TECHNISCHE MECHANIK 8 (1987) Heft 3
Manuskripteingang: 20. 02. 1987

Zum Einsatz von Vektorprozessoren im Rahmen der Finite-Elemente-Methode

Erste Erfahrungen mit dem Matrixmodul

J. Linek

1. Einleitung

Im Bereich der Brennstabmodelltheorie, einem Spezialgebiet der Kerntechnik, interessieren die Spannungs- und Dehnungsverteilung im Hüllrohr eines Brennstabes im Falle des Kontaktes zwischen dem Hüllrohr und der Brennstofftablette. Für die realistische Modellierung dieses Falles sind plastische Deformationen des Hüllrohres zu berücksichtigen. Bei der Lösung dieses Kontaktproblems [18] mit Hilfe der Finit-Element-Methode (FEM) traten unvertretbar hohe Rechenzeiten auf.
Auch für andere Anwendungsfälle der FEM können zu große Rechenzeiten auftreten, z. B. bei Problemen mit vielen Freiheitsgraden, bei der Berücksichtigung großer Deformationen, beim Auftreten nichtlinearen Materialverhaltens usw.
In diesen Fällen muß schnellere Rechentechnik, wenn vorhanden, genutzt oder zu schnelleren Algorithmen übergegangen werden. Schnellere Rechentechnik steht mit den neuen Vektorprozessoren und anderen Parallelprozessoren zur Verfügung.
Erste Erfahrungen für den Einsatz eines Vektorprozessors im Rahmen der FEM konnten mit dem Matrixmodul, einem Vektorprozessor der EDVA ES–1055, gesammelt werden.

2. Spezifik der Anwendung von Vektorprozessoren

Mit den Vektorprozessoren stehen neuartige Instrumente zur Effektivitätssteigerung von Rechenprogrammen zur Verfügung [1] bis [4]. Dabei beruht allgemein die Effektivitätssteigerung von Vektorprozessoren auf ihrer speziellen Hardware-Struktur, die es gestattet, in einem Durchlauf ein oder mehrere Eingabedatenströme arithmetisch zu einem Ausgabedatenstrom zu verknüpfen. Damit eignen sich diese Prozessoren besonders für die Ausführung von Vektor- und Matrizenoperationen. Die Verarbeitungsgeschwindigkeit, gemessen in Mflop (10^6 floating point operations per second), hängt direkt von der zu verarbeitenden Vektorlänge ab und steigt mit zunehmender Vektorlänge bis zu einem Grenzwert (vgl. Bild 1).
Da der Vektorprozessor zu Beginn einer Operation eine gewisse Zeit für die Parametervermittlung und zur Initialisierung benötigt, können sich für kleine Vektorlängen geringere Verarbeitungsgeschwindigkeiten als bei sequentieller Abarbeitung ergeben. Aus diesem Grunde wird die spezielle Aufbereitung eines existierenden Algorithmus, die Vektorisierung, nicht eine formale Umprogrammierung sein.

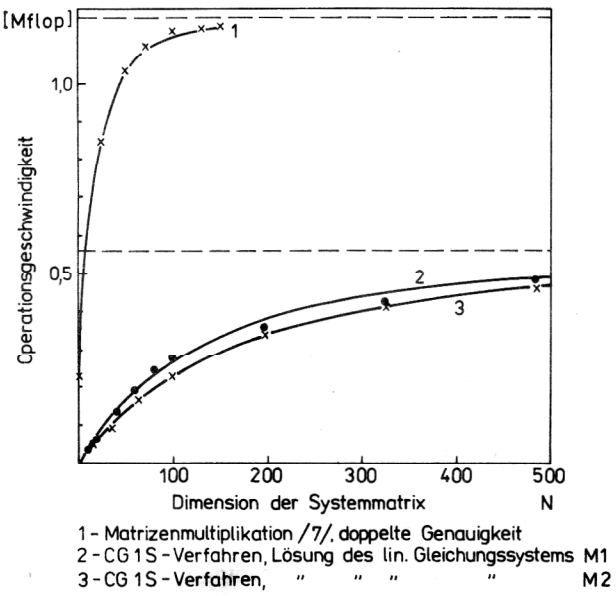

Bild 1
Operationsgeschwindigkeit des Vektorprozessors Matrixmodul für ausgewählte Beispiele, doppelt genaue Zahlendarstellung

Die Anwendung der Vektorprozessoren erfolgt in Rechenprogrammen über eine Spracherweiterung der problemorientierten Programmiersprache, z. B. FORTRAN, wie in [3] beschrieben ist, oder durch den Aufruf von Unterprogrammen. In einigen Fällen ist die Nutzung des Vektorprozessors nur vom ASSEMBLER-Niveau aus möglich [5], d. h. es müssen ggf. eigene ASSEMBLER-Unterprogramme erstellt werden.

3. Der Vektorprozessor Matrixmodul

Mit dem Matrixmodul (MAMO) steht an den Robotron-Rechnern ES–1055 und ES–1055M ein vielseitig anwendbarer Vektorprozessor zur Verfügung. Eine vollständige Übersicht über Funktion und Anwendung des MAMO wird in [5] bis [7] gegeben.
Ursprünglich erfolgte die MAMO-Anwendung mittels ASSEMBLER, wofür Befehlsbeschreibung [8] und Benutzungshinweise [9] existieren. Im ZfK Rossendorf wurde eine Bibliothek mit Unterprogrammen erstellt [10] bis [12], die eine universelle Nutzung des MAMO von problemorientierten Sprachen FORTRAN und PL/1 aus ermöglichen.

