



Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação (ICMC)
SSC0903 - Computação de Alto Desempenho

Projeto de Algoritmos Paralelos (PCAM) - Método Iterativo de Jacobi-Richardson

Matheus Yasuo Ribeiro Utino - 11233689

Pedro Ribas Serras - 11234328

Vinícius Silva Montanari - 11233709

Docente: Dr. Paulo Sérgio Lopes de Souza

São Carlos
27 de maio de 2022

Sumário

1 Projeto de Algoritmos Paralelos (PCAM)	2
1.1 Particionamento	2
1.1.1 Convergência do método	2
1.1.2 Método Iterativo de Jacobi-Richardson	4
1.2 Comunicação	7
1.2.1 Convergência do método	7
1.2.2 Método Iterativo de Jacobi-Richardson	7
1.3 Aglomeração	8
1.3.1 Convergência do método	8
1.3.2 Método Iterativo de Jacobi-Richardson	8
1.4 Mapeamento	9

Lista de Figuras

1	Redução e particionamento para a convergência.	3
2	Redução e particionamento para a convergência de várias linhas.	3
3	Sistema linear isolando x.	4
4	Redução e particionamento - Método Iterativo de Jacobi-Richardson	5
5	Redução e particionamento para a convergência de várias linhas - Método Iterativo de Jacobi-Richardson.	6
6	Critério de parada do algoritmo.	6

1 Projeto de Algoritmos Paralelos (PCAM)

Serão descritos os passos para o desenvolvimento do PCAM, que é composto por quatro etapas: particionamento, comunicação, aglomeração e mapeamento. Durante o processo será explicado detalhadamente como será desenvolvido o algoritmo paralelo, assim como as decisões de projeto que foram tomadas.

1.1 Particionamento

Essa etapa visa extrair as tarefas e suas dependências, no caso sera realizado o particionamento de dados.

1.1.1 Convergência do método

Inicialmente precisamos verificar se o método converge para o sistema linear analisado. No caso, a matriz A será particionada em $N^2 - N$ tarefas, em que cada tarefa vai possuir o elemento $A[i, j]$, desconsiderando o elemento da diagonal principal $A[i, i]$. Após isso, as tarefas que são responsáveis pela linha i de A devem fazer um somatório dos módulos dos elementos dessa linha, preferencialmente pelo processo de redução. Por fim, esse valor do somatório vai ser dividido pelo elemento $A[i, i]$, em que $A[i, i] \neq 0$ para a convergência do método, caso o somatório seja superior a 1 o método não converge, por outro lado se for inferior a esse valor significa que o método converge para o sistema linear escolhido. A ideia da redução e do particionamento para uma linha i da matriz A pode ser visualizada na fig. 1.

