Parallelisierung eines expliziten Finite–Volumen–Verfahrens

Siegfried Müller, Frank Knoben
Institut für Geometrie und
Praktische Mathematik
RWTH Aachen
52056 Aachen

E-Mail: mueller@igpm.rwth-aachen.de
knoben@igpm.rwth-aachen.de

Zusammenfassung

Zunächst wird ein explizites Finite–Volumen–Verfahren zur Diskretisierung von Bilanzgleichungen auf strukturierten Gittern vorgestellt. Dieses wurde für die kompressiblen Navier–Stokes–Gleichungen im chemischen Nichtgleichgewicht implementiert und am Beispiel der Umströmung stumpfer Körper, z.B. Wiedereintritt eines Raumgleiters in die Erdatmosphäre, validiert.
Auf diesem Programm aufbauend soll demonstriert werden, wie ohne Änderungen an der Programmstruktur vorzunehmen eine Parallelisierung durchgeführt werden kann. Dazu wird das bisher verwendete Gitter in mehrere Zonen unterteilt, in denen jeweils die gleichen Programmschritte parallel durchgeführt werden.
Um den hohen Aufwand pro Gitterzelle zu reduzieren, müssen möglichst viele Zellen parallel bearbeitet werden, d.h. viele Zonen mit wenigen Zellen. Dies wurde bisher allerdings nur auf einem Workstation–Cluster von 4 Maschinen (IRIS–Indigo–R4000) realisiert. Hierbei wurde das an der Fakultät für Mathematik der TU Chemnitz–Zwickau entwickelte Software–Paket **PARLIB** sowie die am Oak Ridge National Laboratory erstellte Bibliothek **PVM 3** zum Aufbau einer parallelen virtuellen Maschine eines heterogenen Netzes benutzt.
Es wird hier insbesondere die Verwendung der unterschiedlichen Strategien des Datentransfers und den damit verbundenen Transferzeiten diskutiert. Daraus resultieren einige Grundregeln, die beachtet werden müssen, um einen möglichst optimalen Speedup–Wert zu erzielen.

Key Words: parallel computation, domain decomposition, finite volume methods, Navier–Stokes equations, chemically reacting flows

AMS subject classification: 65Y05, 65M55, 76M25, 35Q30, 80A32

Inhaltsverzeichnis

1 Einleitung

Im Rahmen des Sonderforschungsbereichs 253 wurde im Teilprojekt B5 ein FORTRAN 77-Programm entwickelt, das die numerische Simulation von kompressiblen Strömungen ermöglicht. Dieses wurde dazu verwendet, um Strömungen um einen stumpfen Körper bei einer Überschallanströmung ($Ma_\infty \gg 1$) zu untersuchen. Diese sind an räumlich zweidimensionalen Geometrien beziehungsweise an rotationssymmetrischen Körpern (z.B. Kugel mit Radius r) durchgeführt worden. Die Ergebnisse sind in [10], [11] dargestellt.
Die Grundlage des Programms bildet die Diskretisierung der Navier-Stokes-Gleichungen (NS-Gen), bestehend aus den Komponentengleichungen für die Partialdichten, den Impulsgleichungen und der Energiegleichung, auf einem strukturierten krummlinigen Gitter, das sich aus regelmäßigen Vierecken zusammensetzt, diskretisiert. Dazu wurde ein explizites Shock-Capturing-Verfahren implementiert. In jedem Zeitschritt wird der Quellterm der chemischen Reaktionen mittels eines Operatorsplittings von den NSGen getrennt. Daraus resultiert ein System gewöhnlicher Differentialgleichungen (GDGen) und ein homogenes System partieller Differentialgleichungen (PDGen). Die Lösung der GDGen wird mittels eines expliziten Eulerschrittes approximiert. Die homogenen NSGen werden durch die Finite-Volumen-Methode, d.h. Integration über ein räumlich-zeitliches Kontinuum, diskretisiert. Die räumliche Integration wird mittels des Satzes von Gauß auf die Integration entlang des Zellrandes überführt. Bei der Diskretisierung der Randflüsse wird zwischen dem reibungsfreien und dem viskosen Anteil unterschieden. Die im viskosen Fluß auftretenden räumlichen Ableitungen 1. Ordnung werden durch zentrale Differenzen approximiert. Zur Diskretisierung des reibungsfreien Flusses werden in den PDGen die viskosen Terme vernachlässigt. Dies führt auf die Eulergleichung (EGen) in Integraldarstellung. Auf diese wird die Dimensionssplitting-Technik angewendet, d.h. die Erhaltungsgrößen werden auf jeder Zellkante konstant angenommen, und somit auch der Randfluß. Letzterer muß nur noch in den Zellrandmittelpunkten bestimmt werden. Dazu werden die Mittelpunkte zweier gegenüberliegender Zellkanten miteinander verbunden. Dadurch wird eine Schar einparametriger Kurven konstruiert. Entlang jeder dieser Kurven werden die räumlich eindimensionalen EGen mittels eines Godunov-Typ-Verfahrens – dem sogenannten Essentially Non-Oscillatory-Verfahren (ENO) [1] – diskretisiert. Dieses basiert auf einer Rekonstruktionstechnik (hier: RP-Methode, [1]), die aus den gegebenen Mittelwerten einer Funktion an diskreten Stellen eine kontinuierliche, stückweise glatte Funktion, die an den Zellrändern springen darf, approximiert. In den Zellrandmittelpunkten werden nun eindimensionale Riemannprobleme gelöst. Dies geschieht aus Rechenzeitgründen nicht exakt, sondern approximativ mit dem verallgemeinerten Löser von Liu und Vinokur [7] für die EGen im thermischen Gleichgewicht und chemischen Nichtgleichgewicht. Daraus ergibt sich eine Approximation des reibungsfreien Randflusses.
Das oben beschriebene Programm kann auf jedes System von Bilanzgleichungen 2. Ordnung, das konvektionsdominant ist, angewendet werden. Die NSGen sind nur ein Beispiel, das in der Aerodynamik von großer Bedeutung ist. In dieser Arbeit soll nun nicht auf die physikalische Modellierung und damit auf ein konkretes Gleichungssytem eingegangen werden. Statt dessen steht eine Reduzierung des enormen Rechenaufwands im Vordergrund. Dieser ist durch zwei Faktoren wesentlich bestimmt:

- kleine Zeitschrittweiten:
  - explizites Verfahren (CFL–Bedingung),
  - kurze Relaxationszeiten (steifes Problem),
  - dünne Grenzschichten (hohe Auflösung);

- hoher Aufwand pro Zelle:
  - Verfahren hoher Ordnung,
  - Riemannlöser,
  - physikalische Modellierung.

Für eine Staupunktströmung mit obigem physikalischen Modell müssen beispielsweise auf einem Gitter, das sich aus 31×148 Zellen zusammensetzt, bei einer Zeitschrittweite von $\Delta t = 10^{-8}$ sec etwa 20000 Schritte durchgeführt werden, bis die Strömung stationär ist. Dazu werden auf einer CRAY–Y/MP circa 45000 CPU sec benötigt. Wollte man eine entsprechende dreidimensionale Simulation durchführen, so wären also bei einer Auflösung von nur 30 Zellen in Umfangsrichtung zwei Wochen erforderlich. Um die Rechenzeit zu reduzieren werden also entweder schnellere Rechner benötigt oder man führt die Berechnung in möglichst vielen Zellen gleichzeitig durch. Eine gute Vektorisierbarkeit erfordert einen erheblichen Eingriff in die Programmstruktur und kann bei einigen Problemstellungen nur ungenügend realisiert werden. Daher ist eine Parallelisierung manchmal die einzig effektive Alternative. Diese hat den großen Vorteil, daß ein Eingriff in eine schon bestehende Programmstruktur nicht erforderlich ist. Dies soll anhand des oben beschriebenen Programms demonstriert werden. Dazu werden zunächst die allgemeinen zugrundeliegenden Gleichungen und die wesentlichen Techniken der Diskretisierung vorgestellt. Algorithmen werden hier nicht angegeben. Diese findet man für obiges Beispiel in [11]. Die Struktur des seriellen Programms wird darauf aufbauend vorgestellt. Die Details können in [12] nachgelesen werden.
Zur Parallelisierung wird in einem ersten Schritt ein zonales Konzept entwickelt, d.h. das strukturierte Gitter, auf dem bisher die Rechnungen durchgeführt wurden, wird in mehrere Zonen unterteilt, die dieselbe Struktur wie das Ausgangsgitter haben. Dadurch ist es möglich, das gleiche Programm wie bisher auf jede einzelne Zone anzuwenden. Dazu müssen nur einige wenige zusätzliche Variablen deklariert werden, die zonenabhängig sind. Bevor ein Zeitschritt in einer Zone durchgeführt werden kann, müssen dann die zonenabhängigen Daten in die schon existierenden zonenunabhängigen Variablen kopiert werden und danach vice versa. Was zusätzlich gewährleistet werden muß, ist der Datenaustausch zwischen zwei benachbarten Zonen. Dies geschieht nach jedem Zeitschritt. Dieses zonale Konzept kann auch dazu verwendet werden, um komplexere Geometrien mit geometrischen Singularitäten (z.B. Kanten, Knicke, Spalten, etc.) zu diskretisieren.
Die Parallelisierung besteht nun darin, jede Zone von einem Prozessor bearbeiten zu lassen. Dazu werden die zonenabhängigen Daten nicht mehr umgespeichert, sondern nun an den, die entsprechende Zone bearbeitenden, Prozessor geschickt, der nach einem Zeitschritt die Daten der Zonenränder zum Hauptprozessor zurücksendet. Dort findet dann der Daten–Abgleich zwischen den Zonen statt.
Hier ist die Tatsache, daß das Verfahren explizit ist, von enormen Vorteil. Dadurch ist die Abhängigkeit zwischen den einzelnen Zellen des Gitters nur lokal. Somit müssen nur die Zonenränder ausgetauscht werden. Bei einem impliziten Verfahren sind dagegen alle Zellen und somit auch alle Zonen miteinander verbundenen. Dies führt auf große Gleichungssyteme, die sich aus den Teilsystemen für die einzelnen Zonen zusammensetzen und miteinander gekoppelt sind. Entkoppelt man diese, und berechnet die Lösung dieser Teilsysteme parallel, so muß dies iterativ wiederholt werden, um den Entkopplungsfehler sukzessive zu verringern.
Der Datentransfer zwischen den Prozessoren wird über entsprechende Routinen gesteuert. Diese sind von der zugrundeliegenden Architektur des Parallelrechners abhängig. Da es heute eine Vielzahl von verschiedenen Architekturen gibt, führt dies zu Portabilitätsproblemen, d.h. das

Programm ist maschinenabhängig. Um dies zu vermeiden wurde das an der Fakultät für Mathematik der TU Chemnitz–Zwickau erstellte Software–Paket **PAR-LIB** zur Entwicklung paralleler Algorithmen verwendet. Dieses stellt Routinen zur Verfügung, die auf verschiedenen Systemen verwendet werden können:

- Transputersysteme (3L, Helios, Parix),
- HyperCube,
- Workstation–Cluster.

Am Beispiel eines Workstation–Clusters bestehend aus maximal vier Maschinen vom Typ IRIS–Indigo–R4000 werden die Transferzeiten untersucht. Um einen optimalen Speedup zu erzielen, muß dabei gewährleistet sein, daß die Transferzeiten möglichst gering sind im Vergleich zu den Rechenzeiten der einzelnen Prozessoren.

2 Die zugrundeliegenden Gleichungen

Es werden Gleichungen der Form

$$\begin{aligned} \boldsymbol{u}_{,t} &+ \text{div} \Big(\boldsymbol{F}_1^K(\boldsymbol{u}), \ldots, \boldsymbol{F}_d^K(\boldsymbol{u})\Big) \\ &- \text{div} \Big(\boldsymbol{F}_1^D(\boldsymbol{u}, \boldsymbol{u}_{,x_1}, \ldots, \boldsymbol{u}_{,x_d}), \ldots, \boldsymbol{F}_d^D(\boldsymbol{u}, \boldsymbol{u}_{,x_1}, \ldots, \boldsymbol{u}_{,x_d})\Big) = \boldsymbol{S}(\boldsymbol{u}) \end{aligned} \tag{1}$$

betrachtet.
Die Notation $(\cdot)_{,x_i}$ bedeutet dabei die partielle Ableitung der Größe $(\cdot)$ nach der Komponente x_i. Hierin ist $\boldsymbol{u}$ der Vektor der sogenannten Bilanzgrößen (z.B. Partialdichten, Impuls, Energien etc.), $\boldsymbol{F}_i^K$, $i = 1, d$, der konvektive Fluß in die $i-$te Ortsrichtung Transport der Informationen), $\boldsymbol{F}_i^D$, $i = 1, d$, der dissipative Fluß in die $i-$te Ortsrichtung (z.B. Diffusion, Reibung, Wärmeleitung, etc.) und $\boldsymbol{S}(\boldsymbol{u})$ der Quellterm (z.B. chemische Produktionsprozesse, thermische Austauschprozesse, äußere Kräfte, etc.).
Ein sehr wichtiges Beispiel in dieser Problemklasse sind die NSGen. Im Fall des chemischen Nichtgleichgewichts bestehen diese aus den Speziengleichungen, der Impulsgleichung und der Energiegleichung:

$$\frac{\partial \rho_i}{\partial t} + \text{div } \rho_i \boldsymbol{v} + \text{div } \boldsymbol{q}_i^D = s_i, \qquad i = 1, K \tag{2}$$

$$\frac{\partial \rho \boldsymbol{v}}{\partial t} + \Big(\text{div } \rho \boldsymbol{v}\boldsymbol{v}^T + p\boldsymbol{I}\Big) - \text{div } \boldsymbol{\tau} = \boldsymbol{0}, \tag{3}$$

$$\frac{\partial \rho E}{\partial t} + \text{div } (\rho \boldsymbol{v}(E + p/\rho)) - \text{div } (\boldsymbol{q} - \boldsymbol{\tau}\boldsymbol{v}) = 0 \,. \tag{4}$$

Dieses Gleichungssystem läßt sich in die Gestalt von Gleichung (1) bringen mit

- Vektor der Bilanzgrößen:
$$\boldsymbol{u} := (\rho_1, \ldots, \rho_K, \rho\boldsymbol{v}, \rho E)^T \tag{5}$$

