95_13 A. Meyer, D. Michael. Some remarks on the simulation of elasto-plastic problems on parallel computers. March 1995

95_14 B. Heinrich, S. Nicaise, B. Weber. Elliptic interface problems in axisymmetric domains. Part I: Singular functions of non-tensorial type. April 1995

95_16 W. Rath. Canonical forms for linear descriptor systems with variable coefficients. May 1995

95_17 C. He, A. J. Laub, V. Mehrmann. Placing plenty of poles is pretty preposterous. May 1995

95_18 J. J. Hench, C. He, V. Kučera, V. Mehrmann. Dampening controllers via a Riccati equation approach. May 1995

95_19 M. Meisel, A. Meyer. Kommunikationstechnologien beim parallelen vorkonditionierten Schur-Komplement CG-Verfahren. Juni 1995

95_20 G. Haase, T. Hommel, A. Meyer and M. Pester. Bibliotheken zur Entwicklung paralleler Algorithmen. Juni 1995.

95_21 A. Vogel. Solvers for Lamé equations with Poisson ratio near 0.5. June 1995.

95_22 P. Benner, A. J. Laub, V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations I: Continuous-time case. July 1995.

95_23 P. Benner, A. J. Laub, V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations II: Discrete-time case. July 1995.

95_24 P. Benner, R. Byers. Newton's method with exact line search for solving the algebraic Riccati equation. July 1995.

Some papers can be accessed via anonymous ftp from server `ftp.tu-chemnitz.de`, directory `pub/Local/mathematik/SPC`. (Note the capital L in Local!)

Technische Universität Chemnitz-Zwickau

DFG-Forschergruppe "SPC" · Fakultät für Mathematik

Christoph Israel

NETGEN69
Ein hierarchischer paralleler Netzgenerator

Fakultät für Mathematik
TU Chemnitz-Zwickau
D-09107 Chemnitz, FRG
(0371)-531-4685
israel@imech.tu-chemnitz.de

Preprint-Reihe der Chemnitzer DFG-Forschergruppe "Scientific Parallel Computing"

SPC 95_26 **September 1995**

NETGEN69

Ein hierarchischer paralleler Netzgenerator

Summary

In this paper a hierarchical parallel mesh generator is presented. Starting with a coarse mesh a fine mesh of 6-nodes-triangles or 9-nodes-rectangles is generated after performing a given number of refinement steps. Within a parallel finite element program the mesh generator yields the hierarchical mesh for every processor including the boundary conditions, material data and essential information for communication.

Inhaltsverzeichnis

94_17 B. Heinrich. The Fourier-finite-element method for Poisson's equation in axisymmetric domains with edges. August 1994.

94_18 M. Pester and S. Rjasanow. A parallel preconditioned iterative realization of the panel method in 3D. September 1994.

94_19 A. Meyer. Preconditoning the Pseudo-Laplacian for Finite Element simulation of incompressible flow. October 1994.

94_20 V. Mehrmann. A step towards a unified treatment of continous and discrete time control problems. October 1994.

94_21 C. He, V. Mehrmann. Stabilization of large linear systems. October 1994.

94_22 B. Heinrich and B. Weber. The Fourier-finite-element method for three-dimensional elliptic problems with axisymmetric interfaces. November 1994.

94_23 M. Pester. On-line visualization in parallel computations. November 1994.

94_24 M. Pester. Grafik-Ausgabe vom Parallelrechner für 2D-Gebiete. November 1994.

94_25 R. Byers, C. He, V. Mehrmann. The Matrix Sign Function Method and the Computation of Invariant Subspaces. November 1994.

94_26 T. Apel, G. Lube. Local inequalities for anisotropic finite elements and their application to convection-diffusion problems. Dezember 1994.

94_27 P. Kunkel, V. Mehrmann. Analysis und Numerik linearer differentiell-algebraischer Gleichungen. Dezember 1994.

95_1 T. Apel, G. Lube. Anisotropic mesh refinement in stabilized Galerkin methods Januar 1995.

95_2 M. Meisel, A. Meyer. Implementierung eines parallelen vorkonditionierten Schur-Komplement CG-Verfahrens in das Programmpaket FEAP. Januar 1995.

95_3 S. V. Nepomnyaschikh. Optimal multilevel extension operators. January 1995

95_4 M. Meyer. Grafik-Ausgabe vom Parallelrechner für 3D-Gebiete. Januar 1995

95_5 T. Apel, G. Haase, A. Meyer, M. Pester. Parallel solution of finite element equation systems: efficient inter-processor communication. Februar 1995

95_6 U. Groh. Ein technologisches Konzept zur Erzeugung adaptiver hierarchischer Netze für FEM-Schemata. Mai 1995

95_7 M. Bollhöfer, C. He, V. Mehrmann. Modified block Jacobi preconditioners for the conjugate gradient method. Part I: The positive definit case. January 1995

95_8 P. Kunkel, V. Mehrmann, W. Rath, J. Weickert. GELDA: A Software Package for the Solution of General Linear Differential Algebraic Equation. February 1995

95_9 H. Matthes. A DD preconditioner for the clamped plate problem. February 1995

95_10 G. Kunert. Ein Residuenfehlerschätzer für anisotrope Tetraedernetze und Dreiecksnetze in der Finite-Elemente-Methode. März 1995

95_11 M. Bollhöfer. Algebraic Domain Decomposition. March 1995

95_12 B. Nkemzi. Partielle Fourierdekomposition für das lineare Elastizitätsproblem in rotationssymmetrischen Gebieten. März 1995

Other titles in the SPC series:

93_1 G. Haase, T. Hommel, A. Meyer and M. Pester. Bibliotheken zur Entwicklung paralleler Algorithmen. May 1993.

93_2 M. Pester and S. Rjasanow. A parallel version of the preconditioned conjugate gradient method for boundary element equations. June 1993.

93_3 G. Globisch. PARMESH – a parallel mesh generator. June 1993.
in: Parallel Computing 21, No. 3, March 1995, pp. 509-524.

94_1 J. Weickert and T. Steidten. Efficient time step parallelization of full-multigrid techniques. January 1994.

94_2 U. Groh. Lokale Realisierung von Vektoroperationen auf Parallelrechnern.
March 1994.

94_3 A. Meyer. Preconditoning the Pseudo-Laplacian for Finite Element simulation of incompressible flow. February 1994.

94_4 M. Pester. Bibliotheken zur Entwicklung paralleler Algorithmen. (aktualisierte Fassung). March 1994.

94_5 U. Groh, Chr. Israel, St. Meinel and A. Meyer. On the numerical simulation of coupled transient problems on MIMD parallel systems. April 1994.

94_6 G. Globisch. On an automatically parallel generation technique for tetrahedral meshes. April 1994.

94_7 K. Bernert. Tauextrapolation – theoretische Grundlagen, numerische Experimente und Anwendungen auf die Navier-Stokes-Gleichungen. June 1994.

94_8 G. Haase, U. Langer, A. Meyer and S. V. Nepomnyaschikh. Hierarchical extension and local multigrid methods in domain decomposition preconditoners. June 1994.

94_9 G. Kunert. On the choice of the basis transformation for the definition of DD Dirichlet preconditioners. June 1994.

94_10 M. Pester and T. Steidten. Parallel implementation of the Fourier Finite Element Method. June 1994.

94_11 M. Jung and U. Rüde. Implicit Extrapolation Methods for Multilevel Finite Element Computations: Theory and Applications. June 1994.

94_12 A. Meyer and M. Pester. Verarbeitung von Sparse-Matrizen in Kompaktspeicherform KLZ/KZU. June 1994.

94_13 B. Heinrich and B. Weber. Singularities of the solution of axisymmetric elliptic interface problems. June 1994.

94_14 K. Gürlebeck, A. Hommel and T. Steidten. The method of lumped masses in cylindrical coordinates. July 1994.

94_15 Th. Apel and F. Milde. Realization and comparison of various mesh refinement strategies near edges. August 1994.

94_16 Th. Apel and S. Nicaise. Elliptic problems in domains with edges: anisotropic regularity and anisotropic finite element meshes. August 1994.

1 Einleitung

Für die numerische Lösung zweidimensionaler Probleme der Kontinuumsmechanik auf Parallelrechnern [3] wurde ein Netzgenerator entwickelt, der - ausgehend von einem Grobnetz und der Angabe einer gewünschten Anzahl von Verfeinerungsstufen - ein gleichmäßig hierarchisches Feinnetz liefert.
Neben der parallelen Assemblierung der FEM-Matrizen und der Verwendung paralleler Algorithmen zur effektiven Lösung der entstehenden Gleichungssysteme [5], [8] ist es wichtig, bereits das Finite-Elemente-Netz vollständig parallel zu generieren [1]. Das beschleunigt zum einen die Gesamtlösung des Problems, zum anderen liefert jeder Prozessor nach der Generierung des lokalen Feinnetzes bereits die komplette Datenstruktur, die für die parallele Berechnung der Lösung erforderlich ist.
Den Ausgangspunkt bildet ein vom Nutzer zu definierendes Grobnetz, das eine vollständige Gebietszerlegung des Gesamtgebietes Ω in mehrere, einfach zusammenhängende Teilgebiete Ω_i realisiert. Diese Gebietszerlegung ergibt sich zum Beispiel bei einer Vereinigung von Teilgebieten mit verschiedenen Materialeigenschaften von selbst. Wenn solche natürlichen Gründe für eine spezielle Gebietszerlegung nicht vorliegen, erfolgt eine nutzerdefinierte Zerlegung in Dreiecks- oder Viereckselemente, die meistens unter dem Aspekt einer gut ausbalancierten Lastverteilung auf dem Parallelrechner gewählt wird.
Der Netzgenerator NETGEN69 generiert daraus ein Feinnetz aus 6-Knoten-Dreiecks-Elementen oder 9-Knoten-Vierecks-Elementen, wobei der Elementtyp mit dem des Grobnetzes übereinstimmt.
Der Netzgenerator arbeitet kantenorientiert. Die zweidimensionalen Flächen (Dreiecke beziehungsweise Vierecke) werden durch die zugehörigen Kanten repräsentiert, die Randbedingungsinformationen beziehen sich auf die Kanten, und die hierarchische Verfeinerung erfolgt im wesentlichen durch Kantenteilung.

