Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
SSC0143 - Programação Concorrente

Atividade: Modelagem do Método de Gauss-Jordan

Profº: Paulo Sérgio Lopes de Souza

Alunos:

Bernardo Simões Lage Gomes Duarte (8598861)

Giovani Ortolani Barbosa (8936648)

Jorge Luiz da Silva Vilaça (9066491)

Luiz Augusto Vieira Manoel (8937308)

Setembro de 2017

Introdução

Neste projeto, utilizamos a modelagem PCAM para modelar uma solução paralela para o método de Gauss-Jordan, utilizado para escalonar matrizes. Este método consiste em aplicar operações elementares à matriz aumentada de um sistema de equações lineares, de forma a deixá-la na forma escalonada reduzida. Seu objetivo é deixar o sistema em uma forma que apresenta solução imediata, sem que haja necessidade de realizar substituições.

Particionamento:

Considerando uma matriz aumentada de um sistema de equações:

1.Iniciar as variáveis ‘c’ e ‘l’ com 0.
2.Enquanto ‘c’ for menor que o número de colunas (ignorando coluna da matriz aumentada):
    2.1.Verificar se na coluna ‘c’ há um elemento não nulo, preferencialmente um ‘1’, em uma linha ainda não pivotada.
    2.2.Se houver, trocar a linha com elemento não nulo ‘j’ pela linha ‘l’, caso ‘j’ e ‘l’ sejam diferentes. Se não, incrementar ‘c’ e voltar para a etapa 2.
    2.3. Pivotar a linha ‘l’, caso o elemento da coluna ‘c’ e linha ‘l’ seja diferente de ‘1’, e incrementar ‘l’.
    2.4.Através de operações elementares matriciais, zerar os demais elementos não nulos da coluna ‘c’ e então incrementar ‘c’.
3.Se houverem variáveis sem um pivô associado, tomá-las como variáveis livres. As variáveis com pivôs associados podem ter seus valores dependentes das variáveis livres.
4.Montar o conjunto solução.

Exemplo:
c = 0, l = 0. Na coluna c há um elemento não nulo.

$$\left[\begin{array}{ccc|c} 5 & 5 & 0 & 15 \\ 2 & 4 & 1 & 10 \\ 3 & 4 & 0 & 11 \end{array}\right]$$

Pivotou a linha l. *l = 1.*

$$\left[\begin{array}{ccc|c} 1 & 1 & 0 & 3 \\ 2 & 4 & 1 & 10 \\ 3 & 4 & 0 & 11 \end{array}\right]$$

Zerou os demais elementos da coluna c subtraindo a linha pivotada multiplicada pelo elemento a ser zerado. *c = 1.*

$$\left[\begin{array}{ccc|c} 1 & 1 & 0 & 3 \\ 0 & 2 & 1 & 4 \\ 0 & 1 & 0 & 2 \end{array}\right]$$

Na coluna c há um elemento ‘1’ em uma linha não pivotada. A linha 1 e 2 foram trocadas. Não foi necessário pivotar a linha 1. *l = 2.*

$$\left[\begin{array}{ccc|c} 1 & 1 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 2 & 1 & 4 \end{array}\right]$$

Zerou os demais elementos da coluna c subtraindo a linha pivotada multiplicada pelo elemento a ser zerado. c = 2.
Na coluna c há um elemento ‘1’ em uma linha não pivotada. Não foi necessário pivotar a linha l, nem zerar os outros elementos da coluna ‘c’.

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 \end{array}\right]$$

Não ocorreram variáveis livres.
Por fim foi montado o conjunto solução.

$$x_1 = 1,$$
$$x_2 = 2,$$
$$x_3 = 0.$$

Comunicação:

A paralelização deste algoritmo se dá nas operações aritméticas, que são sempre feitas em todos os elementos de uma linha. Essas operações podem ser feitas simultaneamente, dado uma etapa de comunicação para que os processos conheçam o valor a ser operado.

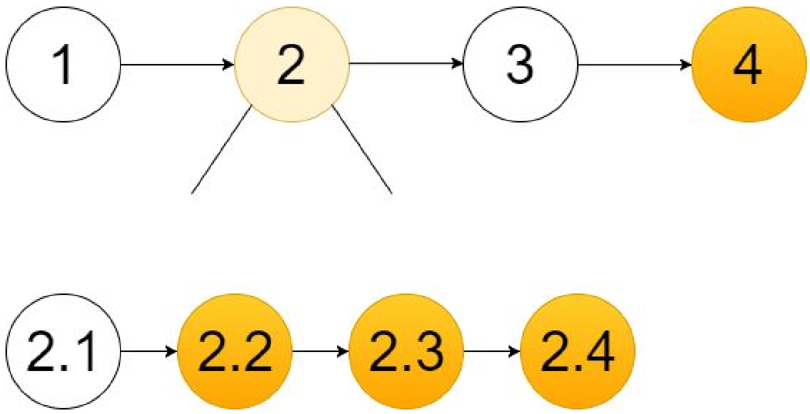
As trocas de linhas, feitas na etapa 2.2 do particionamento podem ser totalmente paralelizadas, sem necessidade de comunicação, apenas sincronização para garantir que todos os elementos tenham sido trocados.