- konvektiver Fluß in Richtung x_i:
$$\begin{aligned} F_i^K(\boldsymbol{u}) &:= (\rho_1 v_i, \ldots, \rho_K v_i, \rho\boldsymbol{v} v_i + p\boldsymbol{e}_i, \rho v_i H)^T \\ \boldsymbol{e}_i &= i\text{--ter Einheitsvektor des } R^{K+d+1} \end{aligned} \tag{6}$$

- dissipativer Fluß in Richtung x_i:
$$F_i^D(\boldsymbol{u}, \boldsymbol{u}_{,x_1}, \ldots, \boldsymbol{u}_{,x_d}) := (-q_{1,i}^D, \ldots, -q_{K,i}^D, (\boldsymbol{\tau})^i, (\boldsymbol{\tau}\boldsymbol{v})_i - q_i)^T \tag{7}$$

- Quellterm der chemischen Produktionsraten:
$$\begin{aligned} \boldsymbol{S}(\boldsymbol{u}) &:= (s_1, \ldots, s_K, 0, \ldots, 0)^T \\ s_i &= s_i(\rho_1, \ldots, \rho_K, T) \end{aligned} \tag{8}$$

Darin sind die Flüsse folgendermaßen definiert:

- Diffusionsfluß in Richtung x_i (nach [6], S. 516):
$$\begin{aligned} \boldsymbol{q}_i^D &:= \frac{c^2}{\rho} \sum_{j \neq i} M_i M_j D_{ij} \boldsymbol{d}_j - D_i^T \frac{1}{T} \,\textbf{grad}\; T \\ \boldsymbol{d}_i &:= \textbf{grad}\; \frac{c_i}{c} + \left(\frac{c_i}{c} - \frac{\rho_i}{\rho}\right) \frac{1}{p} \,\textbf{grad}\; p \end{aligned} \tag{9}$$

- Tensor der Reibungsspannungen:

$$\begin{aligned} \boldsymbol{\tau} &:= \mu(\,\mathbf{grad}\ \boldsymbol{v} + (\,\mathbf{grad}\ \boldsymbol{v})^T - \tfrac{2}{3}(\,\mathrm{div}\ \boldsymbol{v})\,\mathbf{I}) \\ (\boldsymbol{\tau})^i &= \quad i\text{-te Spalte von } \boldsymbol{\tau} \\ (\boldsymbol{\tau v})_i &= \quad i\text{-te Komponente von } \boldsymbol{\tau v} \end{aligned} \tag{10}$$

- Wärmefluß:

$$\begin{aligned} \boldsymbol{q} := -\lambda\ \mathbf{grad}\ T \quad &+ \quad \sum_i h_i \boldsymbol{q}_i^D \\ &+ \quad \frac{RT}{c} \sum_i \sum_{j \neq i} \frac{c_j}{M_i} \frac{D_i^T}{\mathcal{D}_{ij}} \left(\frac{\boldsymbol{q}_i^D}{\rho_i} - \frac{\boldsymbol{q}_j^D}{\rho_j} \right) \end{aligned} \tag{11}$$

Die hier auftretenden physikalischen Größen sind:

- Massendichte:

$$\rho := \sum_{i=1}^{K} \rho_i \tag{12}$$

- Massenbruch der Komponenete i:

$$Y_i := \frac{\rho_i}{\rho} \tag{13}$$

- molare Konzentration der Komponente i:

$$c_i := \frac{\rho_i}{M_i} \tag{14}$$

- Summe der molaren Konzentrationen:

$$c := \sum_{i=1}^{K} c_i \tag{15}$$

- Molenbruch der Komponente i:

$$X_i := \frac{c_i}{c} \tag{16}$$

- massenspezifische innere Energie (kalorische Zustandsgleichung):

$$e := \sum_{i=1}^{K} Y_i \left(e_i(T) + e_i^0 \right) \tag{17}$$

- massenspezifische Totalenergie:

$$E := e + \frac{1}{2} \boldsymbol{v}^2 \tag{18}$$

- Druck (thermische Zustandsgleichung):

$$p := \rho RT \sum_{i=1}^{K} Y_i / M_i \tag{19}$$

- massenspezifische Enthalpie der Komponente i:

$$h_i := e_i(T) + \frac{RT}{M_i} \tag{20}$$

- massenspezifische Enthalpie:

$$h := \sum_{i=1}^{K} Y_i \left(h_i + e_i^0 \right) = e + p/\rho \tag{21}$$

- massenspezifische Totalenthalpie:

$$H := E + p/\rho = h + \frac{1}{2}\boldsymbol{v}^2 \tag{22}$$

Die Gleichungen (2), (3) und (4) werden vervollständigt durch die Transport- und Diffusionskoeffizienten:

- Viskositätskoeffizient des Gasgemisches:

$$\mu = \mu(T, X_1, \ldots, X_K) \tag{23}$$

- Wärmeleitungskoeffizient des Gasgemisches:

$$\lambda = \lambda(T, X_1, \ldots, X_K) \tag{24}$$

- thermischer Diffusionskoeffizient der Komponente i:

$$D_i^T = D_i^T(T, p, X_1, \ldots, X_K) \tag{25}$$

- binärer Diffusionskoeffizient der Komponenten i und j:

$$\mathcal{D}_{ij} = \mathcal{D}_{ij}(T, p) \tag{26}$$

- Koeffizient der Mehrkomponentendiffusion der Komponenten i und j:

$$D_{ij} = D_{ij}(T, p, X_1, \ldots, X_K) \tag{27}$$

Die NSGen (1) sind für den räumlichen d-dimensionalen Fall formuliert. Sie wurden jedoch nur für $d = 2$ implementiert. Die hier präsentierten Testrechnungen wurden an einem rotationssymmetrischen Körper durchgeführt. Dadurch tritt in den räumlich zweidimensionalen Gleichungen ein zusätzlicher Quellterm auf. Auf diesen soll hier nicht weiter eingegangen werden. Näheres dazu entnehme man [10] und [12].

3 Numerische Methoden

Am Beispiel der räumlich zweidimensionalen Gleichungen (1) soll die Diskretisierung beschrieben werden. Dazu wurden folgende Techniken benutzt:

- Operatorsplitting für den Quellterm,
- Dimensionssplitting,
- Finite–Volumen–Technik,
- explizites Shock–Capturing–Verfahren für die konvektiven Terme,
- zentrale Differenzen für die dissipativen Terme.

Zu jedem expliziten Zeitschritt wird zunächst der Quellterm $\boldsymbol{S}(\boldsymbol{u})$ von den restlichen Gleichungen getrennt. Dies führt auf ein System von GDGen und ein System von PDGen, die sukzessive gelöst werden:

$$\text{Schritt 1:} \quad \frac{d\boldsymbol{u}^1}{dt} = \boldsymbol{S}(\boldsymbol{u}^1)\,, \tag{28}$$

$$\boldsymbol{u}^1(\boldsymbol{x}, t_n) = \boldsymbol{u}(\boldsymbol{x}, t_n)\,;$$

$$\begin{aligned} \text{Schritt 2:} \quad \boldsymbol{u}^2_{,t} \; &+ \; \operatorname{div}\, \Big(\boldsymbol{F}^I_1(\boldsymbol{u}^2), \boldsymbol{F}^I_2(\boldsymbol{u}^2)\Big) \\ &- \; \operatorname{div}\, \Big(\boldsymbol{F}^V_1(\boldsymbol{u}^2, \boldsymbol{u}^2_{,x_1}, \boldsymbol{u}^2_{,x_2}), \boldsymbol{F}^V_2(\boldsymbol{u}^2, \boldsymbol{u}^2_{,x_1}, \boldsymbol{u}^2_{,x_2})\Big) = 0\,, \end{aligned} \tag{29}$$

$$\boldsymbol{u}^2(\boldsymbol{x}, t_n) = \boldsymbol{u}^1(\boldsymbol{x}, t_{n+1})\,.$$

Da die Flüsse $\boldsymbol{F}^I_i$ und $\boldsymbol{F}^V_i$ zwei physikalische Effekte beschreiben (Konvektion bzw. Dissipation), die von unterschiedlicher Ordnung sind, wurden zu ihrer Diskretisierung zwei unterschiedliche Ansätze verwendet. Dies entspricht einem weiteren Operatorsplitting. Die Gleichung (29) wird somit in zwei voneinander unabhängige Systeme aufgespalten:

$$\text{Schritt 2.1:} \quad \boldsymbol{u}^{21}_{,t} + \operatorname{div}\, \Big(\boldsymbol{F}^I_1(\boldsymbol{u}^{21}), \boldsymbol{F}^I_2(\boldsymbol{u}^{21})\Big) = 0\,, \tag{30}$$

$$\boldsymbol{u}^{21}(\boldsymbol{x}, t_n) = \boldsymbol{u}^1(\boldsymbol{x}, t_{n+1})\,;$$

$$\text{Schritt 2.2:} \quad \boldsymbol{u}^{22}_{,t} - \operatorname{div}\, \Big(\boldsymbol{F}^V_1(\boldsymbol{u}^{22}, \boldsymbol{u}^{22}_{,x_1}, \boldsymbol{u}^{22}_{,x_2}), \boldsymbol{F}^V_2(\boldsymbol{u}^{22}, \boldsymbol{u}^{22}_{,x_1}, \boldsymbol{u}^{22}_{,x_2})\Big) = 0\,, \tag{31}$$

$$\boldsymbol{u}^{22}(\boldsymbol{x}, t_n) = \boldsymbol{u}^1(\boldsymbol{x}, t_{n+1})\,.$$

Diese werden in einem Zeitschritt simultan und nicht sukzessive gelöst, d.h.

$$\boldsymbol{u}^2_{,t} = \boldsymbol{u}^{21}_{,t} + \boldsymbol{u}^{22}_{,t}\,. \tag{32}$$

Wahlweise kann auf das Separieren des Quellterms verzichtet werden und diese Schritte auch simultan zu 2.1 und 2.2 durchgeführt werden. Dies ist für instationäre Probleme sinnvoll.
Die Systeme (30) und (31) werden mittels der Finiten–Volumen–Technik in Integralgleichungen überführt. Dazu werden sie über ein räumlich–zeitliches Kontinuum $[t_n, t_{n+1}] \times V,\ V \subset \mathbb{R}^2$ integriert und der Satz von Gauß angewendet. Daraus resultieren die Gleichungen

$$\begin{aligned} \text{a)} \quad \widehat{\boldsymbol{u}^{21}}(\boldsymbol{x}_M, t_{n+1}) &= \widehat{\boldsymbol{u}^{21}}(\boldsymbol{x}_M, t) - \frac{\Delta t}{|V|}\widehat{\boldsymbol{F}^I}_{N,\partial V}(\boldsymbol{u}^{21})\,, \\ \text{b)} \quad \widehat{\boldsymbol{u}^{22}}(\boldsymbol{x}_M, t_{n+1}) &= \widehat{\boldsymbol{u}^{22}}(\boldsymbol{x}_M, t) + \frac{\Delta t}{|V|}\widehat{\boldsymbol{F}^V}_{N,\partial V}(\boldsymbol{u}^{22})\,, \end{aligned} \tag{33}$$

mit den Notationen

- räumlicher Mittelwert von $\boldsymbol{u}$:

$$\hat{\boldsymbol{u}}(\boldsymbol{x}_{M,t}) := \frac{1}{|V|}\iint\limits_V \boldsymbol{u}(\boldsymbol{x}, t)\, dV \tag{34}$$

- konvektiver Fluß in Richtung der äußeren Normalen $\boldsymbol{N}$ zum Rand ∂V:

$$\boldsymbol{F}^I_{N,\partial V}\left(\boldsymbol{u}^{21}(\boldsymbol{x}, t)\right) := \left(\boldsymbol{F}^I_1\left(\boldsymbol{u}^{21}(\boldsymbol{x}, t)\right), \boldsymbol{F}^I_2\left(\boldsymbol{u}^{21}(\boldsymbol{x}, t)\right)\right) \cdot \boldsymbol{N}(\boldsymbol{x}) \tag{35}$$

- dissipativer Fluß in Richtung der äußeren Normalen $\boldsymbol{N}$ zum Rand ∂V:

$$\begin{aligned} &\boldsymbol{F}^V_{N,\partial V}\left(\boldsymbol{u}^{22}(\boldsymbol{x}, t), \boldsymbol{u}^{22}_{,x_1}(\boldsymbol{x}, t), \boldsymbol{u}^{22}_{,x_2}(\boldsymbol{x}, t)\right) := \\ &\qquad \Big(\boldsymbol{F}^V_1\left(\boldsymbol{u}^{22}(\boldsymbol{x}, t), \boldsymbol{u}^{22}_{,x_1}(\boldsymbol{x}, t), \boldsymbol{u}^{22}_{,x_2}(\boldsymbol{x}, t)\right), \\ &\qquad\qquad \boldsymbol{F}^V_2\left(\boldsymbol{u}^{22}(\boldsymbol{x}, t), \boldsymbol{u}^{22}_{,x_1}(\boldsymbol{x}, t), \boldsymbol{u}^{22}_{,x_2}(\boldsymbol{x}, t)\right)\Big) \cdot \boldsymbol{N}(\boldsymbol{x}) \end{aligned} \tag{36}$$

- zeitlicher Mittelwert des konvektiven Flusses über den Rand ∂V:

$$\widehat{\boldsymbol{F}^I}_{N,\partial V}(\boldsymbol{u}^{21}) := \frac{1}{\Delta t}\int\limits_{t_n}^{t_{n+1}}\int\limits_{\partial V} \boldsymbol{F}^I_{N,\partial V}\left(\boldsymbol{u}^{21}(\boldsymbol{x}(s), t)\right)\, ds\, dt \tag{37}$$

- zeitlicher Mittelwert des dissipativen Flusses über den Rand ∂V:

$$\widehat{\boldsymbol{F}^V}_{N,\partial V}(\boldsymbol{u}^{22}) := \frac{1}{\Delta t}\int\limits_{t_n}^{t_{n+1}}\int\limits_{\partial V} \boldsymbol{F}^V_{N,\partial V}\left(\boldsymbol{u}^{22}(\boldsymbol{x}(s), t)\right)\, ds\, dt \tag{38}$$

Dabei gibt $\Delta t := t_{n+1} - t_n > 0$ die Länge des Zeitintervalls, $|V|$ die Fläche von V und $\boldsymbol{x}_M \in V$ den Schwerpunkt von V an.
Aus (32) folgt dann die Approximation für (29)

$$\widehat{\boldsymbol{u}^2}(\boldsymbol{x}_M, t_{n+1}) = \widehat{\boldsymbol{u}^2}(\boldsymbol{x}_M, t_n) - \frac{\Delta t}{|V|}\left[\widehat{\boldsymbol{F}^I}_{N,\partial V}(\boldsymbol{u}^{21}) - \widehat{\boldsymbol{F}^V}_{N,\partial V}(\boldsymbol{u}^{22})\right]\,. \tag{39}$$

Wendet man die Finite–Volumen–Technik direkt auf die Gleichung (29) an, so ergibt sich (39) mit $\boldsymbol{u}^{21} = \boldsymbol{u}^{22} = \boldsymbol{u}^2$. Somit bewirkt das Aufspalten von (29) in (33a) und (33b) nur, daß die Flüsse an unterschiedlichen Stellen ausgewertet werden.
Zur Diskretisierung der Gleichung (30) wird ein explizites Shock-Capturing Verfahren verwendet. Es handelt sich dabei um das Essentially–Non–Oscillatory–Verfahren (ENO) von Harten u.a. [1]. Dieses basiert darauf, daß die Gleichung (30) über jede Zelle des diskretisierten Strömungsgebietes integriert bzw. (33a) berechnet wird.

