitiu-se que o tempo de extrapolação, t, pode ser obtido por:

$$t = (\gamma s)*S + (\gamma p)*P \quad (5)$$

Os coeficientes (γs) e (γp) foram aferidos a partir dos tempos medidos em cálculos compatíveis com a capacidade de memória de um processador, tendo-se constatado uma boa constância para o conjunto de sistema seleccionado até 1500 equações, número máximo que é possível resolver num só processador.

A retrosubstituição para uma matriz simétrica em banda necessita de:

$$S = N*LSB - 1/2*LSB\hat{}2 - N + 1/2*LSB \quad (6)$$

somas e subtracções e

$$P = N*LSB - 1/2*LSB\hat{}2 + 1/2*LSB \quad (7)$$

multiplicações e divisões.

Os valores extrapolados, para o tempo de retrosubstituição, foram obtidos pela expressão

$$t = (\gamma)*(S + P) \quad (8)$$

que mostrou ser bastante eficiente dada a grande constância do coeficiente (γ).

4.2. *Algoritmo paralelo*

Para o desenvolvimento do algoritmo de resolução do sistema de equações em processamento paralelo adoptou-se a estrutura Farm. Como se pode depreender desta estrutura, é-se obrigado a elaborar dois algoritmos, um para correr no processador Master e outro para correr nos processadores Slaves.

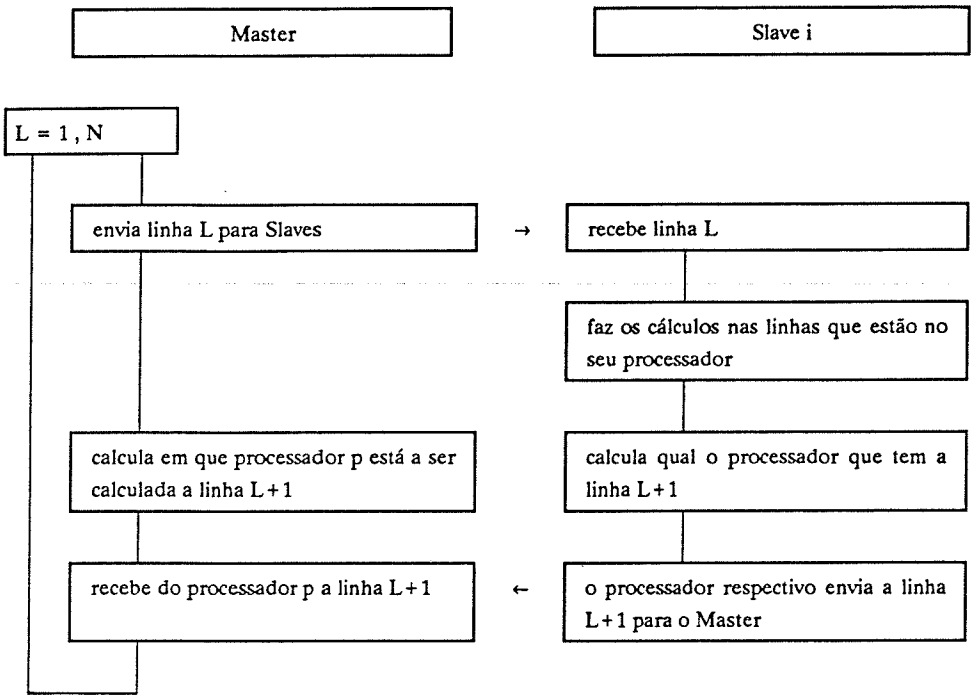
O Master encarrega-se de enviar informação para os Slaves, estes procedem aos cálculos inerentes a essa informação após os quais enviam para o Master os consequentes resultados, repetindo-se este processo o número de vezes necessário até ser resolvido o sistema.

A forma ideal de distribuição da matriz dos coeficientes pelos Slaves deverá ser tal que o processador i tenha as linhas i, i+p, i+2p,.. , sendo p o número de processadores. Esta distribuição conduz a um maior equilíbrio da quantidade de informação existente em cada procesador, sendo por isso mais fácil obter o sincronismo entre processadores.

4.2.1. Eliminação

O algoritmo para o Master encarrega-se de proceder ao envio de cada linha do sistema para todos os processadores e em seguida receber dum deles a linha seguinte do sistema, linha esta que será enviada no passo seguinte.