As pivotações, feitas na etapa 2.3, requerem que o elemento pivotador (que deverá ser ‘1’) seja comunicado do processo responsável por este para os demais. Após esta etapa de comunicação, que pode feita com map-reduce, a linha pode ser pivotada totalmente em paralelo, novamente necessitando de uma sincronização após o término.

Para zerar os outros elementos da coluna do pivô, etapa 2.4, o processo é semelhante ao de pivotação. Um processo informa aos demais, via map-reduce, qual o elemento da coluna e os processos devem reter os valores dos elementos da linha pivotada da última pivotação. Assim, as operações aritméticas podem ser feitas totalmente em paralelo, sendo necessária uma sincronização após o feito.

É importante ressaltar que, para cada incremento de ‘c’, as colunas de valor menor que ‘c’ não são alteradas, logo, não é necessário atribuir processos para iterar sobre todos os elementos da linha. Além disto, nas instâncias onde há comunicação entre processos, após o envio da mensagem, o processo pode realizar suas operações, sem a necessidade de aguardar o recebimento das mensagens.

Por fim, cada conversão de linha da matriz final para parte do conjunto solução criado na etapa 4 pode ser feita por um processo.

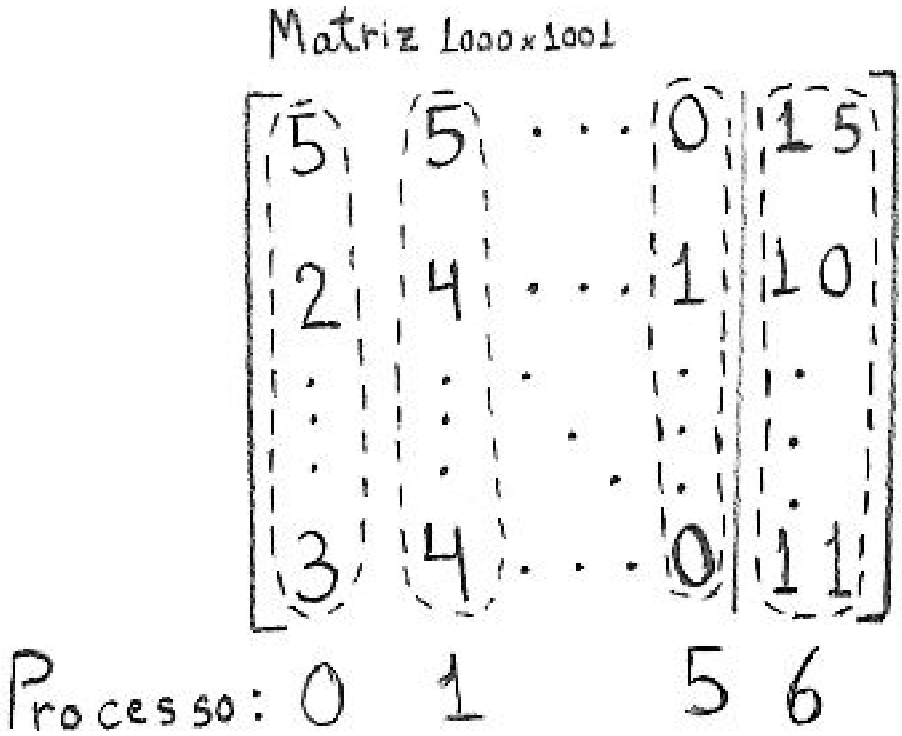


Aglomeração:

A partir da etapa de comunicação, podemos aglomerar as tarefas em processos. Aos processos são atribuídas colunas: em uma matriz NxM, com P processos, sendo M > P, o processo ‘p’ ficaria com as colunas onde o resto da divisão de M por P é ‘p’. A partir do momento em que o algoritmo iterar um número de elementos de cada linha menor que ‘p’, é possível eliminar processos até que sobrem N processos, caso ‘p’ seja maior que N.

Mapeamento:

Assumindo uma matriz 1000x1001, um processador com quatro núcleos e outro com três, totalizando sete processos:



A distribuição de colunas aos processos segue a fórmula:

$$p_j = c \% p$$

Onde ‘p_j’ é o índice do processo que vai cuidar da coluna ‘c’ e ‘p’ é o número de processos.