Jül-4323

Jülich Supercomputing Centre (JSC)

Adaptives Lastbalance-Verfahren für Gebietszerlegung in der Molekulardynamik

Rene Halver

Mitglied der Helmholtz-Gemeinschaft

JÜLICH
FORSCHUNGSZENTRUM

Adaptives Lastbalance-Verfahren für Gebietszerlegung in der Molekulardynamik

Rene Halver

Berichte des Forschungszentrums Jülich; 4323
ISSN 0944-2952
Jülich Supercomputing Centre (JSC)
Jül-4323

Vollständig frei verfügbar im Internet auf dem Jülicher Open Access Server (JUWEL)
unter http://www.fz-juelich.de/zb/juwel

Zu beziehen durch: Forschungszentrum Jülich GmbH · Zentralbibliothek, Verlag
D-52425 Jülich · Bundesrepublik Deutschland
☎ 02461 61-5220 · Telefax: 02461 61-6103 · e-mail: zb-publikation@fz-juelich.de

Zusammenfassung der Arbeit

Diese Arbeit wird im Rahmen des Projektes ScaFaCoS (Scaleable Fast Coulomb Solvers) erstellt. Das Ziel von ScaFaCoS ist die Bereitstellung von Coulomblösern, die auf massiv parallelen Großrechnern skalieren sollen. Als Coulomblöser werden dabei Algorithmen bezeichnet, die in der Lage sind langreichweitige Wechselwirkungen zwischen Teilchen in einer Teilchensimulation zu berechnen. Da in einer Teilchensimulation die Teilchen sich entsprechend der durch die Coulomblöser simulierten Wechselwirkungen bewegen, muss im Laufe der Simulation ein weiterer Algorithmus dafür sorgen, dass die dadurch entstehende Lastungleichverteilung auf den verschiedenen Prozessoren eines Parallelrechners ausgeglichen wird. Dieser zweite Algorithmus heißt Lastbalance- oder Loadbalancing-Verfahren.
In dieser Arbeit ist nun die Entwicklung und Implementierung eines solchen Loadbalancingverfahrens für den Bestandteil MP2C des ScaFaCoS-Projektes beschrieben. Ebenso wird in dieser Arbeit die Entwicklung von effizienten Methoden zur Findung von Nachbarn der einzelnen Prozesse und der Entwurf eines effizienten Kommunikationsschemas nach Anwendung des Loadbalancing-Algorithmus behandelt. Abschließend wurden diese Implementierungen getestet und die Ergebnisse mit denen des unmodifizierten MP2C zu verglichen.

Summary

This thesis was produced within the German network project ScaFaCos (Scaleable Fast Coulomb Solvers) project. Its main goal is to develop a parallel library for various fast methods for electrostatics which can be used as molecular dynamics simulations. For efficient scaling of the methods on the massive parallel architectures, load-balancing is a very important topic to be applied in order to balance inhomogenous work load between the processors. The development, implementation and mathematical analysis of a diffusion based algorithm is the goal of this master thesis. The algorithm was tested for the parallel program MP2C, part of the application workpackage of ScaFaCos, containg a molecular dynamics module based a domain-decomposition scheme. As a side-effect the communications scheme in MP2C was generalized in order to handle communication with more then six neighbour domains. The thesis is concluded with tests of the implemented algorithms for inhomogenously distributed systems of polymer chains and a comparison of the original version with the modified one.

Inhaltsverzeichnis

1 Motivation dieser Arbeit **5**

1.1 Molekulardynamik und das Programmpaket MP2C 5

1.2 Allgemeine Probleme der Parallelisierung 6

1.3 Lösungsansätze 8

2 Grundlagen **11**

2.1 Wechselwirkungen 11

2.2 Molekulardynamik 13

2.3 Parallelisierte Molekulardynamik 16

2.4 Alternative Loadbalancing-Strategien 17

2.4.1 “Hashed-Oct-Tree”-Methode 18

2.4.2 Strategie der Wechselwirkungsverteilung 20

3 Verfahren zur Gebietsanpassung **21**

3.1 Definition der Arbeit 21

3.2 Bestimmung der neuen Grenzen 23

3.2.1 Beschreibung der Vorgehensweise 24

3.2.2 Kommunikation der lokalen Arbeiten 25

3.2.3 Idee zur Verschiebung der Grenzen 27

3.2.4 Globaler Arbeitsausgleich durch anteilige Übertragung von lokalen Arbeitsdifferenzen 28
3.2.5 Gitterpunktverschiebung anhand der Übertragung von anteiligen Arbeitsdifferenzen 30
3.2.6 Stabilität des Verfahrens 33
3.2.7 Überlegungen zur Konvergenz des Verfahrens 38
3.3 Kommunikationsschemata 42
3.3.1 Nachbarschaftsbestimmung 43
3.3.2 Kommunikation beim Teilchentransfer 44
3.3.3 Kommunikation bei der Kraftberechnung 46

4 Implementierung des Loadbalacingverfahrens 49
4.1 Beschreibung von MP2C 49
4.1.1 Wichtige Module 50
4.1.2 Geänderte Module 51

5 Untersuchung der Verfahrensparameter 55
5.1 Untersuchungen zur Anpassungsqualität des Verfahrens unabhängig von MP2C 55
5.2 Performanceuntersuchungen mit MP2C 56

6 Fazit und Ausblick 75
6.1 Bewertung des Verfahrens 75
6.2 Mögliche Ergänzungen und Erweiterungen 76

Abbildungsverzeichnis I

Tabellenverzeichnis III

Literaturverzeichnis **V**

Kapitel 1

Motivation dieser Arbeit

Als Einleitung in die Arbeit wird in diesem ersten Kapitel die Motivation dargestellt, warum die Arbeit sich mit dem Thema Loadbalancing im Zusammenhang mit Molekulardynamik beschäftigt. Außerdem wird beschrieben, was für ein Programm sich hinter dem Kürzel "MP2C" verbirgt. Abschließend werden einige Lösungsansätze, die dann im Verlauf der Arbeit genauer herausgearbeitet werden und beschrieben werden, dargelegt.

1.1 Molekulardynamik und das Programmpaket MP2C

Mit dem Begriff Molekulardynamik (MD) werden im Allgemeinen Computersimulationen beschrieben, mit denen Teilchensysteme und ihre inneren Wechselwirkungen untersucht werden. Hierbei wird nicht, wie bei Monte-Carlo Simulationen, auf stochastische Verfahren zurückgegriffen, sondern über einen kurzen Zeitraum die Wechselwirkungen zwischen einzelnen Teilchen oder Molekülen deterministisch berechnet. Dabei wird der Zeitraum in kleinere Zeitabschnitte diskretisiert, für jeden dieser Zeitabschnitte Kräfte und daraus resultierende Geschwindigkeitsänderungen der Teilchen berechnet und die Teilchen anschließend anhand der Resultate im System bewegt. Man kann

MD in verschiedenen statistischen Ensembles durchführen. Sind dabei die Energien, die Teilchenanzahl und das Volumen konstant, so spricht man von einem mikrokanonischem Ensemble, hält man an Stelle der Energie die Temperatur konstant, so befindet man sich in einem kanonischen Ensemble.
In der Molekulardynamik unterscheidet man zwischen lang- und kurzreichweitigen Wechselwirkungen, die zwischen Teilchen herrschen. Wie die Unterscheidung getroffen wird, wird im Detail im Kapitel Grundlagen (2.1) erklärt. An dieser Stelle sei nur erwähnt, dass langreichweitige Wechselwirkungen die Wechselwirkungen sind, die eine MD-Simulation aufwendig machen, da die Simulation durch sie zu einem $O(N^2)$-Problem wird, d.h. die Laufzeit vom Quadrat der Anzahl der betrachteten Teilchen abhängig ist.
MP2C ist nun ein Programmpaket, mit dem sowohl lang- als auch kurzreichweitige Wechselwirkungen betrachtet werden sollen. Allerdings verwendet MP2C nicht nur MD-Simulationen, sondern ermöglicht durch Verwendung von Monte-Carlo-Methoden auch die Simulation von hydrodynamischen Wechselwirkungen. Außerdem ist ein weiteres Ziel von MP2C , dass das Programmpaket auf massiv parallelen Hochleistungsrechnern auch bei der Verwendung von sehr hohen Prozessorzahlen noch skalieren soll. Das heißt dass ein möglichst hoher Teil der Rechnerleistung auch für die Berechnung genutzt werden kann und nicht beispielsweise durch Kommunikation verloren geht.
MP2C kommt vor allem bei Systemen zum Einsatz in denen sowohl hydrodymnamische, als auch molekulardynamische Wechselwirkungen berechnet werden müssen. Beispiele hier für sind Transport von Molekülketten in einer Flüssigkeit oder Simulationen von Blutkörperchen in der Blutbahn (Abb. (1.1) und (2.1)). Diese Systeme beinhalten durch die jeweiligen Flüssigkeiten hydrodynamische und durch die Moleküle, bzw. Blutkörperchen molekulardymnaische Wechselwirkungen, die alle berücksichtigt werden müssen.

1.2 Allgemeine Probleme der Parallelisierung

Im vorherigen Abschnitt wurde als Ziel von MP2C die Skalierbarkeit auch für große Prozessorzahlen angesprochen. Damit erkennbar wird, wo die Pro-

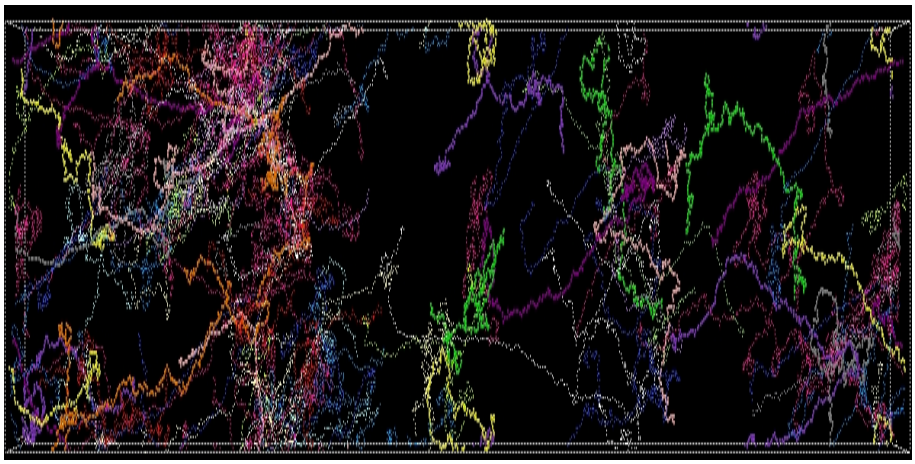

Abbildung 1.1: Beispiel für ein System mit Polymere, dessen Laufzeiten für das Verfahren in dieser Arbeit verbessert werden soll. Dieses Beispiel dient in den weiteren Kapiteln als Testsystem für Laufzeituntersuchungen. (s. Kapitel (5))

bleme der Erreichbarkeit dieses Zieles liegen muss man sich zunächst einmal die Systeme, die mit MP2C behandelt werden sollen, genauer anschauen. Als Beispiele wurden hier Polymere und Blutkörperchen genannt. Bei beiden Systemen kann es gebietsmäßig zu starken Inhomogenitäten in Bezug auf die Verteilung von Teilchen im System kommen. So sind bei linearen Polymeren die Teilchen alle entlang der Kette, zu der sie gehören, angeordnet. Die Ketten müssen aber keineswegs das System gleichmäßig ausfüllen, sondern werden sehr wahrscheinlich dazu neigen, sich ineinander zu verketten und so eine hohe Konzentration von Teilchen an einem Ort hervorrufen, während an anderen Stellen im System kaum ein Teilchen vorhanden ist. Dieses ist auch in Abbildung (1.1) zu erkennen. Dort sieht man, dass die Polymere sich teilweise häufen, aber andere Bereiche des Systemes kam ausfüllen.
Ein gängiges Vorgehen bei der Parallelisierung von MD-Programmen ist nun die Aufteilung des Systems in verschiedene Gebiete, das sogenannte **Domain Decomposition**. Diese Gebiete werden dann auf die einzelnen Prozesse aufgeteilt und jeder Prozess rechnet in jedem Rechenschritt die Kräfte und Geschwindigkeiten der Teilchen in seinem Gebiet aus. Am Ende wer-

den die Ergebnisse zusammengeführt und der Rechenschritt abgeschlossen. Liegen nun starke Inhomogenitäten in der Verteilung der Teilchen über dem Gebiet vor, so ist die Folge davon, dass einige Prozesse wenige bis gar keine Berechnungen durchzuführen haben, während andere sehr viele Berechnungen durchzuführen haben. Somit ist der Arbeitsaufwand auf den einzelnen Prozessen nicht mehr gleich und durch das dadurch bedingte Warten geht Zeit verloren, die ansonsten schon mit Folgeberechnungen genutzt werden könnte.
Desweiteren gibt es auch andere Probleme beim Parallelisieren, die bei seriellen Programmen in dieser Form nicht auftreten. Als Beispiel dafür sei hier das Problem der parallelen Ausgabe erwähnt, auf das in dieser Arbeit nicht weiter eingegangen wird.

1.3 Lösungsansätze

Um das Problem der Lastungleichverteilung in den Griff zu bekommen gibt es unterschiedliche Decomposition Verfahren, die auf verschiedenen Ansätzen beruhen. Dabei kann man zwei Hauptansätze unterscheiden. Das eine sind Methoden mit Domain Decomposition und das andere sind Methoden mit Force Decomposition.
Bei Force Decomposition werden die Berechnungen der Wechselwirkungen zwischen den Teilchen, die man als Matrix auffassen kann, auf die Prozesse verteilt und hierbei versucht, eine möglichst gleichmäßige Verteilung der Berechnungen, zu erreichen. Ein Problem, das hierbei auftritt ist vor allem das hohe Kommunikationsvolumen, um die berechneten Kräfte wieder auf alle Prozesse, die sie benötigen zurück zu verteilen. [13]
Demhingegen wird bei der Lastverteilung durch Domain Decomposition das Gebiet in dem das System betrachtet wird aufgeteilt und auf die Prozesse verteilt. Die Prozesse berechnen dann die Kräfte, die auf die Teilchen in dem Gebiet, das ihnen zugeteilt wurde, wirken und bewegen diese entsprechend. Beim Loadbalancing-Verfahren werden dann die Gebiete so angepasst, dass in jedem Gebiet etwa gleichviele Berechnungen anfallen und so die Last zwi-

schen den Prozessen ausgeglichen wird. Wie das im Einzelnen realisiert wird, wird im weiteren Verlauf der Arbeit genauer beschrieben.
Im Rahmen dieser Arbeit wird ausschließlich auf Verfahren mit Domain De-

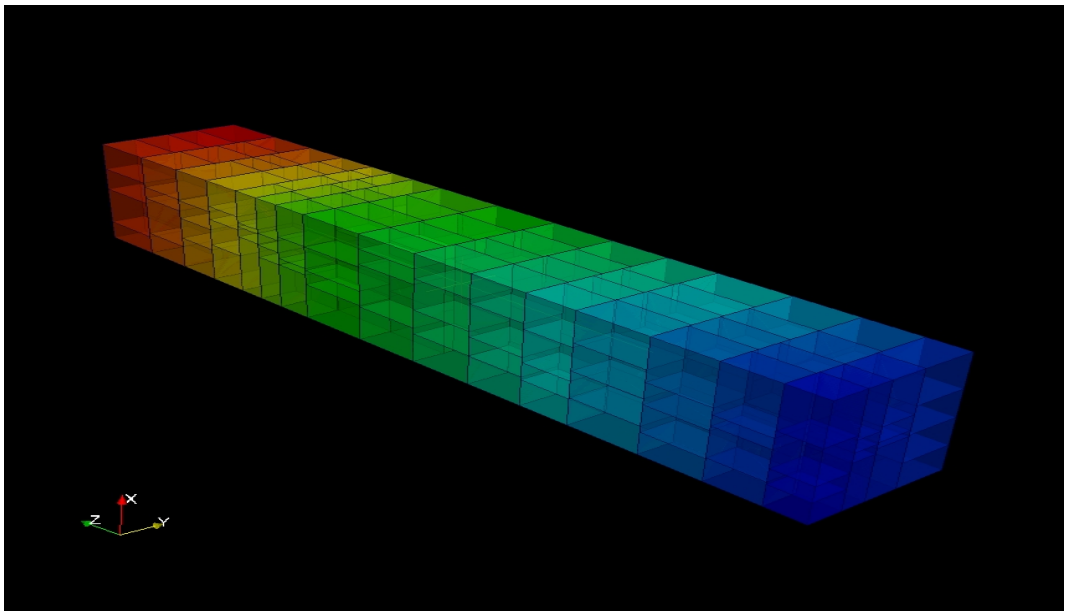

Abbildung 1.2: Beispielhafte Endunterteilung für das Testbeispiel, das in Abbildung (1.1) vorgestellt wurde. Gut zu sehen ist die unterschiedliche Einteilung der Gebiete in jeder z-Ebene.

composition eingegangen, da entschieden wurde für MP2C ein Loadbalancing mit Domain Decomposition zu entwerfen. Diese Entscheidung fiel auf Grund des Aufbaus von MP2C und der Art, wie in MP2C die Wechselwirkungen berechnet werden. Im nun folgenden Kapitel über Grundlagen werden am Ende kurz alternative Herangehensweise für Loadbalancing vorgestellt und erklärt, warum diese Verfahren sich nicht für MP2C eignen. Das Verfahren, das im Verlauf dieser Arbeit entwickelt wurde, wird dann im darauf folgenden Kapitel (3) erläutert.
Allerdings seien an dieser Stelle die Anforderungen an das zu entwerfende Verfahren notiert:
Das Verfahren sollte:

- möglichst nur lokale oder Informationen von direkten Nachbarn benötigen
- eine Gebietsaufteilung liefern, die zusammenhängende Gebiete auf Prozesse abbildet
- möglichst einfach nachzuvollziehen sein

Wie eine mögliche Gebietsanpassung am Ende aussehen könnte, zeigt Abbildung (1.2). Dies ist eine mögliche Unterteilung des Gebietes in Untergebiete für das in Abbildung (1.1) präsentierte Testbeispiel. Es ist deutlich zu sehen, dass die endgültige Unterteilung sich stark von der äquidistanten Unterteilung unterscheidet.

Kapitel 2

Grundlagen

In diesem Kapitel soll eine einführende Übersicht über das Thema Molekulardynamik und die der Molekulardynamik zu Grunde liegenden Physik gegeben werden. Außerdem wird kurz auf parallelisierte Molekulardynamik eingegangen, um die Probleme aufzuzeigen, die damit einher gehen und wie diese Arbeit in diesen Zusammenhang einzuordnen ist. Abgeschlossen wird das Kapitel mit einer kurzen Übersicht über alternative Loadbalancing-Algorithmen, sowie die Gründe, warum diese nicht für MP2C verwendet wurden.

2.1 Wechselwirkungen

In der Molekulardynamik werden physikalische Wechselwirkungen zwischen Teilchen simuliert. Als Wechselwirkung wird hierbei die Tatsache bezeichnet, dass Kräfte immer paarweise auftreten. Zu jeder Kraft, die von einer Größe auf eine andere ausgeübt wird, gibt es eine Kraft, die parallel zur ersten Kraft wirkt, aber zu dieser antiparallel ist:

Nach dem dritten Newtonschen Axiom entspricht die Kraft, die von einer physikalischen Größe auf die anderen wirkt, der negativen Kraft, die anderen Größen auf die erste ausüben. [11] Ein Beispiel für eine solche Wechselwir-

Formel 1 (Wechselwirkung (3. Newtonsches Axiom))
$\overrightarrow{F_{ij}} = -\overrightarrow{F_{ji}}$ *("actio = reactio", Drittes Newtonsches Axiom)*

Formel 2 (Coulombsches Gesetz)
$\overrightarrow{F_{ij}} = \frac{1}{4\pi\varepsilon_0}\frac{q_i \cdot q_j}{r^3} \cdot \overrightarrow{r_{ij}}$

kungskraft im Alltag ist die Gravitationskraft, die sowohl von der Erde auf jeden Gegenstand auf ihrer Oberfläche ausgeübt wird, als auch von jedem Gegenstand auf ihrer Oberfläche auf sie.
Wechselwirkungen treten aber nicht nur im makroskopischen sondern auch im mikroskopischen Raum auf. Anziehungskräfte zwischen Atomen und Molekülen sind ebenfalls Wechselwirkungen. Elektrische Ladungen üben nach dem Coulombschen Gesetz [5] eine Kraft aufeinander aus:

In dieser Formel bezeichnet $\overrightarrow{r_{ij}}$ den Verbindungsvektor zweier Teilchen i und j, mit Ladungen q_i und q_j. Wird nun die Kraft $\overrightarrow{F_{ji}}$ berechnet, die von Ladung q_j auf Ladung q_i wirkt, folgt daraus:

$$
\begin{aligned}
&\overrightarrow{F_{ji}} && (Formel2)\\
=&\frac{1}{4\pi\varepsilon_0}\frac{q_j \cdot q_i}{r^3} \cdot \overrightarrow{r_{ji}} && \overrightarrow{r_{ji}} = -\overrightarrow{r_{ij}}\\
=&\frac{1}{4\pi\varepsilon_0}\frac{q_i \cdot q_j}{r^3} \cdot -\overrightarrow{r_{ij}}\\
=&-\left(\frac{1}{4\pi\varepsilon_0}\frac{q_i \cdot q_j}{r^3} \cdot \overrightarrow{r_{ij}}\right)\\
=&-\overrightarrow{F_{ij}}
\end{aligned}
$$

Somit ist gezeigt, dass die elektrostatische Anziehung eine Wechselwirkung ist. Die Coulomb-Kraft, die Ladung q_i auf Ladung q_j ausübt, ist antiparallel

zur Kraft, die Ladung q_j auf Ladung q_i ausübt. Allgemein kann man Wechselwirkungen in der Form $\overrightarrow{F_{ij}} = -\frac{\partial u_{ij}(\overrightarrow{r_{ij}})}{\partial \overrightarrow{r_{ij}}}$ schreiben. Dabei beschreibt der Ausdruck $u_{ij}(r_{ij})$ ein Potenzial, das zwischen den beiden Körpern i und j herrscht. Ein Potenzial ist dabei reziprok proportional von einer n-ten Potenz des Abstandes der beiden Teilchen abhängig ($u_{ij} \sim \frac{1}{r_{ij}^n}$). Gilt nun, dass $n \leq d$ ist, wobei d die Dimension des betrachteten Raumes bezeichnet, so nennt man das Potenzial **langreichweitig**, andernfalls nennt man das Potenzial **kurzreichweitig**.
Der Unterschied ist im Folgenden für die Molekulardynamik von großer Bedeutung, da langreichweitige Potenziale sich durch das ganze betrachtete System ausbreiten und so Wechselwirkungskräfte zwischen allen Teilchen hervorrufen. Demhingegen rufen kurzreichweitige Wechselwirkungen außerhalb eines bestimmten Radius um ein Teilchen herum keinerlei Wechselwirkung mit anderen Teilchen hervor.