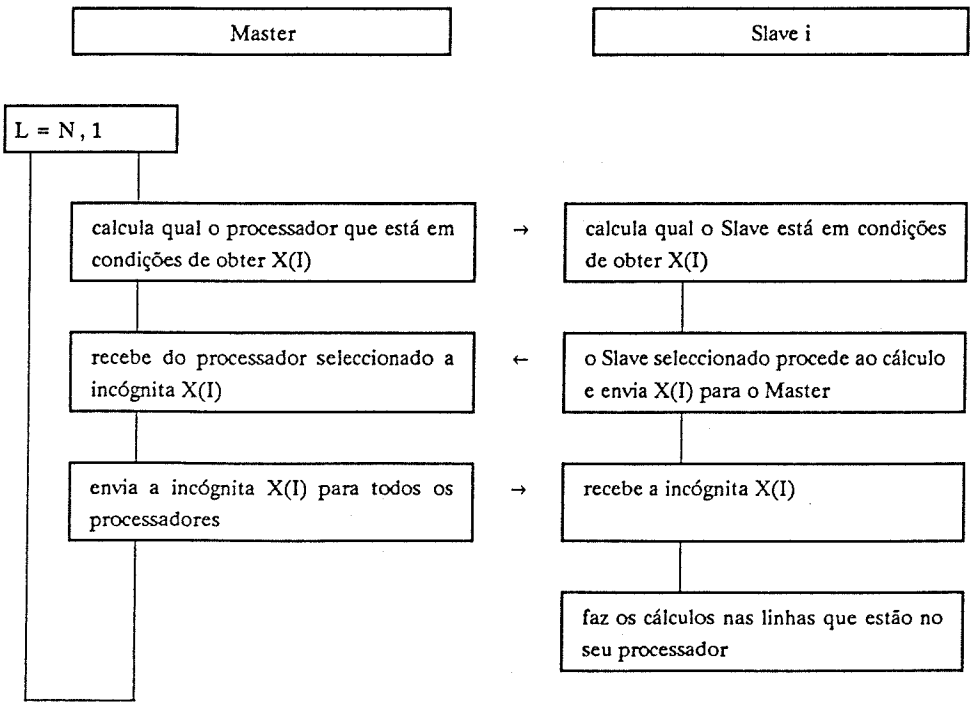
Cada Slave lê uma linha enviada pelo Master e para cada uma delas procede aos cálculos necessários nas linhas armazenadas nesse processador.



4.2.2. Retrosubstituição

O algoritmo para o Master recebe uma incógnita já calculada e envia-a a todos os processadores, até que se esgotem as incógnitas.

Os Slaves recebem uma incógnita já calculada e com ela procede a cálculos nas linhas que armazena. Seguidamente, um dos Slaves calcula a incógnita que irá para o passo seguinte.



4.3. *Resultados para matriz em banda simétrica*

A figura 3 mostra que, como era de esperar, o tempo de eliminação diminui à medida que aumenta o número de processadores. Embora se constate que esta diminuição se atenua com o número de processadores, por aumentar o tempo de comunicação, pode observar-se na figura 4 que se obtêm uma muito significativa diminuição do tempo de cálculo em relação ao cálculo sequencial.

A figura 5 evidencia que, na gama dos cálculos efectuados, a fase de retrosubstituição, por necessitar de um grande número de comunicações e um pequeno número de operações matemáticas, é mais demorada que a correspondente sequencial, atenuando-se no entanto com o aumento do número de equações. Pode ainda observar-se no entanto que o tempo para esta fase é de um modo inferior a 10% do tempo da fase de eliminação.

Em termos de eficiência constata-se, nas figuras 6 e 7, que esta é crescente com o número de equações e que para sistemas de grandes dimensões se obtêm valores que se aproximam da eficiência máxima.

4.4. *Resultados para matriz simétrica cheia*

Para o caso de matriz simétrica cheia podem extraír-se as mesmas conclusões que para matrizes em banda simétrica, sendo de destacar o facto que para as cheias se obtêm eficiências muito maiores, devendo-se isto ao facto de se proceder a um maior número de operações para resolver um sistema simétrico.

A figura 8 é exemplo disto, na qual se pode observar que a retrosubstituição paralela, para a gama de cálculos efectuados, começa a gastar tempos da mesma ordem de grandeza que a sequencial.

Na figura 9 e 10 observa-se que na eliminação paralela para matrizes simétricas as eficiências tendem rapidamente para o seu valor máximo para quase qualquer gama de sistemas, podendo-se concluir que o processamento paralelo é tanto mais vantajoso quanto mais cheia for a matriz.

5. MÉTODO DE CHOLESKI (FACTORIZAÇÃO)

O método de Choleski baseia-se na decomposição da matriz dos coeficientes no produto de duas matrizes, ou seja:

$$A = L * L^T \tag{9}$$

sendo L uma matriz triangular inferior e L^T a sua transposta [3] e [4].