Für eine Nutzung des MAMO ist der zu erwartende Rechenzeitgewinn von großem Interesse. Leider läßt sich durch die Vielseitigkeit des MAMO eine allgemeingültige Aussage diesbezüglich nicht treffen. In [7] wird für die Verarbeitung von Vektoren eine 10 bis 20fache Erhöhung der Verarbeitungsgeschwindigkeit angegeben. An Hand der Matrizenmultiplikation wird in [6] die Verarbeitungsgeschwindigkeit in Abhängigkeit von der Matrixdimension angegeben (Bild 1), wobei die maximale Verarbeitungsgeschwindigkeit bei einfacher Genauigkeit 5.1 Mflop und bei doppelter Genauigkeit 1.2 Mflop beträgt. Außerdem zeigt Bild 1, daß erst bei größeren Dimensionen, etwa ab N = 50, die Möglichkeiten des MAMO effektiv genutzt werden.
Somit ist der Einsatz des MAMO und anderer Vektorprozessoren nur dort sinnvoll, wo große Vektorlängen bzw. Matrixdimensionen auftreten. Dies ist besonders bei der Vektorisierung eines Algorithmus zu beachten, da hierbei durch geeignete Speicherung und Verarbeitung der Daten große Vektorlängen erzeugt werden können und müssen.

4. Zur Vektorisierung des FEM-Algorithmus

Der gesamte FEM-Algorithmus basiert im wesentlichen auf Vektor- und Matrizenoperationen und bietet damit eine günstige Voraussetzung zur Vektorisierung. Für den MAMO-Einsatz eignen sich allerdings nur die Operationen mit großen Vektorlängen.
Die größten Vektorlängen treten im FEM-Algorithmus in Verbindung mit der Systemsteifigkeitsbeziehung

$$K_s * \vec{v}_s = \vec{f}_s \qquad (1)$$

K_s Systemsteifigkeitsmatrix, $\vec{v}_s$ Systemverschiebungsvektor, $\vec{f}_s$ Systemlastvektor

auf. Die Operationen auf der Elementebene (Elementsteifigkeitsmatrix, Elementlastvektor usw.) bieten wesentlich geringere Vektorlängen. Für den Fall einfacher Verschiebungsansätze läßt sich eindeutig feststellen, daß die hier auftretenden Vektorlängen für einen effektiven MAMO-Einsatz zu kurz sind. Somit konzentriert sich die MAMO-Anwendung auf Operationen mit der Systemsteifigkeitsbeziehung (1), und damit hauptsächlich auf die Lösung dieses linearen Gleichungssystems. Berücksichtigt man weiterhin, daß bei nichtlinearen Problemen das Gleichungssystem (1) im Verlaufe einer Iteration, z. B. eines Newton-Raphson-Verfahrens, mehrfach gelöst werden muß, kommt einem effektiven Lösungsverfahren für (1) unter Nutzung eines Vektorprozessors besondere Bedeutung zu.
Auch bei der Auswahl eines geeigneten Lösungsverfahrens für (1) muß das Auswahlkriterium „große Vektorlänge" berücksichtigt werden.
Allgemein läßt sich für Verfahren, die auf einer Dreieckszerlegung von K_s basieren (z. B. Cholesky-Verfahren) feststellen, daß sowohl bei der Triangulation als auch beim Vorwärts- und Rückwärtseinsetzen die Vektorlängen am Beginn und Ende des Algorithmus zu kurz werden. Außerdem sind diese Verfahren hochgradig rekursiv und eignen sich daher nicht zur Vektorisierung [14].

Iterative Lösungsverfahren bieten für den Einsatz von Vektorprozessoren den entscheidenden Vorteil gegenüber der obengenannten Klasse von direkten Lösungsverfahren, daß sie i. allgm. gut vektorisierbar sind. Natürlich sind auch die anderen Vorteile der iterativen Verfahren gegenüber den direkten Lösungsverfahren, wie sie z. B. in [15] genannt sind, bei der MAMO-Anwendung weitestgehend zu nutzen:

- Die Matrix K_s wird durch das Lösungsverfahren nicht verändert, was die einfache Anwendung der Methode der variablen Steifigkeit für nichtlineare Probleme ermöglicht.
- Das Gleichungssystem (1) kann mit beliebig vorgegebener Genauigkeit gelöst werden.
- Die Speicherung der Matrix K_s ist in verdichteter Form möglich, womit die Probleme der Bandbreitenoptimierung entfallen.

Bilfinger und Schmidt [16] haben verschiedene iterative Verfahren untersucht und das Konjugierte Gradientenverfahren (CG-Verfahren) als für die FEM am besten geeignet eingeschätzt. Weitere Untersuchungen wurden an Hand der CG-Verfahren durchgeführt.

5. Die Vektorisierung des CG-Verfahrens

Die Vektorisierung des CG-Verfahrens erfolgte nach dem Algorithmus von Reid [17]. Das zu lösende lineare Gleichungssystem sei

$$A * \vec{x} = \vec{b} \qquad (2)$$

mit der Matrix A (N, N), symmetrisch und positiv definit. Das CG-Verfahren läuft nach dem folgenden Algorithmus ab:

$$\vec{p}_0 = \vec{r}_0 = \vec{b} - A * \vec{x}_0 \qquad \text{Startrechnung}$$

$$a_i = (\vec{r}_i^{\,T} * \vec{r}_i)/(\vec{p}_i^{\,T} * (A * \vec{p}_i))$$

$$\vec{x}_{i+1} = \vec{x}_i + a_i * \vec{p}_i$$

$$\vec{r}_{i+1} = \vec{r}_i - a_i * (A * \vec{p}_i) \qquad (3)$$

$$b_i = (\vec{r}_{i+1}^{\,T} * \vec{r}_{i+1})/(\vec{r}_i^{\,T} * \vec{r}_i)$$

$$\vec{p}_{i+1} = \vec{r}_i + b_i * \vec{p}_i$$

für i = 0, 1, . . . , N; $\vec{x}_0$ – Anfangsnäherung

Der wesentliche Aufwand pro Iterationsschritt besteht in den Operationen:

- Bandmatrix * Vektor $A * \vec{p}_i$
- 2 Skalarprodukten $\vec{r}_i^{\,T} * \vec{r}_i$ und $\vec{p}_i^{\,T} * (A * \vec{p}_i)$
- 3 Vektoroperationen des Typs $\vec{x}_{i+1} = \vec{x}_i + a_i * \vec{p}_i$
- 4 skalaren Divisionen

Mit Hilfe der Unterprogramme der MAMO-Bibliothek [10] bis [12] sind die Operationen des 2. und 3. Typs direkt vektorisierbar durch den Aufruf der Unterprogramme

CALL VXV (RI, RI, N, 'stk')

CALL SXFAF (AI, PI, XI, XI, N, 'stk')

wobei RI, AI, PI, XI den Größen $\vec{r}_i$, a_i, $\vec{p}_i$, $\vec{x}_i$ aus (3) und N der Vektorlänge entsprechen. 'stk' stellt eine Zeichenkette zur Steuerung des MAMO dar [10] bis [12].
Die Multiplikation Bandmatrix * Vektor läßt sich nicht direkt mit den Unterprogrammen der MAMO-Bibliothek vektorisieren. Hierfür wurde ein eigener Algorithmus entwickelt, der an Stelle des üblichen Algorithmus für die Multiplikation Matrix * Vektor die diagonalvektororientierte Multiplikation nutzt [16]. Für die diagonalvektororientierte Multiplikation muß die Matrix A so abgespeichert werden, daß man zu den Haupt- und Nebendiagonalen leichten Zugriff hat, z. B. in der folgenden Form

$$A(N,N) = \begin{bmatrix} \vec{d}_o & \vec{d}_1 & \vec{d}_2 & & & & \vec{d}_{NB} \\ * & * & * & . & . & . & * \\ & * & * & * & . & . & . & * \\ & & * & * & * & . & . & . & * \\ & & & * & * & * & . & . & . \\ & & & & * & * & * & . & . \\ & & & & & * & * & * & . \\ & & & & & & * & * & * \\ & & & & & & & * & * \\ & & & & & & & & * \end{bmatrix}$$

$$A(N,N) = \begin{bmatrix} \vec{d}_o & \vec{d}_1 & \vec{d}_2 & & & \vec{d}_{NB} \\ * & * & * & . & . & * \\ * & * & * & . & . & * \\ * & * & * & . & . & * \\ * & * & * & . & . & 0 \\ * & * & * & . & 0 & 0 \\ * & * & * & 0 & 0 & 0 \\ * & * & 0 & 0 & 0 & 0 \\ * & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\vec{d}_o$ – Hauptdiagonale, $\vec{d}_1 \ldots \vec{d}_{NB}$ – Nebendiagonalen

Die Multiplikation $A * \vec{p} = \vec{q}$ läßt sich dann in einer DO-Schleife mit den Vektoroperationen

$$\vec{q}(1:N) = \vec{d}_o(1:N) * \vec{p}(1:N)$$

DO I=1 TO NB;

$$\vec{q}(1{:}N-1) = \vec{q}(1{:}N-I) + \vec{d}_i(1{:}N-I) * \vec{p}(I+1{:}N) \quad (4)$$

$$\vec{q}(I+1{:}N) = \vec{q}(I+1{:}N) + \vec{d}_i(1{:}N-I) * \vec{p}(1{:}N-I)$$

END;

darstellen. Dabei soll der Ausdruck $\vec{d}_i(1:N-I)$ bedeuten, daß der Vektor $\vec{d}_i$ (Nebendiagonale i) mit den Komponenten 1 bis N–I an der Operation beteiligt ist. Die Operation „*" stellt hier die Multiplikation der Vektorelemente dar und darf nicht mit dem Skalarprodukt verwechselt werden.
Die in (4) auftretenden Operationen des Typs

Vektor = Vektor * Vektor und

Vektor = Vektor + (Vektor * Vektor)

sind mit den Unterprogrammen FXF und FXFAF aus [10] bis [12] direkt vektorisierbar.

Mit der Anwendung des Algorithmus (4) für den Vektorprozessor erreicht man gegenüber dem üblichen Algorithmus für die Multiplikation Matrix * Vektor die folgenden Vorteile:

- Die Vektorlänge erhöht sich von 2*NB+1 auf den Bereich N . . . (N–NB).
- Die Anzahl der Vektoroperationen verringert sich von N auf 2*NB–1.
- Diagonalvektoren $\vec{d}_i$, die nur Nullen enthalten, müssen nicht abgespeichert werden oder können zumindest bei der Verarbeitung übersprungen werden. Der Test

  $\vec{d}_i = \vec{o}$

  kann mit dem MAMO realisiert werden.

Für Testzwecke wurde nach der Methode der Konjugierten Gradienten ein Einschrittverfahren entsprechend den Algorithmen (3) und (4) mit Nutzung und ohne Nutzung des Vektorprozessors programmiert. Darüber hinaus wurden nach [16] weitere CG-Verfahren und das Cholesky-Verfahren zur Lösung linearer Gleichungssysteme programmiert. (vgl. Tab. 1). Für die MAMO-Anwendung wurden die Unterprogramme aus [10] bis [12] genutzt, wobei die Rahmenprogramme in PL/1 erstellt wurden.