2.2 Molekulardynamik

Molekulardynamik ist nun eine Technik, mit der Systeme und ihre inneren Wechselwirkungen simuliert werden können. Hierbei müssen zunächst einige Anforderungen an das zu simuliernde System gestellt werden. Für dieses muss gelten, dass es räumlich abgegrenzt ist und Teilchen das System nicht verlassen können. Dies wird meist durch *periodische Randbedingungen* sicher gestellt. Periodische Randbedingungen bedeutet dabei, dass falls ein Teilchen durch Auswirkungen der Wechselwirkungskräfte innerhalb des Systems, dieses verlassen würde, es stattdessen auf der gegenüberliegenden Seite des Systemes wieder in dieses eintritt. Dieses Verhalten wird dadurch erzeugt, dass das simulierte System Phantom-Nachbaren bekommt, in denen die gleiche Teilchenkonfiguration vorliegt. Von diesen Phantom-Nachbaren gibt es in jeder Richtung unendlich viele, so dass zu keiner Zeit außerhalb des Systems leerer Raum vorliegt. Bei der Simulation müssen somit auch Wechselwirkungen zwischen Teilchen am Rand des betrachteten Gebietes und solchen, die in den Phantomnachbaren liegen würden berück-

sichtigt werden. Dies wird durch Verschiebung der Teilchen auf die jeweils andere Seite des Systemgebiets gelöst. Findet dann eine Verschiebung aus dem ursprünglichen Gebiet statt, ist das gleichbedeutend mit dem Eintritt eines Phantom-Nachbarteilchens, welches durch Verschiebung des austretenden Teilchens beschrieben wird. Alternativ kann das System in bestimmte Dimensionen auch begrenzt sein. Dann findet in dieser Richtung keine Simulation von Wechselwirkungskräften mit Phantom-Nachbarn statt. Man kann beide Randbedingungen im gleichen System einsetzen, um zum Beispiel des Transport von Blutkörperchen durch die Blutbahn zu simulieren. Hier würde man zwei Dimensionen beschränken und die dritte periodisch betrachten. Dieses System ist in Abbildung (2.1) dargestellt. Durch den Blutfluss werden Teilchen vorne aus dem System getragen und durch die periodische Randbedingung hinten wieder eingefügt.

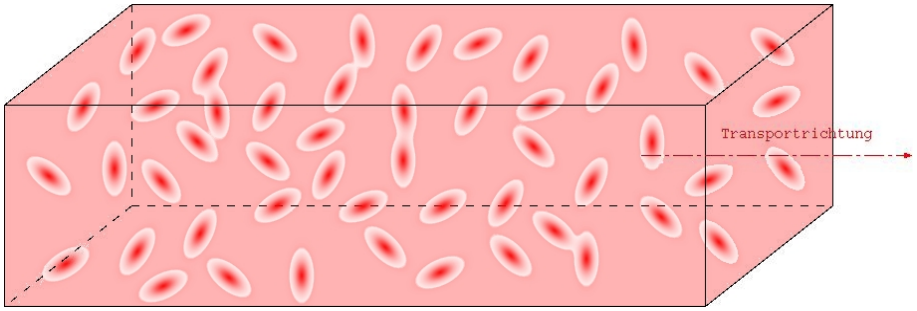

Abbildung 2.1: Skizzierte Blutbahn mit zwei nicht-periodischen und einer periodischen Randbedingung. Das Blut fließt in Transportrichtung und Teilchen, die aus dem vorderen Ende des simulierten Bereiches austreten, werden am hinteren Ende wieder eingefügt.

Die Simulation der Wechselwirkungskräfte erfolgt durch das Lösen von Bewegungsgleichungen, die sich aus den Wechselwirkungskräften zwischen den Teilchen des Systems ergeben. Diese Bewegungsgleichungen werden auf einem diskreten Zeitgitter gelöst, d.h. der kontinuierliche Zeitrahmen, der betrachtet wird, wird in Zeitabschnitte unterteilt und die Bewegungsgleichungen "Schritt für Schritt" gelöst. Nach dem Lösen der Bewegungsgleichungen für den aktuellen Schritt werden die Teilchen entsprechend der Lösung weiter-

bewegt.
Es gibt verschiedene Methoden, wie Teilchen im Raum bewegt werden. Bei Methoden mit konstantem Zeitschritt werden die klassischen Bewegungsgleichungen, gewöhnliche DGLs, mit finiten Differenzenverfahren gelöst, wobei der Zeitschritt nach Stabilitätskriterien des Lösers und der Dynamik im System bestimmt wird. Bei ereignisbasierten Methoden, bei denen Kontaktwechselwirkugnen zwischen Teilchen bestimmt werden, verändert sich der Zeitschritt stets und richtet sich nach dem Zeitintervall zwischen zwei aufeinanderfolgenden Kollisionen im System.
Ein großes Problem für die Molekulardynamik ist der zu betrachtende Aufwand, da dieser zunächst quadratisch von der Anzahl der Teilchen im System abhängig ist. Dies liegt daran, dass ohne weitere Kenntnisse des Systems davon ausgegangen werden muss, dass jedes Teilchen mit jedem Teilchen des Systemes wechselwirkt. Daraus folgt, dass in jedem Rechenschritt jedes Teilchen mit jedem anderen Teilchen verglichen werden muss. Selbst wenn man berücksichtigt, dass die Wechselwirkung symmetrisch ist, so halbiert man zwar den Rechenaufwand, bleibt insgesamt aber dennoch in der Klasse der Probleme mit quadratischem Aufwand.
An diesem Punkt wird klar, warum grade die langreichweitige Wechselwirkungen so problematisch für die Molekulardynamik sind. Während kurzreichweitge Wechselwirkungen im Wesentlichen nur Teilchen in einem gewissen Radius um jedes Teilchen beeinflußen, wirken die langreichweitigen auf das ganze System. Dieses Problem wird teilweise damit zu umgehen versucht, dass man einen gewissen Radius für Wechselwirkungen einführt, hinter dem die Wechselwirkung einfach gleich Null gesetzt wird, da sie ab dem Radius nur kleine Auswirkungen hat. Aus diesem Grund wird dieser Radius als **Cut-Off Radius** bezeichnet, da an dieser Grenze die Wechselwirkungen "abgeschnitten" werden.

2.3 Parallelisierte Molekulardynamik

Ein Kernbestandteil der Molekulardynamik ist das numerische Lösen von gekoppelten geöhnlichen Differentialgleichungen. Um die Koeffizienten in den Gleichungen zu bestimmen, müssen die Wechselwirkungen zwischen den Teilchen bestimmt werden. Diese Bestimmung der Wechselwirkungen, welche durch Kraftfelder beschrieben werden, lässt sich häufig gut und effizient parallelisieren. Speziell für kurzreichweitige Wechselwirkungen, die in der vorliegenden Arbeit betrachtet werden, gibt es effiziente Parallelisierungsansätze, wie z.B. die Gebietszerlegung. Allerdings können während der Simulation zwischen den Prozessoren Lastunterschiede auftreten, welche nach Möglichkeit ausgeglichen werden sollten, damit Prozessoren mit höherem Arbeitsvolumen entlastet werden. Dies ist ein wichtiger Aspekt, da die Prozessoren mit der größten Arbeit die Gesamtperformance und auch die Skalierbarkeit dominieren. Einen Lastausgleich kann man auf zwei verschiedene Möglichkeiten realisieren, entweder durch Verteilung der Teilchen oder als Gebietszerlegung: Bei der Teilchenverteilung werden die Teilchen im System auf die verschiedenen Prozesse aufgeteilt und so wird dafür zu gesorgt, dass jeder Prozess für die gleiche Anzahl von Teilchen die Wechselwirkungen bestimmen muss. Vorteil ist, dass jeder Prozess theoretisch die gleiche Rechenarbeit zu vollbringen hat. Allerdings muss für jedes Teilchen eine Liste geführt werden, mit welchen Teilchen es wechselwirkt. Dann muss in jedem Rechenschritt dafür gesorgt werden, dass alle miteinander wechselwirkenden Teilchen bei dem Prozess vorhanden sind, der die Wechselwirkungen berechnet. Nach Berechnung der Wechselwirkungen und der darauf basierenden Teilchenbewegung müssen gegebenfalls die Teilchenlisten aktualisiert werden, da durch die Bewegung Teilchen sich aus dem den entsprechenden Wechselwirkungsbereichen entfernt haben oder neu hinzugekommen sind. Diese Aktualisierungen können in hohem Kommunikationsaufwand resultieren.
Bei der Gebietszerlegung ist es das Ziel das System räumlich aufzuteilen und diese Raumgebiete mit den beinhalteten Teilchen auf die Prozesse aufzuteilen. Vorteil hier ist, dass durch die räumliche Verbundenheit Teilchen, die nah beieinander liegen und so vermutlich miteinander wechselwirken schon

auf dem gleichen Prozess vorhanden sind. Gegebenenfalls muss nur mit den direkten Nachbarn Teilchen ausgetauscht werden, um alle Wechselwirkungen berechnen zu können. Bei dieser Methode gibt es jedoch den Nachteil, dass die Gebiete ohne weiteres gleichmäßig aufgeteilt werden können, dies aber nicht bedeutet, dass der Rechenaufwand, bzw. die Anzahl der Teilchen gleichmäßig verteilt wird. So kann es sein, dass gerade bei stark inhomogenen Systemen, wie sie teilweise mit MP2C simuliert werden, durch eine gleichmäßige Gebietsaufteilung einige Prozesse viel, andere wenig Rechenarbeit leisten müssen.
Wie in der Einleitung schon erwähnt, wird in dieser Arbeit nur der zweite Fall behandelt. Aus diesem Grund werden im folgenden Abschnitt einige Verfahren vorgestellt, die Ungleichmäßigkeit der Lastverteilung angleichen sollen, um Wartezeiten zwischen den Prozessen zu minimieren und die Gesamtlaufzeit so zu optimieren.

2.4 Alternative Loadbalancing-Strategien

Zum Abschluß des Grundlagen-Kapitels werden hier nun alternative Strategien des Loadbalancings vorgestellt. Die Gründe warum diese Verfahren jeweils nicht verwendet wurden, werden ebenfalls kurz erläutert, bevor im nächsten Kapitel das eigentliche Verfahren dieser Arbeit erklärt wird.
Bei den folgenden Strategien zum Loadbalancing handelt es sich um die Methode mit raumfüllenden Kurven, wie sie z.B. bei der “Hashed-Oct-Tree”-Methode nach Warren und Salman [15] verwendet wird und um ein Verfahren, das am JSC im Rahmen der Masterarbeit von Bücker entwickelt wurde [3] und das auf der Verteilung von Blöcken von Teilchen, deren Größe über die Anzahl ihrer Wechselwirkungen und Position auf einer raumfüllenden Kurve bestimmt werden, basiert.

2.4.1 "Hashed-Oct-Tree"-Methode

Für die mathematischen Beweise der Methode, sowie die Fehleruntersuchungen zu dieser sei auf [15, 14] verwiesen. An dieser Stelle wird die Methode kurz anhand der Implementierung, wie sie in [7] durchgeführt wurde, beschrieben.

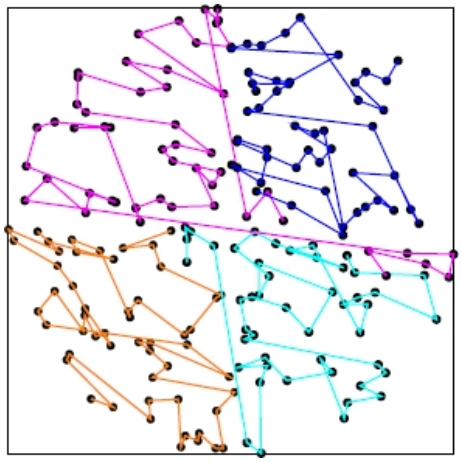

Abbildung 2.2: Beispiel einer zweidimensionalen Sortierung von Teilchen mit einer Morten-Kurve, auf die alle Teilchen abgebildet werden und danach auf vier Prozesse aufgeteilt werden. Dabei wird gleichfarbige Kurvenstücke jeweils auf einen Prozess abgebildet. Man sieht, dass die daraus resultierenden Gebiete nicht zusammenhängend sein müssen (violettes Kurvenstück) (Bildquelle: [7], S.3)

Zunächst werden die Koordinaten der Teilchen in eindeutige binäre Schlüssel konvertiert, die den jeweiligen Teilchen zusätzlich zugeordnet werden, z.B. 64-Bit Integerzahlen. Anhand dieser Schlüssel werden die Teilchen in ihrer dreidimensionalen Verteilung im System nun auf eine eindimensionale Kurve abgebildet, in der einfachsten Implementierung ist dies eine Z-Kurve [10]. Es sind aber auch andere raumfüllende Kurven, wie Hilbert-Kurven [9], möglich. Als Folge davon werden Teilchen, die im System in kurzer Entfernung zueinander liegen, oft hintereinander auf der Kurve abgebildet. Die Kurve wird im Folgenden dann so unter den verfügbaren Prozessen aufgeteilt, dass

allen Prozessen eine etwa gleichgroße Menge an Teilchen zugewiesen wird. In Abbildung (2.2) sieht man, dass allen Prozessen in etwa die gleiche Teilchenanzahl zugeordnet wurde, allerdings sind die Gebiete, in denen die Teilchen jeweils liegen, nicht notwendigerweise zusammenhängend.
Im fortlaufenden Programm wird die Kurve angepasst, sobald sich die Teilchen im System in den Zeitschritten, die simuliert werden, zu stark bewegt haben und der Verlauf der Kurve dadurch geändert wird. Dann wird die Kurve entsprechend neu aufgeteilt. Da somit immer alle zu behandelnden Teilchen überprüft werden müssen und bei der Änderung eines einzigen Teilchens gegebenenfalls die ganze Kurve angepasst werden muss, kann diese sehr aufwendig sein, falls global alle Teilchen auf eine einzige Kurve abgebildet werden. Bei der "Hashed-Oct-Tree"-Methode wird daher zunächst das Gesamtgebiet durch rekursive Bisektion der Kantenlängen solange unterteilt, bis sich in jedem Teilgebiet weniger Teilchen befinden, als eine zuvor vorgegebene Schranke bestimmt. Danach werden für die Einzelgebiete jeweils eigene Kurven gebildet, auf die die Teilchen aus dem jeweiligen Gebiet abgebildet werden. Anschließend werden dann Teilstücke aus allen Gebieten auf den Prozessen verteilt. Verändert sich nun die Position von Teilchen müssen nur die Kurven in den Teilgebieten angepasst werden, was deutlich weniger Aufwand ist, als das Neubestimmen der Kurve im Gesamtgebiet. Der Name der Methode stammt nun daher, dass jedes zu unterteilende Gebiet durch die Bisektion zunächst in acht gleichgroße Teilgebiete aufgeteilt wird, die gegebenenfalls nochmals geachtelt werden, bis schließlich alle Teilgebiete weniger als die gewünschte Maximalanzahl an Teilchen beinhaltet.
Für MP2C sollen die Gebiete, die auf die Prozesse verteilt werden jedoch räumlich zusammenhängend sein. Da außerdem die Informationen, die zur Neueinteilung der Gebiete notwendig sind, möglichst lokal sein sollen, d.h. im besten Fall nur von direkten Nachbaren angefordert werden müssen, eignet sich ein Verfahren, das raumfüllende Kurven zur Gebietsauteilung verwendet, im Rahmen dieser Arbeit nicht.

2.4.2 Strategie der Wechselwirkungsverteilung

Bei der anderen, alternativen Strategie [3] werden die Teilchen zunächst in Teilgebiete (Boxen) einsortiert. Diese Boxen werden dann mittels einer raumfüllenden Kurve durchnummeriert. Danach werden die Boxen im ersten Schritt so auf die Prozesse verteilt, dass auf jedem Prozessor möglichst die gleiche Teilchenanzahl liegt. Anschließend werden für die einzelnen Boxen die Anzahl der Wechselwirkungen, die innerhalb einer Box stattfinden, berechnet. Die Anzahl der Wechselwirkungen gewichtet die jeweilige Box gegenüber den anderen. Dann wird der Mittelwert der Anzahl an zu berechnenden Wechselwirkungen für jeden Prozess bestimmt. Um diesen herum wird ein Konfidenzbereich gelegt. Alle Prozesse, die mehr Wechselwirkungen zu berechnen haben, als der obere Rand des Konfidenzbereiches, geben solange Boxen an Prozesse, deren Anzahl an Wechselwirkungen unter dem Konfidenzbereich liegt, ab, bis für alle Prozesse die Anzahl an zu berechnenden Wechselwirkungen im Konfidenzbereich liegt. Da es hierbei vorkommen kann, dass Boxen mit zu vielen Wechselwirkungen eine solche Aufteilung verhindern, müssen diese Boxen gegebenenfalls aufgeteilt werden. Aufteilt wird durch Halbieren der Kantenlängen der Box, also Achtelung des Gebietes. Die neuen, kleineren Boxen werden dann verteilt, bis für alle Prozesse die Anzahl der zu berechnenden Wechselwirkungen in den Konfidenzbereich korrigiert werden kann.
Die Gründe, warum dieses Verfahren in dieser Arbeit nicht verwendet wird, sind ähnlich zu denen, warum das “Hashed-Oct-Tree”-Verfahren nicht verwendet wird. Da durch die Verteilung der Boxen auf die Prozesse die Gebiete, die ein Prozess verwaltet nicht räumlich zusammenhängend sind, ist dieses Verfahren nicht für diese Arbeit geeignet.

Kapitel 3

Verfahren zur Gebietsanpassung

In diesem Kapitel wird das Verfahren beschrieben, nach dem das Loadbalancing durchgeführt wird. Dazu ist es in zwei Abschnitte gegliedert. Im ersten Teil wird beschrieben, wie der Begriff "Arbeit" definiert ist und wie die "Arbeit" zwischen den Prozessen ausgeglichen werden soll, während im zweiten Teil beschrieben wird, wie sich diese Änderungen auf die benötigte Kommunikation zwischen den Prozessen, die die Gebiete abbilden, auswirken. Außerdem wird im zweiten Abschnitt aufgezeigt, welche Anpassungen im Code durch diese Veränderungen notwendig wurden.

3.1 Definition der Arbeit

Der Begriff der "Arbeit" W soll in der vorliegenden Darstellung für eine Größe stehen, die ein Maß für den Arbeitsaufwand ist, den ein Prozess mit seiner Gebietsverwaltung hat. Somit ist "Arbeit" keine fest definierte Größe, sondern stellt ein Maß für eine bestimmte Größe dar. Als mögliche "Arbeiten" wurden im Rahmen dieser Arbeit zwei Größen betrachtet:

- Anzahl der Teilchen in einem Gebiet.
- Zeit, die der Prozess zwischen zwei Loadbalancing-Schritten für die

Durchführung von Kommunikations- oder Interaktionsroutinen oder einer Kombination von beidem benötigt.

Die Anzahl der Teilchen bestimmt die Zeit, die ein Prozess benötigt, um einen Rechenschritt auf seinem Gebiet durchführen zu können. Je mehr Teilchen in einem Gebiet vorhanden sind, desto länger wird im Allgemeinen die Berechnung der Wechselwirkungen benötigen. Ist eine starke Schwankung in der Teilchenanzahl zwischen den Prozessen vorhanden, werden Wartezeiten zwischen dem Prozessor mit der minimalen Anzahl und der maximalen Anzahl an Teilchen unvermeidbar sein.
Alternativ kann man die verbrauchte Zeit zwischen zwei Loadbalancing-Schritten betrachten. Indirekt fließt in diese "Arbeit" zwar auch die Teilchenanzahl pro Prozessor ein, aber nicht nur diese. Da mehr Faktoren in die benötigte Zeit einfließen, als nur die Teilchenanzahl, sollte eine Anpassung, bei der die Zeit als "Arbeit" gewählt wird einen besseren Ausgleich zwischen den Prozessen liefern, als eine Anpassung, die nur über den Vergleich von Teilchenanzahlen arbeitet.
Sicherlich ist es also auch möglich noch andere Größen zu finden, die man sinnvoll als "Arbeit" einsetzen könnte. Im Rahmen dieser Arbeit wird sich allerdings auf die beiden beschriebenen Kenngrößen beschränkt. Allerdings wird durch die Art des hier beschriebenen Loadbalancing-Verfahrens die Anpassung auch immer zum Teil auf eine Anpassung der Teilchenanzahlen pro Prozesse hinauslaufen, da die Gebietsgrenzen angepasst und so im Allgemeinen Teilchen ebenfalls verschoben werden. Allerdings bedeutet das nicht, dass das Ergebnis des Verfahrens immer eine Gleichverteilung der Teilchenanzahl auf allen Prozessen ist, wie im Fall der Teilchenanzahl als "Arbeit". Der Grund dafür ist, dass z.B die Zeit, die für die Kommunikationsroutinen aufgewendet werden muss, nicht linear von der Teilchenanzahl abhängig ist.

3.2 Bestimmung der neuen Grenzen

Da sich Teilchen in der Simulation über Grenzen hinweg bewegen, oder gegebenenfalls schon eine inhomogene, d.h. nicht auf dem Gesamtgebiet gleichverteilte Anfangsverteilung der Teilchen vorliegt, ist es notwendig die Grenzen der Gebiete, die auf die Prozesse abgebildet werden, so anzupassen, dass eine möglichst homogene Verteilung der "Arbeit" auf den Gebieten entsteht. Dieses soll unabhängig von der verwendeten Definition der Arbeit gelten (s. 3.1).
Das Verfahren ist dreidimensional, wird aber aus der Verknüpfung von eindimensionalen Teilschritten aufgebaut. Dabei wird auf "Ebenen", "Säulen" und "Zellen" opereriert. Abbildung (3.1) zeigt, wie diese Teilgebiete definiert sind. Die elementarste Einteilung ist die Zelle, diese stellt ein Gebiet dar, das auf einen Prozessor abgebildet wird. Alle Zellen, deren Ausdehnung in y-Richtung identisch sind, werden zu einer Säule zusammengefasst. Säulen mit gleicher Ausdehnung in z-Richtung werden wiederum zu Ebenen zusammengefasst. In der Abbildung ist außerdem dargestellt, was es heißt, dass die Gebiete nicht miteinander "verzahnt" sind. Alle Ebenen liegen aufeinander, ohne dass eine Säule oder Zelle sich in ihrer z-Ausdehnung mit einer benachbarten Ebene überschneiden würde. Analoges gilt bezüglich der y-Ausdehnung der Zellen auch für die Säulen. Das Verfahren verknüpft eindimsionale Anpassungen der Arbeiten von Ebenen, Säulen und Zellen zu einer Gesamtanpassung aller Arbeiten in den Zellen. Dabei werden zunächst die Arbeiten der Ebenen, dann die der Säulen und abschließend die Arbeiten der Zellen angepasst.

