Parallele Netzgenerierung

Dr.-Ing. L. Lämmer, Dipl.-Ing. M. Burghardt, Prof. Dr.-Ing. U. Meißner

1 Vorgehensweise

1.1 Grundgedanke der Methode

Die Generierung von FE-Netzen auf dem Parallelrechner weist gegenüber der sequentiellen Generierung einige zusätzliche Schwierigkeiten auf. Es muß sichergestellt werden, daß nicht zwei Prozessoren zur gleichen Zeit an der gleichen Stelle Elemente generieren. Die Teilnetze der einzelnen Prozessoren müssen an den Nahtstellen zusammenpassen. Die Rechenlast für alle Prozessoren sollte möglichst gleich sein.

Die hier beschriebene Methode weist jedem Prozessor einen Teil des Gesamtgebietes zu und generiert dann auf jedem Prozessor ein Teilnetz. Durch das Aufteilen des Gebietes wird verhindert, daß Elemente an derselben Stelle generiert werden. Auf den Teilgebietsrändern werden die Knoten durch eine eindeutige Zuordnung vorgegeben. Damit wird realisiert, daß das Netz über die Teilgebietsgrenzen hinweg konform ist. Nachdem die Knoten auf den Gebietsrändern festgelegt wurden, ist eine Netzaufteilung ohne Austausch auf den einzelnen Prozessoren möglich. Ein sequentieller Netzgenerierungsalgorithmus kann nun die Teilnetze auf den Prozessoren erzeugen.

1.2 Eingabedaten

Das Verfahren ist so entwickelt worden, daß als Eingabedaten nur geringfügig modifizierte Ausgabedaten von CAD-Programmen dienen. Als Beschreibung der Geometrie dienen Punkte und Polygonzüge. Punkte können sich hierbei in den Eckpunkten oder im Inneren der Polygone befinden. In einzelnen Punkten wird die angestrebte Kantenlänge für die Elemente in der Umgebung des Punktes vorgegeben.

1.3 Schwerachsenmethode

Das Gebietes wird mit der *Schwerachsenmethode* [2] geteilt. Diese halbiert eine Geometrie entlang ihrer stärkeren Hauptachse in zwei, etwa gleich große Teilgebiete. Die Qualität der Teilung ist problemabhängig. Diese Schritte laufen währende des Programmablaufes parallel auf allen Prozessoren ab, wobei jeder Prozessor am Ende sein eigenes Teilgebiet berechnet. Für die Gebietszerlegung ist daher keine Kommunikation notwendig.

Das Vorgehen ist in jedem einzelnen Rekursionsschritt wie folgt. Zuerst wird der Schwerpunkt der Geometrie mit den Koordinaten x_s und y_s bestimmt und eine Koordinatentransformation des Ursprungs in den Schwerpunkt vorgenommen. Mit den Flächenträgheitsmomenten I_x, I_y sowie dem Deviationsmoment I_{xy} wird der Winkel der Hauptachse zur Koordinatenachse mit $tan2\alpha = \frac{2I_{xy}}{I_x - I_y}$ bestimmen. Die Geometrie wird nun entlang der Hauptachse mit dem größeren Flächenträgheitsmoment geteilt.

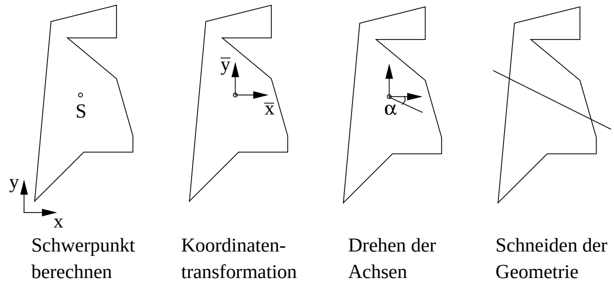


Abbildung 1: *Rekursives Teilen mit Schwerachsenmethode*

1.4 Lastverteilung

Um auf allen Prozessoren eine gleiche Rechenlast zu erreichen, wird eine Lastfunktion für die Geometrie eingeführt. Als brauchbare Ansätze haben sich Funktionen wie $f(\bar{x},\bar{y}) = \frac{1}{w^2}$ oder $f(\bar{x},\bar{y}) = (\bar{w} - w)^2$ erwiesen, wobei w die angestrebte Kantenlänge und $\bar{w}$ der Mittelwert oder das Maximum der angestrebten Kantenlänge ist. Diese Lastfunktion wird berücksichtigt bei der Berechnung des Schwepunktes bzw. der Trägheitsmomente und bewirkt eine Verschiebung des Schwerpunktes des Gebietes in Richtung der kleineren Elemente.