Der Abschnitt **2** beschreibt die Struktur des Eingabedatenfiles für NETGEN69 und zeigt zwei Möglichkeiten der Erstellung dieses Files.

Im Abschnitt **3** wird der Ablauf der parallelen Netzgenerierung von der Eingabe der Grobnetzdaten und Steuerparameter über die eigentliche hierarchische Generierung bis zur grafischen Ausgabe des Feinnetzes erläutert.

Der Abschnitt **4** gibt eine Beschreibung der Struktur der skalaren und vektoriellen Daten, die NETGEN69 als Schnittstelle für die weitere parallele Behandlung des FEM-Problems zur Verfügung stellt. Diese Datenstruktur ist den Erfordernissen für eine massiv parallele Lösung von FEM-Problemen in der Chemnitzer DFG-Forschergruppe „Scientific Parallel Computing“ [6] voll angepaßt.

Im Abschnitt **5** werden abschließend zwei Beispiele für 2D-Netze gezeigt, die mit dem parallelen Netzgenerator NETGEN69 erzeugt wurden. An diesen Beispielen wird die Einbindung des Netzgenerators in das Finite- Element-Programm „SPC-PM CFD“ zur parallelen Strömungssimulation [4] demonstriert.

2 Die Struktur des Eingabedatenfiles

Das Eingabedatenfile besteht aus

- der Versionsnummer,
- der Kopfzeile sowie
- sechs Datenblöcken.

• Die Versionsnummer bestimmt den Umfang der Informationen des Eingabedatenfiles und hat Einfluß auf dessen Struktur. Zur Zeit existieren die Versionen

0 : Grobnetz + Randbedingungen
1 : Grobnetz + Randbedingungen + Materialbereiche.

• Die Kopfzeile enthält in der Version 0 die Anzahl der

- Knoten Nn
- Kanten Ne
- Flächen Nf des Grobnetzes
- Freiheitsgrade NFhg des Problems
- Kanten mit Dirichlet-Randbedingungen NDir
- Kanten mit Neumann-Randbedingungen NNeum .

Bei der Version 1 kommen zwei Informationen dazu:

- die Anzahl der Materialbereiche NMat
- die maximale Anzahl der Materialparameter pro Materialbereich Nmax.

• Der erste Datenblock repräsentiert die Lage der Nn Knoten im Grobnetz.
Für jeden Knoten sind
- ein Knotenname
- die x-Koordinate
- die y-Koordinate
- ein Knotenanzeiger (nur für Strömungssimulation)

im kartesischen Koordinatensystem ausgewiesen.

• Im zweiten Datenblock sind die Kanteninformationen gespeichert.
Alle Ne Kanten des Grobnetzes werden durch
- einen Kantennamen
- den Startknoten
- den Endknoten
- den Mittelknoten
- die Kantenform

dargestellt.
Dabei kann eine Kante entweder durch drei oder durch zwei Knoten beschrieben werden.
Im letzteren Fall wird der Mittelknoten mit "0"gekennzeichnet.

Als mögliche Kantenformen wurden realisiert:

0		Gerade
1	für	Kreisbogen
2		Parabel .

Literatur

[1] G. Globisch: PARMESH - a parallel mesh generator, Preprint-Reihe der *Chemnitzer DFG-Forschergruppe „Scientific Parallel Computing“*, (SPC 93_3, Juni 1993) .

[2] U. Groh: Ein technologisches Konzept zur Erzeugung adaptiver hierarchischer Netze für FEM-Schemata, Preprint-Reihe der *Chemnitzer DFG-Forschergruppe „Scientific Parallel Computing“*, (SPC 95_6, Mai 1995) .

[3] U. Groh, C. Israel, S. Meinel, A. Meyer: On the Numerical Simulation of Coupled Transient Problems on MIMD Parallel Systems, Preprint-Reihe der *Chemnitzer DFG-Forschergruppe „Scientific Parallel Computing“*, (SPC 94_5, April 1994) .

[4] U. Groh, S. Meinel, A. Meyer, C. Israel: „SPC-PM CFD“ - Programm-Modul 2D-CFD-Simulation, Version 1.00, 1994.

[5] G. Haase, U. Langer, A. Meyer: Parallelisierung und Vorkonditionierung des CG-Verfahrens durch Gebietszerlegung. In: Bader, Rannacher, Wittum (Hrsg.), Numerische Algorithmen auf Transputersystemen, B. G. Teubner Stuttgart 1993.

[6] B. Heinrich, U. Langer, A. Meyer, M. Pester: Algorithmische Grundlagen der Simulation von angewandten Problemen der Kontinuumsmechanik auf massiv parallelen Rechnern. *Antrag zur Gründung einer DFG-Forschergruppe an der Technischen Universität Chemnitz*, Mai 1992.

[7] C. Israel: Ein hierarchischer paralleler 2D-Netzgenerator für 6-Knoten-Dreiecke, in: M.Pester,ed.,*Proceedings zum Workshop „Paralleles Pre- und Postprocessing“*, (DFG-Forschergruppe SPC, Fachbereich Mathematik, Technische Universität Chemnitz, Juni 1993).

[8] A. Meyer: A parallel preconditioned conjugate gradient method using domain decomposition and inexact solvers on each subdomain. Computing, *45*, 217-234 (1990).

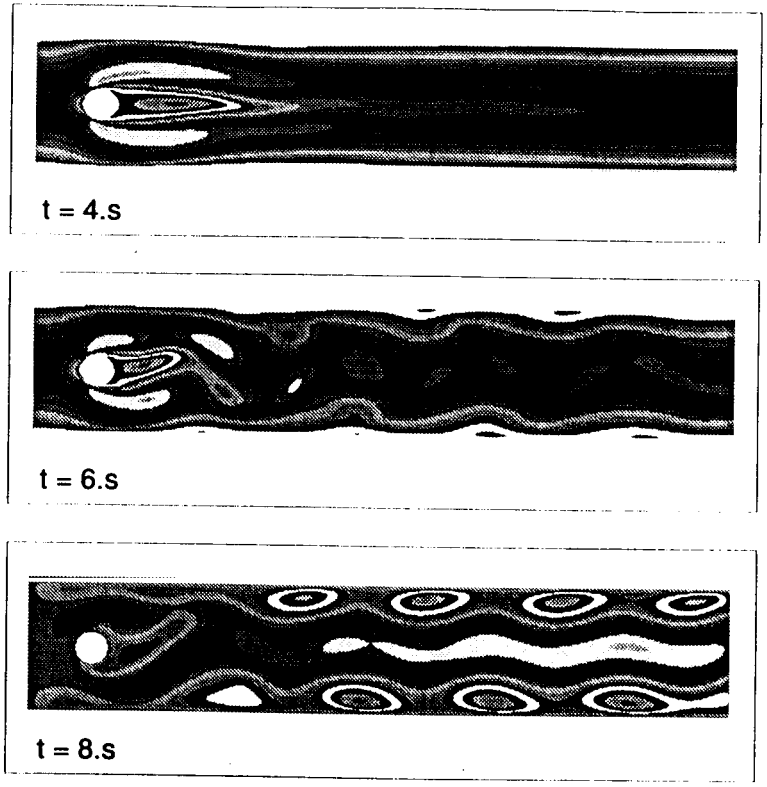
[9] A. Meyer, S. Meinel, U. Groh, M. Pester: Benchmarkergebnisse (Inkompressible laminare 2d-Strömungen), DFG-SPP „Strömungssimulation mit Hochleistungsrechnern“, 1995.

[10] M. Pester: Grafik-Ausgabe vom Parallelrechner für 2D-Gebiete, Preprint-Reihe der *Chemnitzer DFG-Forschergruppe „Scientific Parallel Computing“*, (SPC 94_24, November 1994) .

[11] M. Seibt: Dokumentation zu NET.EXE v2.2 - Werkzeug zur Eingabe und Bearbeitung 2-dimensionaler Netze. Belegarbeit, TU Chemnitz-Zwickau, Januar 1994, Update für Strömungssimulationen NET.EXE v3.0, März 1995.

Adresse des Autors:

Dipl.-Math. Christoph Israel, Technische Universität Chemnitz-Zwickau,
Fakultät für Mathematik, 09107 Chemnitz



In den **Abbildungen 15,16 und 17** werden die x-Komponente der Geschwindigkeit dieser Strömung zu drei verschiedenen Zeitpunkten dargestellt.
Zum Endzeitpunkt t = 8 s gehört die Stromliniendarstellung (Abbildung 11) vom Beginn dieses Abschnittes.

6 Ausblick

Der Netzgenerator NETGEN69 erzeugt durch globale hierarchische Verfeinerung aus einem Grobnetz in einer vorzugebenden Anzahl von Verfeinerungsstufen ein FEM-Netz aus 6-Knoten-Dreiecken oder 9-Knoten-Vierecken.
NETGEN69 arbeitet kantenorientiert und nach der Aufteilung des Grobnetzes auf die Prozessoren vollstaendig parallel.
Der Netzgenerator dient als Grundlage fuer adaptive hierarchische Netze. Das Grundkonzept fuer diese Entwicklung wurde von GROH in [2] beschrieben.

• Der dritte Datenblock enthält die Beschreibung der Nf zweidimensionalen Flächen (Dreiecke oder Vierecke) des Grobnetzes. Dabei werden ein Dreieck durch drei Kanten und ein Viereck durch vier zusammenhängende Kanten repräsentiert.

Jede Fläche wird in der Version 0 durch
- einen Flächennamen
- 3 Kantennamen (Dreieck) / 4 Kantennamen (Viereck)

beschrieben.

In der Version 1 wird jede Fläche zusätzlich mit einer Materialbereichsnummer gekennzeichnet.