An dieser Stelle nun noch eine Definition der im Folgenden benutzten Begriffe. Zunächst sei die "Arbeit" von hier an als W bezeichnet.Des Weiteren werde die Gebietsgrenze als x_i bezeichnet. Links von dieser Grenze liegt das Gebiet, das die Arbeit W_i beinhaltet ("unterer Nachbar"), während rechts von dieser Grenze das Gebiet mit der Arbeit W_{i+1} ("oberer Nachbar") angrenzt. Der untere Nachbar ist nach links durch die Grenze x_{i-1} beschränkt und der obere durch die Grenze x_{i+1} nach rechts. Alle Arbeiten zusammen lassen sich als Vektor $\overrightarrow{W}$ darstellen, wobei die i-te Komponente die Arbeit im i-ten Gebiet repräsentiert.

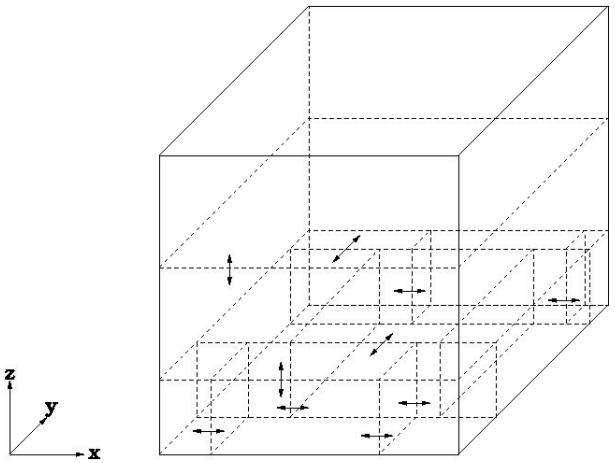

Abbildung 3.1: Räumliche Aufteilung des Gesamtgebietes
Das Gesamtgebiet wird in Teilgebiete unterteilt, deren Arbeit jeweils zueinander angepasst werden soll. Dabei stellen die Teilbereiche in z-Richtung die "Ebenen" dar. In den Ebenen liegen in y-Richtung die "Säulen" nebeneinander, in denen sich schließlich in x-Richtung die "Zellen" befinden. Jeweils eine Zelle wird auf einen Prozess abgebildet.

3.2.1 Beschreibung der Vorgehensweise

Bei dem nun beschriebenen Verfahren zum Loadbalancing handelt es sich, wie zuvor erwähnt, um ein Verfahren des Domain Decompositioning. Somit werden Gebiete angepasst, um so eine Anpassung der Lastverteilung durchführen zu können. Um die Grenzen der Gebiete anpassen zu können, müssen zunächst Informationen übermittelt werden. Ohne im Konkreten der Beschreibung der Berechnung vorzugreifen sei an dieser Stelle das Vorgehen zur Bestimmung der Grenzen skizziert, um anhand dieser Beschreibung die Kommunikation verständlich zu machen.
Der Grundgedanke des Verfahrens ist es, die Gebiete auf den Prozessen durch lokalen Vergleich nur mit benachbarten Prozessen anzupassen. Dabei sollen aber keine Verzahnungen entstehen, das bedeutet, dass alle Ebenen eine glatte Grenze zueinander haben, also keine Zelle einer Ebene in die benachbarte

'hineinreicht'. Analoges gilt für die Säulen innerhalb einzelner Ebenen.
Um diese Bedingung zu erfüllen wird zunächst die Arbeit W einer Ebene mit der nächst höheren, d.h. der Ebene, die an die obere Grenze der Ebene angrenzt verglichen. Dann wird die gemeinsame Grenze angepasst und das ganze Verfahren parallel für alle Ebenen durchgeführt. Ist dieser Vorgang in allen Ebenen abgeschlossen, so wird er analog für benachbarte Säulen in den einzelnen Ebenen und danach für benachbarte Zellen wiederholt. Damit wird außerdem die dreidimensionale Aufteilung in drei eindimensionale Aufteilungen unterteilt. Das Verfahren ist also keine wirkliche dreidimensionale Aufteilung, sondern eine Verknüpfung von eindimensionalen Aufteilungen. Durch die Anforderung, dass es keine Überlappungen geben soll, ist dies ohne weiteres möglich (s. Abbildung (3.2)). Die Abbildung zeigt eine Skizze einer zweidimensionalen Aufteilung. Dabei fällt auf, dass die Zellen einer Säule jeweils die gleichen y-Grenzen haben, aber in jeder Säule die Gebiete zu einander verschoben sind. Man kann die x-Aufteilung einer Säule also völlig unabhängig von den anderen Säulen anpassen, da diese keine Auswirkungen diesbezüglich aufeinander haben.
Allerdings ist zu beachten, dass die Annahme, die an dieses Verfahren gestellt wird, ist, dass das Loadbalancing-Verfahren schneller konvergiert, als die Zeitskala, auf der das System eine neue Imbalance entwickelt. Diese Annahme kann man als "quasi-statisch" bezeichnen. Für Systeme, die nicht zu stark aus dem thermischen Gleichgewicht bewegt wurden, ist diese Annahme meistens erfüllt.

3.2.2 Kommunikation der lokalen Arbeiten

Es gibt zwei mögliche Verfahren, um die benötigten Informationen zur Berechnung zu sammeln. Die erste Möglichkeit ist die Arbeiten einer Säule zunächst zu einem Prozess zu senden, um dann die Arbeit aller Säulen, die auf diesen 'Sammelprozessoren' liegen, zu einem Prozess zu schicken, der so die Arbeit der gesamten Ebene enthält. Wird dann die Arbeit aller Ebenen wieder zu einem Prozess kommuniziert, so kann dieser die neuen Ebenendicken ermitteln. Anschließend werden die Dicken an die Prozesse zurückvermittelt,

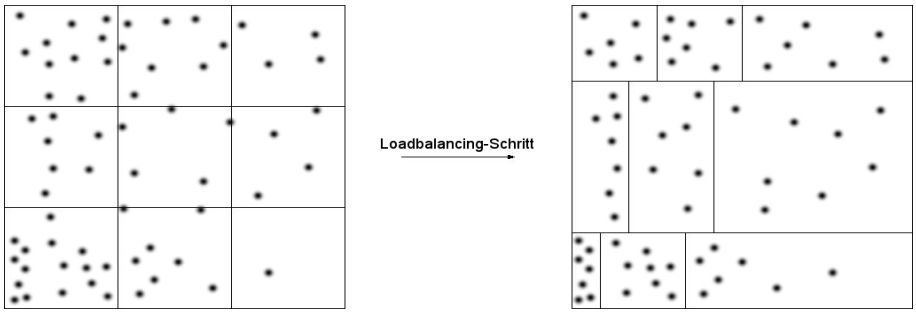

Abbildung 3.2: Verschiebung der Grenzen beim Loadbalancing-Schritt
Skizze zur Verschiebung der Grenzen bei Anwendundung des Loadbalancing-Verfahrens bei einem zwei-dimensionalen Fall unter Verwendung der Teilchenanzahl als Arbeit.

die Inhalte angepasst, der Vorgang bis zu den Säulenarbeiten wiederholt und am Schluß für die Zellen ebenfalls wiederholt. Dieses Schema hat den Makel, dass zwar wenig kommuniziert wird, dafür aber nur ein Prozess rechnet, während die anderen nichts tun.
Demhingegen werden bei der zweiten Methode die Arbeiten zunächst an alle Prozesse in der x-Dimension geschickt, dann die Arbeiten in die y-Dimension und schließlich in die z-Dimension. So haben alle Prozesse die benötigten Informationen um parallel die neuen Grenzen in z-Dimension zu bestimmen und danach ihren Inhalt auf die neuen Grenzen anzupassen. Dann werden die neuen Arbeiten bestimmt und die Kommunikation bis zur y-Dimension wiederholt. Sind die Dicken der Säulen bestimmt wird mit den direkten Nachbaren in x-Dimension die Arbeit ausgetauscht, so dass auch diese Grenzen parallel bestimmt werden können.
Da ein Hauptmerkmal der Arbeit das Reduzieren von Wartezeiten ist und das Programm auch auf großen Prozesszahlen gut skalieren soll, ist die zweite Methode vorzuziehen, da sich so, gerade bei großen Prozesszahlen, nicht ein Großteil der Prozesse sich im Wartezustand befinden.

Formel 3 (Arbeitsdichte über einem Gebiet)
$$\tilde{\rho}_i := \frac{W_i}{x_{i+1} - x_i}$$

3.2.3 Idee zur Verschiebung der Grenzen

Grundlegende Idee des Verfahrens ist es, die lokalen Ungleichheiten in der Arbeit zwischen benachbarten Gebieten auszugleichen. Dafür wird zunächst eine Differenz zwischen den Arbeiten gebildet und ein Anteil dieser Differenz vom Gebiet mit der größeren Arbeit in das Gebiet mit der kleineren Arbeit verschoben. Das Verfahren ist, wie zuvor erwähnt (3.2.1) eindimensional und soll für alle Gebiete in einem Schritt gleichzeitig ausgeführt werden. Daher muss darauf geachtet werden, Grenzen nicht zu stark zu verschieben, da durch die Verschiebungen sonst Gebiete mit negativer Breite entstehen könnten, d.h. Gebiete, bei denen die obere Grenze kleiner als die untere Grenze wäre. Um dies zu vermeiden, wird eben nicht die gesamte Arbeitsdifferenz vom Prozessor mit der größeren Arbeit übertragen, sondern nur ein Anteil davon. Wie groß dieser Anteil ist, muss entsprechend der Verhältnisse der Nachbarschaftsgebiete zueinander festgelegt werden.
Um eine Annäherung für den Zusammenhang zwischen Gebietsbreite und darauf enthaltener Arbeit zu haben, wird im Rahmen dieser Arbeit davon ausgegangen, dass die Arbeit innerhalb eines Gebietes gleichverteilt über dieses ist. Hintergrund hierfür ist, dass bei der Definition der Arbeit als Teilchenanzahl auf dem Prozess zwar noch eine empirische Verteilung über Histogramme erzeugt werden kann, aber z.B. bei der Verwendung der benötigten Zeit zwischen zwei Loadbalancing-Schritten, hierfür keine einfache empirische Verteilung über das Gebiet angelegt werden kann. Als geschätzte Dichte der Arbeit über einer Gebietsbreite erhält man dann Formel (3). Dabei beschreibt $\tilde{\rho}$ die geschätzte Dichte, W_i die Arbeit auf dem betrachteten Gebiet und $x_{i+1} - x_i$ die Breite des betrachteten Gebietes. Was nun zu zeigen ist, ist ob ein Vektor von Arbeiten durch iterative Anwendung eines Operators so angepasst werden kann, dass er gegen den Vektor von Arbeiten konver-

giert, dessen Arbeiten alle identisch mit dem Mittelwert aller Arbeiten des Startvektors sind. Im zweiten Schritt ist dann, falls ein solcher Operator exisitert, dieser so anzupassen, dass die Gebiete jeweils so korrigiert werden, dass möglichst exakt die Arbeit, die nach dem ersten Schema zu übertragen wäre, übertragen wird. Abschließend muss für diesen angepassten Operator geprüft werden, ob und in welchem Rahmen dieser stabil ist, d.h. nicht zu Gebieten mit negativer Breite führt.
Bei all diesen Überlegung ist von der Voraussetzung auszugehen, dass die Geschwindigkeit der Teilchen so gering ist, dass sie sehr viel langsamer sind, als die Grenzanpassung. Es exisistiert also durch Teilchenbewegungen kein der Gebietsanpassung entgegengesetzter Teilchenstrom. Dieser Zustand wird als "quasi-statisch" bezeichnet.

3.2.4 Globaler Arbeitsausgleich durch anteilige Übertragung von lokalen Arbeitsdifferenzen

Zunächst wird nun geprüft, ob man eine globale Anpassung erreichen kann, indem man Anteile von lokalen Arbeitsdifferenzen als Anpassung verwendet. Abbildung (3.3) zeigt eine Skizze, anhand der die Bezeichnungen im Folgenden illustriert werden. Außerdem seien mit d_{min} und d_{max} die Breite der Gebiete bezeichnet, die betrachtet werden, wobei d_{min} dabei die Breite des kleineren Gebietes und d_{max} die Breite des größeren Gebietes bezeichnet. Betrachtet wird immer ein beliebiges Gebiet mit innerer Grenze x_i und Nachbargrenzen x_{i-1} und x_{i+1}. Somit sind die betrachteten Arbeiten jeweils W_i und W_{i+1}. $\frac{1}{\gamma}$ ist schließlich der Anteil der Arbeitsdifferenz, der übertragen werden soll, mit $\gamma \in \mathbb{R}^+$

Formel (4) gibt nun eine Beschreibung für die zu übertragende Arbeit an. Dabei sorgt das Vorzeichen dafür, dass die Arbeit immer von dem Gebiet, das die größere Arbeit beinhaltet, in das andere Gebiet übertragen wird. Ein negativer Arbeitsübertrag bedeutet dabei das Empfangen von Arbeit des Nachbargebietes, während ein positiver Übertrag das Senden zum Nachbargebiet bedeuetet. Durch diese Formulierung werden Fallunterscheidungen

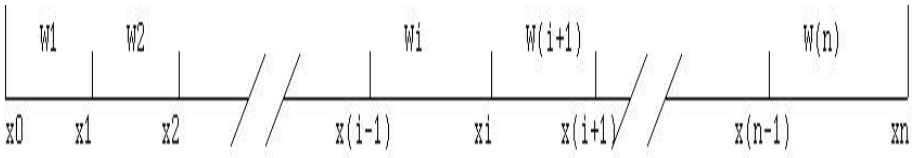

Abbildung 3.3: Aufteilung des Gebietes in Teilgebiete mit jeweiliger Arbeit und ihre Benennung

Formel 4 (Übertragene Arbeitsdifferenz)

$$\Delta_W(i, i+1; \gamma)^{\{k\}} = \frac{W_i^{\{k\}} - W_{i+1}^{\{k\}}}{\gamma}$$

überflüssig. Wird dieses Übertragen nun für alle Gebiete durchgeführt, erhält man einen Matrixoperator, den man auf den Vektor der Arbeiten anwenden kann. Dieser Operator hat Tridiagonalform und ist in Formel (5) dargestellt. Zur Prüfung, ob das Iterationsverfahren konvergiert, müssen die Eigenwerte λ_i dieser Matrix betrachtet werden. Diese sind in Formel (6) angegeben. Dabei ist zu erkennen, dass die Eigenwerte für beliebige Dimension der Matrix nach oben durch Eins beschränkt sind, d.h. $\max_i \lambda_i = 1$. Weiterhin ist zu sehen, dass der Eigenwert $\lambda_i = 1$ die Vielfachheit Eins hat, d.h genau ein Mal auftritt. Was nun zu zeigen bleibt, ist dass bei wiederholter Anwendung des Matrixoperators auf den Arbeitsvektor das Ergebnis gegen den Vektor konvergiert, dessen Komponenten alle identisch dem globalen Mittelwert der Arbeit sind.

Der aktuelle Zustand der Arbeiten nach k Anwendungen des Matrixoperators auf den Arbeitsvektor $\vec{W}^{\{0\}}$ wird als $\vec{W}^{\{k\}}$ bezeichnet. Dabei kann der Startvektor $\vec{W}^{\{0\}}$ als Linearkombination der normierten Eigenvektoren $\vec{e}_i$ des Matrixoperators dargestellt werden, d.h. $\sum_{i=1}^{n} \alpha_i \vec{e}_i$. Bei wiederholter Anwendung von $A_W(\gamma) \in \mathbb{R}^{n \times n}$ auf den Startvektor der Arbeiten $\vec{W}^{\{0\}} \in \mathbb{R}^n$

ergibt sich somit:

$$\begin{aligned} & \overrightarrow{W}^{\{l\}} \\ = \; & (A_W(\gamma))^l \cdot \overrightarrow{W}^{\{0\}} \\ = \; & (A_W(\gamma))^l \cdot \sum_{i=1,..,n} (\alpha_i \cdot \overrightarrow{e_i}) \\ = \; & \sum_{i=1,..,n} (\alpha_i \cdot \lambda_i^l \cdot e_i) \underset{l \to \infty}{\to} \alpha_1 \cdot \overrightarrow{e_1} \end{aligned}$$

Um den Koeffizienten α_1 zu bestimmen, projiziert man den Startvektor der Arbeiten auf den normierten Eigenvektor $\overrightarrow{e_1} = \frac{1}{\sqrt{n}} \cdot (1, \ldots, 1)^T$. Damit erhält man:

$$\begin{aligned} & \alpha_1 \cdot \overrightarrow{e_1} \\ = \; & \left\langle \overrightarrow{e_1}, \overrightarrow{W}^{\{0\}} \right\rangle \cdot \overrightarrow{e_1} \\ = \; & \frac{1}{\sqrt{n}} \cdot \sum_{i=1}^{n} W_i^{\{0\}} \cdot \frac{1}{\sqrt{n}} (1, \ldots, 1)^T \\ = \; & \frac{1}{n} \cdot \sum_{i=1}^{n} W_i^{\{0\}} \cdot (1, \ldots, 1)^T \end{aligned}$$

Bei wiederholter Anwendung des Matrixoperators auf den Startvektor konvergiert das Ergebnis somit gegen den Vektor, der als Komponenten an jeder Position den globalen Mittelwert der Arbeiten enthält. Damit ist gezeigt, dass das Verfahren für die Übertragung von beliebig kleinen Anteilen an Arbeit gegen das gewünschte Ziel, einen globalen Ausgleich, konvergiert.

3.2.5 Gitterpunktverschiebung anhand der Übertragung von anteiligen Arbeitsdifferenzen

Die Arbeit lässt sich nach Abschnitt (3.2.4) durch lokale Anpassungen der lokalen Arbeiten an den globalen Mittelwert aller Arbeiten anpassen. Nun bleibt zu zeigen, dass man daraus ein Verfahren zur Verschiebung der Grenzen herleiten kann. Zunächst wird eine Gitterpunktdichte um den inneren

Formel 5 (Schema zur Arbeitsangleichung)

$$W_i^{\{k+1\}} = \begin{cases} \frac{\gamma-1}{\gamma} \cdot W_i^{\{k\}} + \frac{1}{\gamma} \cdot W_{i+1}^{\{k\}} & i = 1 \\ \frac{1}{\gamma} \cdot W_{i-1}^{\{k\}} + \frac{\gamma-2}{\gamma} \cdot W_i^{\{k\}} + \frac{1}{\gamma} W_{i+1}^{\{k\}} & i \in [2, \ldots, n-1] \\ \frac{\gamma-1}{\gamma} \cdot W_i^{\{k\}} + \frac{1}{\gamma} \cdot W_{i-1}^{\{k\}} & i = n \end{cases}$$

daraus folgt:

$$\overrightarrow{W}^{\{k+1\}} = A_W(\gamma) \cdot W^{\{k\}}$$

mit

$$A_W(\gamma) := \begin{pmatrix} \frac{\gamma-1}{\gamma} & \frac{1}{\gamma} & 0 & \cdots & \cdots & \cdots & 0 \\ \frac{1}{\gamma} & \frac{\gamma-2}{\gamma} & \frac{1}{\gamma} & \ddots & & & \vdots \\ 0 & \frac{1}{\gamma} & \frac{\gamma-2}{\gamma} & \frac{1}{\gamma} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \frac{1}{\gamma} & \frac{\gamma-2}{\gamma} & \frac{1}{\gamma} & 0 \\ \vdots & & & \ddots & \frac{1}{\gamma} & \frac{\gamma-2}{\gamma} & \frac{1}{\gamma} \\ 0 & \cdots & \cdots & \cdots & 0 & \frac{1}{\gamma} & \frac{\gamma-1}{\gamma} \end{pmatrix}$$

$$W^{\{k\}} = \begin{pmatrix} W_1^{\{k\}} \\ W_2^{\{k\}} \\ \vdots \\ W_n^{\{k\}} \end{pmatrix}$$

Formel 6 (Eigenwerte der Matrix des Arbeitsschemas)
$\lambda_k = \frac{\gamma + 2 - 2 \cdot \cos\left((k-1) \cdot \pi \cdot \frac{1}{n}\right)}{\gamma}, i = 1, .., n$

Formel 7 (Gitterpunktdichte)
$\rho_{i+\frac{1}{2}} = \frac{W_i + W_{i+1}}{x_{i+1} - x_{i-1}}$

Gitterpunkt herum bestimmt. Diese Dichte ist ein Hilfsmittel, um Fallunterscheidungen zu vermeiden. Betrachtet man nur die Dichten der lokal angenommenen Gleichverteilungen der Arbeitsverteilung über den Gebieten, so befindet sich genau am innneren Gitterpunkt eine Sprungstelle zwischen den jeweiligen Dichten, da diese in der Regel nicht gleich sind. Um diese Unstetigkeitsstelle am inneren Gitterpunkt zu beheben, wird die Gitterpunktdichte $\rho_{i+\frac{1}{2}}$ wie in Formel (7) definiert, wobei die wahre Verteilung der Arbeit durch eine Gleichverteilung über beide Nachbargebiete des inneren Gitterpunktes approximiert wird. Wie zuvor gezeigt, lässt sich die Arbeit durch Übertragen von Anteilen der lokalen Arbeitsunterschiede auf den jeweiligen Prozessor, der weniger Arbeit beinhaltet, global ausgleichen. Benutzt man die Gitterpunktdichte nun, um eine Breite zu bestimmen, in der nach der Gleichverteilungsannahme die zu übertragende Arbeit (nach Formel (4)) liegt, so erhält man als Breite $\Delta_x(i, i+1; \gamma)$:

$$\begin{aligned}
\Delta_x(i, i+1; \gamma) =& \frac{\Delta_W(i, i+1; \gamma)^{\{k\}}}{\rho_{i+\frac{1}{2}}} \\
=& \frac{\frac{W_i - W_{i+1}}{\gamma}}{\frac{W_i + W_{i+1}}{x_{i+1} - x_{i-1}}} \\
=& \frac{W_i - W_{i+1}}{\gamma \cdot (W_i + W_{i+1})} \cdot (x_{i+1} - x_{i-1}) \qquad (3.1)
\end{aligned}$$

Mit $\Delta_x(i, i+1; \gamma)$ hat man nun eine Gebietsbreite, die zwischen den Gebieten verschoben werden muss, um die Arbeiten anzugleichen. Wird dieses Gebiet so gewählt, dass es am inneren Gitterpunkt liegt, so erhält eine Rechenvorschrift für die Verschiebung des inneren Gitterpunktes. Dieses Gebiet werde bezeichnet mit

$$\delta_x(i, i+1; \gamma) := \{x | x \in [\min(x_i, x_i + \Delta_x(i, i+1; \gamma)), \max(x_i, x_i + \Delta_x(i, i+1; \gamma))]\}$$

Die Verschiebung des inneren Gitterpunktes um die Breite $\Delta_x(i, i+1; \gamma)$ ist nichts anderes, als die Übertragung des Gebietes dieser Breite von einem Nachbarprozessor auf den anderen. Auch hier lässt sich für alle inneren Gitterpunkte eine Iterationsvorschrift formulieren, die schließlich als Matrixoperator auf den Vektor der Gitterpunkte angewandt werden kann (Formel (8)). Dieses Schema ist allerdings im Gegensatz zum Arbeitsschema (Formel (5)) nicht stationär, d.h. die Koeffizienten des Matrixoperators müssen in jedem Iterationsschritt neu bestimmt werden. Dies ist u.a. deswegen notwendig, weil die Arbeiten, die tatsächlich übertragen werden von den geschätzten Werten abweichen, da die Arbeit im Allgemeinen nicht homogen innerhalb der Gebiete verteilt ist.

