Technische Mechanik, Band 14, Heft 1, (1994)
Manuskripteingang: 10. Januar 1994

Ein Programmkonzept zur Parallelisierung von FEM-Algorithmen

G. Blanke, W. Weese

Für die Finite-Elemente-Methode eröffnen sich durch Parallelrechner neue Aspekte, die sowohl kürzere Rechenzeiten als auch erhöhte Speicherplatzkapazitäten bedeuten. Das erforderliche neue Programmierkonzept, Lösungsstrategien und ein Rahmenprogramm werden vorgestellt.

1 Einleitung

Seit dem Auftreten der ersten elektronischen Rechenanlage, Anfang der fünfziger Jahre, hat sich die Rechenleistung solcher Anlagen etwa alle 3,5 Jahre verzehnfacht. Das zog wiederum einen immensen Preisverfall der Hardware nach sich. Somit ist es in der gegenwärtigen Zeit nicht nur den großen Firmen vorbehalten, hochwertige Rechenanlagen zu erwerben. Damit erlangt die Entwicklung von Programmen auf Mehrprozessormaschinen mit verteiltem Speicher (distributed memory) durch die Verfügbarkeit der entsprechenden Rechentechnik eine immer größere Bedeutung.

Instructions \ data	single	multiple
single	SISD	SIMD
multiple	MISD	MIMD

Bild 1. Taxonomie nach Flynn (1992)

Der Aufbau (Bild 1) eines Parallelrechners mit 128 Prozessoren (INMOS Prozessoren T805, 4MB RAM, 30 MHZ) entspricht dem eines MIMD (multiple instruction stream multiple data stream). Er ist dadurch gekennzeichnet, daß dasselbe Programm auf allen Prozessoren läuft. Der GC 128 zählt zu den Transputern. Ein Transputerbaustein besteht aus einem Prozessor, einem Coprozessor, einem Speicher (RAM) und einer Kommunikationseinheit, die mit einer endlichen Zahl von Anschlüssen (links) ausgerüstet ist. Für die FEM eröffnen sich mit der Einführung von Parallelrechnern neue Perspektiven. Auf der einen Seite verkürzt sich die Rechenzeit und andererseits bekommt man mit dem verteilten Speicher einen enormen Speicherplatzzuwachs.

Die Entwicklung eines FEM-Programms auf einem Parallelrechner erfordert ein neues Programmkonzept. Dieses Konzept (Bild 2) sollte folgende Punkte berücksichtigen:

- Gleichmäßiges Verteilen des generierten Netzes (load balancing)
- Benachbarte Substrukturen auf benachbarte Prozessoren
- Minimierung der Kommunikation
- Minimierung der externen Knoten
- Geeignete Wahl des Gleichungslösers
- Hardwareunabhängigkeit
- Implementieren geeigneter Schnittstellen

