Institut für Numerische Methoden und Informatik im Bauwesen
Prof. Dr.-Ing. habil. Lutz Lämmer (apl.)

Hochleistungssimulationen im Ingenieurwesen (HSI)
Wintersemester 2018/19

TECHNISCHE UNIVERSITÄT DARMSTADT

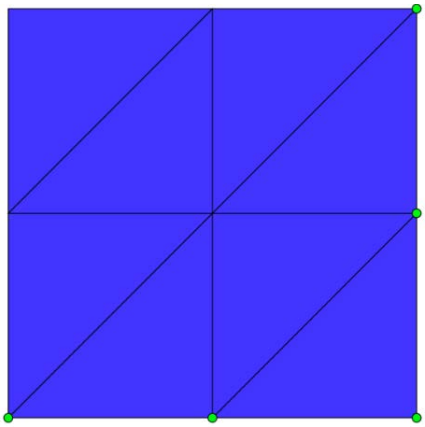
Hausübung 3 – PCG-Solver

21.01.2019

Übungsabgabe: Fr. 08.02.2019 (23:55 Uhr)

Einführung

Gegeben ist ein FEM-Netz, welches aus 8 Dreieckselementen besteht:



Gegeben ist ein lineares Gleichungssystem der Form $\boldsymbol{K}v = f$ für eine Berechnung nach der Finite Elemente Methode. Die Matrix $\boldsymbol{K}$ ist symmetrisch und dünnbesetzt.

$$\begin{bmatrix} 2.0 & & & & & & & & \\ & 2.0 & -0.5 & & -1.0 & & & & \\ & -0.5 & 1.0 & & & -0.5 & & & \\ & & & 3.0 & & & & & \\ & -1.0 & & & 4.0 & -1.0 & & & \\ & & -0.5 & & -1.0 & 2.0 & & & \\ & & & & & & 1.0 & & \\ & & & & & & & 3.0 & \\ & & & & & & & & 2.0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \\ v_9 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{bmatrix}$$

Nach der Berücksichtigung der Randbedingungen nach Neumann, Robin und Dirichlet ergibt sich folgendes Gleichungssystem:

$$\begin{bmatrix} 1.0 & & & & & & & & \\ & 2.0 & -0.5 & & -1.0 & & & & \\ & -0.5 & 1.0 & & & -0.5 & & & \\ & & & 1.0 & & & & & \\ & -1.0 & & & 4.0 & -1.0 & & & \\ & & -0.5 & & -1.0 & 2.0 & & & \\ & & & & & & 1.0 & & \\ & & & & & & & 1.0 & \\ & & & & & & & & 1.0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \\ v_9 \end{bmatrix} = \begin{bmatrix} 0.0 \\ -0.25 \\ -0.125 \\ 0.0 \\ 0.5 \\ 0.75 \\ 0.0 \\ 0.5 \\ 1.0 \end{bmatrix}$$

Aufgabe

1.1. Sequentieller PCG-Solver

Implementieren Sie in Java einen iterativen Gleichungslöser nach dem vorkonditionierten CG-Verfahren (engl. Preconditioned Conjugate Gradient – PCG), vgl. Formel 1.

- Das Abbruchkriterium ε ist in Abhängigkeit von der initialen Vektornorm γ_0 zu wählen ($\gamma_0 \cdot$ 10E-7).
- Die Berechnung ist in doppelter Genauigkeit durchzuführen (64-bit Gleitkommazahlen).

– Formel 1 –

$$
\begin{aligned}
&\textit{wähle} \quad v_0 \\
&\qquad r_0 = f - K v_0 \\
&\qquad d_0 = C^{-1} r_0 \\
&\qquad \gamma_0 = d_0^T r_0 \\
&\textit{Iteration für } k = 1, 2, \ldots \\
&\qquad u = K d_{k-1} \\
&\qquad \alpha = \frac{\gamma_0}{d_{k-1}^T u} \\
&\qquad v_k = v_{k-1} + \alpha d_{k-1} \\
&\qquad r_k = r_{k-1} - \alpha u \\
&\qquad p = C^{-1} r_k \\
&\qquad \gamma_k = r_k^T p \qquad \rightarrow \textit{Abbruchkriterium } \gamma_k < \epsilon \\
&\qquad \beta = \frac{\gamma_k}{\gamma_{k-1}} \\
&\qquad d_k = p + \beta d_{k-1}
\end{aligned}
$$

1.2. Paralleler PCG-Solver

Entwickeln Sie aus Ihrer sequentiellen Lösung einen parallelen PCG-Gleichungslöser nach Formel 2 unter Verwendung der Java message passing Bibliothek MPJ-Express. Das Problem soll dabei auf mindestens zwei Prozessoren verteilt werden. Die Verteilung auf die Prozessoren bzw. Gebiete erfolgt additiv, so dass für die daraus resultierende Verteilung für $\boldsymbol{K}$ und f gilt $\sum(\boldsymbol{B}_i^T \boldsymbol{K}_i) v = \sum(\boldsymbol{B}_i^T f_i)$.