Após a factorização, a obtenção da solução do sistema far-se-á com uma substituição progressiva e uma retrosubstituição, ou seja:

$$A * X = B \tag{10}$$

$$(L * L^T) * X = B \tag{11}$$

$$L * Y = B (\text{substituição progressiva}) \tag{12}$$

$$L^T * X = Y (\text{retrosubstituição}) \tag{13}$$

5.1. *Algoritmo sequencial*

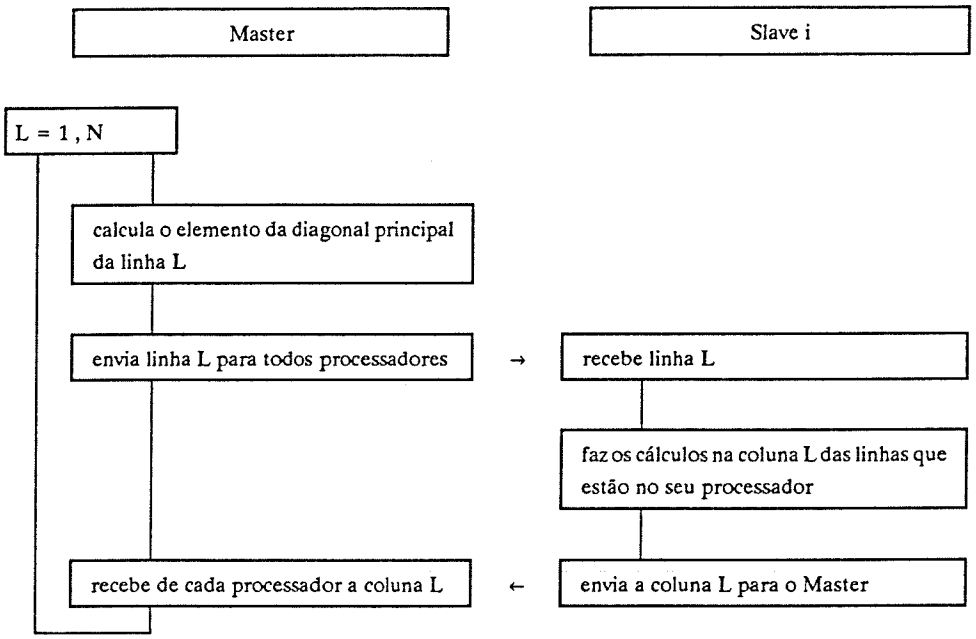
Admitindo que temos uma matriz simétrica e definida positiva, para cada coluna, este método executa as seguintes fases na obtenção da matriz *L*:

- Transformação apropriada do elemento da diagonal principal.
- Com o valor anterior calcula os restantes elementos da coluna em estudo.

5.2. *Algoritmo paralelo*

Tal como no método anterior, o Master encarrega-se essencialmente da comunicação entre processadores, sendo os cálculos necessários à obtenção dos elementos da matriz *L* efectuado nos Slaves.

Para a factorização tem-se:



A substituição progressiva e a retrosubstituição são paralelizáveis de modo idêntico ao feito para o método de Gauss.

5.3. *Resultados para matriz simétrica*

Do gráfico da figura 11 conclui-se que, neste método, os ganhos nos tempos de factorização para vários processadores são, para este número de equações, pouco significativos. Se se extrapolarem estes valores pode constatar-se que só para sistemas muito maiores é que começa a haver ganhos substanciais.

A principal conclusão que se pode tirar do gráfico da figura 12 é uma consequência do que foi dito para o gráfico anterior, constatando-se que as eficiências deste método para sistemas desta ordem de grandeza são bastante baixas. Este método só se revela eficiente para sistemas de dimensões bastante maiores que as ensaiadas.

6. MÉTODO DOS GRADIENTES CONJUGADOS (ITERATIVO)

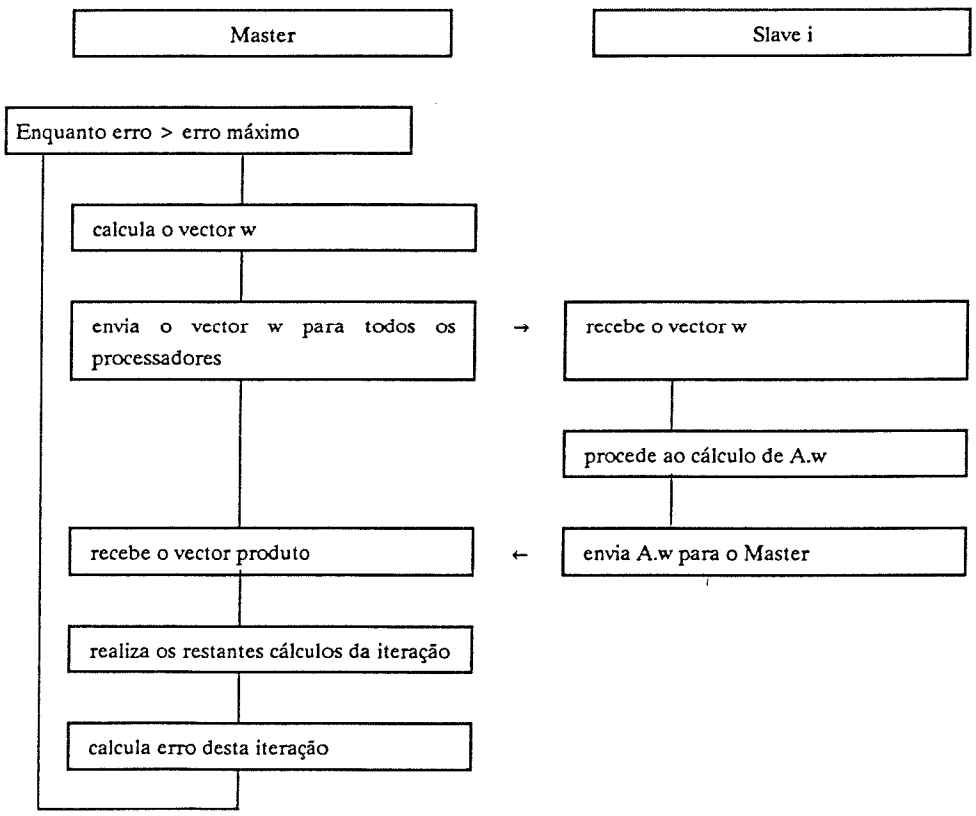
Neste ponto faz-se referência ao método dos gradientes conjugados, um método iterativo cujos fundamentos [5], dada a sua complexidade, se encontram fora do âmbito deste trabalho.