In der hier durchgeführten numerischen Simulation wird der Strömungsbereich in allgemeine Vierecke unterteilt. Diese sollen so gewählt sein, daß die Seiten eines Viereckes möglichst senkrecht aufeinander stehen und etwa gleiche Seitenlängen haben. Ein Ausschnitt eines derart diskretisierten Gebietes ist in Abb. 1 dargestellt. Dabei sind die Punkte $\boldsymbol{P}_i^{j,k}$ die Mittelpunkte der vier Seiten $\Gamma_i^{j,k}$ der Zelle $V_{j,k}$ mit der konstanten äußeren Normalen $\boldsymbol{N}_i^{j,k}$. Man beachte, daß die äußeren Normalen zu einer Seite, die zwei benachbarten Zellen angehören, sich nur durch das Vorzeichen unterscheiden, d.h.

$$N_1^{j,k} = -N_3^{j,k+1} \quad \text{und} \quad N_2^{j,k} = -N_4^{j-1,k} \,. \tag{40}$$

Nun wird (33a) für jede Zelle $V_{j,k}$ berechnet. Die Mittelwerte $\widehat{\boldsymbol{u}^{21}}$ zum neuen Zeitpunkt ergeben sich aus denen des alten Zeitniveaus und der Differenz der Flüsse, d.h. dessen, was in dem Zeitintervall über die Zellgrenzen fließt. Zur Auswertung der Flüsse wird aber $\boldsymbol{u}^{21}$ und nicht der Mittelwert benötigt, d.h. aus den diskreten Mittelwerten $\widehat{\boldsymbol{u}^{21}}$ muß $\boldsymbol{u}^{21}$ rekonstruiert werden. Somit erhält man für das ENO-Verfahren folgende Schrittfolge eines expliziten Zeitschrittes auf obigem Strömungsgebiet:

a) **Initialisierung:**

$$\boldsymbol{v}_{j,k}^n \approx \widehat{\boldsymbol{u}^{21}}(\boldsymbol{x}_{j,k}, t_n) \,;$$

b) **Rekonstruktion von Ordnung $\boldsymbol{N}$:**

$$\boldsymbol{R}^N(\boldsymbol{x}, \boldsymbol{v}_{j,k}^n) = \boldsymbol{u}^{21}(\boldsymbol{x}, t_n) + \mathrm{O}\left(||\Delta x||^N\right) \,,$$

$$\boldsymbol{R}_{j,k}^N := \boldsymbol{R}^N|_{V_{j,k}} \in \mathcal{P}_2^N := \left\{ x_1^{l_1} x_2^{l_2} \;\middle|\; 0 \le l_1 + l_2 \le N \right\} \,;$$

c) **Numerischer Fluß:**

$$\widehat{\boldsymbol{F}^I}_{i,j,k} := \frac{1}{\Delta t} \int_{t_n}^{t_{n+1}} \int_{\Gamma_I^{j,k}} \boldsymbol{F}^I_{N_i^{j,k}, \Gamma_i^{j,k}}(\boldsymbol{w}(\boldsymbol{x}(s), t))\, ds\, dt \,, \tag{41}$$

$$\boldsymbol{F}^I_{N_i^{j,k}, \Gamma_i^{j,k}} := (\boldsymbol{F}_1^I, \boldsymbol{F}_2^I) \cdot \boldsymbol{N}_i^{j,k} \,;$$

$\boldsymbol{w}$ Lösung von (30) zu den Anfangsbedingungen $\boldsymbol{w}(\boldsymbol{x}, t_n) = \boldsymbol{R}^N(\boldsymbol{x}, \boldsymbol{v}_{j,k}^n)$;

d) **Mittelwertbildung:**

$$\begin{aligned} \boldsymbol{v}_{j,k}^{n+1} &= \boldsymbol{v}_{j,k}^n - \frac{\Delta t}{|V_{j,k}|} \sum_{i=1}^{4} \widehat{\boldsymbol{F}^I}_{i,j,k} \\ &\overset{(40)}{=} \boldsymbol{v}_{j,k}^n - \frac{\Delta t}{|V_{j,k}|} \left\{ \left[\widehat{\boldsymbol{F}^I}_{3,j,k} - \widehat{\boldsymbol{F}^I}_{1,j,k-1} \right] + \left[\widehat{\boldsymbol{F}^I}_{2,j,k} - \widehat{\boldsymbol{F}^I}_{4,j-1,k} \right] \right\} \,. \end{aligned}$$

Die Berechnung der Rekonstruktionspolynome $\boldsymbol{R}_{j,k}^N$ und der Flüsse $\widehat{\boldsymbol{F}^I}_{i,j,k}$ wird nun mittels der Dimensions–Splitting–Technik auf den räumlich eindimensionalen Fall zurückgeführt. Dazu wird angenommen, daß die Größe $\boldsymbol{u}^{21}$ entlang jeder Zellkante konstant ist. Dadurch muß der Fluß über den Zellrand nur in den Zellrandmittelpunkten ausgewertet werden. Die Zeitintegration wird durch die Mittelpunktsregel approximiert:

$$\widehat{\boldsymbol{F}^I}_{i,j,k} \approx \left|\Gamma_i^{j,k}\right| \; \boldsymbol{F}^I_{N_I^{j,k}, \Gamma_I^{j,k}} \left(\boldsymbol{w}(\boldsymbol{P}_i^{j,k}, t_n + \Delta t/2) \right) \,. \tag{42}$$

Zur Berechnung von $\boldsymbol{w}$ an der Stelle $\boldsymbol{P}_i^{j,k}$ reicht aufgrund des Dimensionssplittings ebenfalls eine eindimensionale Rekonstruktionstechnik. Durch Verbinden der Punkte $\boldsymbol{P}_2^{j,k}$ und $\boldsymbol{P}_4^{j,k}$ bzw. $\boldsymbol{P}_1^{j,k}$ und $\boldsymbol{P}_3^{j,k}$ für alle Zellen $V_{j,k}$ ergeben sich einparametrige Kurven $\boldsymbol{z}_k(\zeta)$ bzw. $\boldsymbol{z}_j(\eta)$ entlang denen $\boldsymbol{u}^{21}$ mittels der RP-Methode (siehe [1]) rekonstruiert wird (siehe Abb. 2).
In den Stellen $\boldsymbol{P}_i^{j,k}$ haben die Rekonstruktionspolynome $\boldsymbol{R}_j^N(\boldsymbol{z}_k(\zeta))$ bzw. $\boldsymbol{R}_k^N(\boldsymbol{z}_j(\eta))$ Unstetigkeiten. Um den physikalisch sinnvollen Zustand zu bestimmen, werden eindimensionale Riemannprobleme in Richtung der äußeren Normale zum Zellrand gelöst:

$$\begin{aligned} \boldsymbol{w}_{,t}(\xi,t) + \left(\left(\boldsymbol{F}_1^I(\boldsymbol{w}(\xi,t)), \boldsymbol{F}_2^I(\boldsymbol{w}(\xi,t))\right)\cdot \boldsymbol{N}_i^{j,k}\right)_{,\xi} &= 0\,, \quad \xi = \boldsymbol{N}_i^{j,k}\cdot \boldsymbol{x} \\ \boldsymbol{w}(\xi, t_n + \tfrac{\Delta t}{2}) &= \begin{cases} \boldsymbol{w}^N\left(\xi_i - 0, t_n + \frac{\Delta t}{2}\right), & \xi < \xi_i \\ \boldsymbol{w}^N\left(\xi_i + 0, t_n + \frac{\Delta t}{2}\right), & \xi > \xi_i \end{cases} \end{aligned} \tag{43}$$

zu den Zuständen

$$\boldsymbol{w}(\xi,t_n) = \boldsymbol{R}_j^N(\boldsymbol{z}_k(\xi)) \quad \text{bzw.} \quad \boldsymbol{w}(\xi,t_n) = \boldsymbol{R}_k^N(\boldsymbol{z}_j(\xi))\,. \tag{44}$$

Zur Lösung des Riemannproblems (43) wird der approximative Riemannlöser von Liu und Vinokur [7] für ein Gasgemisch, das sich im chemischen Nichtgleichgewicht befindet (siehe [11], S. 55 ff.) verwendet. Er wird jedoch nicht explizit der Zwischenzustand sondern der approximative Fluß ausgerechnet. Bei der Berechnung der Flüsse am Rand des Strömungsgebietes werden keine Riemannprobleme gelöst, sondern die Randwerte explizit in den Fluß eingesetzt.
Die Diskretisierung der Eulergleichungen (30) ist damit beschrieben. Sie stellt wohl den aufwendigsten Teil des Algorithmus' dar. In der Berechnung von (33b) wird der Mittelwert des viskosen Flusses durch die Zellmittelwerte auf dem alten Zeitniveau approximiert. Analog zu (42) ergibt sich

$$\widehat{\boldsymbol{F}^V}_{i,j,k} \approx \left|\Gamma_i^{j,k}\right| \; \boldsymbol{F}^V_{N_I^{j,k},\Gamma_i^{j,k}}\left(\boldsymbol{u}^{22}(\boldsymbol{P}_i,t_n), \boldsymbol{u}^{22}_{,x_1}(\boldsymbol{P}_i,t_n), \boldsymbol{u}^{22}_{,x_2}(\boldsymbol{P}_i,t_n)\right)\,. \tag{45}$$

Um $\boldsymbol{F}^V_{N_i^{j,k},\Gamma_i^{j,k}}$ im Punkt $\boldsymbol{P}_i$ auswerten zu können, werden sowohl die Größen $\boldsymbol{u}^{22}(\boldsymbol{P}_i)$ als auch die Ableitungen des Geschwindigkeitsvektors $\boldsymbol{v}_\xi(\boldsymbol{P}_i)$ und $\boldsymbol{v}_\eta(\boldsymbol{P}_i)$ benötigt, wobei (x_1,x_2) die kartesischen und (ξ,η) beliebige krummlinige Koordinaten sind (vgl. (36)). Für die Ableitungen gilt lokal:

$$\textbf{grad}\ \boldsymbol{v}(x_1,x_2) \approx \begin{pmatrix} N_1 & -N_2 \\ N_2 & N_1 \end{pmatrix} \textbf{grad}\ \boldsymbol{v}(\xi,\eta)\,. \tag{46}$$

Diese lassen sich durch zentrale Differenzen

$$\boldsymbol{v}_\xi(\boldsymbol{P}_i) \approx \frac{\boldsymbol{v}(\boldsymbol{P}_O) - \boldsymbol{v}(\boldsymbol{P}_W)}{||\boldsymbol{P}_W - \boldsymbol{P}_O||} \quad \text{bzw.} \quad \boldsymbol{v}_\eta(\boldsymbol{P}_i) \approx \frac{\boldsymbol{v}(\boldsymbol{P}_N) - \boldsymbol{v}(\boldsymbol{P}_S)}{||\boldsymbol{P}_N - \boldsymbol{P}_S||} \tag{47}$$

mit

$$\begin{aligned} \boldsymbol{v}(\boldsymbol{P}_N) &= \frac{1}{4}\Big(\boldsymbol{v}(\boldsymbol{P}_{NW}) + \boldsymbol{v}(\boldsymbol{P}_W) + \boldsymbol{v}(\boldsymbol{P}_O) + \boldsymbol{v}(\boldsymbol{P}_{NO})\Big) \\ \boldsymbol{v}(\boldsymbol{P}_S) &= \frac{1}{4}\Big(\boldsymbol{v}(\boldsymbol{P}_{SO}) + \boldsymbol{v}(\boldsymbol{P}_O) + \boldsymbol{v}(\boldsymbol{P}_W) + \boldsymbol{v}(\boldsymbol{P}_{SW})\Big) \end{aligned}$$

diskretisieren (vgl. Abb. 3). Analog gilt dies auch für alle anderen Größen und deren Ableitungen bzgl. ξ und η. Die Werte in den Zellmittelpunkten sind gerade die Werte auf dem alten Zeitniveau.
An der Körperwand muß man zwei Fälle unterscheiden (vgl. Abb. 4, 5):
Im Fall von Abb. 4 müssen in den Punkten $\boldsymbol{P}_{SW}$ und $\boldsymbol{P}_{SO}$ die Wanddaten verwendet werden. Durch Mittellung dieser Werte erhält man dann eine Diskretisierung für den Punkt $\boldsymbol{P}_S$. Bei der Diskretisierung des Flusses normal zur Körperwand (5) müssen in den Punkten $\boldsymbol{P}_{SW}, \boldsymbol{P}_W = \boldsymbol{P}_i$ und $\boldsymbol{P}_{NW}$ die Wanddaten verwendet werden. Durch Mittellung erhält man dann wiederum approximative Werte in den Punkten $\boldsymbol{P}_N$ und $\boldsymbol{P}_S$. Man beachte, daß in diesem Fall der Fluß eine andere Darstellung hat, da hier eventuell die Transportvorgänge an der Wand berücksichtigt werden müssen.