• Im vierten Datenblock sind Informationen für die NDir Kanten gespeichert, die in mindestens einem Freiheitsgrad Dirichlet-Randbedingungen tragen.
Die Informationen bestehen aus
- dem Namen der Kante
- einem Randbedingungscode
- jeweils drei Randwerten für Startknoten, Endknoten und Mittelknoten der Kante für jeden der NFhg Freiheitsgrade.

Dabei werden im Eingabedatenfile für alle Freiheitsgrade drei Randwerte eingetragen, auch wenn die Kante nur für ausgewählte Freiheitsgrade Randbedingungen trägt. Alle eingetragenen Randbedingungs-Informationen werden den im Generierungsprozeß neu entstehenden Teilkanten weitervererbt beziehungsweise durch Interpolation angepaßt.
Nach der Netzgenerierung kann anhand des Randbedingungscodes ausgewählt werden, für welchen Freiheitsgrad die eingegebenen beziehungsweise interpolierten Randwerte Gültigkeit besitzen.
Der Randbedingungscode Idncode wird gebildet als Summe aus Zweierpotenzen der gültigen Randwerte-Tripel.
Dabei bezeichnen bei fest vorgegebener Reihenfolge der Freiheitsgrade 2^0 den ersten Freiheitsgrad, ... , 2^{NFhg-1} den letzten der NFhg Freiheitsgrade. Sind zum Beispiel an einer Kante Dirichlet-Randbedingungen für alle Freiheitsgrade vorgegeben, so erhält Idncode den maximalen Wert $2^{NFhg} - 1$. Falls NDir=0, dann entfällt dieser Datenblock.

• Im fünften Datenblock sind die Informationen für die NNeum Kanten enthalten, die in mindestens einem Freiheitsgrad Neumann-Randbedingungen tragen.
Die Datenstruktur dieses Blockes stimmt mit der des Datenblockes für die Dirichlet-Randbedingungen völlig überein. Der Datenblock entfällt, falls NNeum=0.

• Der sechste Datenblock (nur bei Version 1) beschreibt die Materialdaten aller Materialbereiche des Grobnetzes. Für jeden Materialbereich werden zwei Zeilen eingetragen.
Die erste Zeile enthält
- die Materialbereichsnummer MNr
- die tatsächliche Anzahl der Materialparameter MPar dieses Materialbereichs (MPar <= Nmax).

Die zweite Zeile enthält
- die Werte der Materialparameter.

Damit ist die Datenstruktur für das Eingabefile komplett beschrieben. Nach der Aufteilung aller Daten auf die verfügbaren Prozessoren wird vollständig parallel in Abhängigkeit von der Anzahl der gewünschten Hierarchielevel das Feinnetz generiert. Die Daten für die Randbedingungen und die Materialbereiche (sowie für Strömungssimulationen die Knotenanzeiger) werden bei der Netzgenerierung mitgeführt und an das entsprechende Feinnetz angepaßt. Sie können anschließend vom Nutzer im Programm in geeigneter Weise bei der Assemblierung der Matrizen und der rechten Seite eingebaut werden.

Für die Erstellung des Eingabedatenfiles gibt es im wesentlichen zwei Möglichkeiten:

1. Das Eingabedatenfile wird "von Hand"geschrieben. Das ist vor allem bei einem kleinen Grobnetz sinnvoll, das aus relativ wenigen Knoten, Kanten und Flächen besteht. Diese Herangehensweise kann auch dann angewendet werden, wenn der Nutzer auf eine spezielle Numerierung der Knoten, Kanten und Flächen Wert legt. Zunächst wird das zu vernetzende Gesamtgebiet Ω in eine endliche Anzahl Nf von einfach zusammenhängenden Teilgebieten Ω_i (i=1,Nf) zerlegt. Diese bilden die Flächen (Dreieck/Viereck).

   Eine Dreiecksfläche wird durch die Angabe von drei Kantennamen, eine Viereckfläche durch vier zusammenhängende Kanten beschrieben.

   Eine Kante im Eingabedatenfile besteht aus zwei oder drei Knoten. Eine Gerade ist durch die Angabe von zwei Knoten (Anfangs- und Endknoten) eindeutig bestimmt, während für eine krummlinige Kante (Kreisbogen oder Parabelstück) natürlicherweise die Angabe eines dritten Knotens (Mittelknoten) erforderlich ist. Der Richtungssinn der Kante spielt bei der Eingabe keine Rolle, er wird aber bei der Verfeinerung beibehalten.

   Wird der Netzgenerator für eine nachfolgende Strömungssimulation verwendet,so erhält jeder Knoten neben seiner Definition durch x-Koordinate und y-Koordinate noch einen spezifischen Knotenanzeiger, mit dem besondere Eigenschaften des Knotens gekennzeichnet werden.

   Zur Zeit werden folgende Knotenmarkierungen verwendet:

   0 : Knoten ohne besondere Merkmale
   128 : Knoten innerhalb eines Festkörpers
   256 : zeitlich veränderliche Dirichlet-Randbedingungen
   512 : Umschaltbereich von Dirichlet- zu Neumann-Randbedingungen, zum Beispiel bei Verbrennungsvorgängen
   1024: zeitlich veränderliche Neumann-Randbedingungen
   2048: Kennzeichnung des Randes zur Berechnung von Oberflächenkräften.

Die Abbildung 12 zeigt das generierte Gesamtnetz nach 1 Verfeinerungsschritt.
Die Abbildungen 13 und 14 demonstrieren die „Zooming-Funktion" des Postprocessing-Moduls.
Die Abbildung 13 stellt den Ausschnitt um die Kreisöffnung im Level-3-Netz dar, wobei die Vergrößerung um den Kreismittelpunkt (20,20) mit dem Radius von 25 Einheiten gewählt wurde.
Eine weitere Vergrößerung dieses Gebietes zeigt die Abbildung 14, hier ist das Netz nach 4 Verfeinerungsschritten mit einem Radius von 9 Einheiten um den Kreismittelpunkt (20,20) dargestellt.

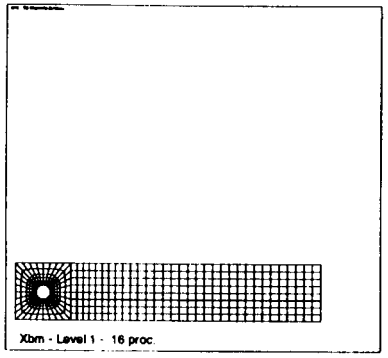


Abbildung 12: Level-1-Netz

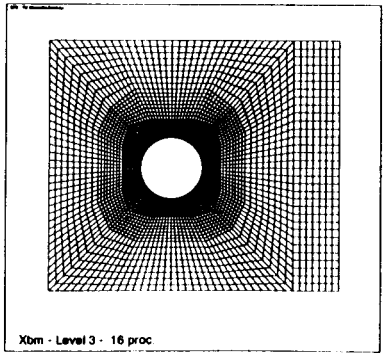


Abbildung 13: Level-3-Netz (Zooming)

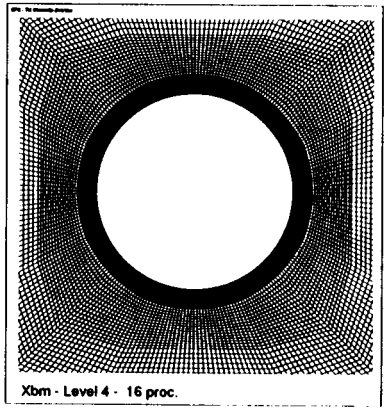


Abbildung 14: Level-4-Netz (Zooming)

5.2 Kreiszylinderumströmung im ebenen Kanal mit zeitlich veränderlicher Zuströmung

Das Anwendungsbeispiel wurde zu Vergleichszwecken einer Benchmarkserie des DFG-SPP „Strömungssimulation mit Hochleistungsrechnern" gerechnet [9].

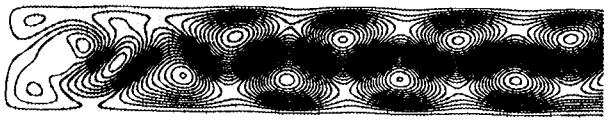


Abbildung 11

Dieses Beispiel steht für eine Vierecks-Grobvernetzung. Daraus wird ein Feinnetz aus 9-Knoten-Vierecken generiert.
Es handelt sich um einen ebenen Kanal mit einer asymmetrisch angeordneten Kreisöffnung im Inneren. Als Randbedingungen sind vorgegeben:

- Randbedingungen 1. Art für die Geschwindigkeit:

  x-Komponente: - parabolisches Profil ($*sin(\pi/8 * t)$) am Einströmrand
  - $u_x = 0$ an den anderen Außenrändern und am Innenkreisrand (außer Ausströmrand)

  y-Komponente: - $u_y = 0$ an den Außenrändern und am Innenkreisrand (außer Ausströmrand)

- Ausströmrand

Das Grobnetz besteht aus 196 Knoten, 288 Kanten und 128 Viereckflächen. Die Rechnungen wurden auf 16 und auf 32 Prozessoren durchgeführt.

Im Falle der Aufteilung des Grobnetzes auf 16 Prozessoren ergaben sich in Abhängigkeit von der Anzahl der Verfeinerungslevel folgende Feinnetzdaten:

	- lokal -				- global -			
Levels	NK	NC	NI	NEL	NK	NC	NI	NEL
1	153	81	72	32	2176	1024	1152	512
2	561	169	392	128	8448	2176	6272	2048
3	2145	345	1800	512	33280	4480	28800	8192
4	8385	697	7688	2048	132096	9088	123008	32768

Dabei bezeichnen:
- NK : die Anzahl der Gesamtknoten
- NC : die Anzahl der "Koppelknoten"(siehe Abschnitt 4.1)
- NI : die Anzahl der inneren Knoten
- NEL : die Anzahl der Elemente.

Die lokalen Daten stehen für das generierte Teilnetz auf einem Prozessor, die globalen Daten repräsentieren das Gesamtnetz auf allen 16 Prozessoren.