6.1. *Algoritmo sequencial*

Em relação ao algoritmo sequencial pode concluir-se que o tempo de cada iteração é condicionado pelo tempo de um produto duma matriz A por um vector w, tal como pode verificar-se no gráfico da figura 13.

Conclui-se assim que a paralelização deste método deverá incidir essencialmente sobre o produto A.w.

6.2. *Algoritmo paralelo*



6.3. *Resultados para matriz em banda simétrica*

A análise do gráfico da figura 14 permite concluir que os tempos do algoritmo paralelo são superiores ao do sequencial até às 600 equações, verificando-se a partir daí um ganho crescente com o cálculo em processamento paralelo.

O gráfico da eficiência, figura 15, evidencia que, para a gama de sistemas ensaiados, as eficiências são baixas, sendo no entanto de esperar que para sistemas de maiores dimensões, em que o peso do tempo de comunicações diminui, se encontrem eficiências aceitáveis.

7. ANÁLISE COMPARATIVA DOS MÉTODOS

A comparação entre os tempos totais, de resolução dos sistemas de equações ensaiadas, para os métodos apresentados, sugere recomendações de ordem geral quanto ao interesse da sua utilização.

Para sistema de matriz simétrica em banda compara-se na figura 16 os resultados obtidos com o método de Gauss e o método dos gradientes conjugados, constatando-se que este último é substancialmente menos demorado. No entanto deve ter-se em conta que, sendo o método dos gradientes conjugados iterativo, o tempo de cálculo é proporcional ao número de iterações, sendo este número muito sensível ao grau de condicionamento da matriz dos coeficientes e ao número de equações. Deve ainda acrescentar-se que para problemas que envolvam mais do que um segundo membro a situação apresentada se pode alterar, já que o número de operações envolvidas para o cálculo de outros segundos membros se reduz drasticamente no método de Gauss.

No caso de sistemas com matrizes cheias a comparação que se apresenta na figura 17 mostra que o método de Choleski é um pouco mais rápido que o de Gauss.

8. CONCLUSÃO

Sendo o processamento paralelo relativamente recente não se encontram ainda generalizados algoritmos para a resolução de sistemas de equações, ao contrário do que acontece para os sequenciais, constituindo este trabalho uma contribuição no estudo das suas possibilidades e vantagens.

Os algoritmos de resolução de sistemas de equações em processamento paralelo, apresentados de forma sistemática ao longo deste trabalho, evidenciam que é necessário idealizar um ou mais modos de funcionamento e elaborar dois algoritmos distintos, um para correr no Master e outro nos Slaves. O algoritmo Master encarrega-se habitualmente da distribuição das informações para os Slaves, enquanto estes se encarregam das operações necessárias sobre a informação neles armazenada. A elaboração destes algoritmos reveste-se de cuidados especiais já que é necessário obter sincronismo entre as tarefas de todos os processadores de modo a eles entrarem em funcionamento simultâneo.

A paralelização dos métodos apresentados demonstrou que se obtém boas eficiências tendo-se conseguido ganhos de tempo muito significativos relativamente ao processamento sequencial. Conclui-se, ainda, que esta eficiência cresce com o aumento da dimensão dos sistemas já que o peso das comunicações entre processadores diminui relativamente ao do processamento aritmético.

No âmbito dos ensaios efectuados conclui-se que para a resolução de sistemas de equações cheios o método de Choleski apresenta vantagem em relação ao da eliminação de Gauss, e que para sistemas em banda o método iterativo dos gradientes conjugados é mais vantajoso, embora se deva ter em atenção que sendo este método iterativo é muito sensível ao grau de condicionamento e número de equações do sistema.

9. REFERÊNCIAS

[1] - Armando R.C. Almeida. "Distributed continuous simulation of large-scale systems", Ph D Thesis, Joanesburgo, 1990.