4 Programmstruktur

Um die Parallelisierung des Programms besser nachvollziehen zu können, ist die Kenntnis der Programmstruktur von Vorteil. Es sollen deshalb die wesentlichen Schritte zur Umsetzung des im vorangegangenen Kapitel beschriebenen Verfahrens erläutert werden. Darauf aufbauend werden die notwendigen Änderungen, die beim Übergang vom seriellen zum zonalen und parallelen Konzept erforderlich sind, beschrieben. Der Programmablauf der wichtigsten Routinen ist in Flußdiagrammen (vgl. 6) auch graphisch dargestellt, so daß ein Vergleich zwischen den Konzepten auch an diesen nachvollzogen werden kann.

4.1 Serielles Konzept

In Kapitel 3 wurden die zugrundeliegenden numerischen Methoden ausführlich beschrieben. Die Diskretisierung der Schritte (28), (30) und (31) wurde dabei im wesentlichen in der Reihenfolge vorgenommen, in der sie auch im Programm ablaufen. Schwerpunkt des Programms bildet die Diskretisierung von (30). Die Programmstruktur ist in einem Flußdiagramm (Abb. 11) dargestellt. Die einzelnen Schritte sind

INPUT Eingabe der strömungs– und diskretisierungsspezifischen Parameter über eine Datei, deren Datei zu Programmbeginn eingegeben wird;

INITIAL Einlesen der Daten und des Gitters sowie Initialisierung der zur Laufzeit benötigten physikalischen Größen (z.B. Tabellen für Kurvenfits), der Anströmwerte, der gitterspezifischen Größen, (z.B. Normale in den Zellrandmittelpunkten, Flächeninhalt der einzelnen Zellen, Bogenlänge der Zellkanten) und der rekonstruktionsspezifischen Größen (z.B. Parametrisierung eindimensionaler Kurven);

QUELLE Berechnung der Diskretisierung von (28), zur Modellierung des Quellterms mittels dem Operatorsplitting;

CFD Berechnung der Bilanzgrößen (5) auf dem neuen Zeitlevel gemäß (35) aus den alten Werten und den Flüssen über die Zellgrenzen in den Zellrandmittelpunkten;

BOUND Berechnung der Bilanzgrößen (5) in den Randzellen des Gitters;

OUTPUT Ausgabe der neuen Mittelwerte nach der Berechnung einer vorgegebenen Anzahl von Zeitschritten.

4.1.1 Strömungsroutinen

Das Kernstück des Programms, das auch mit Abstand am meisten Rechenzeit verbraucht, bildet die Berechnung der Strömung, d.h. die Diskretisierung von (29). Diese wird von der Prozedur **CFD** gesteuert und gliedert sich im wesentlichen in die Berechnung von (30) und (31). Die Struktur ist in einem Flußdiagramm (Abb. 12) dargestellt.

BOUND Berechnung der Bilanzgrößen (5) in den Randzellen des Gitters;

MWQUAN Berechnung der zur Laufzeit benötigten physikalischen Größen (z.B. Temperatur, Massenbrüche, Druck,etc.) aus den Bilanzgrößen zum alten Zeitpunkt;

REKON Berechnung der Rekonstruktion entlang der Kurven in ξ-Richtung – Kurve $\boldsymbol{z}_k(\xi)$ – und in η-Richtung – Kurve $\boldsymbol{z}_j(\eta)$ – über die einparametrige RP-Methode ([11], S. 33 ff.) und Auswertung der Polynome in den Zellrandmittelpunkten;

FLKON Lösung der Riemannprobleme in den Zellrandmittelpunkten entlang der Kurven $\boldsymbol{z}_k(\xi)$ und $\boldsymbol{z}_j(\eta)$ mittels (z.B. approximativer Riemannlöser von Liu und Vinokur [7] für Gas im Nichtgleichgewicht) und Berechnung des konvektiven numerischen Flusses (42);

FLDIS Berechnung der dissipativen Flüsse (7) und bei einer rotationssymmetrischen Rechnung des zusätzlichen dissipativen Quellterms über zentrale Differenzen gemäß (46) und (47);

MWSTEP Berechnung der Mittelwerte $\widehat{\boldsymbol{u}^{21}}$ und $\widehat{\boldsymbol{u}^{22}}$ gemäß (33a) und (33b) sowie bei einer rotationssymetrischen Rechnung die Berechnung des zusätzlichen konvektiven Quellterms; Dazu müssen die Flüsse über den Zellgrenzen gemäß (39) aufaddiert werden. Dies geschieht für den konvektiven und den dissipativen Fluß getrennt nach der Berechnung der entsprechenden numerischen Flüsse.

TEST Optionale Überprüfung, ob aus den Mittelwerten auf dem neuen Zeitniveau keine negativen Werte für positive physikalische Größen resultieren und eventuell erforderliche Korrekturmaßnahmen;

4.1.2 Randroutinen

Aufgrund der Verwendung von strukturierten Gittern setzt sich der Rand des Diskretisierungsgebietes aus vier Teilstücken und vier Eckzellen zusammen. Da das Verfahren auf dem Dimensionssplitting beruht, wird jedoch auf die Eckpunkte nur bei der Diskretisierung von (38) zugegriffen. Die Modellierung der Ränder bestimmt wesentlich das zu simulierende Strömungsproblem (z.B. Staupunktströmung, Stoßwellenkanal, Düsenströmung, etc.) Daher ist es sinnvoll, den vier Rändern keinen bestimmten Randtyp zuzuordnen, sondern die Entscheidung dem Benutzer über Flags vor Beginn der Rechnung zu überlassen. Dadurch können unterschiedliche Problemstellungen mit dem Programm simuliert werden. Die Berechnung der Randdaten erfolgt in vier Schritten:

- Vorbesetzung von gewissen Randzellen, um bei der Berechnung der Randdaten auf vorbesetzte Werte zuzugreifen;
- Berechnung der Randdaten bezüglich des gewählten Randtyps;
- Belegung der Eckzellen;
- Randwerte zur Berechnung der Rekonstruktion von Ordnung > 1.

4.1.3 Rekonstruktion

Die Berechnung der konvektiven numerischen Flüssen erfordert die Lösung von Riemannproblemen in den Zellrandmittelpunkten. Dazu werden die Bilanzgrößen aus den gegebenen Mittelwerten rekonstruiert und an den Zellrandmittelpunkten jeder Zelle ausgewertet (siehe (43),(44)). Die Berechnung ist sehr aufwendig. Zur Reduzierung des Aufwandes gibt es zwei Alternativen:

- Verwendung einer eindimensionalen Rekonstruktionstechnik entlang einer Parameterkurve statt einer zweidimensionalen Rekonstruktion (vgl. [11], Kap. 2.2, 2.3 und 3.2);

- Verwendung lokaler (physikalischer) Annahmen, die nur die Rekonstruktion einiger der Bilanzgrößen erfordert, z.B. ist bei einer lokal gefrorenen Strömung die Zusammensetzung in einer Zelle konstant, d.h. statt die Größen $\rho_1, \ldots, \rho_d, \rho\boldsymbol{v}, \rho E$ zu rekonstruieren, reicht die Rekonstruktion von $\rho, \rho\boldsymbol{v}, \rho E$ und die Annahme $Y_i = const.$ aus. Der Rekonstruktionsaufwand ist somit unabhängig von der Anzahl der Komponenten des Gasgemisches.

In dem hier verwendeten Programm wird die komponentenweise RP–Rekonstruktion entlang einer Parameterkurve berechnet. Diese setzt sich aus drei Schritten zusammen:

- Berechnung der ENO–Interpolationspolynome der Stammfunktion W (siehe [11], Kap. 2.2 und 3.2) in Newtondarstellung zu jeder Zelle;
- Taylorentwicklung der Interpolationspolynome um die Zellrandmittelwerte (siehe [1]);
- Ableiten der Interpolationspolynome zur Berechnung der gesuchten Rekonstruktion;
- Berechnung der Anfangszustände für die Riemannprobleme (43) durch Taylorentwicklung von $\boldsymbol{u}(\boldsymbol{z}_k(\xi), t_{n+1/2})$ bzw. $\boldsymbol{u}(\boldsymbol{z}_j(\eta), t_{n+1/2})$ um den Zellrandmittelpunkt $(\boldsymbol{z}_k(\xi_{j+1/2}), t_n)$ bzw. $(\boldsymbol{z}_j(\eta_{k+1/2}), t_n)$ (vgl. [11], Kap. 3.3).

4.1.4 Konvektiver numerischer Fluß

In den Zellrandmittelpunkten müssen die durch die Rekonstruktion ermittelten Riemannprobleme gelöst werden. Die Anfangszustände ergeben sich durch Auswertung der Rekonstruktionspolynome zu den benachbarten Zellen in diesem Punkt. Es interessiert hier nicht der Zwischenzustand $\tilde{\boldsymbol{u}}$ in einem Punkt, sondern der Fluß über den Zellrand in Normalenrichtung, also $\boldsymbol{F}_N^I(\tilde{\boldsymbol{u}}) := \left(\boldsymbol{F}_1^I(\tilde{\boldsymbol{u}}), \boldsymbol{F}_2^I(\tilde{\boldsymbol{u}})\right) \cdot \boldsymbol{N}$. Dabei ist zwischen dem inneren des Strömungsgebietes und dem Rand des Diskretisierungsbereiches zu unterscheiden:

- Berechnung aller konvektiven Flüsse über den Zellrand innerhalb des Strömungsbereiches und am Rand über die Lösung der eindimensionalen Riemannprobleme in Normalenrichtung;
- Berechnung aller konvektiven Flüsse über die Zellränder des Diskretisierungsrandes aus den Randwerten.

4.1.5 Dissipativer numerischer Fluß

Zur Bestimmung des dissipativen Flusses muß in den Zellrandmittelpunkten der dissipative Fluß $\boldsymbol{F}_N^V$ in Normalenrichtung $\boldsymbol{N}$ gemäß (36) berechnet werden. Dabei muß zwischen solchen Punkten, die in der Strömung liegen und solchen auf einer festen Körperwand, unterschieden:

- Berechnung aller viskosen Flüsse in den Zellrandmittelpunkten, die nicht auf einer Körperwand liegen;
- Berechnung aller viskosen Flüsse in den Zellrandmittelpunkten, die mit einer festen Körperwand zusammenfallen. In diesen Punkten ergeben sich Änderungen in der Berechnung, die in der physikalischen Modellierung der Wand begründet sind (z.B. Haftbedingung, Wanddiffusion, Wärmefluß, etc.). Diesen Wandeffekten muß gesondert Rechnung getragen werden.

4.2 Zonales Konzept

Der Einsatzbereich des in dem vorangegangenen Kapitel beschriebenen Programms wird wesentlich durch die Wahl eines strukturierten Gitters eingeschränkt. Dadurch können Geometrien mit lokalen Singularitäten, z.B. bei einer Kante oder einer Rampe, nicht hinreichend aufgelöst werden. Um dennoch nicht auf die Berechnung solcher Geometrien verzichten zu müssen, empfiehlt es sich, das Gitter aus mehreren strukturierten Teilzonen zusammenzusetzen. Die Berechnung der Strömung in den einzelnen Zonen kann dann wie bisher erfolgen. Problematisch sind hier allerdings die Zonenränder, an denen zwei Zonen aufeinander treffen. Man muß sich hier entscheiden, ob sich die Zonen überlappen sollen oder nicht. Es wurde hier das Konzept der Überlappung gewählt, da das Verfahren auf einer Finte–Volumen–Diskretisierung basiert. Dadurch kann die Gitterstruktur in den einzelnen Zonen beibehalten werden, d.h. es liegt jeweils eine Zelle außerhalb des Strömungsgebietes (vgl. Abb. 6). Zur Zeit wird gefordert, daß zwei angrenzende Zonen einen identischen Zonenrand haben (vgl. Abb. 7), d.h. die Ränder der beiden Zonen sind nicht versetzt (vgl. Abb. 8), und die Zonenaußenrandzelle der Zone A muß deckungsgleich mit der Zoneninnenrandzelle der Zone B sein (vgl. Abb. 9). Somit ist insbesondere die Anzahl der Zellen entlang eines Zonenrandes auf beiden Seiten identisch. Die Situation aus Abbildung 10 ist also ausgeschlossen. Löst man nun am Zonenrand ein Riemannproblem, so ist gewährleistet, daß der Fluß aus der Zone A in die Zone B identisch mit dem Fluß aus Zone B in Zone A ist. Dies ist der kritische Punkt, falls man allgemeinere Situationen zulassen möchte.
Aufgrund der Einschränkungen sind nur einige wenige Änderungen in dem Programm erforderlich. Sie betreffen im wesentlichen

- die Ein- und Ausgabe der Daten,
- die Initialiserung sowie
- die Schnittstellen zwischen den Zonen.

So setzt sich das Gitter nun aus mehreren benachbarten Zonen zusammen.
Zum Abspeichern der zonenabhängigen Daten und Parameter müssen nur einige wenige übergeordnete globale Variablen eingeführt, die dieselbe Struktur wie die sequentiellen Variablen haben, allerdings um einen Index für die Zonennummer erweitert. Die Routinen zur Berechnung des Quellterms **QUELLE** und der Strömungssimulation **CFD** werden somit zu Modulen, die für alle Zonen identisch sind. Im folgenden soll die Einbettung dieser Module in ein übergeordnetes Programm des zonalen Konzeptes beschrieben werden. Dieses weist keinerlei Änderungen gegenüber dem seriellen Programm auf. Allerdings müssen in einigen Routinen die entsprechenden Funktionen für alle Zonen statt bisher einer einzigen Zone durchgeführt werden:

ZINPUT Eingabe der strömungs– und diskretisierungsspezifischen Parameter für alle Zonen über eine Datei, deren Datei zu Programmbeginn eingegeben wird (analog zu INPUT);

ZINITIAL Einlesen der Daten und des Gitters sowie Initialisierung der zur Laufzeit benötigten physikalischen Größen (z.B. Tabellen für Kurvenfits), der Anströmwerte, der gitterspezifischen Größen, (z.B. Normale in den Zellrandmittelpunkten, Flächeninhalt der einzelnen Zellen, Bogenlänge der Zellkanten), der rekonstruktionsspezifischen Größen (z.B. Parametrisierung eindimensionaler Kurven) und zusätzlich der Zonenränder für alle Zonen (analog zu INPUT);

ZQUELLE Berechnung der Diskretisierung von (28) zur Modellierung des Quellterms mittels dem Operatorsplitting für alle Zonen;

ZCFD Berechnung der Bilanzgrößen (5) auf dem neuen Zeitlevel gemäß (35) aus den alten Werten und den Flüssen über die Zellgrenzen in den Zellrandmittelpunkten in allen Zonen;

ZONEIN(l) Umspeichern der zonenabhängigen Parameter und Daten der Zone l in die entsprechenden zonenunabhängigen Variablen;

BOUND Berechnung der Bilanzgrößen (5) in den Randzellen des Gitters;

ZONEOUT(l) Umspeichern der Zonenrandwerte und der neuen Bilanzgrößen in die entsprechenden zonenabhängigen Variablen der Zone l;

ZOUTPUT Ausgabe der neuen Mittelwerte aller Zonen nach der Berechnung einer vorgegebenen Anzahl von Zeitschritten (analog zu OUTPUT).

