KfK 4495
Januar 1989

Modelle zur Parallelisierung der Teilchenbehandlung in Particle-in-Cell Codes auf MIMD-Rechnern mit lokalem Speicher am Beispiel SUPRENUM

D. Seldner
Institut für Datenverarbeitung in der Technik

Kernforschungszentrum Karlsruhe

KERNFORSCHUNGSZENTRUM KARLSRUHE

Institut für Datenverarbeitung in der Technik

KfK 4495

Modelle zur Parallelisierung der Teilchenbehandlung in Particle-in-Cell Codes auf MIMD-Rechnern mit lokalem Speicher - am Beispiel SUPRENUM*

David Seldner

* Das diesem Bericht zugrundeliegende Vorhaben wurde aus Mitteln des Bundesministers für Forschung und Technologie unter dem Förderkennzeichen ITR8502K/4 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kernforschungszentrum Karlsruhe GmbH, Karlsruhe

Kernforschungszentrum Karlsruhe GmbH
Postfach 3640, 7500 Karlsruhe 1

ISSN 0303-4003

Modelle zur Parallelisierung der Teilchenbehandlung in Particle-in-Cell Codes auf MIMD-Rechnern mit lokalem Speicher - am Beispiel SUPRENUM

Zusammenfassung

Wegen der hohen Rechenzeiten von Particle-in-Cell-Programmen müssen Wege zur Beschleunigung der Programme gesucht werden. Eine mögliche Vorgehensweise liegt in der Parallelisierung. In diesem Artikel werden zwei Strategien zur Implementierung des kostenintensivsten Teils, der Behandlung der Simulationsteilchen, auf dem Parallelrechner SUPRENUM vorgestellt. Außerdem werden Modelle entwickelt, um die dadurch anfallende Verwaltung und die Kommunikation abzuschätzen. Nach einem Vergleich dieser beiden Strategien wird an zwei konkreten Beispielen der durch eine Parallelisierung zu erwartende Speed-up abgeschätzt.

Models for Parallelization of the Particle Treatment in Particle-in-Cell Codes for MIMD-computers With Local Memory - Example SUPRENUM

Summary

Particle-in-Cell Codes are very expensive in computer-time, so that an acceleration in order to reduce computer time is desired. One possibility lies in parallelizing the code. This paper deals with two strategies for an implementation of the most time expensive parts of the code, the particle treatment, on the SUPRENUM-machine. Models are developed in order to estimate the necessary cost of administration and communication. After a comparison of the two strategies the expected speed-up of a parallelization is estimated for two special cases.

Inhaltsverzeichnis

1. Einleitung

Seit 1985 wird im Kernforschungszentrum innerhalb des Arbeitsgebiets „Physik hochverdichteter Materie" an der Entwicklung von elektromagnetischen Particle-in-Cell-(PIC) Codes gearbeitet. Mit diesen PIC Codes werden Ionen-Dioden modelliert, um ein besseres Verständnis für die in der Diode ablaufenden Prozesse zu erzielen und sie zu optimieren. Zur Zeit existiert eine auf rand-angepaßten Koordinaten basierende zweidimensionale quasi-stationäre Version [3,4,9,16], eine Weiterentwicklung auf 2.5 und später drei Dimensionen ist geplant wie auch ein voll elektromagnetischer Code. Für die Parameterstudien bzgl. der Geometrie zur Optimierung der Dioden müssen Simulationen mit verschiedenen Anoden-Kathoden-Spaltabständen, angelegten Spannungen, etc. durchgeführt werden. Da ein einzelner Lauf des zweidimensionalen quasi-stationären Codes in den von uns gerechneten Beispielen etwa 10 CPU-Stunden kostet, müssen Wege zur Beschleunigung des Programms gesucht werden. Neben der Vektorisierung einzelner Programmodule [1,3,14,15] bietet sich die Implementierung auf Parallelrechnern an. Daher werden im Rahmen des SUPRENUM-Projekts [5,11,12] Teile des Codes parallelisiert [1,3,4,9,14]. Es ist geplant, sowohl die Feldberechnungen als auch die Behandlung der Simulationsteilchen zu parallelisieren.

Ziel dieser Arbeit ist es, den Rechen- und Kommunikationsaufwand für die Behandlung der Simulationsteilchen für zwei verschiedene Parallelisierungsstrategien abzuschätzen. Nach einem kurzen Überblick über das SUPRENUM-Projekt skizzieren wir den Aufbau des Particle-in-Cell Codes und stellen das zu parallelisierende Programm und die bei der Parallelisierung zu beachtenden Rahmenbedingungen vor. Anschließend erläutern wir die beiden Parallelisierungsstrategien und schätzen den benötigten Rechen- und Kommunikationsaufwand ab. Anhand zweier konkreter Beispiele wird schließlich der durch die Parallelisierung zu erwartende Gewinn an Rechenzeit abgeschätzt.

2. Das SUPRENUM-Projekt

Seit 1985 wird vom Bundesministerium für Forschung und Technologie die Entwicklung eines Hochleistungsrechner speziell für rechenintensive Aufgaben vor allem in der numerischen Simulation gefördert. Der als MIMD-Parallel-Rechner

(MIMD = „Multiple-Instructions - Multiple-Data") entwickelte Rechner SUPRENUM soll mit zunächst 256, später vielleicht mehr, parallel arbeitenden Prozessoren ausgestattet werden. Jeweils bis zu 16 Prozessoren, von denen jeder Skalar- und Vektorprozessoren sowie lokale Speicher von jeweils 8 MByte enthält, bilden einen Cluster. Datenaustausch zwischen den einzelnen Prozessoren ist nur durch den Austausch von Nachrichten möglich. Innerhalb und zwischen den Clustern wird der Datenaustausch über ein zweistufiges, schnelles Bussystem abgewickelt. Die angestrebte Spitzenleistung der 256-Prozessoren-Konfiguration ist 4 GFLOPS. Für eine weitere Beschreibung der zugrundeliegenden Konzepte verweisen wir auf [5,11,12]. Wir skizzieren kurz den Ablauf eines parallelen Programmes:

Zu Beginn der Rechnung wird ein Programm auf dem Betriebsrechner gestartet. Dieses Programm heißt „initialer Prozeß" oder Hauptprozeß. Von diesem Prozeß aus können weitere Knotenprozesse gestartet („kreiert") werden. Jeder Prozeß kann neue Prozesse kreieren, jedoch nur sich selbst und die von ihm kreierten Prozesse terminieren.

Ein Prozeß arbeitet die in einem Programm stehenden Befehle ab, wobei ein solches Programm eine Folge von üblichen FORTRAN-77-Befehlen ist, zusätzlich der Vektorkonstrukte und der Kommunikationsbefehle. Jeder neu erzeugte Prozeß erhält eine eigene Mailbox, in die eingehende Nachrichten, die von anderen Prozessen abgeschickt wurden, abgelegt werden.

Um eine größtmögliche Portabilität zu ermöglichen, wird eine Kommunikationsbibliothek [6] eingerichtet. Damit soll erreicht werden, daß die SUPRENUM-spezifischen Konstrukte in Unterprogrammen zur Verfügung gestellt werden, damit das Programm möglichst einfach auf andere Parallelrechner übertragen werden kann, falls dort diese Kommunikationsbibliothek ebenfalls implementiert ist.