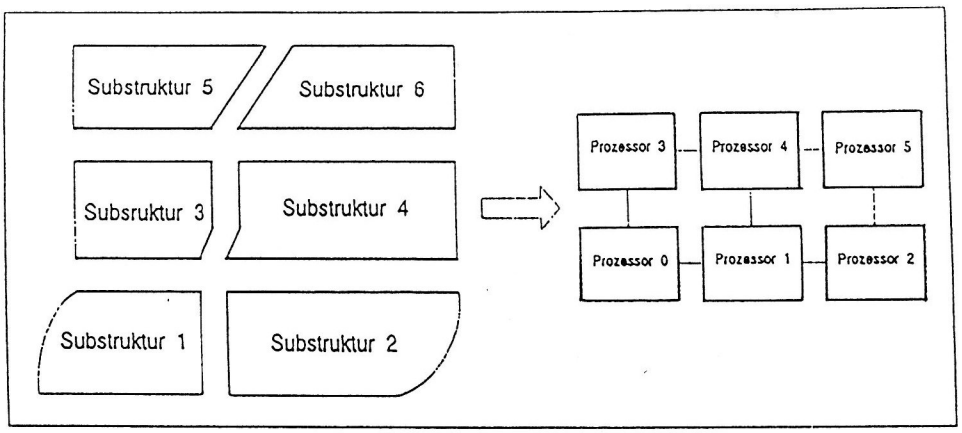

Bild 2. Aufteilung der Substrukturen auf die Prozessoren

In der Literatur findet man mehrere Ansätze, die zu einer Lösung von FEM-Problemen mit Hilfe der parallelen Rechentechnik führen (Meyer, 1990; Langer, 1991; Haase, 1991; Meißner und Lämmer, 1992; Ratke,1993). Der vorliegende Beitrag soll als Einführung der Substrukturtechnik in die parallele Rechentechnik dienen. Die Finite-Elemente-Substrukturtechnik geht auf die 60er Jahre zurück. Sie basiert auf einer nichtüberlappenden Gebietszerlegung, wobei das globale Problem auf ein Schurkomplement zur Bestimmung der Unbekannten auf den Koppelrändern zurückgeführt wird. Diese Technik läßt sich hervorragend auf Transputern implementieren.

2 Lösungsstrategien

In der Finite-Elemente-Methode wird eine Struktur in endliche große Elemente unterteilt. Für jedes dieser Elemente wird ein lokales Koordinatensystem eingeführt. Die Komponenten des Verschiebungsvektors werden durch lokale Ansatzfunktionen interpoliert. Wird der Finite-Element-Ansatz in eine Formulierung des Randwertproblems eingesetzt, so entsteht die Elementsteifigkeitsbeziehung. Nach deren Integration erhält man ein lineares algebraisches Gleichungssystem

$$\mathbf{K}_e \mathbf{v}_e = \mathbf{f}_e \tag{1}$$

$\mathbf{K}_e$ ist die Elementsteifigkeitsmatrix , $\mathbf{v}_e$ der gesuchte Verschiebungsvektor und $\mathbf{f}_e$ die rechte Seite oder der Lastvektor. Das Zusammenfassen aller Elementsteifigkeitsmatrizen, Knotenverschiebungsvektoren und Lastvektoren führt zu einem Gleichungssystem, das im allgemeinen großdimensioniert und schwachbesetzt ist. Die Gesamtsteifigkeitsmatrix $\mathbf{K}$ weist eine Bandstruktur auf. In der Substrukturtechnik werden die Elementsteifigkeitsmatrizen verschiedenen Substrukturen zugeordnet (Bild 3).

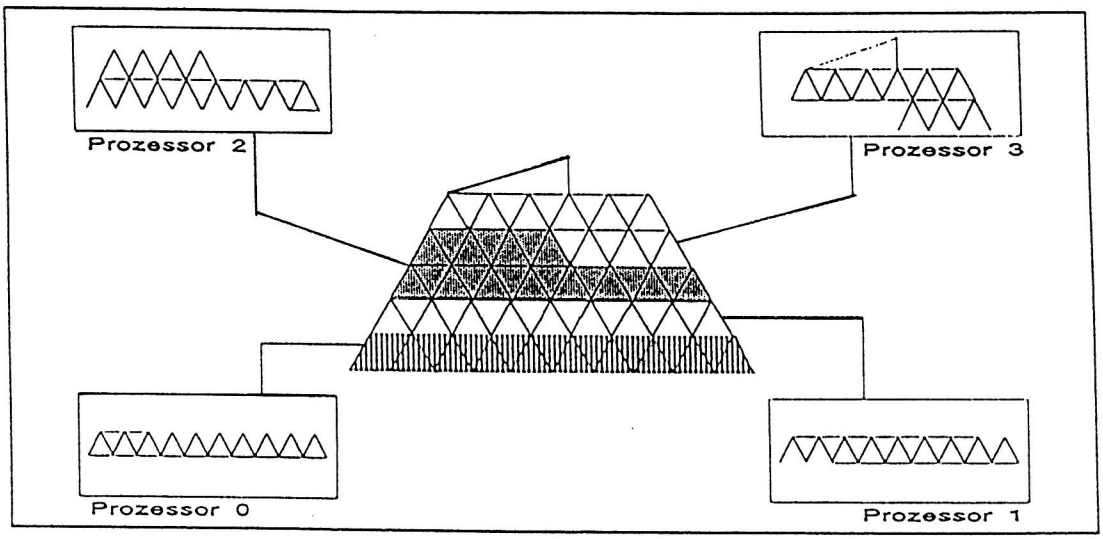

Bild 3. Mappen eines Balkentragwerkes

Die daraus resultierenden Gleichungssysteme sind Teile des Gesamtsystems. Beim Einarbeiten der Elementsteifigkeitsbeziehungen in die Substrukturen ordnet man nach lokalen L und externen E Knoten. Über externen Knoten sind die einzelnen Substrukturen miteinander gekoppelt.