Zur Verwaltung der einzelnen Zonen werden nur folgende Informationen benötigt:

ZONANZ Anzahl der Zonen (zur Zeit nur 2 oder 4)

RZANZ Anzahl der aneinandergrenzenden Zonenränder

ZRAND(l, i) Zuordnung der Zonenränder $l = 1$,**RZANZ** zu den lokalen Rand– und Zonennummern der angrenzenden Zonen

Zusätzlich wird eine weiterer Randtyp benötigt. Dieser besetzt den Randwert an einem Zonenrand mit den inneren Werten der angrenzenden Zone.

4.2.1 Steuerung der Module

Die Berechnung des Quellterms bzw. der Strömung in einer Zone erfolgt wie bisher in der Routine **QUELLE** bzw. **CFD**. Die Routine **ZQUELLE** bzw. **ZCFD** steuert die sequentielle Berechnung in allen Zonen $l = 1$,**ZONANZ**. Dazu müssen vor dem Aufruf des entsprechenden Moduls die zonenabhängigen Daten umgespeichert werden in die entsprechenden zonenunabhängigen Variablen (vgl. Routine **ZONEIN**(l)) und nach jeder Zone die neu berechneten Daten wieder in die zonenabhängigen Variablen geschrieben werden (vgl. Routine **ZONEOUT**(l)). Die Zonenränder müssen, nachdem der Quellterm bzw. die Strömung in zwei aneinandergrenzenden Zonen berechnet wurde, ebenfalls auf den aktuellen Stand gebracht werden. Dies geschieht in der Routine **ZRANDUP**(l).

ZONEIN(l) Umspeichern der zonenabhängigen Parameter und Daten der Zone l in die entsprechenden zonenunabhängigen Variablen;

Modul Berechnung des Quellterms und Ausführung des Operatorsplittings bzw. Berechnung der Bilanzgrößen (5) auf dem neuen Zeitlevel in der Zone l. Diese Routine wird sequentiell für alle Zonen durchgeführt;

ZBOUT Aufdatieren der Zonenränder, die an die Nachbarzone übergeben werden, der gerade bearbeiteten Zone l vor dem Verlassen des zonenunabhängigen Bereiches;

ZONEOUT(l) Umspeichern der Zonenrandwerte und der neuen Bilanzgrößen in die entsprechenden zonenabhängigen Variablen der Zone l;

ZRANDUP(l) Aktualisierung der Zonenränder nachdem die Module in zwei aneinandergrenzenden Zonen berechnet wurden, d.h. die am Zonenrand anliegenden Werte einer Zone müssen als Randwerte in der angrenzenden Zone gesetzt werden.

4.2.2 Kommunikation zwischen den Zonen

Die Berechnung der chemischen Produktionsraten und der Strömung erfolgt für jede Zone sequentiell in den Modulen **QUELLE** und **CFD**. Dazu müssen vor und nach dem Aufruf dieser Routinen die benötigten zonenabhängigen Daten und Parameter umgespeichert werden in die entsprechenden zonenunabhängigen Variablen. Da die Strömung in den einzelnen Zonen nicht unabhängig von den angrenzenden Zonen ist, muß gewährleistet werden, daß ein Austausch zwischen zwei benachbarten Zonen über den Zonenrand stattfindet. Dies wird durch die vier folgenden Routinen gewährleistet:

ZONEIN(l) Vor der Berechnung des Quellterms oder der Strömung innerhalb der Zone l werden hier die folgenden zonenabhängigen Größen wie Zonengröße, zonenabhängige Parameter, zonales Gitter, zonale Bilanzgrößen, zonale Rekonstruktion und Zonenrandwerte in die entsprechenden zonenunabhängigen Variablen umgespeichert. Es ist sinnvoll, die Namen dieser Größen bis auf den Anfangsbuchstaben **Z** und den Index l identisch zu wählen;

ZBOUT Vor dem Verlassen des zonenunabhängigen Bereiches der Module **QUELLE** und **CFD** müssen die Zonenrandwerte, die an die Nachbarzone übergeben werden, der gerade bearbeiteten Zone l aufdatiert werden, bevor diese an die übergeordnete zonenabhängige Variable zurück übergeben werden und die nächste Zone bearbietet wird;

ZONEOUT(l) Nachdem die Bilanzgrößen in einer der beiden Modulen **QUELLE** und **CFD** für die Zone l berechnet wurden, müssen diese an die übergeordneten zonenabhängigen Variablen wie zonale Bilanzgrößen und Zonenrandwerte zurück übergeben werden, bevor die nächste Zone bearbeitet wird;

ZRANDUP(l) Nachdem die Produktionsterme oder die Strömung in zwei angrenzenden Zonen berechnet wurden, müssen die Zonenränder ebenfalls aktualisiert werden, d.h. die am Zonenrand anliegenden Werte einer Zone müssen als Randwerte in der angrenzenden Zone gesetzt werden. Dazu wird die Variable **ZRAND** benötigt. Sie ordnet einem Zonenrand die Nummern der angrenzenden Zonen und der lokalen Randnummer der jeweiligen Zone zu.

Das zonale Konzept befindet sich zur Zeit noch in der Testphase. Erste Rechnungen zeigen gute Ergebnisse. Es kann allerdings zu Problemen kommen, wenn eine Stoßwelle gegen einen Zonenrand läuft und über diesen hinwegtransportiert werden muß. Bei Rekonstruktionsordnung größer als 1 kann es passieren, daß der Stoß vor dem Zonenrand stehen bleibt. Der Grund dafür liegt wohl darin, daß zwischen zwei benachbarten Zonen nur die unmittelbaren Nachbarzellen ausgetauscht werden. Bei der Rekonstruktion höherer Ordnung sind allerdings entsprechend mehr Informationen zwischen den Zonen auszutauschen, jeweils der Ordnungszahl entsprechend benötigte Zellen auf beiden Seiten des Zonenrandes. Es empfiehlt sich deshalb zunächst mit einer Rechnung erster Ordnung zu beginnen, bis die Strömung fast stationär ist und dann zu höherer Ordnung und eventuell aufwendigeren physikalischen Modellen zu wechseln.

4.3 Paralleles Konzept

Die Parallelisierung des zonalen Konzeptes ist nun offensichtlich. Die Module zur Berechnung des Quellterms bzw. der Strömung erfolgt jetzt nicht mehr sequentiell sondern parallel. Dadurch entfällt das Umspeichern der zonenabhängigen Parameter und Variablen, die für alle Zeitschritte konstant sind. Diese müssen also nur einmalig an den entsprechenden Prozessor versendet werden. Dies gilt zusätzlich auch für die zonenunabhängigen Größen. Dazu werden einige Routinen benötigt, die die Initialisierung der Prozessoren, das Versenden und Empfangen von Daten zwischen den Prozessoren und das ordnungsgemäße Abschließen der Prozessorverbindungen nach Beendigung der Rechnung steuern. Hier wurde auf die in der Einleitung erwähnte Bibliothek **PAR-LIB** der TU Chemnitz–Zwickau zugegriffen, um das Programm portabel zu halten. Aus ihr wurden die folgenden sechs Routinen verwendet:

- **TRINIT**, **TRCLOSE**,
- **SEND_NODE_1**, **RECV_NODE_1**,
- **TREE_UP**, **TREE_DOWN**.

Die parallele Programmstruktur ist in Abbildung 16 dargestellt. Die Änderung gegenüber dem zonalen Konzept besteht darin, daß das Programm auf **NPROC**$(= 2^{\mathbf{NCUBE}})$ Prozessoren gleichzeitig läuft. Die Initialisierung der Rechnung wird nur auf einem Prozessor — dem Hauptprozessor (**ICH** $= 0$) — durchgeführt. Die initialisierten Werte werden dann an die Unterprozessoren geschickt (vgl. Routine **PPARSND**) und von diesen empfangen (vgl. Routine **PPARRCV**). Dies geschieht nur einmal zu Beginn der Rechnung, da die Speicherbereiche der einzelnen Prozessoren für die gleiche Variable verschieden sind. Vor der Berechnung des Quellterms bzw. der Strömung müssen die aktuellen Zonenränder vom Hauptprozessor an alle Unterprozessoren verschickt werden (vgl. Routine **PZONEIN**). Nach der gleichzeitigen Berechnung auf allen Prozessoren für einen Zeitschritt, werden die Zonenränder an den Hauptprozessor zurückgeschickt (vgl. Routine **PZONEOUT**) und dort aktualisiert (vgl. Routine **ZRANDUP**). Man beachte, daß der Hauptprozessor eine Doppelfunktion ausübt. Zum einen ist er für die Kommunikation zwischen den einzelnen Zonen (Prozessoren) zuständig, und zum anderen wird auf ihm die Berechnung für die erste Zone durchgeführt. Die Struktur des Hauptprogramms ist somit wie folgt gegeben:

TRINIT Initialisierung des parallelen Programms auf allen Prozessoren;

ZINPUT (nur auf dem Hauptprozessor)

ZINITIAL (nur auf dem Hauptprozessor)

PPARSND (nur auf dem Hauptprozessor)

Senden der zonenabhängigen und zonenunabhängigen Variablen und Parameter an die Unterprozessoren und Ablegen dieser Daten für die erste Zone in den entsprechenden zonenunabhängigen Größen auf dem Hauptprozessor;

PPARRCV (nur auf den Unterprozessoren (**ICH** $\neq 0$))

Empfangen der zonenabhängigen und zonenunabhängigen Variablen und Parameter der Zone l von dem Hauptprozessor auf dem entsprechenden Unterprozessor (**ICH** $= l - 1$) und Ablegen dieser Daten in den entsprechenden zonenunabhängigen Größen des Prozessors;

PQUELLE (optional, falls der Quellterm mit Operatorsplitting modelliert werden soll)

Parallele Berechnung der Diskretisierung von (28) zur Modellierung des Quellterms mittels dem Operatorsplitting für alle Zonen;

PCFD Parallele Berechnung der Bilanzgrößen (5) auf dem neuen Zeitlevel gemäß (35) aus den alten Werten und den Flüssen über die Zellgrenzen in den Zellrandmittelpunkten in allen Zonen;

PZONEIN Senden der Zonenränder der Zone l vom Hauptprozessor an den Unterprozessor (**ICH** $= l - 1$) und Empfangen der Zonenränder der Zone l auf dem Unterprozessor (**ICH** $= l - 1$) sowie Ablegen dieser Daten in die entsprechenden zonenunabhängigen Variablen;

BOUND Parallele Berechnung der Bilanzgrößen (5) in den Randzellen des Gitters der Zone l;

PZONEOUT Senden der Zonenränder der Zone l von dem Unterprozessor (**ICH** $= l - 1$) an den Hauptprozessor und Empfangen der Zonenränder der Zone l vom Unterprozessor (**ICH** $= l - 1$) auf dem Hauptprozessor sowie Ablegen dieser Daten in die entsprechenden zonenabhängigen Variablen;

DATABACK (nur auf dem Hauptprozessor)

Senden der neu berechneten Bilanzgrößen der Zone l von dem Unterprozessor (**ICH** $= l - 1$) an den Hauptprozessor und Empfangen der neu berechneten Bilanzgrößen der Zone l von dem Unterprozessor (**ICH**$= l-1$) auf dem Hauptprozessor sowie Ablegen dieser Daten in die entsprechenden zonenabhängigen Variablen;

ZOUTPUT (nur auf dem Hauptprozessor)

TRCLOSE Beenden des parallelen Programms auf allen Prozessoren.

4.3.1 Steuerung der Module

Die Berechnung des Quellterms bzw. der Strömung in einer Zone erfolgt wie bisher in der Routinen **QUELLE** bzw. **CFD**. Die Routine **PQUELLE** bzw. **PCFD** steuern die parallele Berechnung in allen Zonen $l = 1$,**ZONANZ**. Dazu müssen vor dem Aufruf der Module die Zonenrandwerte der Zone l vom Hauptprozessor an den Unterprozessor (**ICH** $= l - 1$) verschickt werden und dort in die entsprechenden zonenunabhängigen Variablen (vgl. Routine **PZONEIN**) abgelegt werden. Nach der Berechnung auf dem Unterprozessor werden die neuen Zonenrandwerte wieder an den Hauptprozessor zurückgeschickt und dort in die zonenabhängigen Variablen geschrieben werden (vgl. Routine **PZONEOUT**). Die Zonenränder müssen, nachdem die Module in zwei aneinandergrenzenden Zonen ausgeführt wurden, ebenfalls auf den aktuellen Stand gebracht werden. Dies geschieht in der Routine **ZRANDUP**.

PZONEIN Senden der Zonenränder der Zone l vom Hauptprozessor an den Unterprozessor (**ICH** $= l - 1$) und Empfangen der Zonenränder der Zone l auf dem Unterprozessor (**ICH** $= l - 1$) sowie Ablegen dieser Daten in die entsprechenden zonenunabhängigen Variablen;

Modul Berechnung des Quellterms und Ausführung des Operatorsplittings bzw. Berechnung der Bilanzgrößen (5) auf dem neuen Zeitlevel in der Zone l. Diese Routine wird parallel für alle Zonen durchgeführt;

ZBOUT Aufdatieren der Zonenränder, die an die Nachbarzone übergeben werden, der gerade bearbeiteten Zone l vor dem Verlassen des zonenunabhängigen Bereiches;

PZONEOUT Senden der Zonenränder der Zone l von dem Unterprozessor (**ICH** $= l-1$) an den Hauptprozessor und Empfangen der Zonenränder der Zone l vom Unterprozessor (**ICH**$= l-1$) auf dem Hauptprozessor sowie Ablegen dieser Daten in die entsprechenden zonenabhängigen Variablen;

ZRANDUP(l) Aktualisierung der Zonenränder nachdem die Module in zwei aneinandergrenzenden Zonen berechnet wurden, d.h. die am Zonenrand anliegenden Werte einer Zone müssen als Randwerte in der angrenzenden Zone gesetzt werden.