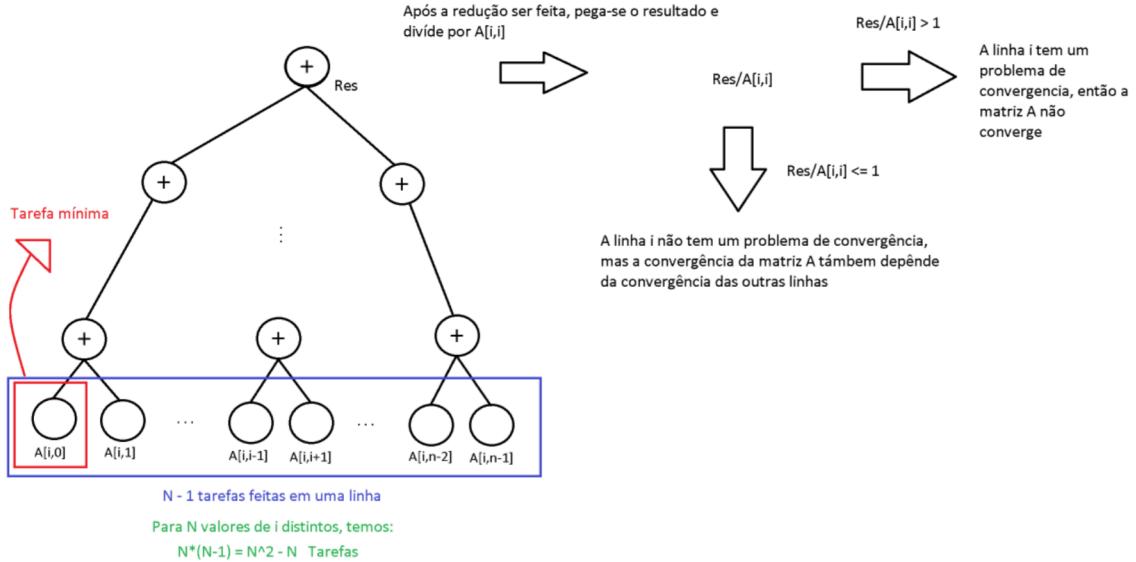


Figura 1: Redução e particionamento para a convergência.

Também podemos demonstrar o comportamento para mais de uma linha, conforme a fig. 2.

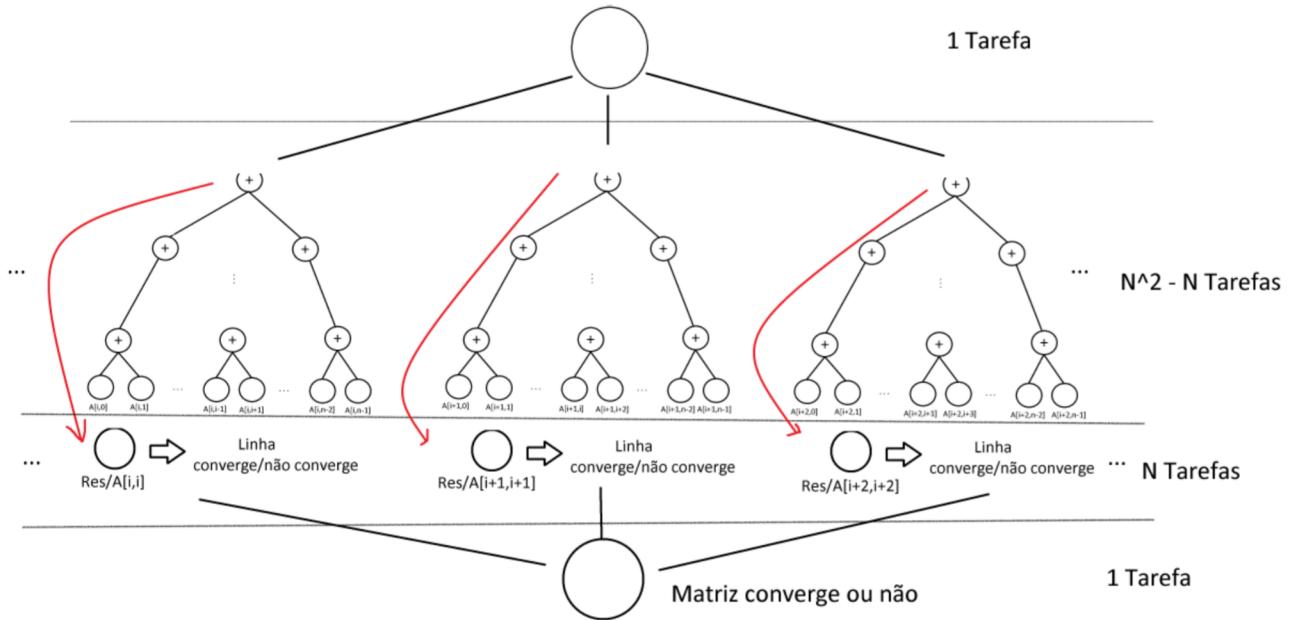


Figura 2: Redução e particionamento para a convergência de várias linhas.