3.2.6 Stabilität des Verfahrens

An das Verfahren müssen Stabilitätsanforderungen gestellt werden. Als stabil wird das Verfahren bezeichnet, wenn alle Gebietbreiten positiv sind, d.h. sich während des Iterationsverfahrens keine Gitterpunkte überkreuzen (s. Abschnitt 3.2.3). Um das zu verhindern wird die übertragbare Gebietsbreite nach $\frac{d_{min}}{2}$ beschränkt. Damit wird sichergestellt, dass selbst wenn ein Gebiet Arbeit an beide Nachbarn abgibt, es dennoch nicht verschwindet, oder die Grenzen sich überkreuzen. Die folgende Ungleichung liefert eine Abschätzung nach unten für γ, d.h. den größten Anteil an Arbeit, der übertragen werden darf, damit das Verfahren im aktuellen Schritt stabil bleibt. Da die folgenden

Formel 8 (Schema zur Grenzverschiebung)

$$x_i^{\{k+1\}} = \begin{cases} x_0^{\{k\}} & i = 0 \\ x_i^{\{k\}} - \dfrac{W_i^{\{k\}} - W_{i+1}^{\{k\}}}{\gamma\left(W_i^{\{k\}} + W_{i+1}^{\{k\}}\right)} \cdot \left(x_{i-1}^{\{k\}} - x_{i+1}^{\{k\}}\right) & i \in [1, n-1] \\ x_n^{\{k\}} & i = n \end{cases}$$

daraus folgt:

$$x_i^{\{k+1\}} = A_x^{\{k\}} \cdot x^{\{k\}}$$

mit

$$\widetilde{W_i^{\{k\}}} := \frac{W_i^{\{k\}} - W_{i+1}^{\{k\}}}{\gamma\left(W_i^{\{k\}} + W_{i+1}^{\{k\}}\right)}$$

sowie

$$A_x^{\{k+1\}} := \begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ \widetilde{W_1^{\{k\}}} & 1 & -\widetilde{W_1^{\{k\}}} & \ddots & & & \vdots \\ 0 & \widetilde{W_2^{\{k\}}} & 1 & -\widetilde{W_2^{\{k\}}} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \widetilde{W_{n-2}^{\{k\}}} & 1 & -\widetilde{W_{n-2}^{\{k\}}} & 0 \\ \vdots & & & \ddots & \widetilde{W_{n-1}^{\{k\}}} & 1 & -\widetilde{W_{n-1}^{\{k\}}} \\ 0 & \cdots & \cdots & \cdots & 0 & 0 & 1 \end{pmatrix}$$

und

$$x^{\{k\}} = \begin{pmatrix} x_0^{\{k\}} \\ x_1^{\{k\}} \\ \vdots \\ x_n^{\{k\}} \end{pmatrix}$$

Betrachtungen alle unabhängig vom Iterationsschritt k sind, wird in diesem Abschnitt zur Übersichtlichkeit auf die entsprechende Kennzeichnung verzichtet.

$$\begin{aligned} |\Delta_x(i, i+1; \gamma)| &< \frac{d_{min}}{2} \\ \Leftrightarrow \frac{1}{\gamma} \frac{|W_i - W_{i+1}|}{W_i + W_{i+1}} (x_{i+1} - x_{i-1}) &< \frac{d_{min}}{2} \\ \Leftrightarrow \frac{|W_i - W_{i+1}|}{W_i + W_{i+1}} &< \frac{\gamma}{2} \frac{d_{min}}{(x_{i+1} - x_{i-1})} \\ \Leftrightarrow \frac{|W_i - W_{i+1}|}{W_i + W_{i+1}} &< \frac{\gamma}{2} \frac{d_{min}}{d_{min} + d_{max}} \\ \Leftrightarrow \frac{W_i + W_{i+1}}{|W_i - W_{i+1}|} &> \frac{2}{\gamma} + \frac{2}{\gamma} \cdot \frac{d_{max}}{d_{min}} \end{aligned} \tag{3.2}$$

Diese Ungleichung gibt ein Kriterum an, in welchem Verhältnis die Arbeiten der beiden Gebiete zu einander stehen dürfen, im Vergleich zu den Breiten der beiden Gebiete. Betrachtet man die drei folgenden Sonderfälle, so erhält man Aussagen darüber, wie γ zu wählen ist, damit das Verfahren nach obiger Definition stabil ist. Diese drei Sonderfälle werden in den folgenden Abschnitten behandelt. Mit den Ergebnissen dieser Betrachtungen lässt sich schließlich eine Aussage zu Konvergenz der Verfahren im Rahmen der in MP2C simulierten Systeme treffen.

$\underline{W_i = W_{i+1}}$**:** Um zu überprüfen, ob das Verfahren bei gleichen Arbeiten stabil ist, werde in die Gleichung (3.1) sowohl für W_i als auch für W_{i+1} die Arbeit $\bar{W}$ eingesetzt. Somit ergibt sich:

$$\frac{\bar{W} - \bar{W}}{\gamma \cdot (\bar{W} + \bar{W})} \cdot (x_{i+1} - x_{i-1}) = 0$$

Die zu übertragende Gebietsbreite verschwindet. Damit ist gezeigt, dass das Verfahren, wenn es auf einen ausgeglichenen Arbeitsvektor angewandt wird, keine Verschiebungen durchführen wird. Außerdem ist gezeigt, dass diese Eigenschaft für alle γ gegeben ist, so dass die Größe des Anteils der übertragenen Arbeit an der Arbeitsdifferenz keinen Einfluß hat. Als Sonderfall ist

hier der Fall zu betrachten, dass beide Gebiete keine Arbeit beinhalten. In diesem Fall ist der Quotient $\widetilde{W} := \frac{W_i - W_{i+1}}{\gamma \cdot (W_i + W_{i+1})}$ undefiniert. Um diesen Fall abzufangen, werde definiert:

$$\widetilde{W} = \begin{cases} 0 & W_i = W_{i+1} = 0 \\ \widetilde{W} & sonst \end{cases}$$

Mit dieser Definition wird auch für den Fall, dass beide Arbeiten Null sind, die gemeinsame Gebietsgrenze nicht verschoben.

$\underline{d_{min} = d_{max}}$**:** Durch die Gleichheit der Gebietsbreiten vereinfacht sich die Ungleichung (3.2) zu $\frac{W_i + W_{i+1}}{|W_i - W_{i+1}|} < \frac{4}{\gamma}$. Diese vereinfachte Ungleichung kann man einerseits nach γ und andererseits zu einer der Arbeiten auflösen und erhält so zwei Aussagen über das Verfahren bei diesem Sonderfall.
Die erste Aussage bezüglich γ ist, dass $\gamma > 4 \cdot \frac{|W_i - W_{i+1}|}{W_i + W_{i+1}}$ sein muss, bei gleichen Gebietsgrenzen, um ein stabiles Verfahren zu garantieren. Die rechte Seite kann man nach oben durch 4 abschätzen. Damit muss zur Verhinderung von sich überkreuzenden oder überlagernden Gitterpunkten $\gamma > 4$ gewählt werden. Bei $\gamma = 4$ erhält man den Sonderfall, bei dem Gitterpunkte aufeinander abgebildet werden können und es möglich wird, dass Gebiete so eine Breite von $d = 0$ erhielten, was ausgeschlossen werden soll.
Löst man die vereinfachte Ungleichung nach den Arbeiten auf, erhält man:

$$\begin{aligned} \frac{W_i + W_{i+1}}{|W_i - W_{i+1}|} &> \frac{4}{\gamma} \\ \Leftrightarrow \frac{W_i + W_{i+1}}{W_i - W_{i+1}} > \frac{4}{\gamma} \quad &\vee \quad \frac{W_i + W_{i+1}}{W_{i+1} - W_i} > \frac{4}{\gamma} \\ \Leftrightarrow (1 + \frac{4}{\gamma}) W_{i+1} > (-\frac{4}{\gamma} - 1) W_i \quad &\vee \quad (1 + \frac{4}{\gamma}) W_{i+1} > (\frac{4}{\gamma} - 1) W_i \\ \Leftrightarrow W_i > -W_{i+1} \quad &\vee \quad W_i > \frac{\frac{4}{\gamma} - 1}{1 + \frac{4}{\gamma}} W_{i+1} \end{aligned}$$

Formel 9 (Stabilitätskriterium)
$\gamma > 2 \cdot (1 + \frac{d_{max}}{d_{min}})$

Da für den Vorfaktor $\frac{\frac{4}{\gamma} - 1}{1 + \frac{4}{\gamma}}$ gilt, dass $\lim_{\gamma \to \infty} \frac{\frac{4}{\gamma} - 1}{1 + \frac{4}{\gamma}} = -1$ und $\forall_{\gamma > 4} \frac{\frac{4}{\gamma} - 1}{1 + \frac{4}{\gamma}} \in (-1, 0)$ ist, lassen sich Aussagen über die möglichen Arbeitsverhältnisse zwischen den Gebieten bei gleicher Gebietsbreite treffen. Legt man die Aussagen bezüglich γ zu Grunde, dass $\gamma > 4$ sein, damit die Stabilität immer gegeben ist, so kann man bei einem äquidistanten Gitter beliebige Arbeiten auf den Gebieten verteilen. Beliebige Arbeiten deswegen, weil die Arbeiten alle per Definition positiv oder Null sein müssen. Einziger Ausnahmefall ist, falls beide Gebiete keine Arbeit beinhalten. Dann greift allerdings die Definition aus dem vorherigen Abschnitt (3.2.6) und das Verfahren ist wieder stabil, da keine Gebietsverschiebung stattfindet.

$\underline{W_i = 0, W_{i+1} \neq 0}$**:** Als letzter Sonderfall wird der Fall betrachtet, dass genau ein Gebiet ohne Arbeit sei. O.B.d.A sei dies das rechte Gebiet, also $W_{i+1} = 0$. Eingesetzt in Ungleichung (3.2) ergibt sich damit:

$$\begin{aligned} \frac{W_i}{W_i} &> \frac{2}{\gamma} + \frac{2}{\gamma}\frac{d_{max}}{d_{min}} \\ \Leftrightarrow \gamma &> 2 \cdot (1 + \frac{d_{max}}{d_{min}}) \end{aligned} \tag{3.3}$$

Mit Ungleichung (3.3) hat man nun ein Kriterum, mit dem man eine Untergrenze für γ angeben kann. Da der betrachtete Sonderfall für die linke Seite der Ungleichung eine Untergrenze liefert, kann man für beliebige Arbeitsverhältnisse γ so nach unten abschätzen. Die rechte Seite ist unten ihrerseits durch Vier abschätzbar, da nach Definition $d_{max} \geq d_{min}$ ist. Somit ist für durch Formel (9) ein Kriterium für den größtmöglich übertragbaren Arbeitsanteil in Abhänigkeit vom aktuellen Gebietsverhältnis gegeben. Die Verteilung der Arbeit auf die Gebiete ist dabei irrelevant.

3.2.7 Überlegungen zur Konvergenz des Verfahrens

Bedeutung der Sonderfälle

Als abschließender Punkt muss nun betrachtet werden, ob das Verfahren in der Form auch noch dafür sorgen kann, dass die Arbeiten ausgeglichen werden. Zur Beantwortung dieses Punktes ist die Vorraussetzung notwendig, dass das betrachtete System quasi-statisch ist. Wird mit einem äquidistanten Gitter begonnen, so ist die Verteilung der Arbeit auf die Gebiete nach Abschnitt 3.2.6 unerheblich. Im nächsten Schritt werden die Gebiete angepasst und über das Stabilitätskriterum (Formel 9) lässt sich ein neues, kleinstes γ bestimmen, mit dem die Arbeiten angepasst werden sollen. Nach Abschnitt 3.2.4 konvergiert die Arbeit für beliebige $\gamma > 4$. Somit kann γ beliebig groß werden, d.h. die Anteile der übertragenen Arbeit in jedem Schritt beliebig klein. Da das System, dessen Gitterpunkte angepasst werden, als quasi-statisch angenommen wird, ist die Arbeitsverschiebung, die durch system-inhärente Ströme erfolgt, vernachlässigbar. Sind die Gebiete schließlich so angepasst, dass die Arbeit in allen Gebieten gleich ist, so werden nach Abschnitt 3.2.6 keine Grenzen mehr verschoben und der gewünschte Endzustand ist erreicht. Die Ungenauigkeiten, die durch das Anpassen der Gebietsgrenzen auf Grund der Gleichverteilungsannahme für die lokale Arbeit, gemacht werden, werden dadurch abgefangen, dass die Koeffizenten der Matrix des Gebietanpassungsoperators in jedem Schritt neu bestimmt werden. Somit wird für den Folgeschritt die wirklich übertragene Arbeit zu Grunde gelegt und nicht die Arbeit, die aus der Gleichverteilungsannahme resultiert. Somit wird verhindert, dass ein Gebiet immer weiter Arbeit an Nachbarn abgibt, obwohl es eigentlich schon weniger Arbeit beinhaltet als diese.
Im Rahmen der Implementierung in MP2C kann für viele Simulations-Szenarien von quasi-statischen Systemen ausgegangen werden. Daher kann das Verfahren im Bereich von MP2C als konvergent betrachtet werden. Wie sich das Verfahren bei schnelleren Teilchenbewegungen verhält, muss noch überprüft werden. Mathematisch hieße das, in jedem Schritt eine Verschiebung der Arbeiten zu berücksichtigen, die nicht von der Ungenauigkeit durch die

Gleichverteilungsannahme begründet liegt.

Bedingungen an die Dichte der Arbeitsverteilung für eine Kontraktion

Die größte Einschränkung für die Konvergenz des Verfahrens liefern lokale Dichteunterschiede. Tauchen an einigen Stellen im Gebiet sehr starke Spitzen in der Dichte auf, d.h. liegt viel Arbeit auf kleinem Gebiet, so kann beim Übertragen eines Gebietes viel mehr Arbeit übertragen werden, als beabsichtigt. Durch die Gleichverteilungsannahme auf der Gesamtbreite beider Nachbargebiete werden solche lokalen Dichtespitzen aber verdeckt. Um nun ein Kriterium zu finden, mit dem an die maximale Größe dieser lokalen Dichtespitzen abschätzen kann, wird an dieser Stelle zunächst die Größe $\delta W_i^{\{k\}}$ eingeführt, die die tatsächlich im Schritt k übertragende Arbeit am Gitterpunkt x_i beschreibt. Es gilt, dass

$$\delta W_i^{\{k\}} := \int\limits_{\delta_x(i,i+1;\gamma)} \rho(x)dx \tag{3.4}$$

und dass

$$W_i^{\{k+1\}} = W_i^{\{k\}} + \delta W_i^{\{k\}} \tag{3.5}$$

ist. Dabei fließt mit $\rho(x)$ die wahre Dichte der Arbeitsverteilung ein. Um nun eine Schranke für $\rho(t)$ zu finden, untersucht man, welche Bedingung $\rho(x)$ erfüllen muss, damit gilt:

$$|W_{i+1}^{\{k+1\}} - W_i^{\{k+1\}}| < |W_{i+1}^{\{k\}} - W_i^{\{k\}}| \tag{3.6}$$

Dabei soll ohne Beschränkung der Allgemeinheit gelten, dass $W_{i+1}^{\{k\}} > W_i^{\{k\}}$ ist. Das gilt, da für den Fall $W_{i+1}^{\{k\}} < W_i^{\{k\}}$ die Rechnung analog zur folgenden Rechnung durchgeführt werden kann, nur mit vertauschten Indizes. Mit der Einschränkung auf diesen Fall entfallen in der Folge die Beträge um die Differenzen. Um nun Aussagen über die erlaubten Dichtespitzen treffen zu können, müssen zwei Fälle betrachtet werden, die auftreten können und

im Folgenden behandelt werden. Der dritte Fall, der auftreten könnte, muss nicht gesondert behandelt werden, da dies der Fall ist, in dem die Arbeiten nach der Gebietsübertragung ausgeglichen sind. Da damit das gewünschte Ziel erreicht ist und die Bedingung aus Gleichung (3.6) immer erfüllt ist, wird auf eine explizite Behandlung dieses Falles verzichtet.

1.) $W_{i+1}^{\{k+1\}} > W_i^{\{k+1\}}$**:** In diesem Fall wird zu wenig Arbeit übertragen und nach Übertragung des Gebietes hat das Gebiet mit mehr Arbeit immer noch mehr Arbeit, als das andere Gebiet. Werden nun die Differenzen verglichen, soll gelten:

$$W_{i+1}^{\{k+1\}} - W_i^{\{k+1\}} < W_{i+1}^{\{k\}} - W_i^{\{k\}} \tag{3.7}$$

Setzt man außerdem vorraus, dass für einen beliebigen Iterationszeitschritte l gilt, dass

$$W = W_{i+1}^{\{l\}} + W_i^{\{l\}} \tag{3.8}$$

so folgt mit W als Gesamtarbeit über beiden Gebieten, dass sich alle Arbeitsdifferenzen schreiben lassen als:

$$W_{i+1}^{\{l\}} - W_i^{\{l\}} = (W - W_i^{\{l\}}) - W_i^{\{l\}} = W - 2 \cdot W_i^{\{l\}} \tag{3.9}$$

Setzt man Gleichung (3.9) in Gleichung (3.7) ein, so erhält man:

$$\begin{aligned} W_{i+1}^{\{k+1\}} - W_i^{\{k+1\}} &< W_{i+1}^{\{k\}} - W_i^{\{k\}} \\ \Leftrightarrow W - 2 \cdot W_i^{\{k+1\}} &< W - 2 \cdot W_i^{\{k\}} \\ \Leftrightarrow W_i^{\{k+1\}} &> W_i^{\{k\}} \\ \overset{Gleichungen(3.4,3.5)}{\Rightarrow} \delta W_i^{\{k\}} &> 0 \end{aligned} \tag{3.10}$$

Gleichung (3.10) sagt also aus, dass falls zu wenig Arbeit übertragen wird, die wahre Dichte auf dem Gebiet, das übertragen wird, positiv sein muss, damit die Bedingung erfüllt ist. Diese Bedingung ist für alle betrachteten Systeme erfüllt. Die Folge $(x_i^{\{k\}})$ entspricht damit einer monoton wachsenden Folge, die nach oben durch x_i^* beschränkt ist, wobei gilt, dass x_i^* die Position der Grenze ist, für die die Arbeit der Nachbargebiete gleich groß ist.

2.) $W_{i+1}^{\{k+1\}} < W_i^{\{k+1\}}$**:** Wird beim Gebietsübertrag mehr Arbeit übertragen, als vorgesehen, so vertauschen sich die Größenverhältnisse der Arbeiten im nächsten Schritt. Es gilt dann, dass $W_{i+1}^{\{k+1\}} < W_i^{\{k+1\}}$ ist und es muss folgende Bedingung betrachtet werden, um eine Einschränkung für $\rho(x)$ zu finden:

$$W_i^{\{k+1\}} - W_{i+1}^{\{k+1\}} < W_{i+1}^{\{k\}} - W_i^{\{k\}} \tag{3.11}$$

Setzt man nun in Gleichung (3.11) Ergebnisse aus den vorherigen Gleichungen ein, so erhält man:

$$\begin{aligned} W_i^{\{k+1\}} - W_{i+1}^{\{k+1\}} &< W_{i+1}^{\{k\}} - W_i^{\{k\}} \\ \overset{Gleichung(3.8)}{\Leftrightarrow} 2 \cdot W_{i+1}^{\{k\}} - W &< W - 2 \cdot W_i^{\{k\}} \\ \overset{Gleichung(3.5)}{\Leftrightarrow} 2 \cdot (W_i^{\{k\}} + \delta W_i^{\{k\}})^{\{k\}} - W &< W - 2 \cdot W_i^{\{k\}} \\ \Leftrightarrow \delta W_i^{\{k\}} &< W - 2 \cdot W_i^{\{k\}} \end{aligned} \tag{3.12}$$

Wendet man auf Gleichung (3.4) den Mittelwertsatz der Integralrechnung an, so erhält man

$$\delta W_i^{\{k\}} := \int\limits_{\delta_x(i,i+1;\gamma)} \rho(x)dx = \rho(\xi) \cdot \delta_x(i, i+1; \gamma), \xi \in \delta_x(i, i+1; \gamma) \tag{3.13}$$

Definiert man nun $\rho_\delta := \rho(\xi)$, so kann man mit diesem Ergebnis und Gleichung (3.12) abschätzen:

$$\begin{aligned} \delta W_i^{\{k\}} &< W - 2 \cdot W_i^{\{k\}} \\ \overset{Gleichung(3.13)}{\Leftrightarrow} \Delta_x(i, i+1; \gamma) \cdot \rho_\delta &< W - 2 \cdot W_i^{\{k\}} \\ \overset{Gleichung(3.1)}{\Leftrightarrow} \rho_\delta \cdot \frac{1}{\gamma \cdot \rho_{i+\frac{1}{2}}} \cdot (W - 2 \cdot W_i^{\{k\}}) &< W - 2 \cdot W_i^{\{k\}} \\ \Leftrightarrow \rho_\delta &< \gamma \cdot \rho_{i+\frac{1}{2}} \\ \Rightarrow \max_{x \in \delta_x(i,i+1;\gamma)} \rho(x) &\overset{!}{<} \gamma \cdot \rho_{i+\frac{1}{2}} \end{aligned} \tag{3.14}$$

Um ein Kriterium für $\rho(x)$ zu finden, wird die Größe ρ_δ abgeschätzt durch das Maximum der wahren Dichte auf dem Bereich $\delta_x(i, i+1; \gamma)$. Somit wird

mit Gleichung (3.14) eine Bedingung angegeben, wie die wahre Dichte der Arbeitsverteilung aussehen darf, bei vorgegebenem γ. Umgekehrt kann man auch eine untere Schranke für γ angeben, bei vorgegebener Dichte der Arbeitsverteilung. Um ein globales γ für alle Gitterpunkte bestimmen zu können, muss man das globale Maximum der Arbeitsverteilungsdichte in Gleichung (3.14) einsetzen und statt der lokalen Gitterpunktsdichte das Minimum der Gitterpunktsdichten einsetzen. Somit hätte man mit

$$\gamma > \frac{\max\limits_{x \in \delta_x(i,i+1;\gamma)} \rho(x)}{\min\limits_{1 \leq i \leq n-1} \rho_{i+\frac{1}{2}}}$$

eine Abschätzung für ein globales γ, das für alle Gebietsgrenzen eingesetzt werden kann.