$$x_s = \frac{\int x \cdot f(x,y)dA}{\int f(x,y)dA} \;\; ; \;\; y_s = \frac{\int y \cdot f(x,y)dA}{\int f(x,y)dA} \tag{1}$$

Die Trägheitsmomente bzw. Schwerachsen berechnen sich nach der Koordinatentransformation dann wie folgt:

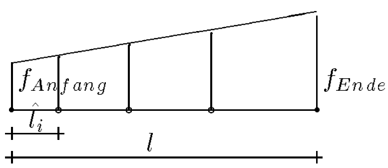
$$I_{\bar{x}} = \int \bar{y}^2 \cdot f(\bar{x},\bar{y})dA \;\; ; \;\; I_{\bar{y}} = \int \bar{x}^2 \cdot f(\bar{x},\bar{y})dA \tag{2}$$

$$I_{\bar{x}\bar{y}} = -\int \bar{x} \cdot \bar{y} \cdot f(\bar{x},\bar{y})dA \tag{3}$$

Die Lösung der Integrale wird mit Hilfe der Gaußintegration durchgeführt. Hierfür ist eine Zerlegung der gesamten Geometrie in grobe Dreiecke notwendig. Diese Triangulierung verwendet nur die vorher schon vorhandenen Knoten und erzeugt keine neuen.

1.5 Erzeugen der Knoten auf den Gebietsgrenzen

Um Kommunikationen während des Generierens des Netzes auf den einzelnen Prozessoren zu vermeiden, werden die Knoten auf den Gebietskanten eindeutig durch eine Beschreibung vorgegeben. Über die angestrebte Kantenlänge in den Endknoten eines Polygonzugabschnittes werden die Zwischenknoten linear interpoliert. Hierbei muß die Kantenlänge in den neuen Knoten, als auch die Stellen, an denen die neuen Knoten auf der Geraden liegen, über eine Geradengleichung bestimmt werden.



Die einzelnen Abstände zwischen den Knoten werden wie folgt berechnet:

$$\hat{l}_1 = f_{Anfang} \cdot \beta_1 \;\;;\;\; \hat{l}_i = \hat{l}_{i-1} + \sum_1^i i \cdot \frac{f_{Anfang} - f_{Ende}}{\tilde{n}} \cdot \beta_i \tag{4}$$

Hierbei ist f die Wichtungsfunktion und $\tilde{n}$ die Anzahl der Zwischenknoten. Der Anfangsknoten ist immer als der Knoten mit dem kleineren Wert der Wichtungsfunktion definiert. β_i sind die Faktoren zum strecken der Abstände, um nur ganze Abstände vorkommen.

$$n = \frac{2 \cdot l}{f_{Anfang} + f_{Ende}} \;\;;\;\; \tilde{n} = \frac{n-1}{2} \cdot n \tag{5}$$

$$\bar{l} = \sum l_i \tag{6}$$

mit:

$$l_1 = f_{Anfang} \;\;;\;\; l_i = l_{i-1} + \sum_1^i i \cdot \frac{f_{Anfang} - f_{Ende}}{\tilde{n}} \tag{7}$$

$$\beta_i = \frac{\sum_1^i l_i}{\bar{l}} \tag{8}$$

Hierbei ist $l = \sqrt{\Delta x^2 + \Delta y^2}$ der Abstand zwischen den Endknoten.

1.6 Sequentielle Netzgenerierung

Nachdem das Gebiet in die einzelnen Teilgebiete für die Prozessoren aufgeteilt und die Knoten auf den Gebietsrändern festgelegt sind, wird das Netz auf jedem einzelnen Prozessoren erzeugt. Hierfür ist ein sequentieller Netzgenerator [4] implementiert, der folgende Anforderungen erfüllt[4] [3]:

- Vernetzung zweidimensionaler, polygonal begrenzter Flächen,
- Generierung von Drei- und Vierecken,
- Berücksichtigung von vorgegebenen Kantenlängen an jeder Stelle, falls dies geometrisch möglich ist,
- Berücksichtigung von Fixpunkten,
- Gebiete mit Löchern und
- Optimierung der Form der resultierenden Elemente.

2 Ergebnisse

Ergebnisse der oben vorgestellten Methode müssen mit dem Ergebnis von sequentiell erzeugten Netzen verglichen werden. Hierbei muß die Netzqualität, die Qualität der Partitionierung und die parallele Effizienz betrachtet werden. An zwei verschiedenen Beispielen (siehe Abbildungen 2, 3) werden die Ergebnisse der Arbeit vorgestellt.