1.1.2 Método Iterativo de Jacobi-Richardson

Nesse processo temos a matriz A e os vetores x e b como descrito no problema. Além deles, teremos o vetor xa para guardar os valores da última iteração de x . Nas iterações para encontrar um novo vetor x , inicia-se cada novo valor $xa[i]$ como $x[i]$, sendo que o valor de x pode ser tomado inicialmente como zero para todos os seus elementos. Tendo o vetor xa pode-se encontrar os novos valores para o vetor x pelo processo da fig. 3 . Esse processo pode ser feito em paralelo.

$$\begin{cases} x_1^{(k+1)} = \left(b_1 - (a_{12}x_2^{(k)} + \dots + a_{1n}x_n^{(k)}) \right) / a_{11}, \\ x_2^{(k+1)} = \left(b_2 - (a_{21}x_1^{(k)} + \dots + a_{2n}x_n^{(k)}) \right) / a_{22}, \\ \vdots \quad \vdots \quad \vdots \quad \vdots \\ x_n^{(k+1)} = \left(b_n - (a_{n1}x_1^{(k)} + \dots + a_{n,n-1}x_{n-1}^{(k)}) \right) / a_{nn}, \end{cases}$$

para $k = 0, 1, \dots$

Figura 3: Sistema linear isolando x.

Agora a matriz A é particionada em $N^2 - N$ tarefas, em que cada uma delas possui um elemento de $A[i, j]$ e o elemento $xa[j]$ correspondente a coluna j que a tarefa está computando. Após as multiplicações $A[i, j]*xa[j]$, no caso desconsiderando os elemento da diagonal principal, as tarefas que são responsáveis pela linha i de A devem fazer suas subtrações (preferencialmente por uma redução) das multiplicações realizadas, gerando assim o novo valor de $x[i]$, levando em conta que $x[i]$ tinha recebido o valor de $b[i]$ anteriormente. Esse processo pode ser verificado para um linha i conforme a fig. 4.

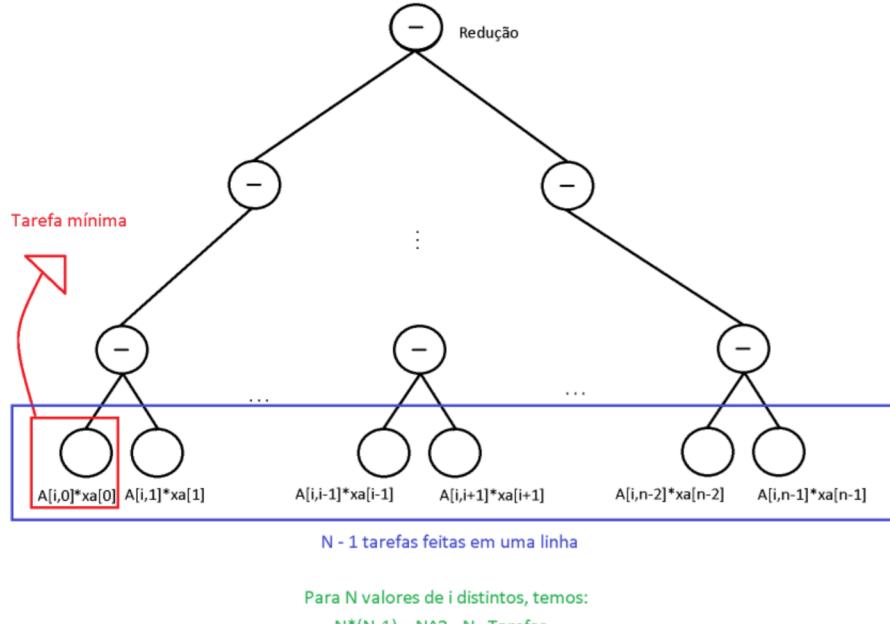


Figura 4: Redução e particionamento - Método Iterativo de Jacobi-Richardson

Também podemos demonstrar o comportamento para mais de uma linha, conforme a fig. 5.