3. Der Particle-in-Cell Code

3.1. Struktur des PIC Codes

Abb. 1 gibt einen Überblick über Aufbau und Ablauf des Codes. Wir beschreiben kurz die Funktion der einzelnen Module, für eine algorithmische Beschreibung verweisen wir auf [1,3,4,10,15]:

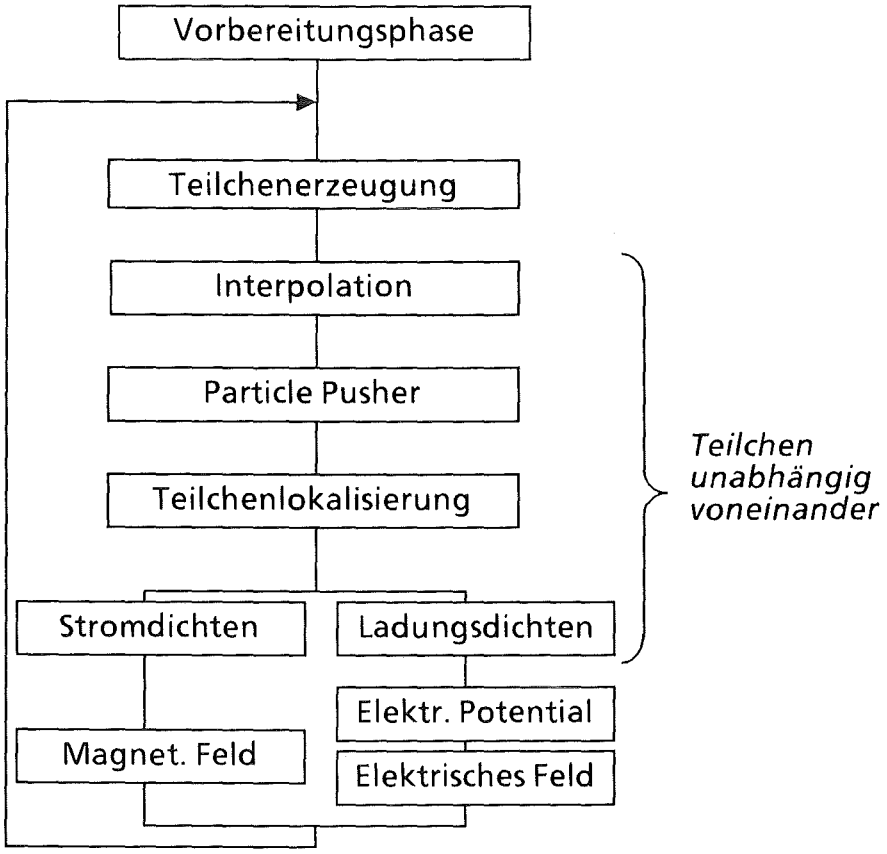


Abb. 1: Ein Zeitschritt eines Particle-In-Cell-Codes.

Nach einer Vorbereitungsphase werden aufgrund der herrschenden Felder neue Teilchen an den Elektroden erzeugt *(Teilchenerzeugung)*. Die an den Gitterpunkten bekannten Feldstärken werden an die Teilchenorte interpoliert *(Interpolation)* und die Teilchen fortbewegt *(Particle pusher)*. Die Teilchen werden im Gitter lokalisiert *(Lokalisierung)*, um dann die neuen Ladungs- und Stromdichten an den Gitterpunkten bestimmen zukönnen *(Dichten)*. Mit Hilfe dieser Daten können dann die neuen elektrischen und magnetischen Felder berechnet werden *(Feldberechnungen)*. Dann beginnt mit der Erzeugung neuer Teilchen der nächste Zeitschritt.

3.2. Benötigte Variable

Bei dem auf Zylinderkoordinaten basierenden zweidimensionalen PIC Code werden u.a. folgende Daten verwendet:

Je Gitterpunkt drei Feldstärken: E_z, E_r (z- und r-Komponente des elektrischen Feldes), B_Φ (Φ-Komponente des magnetischen Feldes),

die Teilchendatenmatrix, die pro Teilchen aus vier Daten besteht: Zwei Ortskoordinaten r,z, zwei Geschwindigkeiten v_r, v_z. Außerdem die Ladung q des Simulationsteilchens und zwei Interpolationsgewichte α_1, α_2, die die Lage im Gitter kennzeichnen.

Das Programm ist so ausgelegt, daß mehrere Teilchensorten (Elektronen, Ionen) berücksichtigt werden können. Um den Kommunikationsaufwand abzuschätzen, beschränken wir uns auf eine Teilchensorte.

Es sei im folgenden:

NDT	Zahl der durchgeführten Zeitschritte
IZMX,IRMX	Zahl der Gitterpunkte in Z- bzw. R- Richtung
Z,R	Koordinaten der Gitterpunkte
NDRZE	Zahl der Elektrodenzellen
NPE	Aktuelle Anzahl der Teilchen
NPEN	Zahl der neu erzeugten Teilchen
NPROC	Zahl der beschäftigten Prozesse
NPPSN	Zahl der neu erzeugten Teilchen je Prozess
EZ, ER, BPHI	Feldstärken an den Gitterpunkten
RHO, AJZ, AJR	Dichten an den Gitterpunkten
QEZ	Ladung in Elektrodenzellen
PE	Teilchendatenmatrix
Q	Ladung der Teilchen
GR, GZ	Interpolationsgewichte der Teilchen
PEN	Teilchendatenmatrix der neu erzeugten Teilchen
QN	Ladung der neu erzeugten Teilchen
GRN, GZN	Interpolationsgewichte der neu erzeugten Teilchen

3.3. Schnittstellen im PIC Code

Wir listen die Schnittstellen der einzelnen Module auf:

Teilchenerzeugung:
Eingabe: Je Elektrodenzelle eine Ladung (QEZ).
Ausgabe: Neu erzeugte Teilchen (PEN,QN,GZN,GRN).

Interpolation:
Eingabe: Je Gitterpunkt 3 Feldstärken (EZ, ER, BPHI).
Je Teilchen 2 Interpolationsgewichte (GZ, GR).
Ausgabe: Je Teilchen drei Feldstärken (EZP, ERP, BPHIP).

Particle pusher:
Eingabe: Je Teilchen drei Feldstärken, Orte und Geschwindigkeiten (EZP, ERP, BPHIP, PE).
Ausgabe: Je Teilchen Orte und Geschwindigkeiten (PE).

Lokalisierung:
Eingabe: Je Teilchen zwei Ortskoordinaten, Gitterkoordinaten (PE, Z,R).
Ausgabe: Je Teilchen zwei Interpolationsgewichte (GZ, GR).

Ladungs- und Stromdichten:
Eingabe: Je Teilchen zwei Interpolationsgewichte, eine Ladung (GZ, GR, Q).
Ausgabe: Je Gitterpunkt drei Dichten, je Elektrodenzelle eine Ladung (RHO, AJR, AJZ, QEZ).

Felder:
Eingabe: Je Gitterpunkt drei Dichten, Gitterkoordinaten (RHO, AJR, AJZ, Z,R).
Ausgabe: Je Gitterpunkt drei Feldstärken (EZ, ER, BPHI).

In Tabelle 1 sind die prozentualen Anteile der einzelnen Module an der Gesamtrechenzeit aufgelistet [4]. Dabei ist zu beachten, daß die Rechenzeiten stark von den Steuerparametern abhängt.

Für jedes Teilchen werden die Felder von den Gitterpunkten auf die Teilchenpositionen interpoliert, dann werden seine neuen Daten (Ort und Geschwindigkeit) bestimmt, sowie seine Lage im Gitter. Mit diesen Daten kann dann seine Ladung auf das Gitter verteilt werden. Erst bei den Feldberechnungen ist es notwendig, *alle* Dichten zu kennen. Wegen dieser Unabhängigkeit bei der numerischen Beschreibung der Teilchen bietet es sich an, die Module en bloc zu parallelisieren. Wir wollen uns daher nur mit den Modulen Interpolation, Particle Pusher, Lokalisierung, Ladungs- und Stromdichten beschäftigen., Diese Module

Programm-Modul	Rechenzeit
Feldberechnungen	6,7%
Interpolation	8,5%
Teilchenbewegung	40,6%
Lokalisierung	27,5%
Ladungs- und Stromdichten	15,6%
Sonstige	1,1%
Summe	100,0%

Tabelle 1:
Rechenzeiten der einzelnen Module auf dem Vektorrechner SIEMENS VP50 *(1 000 Zeitschritte, maximal ca. 20 000 Elektronen und 7 400 Ionen)*

wollen wir im folgenden einfach „Teilchenbehandlung" nennen. Wir betrachten die Schnittstellen der Teilchenbehandlung:

Schnittstellen

a) Eingabe:

Als Eingabedaten werden - abgesehen von in diesem Zusammenhang vernachlässigbaren Kontrollvariablen - Feldstärken, Gitterkoordinaten, und die Teilchendaten benötigt:

EZ, ER, BPHI, Z, R, PE,PEN.

b) Ausgabe

Nach der Teilchenbehandlung müssen Ladungs- und Stromdichten zurückgegeben werden sowie Ladungen in Elektrodenzellen:

RHO, AJR, AJZ, QEZ.

4. Parallelisierung

Wegen der ungleichmäßigen räumlichen Verteilung der Teilchen in der Diode wurde zunächst von einer räumlich abhängigen Zuweisung (Aufspaltung des Berechnungsgebietes) von Teilchen an einzelne Prozesse abgesehen [9]. Vielmehr wurde versucht, jedem beschäftigten Prozeß eine möglichst gleichgroße Anzahl von Teilchen zuzuordnen (unabhängig von der Lage im

Berechnungsgebiet) mit dem Ziel, jeden Prozeß gleich lange arbeiten zu lassen (Strategie I).