Tabelle 1
Verfahren zur Lösung eines linearen Gleichungssystems mit und ohne Nutzung des Vektorprozessors MAMO, die zum Rechenzeitvergleich herangezogen wurden

Unterprogrammname	mathematisches Lösungsverfahren
CG1S	CG-Einschritt-Verfahren nach [16] mit Anwendung des Vektorprozessors MAMO
CG2S	CG-Zweischritt-Verfahren nach [16] mit Anwendung des Vektorprozessors MAMO
CG1SSK	CG-Einschritt-Verfahren mit Skalierung nach [16] mit Anwendung des Vektorprozessors MAMO
CG1SN	CG-Einschritt-Verfahren nach [16] ohne Anwendung des Vektorprozessors MAMO
CHOLBD	Triangulation einer Bandmatrix nach Cholesky
VORUED	Vorwärts- und Rückwärts-Einsetzen

Alle Programme sind in PL/1 programmiert und arbeiten mit doppelt genauer Zahlendarstellung.

6. Testbeispiele

An Hand von drei für die FEM typischen linearen Gleichungssystemen wurden für die obengenannten Lösungsverfahren Vergleichsrechnungen mit Laufzeitmessungen am ES–1055 durchgeführt. Dabei wurde das Betriebssystem OS/ES 6.1M9–SVS im Multiprogrammbetrieb genutzt. Alle Rechnungen wurden mit doppelter Genauigkeit ausgeführt. Für die CG-Verfahren wurde einheitlich als Abbruchkriterium

$$\frac{\|\vec{r}_i\|}{\|\vec{b}\|} \leqslant \epsilon = 10^{-10}$$

genutzt, mit $\|\vec{r}_i\|$ – Norm des Residuenvektors und $\|\vec{b}\|$ – Norm der rechten Seite.

6.1. *Das Balkenproblem*

Die Systemgleichung eines Balkenproblems (Matrix M1 aus [16])

$$A * \vec{x} = \vec{b}$$

wurde für verschiedene Matrixdimensionen (N = 10, . . . , 100) gelöst, mit

$$A(N,N) = \begin{bmatrix} 5 & -4 & 1 & & & & \\ -4 & 6 & -4 & 1 & & & \\ 1 & -4 & 6 & -4 & 1 & & \\ & * & * & * & * & * & \\ & & 1 & -4 & 6 & -4 & 1 \\ & & & 1 & -4 & 6 & -4 \\ & & & & 1 & -4 & 5 \end{bmatrix} \qquad \vec{b}(N) = \begin{bmatrix} 1 \\ 0 \\ * \\ \vdots \\ * \\ * \\ 0 \end{bmatrix} \tag{5}$$

Die Matrix M1 aus (5) besitzt die folgenden Eigenschaften:

1. Sie ist schlecht konditioniert. Für N = 100 liegt die Konditionszahl (max. Eigenwert/min. Eigenwert) bei 10^7.
2. Sie hat eine schmale Bandbreite NB = 2.
3. Innerhalb des Bandes treten keine Nebendiagonalen auf, die nur Nullen enthalten.

In Bild 2 sind die zur Lösung von (5) benötigten Rechenzeiten in Abhängigkeit von der Vektorlänge N dargestellt. Für den Vergleich der CG-Verfahren mit und ohne Nutzung des Vektorprozessors wurde in Bild 3 die Rechenzeit pro Iterationsschritt für CG 1S und CG 1SN angegeben. Außerdem ist die Steigerung der Verarbeitungsgeschwindigkeit SP

$$SP = t_{seriell} / t_{parallel}$$

angegeben.

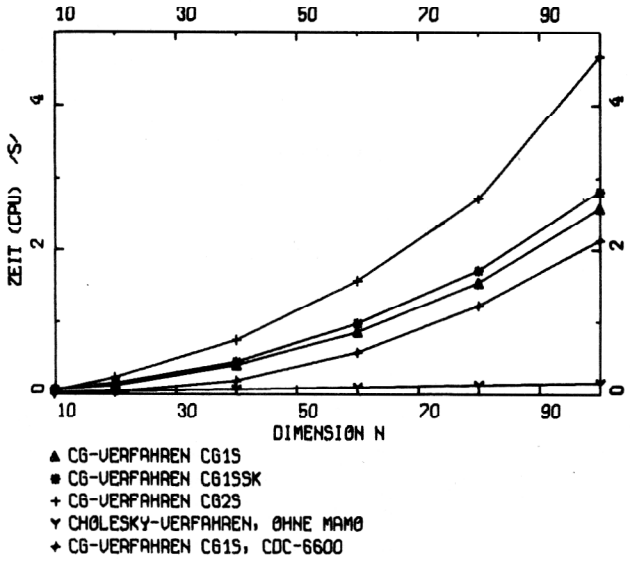

Bild 2
Zeit (CPU-Zeit), die zur Lösung des linearen Gleichungssystems mit der Matrix M1 für unterschiedliche Lösungsverfahren benötigt wurde, in Abhängigkeit von der Matrixdimension

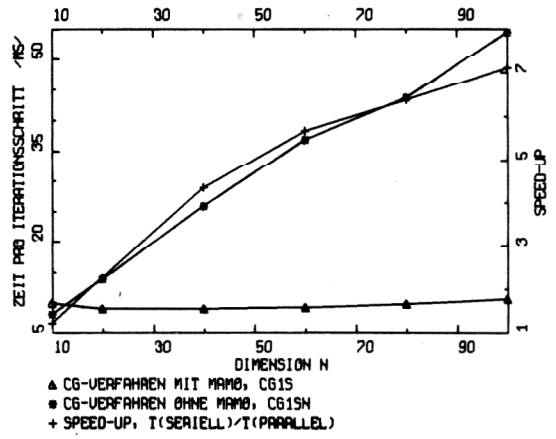

Bild 3
Vergleich der Rechenzeit pro Iterationsschritt (Matrix M1) für CG-Verfahren mit und ohne Nutzung des Vektorprozessors Matrixmodul