Die Randbedingungen 1. und 2.Art beziehen sich im Eingabedatenfile jeweils auf vorher definierte Kanten.

Sind Materialdaten vorgegeben (Version 1), wird jede Fläche mit der entsprechenden Materialbereichsnummer versehen. Außerdem werden die Materialdaten zur Beschreibung der Materialbereiche nach dem Randbedingungsblock aufgelistet.

Die Versionsnummer und die Kopfzeile am Anfang komplettieren das Eingabedatenfile.

Existiert einmal ein solches "von Hand"geschriebenes File, dann lassen sich durch Kopieren und entsprechende Modifizierungen leicht Eingabedatenfiles für andere Gebiete und Aufgabenstellungen erzeugen.

2. Das Eingabedatenfile wird grafisch-interaktiv erstellt. Dazu wurde von SEIBT [11] ein MS/DOS-Programm für den PC geschaffen.
   Es erlaubt eine mausgesteuerte Eingabe der Knoten, Kanten und Flächen, weist den vom Nutzer ausgewählten Kanten ihre Randbedingungen zu und belegt die Flächen mit der zugehörigen Materialbereichsnummer. Eine Vielzahl grafischer Tools und eine während des Programmlaufs abrufbare Programmbeschreibung erleichtern die Arbeit der grafischen Erzeugung des Eingabedatenfiles.

Tabelle 1 : Die Struktur des Eingabedatenfiles von NETGEN69

1. Block: die erste Zeile des NETGEN69-Eingabedatenfiles:

MVERS	die Versionsnummer für die Dateneingabe (0/1)

2. Block: die zweite Zeile des NETGEN69-Eingabedatenfiles:

NN	die Anzahl der Grobnetzknoten
NE	die Anzahl der Grobnetzkanten
NF	die Anzahl der Grobnetzflächen
NFHG	die Anzahl der Problemfreiheitsgrade
NDIR	die Anzahl der Kanten mit Dirichlet-Randbedingungen
NNEUM	die Anzahl der Kanten mit Neumann-Randbedingungen
- nur in der Version 1:	
NMAT	die Anzahl der Materialbereiche
NMAX	die maximale Anzahl der Materialparameter pro Materialbereich

3. Block: der Datenblock zur Knotenbeschreibung:
bestehend aus NN Zeilen; für jeden Knoten:

I	der globale Knotenname
X	die x-Koordinate des Knotens
Y	die y-Koordinate des Knotens
KIND	der Knotenanzeiger (nur für Strömungssimulation)

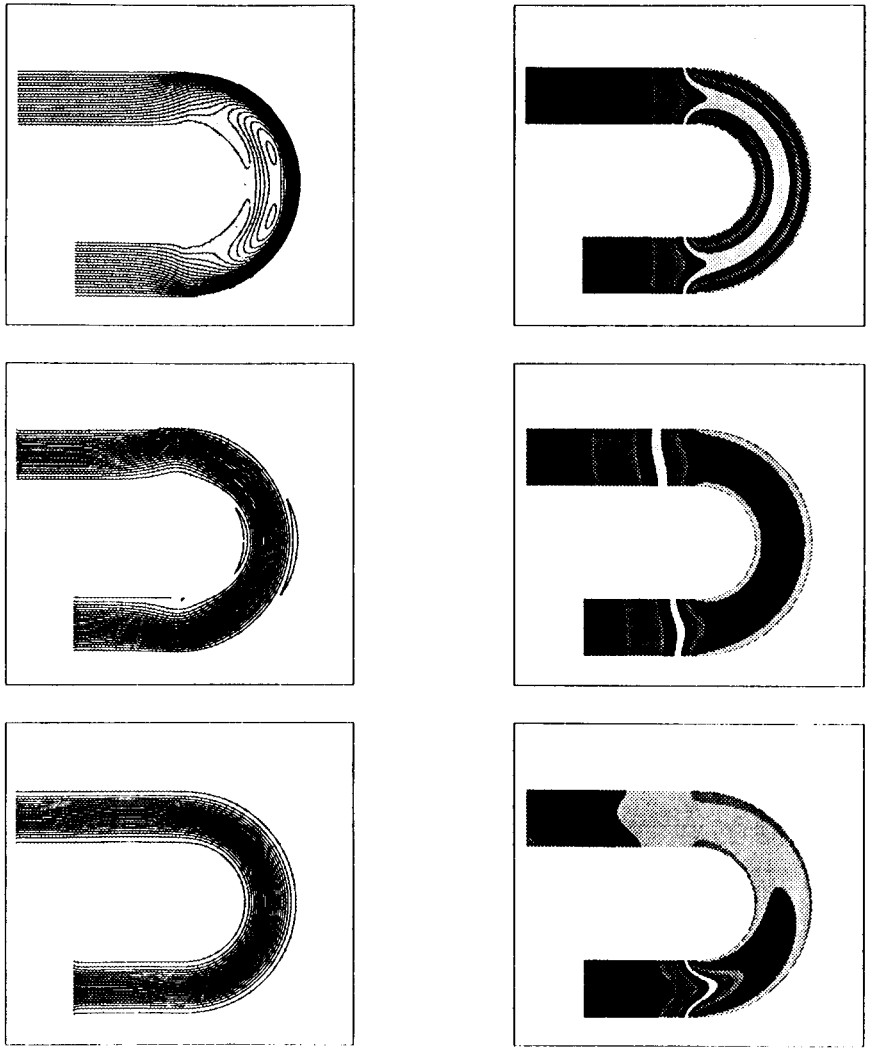
4. Block: der Datenblock zur Kantenbeschreibung:
bestehend aus NE Zeilen; für jede Kante:

I	der globale Kantenname
KNS	der Startknoten der Kante
KNE	der Endknoten der Kante
KNM	der Mittelknoten der Kante (falls vorhanden, sonst 0)
KAFORM	die Kantenform

5. Block: der Datenblock zur Beschreibung der 2D-Flächen:
bestehend aus NF Zeilen; für jede Fläche:

I	der Flächenname des Dreiecks/Vierecks
KA1	der Name der 1. Kante
KA2	der Name der 2. Kante
KA3	der Name der 3. Kante
KA4	der Name der 4. Kante (falls Viereck)
- nur in der Version 1:	
MNR	die Materialbereichsnummer

Den Verlauf des mit einem Level-4-Netz gerechneten instationären Strömungs- und Wärmetransport-Problems beschreiben die folgenden grafischen Darstellungen.
Dabei zeigt die linke Seite (Abbildungen 5,6,7) den Stromlinienverlauf zu drei ausgewählten Zeitpunkten, während die rechte Seite (Abbildungen 8,9,10) die zum jeweiligen Zeitpunkt zugehörige Temperaturverteilung darstellt.



Abbildungen 5 - 10 : Stromlinien- und Temperaturverlauf in drei Zeitpunkten

Das Grobnetz besteht aus 51 Knoten, 33 Kanten und 16 Dreiecksflächen. Die Rechnung wurde auf 16 Prozessoren durchgeführt, so daß jeder Prozessor genau ein Grobdreieck zu verfeinern hatte. In Abhängigkeit von der Anzahl der Verfeinerungslevel ergaben sich folgende Feinnetzdaten:

	- lokal -				- global -			
Levels	NK	NC	NI	NEL	NK	NC	NI	NEL
1	15	12	3	4	165	117	48	64
2	45	24	21	16	585	249	336	256
3	153	48	105	64	2193	513	1680	1024
4	561	96	465	256	8481	1041	7440	4096
5	2145	192	1953	1024	33345	2097	31248	16384

Dabei bezeichnen:
- NK : die Anzahl der Gesamtknoten
- NC : die Anzahl der "Koppelknoten"(siehe Abschnitt 4.1)
- NI : die Anzahl der inneren Knoten
- NEL : die Anzahl der Elemente.

Die lokalen Daten stehen für das generierte Teilnetz auf einem Prozessor, die globalen Daten repräsentieren das Gesamtnetz auf allen 16 Prozessoren.

Die Abbildungen 3 und 4 zeigen das generierte Netz nach 2 beziehungsweise nach 4 Verfeinerungsschritten.

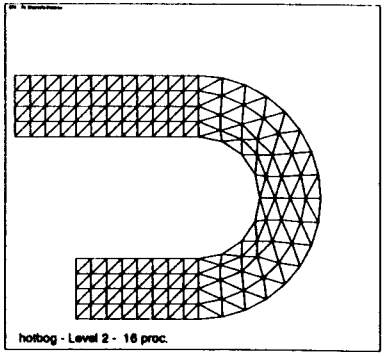


Abbildung 3: Level-2-Netz

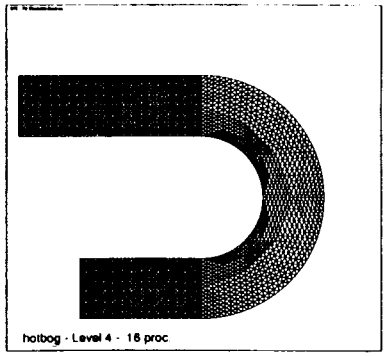


Abbildung 4: Level-4-Netz

6. Block: der Datenblock zur Beschreibung der Kanten mit Dirichlet-Randbedingungen:
bestehend aus NDIR Teilblöcken mit jeweils (NFHG+1)-Zeilen;
für jeden Teilblock:

die Zeile 1 enthält:
IKANTE den Namen der Randkante
IDCODE einen Randbedingungscode

die Zeilen 2,...,NFHG+1 enthalten:
DIR(I),I=1,...,3 die Randwerte für Startknoten, Endknoten und Mittelknoten für den jeweiligen Freiheitsgrad

7. Block: der Datenblock zur Beschreibung der Kanten mit Neumann-Randbedingungen:
bestehend aus NNEUM Teilblöcken mit jeweils (NFHG+1)-Zeilen;
für jeden Teilblock:

die Zeile 1 enthält:
IKANTE den Namen der Randkante
INCODE einen Randbedingungscode

die Zeilen 2,...,NFHG+1 enthalten:
NEU(I),I=1,...,3 die Randwerte für Startknoten, Endknoten und Mittelknoten für den jeweiligen Freiheitsgrad

8. Block: der Datenblock zur Beschreibung der Materialbereiche:
(! nur in der Version 1 !)
bestehend aus NMAT Teilblöcken mit jeweils 2 Zeilen;
für jeden Teilblock:

die Zeile 1 enthält:
MNR die Materialbereichsnummer
MPAR die Anzahl der Materialparameter dieses Materialbereichs

die Zeile 2 enthält:
MATE(I),I=1,...,MPAR die Werte der Materialparameter

Bemerkungen:
- Das Eingabedatenfile besteht aus globalen Daten, die sich auf das Gesamtgebiet Ω beziehen. Den globalen Namen für Knoten, Kanten und Flächen im File stehen lokale Numerierungen nach der Aufteilung der Daten auf die Prozessoren gegenüber, die bei der lokalen Netzgenerierung verwendet werden.