Dem Vorteil, daß die Prozesse keine Daten (speziell Teilchen) untereinander austauschen müssen und der Verwaltungsaufwand erheblich geringer ist, steht der Nachteil gegenüber, daß alle Prozesse gleichzeitig viele Daten an einen einzelnen Prozeß (gewissermaßen dem Koordinator) senden müssen. Außerdem müssen nach der Berechnung der neuen Felder die Prozesse mit Informationen über das gesamte Gitter versorgt werden. Darüberhinaus dürfte sich wegen der Gleichzeitigkeit der zu sendenden bzw. zu empfangenden Nachrichten ein Kommunikationsstau bilden. Daher wurde mit Untersuchungen begonnen, ob eine Aufspaltung des Gebiets auf die Prozesse sinnvoll ist (Strategie II). Dabei ist beabsichtigt, das Berechnungsgebiet in Streifen aufzuteilen und jeden Prozeß die darin befindlichen Teilchen behandeln zu lassen (wobei Überlappungen der Teilgebiete vorhanden sein müssen). Sobald ein Prozeß überlastet ist, wird ein weiterer das gleiche Gebiet bearbeitender Prozeß erzeugt. Teilchen, die ein Teilgebiet verlassen, werden an den entsprechenden (oberen oder unteren) Nachbarn geschickt.

Wir nehmen an, daß bei der Parallelisierung alle Prozesse gleich gut ausgelastet sind. Diese Annahme verschafft natürlich vor allem der zweiten Strategie, der Aufteilung des Gebiets, einen Vorteil. Inwieweit dieser Vorteil ins Gewicht fällt, hängt ab von dem Verhältnis zwischen Kommunikations- und Rechenaufwand. Es geht uns jedoch darum, ein Maß für den für Strategie II zusätzlichen Verwaltungsaufwand bzw. Kommunikationsaufwand zu erhalten.

4.1 Vereinfachende Annahmen

Bei der Abschätzung des Rechen- und Kommunikationsaufwandes gehen wir von folgenden vereinfachenden Annahmen aus:

- Es wird nur mit einer Teilchensorte gerechnet.
- Der Initialisierungsaufwand kann vernachlässigt werden, d.h. es sind bereits genügend Teilchen vorhanden, um die Prozesse auszulasten.
- Es werden jeweils so viele Daten verschickt, daß die Versendung einzelner Variabler in Bezug auf die Länge der Nachricht vernachlässigbar ist.

- Es wird vorausgesetzt. daß der Kommunikationsaufwand zwischen allen Knoten gleich groß ist.

Weitere Annahmen sind unter den jeweiligen Strategien aufgelistet.

4.2 Notation

Bei der Abschätzung der Rechenzeiten setzen wir

T_{OP}	Zeit für das Ausführen einer Gleitpunktoperation (Addition, Multiplikation)
T_{IF}	Zeit für das Ausführen einer logischen Abfrage (Vergleich mit 0).

Zur Abschätzung der Kommunikation verwenden wir folgende Notation:

T_{START}	Durchschnittliche Start-up Zeit (unabhängig von der Länge der Datenliste) für den Aufbau einer Kommunikation zwischen zwei Knoten.
T_{BUS}	Zeit, die benötigt wird, um ein Wort (8 Byte) von der Mailbox des sendenden Prozesses in die Mailbox des empfangenden Prozesses zu transferieren, nachdem die Verbindung aufgebaut ist.

Beispiel

Prozeß 1 sende eine Nachricht der Länge 8 Byte an Prozeß 2. Für den Aufbau des Kommunikationsweges wird die Zeit T_{START} benötigt, wobei Prozeß 1 in dieser Zeit Berechnungen durchführen kann. Die Nachricht ist T_{BUS} unterwegs, so daß Prozeß 2 die Nachricht frühestens nach der Zeit $T_{START} + T_{BUS}$ empfangen kann.

5. Strategie I: Aufspaltung der Teilchen auf die Prozesse

5.1 Aufbau des Programms

In einer Vorbereitungsphase werden NPROC Prozesse kreiert und mit benötigten Daten wie den Gitterkoordinaten versehen. Zu Beginn eines Zeitschritts, nachdem im Hauptprozeß neue Teilchen erzeugt worden sind, werden die Teilchen

und die neuen Felder an alle Prozesse verschickt. Der am wenigsten ausgelastete Prozeß erhält die meisten neuen Teilchen, um so eine möglichst gleichmäßige Auslastung der Prozesse zu ermöglichen. Jeder Prozeß bearbeitet seine ihm zugeteilten Teilchen (d.h., sortiert seine neuen Teilchen ein, interpoliert die Felder auf die Teilchenorte, bewegt die Teilchen fort und verteilt die Ladungen auf die Gitterpunkte). Schließlich schickt er die errechneten Dichten an den Hauptprozeß, der alle erhaltenen Dichten aufsummiert, die neuen Felder berechnet und mit dem nächsten Zeitschritt fortfährt. Wir nehmen vereinfachend an, daß ein Prozeß alle neu erzeugten Teilchen erhält. Dies verringert den Verwaltungsaufwand erheblich, während diese Annahme bei der Betrachtung der Kommunikation äquivalent ist zu der Annahme, daß alle Prozesse gleich viele Teilchen erhalten (die Anzahl der Kommunikationen wie auch die Summe der Längen sind gleich).

5.2 Kommunikationsaufwand

Die Tatsache, daß jeder Prozeß nur mit dem Hauptprozeß kommuniziert, führt bei der Abschätzung der Kommunikation zu einer wesentlichen Vereinfachung.

a) Empfangene Nachrichten:

Jeder Prozeß erhält 3*IZMX*IRMX Gitterdaten (zwei elektrische und eine magnetische Feldstärke pro Gitterpunkt) und 7*NPPSN Teilchendaten, bei REAL*8 - Genauigkeit also eine Nachricht der Länge

$$L^I_E := 3*IZMX*IRMX + 7*NPPSN .$$

b) Gesendete Nachrichten:

Gesendet werden 3*IZMX*IRMX Dichten sowie NDRZE Ladungen, also

$$L^I_S := 3*IZMX*IRMX + NDRZE .$$

5.3 Abschätzung der Kommunikationszeit:

Der Hauptprozeß sendet der Reihe nach die Daten an alle Prozesse. Dann muß er warten, bis er vom letzten Prozeß die Ergebnisse erhalten hat. Es müssen dazu nacheinander NPROC Verbindungen aufgebaut werden, dies beansprucht die Zeit $NPROC*T_{START}$; der letzte Prozeß erhält seine Daten nach der Zeit $NPROC*T_{START} + L^I_E*T_{BUS}$ und kann mit seinen Berechnungen beginnen. Wenn

wir mit T^I_{PROC} die in einem Prozeß benötigte Rechenzeit bezeichnen, so treffen die Daten also nach weiteren $T^I_{PROC} + T_{START} + L^I_S * T_{BUS}$ beim Hauptprozeß ein. Pro Zeitschritt ergibt sich also für den Hauptprozeß die für die Teilchenbehandlung notwendige Wartezeit von

$$T_1 = (NPROC + 1) * T_{START} + (L^I_E + L^I_S) * T_{BUS} + T^I_{PROC} + T^I_{HAUPT}.$$

wobei T^I_{HAUPT} die für die Parallelisierung im Hauptprozeß notwendige Verwaltungsarbeit sei .

5.4 Abschätzung von T^I_{HAUPT}:

Es gilt die benötigte Zeit für den Verwaltungsaufwand (Überprüfen der Auslastung der einzelnen Prozesse, Bestimmung des Prozesses, an den die Teilchen gesendet werden sollen, Addieren der erhaltenen Daten) abzuschätzen.

Ein FORTRAN-77-Programm zum Finden des am wenigsten ausgelasteten Prozesses IPROC (es sei NPE(I) die Anzahl der von Prozeß I bearbeiteten Teilchen) sieht etwa wie folgt aus:

```
      NPEMIN = -100000
      DO 1 I=1,NPROC
      IF ( NPE(I) - NPEMIN .LT. 0 ) THEN
         NPEMIN = NPE(I)
         IPROC = I
      ENDIF
1     CONTINUE
```

Um die Teilchen auf die Prozesse zu verteilen sind also etwa

NPROC

arithmetische Operationen (nämlich die Berechnung von NPE(I) - NPEMIN) und

NPROC

Abfragen nötig.