6.2. *Das Reaktordiffusionsproblem*

Als Beispiel für ein zweidimensionales Problem wurde das Gleichungssystem

$$A * \vec{x} = \vec{b}$$

mit der Testmatrix M2 aus [16] für unterschiedliche Dimensionen N = 4, 16, . . . , 484 gelöst. Die Matrix A (N, N) hat die folgende Gestalt:

$$A(N,N) = \left[\begin{array}{c|c|c|c|c} D & -I & 0 & 0 & 0 \\ \hline -I & D & -I & 0 & 0 \\ \hline * & * & * & * & * \\ \hline 0 & 0 & -I & D & -I \\ \hline 0 & 0 & 0 & -I & D \end{array}\right]$$

$$D(IM,IM) = \begin{bmatrix} 4 & -1 & & & & & \\ -1 & 4 & -1 & & & & \\ & -1 & 4 & -1 & & & \\ & & * & * & * & & \\ & & & -1 & 4 & -1 & \\ & & & & -1 & 4 & \end{bmatrix} \tag{6}$$

$$-I(IM,IM) = \begin{bmatrix} -1 & & & & & & \\ & -1 & & & & & \\ & & * & & & & \\ & & & -1 & & & \\ & & & & -1 & & \\ & & & & & -1 & \end{bmatrix}$$

Die rechte Seite b ergibt sich aus der Rekursionsformel

$$b = 2 * \pi^2 * l^2 * \sin(i * \pi * l) * \sin(j * \pi * l)$$

$$i, j = 1, \ldots, IM-1 .$$

Das Gleichungssystem entsteht bei der numerischen Lösung eines Reaktordiffusionsproblems auf dem Einheitsquadrat. Es umfaßt die inneren Punkte des Phasengebietes bei dessen Einteilung in IM gleiche Maschen der Länge l = 1/M. Die Zahl der Unbekannten beträgt $N = (IM-1)^2$. Die Matrix M2 besitzt Bandstruktur mit

einer Bandbreite von NB = IM. Die Testmatrix M2 ist charakterisiert durch folgende Eigenschaften:

1. Die Matrix M2 ist gut konditioniert. Die Konditionszahl ist proportional zu N.
2. Mit feiner werdender Diskretisierung wächst die Bandbreite.
3. Innerhalb des Bandes enthält M2 Nebendiagonalen, die nur aus Nullen bestehen.

In den Bildern 4 und 5 sind die Rechenzeiten, die Rechenzeiten pro Iterationsschritt und der Rechenzeitgewinn durch die MAMO-Anwendung (Speed-up) dargestellt.

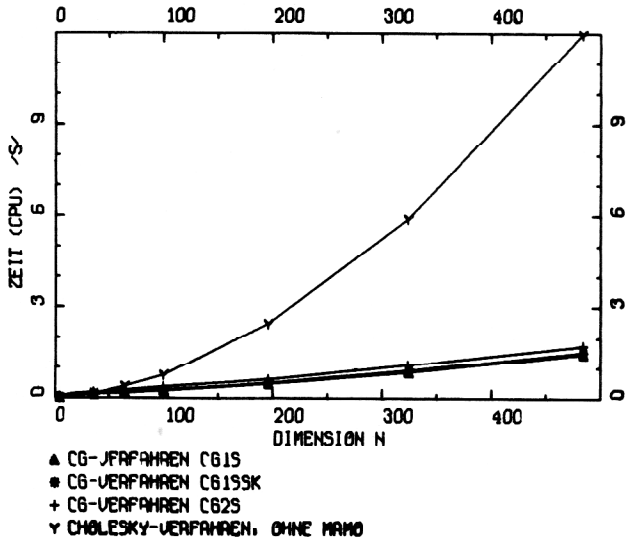

Bild 4
Zeit (CPU-Zeit), die zur Lösung des linearen Gleichungssystems mit der Matrix M2 für unterschiedliche Lösungsverfahren benötigt wurde, in Abhängigkeit von der Matrixdimension

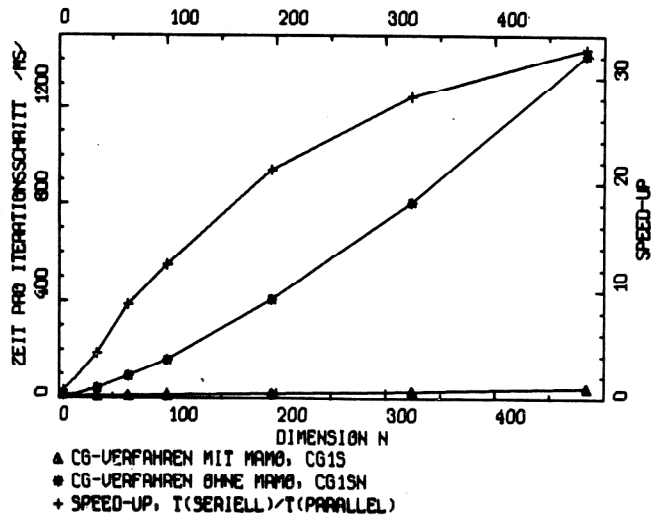

Bild 5
Vergleich der Rechenzeit pro Iterationsschritt (Matrix M2) für CG-Verfahren mit und ohne Nutzung des Vektorprozessors Matrixmodul

6.3. Das Kontaktproblem

Zur Testung der Algorithmen an Hand eines typischen Problems aus der Brennstabmodelltheorie wurde aus der Kontaktaufgabe [18] die Systemsteifigkeitsbeziehung genutzt. In der Kontaktaufgabe [18] wird die Spannungs-Dehnungs-Verteilung in einem Brennstab im Falle des Kontakts zwischen Brennstofftablette und Hüllrohr unter der Voraussetzung der Rotationssymmetrie berechnet (2-dimensionale Berechnung).
Die Systemsteifigkeitsmatrix M3 besitzt N = 338 Freiheitsgrade und eine Bandbreite von NB = 30. Innerhalb des Bandes enthalten die Nebendiagonalen $\vec{d}_i$ ($i = 5, \ldots, 25$) nur Nullen. Die Kondition der Matrix ist schlecht, die Konditionszahl liegt bei 10^7. Die für Matrix M3 gemessenen Rechenzeiten sind für alle Verfahren in Tabelle 2 zusammengestellt.