$$\begin{bmatrix} \mathbf{K}_{LL} & \mathbf{K}_{LE} \\ \mathbf{K}_{EL} & \mathbf{K}_{EE} \end{bmatrix} \begin{bmatrix} \mathbf{v}_L \\ \mathbf{v}_E \end{bmatrix} = \begin{bmatrix} \mathbf{f}_L \\ \mathbf{f}_E \end{bmatrix} \tag{2}$$

Die Substrukturtechnik ist ein geeignetes Verfahren, FEM-Probleme auf Parallelrechnern zu verarbeiten. Dabei werden jedem Prozessor eine oder mehrere Substrukturen zugeordnet. Für die Lösung von Gleichungssystemen unterscheidet man zwei Verfahren, die direkten und die iterativen (Bild 4).

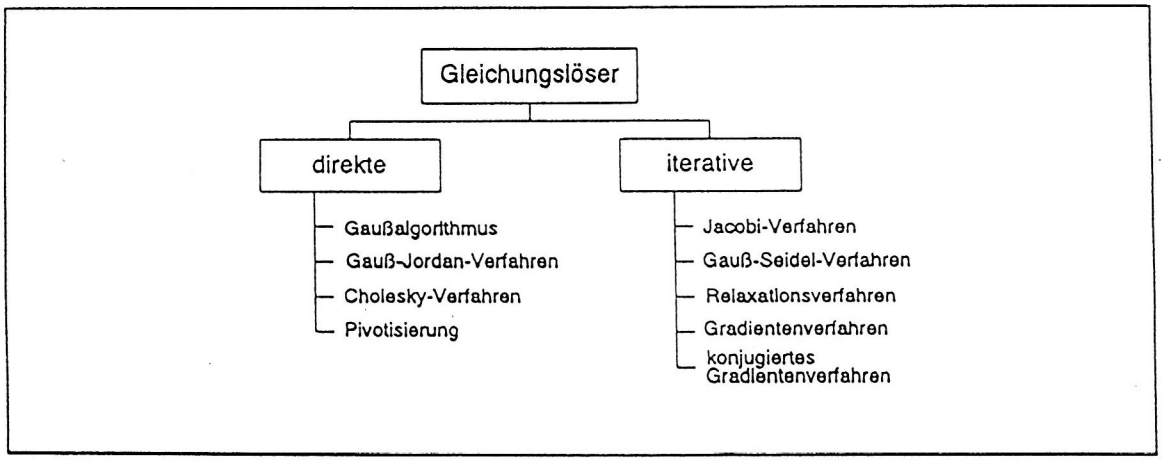

Bild 4. Gleichungslöser

Zu den direkten Lösungsverfahren gehören die Gaußschen Eliminationsverfahren mit Pivotisierung sowie die Verfahren die auf eine Faktorisierung der Systeme zurückgehen. Bei positiv definiten Matrizen ist die LU-Faktorisierung sehr verbreitet. Ist die Matrix zusätzlich noch symmetrisch, kann das Cholesky-Verfahren eingesetzt werden.