Um die erhaltenen Daten zu addieren müssen für NPROC-1 Prozesse in jedem Gitterpunkt drei Dichten und in jeder Elektrodenzelle eine Ladung addiert werden, also

(NPROC-1)*(3*IZMX*IRMX + NDRZE)

Operationen. Insgesamt werden in jedem Zeitschritt also etwa

(NPROC-1)*(3*IZMX*IRMX + NDRZE) + NPROC

Operationen und

NPROC

Abfragen benötigt. Dies ergibt als Zeit für den Verwaltungsaufwand

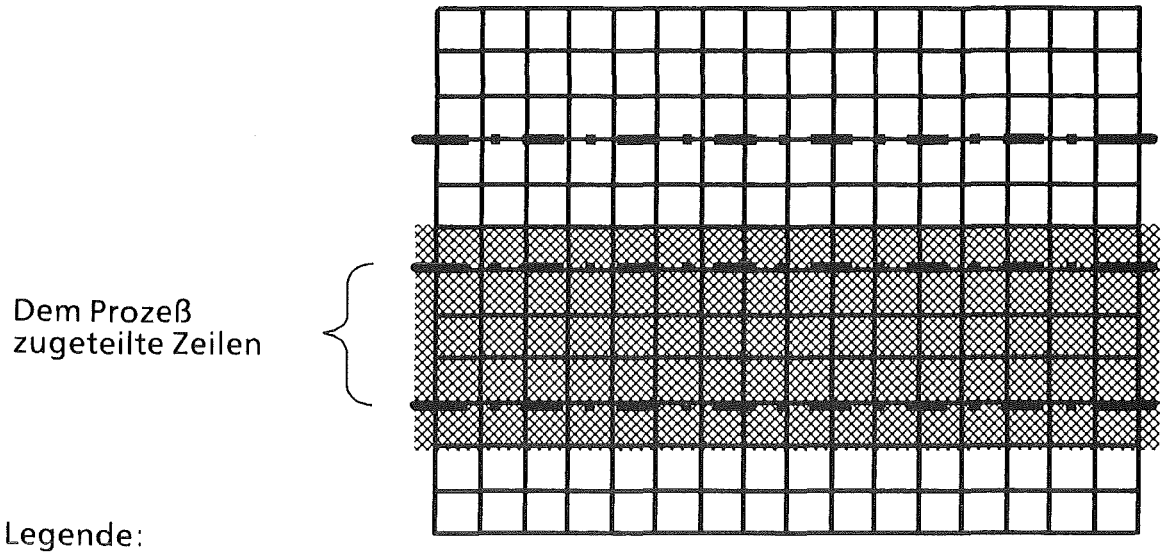
$$T^{I}_{HAUPT} = ((NPROC-1)*(3*IZMX*IRMX + NDRZE) + NPROC)*T_{OP} + NPROC*T_{IF}$$

6. Strategie II: Aufspaltung des Gebiets in Zeilen

6.1 Aufbau des Programms

In der Vorbereitungsphase dieser Strategie wird das Gebiet in LINES ($\leqq$NPROC) Teilgebiete (Streifen) aufgeteilt, die Prozesse kreiert und mit Daten wie den benötigten Gitterkoordinaten versehen. Diese Art der Aufteilung des Gebiets wurde gewählt, da sie sowohl den Kommunikationsaufwand innerhalb der Feldberechnungen wie auch beim Wechsel zwischen Teilchenbehandlung und Feldberechnung minimiert [2]. Zu Beginn eines Zeitschritts, nachdem im Hauptprozeß neue Teilchen erzeugt worden sind, werden die Teilgebiete bestimmt, in denen sich die Teilchen befinden, damit sie zusammen mit den in den Teilgebieten herrschenden Feldern an die zuständigen Prozesse verschickt werden können. Zu den am meisten ausgelasteten Prozessen werden (nach jeweils NIDENT Zeitschritten) „identische" Prozesse erzeugt, um die Arbeit aufzuteilen. Falls mehrere Prozesse das gleiche Teilgebiet bearbeiten, so erhält der am wenigsten ausgelastete die Teilchen, um so eine möglichst gleichmäßige Auslastung der Prozesse zu ermöglichen. Jeder Prozeß bearbeitet seine ihm zugeteilten Teilchen (d.h., sortiert seine neuen Teilchen ein, interpoliert die Felder auf die Teilchenorte, bewegt die Teilchen fort und verteilt die Ladungen auf die Gitterpunkte). Diejenigen Teilchen, die das bearbeitete Teilgebiete verlassen haben und in ein anderes übergewechselt sind, werden dem zuständigen Nachbarn übergeben (wenn mehrere existieren, so erhält der zu Beginn kreierte die Teilchen). Bis er sie versenden kann, muß er sie jedoch weiterbearbeiten können. Daher müssen Überlappungsgebiete eingeführt werden, jeweils eine Zeile oben und unten (siehe Abb. 2).

Die in den Überlappungsgebieten berechneten Ladungs- und Stromdichten werden den Nachbarn zusammen mit den Teilchen übergeben. Schließlich



Legende:

: Zu bearbeitendes Gebiet (mit Überlappungen)

Abb. 2:Aufteilung eines Gebiets in Streifen

schicken die Prozesse die Dichten an den Hauptprozeß, der alle erhaltenen Dichten aufsummiert, die neuen Felder berechnet und mit dem nächsten Zeitschritt fortfährt. Um eine möglichst gleichmäßige Auslastung der Prozesse zu gewährleisten, wird weiterhin alle NIDENT Zeitschritte die Auslastung der Prozesse überprüft, und gegebenenfalls veranlaßt, daß ein Prozeß Teilchen an einen identischen Prozeß abgibt. Der zeitliche Ablauf ist in Abb. 3 schematisch dargestellt.

6.2 Bezeichnungen und vereinfachende Annahmen:

Wie schon oben erwähnt, gehen wir davon aus, daß jeder Prozeß gleichviele Teilchen behandelt. Dies kann zumindest näherungsweise dadurch erreicht werden, daß Teilgebiete, in denen mehr Teilchen vorhanden sind, von mehreren identischen Prozessen bearbeitet werden. Wir beschränken uns bei der Betrachtung von Prozessen auf solche, die keine Randgebiete behandeln, untersuchen also Prozesse mit oberen und unteren Nachbarn. Diese Betrachtung ist realistisch, da ja die am längsten beschäftigten Prozesse die Rechenzeit bestimmen, und dies sind diejenigen, die mehr Kommunikation ausführen.

Weiterhin nehmen wir an, daß zu jedem Prozeß genau ein identischer Prozeß existiert. Insbesondere heißt das, daß das gesamte Gebiet in NPROC/2 Teilgebiete aufgespalten ist, von denen wir voraussetzen, daß sie alle aus gleich vielen Gitterpunkten bestehen. Jeder Prozeß, der kein Randgebiet behandelt, hat also zwei obere und zwei untere Nachbarn. Alle NIDENT Zeitschritte werden Teilchen