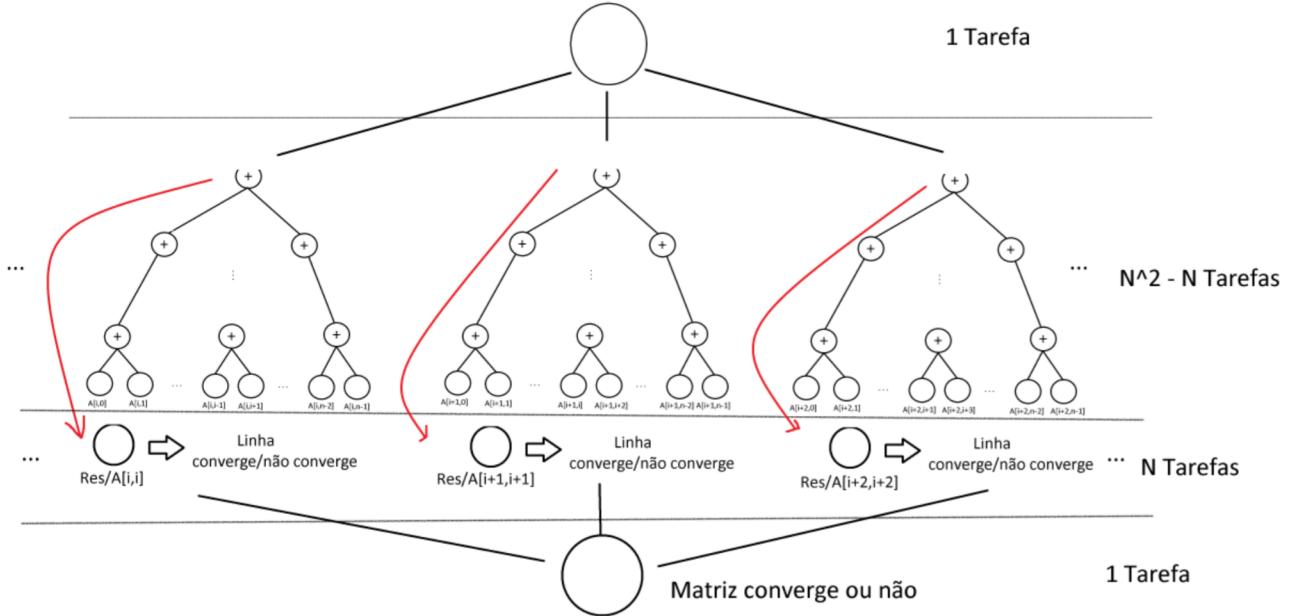


Figura 5: Redução e particionamento para a convergência de várias linhas - Método Iterativo de Jacobi-Richardson.

Tendo realizado a redução é preciso dividir cada $x[i]$ pelo respectivo valor da diagonal principal de A ($A[i, i]$). Esse processo pode ser dividido em N tarefas, cada uma com um $x[i]$ diferente e seu respectivo $A[i, i]$.

Nesse momento é preciso verificar se a aproximação já é boa suficiente. Para isso, precisaremos dos valores dos vetores x e xa e o limiar escolhido. O processo que será feito segue a fig. 6.

$$D_r = \frac{\max\{|x_i^{(k+1)} - x_i^{(k)}|, i = 1, \dots, n\}}{\max\{|x_i^{(k+1)}|, i = 1, \dots, n\}} \leq \tau,$$

em que $\tau > 0$ é uma certa tolerância.

Figura 6: Critério de parada do algoritmo.

Cria-se $2N$ tarefas, duas para cada valor de x. Dividiremos em dois grupos de tarefas de tamanho N. Cada tarefa do primeiro grupo recebe um valor de $x[i]$ e seu respectivo $xa[i]$ e calcula o valor absoluto da diferença desses valores, depois é feita uma função de máximo dos resultados de cada tarefa (preferencialmente por uma redução). E assim, encontramos o numerador da verificação de

parada.

Cada tarefa do segundo grupo recebe um valor de $x[i]$ e calcula o valor absoluto deste, depois é feita uma função de máximo dos resultados de cada tarefa (preferencialmente por uma redução). E assim, encontramos o denominador da verificação de parada. É importante salientar que todas essas tarefas podem ser feitas em paralelo, inclusive tarefas de grupos diferentes.

Caso o valor de verificação encontrado seja maior do que o limiar escolhido, vai para a próxima iteração do método. Caso contrário, chegamos nos valores de X requeridos dentro do limiar escolhido anteriormente.

1.2 Comunicação

Essa etapa visa identificar a comunicação e sincronização entre as tarefas. As comunicações entre as tarefas visam prover os dados de entrada para tarefas e posteriormente recuperar os resultados das computações realizadas.