3.3 Kommunikationsschemata

Da das Loadbalancing fast sicher die Äquidistanz des Gitters aufhebt, ist ein weiterer wichtiger Teil dieser Arbeit der Entwurf von Kommunikationsschemata für Vorgänge innerhalb des Programmes, wie Teilchentransport und Kraftberechnung. Bei diesen Vorgängen muss ein Teilchen von einem Prozess entweder permanent (Teilchentransport) oder temporär (Kraftberechnung) auf einen benachbarten Prozess übertragen werden.
Bei einem äquidistanten Gitter sind diese Vorgänge relativ einfach durchzuführen, während sie bei nicht äquidistanten Gittern aufwändiger werden. Der Grund hierfür ist vor allem, dass einzelne Prozesse nun mehr als einen Nachbarn in y- und z-Dimension haben können. Somit muss sicher gestellt werden, dass Daten nicht doppelt oder gar nicht an ihren designierten Zielprozess übertragen werden.

Dimension	Bedingung, um Nachbar zu sein
x	y-/z-Koordinate müssen übereinstimmen, x-Koordinate muss 1 größer / kleiner sein, als x (bei periodischen Randbedingungen muss gegebenenfalls eine Modulotransformation stattfinden)
y	z-Koordinate muss übereinstimmen, y-Koordinate muss 1 größer / kleiner sein, als y (bei periodischen Randbedingungen muss gegebenenfalls eine Modulotransformation stattfinden) Ausdehnung des Gebietes muss in x-Dimension mit dem Ursprungsgebiet überlappen
z	y-Koordinate muss 1 größer / kleiner sein, als z (bei periodischen Randbedingungen muss gegebenenfalls eine Modulotransformation stattfinden) Ausdehnung des Gebietes muss in x-/ und y-Dimension mit dem Ursprungsgebiet überlappen

Tabelle 3.1: Bedingungen, die ein Gebiet erfüllen muss, um in angegebener Dimension ein Nachbar des betrachteten Gebietes zu sein.

3.3.1 Nachbarschaftsbestimmung

Von zentraler Bedeutung für die Kommunikation ist die korrekte Bestimmung von Nachbarn. Diese Bestimmung muss sowohl zu Beginn des Programmes, als auch nach jedem Loadbalancingschritt durchgeführt werden, da sich jeweils die Nachbarn verändern können. Um die Nachbarn zu bestimmen, werden die Gebietsgrenzen der umliegenden Gebiete benötigt. Sind diese bekannt, können die Nachbarn in der jeweiligen Dimension wie folgt bestimmt werden. Ausgegangen wird jeweils vom Prozessor mit Prozesskoordinaten x,y,z. In Tabelle (3.3.1) bedeutet "überlappen", dass untere und/oder obere Grenze des entsprechenden Gebietes in der jeweiligen Dimension zwischen den Grenzen der jeweiligen Dimension im Ursprungsgebiet liegen.

3.3.2 Kommunikation beim Teilchentransfer

Der weniger komplexe Fall ist die permanente Übertragung eines Teilchens von einem Prozess auf einen Nachbarprozess. Dies findet statt, wenn in Folge von Bewegungen eines Teilchens dieses die Grenzen des Gebietes, in dem es sich zuletzt aufhielt verlässt. Dann muss das Teilchen entsprechend auf den jeweiligen Nachbarprozess, der das entsprechende Nachbargebiet abbildet, übertragen werden.
Problem hierbei ist, dass Teilchen diagonal, d.h. auf einen nicht direkt benachbarten Prozess, gelangen können. Dann muss das zugrunde liegende Kommunikationschema in der Lage sein, dieses Teilchen zum korrekten Prozess zu übertragen. Dabei darf es dann nicht vorkommen, dass ein Teilchen sich vermehrfacht, weil es gegebenenfalls zu oft übertragen wird.
Um diese Probleme zu vermeiden, bzw. zu lösen, wird die Reihenfolge, in der übertragen wird wie folgt vorgegeben:

1.) z-Dimension

2.) y-Dimension

3.) x-Dimension

Darüber hinaus wird bei der Übertragung von Teilchen wie folgt vorgegangen. Zunächst wird für jede Übertragung geprüft, welche Teilchen das aktuelle Gebiet in der jeweiligen Richtung verlassen haben. Diese Teilchen werden an alle Nachbaren in dieser Richtung übertragen. Auf den Nachbarn werden dann nach bestimmten Bedingungen die Teilchen aussortiert, die akzeptiert werden und die, die nicht akzeptiert werden, verworfen. Die Bedingungen richten sich hierbei nach der jeweiligen Dimension, in der übertragen wurde (Tabelle (3.3.2)).

Durch diese Bedingung wird sicher gestellt, dass ein Teilchen, das diagonal transportiert werden muss, auch zum entsprechenden Prozess transportiert wird. Nach jedem Transportschritt in eine Dimension werden die empfangenen Teilchen sofort in den Inhalt dem empfangenden Prozesses einsortiert.

Richtung der Übertragung	Bedingung, dass ein Teilchen nicht verworfen wird
z-Dimension	Empfangendes Gebiet ist Nachbar in z-Richtung a) Teilchen liegt genau (x,y) im Gebiet des Nachbarn b) Teilchen liegt in y-Richtung jenseits der Grenzen und Nachbar ist in y-Richtung einer der oberen z-Nachbarn c) Teilchen liegt in x-/y-Richtung jenseits der Grenzen und Nachbar ist in y-Richtung einer der oberen z-Nachbarn und in x-Richtung einer der äußeren z-Nachbarn
y-Dimension	Empfangendes Gebiet ist Nachbar in y-Richtung a) Teilchen liegt genau (x) im Gebiet des Nachbarn b) Teilchen liegt in x-Richtung außerhalb der Grenzen und Nachbar ist in entsprechender Richtung äußerster Nachbar
x-Dimension	Empfangendes Gebiet ist Nachbar in x-Richtung

Tabelle 3.2: Bedingungen, unter denen Teilchen beim Transport zwischen Gebieten übernommen werden

Wenn dies für alle empfangenen Teilchen von allen Nachbarn in der jeweiligen Dimension durchgeführt wurde, werden aus all diesen Teilchen die Teilchen für den Transport in der nächst niedrigeren Dimension gesucht. Dadurch wird der Transport von Teilchen in der Diagonalen in der Reihenfolge $z \rightarrow y \rightarrow x$ kaskadiert.

3.3.3 Kommunikation bei der Kraftberechnung

Das Loadbalancing-Verfahren hat Auswirkungen auf die Berechnung von kurzreichweitigen Wechselwirkungen.Das Kommunikationsschema zur Kraftberechnung wird deutlich aufwändiger als beim Teilchentransport, da es hierbei notwendig ist, die Kräfte exakt zu dem Teilchen zurück zu kommunizieren, für das sie berechnet wurden. Somit teilt sich das Kommunikationsschema in zwei Schritte auf, zunächst müssen die Teilchen temporär so kommuniziert werden, dass auf allen Prozessen alle benötigten Teilchen zur Kraftberechnung vorhanden sind und danach müssen die berechneten Kräfte zu den Ursprungsprozessen der transferierten Teilchen kommuniziert werden.
Ein zentrales Problem ist also den Weg, über den ein Teilchen transportiert wurde, korrekt nachvollziehen zu können. Es werden nämlich zunächst "Geisterteilchen" als Platzhalter zur Kraftberechnung an die Nachbaren versandt und dann auf dem gleichen Weg die Kräfte, die von den Nachbaren für diese "Geisterteilchen" berechnet wurden an den Prozessor zurück kommuniziert, wo das wirkliche Teilchen liegt. Damit teilt sich das Kommunikationsschema in zwei Teilbereiche. Der erste Teil ist der Hintransport der "Geisterteilchen" gefolgt vom Rücktransport der berechneten Kräfte.
Eine Möglichkeit für den Transport der "Geisterteilchen" ist, im Gegensatz zum Teilchentransport, der simultane Transport von allen Teilchen im Randgebiet an alle Nachbarn. Das Randgebiet wird hierbei u.a. durch den Cut-Off-Radius bestimmt. Die so transportierten Teilchen werden im nächsten Schritt vom Empfänger wieder transportiert und dieser Vorgang wird ein drittes Mal wiederholt, sodass alle Teilchen aus 'Ecken' ebenfalls übertragen worden sind. Wie beim Teilchentransport auch, werden die empfangenen Teilchen auch hier sortiert, so dass nur Teilchen weiter verarbeitet werden,

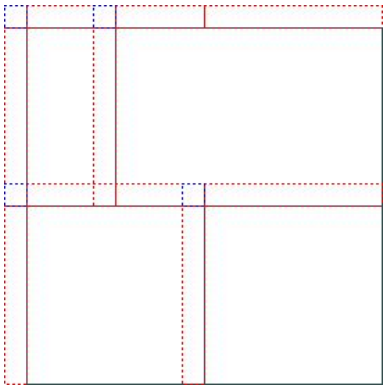

Abbildung 3.4: Erläuterung zum Krafttransport
Gebiete, aus denen den Prozessen (schwarze Gebiete) nach dem Geisterteilchen-Transport temporäre Teilchen zur Berechnung der Kräfte zur Verfügung stehen. Das Beispiel zeigt ein zweidimensionales Beispiel mit periodischen Randbedingungen, weswegen es Gebiete gibt, die außerhalb der wahren Grenzen liegen. Geisterteilchen in diesen Gebieten müssen so transformiert werden, dass sie entsprechende Koordinaten außerhalb des eigentlichen Systems erhalten, damit die Berechnung korrekt verlaufen kann.

die in das Empfangsgebiet gehören.

Die Skizze (3.4) zeigt aus welchen Bereichen den jeweiligen Gebieten Geister-Teilchen in einem zweidimensionalen Beispiel zur Verfügung stehen. Die schwarzen, durchgezogenen Linien sind dabei die wirklichen Gebietsgrenzen und die gestrichelten Linien markieren die Gebiete aus denen Geisterteilchen vorliegen. Im ersten Schritt werden die roten Bereiche nach unten und rechts übertragen, d.h. die roten Bereiche oberhalb und rechts eines Gebietes werden diesem temporär hinzugefügt. Damit fehlen aber noch Bereiche, diese werden der Form halber als 'Ecken' bezeichnet, da es die Ecken zwischen dem von links und dem von oben erhaltenen Bereich sind, die fehlen. Daher wird zum Auffüllen der Ecken, der rechte Bereich des von oben empfangenen Bereiches nun abermals nach rechts übertragen. Somit werden die Ecken aufgefüllt und alle benötigten Teilchen wären in jedem Gebiet vorhanden.
Beim Rücktransport werden zunächst die Ecken nach links und dann die zuerst übertragenen Bereiche nach oben, bzw. links übertragen werden. So-

mit wären nach fertigem Transport auf allen Prozessen für alle Teilchen die Kräfte berechnet. Bei den Teilchen, die vom Rücktransport betroffen sind, werden dann die Kräfte aus den unterschiedlichen Schritten aufaddiert, d.h. wenn die Ecke in den ersten Bereich zurück transportiert wird, werden die berechneten Kräfte aus der Ecke mit den im Bereich berechneten Kräfte addiert und beim Rücktransport ins Ursprungsgebiet diese Kräfte mit denen, die im Ursprungsgebiet berechnet wurden, addiert.
Beim dreidimensionalen Fall kann analog vorgegangen werden. Hier werden dann die in z-Richtung (tiefer) übertragenen Bereiche nochmal teilweise sowohl in x-Richtung (rechts), als auch in y-Richtung (unten) übertragen. Die so in y-Richtung übertragenen Bereiche würden dann abermals in x-Richtung kommuniziert und dann die Kräfte berechnet und analog zum Teilchenübertragungsschema zurückkommuniziert.

Kapitel 4

Implementierung des Loadbalacingverfahrens

4.1 Beschreibung von MP2C

MP2C ist ein in FORTRAN90 geschriebenes Programm, das in der Lage ist hydrodynamische und molekulardynamische Probleme zu simulieren. Bei der Entwicklung von MP2C wurden alle wichtigen Bestandteile des Programms modular entwickelt, so dass eventuell einzelne Programmteile in andere Programmpakete ausgegliedert werden können. Außerdem erleichtert diese Art der Entwicklung die Wartung und Weiterentwicklung des Programmes, so dass neue Funktionalitäten nur in die jeweils betroffenen Module eingepflegt werden müssen, ohne dass hierfür Änderungen an vielen verschiedenen Stellen notwendig sind. Allerdings kann es bei umfassenderen Neuentwicklungen und Verbesserungen durchaus dazukommen, dass in mehreren Modulen Änderungen eingepflegt werden müssen. Dies war auch im Rahmen dieser Arbeit der Fall, da die Gebietsgrößen und Kommunikationswege angepasst werden mussten.
Im ersten Abschnitt dieses Kapitels wird daher kurz auf die wichtigsten Module in MP2C eingegangen und im zweiten Abschnitt kurz erläutert welche Änderungen in welchen Modulen notwendig waren, um das Loadbalancing-

Verfahren in MP2C zu implementieren.

4.1.1 Wichtige Module

Insgesamt kann man die Module von MP2C in vier große Teilbereiche unterteilen. Diese Teilbereiche wären:

- Ein-/Ausgabe (E/A)
- Kommunikation
- Kraftberechnung
- Verwaltung

Bei den Bereichen Kommunikation und Kraftberechnung ist außerdem noch eine grobe Unterteilung in den Molekulardynamik (MD)Teil und den Hydrodynamik (MPC)Teil. Im Rahmen dieser Arbeit wurde sich aber nur mit dem MD-Teil beschäftigt, die Anpassung des MPC-Teils an das Loadbalancing-Verfahren wird im Anschluß an diese Arbeit vorgenommen.
In den E/A-Modulen werden zunächst die Einstellungen und Anfangskonfigurationen eingelesen. Dabei wird aus dem Hauptmodul das Eingabemodul (*input_module*) aufgerufen, welches intern weitere Module zum Einlesen der Konfiguration (*input_slt_module* und *input_slv_module*) und der Programmparameter (z.B. *input_slt_parameters_module*) aufruft. Zur Ausgabe dient entsprechend das Ausgabemodul (*output_module*).
Die Kommunikation wird größtenteils im Module *parallel_module* verarbeitet. Zur Kommunikation zählt hierbei der Transport von Teilchen über Gebietsgrenzen, sowohl für das Eigth-Shell-Verfahren, als auch für den Transport, wenn Gebietsgrenzen verlassen werden im Laufe der Simulation. Diese Routinen liegen doppelt vor, jeweils für den MD-, als auch für den MPC-Teil.
Auf mehrere Module aufgeteilt aber zentral im Modul *interaction_module* verwaltet sind die Module zur Berechnung der Wechselwirkungen. In den

einzelnen Modulen werden die Wechselwirkungen der Teilchen aufeinander berechnet. Dabei ist für jedes bisher implementierte Potential ein eigenes Modul vorgesehen, z.B. *interaction_lj_module* für das Lennard-Jones-Potential. Die letzte große Gruppe von Modulen sind die Verwaltungsmodule. Ihre Aufgabe sind u.a. das Bereitstellen von Datentypen (*data_type_module*), das Initialisieren von Gebieten (*parallel_dd_module*) und die Programmsteuerung im Hauptmodul (*main_module*).

4.1.2 Geänderte Module

In dieser Arbeit wurde ein Loadbalancing-Verfahren in MP2C implementiert, so dass Änderungen am Code notwendig waren. An dieser Stelle wird nun aufgelistet, an welchen Stellen diese Änderungen vorgenommen wurden und welcher Art sie waren, damit der geneigte Leser gegebenenfalls in der Lage ist die Änderungen nachzuvollziehen und seinerseits weitere Anpassungen vorzunehmen.

Das (neue) Loadbalancing-Modul: loadbalancing_module

Die Routinen, die sich überwiegend mit dem Loadbalancing beschäftigen sind in einem seperatem Modul ausgelagert worden, um auch weiterhin die Trennung von Funktionsweisen in jeweilige Module beizubehalten. Die wichtigste Routine hierbei ist die Routine *load_balancing*, über die das Loadbalancing-Verfahren, wie es im vorherigen Kapitel dargestellt wurde, gestartet wird. Außerdem findet sich in dem Modul die Routine *domain_neighbours2*, welche eine Ergänzung der Routine *domain_neighbours* aus dem Modul *parallel_dd_module* ist.
Die Ergänzung wurde notwendig, da MP2C ursprünglich für die Nutzung mit äquidistanten Gittern entworfen wurde und so nur zwei Nachbaren in jeder kartesischen Richtung vorgesehen waren. Einhergebend mit dieser Änderung war das Ändern des Datentyps von *par%rem_id* von einem 3×2-Array in ein genügend großes eindimensionales Array, was Anpassungen an mehreren Mo-

dulen notwendig machte, da die Nachbarn an vielen Stellen abgefragt werden.

Das Kommunikations-Modul: parallel_module

Ein Modul das von den Änderungen an den Nachbarn stark betroffen war, ist das Modul *parallel_module*, da hier die Kommunikationsroutinen *particle_exchange_slt*, *ghost_exchange_slt* und *force_exchange_slt* zu finden sind. Alle drei Routinen mussten so angepasst werden, dass sie in der Lage sind Teilchen, bzw. Teilcheninformationen so zu transportieren, dass trotz mehrerer möglicher Nachbaren die Informationen zu dem Nachbarn gelangen, der sie benötigt, bzw. ursprünglich zur Verfügung stellte.
Hierfür wurden zunächst alle Kommunikationen, die zuvor einzeln durchgeführt wurden, in Schleifen über alle Nachbaren in entsprechender Richtung umformuliert. Dazu mussten dann auch Requestvariablen, die dafür sorgen, dass einem Prozess bekannt ist, wann gewisse Kommunikationen beendet sind, von Skalaren auf ein-dimensionale Arrays erweitert werden. Gleiches gilt für die Statusvariablen, die nun statt eindimensionaler Array mehrdimensionale Arrays geworden sind, um Informationen von allen getätigen Kommunikationen verarbeiten zu können.
Eine große Änderung im Teilchentransport war das Einfügen der Randbedingungen für das Akzeptieren eines empfangenen Teilchens (s. (3.3.2), welche notwendig wurde, um sicher zu stellen, dass Teilchen gegebenenfalls auch zu Nachbaren in der Diagonalen gelangen. Außerdem wurde die Reihenfolge der Dimensionen, in die kommuniziert wird umgekehrt, da der korrekte Teilchentransport sonst auf Grund der verschobenen Grenzen auch nicht sicherzustellen war. Die Kommunikationsreihenfolge ist somit z-y-x.
Der Transport der Geisterteilchen, d.h. Teilchen, deren Positionen für die Berechnung von Kräften auf Nachbargebieten benötigt werden, musste ebenfalls angepasst werden. Da aber das Eigth-Shell-Verfahren angewandt wird, muss hier nur in jeweils eine Richtung kommuniziert werden. Da es im Ursprungscode so gemacht wurde, wird auch weiterhin “nach unten” kommuniziert, d.h. mit dem Nachbarn, an die jeweils untere Grenze angrenzt. Damit

der Rücktransport in *force_exchange_slt* fehlerfrei möglich ist, wurde in *data_type_module* der Datentyp *domain_exchange* um ein mehrdimensionales Array *indices* erweitert, auf dem als Zusatzinformation Quellgebiet und der Index in der Liste auf dem Prozess, der das Quellgebiet verwaltet, abgespeichert werden, sobald ein Geisterteilchen empfangen wird. Ansonsten wurde von den Anpassungen, die auch in *particle_exchange_slt* notwendig waren abgesehen, auf Änderungen am Code verzichtet.

Das Steuerungs-Modul: main_module

Auch wenn die Änderungen im Hauptprogramm gering ausgefallen sind, so sollen die doch erwähnt werden. Der Aufruf der Loadbalancing-Routine wurde hinzugefügt, genauso wie die Ausgaben am Ende des Programmes erweitert wurden, um mehr Informationen im Zusammenhang mit dem Loadbalancing zu vermitteln. Hierfür wurden auch weitere Timer eingebaut. Diese werden teils auch dafür benutzt, um ein Loadbalancing mit der verbrauchten Zeit als Arbeit zu ermöglichen (s.u.).

Das Datentyp-Modul: data_type_module

Im Laufe der Implementierung mussten etliche alte Datentypen angepasst oder geändert werden, um die Kommunikation mit mehreren Nachbaren, wo zuvor nur ein Nachbar vorhanden war, zu ermöglichen. So wurde der Datentyp *parallel_control* u.a. um das Element *prc_time* erweitert, um ein Loadbalancing zu ermöglichen, das als Arbeit die verbrauchte Zeit in gewissen Routinen (einzustellen in *main_module*) verwendet. Genauso wurden wie oben schon erwähnt die Elemente zur Verwaltung von Nachbarschaftsbeziehungen erweitert, um mehr als einen Nachbarn in jede kartesische Richtung zu ermöglichen.
Außerdem wurden als Hilfsmittel für die Kommunikation Arrays von Kommunikatoren eingeführt, die es ermöglichen während des Loadbalancings die benötigten Daten über die Arbeit schnell an alle Prozesse weiter zu leiten, die diese benötigen (*x_comm,y_comm,z_comm*, alle zweidimensional). Da-

mit nicht immer der richtige Kommunikator berechnet werden muss, werden die ensprechenden Indizes im Element *prc_indices* abgespeichert.