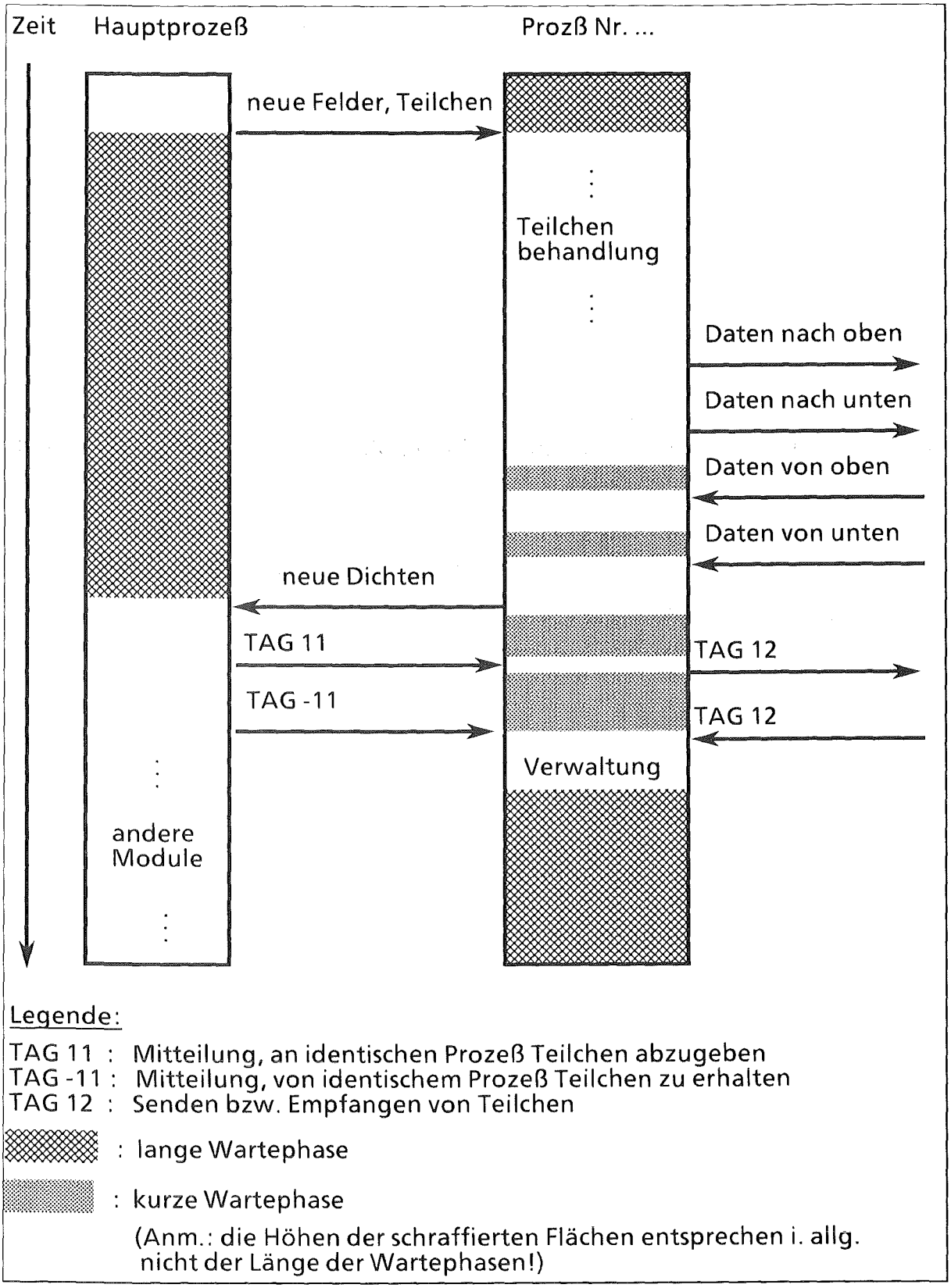


Abb. 3: Zeitlicher Ablauf der Kommunikation.

ausgetauscht, um eine möglichst gleiche Auslastung zu gewährleisten. Die Zahl dieser Teilchen sei NPSSN; die Zahl der Teilchen, die an einen Nachbarn geschickt

werden, sei NSEND. Für jedes Teilgebiet gibt es genau einen Prozeß, der die Daten von den Nachbarprozessen erhält. Wir nehmen an, dies sei auch der Prozeß, der zu Beginn des Zeitschritts die in diesem Teilgebiet befindlichen neu erzeugten Teilchen erhält. Diesen am meisten beschäftigten Prozeß wollen wir untersuchen.

6.3 Kommunikationsaufwand

Hier ist die Abschätzung des Kommunikationsaufwandes um einiges diffiziler, da jeder Prozeß außer der Kommunikation mit dem Hauptprozeß auch Nachrichten an seine Nachbarn sendet, von ihm welche empfängt und - eventuell - mit das gleiche Teilgebiet behandelnden Prozessen kommuniziert. Abb. 3 gibt einen Überblick über die zeitliche Reihenfolge der Kommunikation.

Der Einfachheit halber sei NGITT: = 2*IZMX*IRMX/NPROC die Anzahl der in jedem Prozeß bearbeiteten Gitterpunkte.

1. Kommunikation mit dem Hauptprozeß

a) Empfangene Nachrichten:

Der Prozeß erhält 3*NGITT Gitterdaten und (NPPSN*7)*2 Teilchendaten (das Gebiet wird ist in NPROC/2 Streifen aufgeteilt). Die Länge dieser Nachricht beträgt

$$L^{II}_{E} := 3*NGITT + 14*NPPSN .$$

Alle NIDENT Zeitschritte erhält der Prozeß entweder die Aufforderung, an einen identischen Prozeß Teilchen abzugeben oder welche zu empfangen. Diese Nachricht besteht im ersten Fall aus zwei Variablen, nämlich der Identifikation des Empfängers und der Anzahl der Teilchen, die Länge ist also (bei einer Genauigkeit INTEGER*4).

$$L^{II}_{IS} := 2*\tfrac{1}{2} = 1.$$

Soll der Prozeß Teilchen von einem identischen Prozeß erhalten, so erhält er eine Nachricht der Länge 0:

$$L^{II}_{IE} := 0.$$

b) Gesendete Nachrichten:

Gesendet werden NGITT*3 Dichten sowie NDRZE/(NPROC/2) Ladungen, also

$$L^{II}_{S} := 3*NGITT + 2*NDRZE/NPROC.$$

2. Kommunikation mit den Nachbarn

a) Empfangene Nachrichten:

Von zwei oberen und von zwei unteren Nachbarn werden jeweils 7*NSEND Teilchen erhalten sowie die drei Dichten in den Überlappungsgebieten (jeweils drei Zeilen):

$$L^{II}_{NE} := 7*NSEND + 3*3*IZMX = 7*NSEND + 9*IZMX.$$

b) Gesendete Nachrichten:

Die gleiche Anzahl von Daten wird auch nach oben und unten gesendet, jedoch nur an jeweils einen Nachbarn:

$$L^{II}_{NS} := L^{II}_{NE} = 7*NSEND + 9*IZMX.$$

Alle NIDENT Zeitschritte werden entweder Daten an einen identischen Prozeß gesendet oder empfangen, und zwar 7*NPSSN Teilchendaten:

$$L^{II}_{NI} := 7*NPSSN.$$

Diese Zeit wie auch die zum Einsortieren (bzw. dem Löschen) der Teilchen erforderliche Rechenzeit kann jedoch unberücksichtigt bleiben, da diese Vorgänge zur gleichen Zeit ablaufen können wie die Durchführung anderer Teile des PIC Codes (siehe auch Abb. 3).

6.4 Abschätzung der Kommunikationszeit:

Im Gegensatz zur ersten Strategie findet zwischen dem Empfang der neuen Felder und Teilchen und dem Senden der neuen Dichten eine Synchronisation zwischen den Prozessen statt. Prozeß IP ($1 \leq IP \leq NPROC$) kann seine Berechnungen nach der Zeit

$$IP*T_{START} + L^{II}_{E}*T_{BUS}$$

beginnen (dies ergibt sich wie bei der ersten Strategie). Nach weiteren T^{II}_{PROC} (der in einem Prozeß benötigten Rechenzeit) kann er die Daten, für die seine oberen bzw. unteren Nachbarn zuständig sind, an diese schicken. Die zuletzt verschickten Daten stehen nach weiteren $2*T_{START} + L^{II}_{NS}*T_{BUS}$ zur Verfügung. Nach dem Addieren der Dichten in den Überlappungsgebieten kann mit dem Senden an den Hauptprozeß begonnen werden; die letzten Daten (IP = NPROC) treffen nach insgesamt

$$(NPROC + 3)*T_{START} + (L^{II}_{E} + L^{II}_{NS} + L^{II}_{S})*T_{BUS} + T^{II}_{PROC}$$

beim Hauptprozeß ein. Unter der Voraussetzung, daß innerhalb dieses Zeitschrittes keine Teilchen unter den identischen Prozessen ausgetauscht wurden, muß der Hauptprozeß für die Teilchenbehandlung eine Wartezeit von

$$(NPROC + 3)*T_{START} + (L^{II}_{E} + L^{II}_{NS} + L^{II}_{S})*T_{BUS} + T^{II}_{PROC} + T^{II}_{HAUPT}$$

aufbringen (T^{II}_{HAUPT} sei die für die Parallelisierung im Hauptprozeß notwendige Verwaltungsarbeit).