- Das Eingabedatenfile ist im ASCII-Code geschrieben. Die verschiedenen Datenblöcke werden jeweils durch den Beginn einer neuen Zeile voneinander getrennt. Das Trennzeichen zwischen den Daten innerhalb eines Blockes ist ein Komma oder mindestens ein Leerzeichen. Die Subroutinen zur Dateneingabe lesen unformatierte Daten ein.

- Alle in Tabelle 1 aufgeführten Variablen, deren Namen mit I,K,M oder N beginnen, sind vom Typ INTEGER, alle anderen Variablen und Felder tragen den Datentyp REAL.

3 Die Realisierung des parallelen Netzgenerators

Das Hauptprogramm NETGEN69 gliedert sich in vier Teile:

a) - die Eingabe der Grobnetzdaten und der Steuerparameter über den Rootprozessor

b) - die Verteilung der globalen Daten auf alle Prozessoren und einige Algorithmen, die die parallele Netzgenerierung vorbereiten

c) - die lokale Generierung des gleichmäßig hierarchischen Netzes von 6-Knoten-Dreiecken oder 9-Knoten-Vierecken

d) - die grafische Darstellung des generierten Feinnetzes .

Es werden nachfolgend die vier Teile des Netzgenerators in ihrem Ablauf erläutert, und am Ende des Abschnittes werden in zwei Abbildungen die Subroutinen-Bäume des Dreiecks- und des Vierecks-Generators dargestellt.

a) Der Eingabe- und Steuerteil beginnt mit drei interaktiven Abfragen des Programms. Der Nutzer wählt zuerst aus, ob er ein Grobnetz aus Dreiecken oder Vierecken verfeinern will.
Als nächstes ist der Name des Eingabedatenfiles anzugeben, die Erweiterung '.net' entfällt (SETFILE). In Abhängigkeit vom zuerst eingegebenen Flächentyp (Dreieck/Viereck) sucht das Programm in den Unterverzeichnissen mesh3 / mesh4 nach einem Grobnetzfile mit dem angegebenen Filenamen. Existiert das File dort nicht, erscheint eine Fehlermeldung. Ein neuer Filename ist einzugeben.
Als drittes wird die Anzahl Levels der Hierarchielevel abgefragt, die die Feinheit des entstehenden hierarchischen Netzes charakterisieren.

   Levels kann gleichzeitig zur Programmsteuerung genutzt werden, und es bedeutet:

$$Levels \begin{cases} > 0 & : \text{ Hierarchielevel zur Verfeinerung} \\ = 0 & : \text{ Programmende} \\ < 0 & : \text{ Auswahl eines neuen Datenfiles .} \end{cases}$$

   Nun wird das ausgewählte Eingabedatenfile vom Rootprozessor geöffnet und die in der ersten Zeile stehende Versionsnummer gelesen (OPENFILE). Der komplette restliche Datensatz wird in der Subroutine USERNET eingelesen, anschließend wird das File wieder geschlossen.

b) Jetzt erfolgt das Senden der globalen Eingabedaten an alle Prozessoren (USERNET) sowie die Aufteilung des Gesamtgebietes Ω in die zu dem jeweiligen Prozessor gehörenden Teilgebiete Ω_i . Von nun an arbeitet jeder Prozessor lokal an seinem Teilgebiet ohne jegliche Kommunikation (außer bei auftretenden Fehlermeldungen).

   Der nächste Schritt besteht im lokalen Verdichten der Informationen über die Kanten (COMPREDG), gesetzten Randbedingungen (COMPRBOUND) und Knoten (CHNGNODPOS), so daß jeder Prozessor nur noch die für ihn wesentlichen Informationen des Grobnetzes für sein Teilgebiet besitzt.

5 Anwendungsbeispiele

Das Programm NETGEN69 kann entweder als stand-alone-Version oder als Startmodul eines kompletten parallelen Finite-Elemente-Programmes eingesetzt werden.
Im ersten Fall wird nur die parallele Netzgenerierung einschließlich der grafischen Ausgabe des Netzes realisiert (siehe Abschnitt 3).
Im zweiten und allgemeinen Fall ist der parallele Netzgenerator als Schnittstelle in ein Finite-Elemente-Programm integriert. Nachfolgend werden zwei Beispiele beschrieben, die mit dem parallelen Strömungsmechanik-Paket „SPC-PM CFD“ [4] gerechnet wurden. Dieses Programm dient der Simulation von instationären Strömungs- und Transportvorgängen auf MIMD-Parallelrechnern. Die Rechnungen wurden auf 16 und 32 Prozessoren vom Typ Transputer T 805 und auf dem GC Power Plus 128 durchgeführt.

5.1 Ein - an Teilen des Randes erwärmter - gekrümmter Kanal

Dieses Beispiel steht für eine Dreiecks-Grobvernetzung. Daraus wird ein Feinnetz aus 6-Knoten-Dreiecken generiert.
Es handelt sich um einen gekrümmten ebenen Kanal, der mit folgenden Randbedingungen versehen ist:

- Randbedingungen 1. Art für die Geschwindigkeit an allen Rändern außer dem Ein- und Ausströmrand

$$u_x = u_y = 0$$

- natürliche Randbedingungen am Ein- und Ausströmrand (keine Vorgabe von Dirichlet-Randbedingungen).

- Randbedingungen 1. Art für die Temperatur an den Außenrändern des Bogens

$$T = 500K$$

Da im Inneren des Kanals eine konstante Temperatur von 470 K vorgegeben ist, ergibt sich an den Außenrändern des Bogens eine Temperaturdifferenz von 30 K.

Die beiden abschließenden Vektoren beschreiben die Materialparameter für das zu lösende Problem. Sie existieren nur in der Version 1 .

h) **Mate(k,j) : Vektor der Materialdaten** ,

wobei j=1,...,NMat; k=Nmax+1 (fest); dabei bedeuten:

Mate(1,j) : die Anzahl der Materialwerte des Materialbereiches j
Mate(2,j) : der erste Materialwert des Materialbereiches j
.....
Mate(k,j) : der letzte Materialwert des Materialbereiches j

Bemerkungen:
• In Mate(k,j) steht der letzte Materialwert nur dann, wenn der Materialbereich j wirklich die volle Anzahl Nmax Elemente hat. Anderenfalls sind nur soviele Materialwerte relevant, wie in Mate(1,j) angegeben sind. Diese stehen dann ab Mate(2,j), die restlichen Nmax−Mate(1,j) Feldelemente sind ohne Bedeutung.
• Das Feld der Materialdaten ist ein globales Feld, das vom Netzgenerator nicht verändert wird und auf jedem Prozessor zur Auswahl der Materialdaten für die jeweiligen lokalen Elemente zur Verfügung steht.

i) **Elmat(j) : Vektor der Elementmaterialnummern** ,

wobei j=1,...,NEL;

Bemerkung:
• Jedes generierte Element erhält die Materialbereichsnummer seines Vaterelementes zugeordnet. Damit lassen sich die zugehörigen Materialparameter aus dem Feld **Mate** ablesen.

Für die Realisierung der zur Lösung der Gleichungssysteme auf den entstehenden Netzen vereinbarten Datenstrukturen werden in MARKEDG verschiedene Größen berechnet. Dabei wird davon ausgegangen, daß nach Ablauf der Netzgenerierung lokal alle Knoten in der Reihenfolge Koppelknoten, Randknoten, innere Knoten numeriert worden sind.

Anschließend werden die noch fehlenden Kantenmittelknoten des Grobteilnetzes generiert (GEN3NOD6 / GEN3NOD9). Hierzu muß gesagt werden, daß der hierarchische Netzgenerator in jedem Verfeinerungslevel ein Netz aus 6-Knoten-Dreiecken / 9-Knoten-Vierecken liefert. Daher wird dies auch im Ausgangszustand (Level 0) realisiert. Kanten, die bei der Eingabe bereits drei Knoten besitzen, bleiben (bis auf die Vergabe einer neuen lokalen Nummer der drei Knoten) natürlich unverändert, während Kanten, die nach der Eingabe nur zwei Knoten aufweisen, einen Mittelknoten als arithmetisches Mittel von Anfangs- und Endknoten der Kante erhalten.

Vor der Netzgenerierung können aus den bisher bekannten Eingabeinformationen die entstehenden Gesamtanzahlen für Knoten, Kanten und Elemente genau vorausberechnet werden (PREDNEW6 / PREDNEW9) .

Im Sinne einer minimalen Speicherplatzvergabe werden mit diesen Gesamtanzahlen die Startpointer für alle bei der Netzgenerierung zu verwendenden Felder bestimmt.

Sollte der verfügbare Speicherplatz zur Generierung eines Levels-Feinnetzes nicht ausreichen, kehrt das Programm zur Eingabe der Anzahl der gewünschten Verfeinerungslevel zurück.

c) Dieser Programmteil wird mit dem Start der Zeitmessung für die Generierung des Feinnetzes eingeleitet (INITTIME).
Im Generierungsteil treten erstmals größere Abweichungen bei der Unterteilung der Dreiecke (HDOMAIN6) beziehungsweise Vierecke (HDOMAIN9) auf, so daß die beiden Teile einzeln erläutert werden.