Tabelle 2
Testmatrix M3 (Systemsteifigkeitsmatrix des Kontaktproblems) Rechenzeiten, die bei der Lösung des linearen Gleichungssystems mit der Matrix M3 ermittelt wurden

Lösungsverfahren	CPU-Zeit [s]	CPU-Zeit pro Iterationsschritt [ms]	Speed-up t_{ser}/t_{par}
CG1S	21.5	59.0	} 22.4
CG1SN	480.6	1313.0	
CG2S	26.6	73.3	
CG1SSK	16.5	60.9	} 0.8
CHOLESKY	13.4	–	
dav. CHOLBD	11.9	–	
dav. VORUED	1.5	–	

7. Diskussion

Operationsgeschwindigkeit

Die unter Nutzung des MAMO erzielten Operationsgeschwindigkeiten sind für eine Matrizenmultiplikation in [7] angegeben (Bild 1). Dabei wird für doppelt genaue Verarbeitung eine maximale Operationsgeschwindigkeit von ca. 1.20 Mflop erreicht. Bei der Lösung der Testbeispiele 1 und 2 (Matrizen M1 und M2) wurde die Operationsgeschwindigkeit aus der Anzahl der wesentlichen Gleitkommaoperationen des Algorithmus (3) und den gemessenen Zeiten geschätzt. Für beide Fälle ergibt sich eine maximale Operationsgeschwindigkeit von 0.56 Mflop (Bild 1), was ca. 50 % des obengenannten Wertes entspricht. Dieser Unterschied in den erzielten Verarbeitungsgeschwindigkeiten kann folgende Ursachen haben:

1. Für die eigenen Rechnungen wurde der Vektorprozessor MAMO über Unterprogramme aus [10] aufgerufen, im Vergleichsfalle erfolgte der Aufruf des MAMO vom ASSEMBLER. Die Unterprogramme mit der notwendigen Parametervermittlung und das PL/1-Hauptprogramm ergeben Rechenzeitverluste.
2. Die Laufzeitmessungen erfolgten im Multiprogrammbetrieb, so daß die gemessenen Zeiten unsicher und wahrscheinlich zu groß sind.
3. Bei der Matrizenmultiplikation wird ein sehr schneller MAMO-Befehl benutzt, wohingegen im Programm CG1S mehrere MAMO-Befehle verwendet werden, die in ihrer Effektivität unterschiedlich sind.

Aus Bild 1 läßt sich außerdem ablesen, ab welcher Problemgröße (Vektorlänge) N die Möglichkeiten des MAMO effektiv genutzt werden können. Während sich für die Matrizenmultiplikation schon ab $N > 40$ etwa 80 % der maximalen Verarbeitungsgeschwindigkeit erreichen lassen, trifft dies für das CG1S-Verfahren erst ab einer Vektorlänge von $N > 400$ zu.

Rechenzeitvergleich der CG-Verfahren mit und ohne MAMO

Beim Vergleich der Rechenzeiten pro Iterationsschritt für das CG-Einschritt-Verfahren mit und ohne MAMO-Nutzung (CG1S, CG1SN) treten bei allen drei Testbeispielen deutliche Geschwindigkeitssteigerungen auf (Bild 3, Bild 5, Tab. 2). Der Speed-up nimmt mit wachsender Vektorlänge zu bis 7.11 bei M1, 32.74 bei M2 und 22.4 bei M3. Mit Hilfe des Vektorprozessors erreicht man am ES–1055 in etwa die Verarbeitungsgeschwindigkeit, die für dieses Problem M1 an der CDC–6600 benötigt wurde [16] (Bild 2). Außerdem kann man an Bild 3 und Bild 5 deutlich das unterschiedliche Laufzeitverhalten bei paralleler und serieller Verarbeitung erkennen. Im Falle des eindimensionalen Testbeispiels (Balkenproblem) ist die Vektorlänge N noch so klein, daß die Rechenzeit pro Iterationsschritt bei Nutzung des Vektorprozessors nahezu konstant ist, während sie bei serieller Verarbeitung linear mit N wächst. Beim zweidimensionalen Testbeispiel (Reaktordiffusionsproblem) wächst mit der Dimension N auch die Bandbreite der Matrix M2, so daß die Rechenzeit pro Iterationsschritt bei MAMO-Nutzung linear mit N steigt, während sie bei serieller Verarbeitung progressiv anwächst. So ist auch zu erklären, daß der Rechenzeitgewinn bei Beispiel 2 größer als bei Beispiel 1 ist. Es zeigt sich beim Testbeispiel 2 auch, daß für die kurze Vektorlänge $N = 4$ die serielle Verarbeitung schneller ist als die Verarbeitung mit dem MAMO.

Vergleich der Rechenzeiten für die CG-Verfahren mit MAMO-Nutzung

Bei allen drei Testbeispielen zeigt sich, daß das Zweischritt-Verfahren CG2S die meiste Rechenzeit benötigt. Die Anzahl der Iterationsschritte ist zwar in jedem Fall für CG2S am kleinsten, aber der Aufwand pro Iterationsschritt ist bei diesem Verfahren so groß, daß die Gesamtrechenzeit am ungünstigsten von allen drei Verfahren liegt.