4.3.2 Kommunikation zwischen den Zonen

Da die Initialisierung gemäß den Routinen **ZINPUT** und **ZINITIAL** nur auf dem Hauptprozessor stattfindet, müssen die initialisierten Werte an die Unterprozessoren gesendet werden und dort von den entsprechenden Variablen empfangen werden. Ferner erfolgt die Berechnung der chemischen Produktionsraten und der Strömung für jede Zone parallel in den Modulen **QUELLE** und **CFD**. Dazu müssen vor dem Aufruf dieser Routinen die aktuellen Zonenränder an die Unterprozessoren geschickt werden und nach der Bearbeitung die neuen Werte an den Hauptprozessor zurückgesendet werden, damit ein Austausch zwischen zwei benachbarten Zonen über den Zonenrand stattfinden kann. Nach Beendigung der gesamten Simulation müssen die neuen Bilanzgrößen der einzelnen Zonen, die sich nur auf den Unterprozessoren befinden, an den Hauptprozessor geschickt werden und dort in die zonenabhängigen Variablen abgelegt werden. Dies alles wird durch die fünf folgenden Routinen gewährleistet:

PPARSND Senden der zonenabhängigen und zonenunabhängigen Größen wie Parameter, Zonengröße, Zonenabhängige Parameter, zonales Gitter, zonale Rekonstruktion, zonale Bilanzgrößen und Zonenrandwerte an die Unterprozessoren, und Ablegen dieser Daten für die erste Zone in den zonenunabhängigen Größen auf dem Hauptprozessoren;

PPARRCV Empfangen der zonenabhängigen und zonenunabhängigen Variablen und Parameter der Zone l von dem Hauptprozessor auf dem entsprechenden Unterprozessor (**ICH** $= l-1$) und Ablegen dieser Daten in die entsprechenden zonenunabhängigen Größen des Prozessors;

PZONEIN Vor der Berechnung des Quellterms oder der Strömung innerhalb der Zone l werden hier die Zonenränder der Zone l vom Hauptprozessor an den Unterprozessor (**ICH** $= l-1$) gesendet. Dieser Unterprozessor empfängt die Zonenränder der Zone l und legt diese in die entsprechenden zonenunabhängigen Variablen ab;

PZONEOUT Nachdem die Bilanzgrößen in einer der beiden Modulen **QUELLE** und **CFD** für die Zonen l = 1,**ZONANZ** berechnet wurden, müssen die Zonenränder aktualisiert werden. Dazu werden die Zonenränder der Zone l von dem Unterprozessor (**ICH** $= l-1$) an den Hauptprozessor gesendet. Der Hauptprozessor empfängt die Zonenränder der Zone l vom Unterprozessor (**ICH** $= l - 1$) und legt diese dort in die entsprechenden zonenabhängigen Variablen;

DATABACK Sendet die neu berechneten Bilanzgrößen der Zone l von dem Unterprozessor (**ICH** $= l - 1$) an den Hauptprozessor. Diese Werte werden auf dem Hauptprozessor empfangen und dort in die entsprechenden zonenabhängigen Variablen der Zone l abgelegt.

4.3.3 Kommunikation zwischen den Prozessoren

Aus der Bibliothek zur Entwicklung paralleler Algorithmen (vgl. [4]) wurden folgende Routinen benutzt:

TRINIT(0)

Initialisierung des parallelen Programms auf allen Prozessoren;

TRCLOSE

Beenden des Hauptprogramms auf allen Prozessoren;

SEND_NODE_1$(nWords, Data, NrTo)$

Senden der Daten $Data$ der Länge $nWords$ (1 Wort = 4 Byte) vom aktuellen Prozessor an den Prozessor $NrTo$;

RECV_NODE_1$(nWords, Data, NrFrom)$

Empfangen der Daten der Länge $nWords$ (1 Wort = 4 Byte) vom aktuellen Prozessor von dem Prozessor $NrFrom$ und schreibe diese nach $Data$;

TREE_UP$(Nlocal, Xin, Nout, Xout, MaxX)$

Empfangen der Daten $Xout$ der Länge $Nout$ (1 Wort = 4 Byte) auf dem Hauptprozessor (**ICH** = 0). Die maximale Länge von $Xout$ beträgt $MaxX$. Dieses Datenfeld setzt sich dabei aus den Daten Xin der Länge $Nlocal$, die lokal auf allen Prozessoren verteilt vorliegen, zusammen;

TREE_DOWN$(nWords, Data)$

Senden der Daten $data$ der Länge $nWords$ (1 Wort = 4 Byte) vom Hauptprozessor (**ICH** = 0) an alle Unterprozessoren.

Zur Kommunikation zwischen den Prozessoren werden einige globale Variablen benutzt. Diese stehen in der INCLUDE–Datei **trnet.inc**, die zu den Routinen dazugelinkt werden muß, die auf Routinen der Bibliothek zur Entwicklung paralleler Algorithmen zugreifen.
Die globalen Kommunikationsvariablen zwischen den Prozessoren, auf die hier zugegriffen werden, sind:

NCUBE Hypercube–Dimension (Anzahl der Links je Prozessor)

NPROC Anzahl der Prozessoren ($= 2^{\mathbf{NCUBE}}$)

ICH eigene Prozessornummer ($0, \ldots, \mathbf{NPROC} - 1$)

Die Beschreibung ist [4], S. 10, entnommen.

5 Ergebnisse

Das in den vorangegangenen Kapiteln beschriebene Verfahren wurde speziell für ein Gasgemisch im chemischen Nichtgleichgewicht implementiert. Simuliert wurden Staupunktströmungen um stumpfe Körper (zweidimensional bzw. rotationssymmetrisch). Es soll hier nicht auf die Ergebnisse dieser Untersuchungen eingegangen werden. Näheres dazu findet sich in [10], [11], [8] und [9]. Stattdessen sollen die beiden Fragen

- Lohnt sich die Parallelisierung, d.h. welcher Speedup–Faktor kann erzielt werden und was ist dabei zu beachten?
- Welchen Einfluß hat der Datentransfer auf den Speedup?

untersucht werden. Dies wurde auf einem Workstation–Cluster bestehend aus 4 Maschinen (IRIS–Indigo–R4000) durchgeführt. Hierbei wurde das an der Fakultät für Mathematik der TU Chemnitz–Zwickau entwickelte Software–Paket **PAR-LIB** sowie die am Oak Ridge National Laboratory erstellte Bibliothek **PVM 3** zum Aufbau einer parallelen virtuellen Maschine eines heterogenen Netzes benutzt (siehe [4], [3]).
Die Testrechnungen wurden mit zwei unterschiedlichen Gittern bestehend aus 146×46 bzw. 41×41 Zellen, die jeweils in zwei bzw. vier Zonen unterteilt wurden, durchgeführt. Die Ergebnisse sind in den Tabellen 1 — 4 im Anhang zusammengestellt. Aus ihnen lassen sich folgende Zusammenhänge ablesen:

- Der lokale Speedup–Faktor, d.h. die Rechenzeit in einer Zone bezogen auf die Gesamtzeit der seriellen Rechnung für das komplette Gitter, stimmt in allen Fällen ungefähr mit der Zonenanzahl überein (vgl. Tab. 1 - 4, Spalte 3). Dies war auch zu erwarten, da sich bei einer Unterteilung des Gitters in gleichgroße Zonen der Aufwand reziprok verringert. Der Faktor wird jedoch nicht ganz erreicht, da ein gewisser Overhead für die zusätzlichen Ränder erforderlich ist.
- Der globale Speedup–Faktor, d.h. die Gesamtlaufzeit (Rechnung und Datentransfer) der parallelen Rechnung bezogen auf die Gesamtzeit der seriellen Rechnung für das komplette Gitter, ist wesentlich geringer (vgl. Tab. 1 - 4, Spalte 4). Dies liegt an dem zum Teil recht erheblichen Aufwand für den Datentransfer. Dieser kann an dem Verhältnis der Transferzeit zur Gesamtzeit abgelesen werden (vgl. Tab. 1 - 4, Spalte 5).
- Der Einfluß der Transferzeit wird umso kleiner, desto kleiner das Verhältnis zwischen der Zahl der Randzellen einer Zone zu der Gesamtzahl der Zellen einer Zone wird (vgl. Tab. 1 - 4, Spalten 5, 6). Da die Zahl der Zonenzellen quadratisch von der Randzellenzahl abhängt, wird dieses mit zunehmender Randzellenzahl günstiger, d.h. die Zonen dürfen nicht zu klein gewählt werden.
- Die verwendete Transferroutine spielt keine Rolle, falls die zu transferierende Datenmenge hinreichend groß ist (vgl. Tabelle 1, 2, Spalten 5, 7). Bei kleiner werdender Datenmenge pro Zone (vgl. Tab. 3, 4, Spalten 5, 7) ist es günstiger alle Daten in ein Array zu schreiben und dieses an alle Prozessoren zu schicken (TREE_UP/ TREE_DOWN). Die Verzögerung durch die Kommunikation über das Netzwerk beruht in diesem Fall hauptsächlich auf der Synchronisation des Datentransfers, d.h. die vernetzten Rechner müssen sich über die Anzahl der zu transferierenden Daten sowie die Größe der Datenpakete abstimmen. Zusätzlich zu diesem Aufwand muß auf dem Zielrechner ein Puffer angelegt werden, in

dem die ankommenden Daten zwischengespeichert werden. Der Zeitaufwand für das Bereitstellen dieses Puffers ist abhängig von dessen Größe, derjenige für die Synchronisation des Datentransfers ist dagegen für jeden Transfer konstant. Beim Transfer großer Datenmengen fällt er daher kaum ins Gewicht. In diesem Fall können die zonenabhängigen Daten auch direkt an den entsprechenden Prozessor transferiert werden (SEND_NODE_1/ RECV_NODE_1).

Zusammenfassend ergeben sich daraus bezüglich der Effizienz der Parallelisierung folgende Schlußfolgerungen:

- Der Datentransfer hat einen wesentlichen Einfluß auf den Speedup–Faktor.
- Zur Effizienzsteigerung muß die Zellenanzahl einer Zone wesentlich größer sein als die Anzahl der Randzellen, damit der zonale Rechenaufwand größer ist als die Transferzeit der Randdaten.
- Die Zonen dürfen nicht zu klein sein bzw. der zonale Rechenaufwand muß sehr groß sein (aufwendiges numerisches Verfahren, komplexes physikalisches Modell, etc.).

Abschließend sei darauf hingewiesen, daß ein Workstation–Cluster wie er hier für das oben beschriebene Verfahren benutzt wurde nicht sehr effizient ist. Dies liegt wesentlich an dem Datentransfer. Dabei hat die zugrundeliegende Bibliothek der Parallelisierungsroutinen (z.B. **PVM**, **TCGMSG**, **POSYBL**, **C-Linda**, etc.) keine wesentliche Bedeutung. So wurden die Testrechnungen alternativ mit **TCGMSG** (vgl. [5]) durchgeführt. Eine wesentliche Änderung in den Zeiten konnte nicht festgestellt werden. Dies entspricht einer Untersuchung an der Yale University mit den oben angeführten Bibliotheken (vgl. [2]). Es sei aber darauf hingewiesen, daß die Entwicklung auf dem Gebiet der parallelen Systeme rasant ist und vielleicht schon in naher Zukunft eine effizientere Nutzung von Workstation–Clustern für das oben beschriebene Verfahren möglich ist.

6 Anhang

6.1 Abbildungen

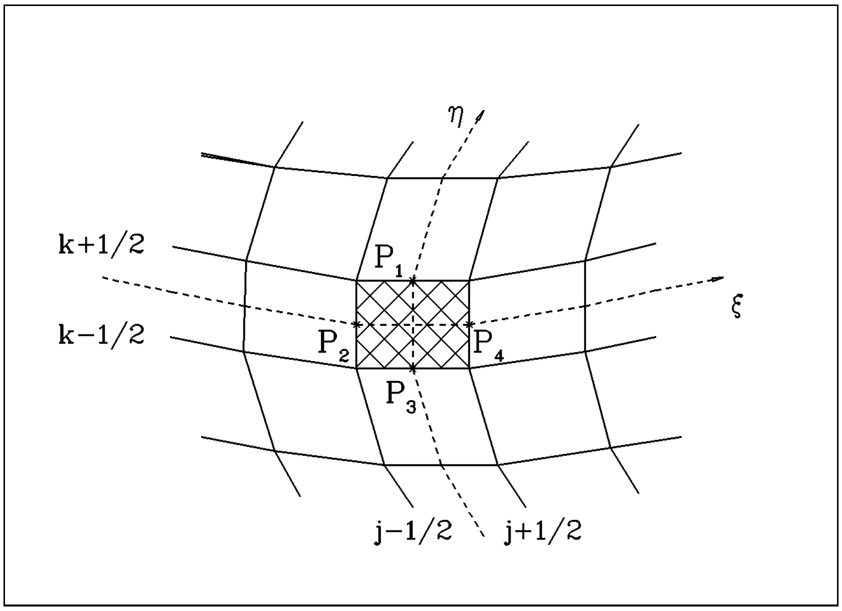

Abbildung 1: (entnommen aus [11], S. 33)

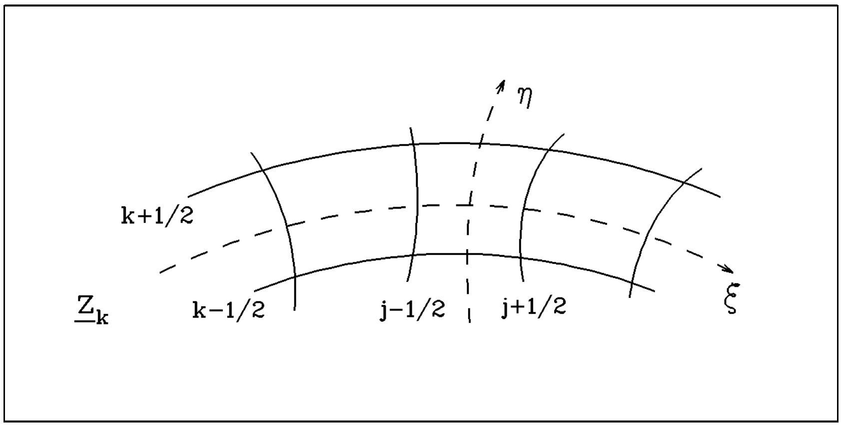

Abbildung 2: (entnommen aus [11])