Sonstige Änderungen

Außer in den zuvorgenannten Modulen, wurden auch in anderen Modulen teilweise Änderungen vorgenommen. Diese zielten meist auf die generelle Verbesserung von MP2C ab und waren nicht immer unbedingt im Zusammenhang mit dem Loadbalancing. Eine der Änderungen war die Anpassung von der Gebietsunterteilung in *parallel_dd_module*. Diese erlaubte zuvor nur Prozessorzahlen, die Zweierpotenzen, bzw. Summen von Zweierpotenzen waren. Nach der Erweiterung werden nun durch Verwendung der Routine *MPI_CREATE_DIMS* alle Prozessorzahlen erlaubt, allerdings ist die Aufteilung nun von der jeweiligen Implementierung der MPI-eigenen Routine abhängig. Im Zuge dieser Anpassung wurden auch in der Routine zur Berechnung des Lennard-Jones-Potenzials und seiner Wechselwirkungen ein Fehler behoben, der bei eindimensionalen Aufteilungen auftrat.
Zusätzlich wurde der Code an vielen weiteren Stellen gestrafft und vereinheitlicht, da er durch die andauernde Entwicklung teilweise Programmierstile im selben Modul enthielt. Dies wurde behoben und der Übersichtlichkeit wegen angepasst.

Kapitel 5

Untersuchung der Verfahrensparameter

5.1 Untersuchungen zur Anpassungsqualität des Verfahrens unabhängig von MP2C

In diesem Abschnitt werden kurz Ergebnisse einer Simulation des zuvor vorgestellten Anpassungsverfahrens präsentiert, die zeigen sollen, wie gut das Verfahren auch stark inhomogene Systeme in der Theorie anpassen kann. Dabei ist an dieser Stelle noch keine Aussage über die Laufzeiten des Programmes zu machen, da zur Erzeugung dieser Ergebnisse ein gesondertes, nicht in MP2C implementiertes Testprogramm entworfen wurde, das zudem keine Teilchensimulation irgendeiner Art durchführt. Aus diesem Grund ist dieser Abschnitt an den Anfang des Kapitels gestellt und losgelöst von MP2C zu sehen.
Zu dem untersuchten System ist zu sagen, dass es aus Teilchenkoordinaten besteht, die auf konzentrischen Kugelschalen angeordnet sind. Von diesen Kugelschalen liegen zehn Stück ineinander, der Mittelpunkt ist der Punkt $(0.5; 0.5; 0.5)^T$ im kartesischen Koordinatensystem, der größte Radius beträgt 0.5 und die inneren Radien werden jeweils um den Faktor 0.5 verkleinert [3].

Der Grund hierfür ist, dass so die erste und letzte Schicht an Gebieten zu Beginn leer ist und in der Folge mit Teilchen gefüllt werden muss.
Am unteren Rand der Abbildungen ist eine Skala abgebildet, die die farbliche Kodierung der Gebiete erläutert. Dabei wird die Teilchenanzahl relativ zur durchschnittlichen Anzahl an Teilchen, die in jedem Gebiet enthalten sein sollte ($\frac{\text{Anzahl Teilchen im System}}{\text{Anzahl der Gebiet des Systems}}$) indiziert. Die Skala ist nach oben bei der dreifachen Anzahl des gewünschten Inhalts gedeckelt, ein Vorkommen der entsprechenden Farbe kann allerdings auch bedeuteten, dass eine weit größere Anzahl an Teilchen in diesem Gebiet vorhanden ist, z.B. im Zentrum der Schalen. Als Arbeit wurde hier, da keine Molekulardynamik-Berechnungen durchgeführt wurden, die Teilchenanzahl gewählt.
Auf Abbildung (5.1) ist zu erkennen, dass die äquidistanten Gitter zu einer stark schwankenden Teilchenanzahl je Gebiet führen. Wie zuvor erwähnt sind die erste und das letzte Schicht leer. Nach einigen Anpassungen (s. Abbildung (5.2)) sind die Gebiete deutlich in Richtung der Mitte gestaucht worden. Dadurch werden die äußeren Gebiete gestreckt und bekommen so Inhalt. Die jeweiligen Inhalte sind nun schon deutlich näher am gewünschten Durchschnitt, als zu Beginn der Simulation (häufigeres Vorkommen von Grüntönen). Zum Ende der Simulation sind nur noch wenige Gebiete vorhanden, die nicht in einen Grünton gefärbt sind (s. Abbildung (5.3)). Dies ist ein guter Indikator dafür, dass das Verfahren in der Lage ist, auch stark inhomogene Systeme, d.h. System bei denen die Arbeitsdichte (hier die Teilchenanzahl) im Gebiet stark schwankt.

5.2 Performanceuntersuchungen mit MP2C

Nachdem das Verfahren implementiert wurde, wurden im Rahmen der Arbeit natürlich auch Vergleiche bezüglich der neuen Performance gemacht. In der Ursprungsversion von MP2C war es üblich Zeiten normiert auf die Zeit, die für ein Teilchen in jedem Simulationschritt aufgewandt wurde zu betrachten. Diese Vorgehensweise wurde aus Kompatibilitätsgründen übernommen. Daher sind im Folgenden alle Zeiten in der Einheit $\frac{\mu s}{n_{Teilchen} \cdot n_{Simulationsschritte}}$

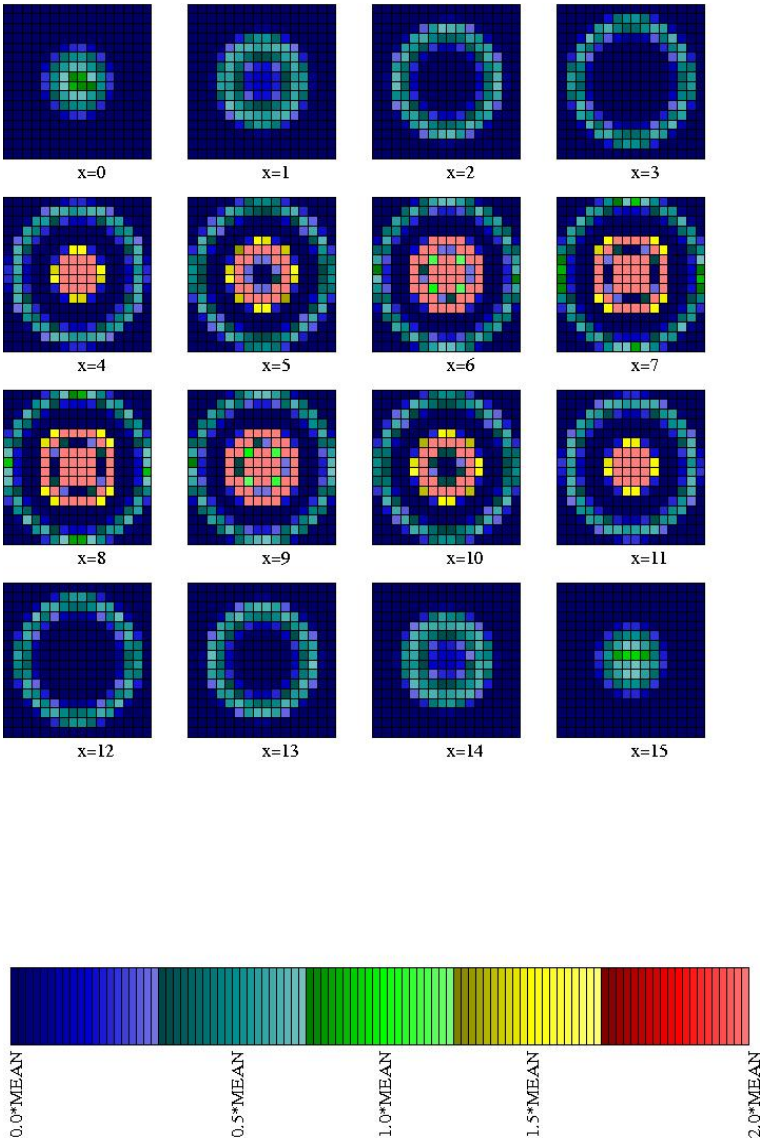

Abbildung 5.1: Anfangszustand eines inhomogenen Systems, das in 16^3 äquidistante Gebiete unterteilt wurde. Die zu sortierenden Teilchen sind in konzentrischen Schichten um einen gemeinsamen Mittelpunkt herum aufgebaut. Dabei halbieren sich die Radien der Schalen bei gleichbleibender Teilchenanzahl auf den Schalen, d.h. auf jeder Schale ist die gleiche Anzahl von Teilchen vorhanden.

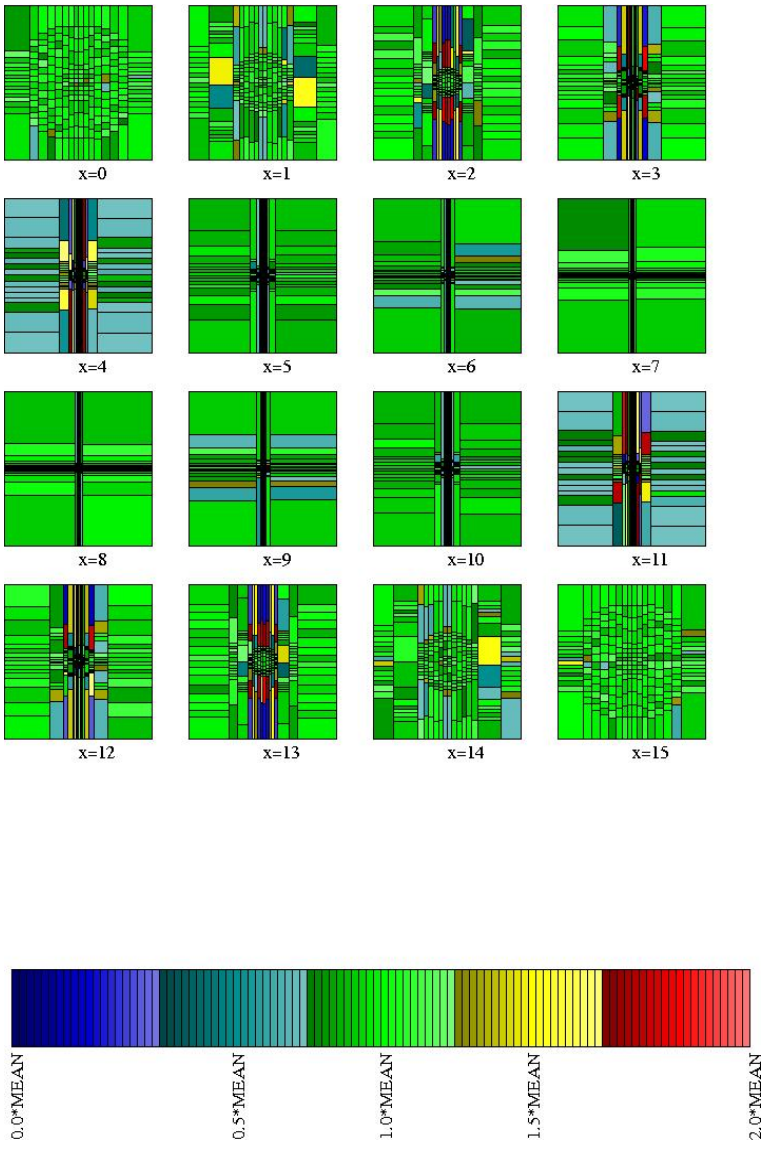

Abbildung 5.2: Zustand des vorherigen Schalen-Systems, das in 16^3 Gebiete unterteilt wurde, nach etwa zehn Anpassungsschritten. Ein Anpassungschritt bedeutet dabei, dass die z-Ebene einmal, die y-Säulen fünfmal und die z-Säulen zehnmal pro Anpassungschritt korrigiert wurden. Man sieht, dass die Gebiete deutlich besser angepasst sind, als zu Beginn der Simulation.

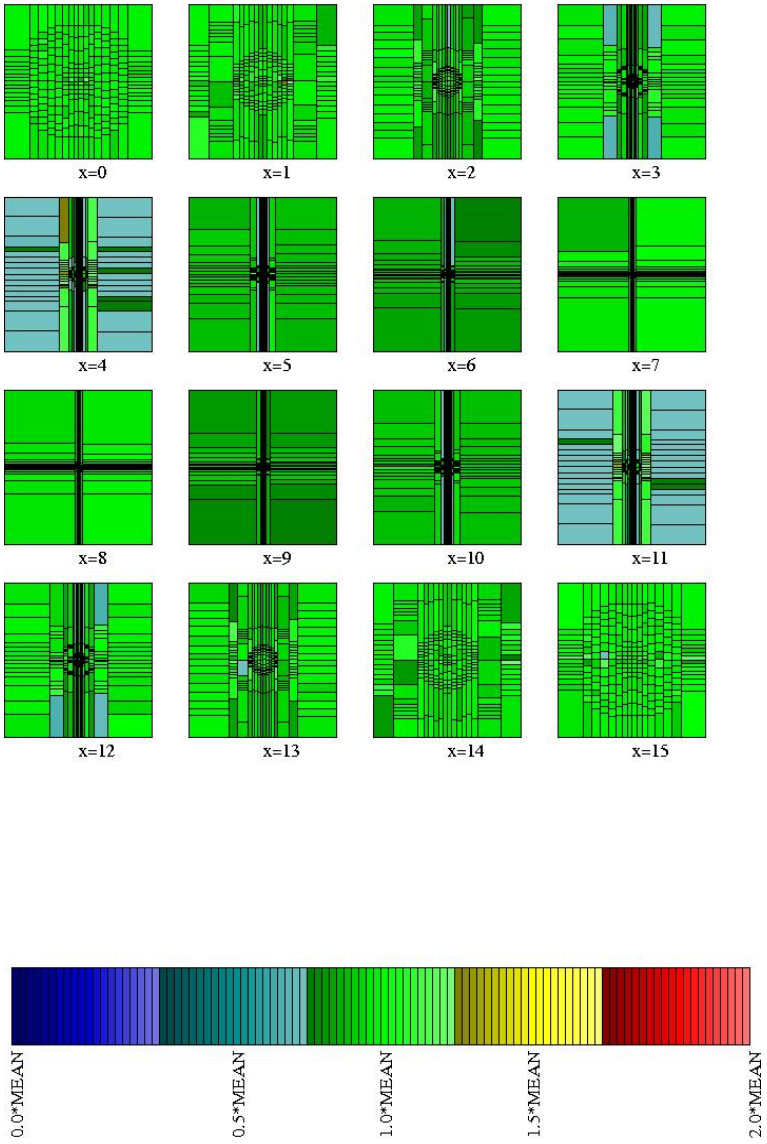

Abbildung 5.3: Das Schalen-System am Ende der Anpassungen. Es wurden etwa 100 Anpassungschritte durchgeführt und das System verfügt nun über Gebiete, in denen sich jeweils etwa gleich viele Teilchen befinden.

angegeben. Die Untersuchungen wurden auf JUGENE [1], einem der Großrechner des Forschungszentrums Jülich durchgeführt.
Es werden hauptsächlich drei Zeiten miteinander verglichen, die **Kommunikationszeit**, die **Interaktionszeit** und die von MP2C intern ermittelte **"Gesamtzeit"** für den MD-Teil. Die Kommunikationszeit ist die maximale Zeit, die ein Prozess in allen drei Kommunikationsroutinen (*particle_exchange_slt*,*ghost_exchange_slt* und *force_exchange_slt*) verbracht hat. Die Interaktionszeit ist die maximale Zeit, die benötigt wurde, um Wechselwirkungen zu berechnen, also die Zeit, die im Interaktionsmodul verbraucht wurde. Die "Gesamtzeit" schließlich ist die Zeit, die das Programm zur Abarbeitung des MD-Teils von MP2C benötigte. Sie ist größer, als die Summe von Kommunikations- und Interaktionszeit, da auch noch andere Berechnungen, die im Rahmen des MD-Teils durchgeführt werden müssen, in sie einfließen. Außerdem umfasst sie alle Wartezeiten, die durch Barrieren entstehen, die nach den einzelnen Routinen dafür sorgen, dass alle Prozesse gleichzeitig neue Teilbereiche des Programmes erreichen.
Das System, mit dem die Laufzeituntersuchungen durchgeführt wurden, besteht aus 25000 Teilchen, die in 100 Molekülketten von jeweils 250 Teilchen angeordnet sind. Durch eine Scherströmung bewegen sich diese Moleküle in z-Richtung durch das System. Dabei ist das System so aufgebaut, dass durch die Scherströmung keine Teilchen das System verlassen, sondern periodisch am gegenüberliegenden Rand wieder in das System eintreten. Abbildung (1.1) zeigt das System. Wie schon in der Einleitung erläutert sind die Moleküle stark inhomogon über das System verteilt.

Die Untersuchungen, wie sich das Programm für verschiedene Parameter verhält, z.B. unterschiedliche adaptive Bestimmung von γ, wurden ebenfalls auf JUGENE durchgeführt. Aus diesem Grund werden in diesem Kapitel sowohl die Ergebnisse der Parameteruntersuchung, als auf die Laufzeituntersuchung vorgestellt. Abschließend werden die auf JUGENE ermittelten Ergebnisse mit denen von JUMP verglichen. Als Anzahl der Prozesse wurden die Untersuchungen bei der Untersuchung der Laufzeit für Zweipotenzen zwischen 8 und 1024 durchgeführt. Bei den Parameteruntersuchungen wurde entweder

mit 1024 Prozessen simuliert, oder im Fall der Untersuchung der Unterschiede in den Standardabweichungen mit 128 und 1024.

Parameteruntersuchungen auf JUGENE

Hauptsächlich wurde bei den Parameteruntersuchungen verglichen, wie sich die Standardabweichung von Laufzeit zwischen Loadbalancing-Schritten und Teilchenanzahl für unterschiedliche gewählte γ entwickelt, bzw. wie sich die Varianzen für die beiden unterschiedlichen Arbeitsdefinitionen entwickeln. Dabei wird sich herausstellen, dass eine niedrige Varianz in der Teilchenanzahl pro Prozessor keineswegs gleichbedeutend mit einer geringen Laufzeit zwischen den Loadbalancing-Schritten sein muss.

Als erstes werde hier untersucht, wie γ am besten adaptiv bestimmt werden

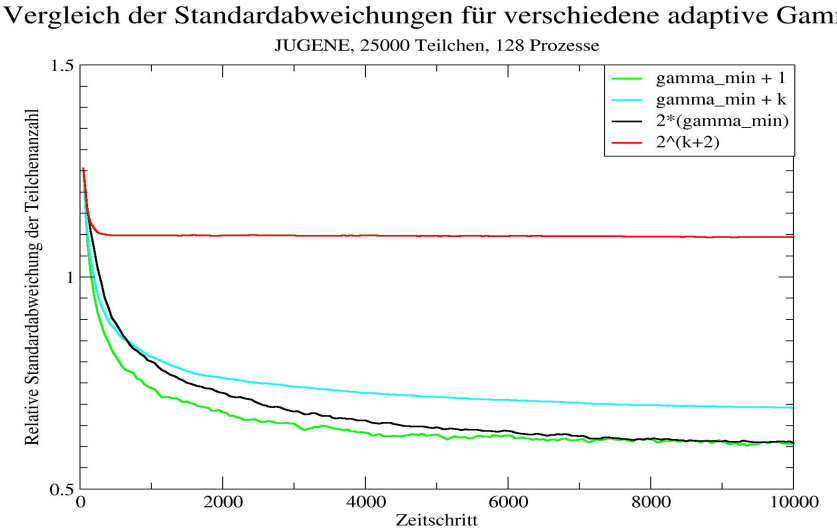

Abbildung 5.4: Vergleich der Standardabweichungen der Laufzeiten zwischen zwei Loadbalancing-Schritten für verschiedene adaptive Bestimmungen von γ. Als Arbeit wurden die Laufzeiten verwendet und das Programm auf 128 Prozessoren ausgeführt.

kann. Das Stabilitätskriterium (Formel (9)) gibt nur eine untere Grenze für γ, die angibt, wie groß γ mindestens muss, macht aber keine Aussagen darüber, wie viel viel größer es sein sollte. Daher wurde empirisch untersucht, wie sich die Varianzen für unterschiedliche Bestimmungsmethoden unterschiedlich entwickeln. γ_{min} bezeichnet dabei die untere Grenze für γ nach dem Stabilitätskriterium. Die Größe k ist die aktuelle Anzahl an Loadbalancingchritten, die Idee hinter dem Ansatz, die Summe $\gamma_{min} + k$ zu verwenden ist die lineare Vergrößerung von γ, um den Anteil der übertragenen Arbeit mit fortlaufender Dauer zu reduzieren. Noch stärker wird der Ansatz bei der Verwendung von 2^k als γ verfolgt, was einer Art Bisektion entspräche. Unabhängig vom Versuch γ mit fortlaufender Dauer der Programmausführung zusätzlich zu vergrößeren, wurden auch zwei Ansätze untersucht, in denen γ nur minimal größer ($\gamma = \gamma_{min}+1$) als die untere Schranke, bzw. das Doppelte der unteren Schranke ist. In Abbildung (5.4) ist zu sehen, dass die Varianz sich am günstigsten entwickelt, wenn man ein γ wählt, das möglichst klein ist. Zwar konvergiert die Varianz in diesen Fällen nicht so gut, wie bei $\gamma = 2^k$, allerdings ist die Varianz selbst mit den Schwankungen für kleine γ deutlich kleiner. Somit zeigt sich, dass die beste Wahl für ein adaptives γ ein solches ist, das sich möglichst minimal von der unteren Schranke unterscheidet. Nachdem diese Untersuchung eine Vorgabe für ein möglichst gutes adaptives γ geliefert hat, wird in den folgenden Untersuchungen γ adaptiv immer mit $\gamma = \gamma + 1$ gewählt.