Für das Einschritt-Verfahren mit Skalierung CG1SSK und ohne Skalierung CG1S sind die Unterschiede in der Rechenzeit für die beiden ersten Testfälle gering, da der Aufwand pro Iterationsschritt in beiden Verfahren nahezu gleich ist, und für Matrix M2 die gleiche sowie für M1 fast die gleiche Anzahl Iterationsschritte benötigt werden. Bei Beispiel 3, wo erstmals größere betragsmäßige Unterschiede in den Hauptdiagonal-Elementen auftreten, wirkt sich die Skalierung positiv aus, so daß Beispiel 3 mit CG1SSK auf Grund der deutlich niedrigeren Anzahl von Iterationsschritten etwa 1.3mal schneller gelöst wird als mit CG1S.

Vergleich der Rechenzeiten CG-Verfahren – Cholesky-Verfahren

Bei diesem Vergleich kann man sich auf die CG-Verfahren mit Nutzung des Vektorprozessors beschränken.

Es zeigt sich erwartungsgemäß, daß beim eindimensionalen Problem (Balkenproblem) das Cholesky-Verfahren allen CG-Verfahren deutlich überlegen ist. Dies läßt sich auf die Eigenschaften der Matrix M1 zurückführen:

- Schmale Bandbreite ohne Nulldiagonalen, damit tritt für das Cholesky-Verfahren kein fill-in auf.
- Schlechte Kondition.

Im Fall des zweidimensionalen Problems (Reaktordiffusionsproblem) sind die CG-Verfahren deutlich schneller als das Cholesky-Verfahren, was sich ebenfalls aus den Eigenschaften der Matrix M2 erklären läßt:

- Größere Bandbreite mit Nulldiagonalen, so daß für das Cholesky-Verfahren fill-in auftritt.
- Gute Kondition.

Bei diesem Beispiel ist noch zu bemerken, daß hier die Rechenzeiten für das Vorwärts- und Rückwärts-Einsetzen in der Größenordnung für CG1SSK liegen. Im Testfall 3 (Kontaktproblem) liegen die Rechenzeiten für das Cholesky-Verfahren ca. 20 % unter denen des schnellsten CG-Verfahren CG1SSK, wobei beim Cholesky-Verfahren die Zeiten für das Vorwärts- und Rückwärts-Einsetzen und für die Dreieckszerlegung in Tabelle 2 angegeben sind. Vor diesem Hintergrund zeigt Beispiel 3, daß die Lösung der Steifigkeitsbeziehung des Kontaktproblems mit einem CG-Verfahren unter Nutzung des Vektorprozessors MAMO allein noch keinen Rechenzeitgewinn bringt. Erst wenn es gelingt, die weiteren Vorteile der CG-Verfahren im FEM-Algorithmus zu nutzen, kann gegenüber der seriellen Verarbeitung mit dem Cholesky-Verfahren ein Rechenzeitgewinn erwartet werden.

Zur Überprüfung dieser Möglichkeit wurde die vollständige Kontaktaufgabe für elastisches und elastisch-plastisches Materialverhalten des Hüllrohres gelöst. Dazu wurde ein kleines FEM-Programm eingesetzt, das die Kontaktbedingungen an den Kontaktpunkten direkt in die Steifigkeitsmatrix einbezieht. Damit wird in einer Iteration über alle möglichen Kontaktpunkte ständig die Steifigkeitsmatrix geändert. Dieses Verfahren soll hier als Methode der variablen Steifigkeit bezeichnet werden. Es nutzt den Vorteil der CG-Verfahren, die Steifigkeitsmatrix beim Lösungsalgorithmus nicht zu verändern. Beim Cholesky-Verfahren hingegen muß in jedem Iterationsschritt vor der Triangulation der Zustand der Matrix gerettet werden (auf ein externes Speichermedium) und bei Fortsetzung der Iteration der alte Zustand der Steifigkeitsmatrix wieder hergestellt werden.

Für die Berechnung des elastisch-plastischen Materialverhaltens wurde ein Verfahren von Koczyk aus [13] genutzt, in dem anschließend an die erfolgte Kontaktiteration eine Iteration zur Bestimmung der plastischen Deformation mit konstanter Steifigkeitsmatrix durchgeführt wird. Man sieht an Hand der Ergebnisse (Tab. 3), daß dieses Verfahren das Cholesky-Verfahren bevorteilt, da bei konstanter Steifigkeitsmatrix nach einmaliger Dreieckszerlegung der Matrix in jedem Iterationsschritt nur das Vorwärts- und Rückwärts-Einsetzen

durchgeführt werden muß. Trotzdem erweist sich sowohl bei der „elastischen" als auch bei der „elastisch-plastischen Kontaktaufgabe" das CG-Verfahren mit Skalierung CG1SSK als das günstigste Lösungsverfahren.

Tabelle 3
Rechenzeitvergleich für die vollständige Lösung des Kontaktproblems [18] nach der Methode der variablen Steifigkeit für die Kontaktiteration. Die angegebenen Zeiten sind CPU-Zeiten (in [s]) für den gesamten GO-Step.