Alle NIDENT Zeitschritte wird an die Hälfte der Prozesse die Aufforderung versendet, Teilchen an einen identischen Prozeß abzgeben, an die andere Hälfte die Aufforderung, welche zu empfangen. Die Länge dieser Nachrichten sind L^{II}_{IS} bzw. L^{II}_{IE} (= 0). Dazu benötigt der Hauptprozeß die Zeit $NPROC*T_{START}$. Die Zeit für den Transport dieser Daten wie die für den Empfang und den Austausch der Teilchen braucht nicht berücksichtigt zu werden, da der Hauptprozeß zur gleichen Zeit arbeiten kann. Die alle NIDENT Zeitschritte stattfindende Kommunikation versehen wir mit der Gewichtung 1/NIDENT und erhalten somit für die Teilchenbehandlung pro Zeitschritt:

$$T_2 = (NPROC + 3)*T_{START} + (L^{II}_{E} + L^{II}_{NS} + L^{II}_{S})*T_{BUS} + T^{II}_{PROC} + T^{II}_{HAUPT} + NPROC*T_{START}/NIDENT.$$

6.5 Abschätzung von T^{II}_{PROC} und T^{II}_{HAUPT}:

a) T^{II}_{PROC}

Das Markieren der Teilchen, die das Teilgebiet verlassen haben, ist vernachlässigbar (pro abzugebendem Teilchen eine arithmetissche Operation). Das Einsortieren der von anderen Prozessen (Nachbarn und/oder identischen Prozessen) erhaltenen Teilchen und das Löschen der abgegebenen Teilchen kann durchgeführt werden während der Hauptprozeß aktiv ist. Jeder Prozeß muß jedoch die von seinen (zwei oberen und zwei unteren) Nachbarn erhaltenen Dichten in den Überlappungsgebieten zu seinen eigenen hinzuaddieren, bevor er sie abschickt. Dies sind je Dichte oben zwei und unten vier Zeilen, also insgesamt 3*6*IZMX Operationen.

Es ist also $T^{II}_{PROC} \approx T^{I}_{PROC} + 18*IZMX*T_{OP}$.

b) T^{II}_{HAUPT}

Es gilt, die benötigte Zeit für den Verwaltungsaufwand pro Zeitschritt (Bestimmung des Teilgebiets, in dem die neu erzeugten Teilchen sich befinden,

Überprüfung der Auslastung der identischen Prozesse zur Bestimmung des Prozesses, an den die Teilchen gesendet werden sollen, Addieren der erhaltenen Daten) abzuschätzen. Außerdem muß - alle NIDENT Zeitschritte - die Überprüfung der Auslastung der identischen Prozesse berücksichtigt werden (das viel zeitaufwendigere Erzeugen neuer Prozesse wird wegen der Seltenheit vernachlässigt).

In jedem Zeitschritt:

Verteilen der neu erzeugten Teilchen auf den zuständigen Prozess:

Finden des richtigen Teilgebiets für jedes Teilchen (IRMXU(K) bezeichne die unterste von Prozeß K bearbeitete Zeile):

```
      NPPSN = 0
      DO 2 I=1,NPEN
      DO 1 K=1,LINES
         IF ( PEN(I).GE.IRMXU(K) .AND. PEN(I).LT.IRMXU(K+1) ) THEN
           NPPSN = NPPSN + 1
           INDEX(NPPSN) = I
         ENDIF
1     CONTINUE
2     CONTINUE
```

Finden des für dieses Teilgebiet zuständigen und am wenigsten ausgelasteten Prozesses IPROC (IFUNC(I) gibt das von Prozeß I bearbeitete Teilgebiet an und NPE(I) ist die Anzahl der von Prozeß I bearbeiteten Teilchen):

```
      DO 2 K=1,LINES
         NPEMIN = -100000
         DO 1 I=1,NPROC
           IF ( IFUNC(I).EQ.K .AND. NPE(I).LT.NPEMIN ) THEN
              NPEMIN = NPE(I)
              IPROC = I
           ENDIF
1        CONTINUE
2     CONTINUE
```

Um also die Teilchen auf die Prozesse zu verteilen sind etwa

NPEN*(3*LINES + 1) + LINES*(3*NPROC + 1)

Operationen und

2*NPEN*LINES + 2*NPROC*LINES

Abfragen nötig.

Um die erhaltenen Daten zu addieren, müssen für jeden identischen Prozeß in jedem Gitterpunkt drei Dichten und in jeder Elektrodenzelle eine Ladung addiert werden, also

NDRZE + LINES*3*NGITT = NDRZE + 3*IZMX*IRMX

Operationen.

Insgesamt werden in jedem Zeitschritt also etwa

LINES*(3*NPEN + 3*NPROC + 1) + 3*IZMX*IRMX + NDRZE + NPEN

Operationen und

2*NPEN*LINES + 2*NPROC*LINES

Abfragen benötigt.

Alle NIDENT Zeitschritte:

Da für jedes Teilgebiet der am meisten ausgelastete Prozeß Teilchen an den am wenigsten ausgelasteten Prozeß abgeben soll, muß ähnlich wie oben zweimal ein bestimmter Prozeß gefunden werden, das sind

LINES*(6*NPROC + 2)

Operationen und

4*NPROC*LINES

Abfragen.

Pro Teilgebiet muß eine weitere Operation durchgeführt werden, um die Anzahl der abzugebenden Teilchen zu finden. Gewichten wir diese alle NIDENT durchgeführten Berechnungen mit 1/NIDENT, so erhalten wir als Verwaltungsaufwand

LINES*(3*NPEN + 3*NPROC + 1) + 3*IZMX*IRMX + NDRZE + NPEN +
LINES*(6*NPROC + 3) / NIDENT Operationen und
2*NPEN*LINES + 2*NPROC*LINES + 4*NPROC*LINES / NIDENT Abfragen

und damit als benötigte Zeit für den Verwaltungsaufwand

$$T^{II}_{HAUPT} = (LINES*(3*NPEN + 3*NPROC + 1) + 3*IZMX*IRMX + NDRZE + NPEN + LINES*(6*NPROC + 3)/NIDENT)*T_{OP} + (2*NPEN*LINES + 2*NPROC*LINES + 4*NPROC*LINES/NIDENT)*T_{IF}$$

7. Vergleich der in den Strategien benötigten Zeit

7.1 Vergleich der Zeiten

Wir fassen die erhaltenen Werte der Übersichtlichkeit halber zusammen:

$$T_1 = (NPROC + 1)*T_{START} + (L^I_E + L^I_S)*T_{BUS} + T^I_{PROC} + T^I_{HAUPT}.$$

$$T^I_{HAUPT} = ((NPROC-1)*(3*IZMX*IRMX + NDRZE) + NPROC)*T_{OP} + NPROC*T_{IF}$$

$$T_2 = (NPROC + 3)*T_{START} + (L^{II}_E + L^{II}_{NS} + L^{II}_S)*T_{BUS} + T^{II}_{PROC} + T^{II}_{HAUPT} + NPROC*T_{START}/NIDENT.$$

$$T^{II}_{HAUPT} = (LINES*(3*NPEN + 3*NPROC + 1) + 3*IZMX*IRMX + NDRZE + NPEN + LINES*(6*NPROC + 3) / NIDENT)*T_{OP} + (2*NPEN*LINES + 2*NPROC*LINES + 4*NPROC*LINES/NIDENT)*T_{IF}$$

$$T^{II}_{PROC} \approx T^I_{PROC} + 18*IZMX*T_{OP}.$$

mit

$$L^I_E = 3*IZMX*IRMX + 7*NPPSN$$
$$L^I_S = 3*IZMX*IRMX + NDRZE$$
$$L^{II}_E = 3*NGITT + 14*NPPSN$$
$$L^{II}_{IS} = 1$$
$$L^{II}_S = 3*NGITT + 2*NDRZE/NPROC$$
$$L^{II}_{NS} = 7*NSEND + 9*IZMX .$$

Um zwischen den Zeiten für die zwei Strategien vergleichen zu können, bilden wir die Differenzen T_2-T_1 und $T^{II}_{HAUPT} - T^I_{HAUPT}$ (wobei wir NPROC = 2*LINES berücksichtigen):

$$T_2 - T_1 = 2*T_{START} + (L^{II}_E + L^{II}_S + L^{II}_{NS} - L^I_E - L^I_S)*T_{BUS} + T^{II}_{PROC} - T^I_{PROC} + T^{II}_{HAUPT} - T^I_{HAUPT} + 2*LINES*T_{START}/NIDENT$$

$$T^{II}_{HAUPT} - T^I_{HAUPT} = ((LINES*(3*NPEN + 6*LINES - 1) + NPEN + LINES*(12*LINES + 3) / NIDENT) - (2*LINES-2)*(3*IZMX*IRMX + NDRZE))*T_{OP} + (2*NPEN*LINES + 4*LINES*LINES + 8*LINES*LINES / NIDENT - 2*LINES)*T_{IF}$$

Setzen wir nun die Längen ein in T_2-T_1, so erhalten wir:

$$T_2 - T_1 = 2*T_{START} + (6*IZMX*IRMX/LINES + 7*NPPSN + NDRZE/LINES + 7*NSEND + 9*IZMX - 6*IZMX*IRMX - NDRZE)*T_{BUS} + T^{II}_{PROC} - T^{I}_{PROC} + T^{II}_{HAUPT} - T^{I}_{HAUPT} + 2*LINES*T_{START}/NIDENT$$