Als nächstes wird untersucht, wie sich die Standardabweichungen von Laufzeiten und Teilchenanzahlen der einzelnen Prozesse über die Dauer der Simulation für die beiden unterschiedlichen Definitionen von Arbeit unterscheiden. Dazu wurden Simulationsläufe mit 128 und 1024 Prozessen durchgeführt. Die Ergebnisse sind in Abbildungen (5.5) und (5.6) aufgetragen. Auf den Diagrammen erkennt man, dass die Größe, die als Arbeit definiert wird, jeweils besser angepasst wird, als die andere. Wird die Laufzeit als Arbeit verwendet, so ist die Standardabeweichung der Lauflaufzeit deutlich geringer, als bei der Wahl der Teilchenanzahl als Arbeit (s. Abbildung (5.5)). Bei der Teilchenzahl verhält es sich genau umgekehrt, hier sind die Ergebnisse für die Standardabweichung der Teilchen deutlich niedriger für das Loadbalancing,

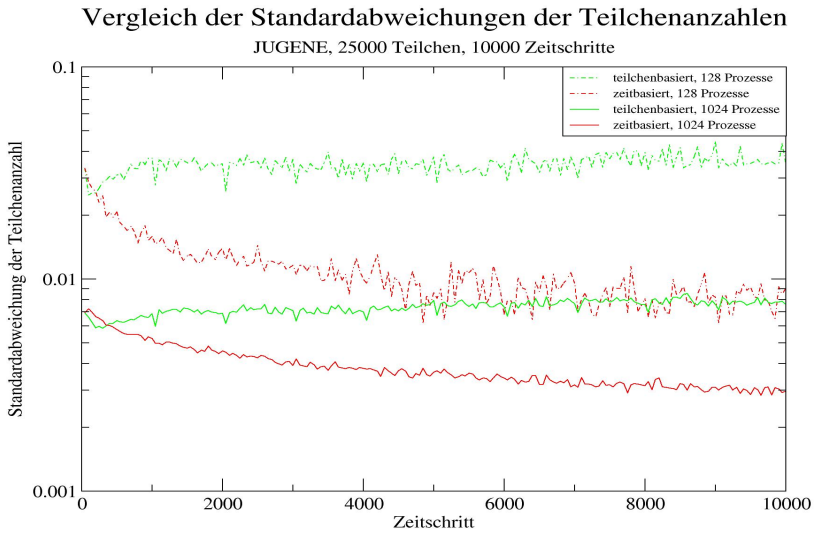

Abbildung 5.5: Vergleich der Standardabweichungen der Laufzeiten der einzelnen Prozesse zwischen zwei Loadbalancing-Schritten für verschiedene Definitionen von Arbeit, d.h. es wurde einmal Arbeit als Teilchenzahl je Prozessor definiert (particles) und das andere Mal als Laufzeit zwischen den Loadbalancing-Schritten (time) definiert. Verglichen wurden Standardabweichungen sowohl bei 128 Prozessen, als auch bei 1024 Prozessen. Alle 50 Zeitschritte wurde ein Loadbalancingschritt durchgeführt.

bei dem die Teilchenanzahl als Arbeit verwendet wird (s. Abbildung(5.6). Da aber das Bestreben ist, die Laufzeit des Programmes zu verbessern, ist nach diesen Ergebnissen die Wahl der Laufzeit als Arbeit deutlich zu bevorzugen.

Laufzeituntersuchungen

Wie in der Einführung beschrieben, wird bei der Laufzeituntersuchung unterschieden zwischen der benötigten Zeit für die Kommunikations- und Wechselwirkungroutinen, sowie die Gesamtlaufzeit des MD-Teils. Diese werden getrennt in Diagrammen darstellt und die Ergebnisse vorgestellt.

Es wird erwartet, dass die Kommunikationszeit höher ist, als im Original-

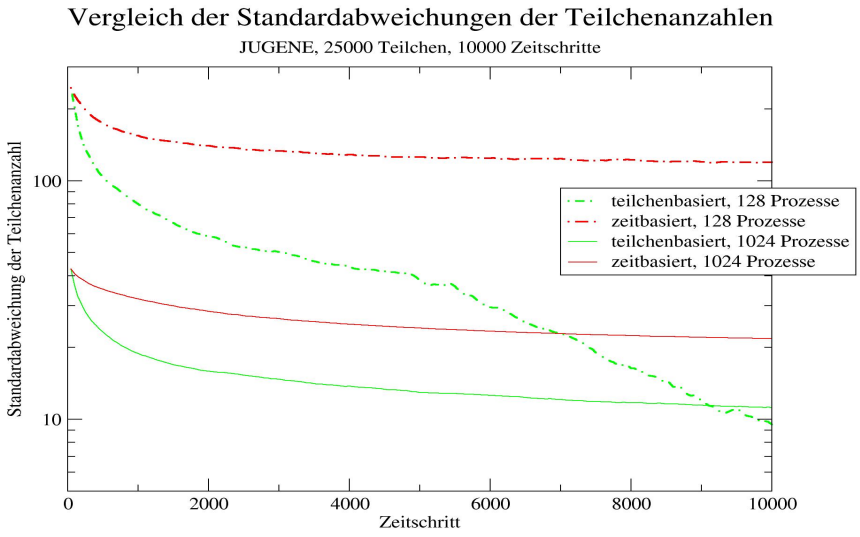

Abbildung 5.6: Vergleich der Standardabweichungen der Teilchenanzahlen auf den Prozessen für verschiedene Definitionen von Arbeit, d.h. es wurde einmal Arbeit als Teilchenzahl je Prozessor definiert (particles) und das andere Mal als Laufzeit zwischen den Loadbalancing-Schritten (time) definiert. Verglichen wurden Standardabweichungen sowohl bei 128 Prozessen, als auch bei 1024 Prozessen. Alle 50 Zeitschritte wurde ein Loadbalancingschritt durchgeführt.

programm, da durch die Vergrößerung der Anzahl der Nachbarn eine erhöhte Kommunikation zu erwarten ist. In Abbildung (5.7) sieht man, dass dies zutriff. Bei beiden Loadbalancingversionen ist die Kommunikationszeit deutlich höher, als beim Originalprogramm. Dafür gibt es mehrere Gründe. Zunächst wird durch die Verschiebung der Gittergrenzen die Anzahl der Nachbaren in y- und z-Richtung erhöht. Dadurch muss statt mit einem Nachbarn, wie beim Originalprogramm, mit deutlich mehr Nachbarn kommuniziert werden. Bei einem Faktor von zwei, das heißt dass jeder Prozess nach Verschiebung des Gitters im Durchschnitt die doppelte Anzahl an Nachbarn hat, bedeutet das schon eine doppelt so aufwendige Kommunikation. Hinzu kommen zudem noch die zusätzlichen Abfragen, die im Abschnitt über Kommunikationsschemata (3.3) behandelt wurden, die nötig sind, damit jedes Teilchen auch den

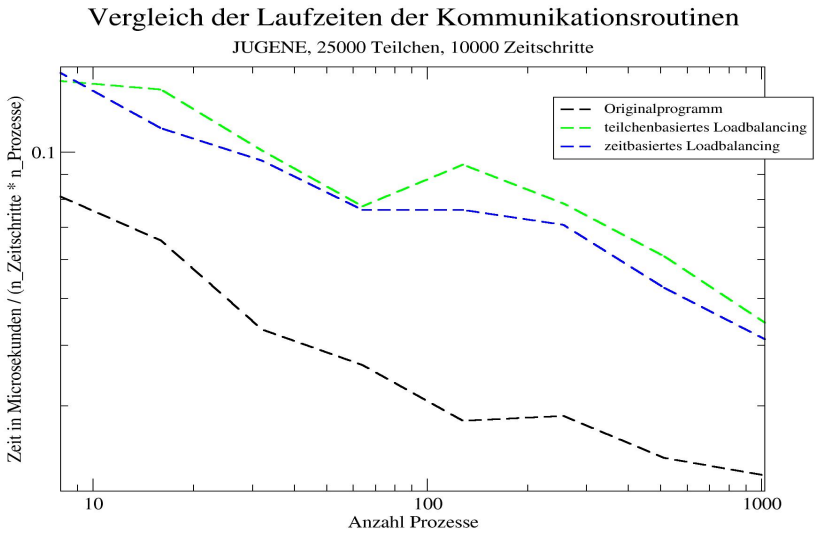

Abbildung 5.7: Vergleich der Kommunikationszeiten für unterschiedliche Loadbalancingverfahren auf JUGENE für 8 bis 1024 Prozesse.

entsprechenden Zielprozess erreicht.
Die zweite untersuchte Laufzeit ist die Zeit, die zur Berechnung der Wechselwirkungen verbraucht wird. Bei dieser Zeit fließen außer der Teilchenanzahl auf einem Gebiet auch die Dichte der Teilchen ein. Je dichter die Teilchen auf einem Gebiet sind, desto mehr Wechselwirkungen müssen bestimmt werden. Damit könnte die Tatsache zu erklären sein, dass in Abbildung (5.8) zu erkennen ist, dass die Interaktionszeit für das Teilchenanzahl basierende Loadbalancing-Verfahren nicht deutlich besser als die Originalzeit ist. Werden die Teilchen alle gleichmäßig aufgeteilt so können bei inhomogenen Systemen, wie dem betrachteten sehr große Gebiete mit kleiner Teilchendichte auftreten. Die Wechselwirkungen werden mit sogenannten “linked cells” berechnet, deren Anzahl von der Größe des Gebietes abhängig ist. Wenn nun alle Gebiete eine annähernd gleiche Teilchenanzahl haben, so kann es durch eine sich stark unterscheidene Gebietsgröße zu Schwankungen in der Laufzeit für die Wechselwirkungsberechnung kommen, da alle (auch leere) “linked cells” durchlaufen werden. Wird hingegen die Laufzeit angepasst, so wird sich

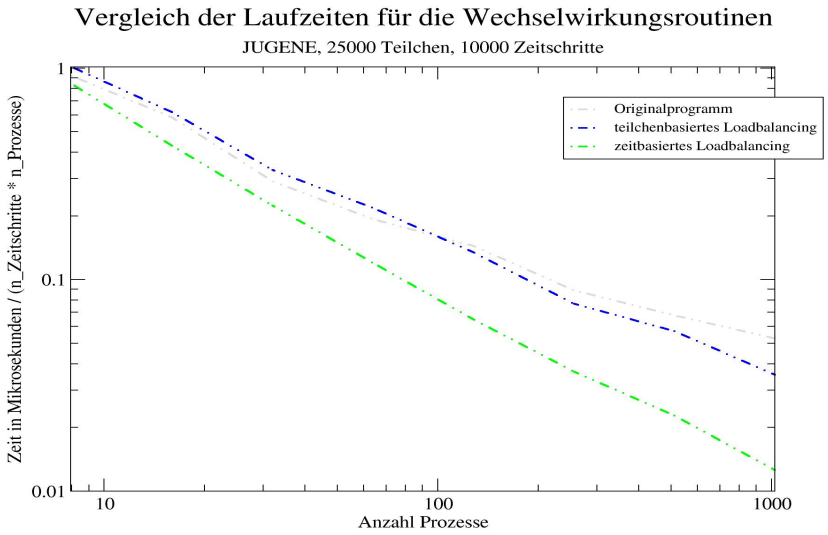

Abbildung 5.8: Vergleich der Laufzeiten für die Wechselwirkungsroutinen bei unterschiedlichen Loadbalancingverfahren auf JUGENE für 8 bis 1024 Prozesse.

ein Gleichgewicht aus Teilchenanzahl und Gebietsgröße einstellen, das in einer möglichst gleichverteilten Laufzeit für die Wechselwirkungsberechnung resultiert. Dies würde die Ergebnisse erklären und auch erklären, warum die Wechselwirkungszeit sogar noch skaliert, wenn nur noch durchschnittlich 25 Teilchen auf jedem Prozessor vorhanden sind.

Als letztes werden die Gesamtlaufzeit für den MD-Teil verglichen. In diese fließen neben Kommunikations- und Interaktionszeit noch andere Zeiten ein, wie zum Beispiel die Zeit, die benötigt wird, um in jedem Zeitschritt die Teilchen weiterzubewegen. Daher ist die Gesamtzeit größer als die Summe von Kommunikations- und Interaktionszeit. Wie in Abbildung (5.9) zu erkennen ist, ist die Gesamtzeit für das teilchenanzahlbasierte Loadbalancingverfahren größer, als beim Originalprogramm. Dies lässt sich darauf zurückführen, dass die Verbesserung der Laufzeit für die Wechselwirkungsroutinen gegen dem Originalprogramm ziemlich gering ist, während die Kommunkationslaufzeit sich dennoch vergrößert hat. Da die Einsparungen in der Wechselwirkungs-

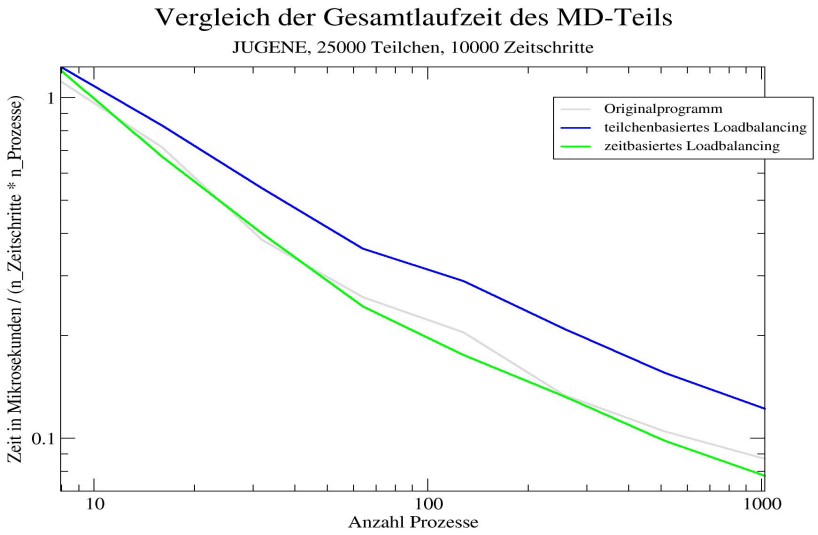

Abbildung 5.9: Vergleich der Gesamtlaufzeiten für den MD-Teil bei unterschiedlichen Loadbalancingverfahren auf JUGENE für 8 bis 1024 Prozesse.

zeit zu klein ist, ist die Folge die größere benötigte Gesamtzeit. Demhingegen ist die Laufzeit für die Laufzeit beim zeitbasierten Loadbalancing-Verfahren kleiner, als beim Originalprogramm. Allerdings ist der Unterschied nicht sehr groß. Ein Grund hierfür könnte sein, dass die Zeit, die für die Kommunikation benötigt wird, schon ab etwa 128 Prozessoren bei allen Varianten größer ist, als die Zeit, die für Wechselwirkungsberechnung benötigt wird. Dadurch ist die Einsparung, die bei der Interaktionszeit mit immer größer werdender Prozesszahl relativ gesehen immer unbedeutender. Es zeigt sich also, dass der erhöhte Kommunikationsaufwand leider die Verbesserung bei der Wechselwirkungsberechnung größtenteils kompensiert.

Um die Verbesserungen der Laufzeiten noch einmal besser darzustellen, zeigt Abbildung (5.10) die einzelnen Laufzeiten der Prozesse vor dem ersten und vor dem letzten Loadbalancing-Schritt. Dabei fällt zwar auf, dass die im Durchschnitt benötigte Laufzeit aller Prozesse zunimmt, allerdings sind die Spitzen der Laufzeiten nach oben, relativ zum Mittelwert gesehen, erheblich kleiner geworden. Da diese Spitzen aber an Barrieren im Programm gerade

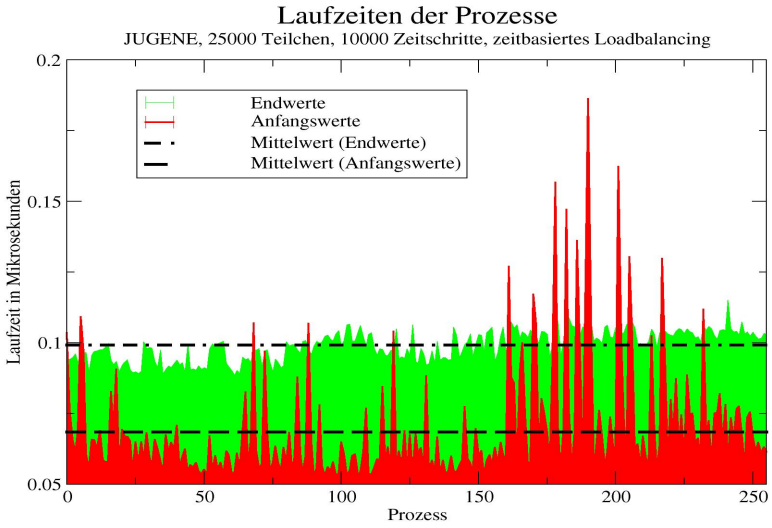

Abbildung 5.10: Laufzeiten der einzelnen Prozesse, sowie die mittlere Laufzeit aller Prozesse zu Beginn und Ende der Simulation.

die Auslöser für Wartezeiten sind, ist dies ein gutes Ergebnis.

Abbildung (5.11) zeigt als Gegenstück dazu ein Histogramm der Laufzeiten. Hierbei wird die Anzahl der Prozesse gezählt, die eine Laufzeit in einem Zeitintervall haben. Beim Histogramm der anfänglichen Laufzeiten fällt auf, dass es eine Spitze bei etwa 0.06 μs gibt, die langsam in positiver Richtung abfällt, d.h. es treten teils sehr große Laufzeiten auf den Prozessen auf. Demhingegen liegt die Spitze bei den Laufzeiten am Ende bei etwa 0.1 μs, die Streuung ist allerdings viel kleiner, so dass nur Zeiten zwischen etwa 0.08 μs und 0.12 μs auftreten, während die maximale Zeit, die bei den Anfangslaufzeiten auftritt bei etwa 0.18 μs liegt.

Als Vergleich zu den Diagrammen, die auf dem Testprogramm basieren (Abbildungen (5.1) bis (5.3)) zeigen die Abbildungen (5.12) bis (5.14) die Laufzeiten der Einzelprozesse auf ähnliche Weise aufgetragen. Die farbigen Bereiche zeigen dabei jeweils die Projektion der einzelnen z-Ebenen in die x-y-Ebene, so dass man die Aufteilung der Säulen und Zellen erkennen kann. Die Dicke der jeweiligen z-Ebenen ist unter der Farbskala relativ zur Gesamt-

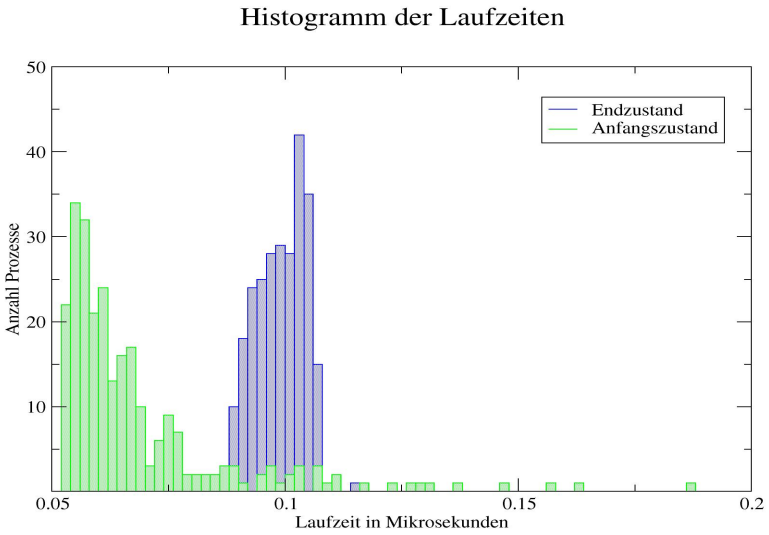

Abbildung 5.11: Histogramm über die Laufzeiten der einzelnen Prozesse, bei Anfang und Ende der Simulation

breite des Systems aufgetragen. Zu erkennen ist, dass die relative Schwankung der Laufzeiten zur mittleren Laufzeit mit fortschreitender Simulationsdauer reduziert werden. So ist in Abbildung (5.14) nur noch ein Gebiet zu sehen, dessen Laufzeit gelb dargestellt wird, was eine Abweichung der Laufzeit von mehr als 25% der mittleren Laufzeit nach oben bedeutet, während beim Anfangszustand sogar mehrere Bereiche eine rot dargestellte Laufzeit (mehr als +75% der mittleren Laufzeit) aufweisen. Dabei ist anzumerken, dass die Skala am oberen Rand gedeckelt ist, d.h. die roten Gebiete können sogar mehr als das doppelte der mittleren Laufzeit benötigen. In den Abbildungen (5.15) und (5.16) werden die Laufzeiten farblich kodiert in einer dreidimensionalen Darstellung der Gebietsaufteilung dargestellt. In dieser Darstellung sind die inneren Gebiete und deren Laufzeiten nicht so gut erkennbar, weswegen es so aussieht, als wäre die Verteilung der Laufzeiten zu Beginn der Simulation homogener als am Ende der Simulation. Auf den Abbildungen (5.12) und (5.14) ist aber zu erkennen, dass dies nicht der Fall ist und beim Endzustand die Laufzeiten deutlich homogener sind.

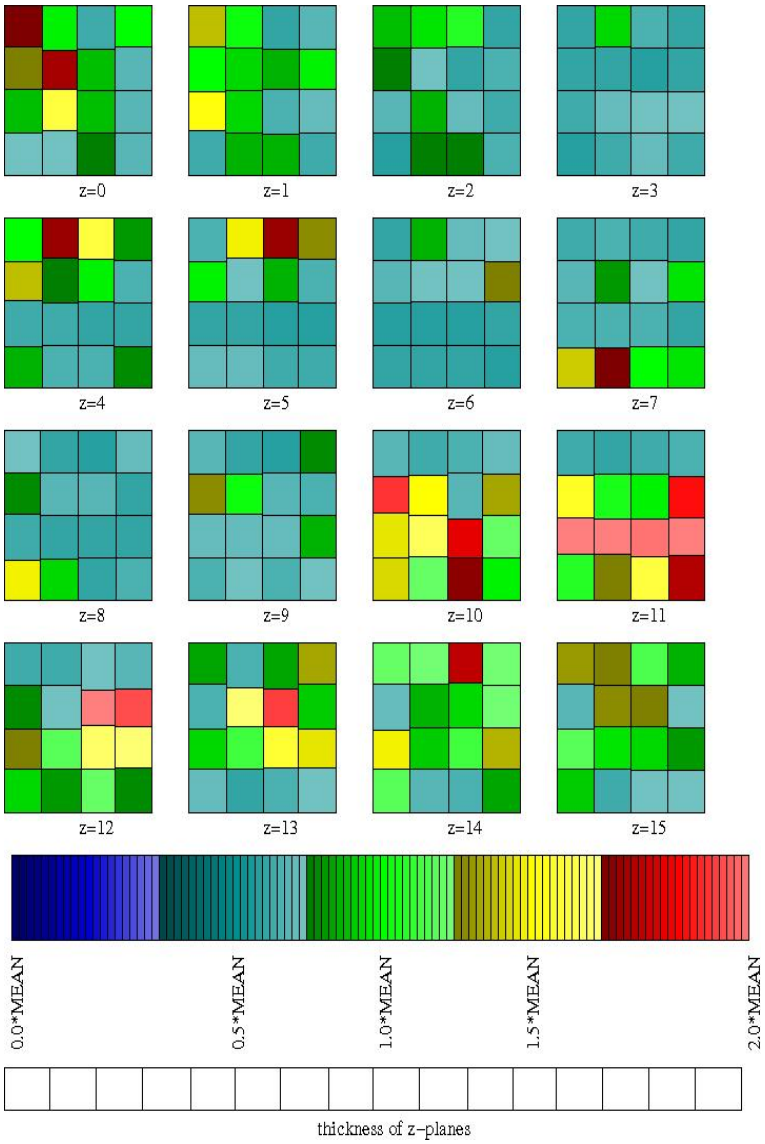

Abbildung 5.12: Die Laufzeiten in Bezug zur mittleren Laufzeit über allen Prozessen für das Testsystem bei aktiviertem, zeitbasiertem Loadbalancing. Es wird der Zustand vor dem ersten Loadbalancing-Schritt gezeigt.