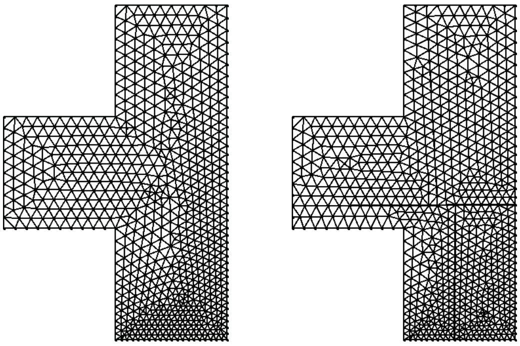


Abbildung 2: *Sequentielle and parallele (auf 4 Prozessoren) Netzgenerierung, Beispiel m1*

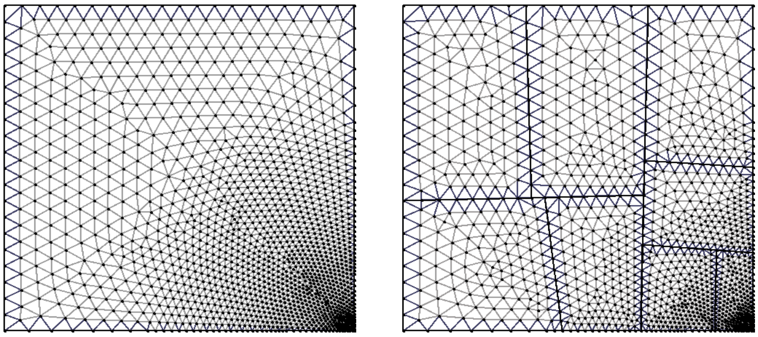


Abbildung 3: *Sequentielle and parallele (auf 8 Prozessoren) Netzgenerierung, Beispiel m3*

2.1 Qualität der Netze

Die resultierenden Netze unterscheiden sich von herkömmlich sequentiell erzeugten Netzen. Die Linien, an denen die Geometrie geschnitten wurde, sind im parallel erzeugten Netz zu erkennen. Diese Linien haben Vor- und Nachteile. Ein Nachteil ist zum Beispiel, daß die Verteilung der Elemente im Gesamtgebie unregelmäßig ist. Das Glätten des Netzes wird nur für ein Teilnetz durchgeführt um Kommunikation zu vermeiden.

Die Winkelverteilung der Elemente ist ein guter Indikator zur Beurteilung der Netzqualität. Bei paralleler Berechnung erhält man bei großen Elementzahlen eine sehr ähnliche Winkelverteilung wie bei der herkömmlichen sequentiellen Berechnung (siehe Abbildung Tabelle 1).

Tabelle 1: *Mittelwert und Standardabweichung der Winkelverteilung bei sequentieller und paralleler Netzgenerierung auf 2, 4 und 8 Prozessoren*

nProcs	1	2	4	8
$\bar{\alpha}$	59.73	59.70	59.70	59.66
σ	7.43	7.44	7.74	7.96

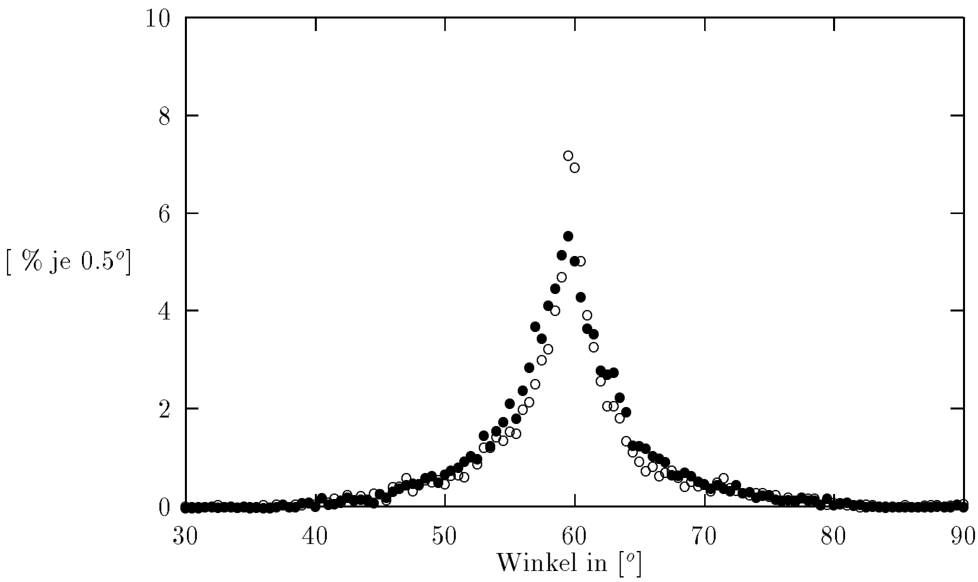


Abbildung 4: *Verteilung der Innenwinkel der Elemente mit einem • und 16 ∘ Prozessoren und einem Beispiel mit ca. 3100 Elementen*