Wir beginnen mit der **Verfeinerung der Dreiecksnetze(HDOMAIN6):**

c1a) Vorbereitende Subroutinen dienen
- zur Markierung der Koppelkanten(MARKCOUPED),
- zur Markierung der Randkanten, die Dirichlet- und/oder Neumann-Randbedingungen tragen (MARKBOUNED),
- zur Markierung der inneren Kanten (MARKINED) und
- zum Aufbau des Feldes der globalen Knotennummern für die Crosspoints (CROSSP).

c2a) Die Subroutine NETZHI6 besteht im wesentlichen aus einer Schleifenanweisung, bei der die hierarchische Verfeinerung der 6-Knoten-Dreiecke in Levels Stufen durchgeführt wird. Somit wird die Kernroutine (HIESTEP6) für jedes Level einmal durchlaufen.

Der Ablauf eines Verfeinerungsschrittes läßt sich relativ anschaulich in folgender übersicht darstellen:

```
DO J über alle NF Dreiecke:
    DO K über alle 3 Kanten:
        IF (Kante noch nicht geteilt) THEN
          - Generieren der beiden neuen Kanten (GENEDGE)
          - Generieren der neuen Mittelknoten auf den neuen Kanten als
            Gerade, Kreisbogen oder Parabelstück (GENNODE6)
          - Vergabe eines globalen Namens für eine Koppelkante
        ENDIF
    ENDDO(K)
    Generieren der 3 inneren Kanten und Knoten (GENINEN6)
    Generieren der 4 neuen Flächen aus den Kanten (SETFACE6/SETFACE3)
ENDDO(J)
```

c3a) Nach der eigentlichen Generierung des verfeinerten 6-Knoten-Dreiecksnetzes werden jetzt die Felder belegt, die anschließend für die parallele FEM gebraucht werden. Es handelt sich dabei um
- den Aufbau der Liste der globalen Knotennummern und der Liste der Ketteninformationen (CNACHAIN)
- das Verdichten des Knotenfeldes auf lokal verwendete Knoten (REDUNOD6)
- den Aufbau der hierarchischen Knotenliste (LISTHI)
- den Aufbau der Elementliste für die 6-Knoten-Dreiecke (LIELEM)
- die Generierung der Dirichlet- und Neumann-Randbedingungen für das generierte Netz (BCDIR und BCNEU) .

Bei der **Verfeinerung der Viereckснetze (HDOMAIN9)** treten im Vergleich zu den für HDOMAIN6 beschriebenen Algorithmen folgende Modifizierungen auf:

c1b) Nach der Markierung der Kanten (MARKCOUPED, MARKBOUNED, MARKINED) wird eine Subroutine eingefügt, die für alle Flächen des Startlevels 0 einen 9. Knoten als Mittelknoten des Vierecks definiert (NOD9LEV0). Daran schließt sich wieder die Belegung der globalen Knotennummern für die Crosspoints an (CROSSP).

c2b) Innerhalb der Subroutine NETZHI9 erfolgt die hierarchische Verfeinerung der 9-Knoten-Vierecke, wobei die Kernroutine (HIESTEP9) für jedes Verfeinerungslevel einmal durchlaufen wird.

Welche Variante zur Berechnung des Mittelknotens gewählt wird, hängt von der Anzahl der Kreisbogenkanten (kkreis) im Viereck in folgender Weise ab:

kkreis	Variante	Bemerkung
0	Node9iso	
1	Node9v3	
2	Node9v3	Kreisbogenkanten gegenüberliegend
	Node9v2	Kreisbogenkanten anliegend
3	Node9v3	
4	Node9v2	bei vollständigem Kreis alle Varianten gleichberechtigt

f) **Dir(j) : Vektor der Dirichlet-Randbedingungs-Daten** ,
wobei j=1,...,NDIR*(3*NFHG+4);

Bemerkung:
• Die Daten für die Kanten mit Dirichlet-Randbedingungen sind in einem eindimensionalen Feld fortlaufend gespeichert. Für jede der generierten NDIR Teilkanten sind folgende Daten enthalten:

Inhalt		vom Typ
- Startknoten der Kante		(INTEGER)
- Mittelknoten der Kante		(INTEGER)
- Endknoten der Kante		(INTEGER)
- Randbedingungscode		(INTEGER)
- Rb-Wert des Startknotens	Freiheitsgrad 1	(REAL)
- Rb-Wert des Mittelknotens	Freiheitsgrad 1	(REAL)
- Rb-Wert des Endknotens	Freiheitsgrad 1	(REAL)
.....	.....	.....
- Rb-Wert des Startknotens	Freiheitsgrad NFHG	(REAL)
- Rb-Wert des Mittelknotens	Freiheitsgrad NFHG	(REAL)
- Rb-Wert des Endknotens	Freiheitsgrad NFHG	(REAL)

Der Randbedingungscode ist die übertragung des im Eingabefile beschriebenen Randbedingungscodes auf die generierten Teilkanten.

g) **Neu(3,j) : Vektor der Neumann-Randbedingungs-Daten** ,
wobei j=1,...,NNEUM*(3*NFHG+4);

Bemerkung:
• Die Daten für die Kanten mit Neumann-Randbedingungen sind in der gleichen Weise wie die Dirichlet-Randbedingungen gespeichert. Für jede der generierten NNEUM Teilkanten gelten daher analog die Erklärungen des vorangegangenen Absatzes f) .

e) **Elem(KEL,j) : Vektor des Elementzusammenhangs** ,
wobei KEL=6 (für Dreiecke) oder KEL=9 (für Vierecke)
und j=1,...,NEL; dabei bedeuten:

bei Dreiecken :

Elem(1,j) : erster Eckknoten des Dreiecks j
Elem(2,j) : zweiter Eckknoten des Dreiecks j
Elem(3,j) : dritter Eckknoten des Dreiecks j
Elem(4,j) : erster Kantenmittelknoten des Dreiecks j
Elem(5,j) : zweiter Kantenmittelknoten des Dreiecks j
Elem(6,j) : dritter Kantenmittelknoten des Dreiecks j

bei Vierecken :

Elem(1,j) : erster Eckknoten des Vierecks j
Elem(2,j) : zweiter Eckknoten des Vierecks j
Elem(3,j) : dritter Eckknoten des Vierecks j
Elem(4,j) : vierter Eckknoten des Vierecks j
Elem(5,j) : erster Kantenmittelknoten des Vierecks j
Elem(6,j) : zweiter Kantenmittelknoten des Vierecks j
Elem(7,j) : dritter Kantenmittelknoten des Vierecks j
Elem(8,j) : vierter Kantenmittelknoten des Vierecks j
Elem(9,j) : Elementmittelknoten des Vierecks j

Bemerkungen:
• Aus dem Feld **Face** (Darstellung der Elemente durch Kanten) wird das Feld **Elem** (Darstellung der Elemente durch Knoten) gebildet, um die für die FEM übliche Datenstruktur für die Geometriebeschreibung (Knoten und Elemente) zu erhalten.
• Eine Besonderheit bildet bei Vierecken die Generierung des Elementmittelknotens. Zunächst wird für alle Flächen im Startlevel 0 vor der eigentlichen Generierung ein 9. Knoten als Mittelknoten des Vierecks definiert. Im Verlauf der Netzgenerierung erhalten alle neu entstehenden Elemente ebenfalls einen Mittelknoten, der wie folgt berechnet wird:
Aus der Vielzahl der Möglichkeiten bezüglich der Kantenform (geradlinig oder gekrümmt) wurden einige ausgewählt, die in den bisherigen Anwendungen am häufigsten aufgetreten sind. Dabei wurden folgende Varianten zur Berechnung des Mittelknotens realisiert (weitere Varianten sind möglich):

a) Node9iso : Berechnung aus 8 Randknoten nach isoparametrischem Konzept
b) Node9v1 : Berechnung als arithmetisches Mittel der 8 Randknoten
c) Node9v2 : Berechnung als Schnittpunkt der Geraden, die von den Mittelpunkten der beiden gegenüberliegenden Kanten gebildet werden
d) Node9v3 : Berechnung als arithmetisches Mittel der 2 Mittelknoten von gegenüberliegenden Kreisbogenkanten

Der Ablauf eines Verfeinerungsschrittes läßt sich nun wie folgt darstellen:

```
DO J über alle NF Vierecke:
   DO K über alle 4 Kanten:
      IF (Kante noch nicht geteilt) THEN
        - Generieren der beiden neuen Kanten (GENEDGE)
        - Generieren der neuen Mittelknoten auf den neuen Kanten als
          Gerade, Kreisbogen oder Parabelstück (GENNODE9)
        - Vergabe eines globalen Namens für eine Koppelkante
      ENDIF
   ENDDO(K)
   Generieren der 4 inneren Kanten und Knoten (GRADINTEDG und
    GENINEN9)
   Generieren der 4 neuen Flächen aus den Kanten (CORSEQ, CORMAT
    und SETFACE4) einschließlich der Belegung der hierarchischen
    Knotenliste und der Elementliste für die 9-Knoten-Vierecke
ENDDO(J)
```

c3b) Nach der eigentlichen Generierung des verfeinerten 9-Knoten-Viereckesnetzes werden jetzt noch die zusätzlich benötigten Felder belegt.
Das betrifft im einzelnen:
- die Liste der globalen Knotennummern und die Liste der Ketteninformationen (CNACHAIN)
- das Verdichten des Knotenfeldes auf lokal verwendete Knoten (REDUNOD9)
- die Generierung der Dirichlet- und Neumann-Randbedingungen für das generierte Netz (BCDIR und BCNEU) .