Zu den iterativen Verfahren gehören die schon seit langem bekannten Gauß-Seidel-Iterationsverfahren, das ω - Jacobi-Verfahren sowie das CG-Verfahren (konjugierte Gradienten). Leider konvergieren diese Verfahren im allgemeinen sehr schlecht. Eine Konvergenzverbesserung ist durch Beschleuniger, sogenannte Vorkonditionierer, zu erreichen. Die Idee der Vorkonditionierer liegt darin ein lineares Gleichungssystem so zu verändern, daß es sich leichter und schneller lösen läßt, wobei die Lösung identisch mit der des unveränderten Gleichungssystems ist. Systeme mit einer großen Streuung der Eigenwerte haben eine schlechte Konvergenz. Mit dem Vorkonditionierer soll ein Zusammenrücken des größten und kleinsten Eigenwertes der Koeffizientenmatrix erreicht werden. Für die Lösung von Gleichungssystemen auf Prallelrechnern werden zur gegenwärtigen Zeit fast nur iterative Verfahren und hier besonders das PCGM-Verfahren (parallel conjugate gradient method) eingesetzt. Für die Vorkonditionierung nutzt man verschiedene Verfahren (SSOR, ILU, Muligrid, EBE usw.). In Weiss (1993) findet man eine Gegenüberstellung von Vorkonditionierern (Diagonalisierung, Polynomialisierung, ILU, unvollständiger Cholesky usw.) beim Einsatz zur Lösung von Gleichungssystemen auf Parallelrechnern. Ein Vergleich der Vorkonditionierer hinsichtlich ihrer Robustheit, Vektorisierbarkeit und Parallelisierbarkeit brachte für keines der getesteten Vorkonditionierer optimale Ergebnisse. Einen weiteren Vergleich von Vorkonditionierern findet man in Boersma (1993) (Diagonalisierung, Polynomialisierung, EBE, ILU). Weitere Tests bewiesen, daß die Effizienz von Vorkonditionierern stark problemabhängig ist.

Wie die iterativen Verfahren sind auch die direkten Verfahren parallelisierbar. Einige Vorschläge zu parallelisierten Block-Varianten für Bandmatrizen von Dongorra (1987), LU-Zerlegung von Ristau und Schendel (1991) und Block-Varianten der LU-Zerlegung und der Gauß-Elimination von George (1989) und Schreiber (1966) existieren schon seit längerem, jedoch sind die diesbezüglichen Algorithmen hinsichtlich des Speicherbedarfs und des Rechenaufwandes mit den PPCG-Verfahren (parallel preconditioned conjugate gradient method) und dem Multigridverfahren nicht konkurrenzfähig.

Für das im Kapitel 2 vorgestellte Rahmenprogramm wurde als Gleichungslöser das PPCG-Verfahren, welches auf Law (1986) zurückgeht, für die Berechnung der Verschiebungen der Koppelknoten verwendet. Die Ermittlung der lokalen Verschiebungen erfolgt über die Berechnung des Schurkomplements. Das PPCG-Verfahren mit Schurkomplementbildung wurde von Meyer (1990) übernommen und modifiziert (Tabelle 1).

FOR i=1 TO p do parallel

PCG-Vorbereitung

0. Bestimme Startvektor x_i^0, B_i, C

1. $r_i^0 = f_i - K_i x_i^0$

2. $w^0 = C^{-1} \sum B_i^T r_i^0$ --Kommunikation

3. $w_i^0 = B_i\, w^0$

4. $q_i^0 = w_i^0$

5. $\gamma_i^0 = w_i^{0\,T} r_i^0$

6. $\gamma^0 = \sum \gamma_i$ --Kommunikation

Iteration

1. $v_i^n = K_i\, q_i^n$

2. $\delta_i^{n+1} = v_i^{n\,T} q_i^n$

3. $\delta^{n+1} = \sum \delta_i^{n+1}$ -- Kommunikation

4. $\alpha^{n+1} = -\gamma^n \,/\, \delta^{n+1}$

5. $x_i^{n+1} = x_i^n + \alpha^{n+1}\, q_i^n$

6. $r_i^{n+1} = r_i^n - \alpha^{n+1}\, v_i^n$

7. $w^{n+1} = C^{-1} \sum B_i^T r_i^{n+1}$ -- Kommunikation

8. $w_i^{n+1} = B_i\, w^{n+1}$

9. $\gamma_i^{n+1} = w_i^{n+1\,T} r_i^{n+1}$

10. $\gamma^{n+1} = \sum \gamma_i^{n+1}$ -- Kommunikation

11. $\beta^{n+1} = \gamma^{n+1} \,/\, \gamma^n$

12. $q_i^{n+1} = w_i^{n+1} + \beta^{n+1} q_i^n$

13. $\gamma^{n+1} \le \epsilon^2 * \delta^0$? nein -- goto1
ja -- stop

Tabelle 1. Paralleles PCG-Verfahren nach Meyer (1990)