7.2 Vergleich (mit Zahlenwerten)

Zunächst setzen wir NIDENT: = LINES (d.h., alle LINES Schritte wird die Auslastung der Prozesse überprüft und evtl. ein Austausch von Teilchen zwischen identischen Prozessen durchgeführt) und nehmen an, daß $T_{OP} = 6*T_{IF}$. Es ergibt sich nun (mit $T^{II}_{PROC} \approx T^{I}_{PROC} + 18*IZMX*T_{OP}$):

$$T_2 - T_1 = 4*T_{START} + T^{II}_{HAUPT} - T^{I}_{HAUPT} + 108*IZMX*T_{IF} + (6*IZMX*IRMX/LINES + 7*NPPSN + NDRZE/LINES + 7*NSEND + 9*IZMX - 6*IZMX*IRMX - NDRZE)*T_{BUS}$$

mit

$$T^{II}_{HAUPT} - T^{I}_{HAUPT} = (6*NPEN + 18 - 12*(LINES-1)*(3*IZMX*IRMX + NDRZE) + 20*NPEN*LINES + 40*LINES*LINES + 72*LINES)*T_{IF}$$

Setzen wir die in der Praxis erprobten Werte $NPEN \approx 5*NDRZE$, $NDRZE \approx 4*IRMX$ ein, verwenden - wie in den von uns gerechneten Beispielen meist üblich - Gitter mit $IRMX \approx 2*IZMX$ und beachten, daß NPPSN = NPEN/LINES, so erhalten wir folgende Abschätzung:

$$T_2 - T_1 = 4*T_{START} + T^{II}_{HAUPT} - T^{I}_{HAUPT} + 108*IZMX*T_{IF} + (12*IZMX*IZMX/LINES + 288*IZMX/LINES + 7*NSEND - 12*IZMX*IZMX + IZMX)*T_{BUS}$$

mit

$$T^{II}_{HAUPT} - T^{I}_{HAUPT} = (336*IZMX + 18 - 72*LINES*IZMX*IZMX + 72*IZMX*IZMX + 704*IZMX*LINES + 40*LINES*LINES + 72*LINES)*T_{IF},$$

also

$$T_2 - T_1 = 4*T_{START} + (12*IZMX*IZMX/LINES + 288*IZMX/LINES + 7*NSEND - 12*IZMX*IZMX + IZMX)*T_{BUS} + (444*IZMX - 72*LINES*IZMX*IZMX + 18 + 72*IZMX*IZMX + 704*IZMX*LINES + 40*LINES*LINES + 72*LINES)*T_{IF}$$

Da die zur Verwaltung notwendigen Berechnungen nur skalar durchgeführt werden können, muß von einer Spitzenleistung von etwa 1 MFLOP/s ausge-

gangen werden. Als Ausführungszeit für eine Gleitpunktoperation bei SUPRENUM ergibt sich also:

$T_{OP} = 1\ \mu sec$.

Für die Kommunikationszeiten verwenden wir folgende Werte [8]:

$T_{BUS} = 0.751\ \mu sec$.

$T_{START} = 601.25\ \mu sec$.

Wir setzen diese Werte ein (und verwenden dabei den Erfahrungswert von NSEND = 40):

$$T_2 - T_1 = 2618.28 + IZMX*74.751 + 2.988*IZMX*IZMX + 9.012*IZMX*IZMX/LINES + 216.288*IZMX/LINES - 12*LINES*IZMX*IZMX + 117.3*IZMX*LINES + 6.67*LINES*LINES + 12*LINES)\ \mu sec$$

Tabelle 2 zeigt die Zeitdifferenzen $\Delta t = T_2 - T_1$ für verschiedene Gittergrössen und Prozessorzahlen.

Tabelle 2:

LINES\IZMX	5	10	15	20	30	50
2	4344	5193	5218	4416	338	-17721
3	4458	4701	3444	686	-9330	-47368
4	4694	4477	2123	-2370	-17769	-74223
5	4987	4369	990	-5149	-25708	-99954
6	5316	4325	-40	-7782	-33391	-125115
7	5670	4324	-1008	-10326	--40921	-149945
8	6045	4354	-1930	-12809	-48350	-174563

Zeitdifferenzen (in µsec) zwischen den beiden Parallelisierungsstrategien für verschiedene Werte von IZMX und LINES. $\Delta t > 0$: Strategie 1 vorteilhafter, $\Delta t < 0$: Strategie 2 vorteilhafter.

Aus Tabelle 2 ergibt sich, daß bei kleinen Gittern Strategie I vorteilhafter ist. Da aber i. allg. mit größeren Gittern gerechnet wird und die Aufteilung genügend fein sein wird, wird bei Strategie II der Vorteil der kürzeren Nachrichten den Nachteil der größeren Anzahl der einzelnen Kommunikationen aufwiegen, so daß in den meisten Fällen Strategie II vorzuziehen ist.

8. Abschätzung des Speed-ups

Bei der folgenden Untersuchung beschränken wir uns auf Strategie II. Um eine Abschätzung für die in einem Prozeß benötigte Rechenzeit zu erhalten, gehen wir wie folgt vor [13]:

Ausgehend von den auf einer SIEMENS VP50 benötigten Rechenzeiten in den einzelnen Modulen extrapolieren wir über das Verhältnis der Taktfrequenzen auf die voraussichtlich auf einem SUPRENUM-Knoten benötigte Zeit. Dieser Vorgehensweise liegt die Überlegung zugrunde, daß bei genügend hohem Vektorisierungsgrad pro Takt ein neues Ergebnis vorliegt. Auf diese Art und Weise erhalten wir eine Abschätzung für die je Zeitschritt anfallende CPU-Zeit, falls ohne Parallelisierung auf einem einzigen SUPRENUM-Prozessor gerechnet wird.

Die in den Modulen benötigten Rechenzeiten sind in etwa proportional zur Anzahl der behandelten Teilchen (wenn die zur Initialisierung der Vektorpipelines erforderliche Start-up Zeit vernachlässigt wird), bezüglich der Gittergröße läßt sich jedoch keine ähnliche Abhängigkeit treffen. Daher betrachten wir zwei konkrete Fälle, nämlich Gitter mit 11 x 21 Gitterpunkten bzw. 21 x 45 Gitterpunkten. Tabelle 3 zeigt die für die einzelnen Module benötigten Rechenzeiten.

Tabelle 3:

	CPU-Zeiten VP50		erwartete CPU-Zeiten SUPRENUM	
Programm-Modul	11 x 21 - Gitter	21 x 45 - Gitter	11 x 21 - Gitter	21 x 45 - Gitter
Interpolation	58	144	414	1029
Teilchenbewegung	390	1031	2786	7364
Lokalisierung	191	466	1364	3329
Ladungs- und Stromdichten	354	851	2528	6079
Rest Teilchenbehandlung	3	8	21	57
Summe Teilchenbehandlung	996	2500	7114	17857

Rechenzeiten der einzelnen Module pro Zeitschritt auf dem Vektorrechner SIEMENS VP50 *in Millisekunden (gemittelt über100 Zeitschritte, beim 11 x 21 - Gitter ca. 10 000 Teilchen, beim 21 x 45 -Gitter ca. 20 000 Teilchen) und für SUPRENUM hochgerechnete Werte.*

Zur Zeitmessung wurde beim stationären Zustand über 100 Zeitschritte gemittelt. Im Falle des 11 x 21 Gitters befanden sich etwa 18 000 Teilchen in der

Diode, im Falle des 21 x 45 Gitters waren es ca. 43 000 Teilchen. Die Zeiten wurden durch 7 ns (der Taktzeit der VP50) dividiert und mit der Taktzeit von SUPRENUM, nämlich 50 ns, multipliziert.

Durch diese Vorgehensweise haben wir nun eine - wenn auch sehr grobe - Näherung für die in einem Prozeß benötigte Rechenzeit $T_1{}^{II}{}_{PROC}$ für den Fall, daß ein einzelner Prozeß alle Teilchen behandelt, da in diesem Fall keinerlei Kommunikation notwendig ist.