[2] - Y. Robert. "The impact of vector and parallel architectures on the Gaussian elimination algorithm", Manchester University Press, Grea Britain, 1990.

[3] - Humberto L. Soriano. "Sistemas de equações algébricas lineares em problemas estruturais", Seminário 280, LNEC, 1981.

[4] - Alves Filho. "The use of transputers based computers in Finite Element calculation", Ph D Thesis, University College of Swansea, Swansea, 1989.

[5] - Álvaro F.M. Azevedo e Joaquim A.O. Barros. "Análise comparativa de métodos directos e iterativos na resolução de grandes sistemas de equações lineares", JPEE, Tema A, LNEC, 1990.

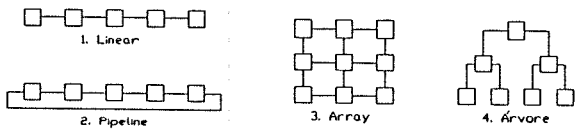


Fig. 1 - Tipos de ligação de redes de processadores.

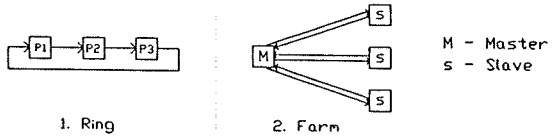
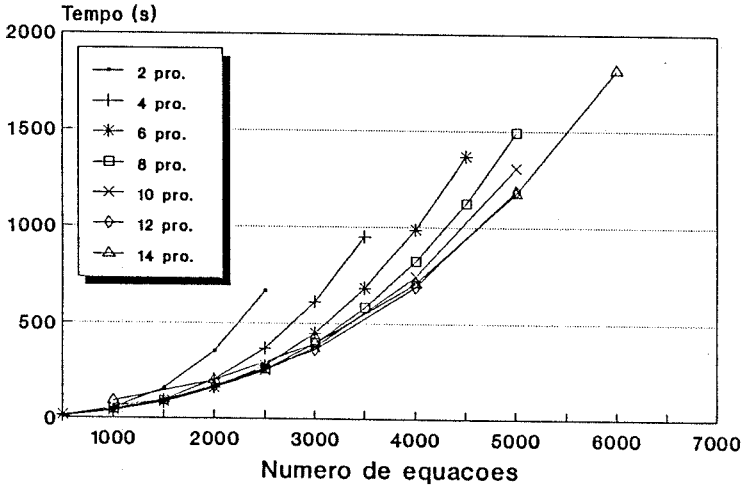
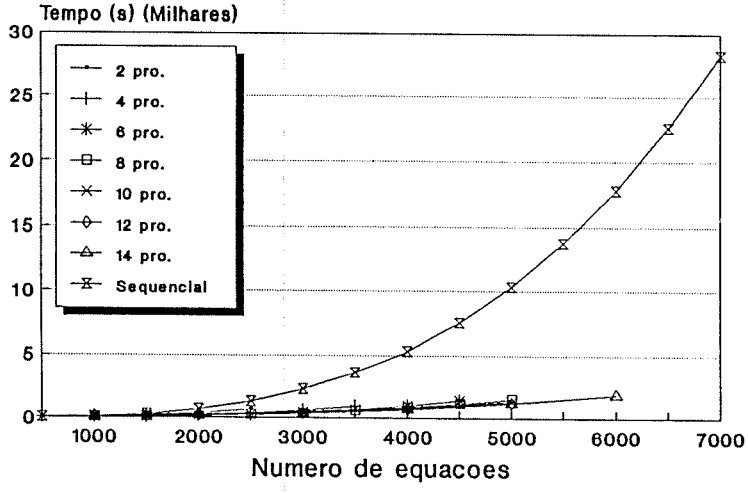


Fig. 2 - Tipos para ligação em CDL.