Die Schurkomplementbildung ist nichts anderes als eine Reduzierung des Gleichungssystems (2) auf die externen Freiheitsgrade. Durch Elimination von $\mathbf{v}_L$ erhält man

$$\bar{\mathbf{K}}_E \mathbf{v}_E = \bar{\mathbf{f}}_E \tag{3}$$

mit dem Schurkomplement

$$\bar{\mathbf{K}}_E = \mathbf{K}_{EE} - \mathbf{K}_{LE}^T \mathbf{K}_{LL}^{-1} \mathbf{K}_{LE} \tag{4}$$

und den dazugehörigen Kraftvektor

$$\bar{\mathbf{f}}_E = \mathbf{f}_E - \mathbf{K}_{LE}^T \mathbf{K}_{LL}^{-1} f_L \tag{5}$$

Für die Berechnung der lokalen Freiheitsgrade nutzt man die Gleichung

$$\mathbf{v}_L = \mathbf{K}_{LL}^{-1} (\mathbf{f}_L - \mathbf{K}_{LE} \mathbf{v}_E) \tag{6}$$

Für die Berechnung der externen Freiheitsgrade wird das PPCG-Verfahren genutzt.
Eine Realisierung eines FE-Algorithmusses auf Parallelrechnern zur Lösung von nichtlinearen Problemen stellten Wriggers (1993) und Boersma (1993) vor. Einen umfassenden Überblick über parallele Gleichungslöser bietet Frommer (1990) an.

3 Entwicklung eines Rahmenprogramms

Die Lösung eines Problems nach der FE-Methode gliedert sich stets in die Teilaufgaben Preprozessing, Berechnung und Postprozessing. Bild 5 zeigt den Aufbau eines FE-Programms. Aufgrund dieser Einteilung besteht die Möglichkeit einige Module zu einem Rahmenprogramm zusammenzufassen. Dieses Rahmenprogramm sollte so konzipiert sein, daß unterschiedliche Elementtypen mühelos eingearbeitet werden können. Ein solcher Grundmodul (Bild 6) wurde auf dem GC 128 der Otto-von-Guericke-Universität Magdeburg erstellt. Eine erste Anwendung schlägt sich in dem Berechnungsprogramm PARSTAB für ebene Balkentragwerke nieder. Es benutzt den zuvor beschriebene Grundmodul, wobei als Rahmen die Programme GRAFEIN, VORBER, PSCGM, AUSGABE, AUSDAT und GRAFAUS gelten.

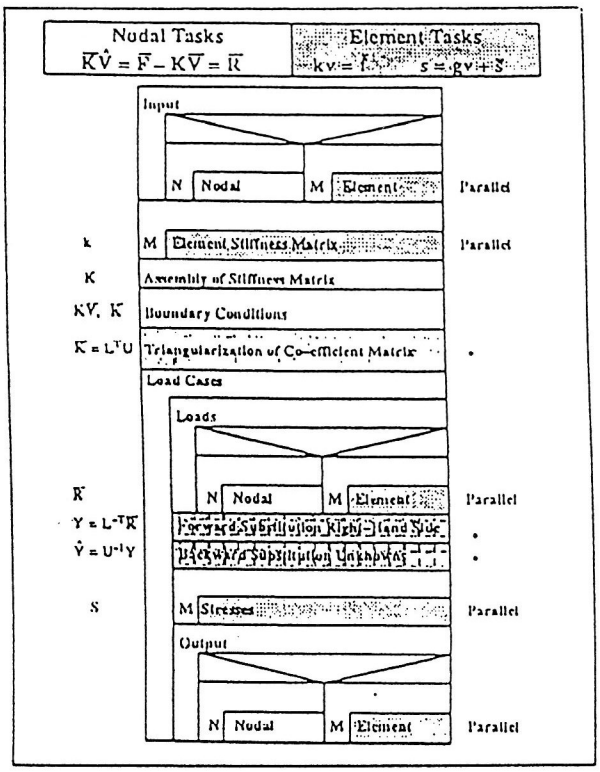

Bild 5. Struktogramm eines sequentiellen FEM-Algorithmus (Quelle: Meißner u. a., 1992)