2.2 Qualität der Partitionierung

Die Qualität der Partitionierung zeigt sich direkt in der Differenz der Elementanzahlen auf den einzelnen Prozessoren und hängt von drei Faktoren ab:

- Qualität der Eingangsdaten
- Qualität der Lastfunktion
- Anzahl der Prozessoren

Bei Verwendung schlechter Eingangsdaten kann es zu Problemen bei der Berechnung der Flächenintegrale (Gleichung 1, 2 und 3) kommen. Dies führt zu einer falschen oder ungenauen Berechnung des Schwerpunktes. Eine falsche oder ungünstige Lastfunktion bewirkt, das die Vorhersage für die Elementanzahl im Gebiet falsch ist und deshalb nur eine ungenaue Lastbalance erreicht wird. Die beiden beschriebenen Fehler erzeugen während jedem Rechenschritt einen Fehler. Der so entstehende Fehler nimmt exponentiell mit der Anzahl der Rekursionsschritte bzw. linear mit der Anzahl der Prozessoren zu.

nProcs	Beispiel m1	Beispiel m3
2	12.18%	4.12%
4	13.01%	4.56%
8	6.60%	3.45%
16	14.18%	5.83%
32	19.86%	9.37%

Tabelle 2: *Abweichung der Elementanzahlen*

Bei Berechnungen mit Testbeispielen kam es mit 32 Prozessoren (5 Rekursionsschritte) zu einer Abweichung der Elementanzahl auf einzelnen Prozessoren von bis zu 19.86% vom Mittelwert, was zu einer geringfügigen Verlängerung der Rechenzeit führt. Die Qualität der Partitionierung für die dargestellten Beispiele ist der Tabelle 2 zu entnehmen.

2.3 Zeitmessungen

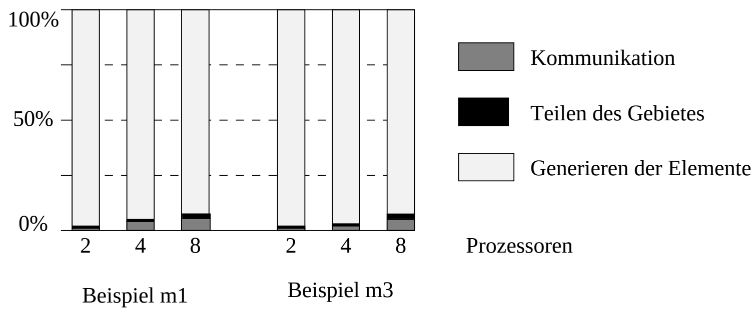


Abbildung 5: *Zeitaufwand zur Netzgenerierung, aufgeteilt nach Kommunikation, Gebitsaufteilung und Generieren der Elemente*

Der Aufwand für die Teilung der Geometrie ist bei Netzen mit mehr als 100 Elementen je Prozessor sehr klein gegenüber der Zeit für die Netzaufteilung. Der Kommunikationsaufwand der beschriebenen Methode ist sehr gering. Daher ist eine gute Effizienz zu erwarten.

Der Zeitaufwand der Netzgenerierung steigt etwa qudratisch mit der Elementanzahl. Da der beschriebene Algorithmus die Elementanzahl je Prozessor proportional zur Prozessoranzahl verringert, verringert sich der Rechenaufwand der Netzgenerierung quadratisch mit der Prozessoranzahl. Zeitmessungen haben dies an verschiedenen Beispielen nachgewiesen.

Bei steigender Prozessorenanzahl verringert sich daher der Rechenaufwand zum Generieren der Elemente sehr stark. Gleichzeitig erhöht sich der Aufwand für das Teilen der Geometrie, da mehr Rekursionsstufen benötigt werden. Wie in Abbildung 5 dargestellt, ist zeitliche Aufwand für die Kommunikation und das Teilen des Gebietes auch bei größeren Prozessorenzahlen klein gegenüber dem Aufwand zur Generierung der Elemente.

Literatur

[1] C. Farhat, *A simple and efficient automatic FEM domain decomposer*, Computers & Structures **28** (1988), Nr. 5, 579–602.

[2] C. Farhat und M. Lesoinne, *Automatic partitioning of unstructured meshes for the parallel solution of problems in computational mechanics*, International Journal for Numerical Methods in Engineering **36** (1993), 745–764.

[3] S. Kumar, *Surface triangulation: A survey*, UNC-CS TR 96-01 (1996).

[4] J. Olden, *Finite-Element-Analyse von Plattentragwerken durch adaptive Software-Techniken*, Diss., Darmstadt, Techn. Hochsch., 1996.

[5] L. Lämmer U. Meißner R. Mücke J. Olden, *Adaptive finite element analysis of plate problems for parallel computers*, Advances in Parallel and Vector Processing for Structural Mechanics (1994), 185–190.