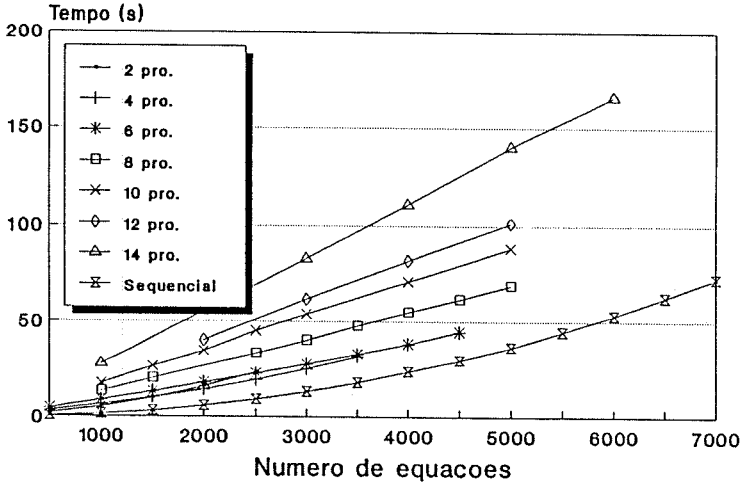


Fig.6-Gauss, simetrica, banda, paralelo
Speed-up (Tseq/Tpar) eliminacao

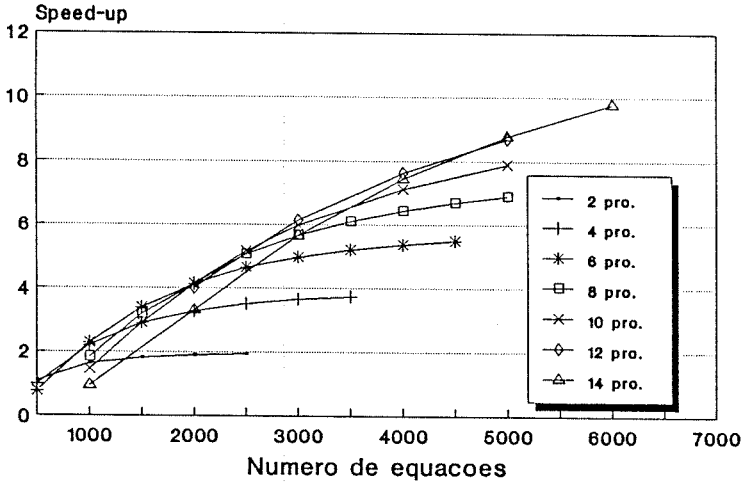


Fig.7-Gauss, simetrica, banda, paralelo
Eficiencia (S-up/pro) eliminacao

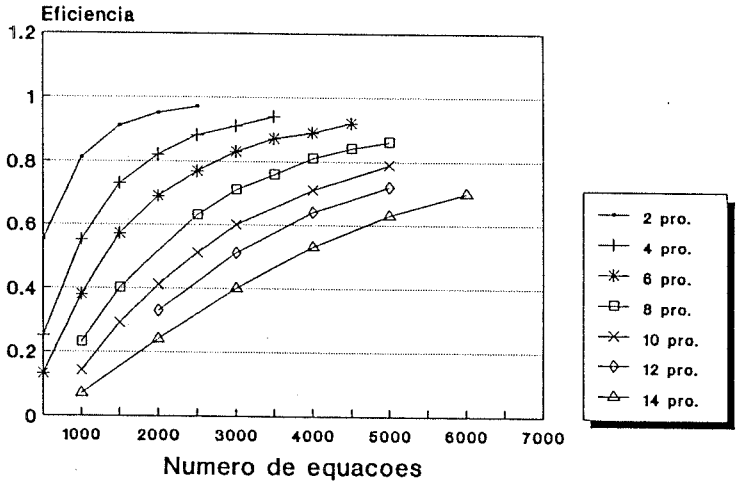


Fig.8-Gauss, simetrica, cheia, paralelo
Tempo de retro-substituicao

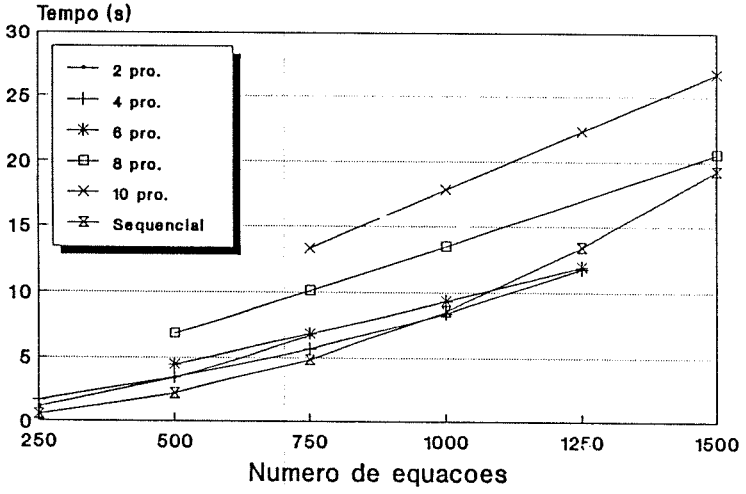


Fig.9-Gauss, simetrica, cheia, paralelo