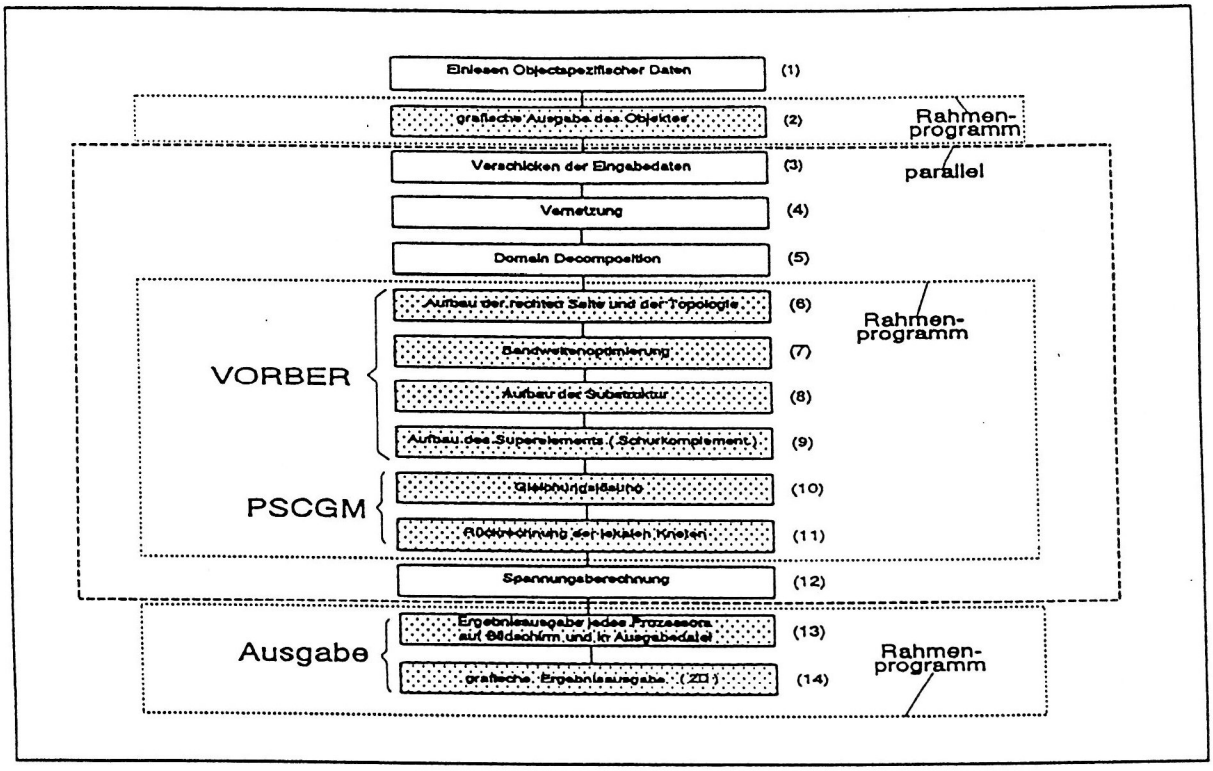

Bild 6. FEM-Rahmenprogramm

Die Routine VORBER ist verantwortlich für die Lastverteilung, die Bandweitenoptimierung der lokalen Steifigkeitsmatrix und für den Aufbau der Superelementmatrix. Entsprechend dem Bild 6 bearbeitet VORBER die Etappen 6 bis 9. PSCGM ist der eigentliche Gleichungslöser. In diesem Modul erfolgt die Berechnung des Schurkomplements , die Vorkonditionierung und die Iteration über die externen Knoten. PSCGM wird in Bild 6 in den Etappen 10 und 11 widergespiegelt. Die Etappe 13 schlägt sich in den Modulen AUSGABE und AUSDAT nieder. Die Subroutine AUSGABE gibt für jeden Knoten die Knotenverschiebungen auf den Bildschirm aus AUSDAT schreibt die Knotenverschiebungen geordnet nach Knotennummern in eine Datei namens PARFEM.OUT. Für die grafische Darstellung der Ergebnisse ist die Routine GRAFOUT, Etappe 13, verantwortlich. Der Nutzer des Rahmenprogramms muß für den Einbau eines neuen FE-Elements nur noch die Etappen 1, 3, 4, 5 und 12 bearbeiten. Aufgrund des modular aufgebauten Programms ist auch ein nachträglicher Austausch des Gleichungslösers oder Vorkonditionierers leicht möglich.