Nun sind alle Output-Felder mit ihren richtigen Daten belegt und die Längen aller Felder exakt bekannt. Somit können jetzt noch vorhandene Lücken im angeforderten Speicherbereich beseitigt werden. Alle Felder werden auf ihren minimalen Speicherplatzbedarf verdichtet und ab Speicheradressse 1 hintereinander abgelegt. Somit bleibt für die Behandlung der Aufgabe (Assemblierung der Matrizen und rechten Seiten, Lösung entstehender Gleichungssysteme, grafische Auswertung der Ergebnisse, ...) das Maximum an verfügbarem Speicherplatz erhalten.
Der Programmteil endet mit der Messung und Ausgabe der Zeit für die Generierung des Feinnetzes (GETTIMES).

d) Nach der Fertigstellung des Netzes und der damit verbundenen Datenfelder wird der Nutzer interaktiv gefragt, ob er eine grafische Darstellung des Netzes wünscht (GRAFIK2D). Bei positiver Antwort erfolgt die grafische Ausgabe im dafür vorgesehenen Fenster, anderenfalls kehrt das Programm zur Eingabe der Anzahl der gewünschten Verfeinerungslevel zurück.
Die Benutzung der grafischen Ausgabetools von GRAFIK2D ist in [10] dokumentiert.

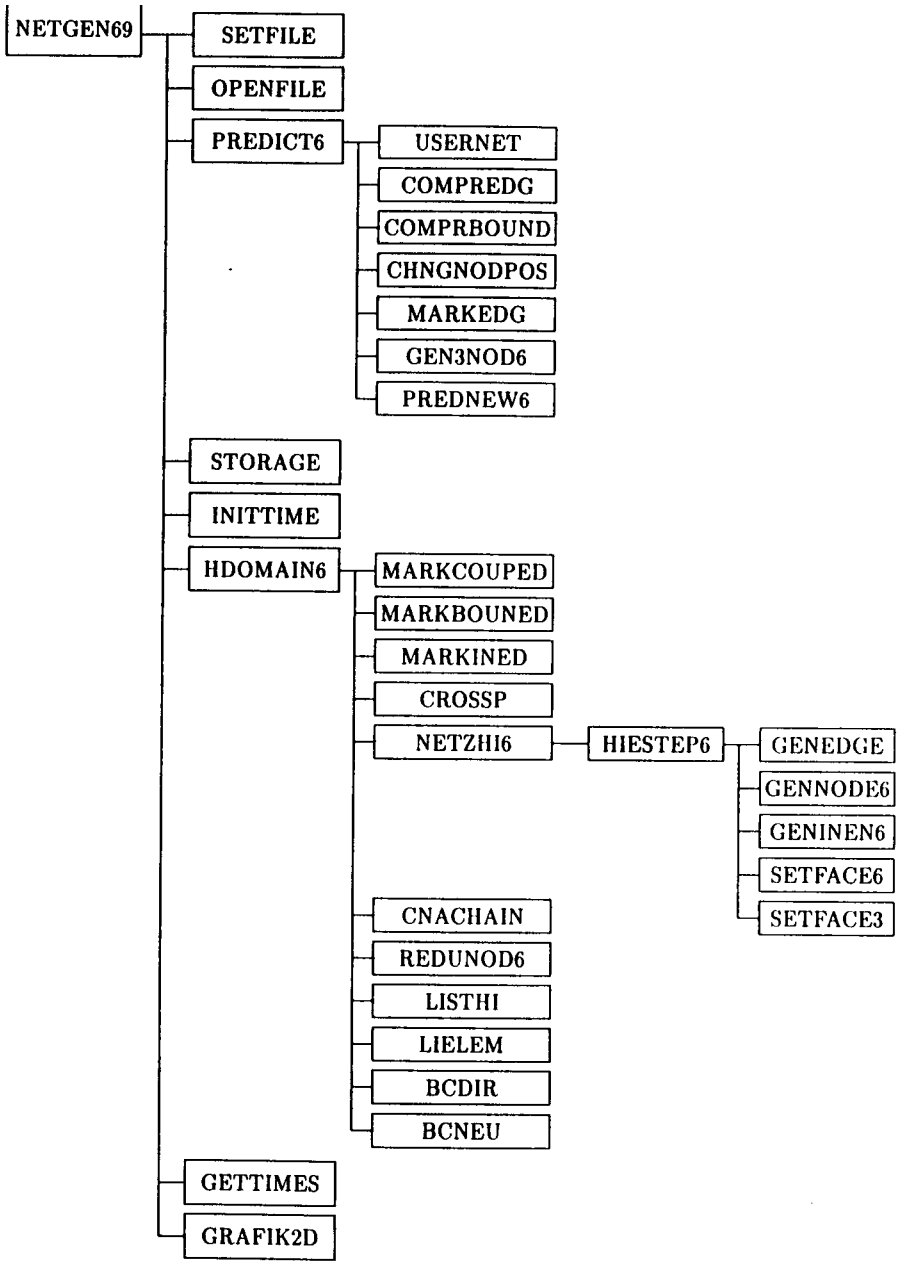


Abbildung 1 : Die Subroutinen des Dreiecks-Netzgenerators

c) **Iglob(j) : Vektor der globalen Knotennamen** ,

wobei j=1,...,NC;

Bemerkung:

• Der Vektor enthält die globalen Namen für alle NC Knoten, d.h. für die Crosspoint-Knoten und Koppelknoten sowie für die Knoten, die durch Generierung aus den Randkanten und inneren Kanten des Grobnetzes entstanden sind.
Zur späteren Kommunikation mit den Prozessoren, die an das jeweilige lokale Gebiet angrenzen, werden aber nur die globalen Namen der Crosspoint-Knoten und der echten Koppelknoten benötigt.
Ein spezieller Algorithmus erzeugt eine eindeutige Zuordnung der globalen Namen für alle Crosspoint-Knoten und Koppelknoten und sichert, daß auf jedem Prozessor für denselben Knoten einer Koppelkante derselbe globale Name generiert wird.

d) **Kette(5,j) : Vektor der Ketteninformationen** ,

wobei j=1,...,NANZK; dabei bedeuten:

Kette(1,j)	:	lokale Nummer des Startknotens der Kette
Kette(2,j)	:	Länge der Kette
Kette(3,j)	:	globaler Name des ersten Crosspoints
Kette(4,j)	:	globaler Name des zweiten Crosspoints
Kette(5,j)	:	Platzhalter für die Inter-Prozessor-Kommunikation

Bemerkungen:

• Mit der Länge der Kette wird die Anzahl der Knoten zwischen den beiden Crosspoint-Knoten bezeichnet.

• Die globalen Namen der Crosspoint-Knoten beziehen sich auf das Feld **Iglob**. Der erste Crosspoint entspricht dem Startknoten, der zweite Crosspoint entspricht dem Endknoten der Kante.

• Auf der Position Kette(5,j) steht nach der Netzgenerierung eine „ 0 “ als Platzhalter. Diese Position wird später neu belegt mit einer Information über den Zielprozessor, über den die Kommunikation bezüglich dieser Kette erfolgt. Dabei gilt:

Kette(5,j)= ± (Nummer des Zielprozessors + 1)

Das Vorzeichen gibt dabei an, ob der Prozessor seine Information zuerst sendet oder zuerst die Information des zugehörigen Prozessors empfängt.

Bemerkungen:

• Die in **Hilis** eingetragenen Knoten sind levelweise in aufsteigender Folge geordnet. Die zum Startlevel gehörenden Crosspoint-Knoten haben natürlich keine Väter, so daß deren Positionen mit "0"gekennzeichnet werden. Alle weiteren Knoten besitzen zwei Väter, die sich aus der Kantenhierarchie bestimmen.

• Bei der hierarchischen Verfeinerung entstehen aus einer Vaterkante immer zwei Sohnkanten. Auf solchen Kanten sind die Väter der neu generierten Knoten also natürlich definiert. Die Väter der neuen Knoten auf inneren Kanten ergeben sich aus der Generierungsvorschrift, die bei Dreiecken und Vierecken differiert.

• Das Teilverhältnis ist eine reelle Zahl zwischen 0 und 1 . Es wird für die Grobnetzknoten, die keine Crosspoint-Knoten sind, aus ihrer Lage zu den beiden Vaterknoten bestimmt. Damit besteht in einem eingeschränkten Maße die Möglichkeit, ohne echte adaptive Verfeinerung eine Netzverdichtung zu erreichen, indem bei der Eingabe des Grobnetzes Kantenmittelknoten nicht in die Mitte der Kante gesetzt werden, sondern nach einem Endpunkt der Kante hin.

Im Falle eines gleichmäßigen Grobnetzes (Mittelknoten alle auf der Kantenmitte) ergibt sich ein konstantes Teilverhältnis von 0.5 für alle generierten Knoten in jedem Verfeinerungslevel.

Bei der hierarchischen Verfeinerung wird das Teilverhältnis wie folgt gebildet:

* NETGEN69 verwendet stets Kanten mit drei Knoten. Enthält das Grobnetz auch Kanten mit nur zwei Knoten, so werden diese im Vorbereitungsteil (GEN3NOD6 / GEN3NOD9) erweitert (siehe Abschnitt 3). Wir gehen somit davon aus, daß im Verlauf der Netzgenerierung jede Kante drei Knoten besitzt.
* Für die Kanten des Grobnetzes wird jetzt das Verhältnis des Abstandes des Endknotens vom Mittelknoten zum Abstand des Startknotens vom Mittelknoten berechnet.
* Durch Bilden der Quadratwurzel aus dem berechneten Verhältnis ergibt sich das für die weitere Generierung benutzte Teilverhältnis (Graduierungsfaktor), das in jedem Verfeinerungslevel eine sich gleichartig fortsetzende Graduierung sichert.
* Diese Bildungsvorschrift wird für die Kanten verwendet, die bei der Verfeinerung unmittelbar durch Teilung der entsprechenden Vaterkante entstehen.
* Für die inneren Kanten, die durch das Verbinden der Kantenmittelknoten der Vaterkanten gebildet werden, ist folgende Regel realisiert worden:

- für 6-Knoten-Dreiecke:
  Die der äußeren Vaterkante gegenüberliegende Sohn-Innenkante erhält deren Graduierungsfaktor.
- für 9-Knoten-Vierecke:
  Die innere Kante, die von dem Kantenmittelknoten der äußeren Vaterkante zum Mittelknoten des Vierecks verläuft, erhält den Graduierungsfaktor der beiden anliegenden äußeren Vaterkanten. Dazu muß gesagt werden, daß die Vererbung des Graduierungsfaktors bei Vierecken nur bei gleicher Unterteilung gegenüberliegender Kanten sinnvoll ist.