Speedup (Tseq/Tpar) eliminacao

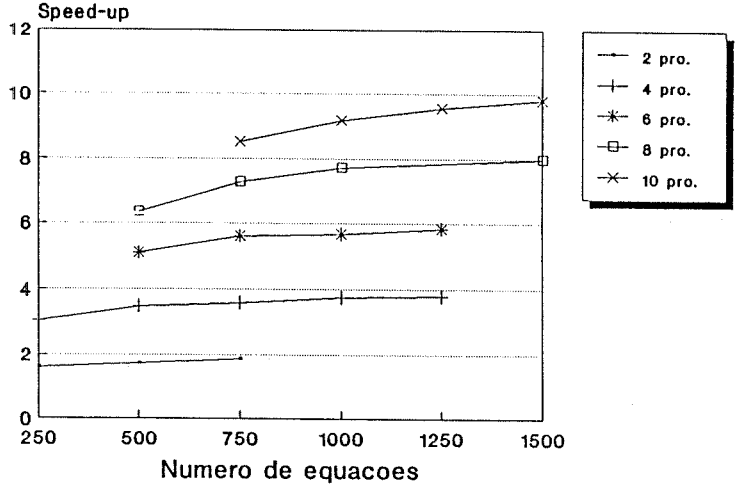


Fig.10-Gauss, simetrica, cheia, paralelo
Eficiencia (S-up/pro) eliminacao

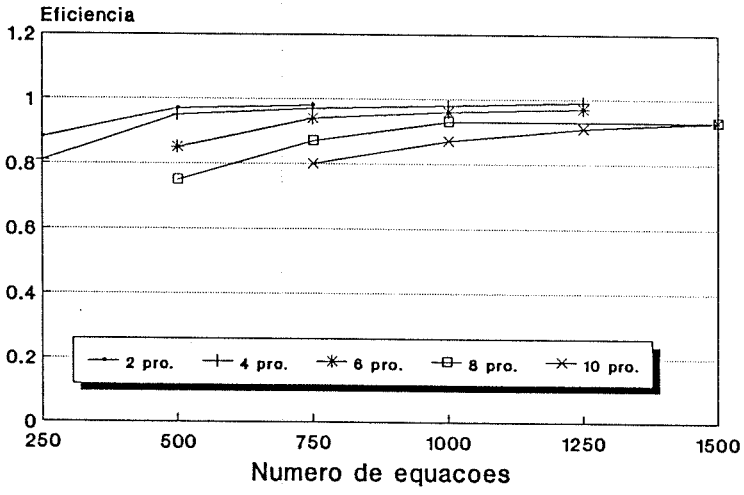


Fig.11-Choleski, simetrica, cheia, paral
Tempo de factorizacao

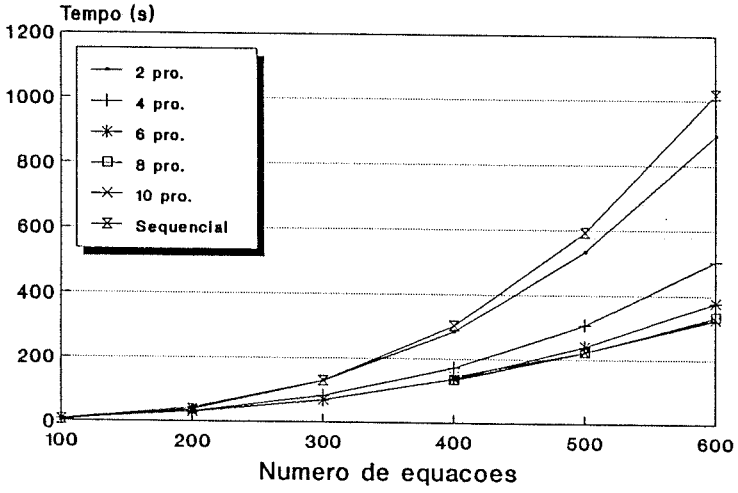


Fig.12-Choleski, simetrica, cheia, paral
Eficiencia (S-up/pro) factorizacao

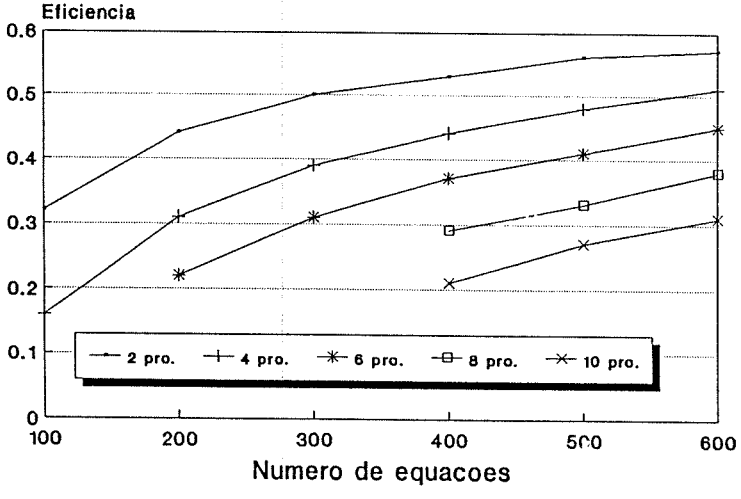


Fig.13-G. C., simetrica, banda, sequenci