Tipps:

Ermittlung des Vorkonditionierers $\boldsymbol{C^{-1}}$ mit der Jakobi-Vorkonditionierung:

- Vorkonditionierer C^{-1} einmal vor Beginn der Iterationen aus den aufaddierten Hauptdiagonalelementen von $\boldsymbol{K}$ als absolut verteilter Operator erzeugen.
- Dazu extrahieren der Hauptdiagonalen als Vektor, superponieren und mit dem reziproken Wert ablegen.

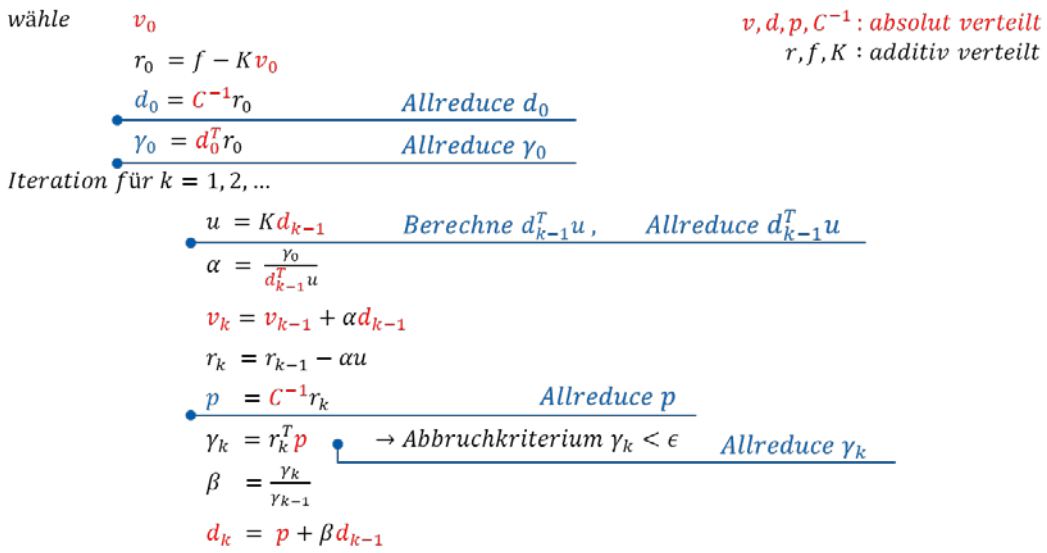
Berechnung der absoluten Skalarprodukte, z.B. $\boldsymbol{\gamma_0}$:

- Lokale Berechnung der Skalarprodukte, z.B. $d_0^T \cdot r_0$.
- Summenbildung über alle Domains (Allreduce).

Berechnung der absolut verteilten Vektoren, z.B. d_0:

- **Nach** der Anwendung der Vorkonditionierung muss der additiv verteilte Ergebnisvektor d_0 mit einer globalen Vektoraddition in einen absolut verteilten Vektor transformiert werden (globale Vektorsuperposition mithilfe von Allreduce oder Punkt-zu-Punkt Kommunikation zwischen benachbarten Teilgebieten für die jeweils betreffenden Vektoren erforderlich).

– Formel 2 –



1.3. Erwartete Ergebnisse

Zeigen Sie, dass die sequentielle und parallele Implementierung identische Ergebnisse liefern (Anzahl der Iterationen, Wert des Abbruchkriteriums, Ergebnisvektor v).

1.4. (Optional)

Mit dem in der Vorlesung demonstrierten Java FEM Programm (FEJavaDemo – s.u.) können weitere Problemstellungen (Tupel von $\boldsymbol{K}$ und f) erzeugt werden. Zeigen Sie, dass ihre Implementierung auch für diese Probleme korrekte Ergebnisse liefert.

Abgabe

Die Abgabe muss bis **Freitag, den 08.02.2019, 23:55 Uhr** in 2er-Gruppen über moodle erfolgen. Abzugeben sind (mindestens) der Quellcode des sequentiellen und parallelen PCG-Solvers sowie eine Dokumentation der Bearbeitung der Übung. Bitte fassen Sie die Dateien zum Upload in einem gepackten Dateiformat (*.zip, *.7z, *.tar) zusammen.

Weiterführende Links

- An Introduction to the Conjugate Gradient Method Without the Agonizing Pain
  http://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf
- FEJavaDemo, Demonstrationsanwendung zum Lösen von Wärmeleitproblemen mit FEM
  http://aurelienboffy.fr/FEJavaDemo/de_doc.pdf
  http://aurelienboffy.fr/FEJavaDemo/FEJavaDemo.zip
- MPJ-Express
  http://mpj-express.org/
- LÄMMER, Lutz, 1997.
  Parallelisierung von Anwendungen der Finite-Element-Methode im Bauingenieurwesen (über moodle)