SZYBKI ALGORYTM Z MACIERZĄ SHURA DLA UKŁADÓW TRÓJDIAGONALNYCH.

(Dla uproszczenia opis algorytmu dla układu o stałych współczynnikach)

Układ równań

$$\textbf{(1)} \qquad A\underline{x} = \underline{f}.$$

Macierz A zapisana blokowo
$N+1$ bloków kwadratowych
wymiaru $R+1$

$$M = (N+1)(R+1)$$

$$M \ll N$$

$$\textbf{(2)} \qquad A = \begin{bmatrix} D & B & \cdot & \cdot & \cdot & \cdot & \cdot \\ A & D & B & \cdot & \cdot & \cdot & \cdot \\ \cdot & A & D & B & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & A & D \end{bmatrix}$$

Macierz A w postaci oryginalnej.

$$A = \begin{bmatrix} d & b & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a & d & b & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & a & d & b & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & a & d \end{bmatrix}$$

Wektory $\underline{x}$ i $\underline{f}$ dostosowane do rozkładu blokowego macierzy A:

$$\underline{x} = [\underline{x}_0, \underline{x}_1, \cdots, \underline{x}_N]^T$$
$$\underline{f} = [\underline{f}_0, \underline{f}_1, \cdots, \underline{f}_N]^T$$

Układ w postaci blokowej po rozpisaniu:

$$D\underline{x}_0 + B\underline{x}_1 = \underline{f}_0 \tag{3}$$

$$A\underline{x}_{j-1} + D\underline{x}_j + B\underline{x}_j = \underline{f}_j, \quad j = 1, 2, \cdots, N-1,$$

$$A\underline{x}_{N-1} + D\underline{x}_N = \underline{f}_N$$

Mamy $N + 1$ procesorów, ponumerowanych $0, 1, \cdots N$. Każdemu procesorowi jest przyporządkowane jedno równanie blokowe, które ten procesor będzie obsługiwał.

PIERWSZY KROK ALGORYTMU

(4) $$D\underline{\tilde{x}}_j = \underline{f}_j$$

$\underline{\tilde{x}}_j$, to pierwsze przybliżenie wektora $\underline{x}_j$,

(5) $$D\underline{w}_0 = \underline{e}_0,$$

(6) $$D\underline{w}_R = \underline{e}_R,$$

$$\underline{e}_k,\ 0 \leqslant k \leqslant R$$

wersory osi współrzędnych.

Zauważmy, że dla każdego j, $0 \leqslant j \leqslant N$,

$$A = a\underline{e}_0 \ \underline{e}_R^T,$$

$$B = b\underline{e}_R \ \underline{e}_0^T,$$

"POPRAWKI"

(7) $$\underline{x}_0 = \underline{\tilde{x}}_0 - (b\underline{w}_R) \ \underline{e}_0^T \underline{x}_1$$

$$\underline{x}_j = \underline{\tilde{x}}_j - (a\underline{w}_0) \ \underline{e}_R^T \underline{x}_{j-1} - (b\underline{w}_R) \ \underline{e}_0^T \underline{x}_{j+1}, \ 1 \leqslant j \leqslant N-1$$

$$\underline{x}_N = \underline{\tilde{x}}_N - (a\underline{w}_0) \ \underline{e}_R^T \underline{x}_{N-1}$$

DRUGI KROK ALGORYTMU

Tworzenie macierzy Shura w każdym procesorze

Opis dla $N = 4$.

(8)

$$
\begin{array}{lll}
 & \underline{e}_R^T| & \underline{x}_0 = \tilde{\underline{x}}_0 - b\underline{w}_R[\underline{e}_0^T\underline{x}_1]_0 \\
\underline{e}_0^T| & \underline{e}_R^T| & \underline{x}_1 = \tilde{\underline{x}}_1 - a\underline{w}_0[\underline{e}_R^T\underline{x}_0]_1 - b\underline{w}_R[\underline{e}_0^T\underline{x}_2]_2 \\
\underline{e}_0^T| & \underline{e}_R^T| & \underline{x}_2 = \tilde{\underline{x}}_2 - a\underline{w}_0[\underline{e}_R^T\underline{x}_1]_3 - b\underline{w}_R[\underline{e}_0^T\underline{x}_3]_4 \\
\underline{e}_0^T| & \underline{e}_R^T| & \underline{x}_3 = \tilde{\underline{x}}_3 - a\underline{w}_0[\underline{e}_R^T\underline{x}_2]_5 - b\underline{w}_R[\underline{e}_0^T\underline{x}_4]_6 \\
\underline{e}_0^T| & & \underline{x}_4 = \tilde{\underline{x}}_4 - a\underline{w}_0[\underline{e}_R^T\underline{x}_3]_7
\end{array}
$$