* Das Teilverhältnis wird bei der Lösung der Gleichungssysteme für die Berechnung der hierarchischen Vorkonditionierungsmatrix verwendet.

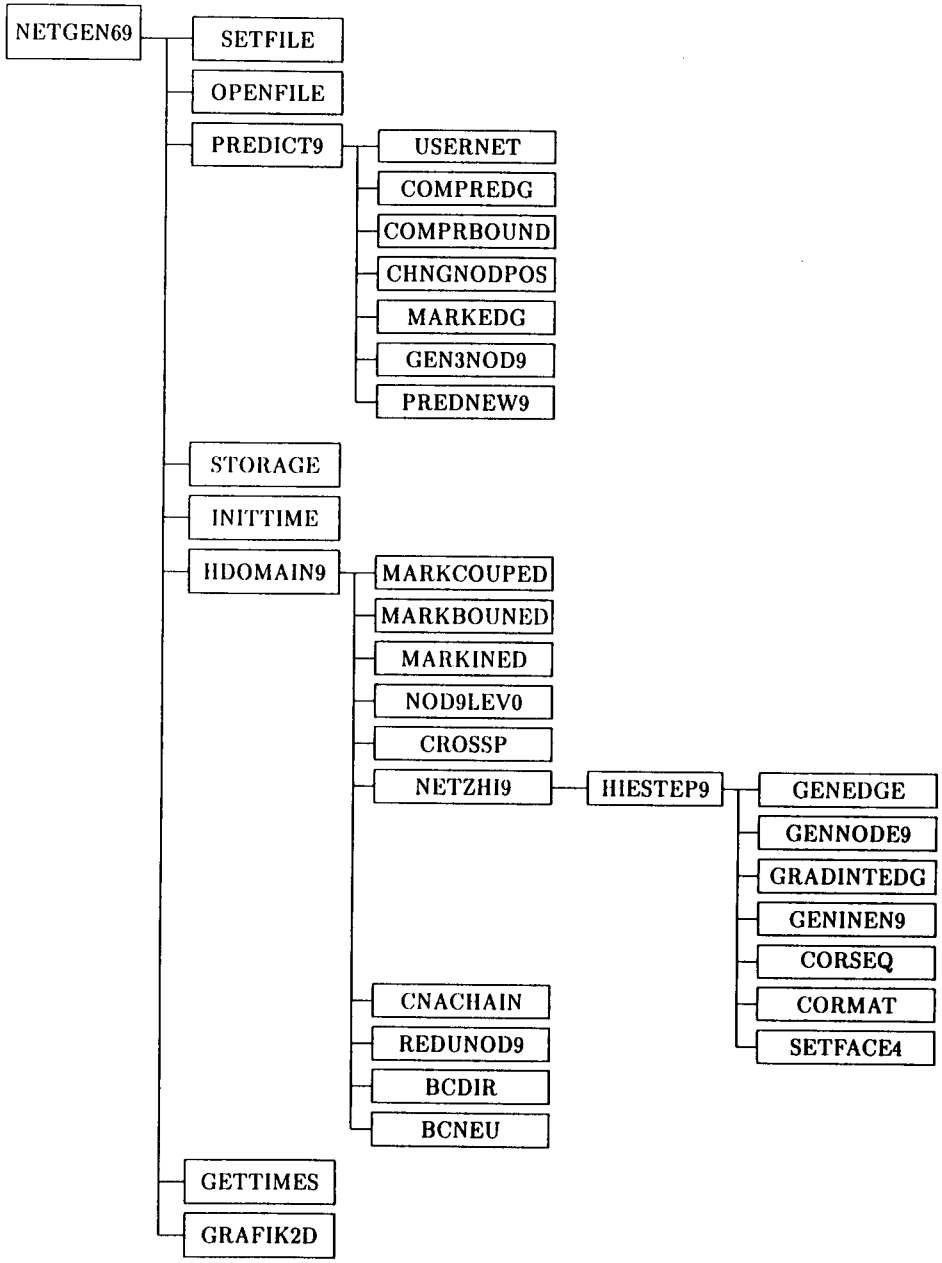


Abbildung 2 : Die Subroutinen des Vierecks-Netzgenerators

4 Die Struktur der Ausgabedaten

Der Netzgenerator NETGEN69 generiert vollständig parallel die lokalen Netze des gewünschten Verfeinerungslevels, die zu den aus dem Grobnetz aufgeteilten Teilgebieten eines jeden Prozessors gehören. Daraus ergibt sich, daß alle Output-Felder lokale Daten enthalten, was für die weitere parallele Rechnung sehr günstig ist. Eine kleine Ausnahme bildet das Feld der Materialdaten **Mate** , das global und für alle Prozessoren konstant ist.
Die Ausgabedatenstruktur entspricht den Festlegungen innerhalb der DFG-Forschergruppe [6] und ist somit universell einsetzbar.

4.1 Skalare Größen

Alle im folgenden aufgelisteten Größen sind auf den einzelnen Prozessor bezogene lokale skalare Größen mit Ausnahme der mit einem (*) gekennzeichneten, die global für alle Prozessoren Gültigkeit besitzen.

NK	:	Gesamtanzahl aller Knoten
NC	:	Anzahl aller "Koppelknoten"
NANZK	:	Anzahl aller Grobnetzkanten
NEL	:	Anzahl aller finiten Elemente (Dreiecke / Vierecke)
*KEL	:	Anzahl der Knoten pro Element (6 / 9)
*NFHG	:	Anzahl der Problemfreiheitsgrade
NDIR	:	Anzahl der generierten Kanten mit Dirichlet-Randbedingung
NNEUM	:	Anzahl der generierten Kanten mit Neumann-Randbedingung
*NDIMMAT	:	Größe des globalen Feldes der Materialdaten

Bemerkungen:
• NANZK beschreibt die Anzahl der lokalen Ketten. Da der bei der Problemlösung verwendete Grobgitterlöser der Einfachheit halber auf das eingangs definierte Grobnetz zurückgreift, wurden neben den Koppelkanten auch die Randkanten und die inneren Kanten in das Feld der Ketten aufgenommen. Somit spiegelt NANZK die lokale Anzahl aller Grobnetzkanten auf dem Prozessor wider.
• Daraus folgt, daß NC nicht nur die Anzahl der echten Koppelknoten (Crosspoints und Knoten der Koppelkanten) repräsentiert, sondern außerdem noch die Anzahl der Knoten der Randkanten und der inneren Kanten enthält (im Sinne der Definition der Ketten). Damit unterscheiden sich NC und NK nur noch um die Anzahl der echt inneren Knoten.
• Im feinsten Level der Verfeinerung beträgt die Anzahl der generierten Kanten, die entsprechende Randbedingungen 1. oder 2. Art tragen,

a) $NDIR = NDIR_{old} * 2^{Levels}$ beziehungsweise
b) $NNEUM = NNEUM_{old} * 2^{Levels}$.

• Für die Größe des Materialdatenfeldes gilt:

NDIMMAT=NMat*(Nmax+1) .

4.2 Vektorielle Größen

Der Netzgenerator NETGEN69 liefert 14 verschiedene Felder zur Beschreibung des Feinnetzes und seiner hierarchischen Struktur, der Randbedingungen, der Materialparameter und der Kommunikationsstrukturen für die parallele Realisierung des Programms. Davon werden am Ende des Generierungsabschnittes 5 Felder durch die dynamische Speicherplatzverwaltung überschrieben, die für den weiteren Assemblierungs- und Lösungsprozeß nicht benötigt werden (für abweichende Aufgabenstellungen ist eine änderung natürlich möglich). Dabei handelt es sich um:
- das Feld der lokalen Kanteninformationen (alle Hierarchielevel gespeichert)
- das Feld der lokalen Flächendaten (nur feinstes Level gespeichert)
- das Feld der globalen Koppelkantennamen
- das Feld der Inputdaten der Dirichlet-Randkanten
- das Feld der Inputdaten der Neumann-Randkanten.

Somit werden 9 Felder an der Schnittstelle Netzgenerator / FE-Programm bereitgestellt, die nachfolgend beschrieben werden:

a) **Node(3,j) : Vektor der Knotenkoordinaten** ,
wobei j=1,...,NK; dabei bedeuten:

Node(1,j)	:	x-Koordinate des Knotens j
Node(2,j)	:	y-Koordinate des Knotens j
Node(3,j)	:	Knotenanzeiger (für grafische Darstellung und Strömungssimulation)

Bemerkung:
• Für die weitere Benutzung des Knotenfeldes spielt die Knotenanordnung eine wichtige Rolle. Die Reihenfolge der lokalen Knotennummern ist wie folgt organisiert:
- 1. Crosspoint-Knoten
- 2. echte Koppelknoten
- 3. auf Randkanten (bezüglich des Grobnetzes) liegende Knoten
- 4. auf inneren Kanten (bezüglich des Grobnetzes) liegende Knoten
- 5. echte innere Knoten

Damit lassen sich auf jedem Prozessor unter Verwendung des COMMON-Blockes /Problem/ die Startpositionen der Crosspoint-Knoten, der in den Ketten vereinigten Knoten und der inneren Knoten sofort finden.

b) **Hilis(4,j) : Vektor der hierarchischen Knotenliste** ,
wobei j=1,...,NK; dabei bedeuten:

Hilis(1,j)	:	lokale Knotennummer k
Hilis(2,j)	:	"Vater-1-Knoten"von k
Hilis(3,j)	:	"Vater-2-Knoten"von k
Hilis(4,j)	:	Teilverhältnis (Lage des Knotens k bezüglich der Vaterkante)