4 Ergebnisse

In Tabelle 2 sind die im PARSTAB erreichten Effizienzen gelistet. Die hier dargestellten Ergebnisse resultieren aus einer Erhöhung der Diskretisierung einzelner Stäbe. Somit änderte sich in jedem Testlauf die Kondition des Gleichungssystems, die Anzahl der externen Knoten jedoch nicht. Es erhöhte sich nur die Dimension der lokalen Teilsteifigkeitsmatrix $\mathbf{K}_{LL}$ und damit die arithmetische Rechenzeit aufgrund der Cholesky-Zerlegung und Schurkomplementbildung.

Prozessoranzahl Freiheitsgrad	16	32	64	128
4131	0,711	-	-	-
8451	0,952	-	-	-
14931	0,964	0,47	-	-
30051	0,98	0,53	0,27	0,09
43011	-	0,58	0,3	0,32
60291	-	0,97	0,91	0,73
86211	-	0,98	0,92	0,77
215811	-	-	-	0,90

Tabelle 2. Effizienzen mit PARSTAB

Bei konstanter Prozessorzahl zeigt sich ein Ansteigen der Effizienz mit größer werdendem Freiheitsgrad. Besonders stark sind die Effizienzunterschiede bei 128 Prozessoren. Sie fallen z. B. von 0,9 bei 215811 Freiheitsgraden auf 0,09 bei 30051 Freiheitsgraden ab. Demzufolge sind unterschiedlich große Aufgabenstellungen nur effektiv mit einer angepaßten Prozessorenzahl zu lösen. Tabelle 3 zeigt den Verlauf der Iterationsanzahl bei einer schrittweisen Änderung der Steifigkeit der Stäbe. Infolge einer Steifigkeitsänderung verbessert oder verschlechtert sich die Kondition der Systemmatrix. Die hier dargestellten Ergebnisse beziehen sich auf ein Balkentragwerk mit Stäben quadratischen Querschnitts aus Stahl. Die Größe I ist das Flächenträgheitsmoment der Stäbe und die Größe A der Querschnitt.

Beispiele von Kreis- bzw. Kreisringquerschnitten zeigen, daß I/A in der Größenordnung von 10 bis 200 liegt und somit für 16 Prozessoren Iterationsanzahlen von ca. 50 bis 150 auftreten werden. In Sonderfällen steigt die Iterationsanzahl noch wesentlich stärker an.

	Iterationsanzahl	
I/A (mm^2)	Prozessoranzahl: 16 Freiheitsgrad: 12717	Prozessoranzahl: 32 Freiheitsgrad: 78795
1000	393	1721
100	109	207
10	57	7
1	58	58
0,1	58	51
0,01	52	35
0,001	50	34

Tabelle 3. Iterationsanzahlverlauf

5 Zusammenfassung und Ausblick

Es wurde ein Programmkonzept zur Parallelisierung von FEM-Algorithmen in der Festkörpermechanik vorgestellt, das auf der für praxisrelevante Aufgaben vorrangig verwendeten Substrukturtechnik basiert. Die Ermittlung der lokalen Verformungsgrößen erfolgte dabei über die Berechnung des Schurkomplements. Das Gleichungssystem für die Koppelknoten wird mit einem PPCG-Verfahren iterativ gelöst. Wesentlicher Bestandteil des Programmkonzeptes ist ein Rahmenprogramm, das das Einarbeiten beliebiger unterschiedlicher Elementtypen gestattet. Eine erste Implementation wurde für Balkentragwerke (Programm PARSTAB) durchgeführt. Die damit erzielten Ergebnisse bestätigten im wesentlichen die Aussagen aus der Literatur hinsichtlich der Effizienz paralleler Algorithmen. Als interessant sind die Ergebnisse einzuschätzen, die sich für unterschiedlich konditionierte Systeme ergeben haben. Bei schlecht konditionierten Systemen (große Unterschiede in Flächenträgheitsmomenten und Querschnittsflächen des Balkens, wie sie für praktische Probleme typisch sind) ergeben sich relativ hohe Iterationsanzahlen. Diese Voruntersuchungen bilden eine solide Grundlage für die Erstellung eines Programmkonzeptes für die Parallelisierung großer, universeller FEM-Systeme.