1.2.1 Convergência do método

Inicialmente, cada tarefa criada receberá o seu respectivo valor da matriz A. Após será feita uma comunicação por meio de uma soma por redução, onde cada tarefa passa seu valor para uma tarefa superior, seus valores são somados e esse processo se repete até chegar em um valor final. Tendo a subtração de cada linha, sobram N tarefas finais (uma para cada linha), as quais receberão os respectivos valores de $A[i,i]$ para fazerem suas divisões. Tendo as divisões feitas e as verificações de convergência por linha, acontece mais uma comunicação, onde todas as convergências por linha são passadas para uma tarefa que assimila se a matriz converge no todo (caso todas as linhas convergirem).

1.2.2 Método Iterativo de Jacobi-Richardson

Já para o caso do método iterativo cada tarefa recebe o seu elemento da matriz A e as tarefas responsáveis pelos dados do vetor xa também receberão seus elementos. Cada tarefa, inicialmente faz a multiplicação $A[i,j]*xa[j]$ e nesse momento acontecerá a outra comunicação por meio de uma subtração por redução, onde cada tarefa passa seu valor para uma tarefa superior, seus valores são subtraídos e esse processo se repete até chegar em um valor final. Após isso, sobrarão N tarefas (uma para cada linha), que por outra comunicação recebem os respectivos valores da diagonal principal

de A para dividir o valor da respectiva redução e assim encontrar o valor final de $x[i]$. E essas são as comunicações necessárias para encontrar os novos valores de x.

Para fazer a cópia de x em x_a , são passadas para cada uma das N tarefas um valor $x[i]$ de x e seu valore é copiado para x_a .

1.3 Aglomeração

Essa etapa visa aglomerar as tarefas em processos, com objetivo de reduzir o overhead de comunicação. Nesse caso, estamos usando uma máquina MIMD com memória compartilhada.

1.3.1 Convergência do método

Considerando a plataforma alvo do algorítimo, deve-se agrupar as $N^2 - N$ tarefas em P processos, onde p é o número de elementos processadores lógicos. Assim, a granularidade será engrossada para diminuir a quantidade de comunicações necessárias entre os processos. Tendo cada processo recebido os $(N^2 - N)/P$ dados, se a divisão não for exata, os processos excedentes devem ser atribuídos ao último processo disponível. Após isso a redução será feita, sendo que o ideal é que todas as tarefas referentes a uma linha i estejam em um mesmo processo, caso não caiba é necessário colocar em outro processo e juntado posteriormente. Tendo a soma final por linha de cada redução, é possível fazer uma redução *and* binária sendo que as linhas que convergiram entram com 1 e as que não convergiram entram com 0. Desta forma, o valor dessa redução será 1 se todas convergirem e 0 caso uma das linhas não convirja.

1.3.2 Método Iterativo de Jacobi-Richardson

Nessa parte, deve-se agrupar as $N^2 - N$ tarefas em P processos, onde p novamente é o número processadores lógicos, engrossando a granularidade para diminuir a quantidade de comunicações necessárias. Tendo cada processo recebido os $\frac{N^2 - N}{P}$ dados, se a divisão não for exata, os processos excedentes devem ser atribuídos ao último processo disponível.

Para a operação de dividir cada o resultado encontrado, que foi armazenado em $x[i]$, pelo respectivo valor da diagonal principal de A ($A[i, i]$), são usadas N tarefas que são agrupadas para P processos. Assim, cada processo ficará encarregado de $\frac{N}{P}$ dados, novamente caso a divisão não seja exata as computações excedentes serão atribuídos ao último processo disponível.

Após isso, para as tarefas de verificação do critério de parada atribuiremos as $2N$ tarefas á P

processos, sendo que os máximos locais serão calculados pelos processos e, no final, acontecerá a redução para encontrar o máximo global. É importante salientar que para diminuir a comunicação, é indicado que um valor do iterador i seja passado apenas para um processo, sendo que esse processo deve fazer todos os cálculos que usam essa posição nos vetores.

1.4 Mapeamento

Para uma máquina MIMD de memória compartilhada o ideal seria atribuir um processo para cada processador, mas, nesse tipo de máquina, o escalonamento de tarefas nos processadores é feito pelo sistema operacional, desta forma, não temos controle.