Zmienne UKŁADU SHURA

$$z_s,\ 0 \leqslant s \leqslant 2N - 1 = 7,$$

$$z_s = [\cdots]_s, 0 \leqslant s \leqslant 2N - 1 = 7.$$

UKŁAD SHURA

$$\Sigma \underline{z} = \underline{\boldsymbol{F}} \tag{9}$$

$$\begin{bmatrix} h & 1 & & & & & & \\ 1 & p & q & & & & & \\ & g & h & 1 & & & & \\ & & 1 & p & q & & & \\ & & & g & h & 1 & & \\ & & & & 1 & p & q & \\ & & & & & g & h & 1 \\ & & & & & & 1 & p \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \\ z_7 \end{bmatrix} = \begin{bmatrix} \underline{e}_R^T \tilde{\underline{x}}_0 \\ \underline{e}_0^T \tilde{\underline{x}}_1 \\ \underline{e}_R^T \tilde{\underline{x}}_1 \\ \underline{e}_0^T \tilde{\underline{x}}_2 \\ \underline{e}_R^T \tilde{\underline{x}}_2 \\ \underline{e}_0^T \tilde{\underline{x}}_3 \\ \underline{e}_R^T \tilde{\underline{x}}_3 \\ \underline{e}_0^T \tilde{\underline{x}}_4 \end{bmatrix}.$$

ELEMENTY MACIERZY SHURA

$g = a\underline{w}_{R,0}, \quad h = b\underline{w}_{R,R}$

$p = a\underline{w}_{0,0}, \quad q = b\underline{w}_{0,R}$

gdzie:

$$\underline{w}_0 = [w_{0,0}, w_{0,1}, \cdots, w_{0,R}]^T$$

i

$$\underline{w}_R = [w_{R,0}, w_{R,1}, \cdots, w_{R,R}]^T.$$

POPRAWKI

$$\underline{x}_0 = \underline{\tilde{x}}_0 \qquad\qquad - b\underline{w}_R z_0$$

$$\underline{x}_1 = \underline{\tilde{x}}_1 - a\underline{w}_0 z_1 - b\underline{w}_R z_2$$

$$\underline{x}_2 = \underline{\tilde{x}}_2 - a\underline{w}_0 z_3 - b\underline{w}_R z_4$$

$$\underline{x}_3 = \underline{\tilde{x}}_3 - a\underline{w}_0 z_5 - b\underline{w}_R z_6$$

$$\underline{x}_4 = \underline{\tilde{x}}_4 - a\underline{w}_0 z_7$$

TRZECI KROK ALGORYTMU

Trzeba w każdym procesorze skompletować całe rozwiązanie UKŁADU SHURA. Na szczęście można to zrobić przy pomocy jednego rozkazu systemu MPI.
Oto jeden ze sposobów. Ponieważ w każdym procesorze (np. o numerze j) mamy obliczony wektor

$$\tilde{\underline{x}}_j,$$

to mamy częśc wektora prawej strony. Rozpatrzmy nasz przykład dla $N = 4$, wtedy UKŁAD SHURA jest wymiaru 8×8

1. jeśli $j = 0$ to tworzymy wektor

$$\boldsymbol{F}_0 = [\underline{\boldsymbol{e}}_{\boldsymbol{R}}^{\boldsymbol{T}} \tilde{\boldsymbol{x}}_0, 0, 0, 0, 0, 0, 0, 0]^{\boldsymbol{T}}$$

2. jeśli $j = 4$ to tworzymy wektor

$$\boldsymbol{F}_4 = [0, 0, 0, 0, 0, 0, 0, \underline{\boldsymbol{e}}_{\boldsymbol{R}}^{\boldsymbol{T}} \tilde{\boldsymbol{x}}_4]^{\boldsymbol{T}}$$

3. jeśli np. $j = 2$ to tworzymy wektor

$$\boldsymbol{F}_j = [0, 0, 0, \underline{\boldsymbol{e}}_{\boldsymbol{0}}^{\boldsymbol{T}} \tilde{\boldsymbol{x}}_2, \boldsymbol{e}_{\boldsymbol{R}}^{\boldsymbol{T}} \tilde{\boldsymbol{x}}_2, 0, 0, 0]^{\boldsymbol{T}}$$

4. i.t.d.

Teraz w każdym procesorze rozwiązujemy UKŁAD SHURA

$$\Sigma \boldsymbol{y_j} = \boldsymbol{F_j}.$$

Następnie w każdym procesorze wykonujemy komendę "Allreduce" z opcją "SUM" dla wektorów y_j. Ze względu na liniową zleżność rozwiązania układu od jego prawej strony, we wszystkich procesorach, na miejscu wektora y_j, pojawi się pełne rozwiązanie $\underline{z}$ UKŁADU SHURA. W końcu, w każdym z procesorów, przy pomocy odpowiedniej poprawki, znajdujemy wektor $\underline{x_j}$, współrzędną dokładnego rozwiązania naszego układu równań.

==========

Taki algorytm po raz pierwszy zastosował i opisał STEFAN BONDELI z ZÜRICHU. "Divide and Conquer " Lecture Notes in Computer Science vol.457 (1990).