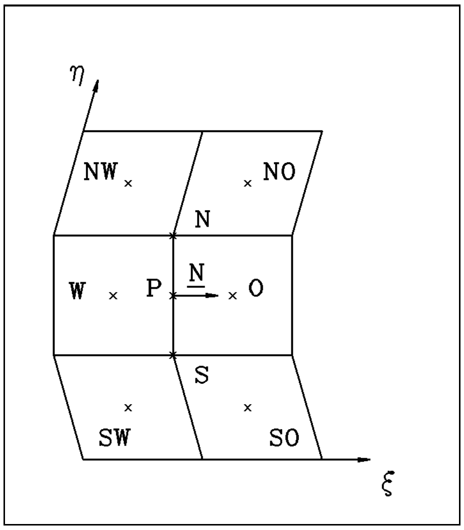

Abbildung 3: (entnommen aus [11])

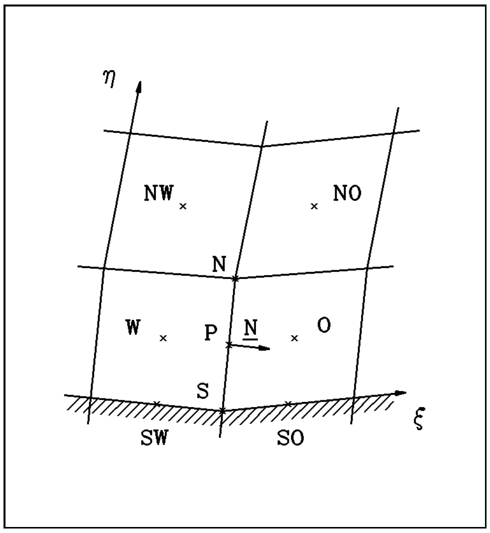

Abbildung 4: (entnommen aus [11])

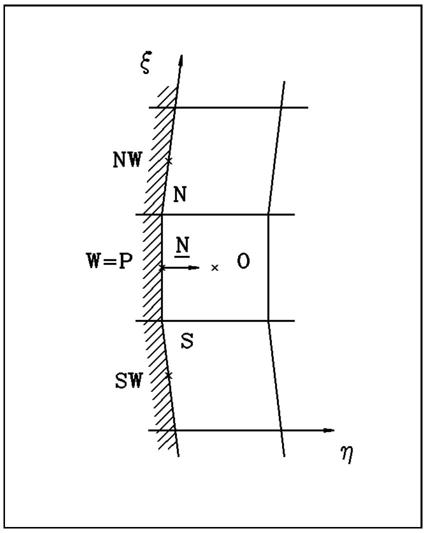

Abbildung 5: (entnommen aus [11])