Tempo iterecao / A.w

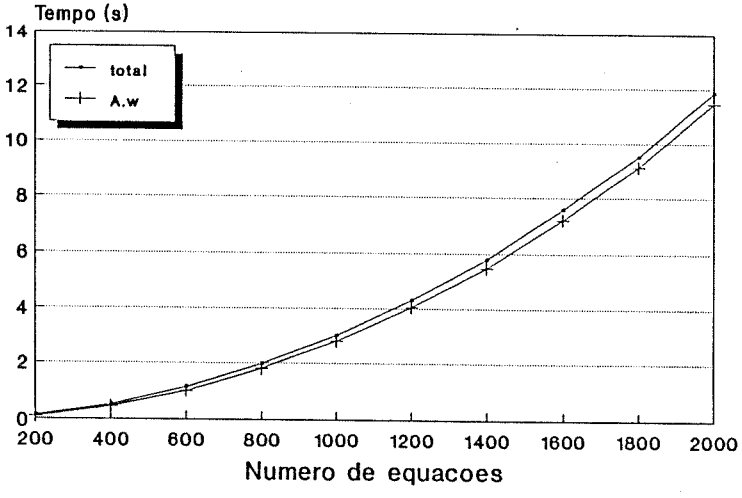


Fig.14-G. C., simetrica, banda, paralelo
Tempo duma iteracao

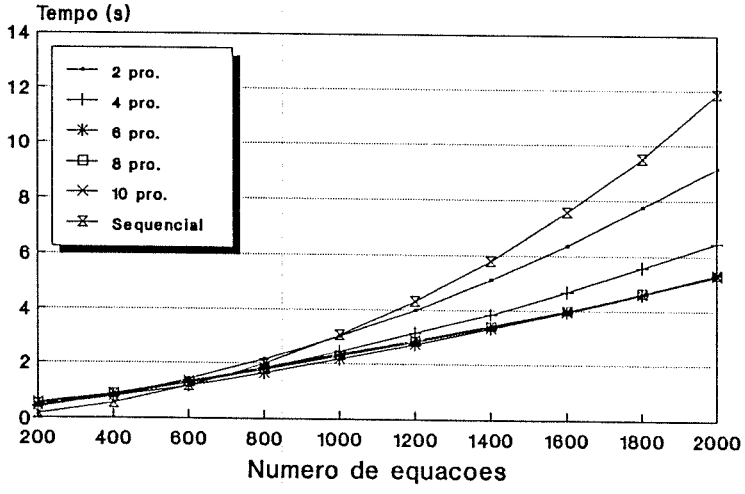


Fig.15-G. C., simetrica, banda, paralelo
Eficiencia (S-up/pro) iteracao

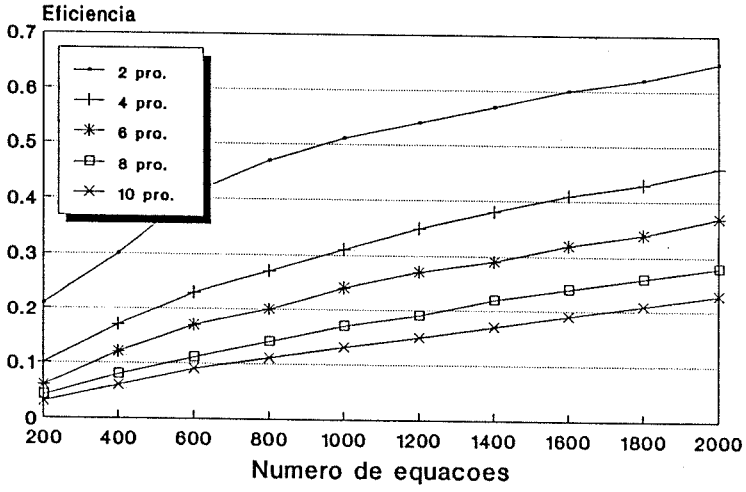


Fig.16-Tempo total Gauss / G.C.
6 processadores

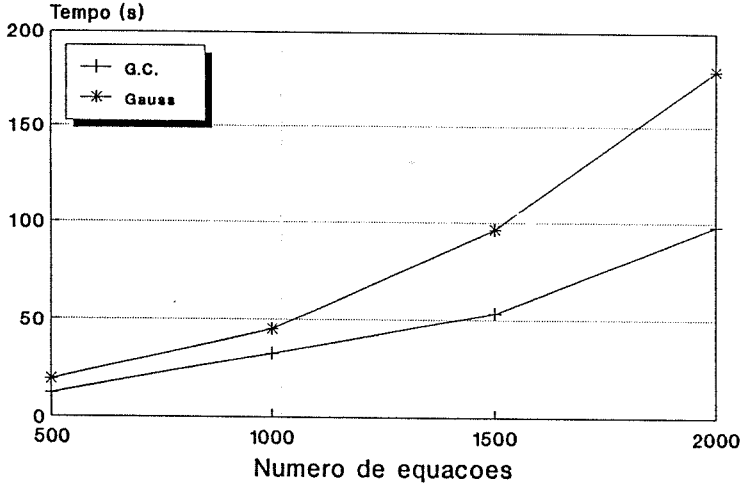


Fig.17-Tempo total Gauss / Choleski
6 processadores