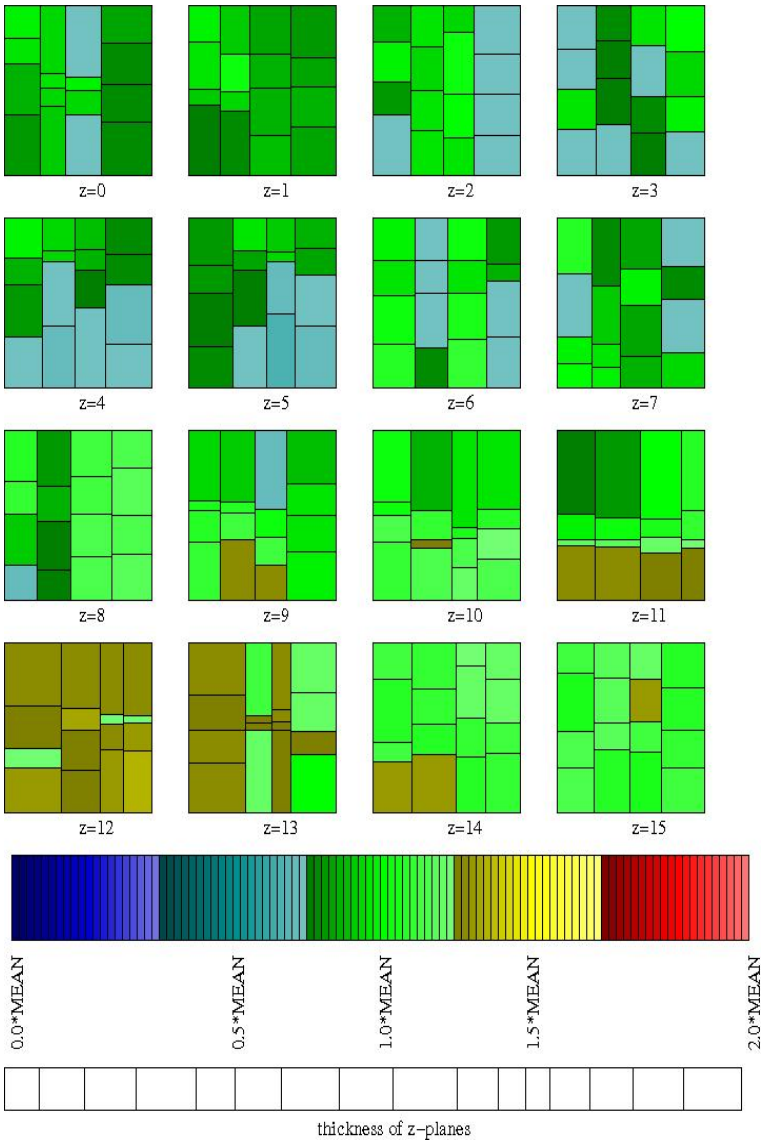

Abbildung 5.13: Die Laufzeiten in Bezug zur mittleren Laufzeit über allen Prozessen für das Testsystem bei aktiviertem, zeitbasiertem Loadbalancing. Es wird ein Zustand in der Mitte der Simulation gezeigt.

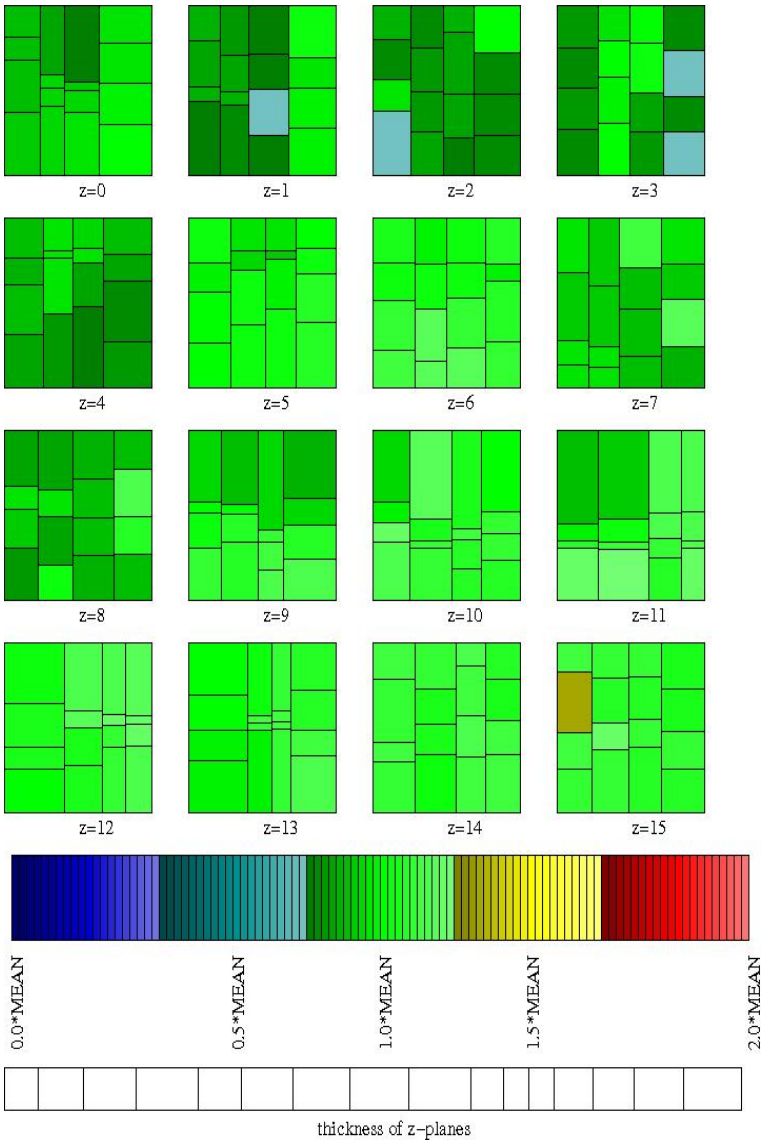

Abbildung 5.14: Die Laufzeiten in Bezug zur mittleren Laufzeit über allen Prozessen für das Testsystem bei aktiviertem, zeitbasiertem Loadbalancing. Es wird ein Zustand am Ende der Simulation gezeigt.

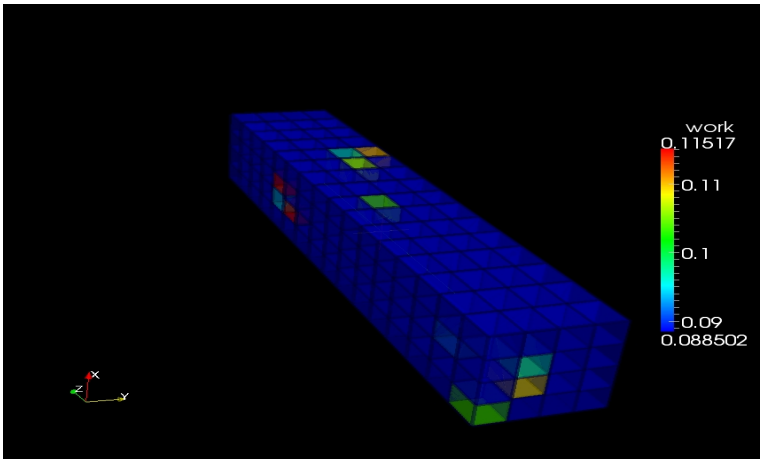

Abbildung 5.15: Dreidimensionale Darstellung der Gebietsaufteilung zu Beginn der Simulation mit farblicher Kodierung der Laufzeiten bis zum ersten Loadbalancing-Schritt. Die Zeiteinheit ist μs.

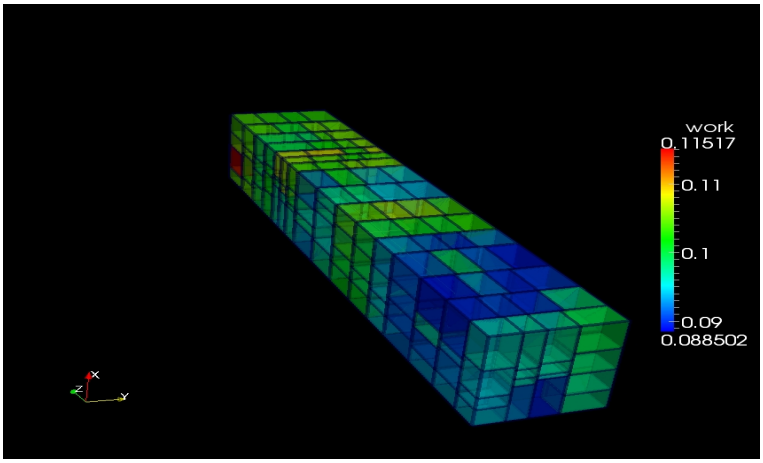

Abbildung 5.16: Dreidimensionale Darstellung der Gebietsaufteilung am Ende der Simulation mit farblicher Kodierung der Laufzeiten zwischen vorletztem und letztem Loadbalancing-Schritt. Die Zeiteinheit ist μs.

Kapitel 6

Fazit und Ausblick

Im abschließenden Kapitel erfolgt nun eine Bewertung des Verfahrens anhand der Daten, die im vorherigen Kapitel vorgestellt und untersucht wurden. Ausserdem wird kurz beschrieben, was mögliche Ergänzungen und Erweiterungen des Verfahren sein könnten, um die Qualtität zu verbessern.

6.1 Bewertung des Verfahrens

Die Ergebnisse von JUGENE liefern ein klares Bild; das Verfahren reduziert, zumindest bei Wahl der Laufzeit als Arbeitskriterium, die Zeit, die zur Berechnung der Teilcheninteraktionen benötigt wird, erheblich. Allerdings fällt auf, dass die Zeit, die für die Kommunikation benötigt wird, sich drastisch erhöht. Auf JUGENE ist die Reduktion des Zeitaufwandes für die Berechnung der Teilchenwechselwirkungen, relativ zur Gesamtlaufzeit gesehen, so gering, dass der erhöhte Kommunikationsaufwand dafür sorgt, dass beim Loadbalancing über die Teilchenanzahl pro Prozessor die Gesamtlaufzeit im MD-Teil sogar größer als das Originalprogramm ist. Dies ist sicher ein Punkt, an dem zukünftige Verbesserungen (s. Abschnitt 6.2) ansetzen müssen. Das Loadbalancing über die benötigte Zeit der Prozesse zwischen zwei Loadbalancing-Schritten hingegen liefert gute Ergebnisse. Die benötigte Zeit wurde für die Wechselwirkungsberechnung massiv reduziert. Es ist zu

erkennen, dass die Zeit bei Verwendung des Loadbalancings der Zeit sogar bis zu durchschnittlich 25 Teilchen pro Prozess, noch mit der Anzahl der Prozesse skaliert (Verwendung von 1024 Prozessen beim Testsystem mit 25000 Teilchen). Allerdings ist hier festzustellen, dass der Kommunikationsaufwand erheblich zunimmt.
Insgesamt ist festzustellen, dass das in dieser Arbeit vorgestellte Verfahren eine Beschleunigung der Gesamtlaufzeit erwirken kann, sofern die auszugleichende Arbeit sinnvoll definiert wird. Insbesondere die Ergebnisse bezüglich der Zeiten für die Wechselwirkungsroutinenen sind hervorragend, da sie wie in Abbildung (5.8) zu erkennen ist, auch für sehr geringe Teilchenzahlen je Prozess noch mit der Anzahl der Prozesse noch skaliert. Interessant ist dabei die Beobachtung, dass die Laufzeiten der einzelnen Prozesse zwar geringer werden, aber damit keine Anpassung der Teilchenanzahl einhergeht. Die Anpassung der Teilchenanzahlen führt offensichtlich zu erhöhtem Aufwand für einige Prozesse, so dass die Standardabweichungen der Laufzeiten sogar noch zunehmen (s. Abbildung (5.5)). Was allerdings gezeigt ist, ist dass das Verfahren die jeweils anzugleichende Größe gegenüber anfänglichen Inhomogenitäten gut ausgleicht. Bei den Laufzeiten werden Standardabweichungen von zuvor etwa 30% auf unter 10% , während bei der Teilchenanzahl Standardabweichungen von ca. 200% des Mittelwertes auf 5% (128 Prozesse), bzw. 80% (1024 Prozesse) bei der Verwendung des jeweiligen Loadbalancingverfahrens reduziert werden (s. Abbildungen 5.5 und 5.6). Zu den 80% Standardabweichungen sei gesagt, dass das Verfahren innerhalb der 200 Loadbalancing-Schritte noch nicht auskonvergiert ist und es bei zusätzlichen Einschränkungen, wie Mindestgrößen für Gebieten unter Umständen gar nicht möglich ist, eine noch bessere Angleichung zu erreichen, da es vorkommen kann, dass Häufung von Teilchen in einem Gebiet dieser Minimalgröße vorkommen.

6.2 Mögliche Ergänzungen und Erweiterungen

Aber das Verfahren ist bei weitem noch nicht perfekt. Es gibt einige Stellen, an denen noch Verbesserungen möglich sind, die im Rahmen dieser Arbeit

aber leider nicht mehr durchgeführt werden konnten. Drei wichtige Punkte werden im folgenden etwas ausführlicher beschrieben. Der wichtigste von diesen ist sicherlich die Optimierung der Kommunikationsroutinen, da durch die Erhöhung der Anzahl der Nachbarzellen die Kommunikation deutlich effizienter werden muss, soll die für die benötigte Zeit für diese gegenüber dem Original falls möglich gleich groß bleiben.

- Optimierung der Kommunikationsroutinen
- Anpassung des Verfahrens, um zusätzliche Kommunikation zu vermeiden
- Erweiterung des Verfahrens auf den hydrodynamischen Teil von MP2C

Da sich gezeigt hat, dass die Kommunikation offensichtlich die Qualtität des Loadbalancingverfahrens einschränkt, müssen die Kommunikationsroutinen daraufhin geprüft werden, ob sie effizienter implementiert werden können. Ein Beispiel hierfür sind die Akzeptanzkriterien, ob ein Teilchen beim Teilchentransport übernommen wird, weil es vom empfangenden Prozess weitergeleitet werden muss, nicht effizienter prüfen lassen, als dies bisher implementiert ist (vgl. 3.3.2), da in diesen Bereichen ein Großteil der Laufzeit der Kommunikationsroutinen verbracht wird.
Eine andere Möglichkeit wäre es zu überdenken, ob die Anforderung an das Verfahren, dass z-Ebenen bzw. y-Säulen disjunkt zu einander sind, nicht durch die Anforderung, dass die Anzahl der Nachbaren erhalten bleiben soll, ersetzt werden sollte. So wäre es zum Beispiel möglich Gitterpunkte, an denen Gebiete zusammenlaufen, dreidimensional im Raum zu verschieben, statt wie im hiesigen Verfahren Grenzen zwischen Gebieten eindimensional zu verschieben. Die Folge wäre, dass die Nachbarschaftsbeziehung zwischen den Gebieten immer die gleiche bleibt und jeder Prozess immer mit denselben Prozessen kommuniziert. Somit würde die Prüfung, welcher Prozess welche Nachbaren hat, entfallen und die schon angesprochenen Akzeptanzkriterien deutlich vereinfacht werden, was eine Verkürzung der Kommunikationszeit

zur Folge haben müsste. Aber für jede dieser möglichen Verfahrensanpassungen muss unabhängig von ihrer Qualität das Verfahren auf den hydrodynamischen Teil angepasst werden, da jener im Rahmen dieser Arbeit nicht berücksichtigt wurde.
Diese drei Punkte sind nur die wichtigsten der möglichen Ergänzungen und Erweiterungen, die hier stellvertretend für alle noch möglichen Anpassungen des Verfahrens durchgeführt werden können. Sicherlich kann bei weiterer Untersuchung auch noch an anderen Stellen Raum für Verbesserungen gefunden werden, daher ist diese Liste nicht als Liste der letzten möglichen Verbesserungen zu sehen, sondern nur als Hinweis auf die Gebiete, in denen auf jeden Fall noch Verbesserungen möglich sind.

Abbildungsverzeichnis

1.1 Das untersuchte Testsystem 7

1.2 Beispielhafte Endunterteilung der Gebiete 9

2.1 Skizzierte Blutbahn mit zwei nicht-periodischen und einer periodischen Randbedingung 14

2.2 Raumfüllende Kurve in einem 2D-Beispiel 18

3.1 Räumliche Aufteilung des Gesamtgebietes 24

3.2 Verschiebung der Grenzen beim Loadbalancing-Schritt 26

3.3 Aufteilung des Gebietes in Teilgebiete mit jeweiliger Arbeit und ihre Benennung 29

3.4 Erläuterung zum Krafttransport 47

5.1 Schalen-System im Anfangszustand 57

5.2 Schalen-System in einem Zwischenzustand 58

5.3 Schalen-System im Endzustand 59

5.4 Vergleich der Standardabweichungen der Laufzeiten zwischen zwei Loadbalancing-Schritten für verschiedene adaptive Bestimmungen von γ. Als Arbeit wurden die Laufzeiten verwendet und das Programm auf 128 Prozessoren ausgeführt. 61

5.5 Vergleich der Standardabweichungen der Prozesslaufzeiten bei unterschiedlicher Arbeitsdefinition 63

5.6 Vergleich der Standardabweichungen der Teilchenanzahlen auf den einzelnen Prozessen bei unterschiedlicher Arbeitsdefinition 64

5.7 Vergleich der Kommunikationszeiten für unterschiedliche Loadbalancingverfahren auf JUGENE für 8 bis 1024 Prozesse . . . 65

5.8 Vergleich der Laufzeiten für die Wechselwirkungsroutinen bei unterschiedlichen Loadbalancingverfahren auf JUGENE für 8 bis 1024 Prozesse 66

5.9 Vergleich der Gesamtlaufzeiten für den MD-Teil bei unterschiedlichen Loadbalancingverfahren auf JUGENE für 8 bis 1024 Prozesse 67

5.10 Laufzeiten der Einzelprozesse zu Anfang und am Ende der Simulation 68

5.11 Histogramm der Laufzeiten zu Anfang und am Ende der Simulation 69

5.12 Laufzeit und Aufteilung, Anfang der Simulation, Testbeispiel . 70

5.13 Laufzeit und Aufteilung, Mitte der Simulation, Testbeispiel . . 71

5.14 Laufzeit und Aufteilung, Ende der Simulation, Testbeispiel . . 72

5.15 Laufzeiten der Einzelprozesse in 3D-Ansicht (Anfang der Simulation) 73

5.16 Laufzeiten der Einzelprozesse in 3D-Ansicht (Ende der Simulation) 73

Tabellenverzeichnis

3.1 Bedingungen, die ein Gebiet erfüllen muss, um in angegebener Dimension ein Nachbar des betrachteten Gebietes zu sein. . . 43

3.2 Bedingungen, unter denen Teilchen beim Transport zwischen Gebieten übernommen werden 45

Literaturverzeichnis

[1] URL `http://www.fz-juelich.de/jsc/jugene`. – Dokumentation des Höchstleistungsrechners JUGENE der Forschungszentrum Jülich GmbH

[2] URL `http://www.fz-juelich.de/jsc/jump`. – Dokumentation des Höchstleistungsrechners JUMP der Forschungszentrum Jülich GmbH

[3] Bücker, Oliver: *Parallelisierung der Nahfeldberechnung in der schnellen Multipolmethode für stark inhomogene Teilchensysteme*, Fachhochschule Aachen, Campus Jülich, Masterarbeit, Juli 2009

[4] Burden, Richard L. ; Burden, Richard L. (Hrsg.) ; Faires, J. D. (Hrsg.): *Numerical Analysis*. 7th. Brooks/Cole, 2001

[5] Coulomb, Charles-Augustine: Premier Memoire sur l'electricite et le magnetisme. In: *Memoires de l'Acadimie Royale des Sciences des Paris*. Bachelier, 1785

[6] Fleissner, Florian ; Eberhard, Peter: Parallel load-balanced simulation for short-range interaction particle methods with hierarchical particle grouping based on orthogonal recursive bisection. In: *International Journal for Numerical Methods in Engineering* 74 (2008), S. 531–553

[7] Gibbon, P.: PEPC: Pretty Efficient Parallel-Coulomb-solver / Forschungszentrum Jülich GmbH, Zentralinstitut für Mathematik. Mai 2003. – Forschungsbericht

[8] Hess, Berk ; Kutzner, Carsten ; Spoel, David van der ; Lindahl, Erik: GROMACS 4: Algorithms for Highly Efficient, Load-Balanced,

and Scalable Molecular Simulation. In: *Journal of Chemical Theory and Computation* 3 (2008), February, Nr. 4, S. 435–447

[9] Hilbert, D.: Über eine stetige Abbildung einer Linie auf ein Flächenstück. In: *Mathematische Annalen* (1891), Nr. 38

[10] Morton, G.M.: A computer Oriented Geodetic Data Base; and a New Technique in File Sequencing / IBM Ltd. 1966. – Forschungsbericht

[11] Newton, I. ; A.Motte ; Machin, J.: *The Principia: Mathematical Principles of Natural Philosophy.* 1789

[12] Sutmann, Godehard: Classical Molecular Dynamics. In: J.Grotendorst (Hrsg.) ; D.Marx (Hrsg.) ; A.Muramatsu (Hrsg.): *Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms.* John von Neumann Institute for Computing, Jülich, 2002, S. 211–254

[13] Sutmann, Godehard ; Janoscheck, Florian: Communication and Load-Balancing of Force-Decompostion Algorithms for Parallel Molecular Dynamics. In: *Parallel Computing: Architectures, Algorithms and Applications* (2007), September

[14] Warren, Micheal S. ; Salman, John K.: *A Parallel Hashed Oct-Tree N-Body Algorithm*

[15] Warren, Micheal S. ; Salman, John K.: A portable parallel particle program. In: *Communication* Bd. 87. Mai 1995, S. 266–290

Jül-4323
April 2010
ISSN 0944-2952