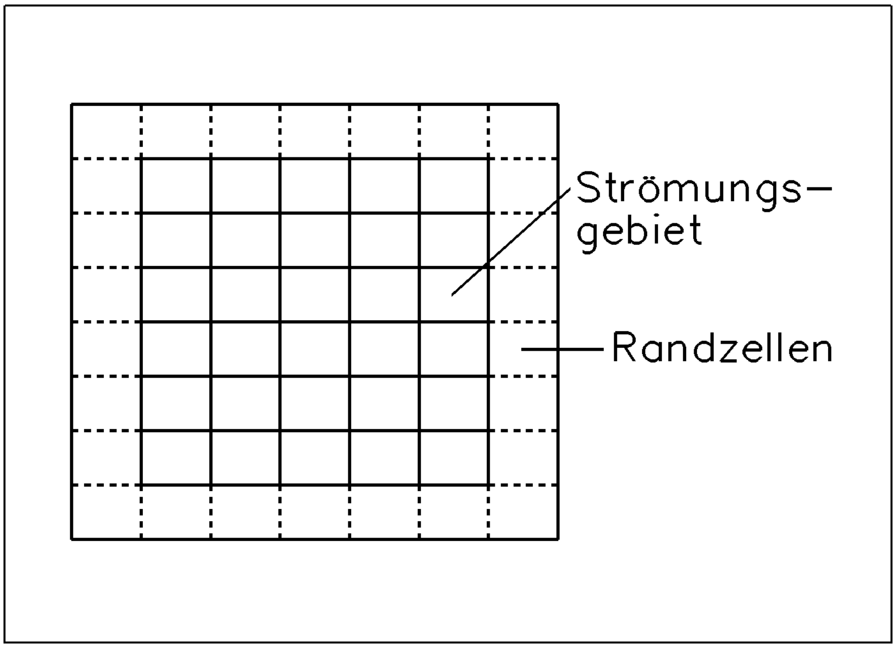

Abbildung 6: Aufbau einer Zone

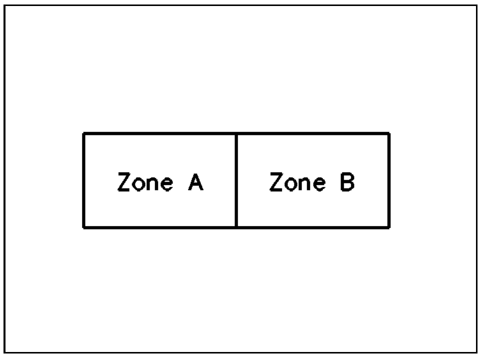

Abbildung 7: richtig

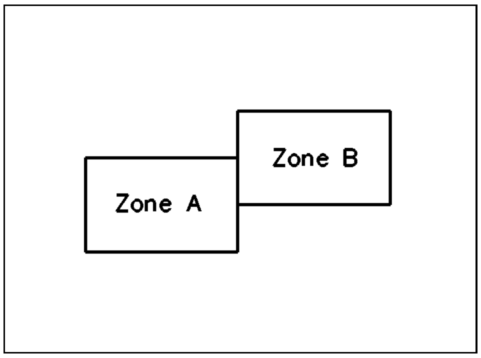

Abbildung 8: falsch

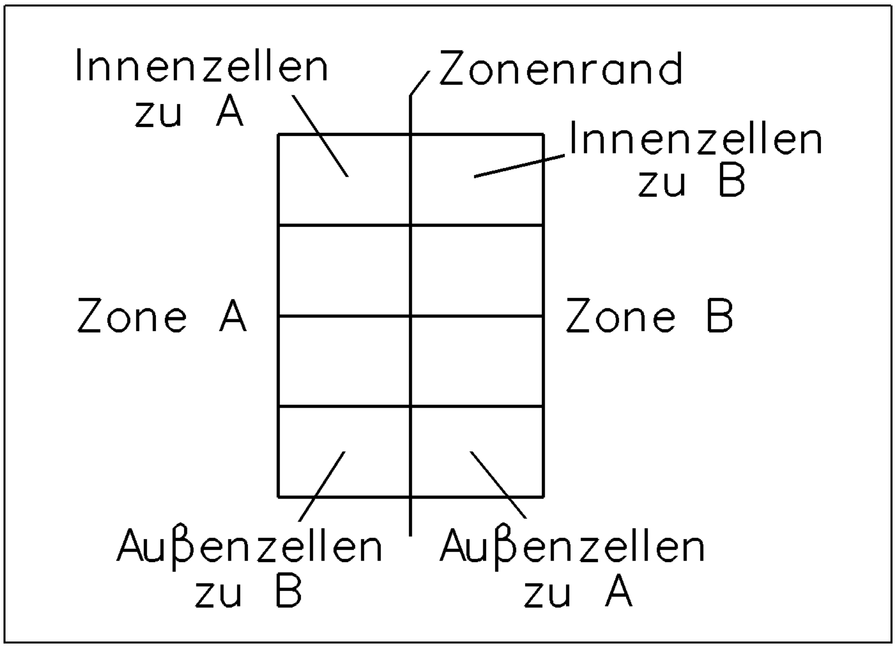

Abbildung 9: richtiger Aufbau eines Zonenrandes

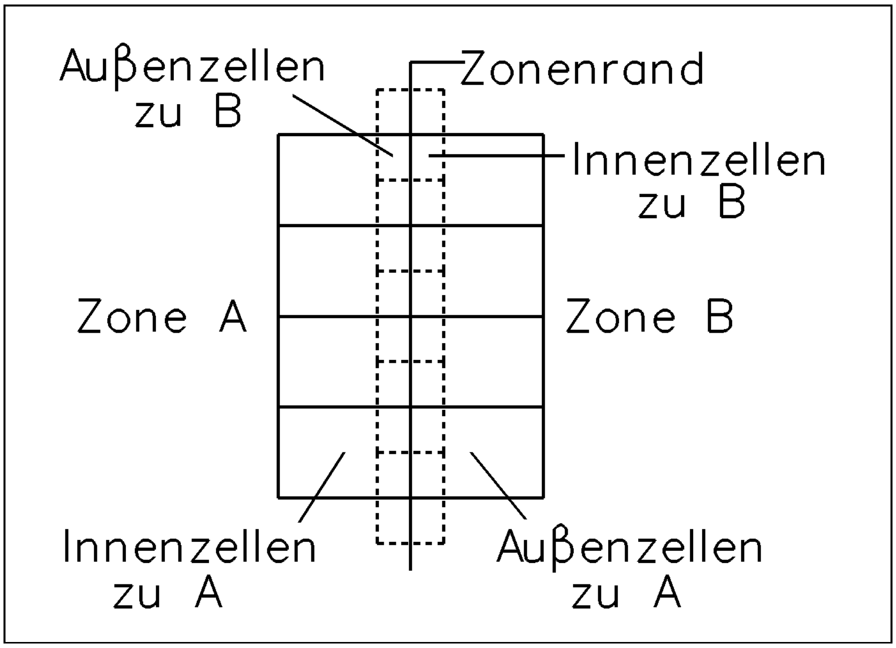

Abbildung 10: falscher Aufbau eines Zonenrandes

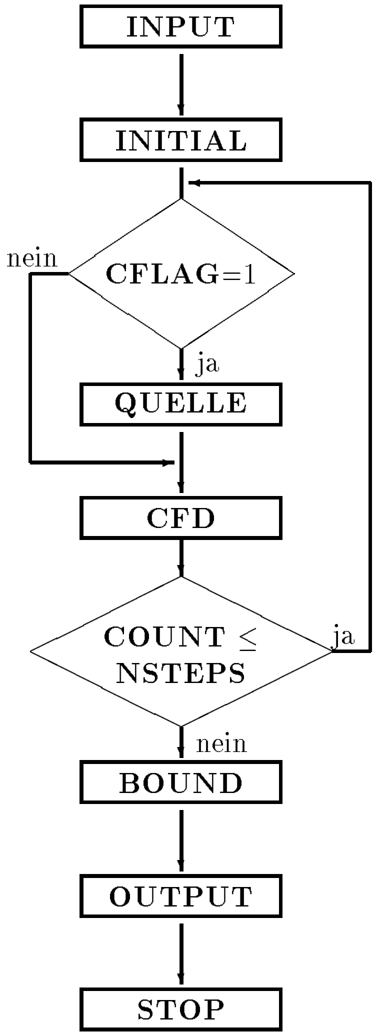

Abbildung 11: Flußdiagramm des seriellen Programms

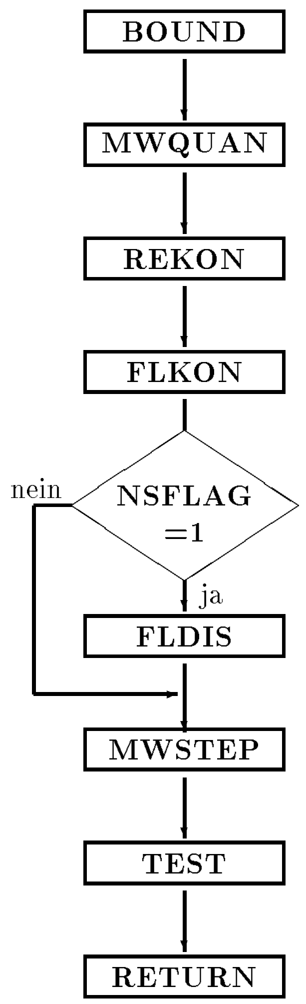

Abbildung 12: Flußdiagramm der Routine **CFD**

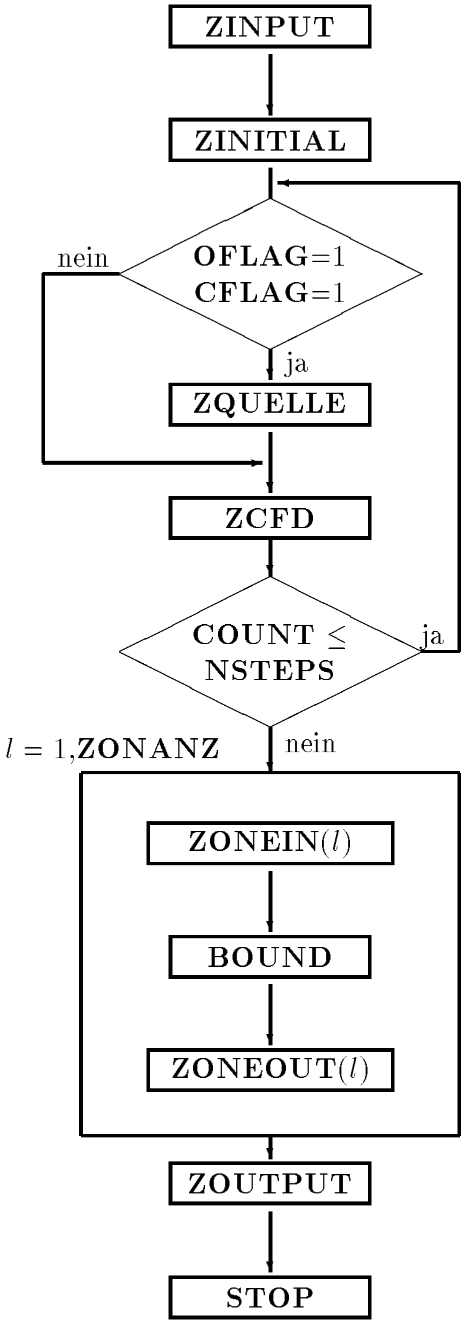

Abbildung 13: Flußdiagramm des zonalen Programms

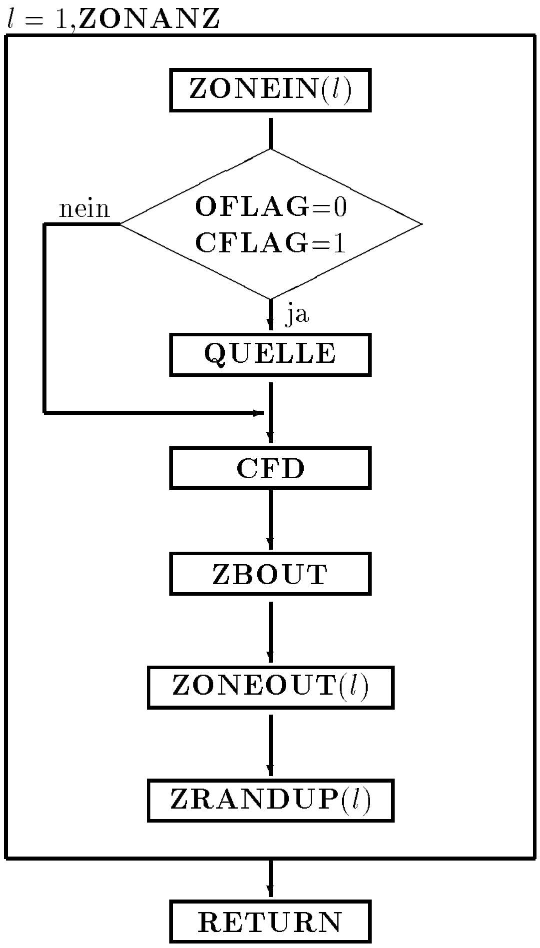

Abbildung 14: Flußdiagramm der Routine **ZCFD**

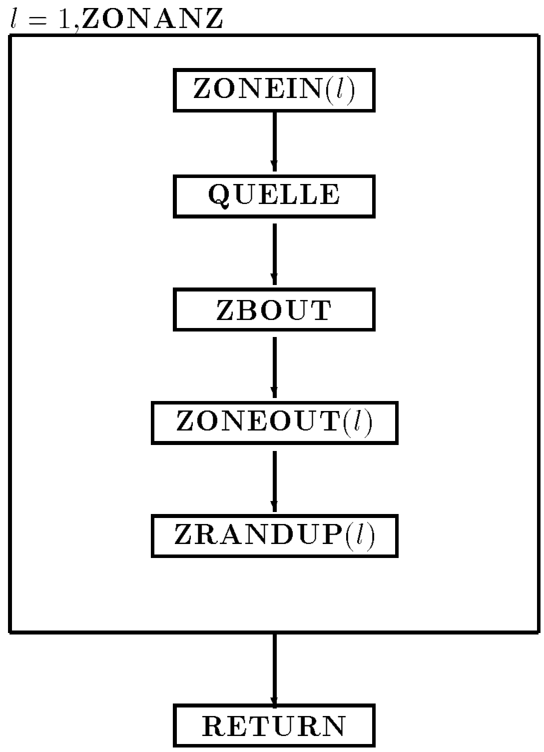

Abbildung 15: Flußdiagramm der Routine **ZQUELLE**

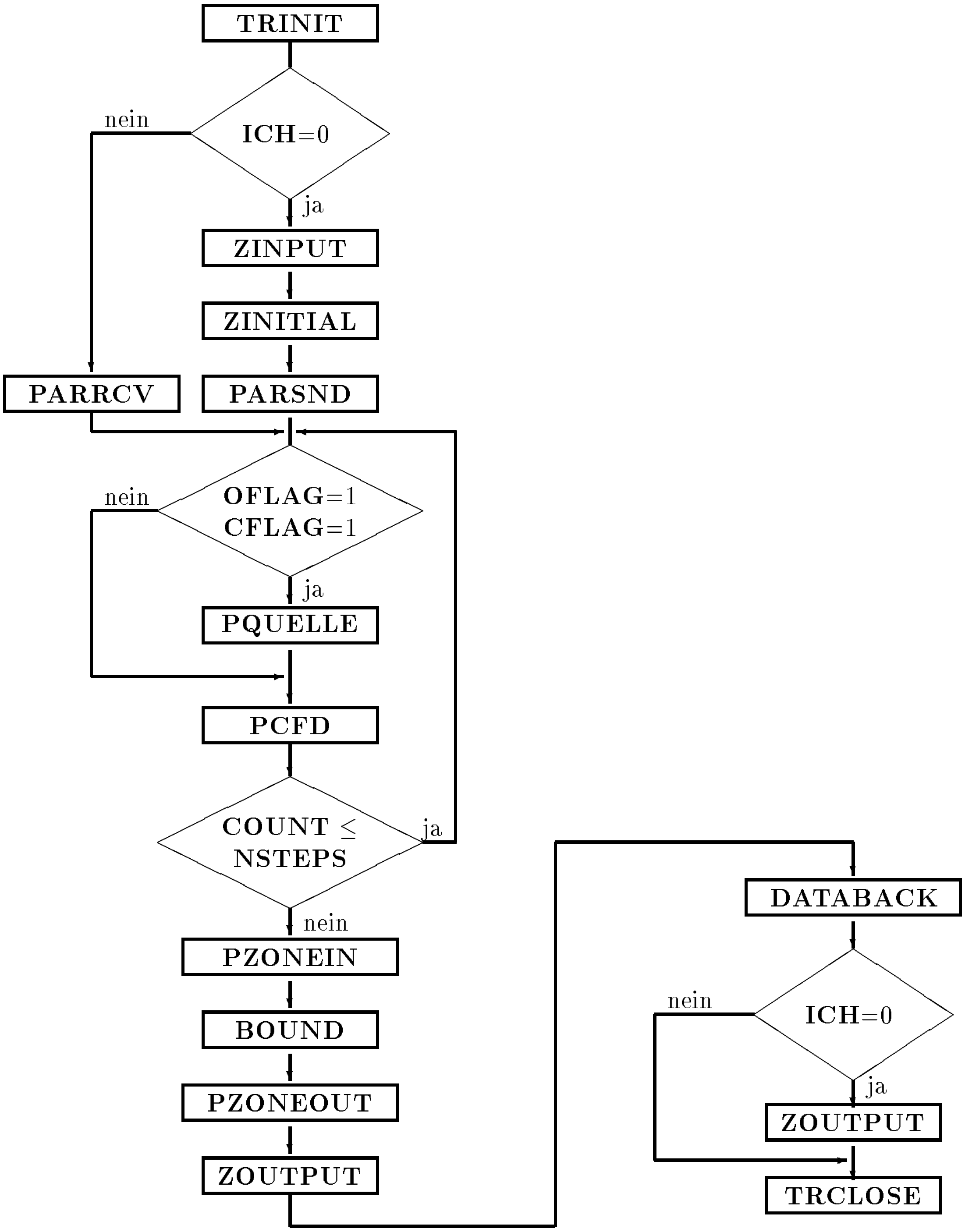

Abbildung 16: Flußdiagramm des parallelen Programms

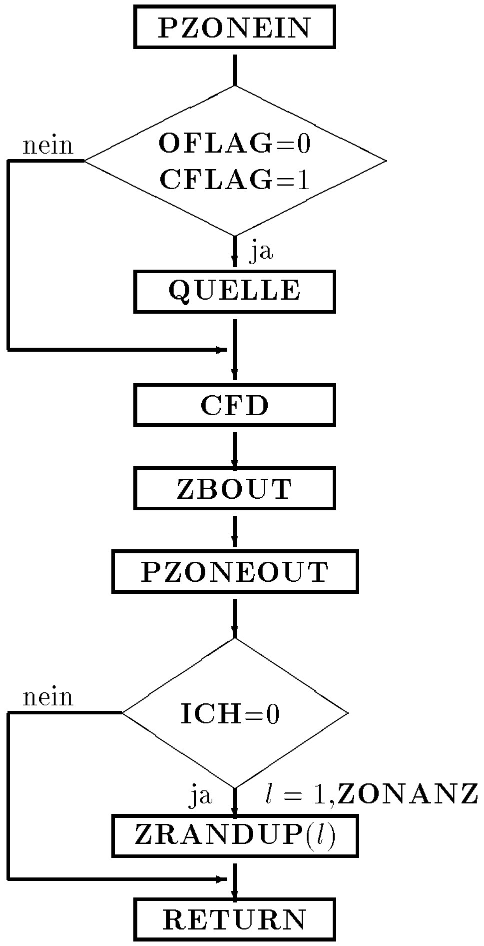

Abbildung 17: Flußdiagramm der Routine **PCFD**

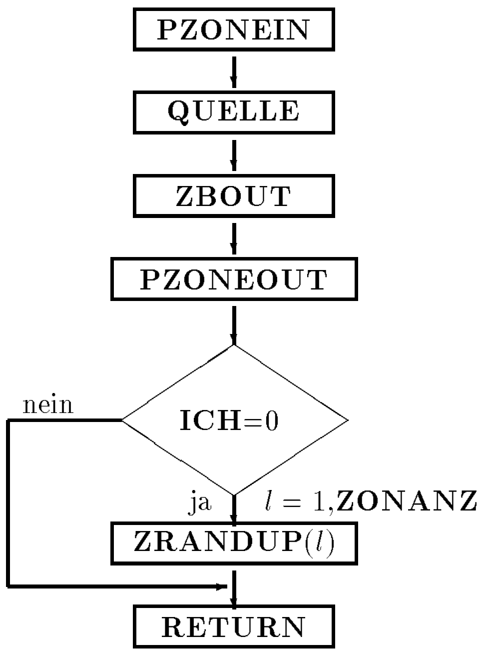

Abbildung 18: Flußdiagramm der Routine **PQUELLE**

6.2 Tabellen

TREE_UP/TREE_DOWN–Routinen						
Zonen	CPU's	Speedup		Transferzeit /	Randzellen /	Daten pro
		lokal	total	Rechenzeit	Zonenzellen	Transfer
4	4	3.7	2.2	0.40	0.13	14112
2	2	1.9	1.7	0.12	0.10	12240

Tabelle 1: 146×46 Zellen

SEND_NODE_1/RECV_NODE_1–Routinen						
Zonen	CPU's	Speedup		Transferzeit /	Randzellen /	Daten pro
		lokal	total	Rechenzeit	Zonenzellen	Transfer
4	4	3.6	2.2	0.39	0.13	3528
2	2	1.8	1.6	0.15	0.10	6120

Tabelle 2: 146×46 Zellen

TREE_UP/TREE_DOWN–Routinen						
Zonen	CPU's	Speedup		Transferzeit /	Randzellen /	Daten pro
		lokal	total	Rechenzeit	Zonenzellen	Transfer
4	4	3.3	1.6	0.50	0.20	6048
2	2	1.8	1.6	0.14	0.14	4320

Tabelle 3: 41×41 Zellen

SEND_NODE_1/RECV_NODE_1–Routinen						
Zonen	CPU's	Speedup		Transferzeit /	Randzellen /	Daten pro
		lokal	total	Rechenzeit	Zonenzellen	Transfer
4	4	3.3	1.1	0.68	0.20	1512
2	2	1.8	1.1	0.37	0.14	2160

Tabelle 4: 41×41 Zellen

Literatur

[1] Chakravarthy S.R., Engquist B., Harten A., Osher S., *"Uniformly High Order Accurate Essentially Non-Oscillatory Schemes III"*, J. Comp. Phys., Vol. 71, 1987, pp. 231 - 303

[2] Douglas, C.C., Mattson, T.G., Schultz, M., *"Parallel Programming Systems for Workstation Clusters"*, Department of Computer Science, Yale University, Research Report YALEU/DCS/TR - 975, August 1993

[3] Geist, A., Beguelin, A., Dongarra, J., Jiang,W., Manchek, R., Sunderam,V., *"PVM 3 User's Guide and Reference Manual"*, Oak Ridge National Laboratory, ORNL–Report TM–12187, Mai 1993

[4] Haase, G., Hommel, T., Meyer, T., Pester, M., *"Bibliotheken zur Entwicklung paralleler Algorithmen"*, Preprint–Reihe der Chemnitzer DFG–Forschergruppe *Scientific Parallel Computing*, TU Chemnitz–Zwickau, SPC 94_4, März 1994

[5] Harrison, R.J., *"Portable Tools and Applications for Parallel Computers"*, Int. J. Quantum Chem., Vol. 40, 1991, pp. 847 - 863

[6] Hirschfelder J.O., Curtiss Ch.F., Bird R.B., *"Molecular Theory of Gases and Liquids"*, John Wiley & Sons Inc., 4. Auflage, 1967

[7] Liu Y. , Vinokur M. , *"Nonequilibrium Flow Computations I. An Analysis of Numerical Formulations of Conservation Laws"*, J. Comp. Phys., Vol. 83, 1989, pp. 373 - 397

[8] Klomfaß A., Müller S., Ballmann J., *"Modellierung Hypersonischer Staupunktströmungen"*, Lehr– und Forschungsgebiet für Mechanik, Institut für Geometrie und Praktische Mathematik, RWTH Aachen, 1993

[9] Klomfaß A., Müller S., Ballmann J., *"Modelling Transport Phenomena for the Hypersonic Stagnation Point Heat Transfer Problem"*, AIAA–Paper AIAA-93-5047, presented at: AIAA, DGLR Fifth International Aerospace Planes and Hypersonics Technologies Conference, 30. Nov. –3. Dec.,1993, Munich (Germany)

[10] Müller S., Dickopp Ch., Ballmann J., Jeltsch R., *"Computation of Viscous Hypersonic Nonequilibrium Blunt Body Flow"*, ZFW Bd. 17, 1993

[11] Müller S., *"Erweiterung von ENO–Verfahren auf zwei Raumdimensionen und Anwendung auf hypersonische Staupunktprobleme"*, Dissertation, RWTH Aachen 1993, Verlag Shaker, Aachen

[12] Müller S., *"NSCNG2 – Ein Programm zur räumlich zweidimensionalen, instationären Strömungssimulation thermisch perfekter aber kalorisch nicht perfekter Gasgemische im chemischen Nichtgleichgewicht"*, Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Institutsbericht Nr. 89, 1993