Stehen nun NPROC (= 2,4,...,16) Prozessoren zur Verfügung, so ergibt sich die für einen Zeitschritt erforderliche Zeit aus

$$T_{NPROC,TOTAL} = T_2$$
$$= (NPROC + 3)*T_{START} + (L^{II}{}_E + L^{II}{}_{NS} + L^{II}{}_S)*T_{BUS} +$$
$$T^{II}{}_{PROC} + T^{II}{}_{HAUPT} + NPROC*T_{START}/NIDENT$$

mit

$$T^{II}{}_{HAUPT} = (LINES*(3*NPEN + 3*NPROC + 1) + 3*IZMX*IRMX + NDRZE + NPEN + LINES*(6*NPROC + 3)/NIDENT)*T_{OP} + (2*NPEN*LINES + 2*NPROC*LINES + 4*NPROC*LINES/NIDENT)*T_{IF}$$
$$L^{II}{}_E = 3*NGITT + 14*NPPSN$$
$$L^{II}{}_S = 3*NGITT + 2*NDRZE/NPROC$$
$$L^{II}{}_{NS} = 7*NSEND + 9*IZMX$$

und

$$T^{II}{}_{PROC} \approx T_1{}^{II}{}_{PROC} / NPROC + 18*IZMX*T_{OP}$$

(siehe Abschnitte 6.3 und 6.4).

Wir setzen diese und die in 7.3 verwendeten Werte

$NIDENT = LINES$
$NPEN = 5*NDRZE$
$NDRZE = 4*IRMX$
$NPPSN = NPEN/LINES = 20*IRMX/LINES$
$NSEND = 40$
$T_{OP} = 6*T_{IF} = 1\ \mu sec$
$T_{START} = 601.25\ \mu sec$
$T_{BUS} = 0.751\ \mu sec$

ein und erhalten:

$$T_{NPROC,TOTAL} = T_1{}^{II}{}_{PROC} / NPROC +$$
$$(NPROC*606.42 + 3216.53 + 9.01*IZMX*IRMX/NPROC +$$
$$426.57*IRMX/NPROC + 24.76*IZMX + 33.33*IRMX*NPROC$$
$$2.5*NPROC*NPROC + 3*IZMX*IRMX + 24*IRMX)\ \mu sec.$$

In Tabelle 4 sind für die beiden Gitter die geschätzten Rechenzeiten für verschiedene Anzahlen von verwendeten Prozessoren aufgelistet.

Tabelle 4

	erwartete CPU-Zeiten		erwarteter Speed-up auf SUPRENUM	
Zahl der Prozessoren	11 x 21 - Gitter	21 x 45 - Gitter	11 x 21 - Gitter	21 x 45 - Gitter
1 (keine Parallelisierung)	7114	17857	-------	-------
2	3570	8954	1.99	1.99
4	1791	4487	3.97	3.98
6	1200	3001	5.93	5.95
8	906	2260	7.85	7.90
10	731	1817	9.74	9.83
12	614	1524	11.58	11.72
14	532	1315	13.36	13.58
16	472	1160	15.09	15.40

Geschätzte Zeit (iN Millisekunden) für die Teilchenbehandlung innerhalb eines Zeitschritts des PIC-Codes auf SUPRENUM in Abhängigkeit der verwendeten Anzahl von Prozessoren; Speed-up.

Während üblicherweise bei numerischen Verfahren wie Mehrgittermethoden sich eine Parallelisierung erst bei feinen Gittern bezahlt macht, besitzt die Gittergröße nur geringen Einfluß auf die Effizienz der Parallelisierung der Teilchenbehandlung. Die zu erwartende Beschleunigung bei einer Parallelisierung ist nahe am optimalen Wert. Trotz der groben Abschätzung mittels dem Verhältnis der Taktzeiten läßt sich aus Tabelle 4 eine hohe Effizienz bei der Parallelisierung der Teilchenbehandlung ablesen. Von daher ist es wichtig, eine gute Verknüpfung zwischen einer Parallelisierung der Feldberechnungen und der Teilchenbehandlung zu erreichen, um auch den gesamten PIC Code effizient parallelisieren zu können.

Schlußfolgerungen und weiteres Vorgehen

In der Praxis muß mit genügend feinen Gittern gerechnet werden, um eine ausreichende Genauigkeit und genügend Detailinformationen zu erhalten. Da außerdem i.a. eine Parallelisierung nur bei einer genügend großen Anzahl von zur Verfügung stehenden Prozessoren sinnvoll erscheint, ist nach dem hier vorgestellten Modell eine Aufteilung bezüglich des Gitters vorzuziehen.

Im Hinblick auf eine Gesamtstrategie zur Parallelisierung des PIC Codes wird im Kernforschungszentrum Karlsruhe auch an einer Parallelisierung der Feldberechnungen gearbeitet. Bei einer gleichen Aufteilung des Gitters auf die Prozesse wie bei der hier vorgestellten Teilchenbehandlung wird die Kommunikation beim Übergang von der Teilchenbehandlung zu den Feldberechnungen (und umgekehrt) wesentlich geringer sein als wenn die Teilchen ohne Rücksicht auf ihre Lage im Gitter auf die Prozesse verteilt werden. Wird die erste Strategie verwendet, so müssen alle Dichten an den Hauptprozeß gesendet werden, der sie dann vor der Feldberechnung an die verschiedene Prozesse weitergeben muß, um danach die neuen Feldstärken zu erhalten und zur erneuten Teilchenbehandlung wieder zu verschicken. Diese Kommunikation kann bei einer geschickten Kopplung der Parallelisierungsstrategien zur Teilchenbehandlung und zur Feldberechnung größtenteils vermieden werden. Dann ist auch mit einer wesentlichen höheren Effizienz zu rechnen.

Danksagung

Mein Dank geht an Thomas Westermann für wertvolle Anregungen bei der Zusammenstellung dieses Artikels. Dank auch an R. Vogelsang von der SUPRENUM GmbH für die Hilfestellung bei der Abschätzung der zu erwartenden Effizienz einer Parallelisierung.

Literatur

[1] M. Alef: Unveröffentlichter Bericht. KfK, Januar 1988

[2] M. Alef: Persönliches Gespräch

[3] M. Alef, D. Seldner, T. Westermann: Numerische Algorithmen für elektrodynamische Modelle und ihre Implementierung auf Supercomputern. Simulationstechnik (J. Halin, Hrsg.). 4. Symposium Simulationstechnik, Zürich, Sept. 1987, Informatik-Fachberichte Nr. 150, Springer-Verlag, 1987, pp. 298-305

[4] M. Alef, D. Grether, D. Seldner, T. Westermann, Diodensimulation mit der „Particle-In-Cell" Methode und mögliche Implementierung auf SUPRENUM, KfK-Nachrichten 20 (3) 1988, Kernforschungszentrum Karlsruhe GmbH

[5] W. K. Giloi: SUPRENUM: A Trendsetter in Modern Supercomputer Development. 2nd Int. SUPRENUM Colloquium, Bonn, Sept./Okt. 1987. Parallel Computing 7, 1988, pp. 283-296

[6] R. Hempel, A. Schüller, Vereinheitlichung und Portabilität paralleler Anwendungssoftware durch Verwendung einer Kommunikationsbibliothek. Arbeitspapiere der GMD 234, St. Augustin, November 1986

[7] R. W. Hockney, J. W. Eastwood: Computer Simulation Using Particles. McGraw-Hill, New York, 1981

[8] O. Kolp, H. Mierendorff: Performance estimations for SUPRENUM systems. 2nd Int. SUPRENUM Colloquium, Bonn, Sept./Okt. 1987. Parallel Computing 7, 1988, pp. 357-366

[9] D. Seldner, M. Alef, T. Westermann, E. Halter: Parallel Particle Simulation in High Voltage Diodes (Algorithms and Concepts for Implementation on SUPRENUM). 2nd Int. SUPRENUM Colloquium, Bonn, Sept./Okt. 1987. Parallel Computing 7, 1988, pp. 445-449

[10] D. Seldner, T. Westermann: Algorithms for Interpolation and Localization in Irregular 2D Meshes. J. Comp. Phys. 79, pp. 1-11, 1988

[11] U. Trottenberg: On the SUPRENUM Conception (Version 2). SUPRENUM Report 1, SUPRENUM GmbH, Bonn, Januar 1987

[12] U. Trottenberg: SUPRENUM - a MIMD system for multilevel scientific supercomputing. SUPRENUM Report 2, SUPRENUM GmbH, Bonn, Februar 1987

[13] R. Vogelsang: Persönliche Mitteilungen, 29.9.1988, 2.11.1988

[14] T. Westermann, D. Seldner: Unveröffentlichter Bericht. KfK, März 1987

[15] T. Westermann: Teilchenfortbewegung in elektro-magnetischen Feldern. KfK 4325, Januar 1988

[16] T. Westermann: A Particle-in-Cell Method as a Tool for Diode Simulations. Nuclear Instruments and Methods in Physics Research A263 (1988), S. 271-279