Verfahren	elastisches Hüllrohrmaterial-Verhalten	elastisch-plastisches Hüllrohrmaterial-Verhalten
FEM mit CHOLESKY	947.8	1012.3
FEM mit CG1SSK	75.0	227.7
Speed-up $^tCHOL/^tCG$	12.6	4.4

8. Zusammenfassung

Mit dem Matrixmodul steht am ES–1055 ein leistungsfähiger Vektorprozessor zur Verfügung. Sein Einsatz im Rahmen der FEM ist dort sinnvoll, wo große Vektorlängen verarbeitet werden, d. h. in erster Linie bei Operationen mit der Systemsteifigkeitsmatrix.
Bei der Vektorisierung eines bestehenden FEM-Programmes darf nicht nur der konventionelle Algorithmus berücksichtigt werden, sondern auch andere Algorithmen, die bisher nicht konkurrenzfähig waren, müssen in die Untersuchung einbezogen werden, da sie auf Grund spezieller Eigenschaften, z. B. guter Vektorisierbarkeit, beim Einsatz von Vektorprozessoren Rechenzeitvorteile ermöglichen können.
Am Beispiel der Konjugierten Gradienten-Verfahren (CG-Verfahren) wurde die Vektorisierung eines Algorithmus demonstriert und die Anwendung des Vektorprozessors MAMO im Rahmen der FEM untersucht. An Hand von drei Testbeispielen wurde mit Hilfe der CG-Verfahren das Laufzeitverhalten bei Anwendung des Vektorprozessors gezeigt und mit dem der seriellen Verarbeitung verglichen.
Im Vergleich zum Cholesky-Verfahren wurde gezeigt, daß schon für zweidimensionale Probleme CG-Verfahren mit Nutzung des MAMO konkurrenzfähig sind. Werden die CG-Verfahren im FEM-Programm bewußt mit ihren vorteiligen Eigenschaften eingesetzt, z. B. Kontaktiteration mit der Methode der variablen Steifigkeit, sind gegenüber dem Cholesky-Verfahren mit serieller Verarbeitung Rechenzeitgewinne zu erwarten. Diese Aussage bezieht sich auf eine Problemklasse von ca. 1000 Freiheitsgraden, die es gestattet, die Steifigkeitsmatrix im Hauptspeicher zu halten.
Außer der hier vorgestellten normalen Verarbeitungsvariante, bei der Matrizen und Vektoren in der üblichen Form mit Null-Elementen gespeichert und verarbeitet werden, bietet der MAMO die Möglichkeit der indizierten Verarbeitung. Dabei werden in einem speziellen Format nur die Nicht-Null-Elemente der Matrizen und Vektoren gespeichert und verarbeitet. Mit der indizierten Verarbeitung entfallen die Probleme der Bandbreitenoptimierung und für die schwach besetzten Matrizen in der FEM kann eine weitere Effektivitätssteigerung erwartet werden.

LITERATUR

[1] Aksenow, W. P., et al. Struktura i charakteristika wysokoproiswoditelnych EWM i sistem, Sarubeshnaja Radioelektronika (1982) H. 3 u. 4.

[2] Hoßfeld, F.: Parallelverarbeitung – Konzepte und Perspektive. Angewandte Informatik 1980, H. 12, S. 485.

[3] Kascic, M. J.: Vector processing on the CYBER 200. Angewandte Informatik 1980, H. 1, S. 27.

[4] Moltschanow, I. N.: Probleme der Parallelverarbeitung. edv-aspekte 1982, H. 4, S. 21.

[5] Geißler, R., et al.: Matrixmodul. Rechentechnik/Datenverarbeitung 1981, H. 2, S. 32.

[6] Krause, G.: Architectural and functional features of matrix modul. Proceedings of IFIP 83, p. 827 – 831, Elsevier Science Publisher B. V. (North-Holland).

[7] Krause, G.: Der Matrixmodul und seine Anwendung. Neue Technik im Büro 26 (1982), H. 4, S. 104.

[8] Systemunterlagen – Dokumentation. Befehlsbeschreibung EC2655.C003. VEB Robotron, Zentrum für Forschung und Technik, A 2013–0001.

[9] Systemunterlagen – Dokumentation. Matrixmodul EC 2655 – Benutzungshinweise für die Arbeit im OS/ES. VEB Robotron, Zentrum für Forschung und Technik, C 5313–0003.

[10] Bergmann, W.: Die Programmbibliothek ZFK.MAMO. ZfK Rossendorf, 1055–Anwenderinformationen, 2. Fortsetzung, Nov. 1983.

[11] Bergmann, W.: Handbuch für die Anwendung des MAMO auf der Basis der Unterprogrammbibliothek ZFK.MAMO. ZfK Rossendorf, 1055–Anwenderinformationen, 5. Fortsetzung, März 1985.

[12] Bergmann, W.: Über Besonderheiten von Algorithmen und Programmierung mathematischer Verfahren für den Matrix-Prozessor MAMO. Report ZfK–545, ZfK Rossendorf 1985.

[13] Altenbach, J. (Hrsg.): Die Methode der finiten Elemente in der Festkörpermechanik. VEB Fachbuchverlag Leipzig 1983.

[14] Diekkämper, R.: Vektorrechnerorientierte FEM-Analyse bei nichtlinearen Problemen. Verlagsgesellschaft R. Müller, Köln 1982.

[15] Christow, D., Todorow, M.: Anwendung der Konjugierten-Gradienten-Methode bei der Lösung elastoplastischer Aufgaben mit Hilfe der FEM. Technische Mechanik 6 (1985), H. 2, S. 51

[16] Bilfinger, T., Schmidt, F.: Vergleich verschiedener Verfahren zur Lösung linearer Gleichungssysteme. Report IKE–4–76, Universität Stuttgart 1978.

[17] Reid, J. K.: A FORTRAN subroutine for the solution of large sparse sets of linear equations by conjugate gradients. Report AERE–R 6545, Harwell, Berkshire 1970.

[18] Linek, J.: Algorithmus zur Lösung des Kontaktproblems im Programm FEMROT. In: Report ZfK–571, ZfK Rossendorf Dezember 1985.

Anschrift des Verfassers:

Dipl.-Math. Joachim Linek
Akademie der Wissenschaften der DDR
Zentralinstitut für Kernforschung Rossendorf
Postfach 19

Dresden
DDR–8051