Förderung

Die Autoren danken der Deutschen Forschungsgemeinschaft für die Förderung im Rahmen des Projektes "Entwicklung neuer Software-Konzepte zur Verbesserung der Ergebnisqualität und Effizienz von numerischen Modellen" (GA 480/1-1-).

Literatur

1. Boersma, A.: Lösung von linearen und nichtlinearen Problemen in der FEM. Workshop Parallelisierung TH Darmstadt, (1993).
2. Dongorra, J.; Sameh, A.: On some Parallel Banded System Solver. Technical Report ANL/MCS-TM-27, Argonne Laboratories, (1984).
3. Flynn, M. J.: Some Computer Organizations and Their Effectiveness. IEEE Transactions on Computers, C-21, (1972), 948-960.
4. Frommer, A.: Lösung linearer Gleichungssysteme auf Parallelrechnern. Vieweg, (1990).
5. George, J.: Solution of Sparce Systems of Equatons on Multiprocessor Architecures. In Turner, editor, Numerical Analysis and Parallel Processing, Springer, (1989), 32-94.
6. Haase, G.: Die nichtüberlappende Gebietszerlegungsmethode zur Parallelisierung und Vorkonditionierung iterativer Verfahren. Dissertation, TU Chemnitz, (1991).
7. Haase, G.; Langer, U.: The Non-overlapping Domain Decomposition Multiplicative Schwarz Method. Preprint 206, TU Chemnitz, (1991).
8. Haase, G.; Langer, U.; Meyer, A.: A New Approach to the Dirichlet Decomposition Method. In S. Hengst, editor, Proceedings of the "5th Multigrid Seminar" held at Eberswalde, DDR, May 14-18, (1990), Berlin, (1990), Akademie der Wissenschaften, 1-59.
9. Haase, G.; Langer, U.; Meyer, A.: The Approximate Dirichlet Decomposition Method. Computing, 74, (1991), 137-167.
10. Haase, G.; Langer, U.; Meyer, A.: Domain Decomposition Preconditioners with Inexact Subdomain Solver. Preprint 192, TU Chemnitz, (1991). Also in: Journal of Numerical Linear Algebra with Applications, 1, 1,)1992), 27-42.
11. Law, K. H.: A Parallel Finite Elements Solution Method. Computer and Structures, 23, 6, (1986), 845-858.
12. Meißner, U.; Möller, B.; Lämmer, L.: Parallelisierung von Finite-Element-Algorithmen für Transputer-Arbeitsplatz-Rechner. 3. COSAR Konferenz, (1992).
13. Meyer, A.: A Parallel Preconditioned Conjugate Gradient Method using Domain Decomposition and Inexact Solvers on each Subdomain. Computing, 45, (1990), 217-234.
14. Ratke, R.: Vollständig paralleles Programmkonzept für unstrukturierte Gitter auf MIMD-Systemen. Workshop Parallelisierung ,TH Darmstadt, (1993).
15. Ristau, P.; Schendel, U.: Parallele LU-Zerlegung und Implementierung auf dem SUPREMUM-Simulator. Preprint A-91-08, Freie Universität Berlin, (1991).
16. Schreiber A.: Block Algorithmus for Parallel Machines. Volume 13 of IMA Volumes in Mathematics and its Applications, (1966).
17. Weiss, R.: Parallelisation and Preconditioning of Iterative Linear Solver, either - or? Workshop über wissenschaftliches Rechnen, TU Braunschweig (1993).
18. Wriggers, P.: Implementation eines FE-Programmes auf Parallelrechner. Workshop Parallelisierung , TH Darmstadt, (1993).

Anschrift: Dipl.-Ing. Günther Blanke und Professor Dr.-Ing. habil. Walter Weese, Institut für Mechanik, Otto-von-Guericke-Universität, Universitätsplatz 2, 39106 Magdeburg.