Kapitel 3

Parallelisierung

3.1 Supercomputer-Architekturen

Hochdimensionale quantendynamische Studien stellen enorme Forderungen an die verwendete Hardware. Aufgrund ihres sehr guten Skalierungsverhaltens, wie in Kapitel 2 aufgezeigt, benötigen solche hochdimensionalen Simulationen über die erforderliche Rechenleistung hinaus sehr große Mengen an Speicher. Diese Rechenressourcen sind nur im Rahmen paralleler Hardware-Architekturen verfügbar. In Abb. 3.1 ist die Struktur verschiedener Supercomputer-Systeme illustriert, die im folgenden mit den zugehörigen Programmier-Schnittstellen vorgestellt werden.

In *Symmetric Multi-Processing (SMP)* Systemen greift eine geringe Anzahl von Prozessoren ($P \leq 64$) über ein Speicher-Netzwerk auf einen globalen Speicher zu *(shared memory access)*. Da der gesamte Speicher allen Prozessoren zugänglich ist, muß in der Programmierung von SMP-Systemen nur der koordinierte Zugriff der einzelnen Prozessoren auf den globalen Speicher sichergestellt werden. Dies kann durch den Einsatz von *multitasking libraries* geschehen, deren Routinen numerische Standardaufgaben unter Nutzung mehrerer Prozessoren erfüllen. Dieses Vorgehen ist leicht zu realisieren, schöpft jedoch die verfügbare Rechenleistung nicht effizient genug aus. Die professionelle Programmierung von SMP-Systemen erfolgt daher über die Schnittstelle OpenMP [62, 63]. Dabei wird über Compiler-Direktiven im Quelltext die Datenverarbeitung der einzelnen Prozessoren zugewiesen und synchronisiert. Typische SMP-Systeme erreichen eine Rechenleistung von $10^1 - 10^2$ GFLOPS *(Floating Point Operations Per Second)*.

Der effiziente Zugriff auf einen globalen Speicher ist für eine größere Anzahl von Prozessoren ($128 \leq P \leq 1024$) technisch nicht mehr realisierbar. Dieser Supercomputer-Bereich wird durch *Distributed Memory Parallel*

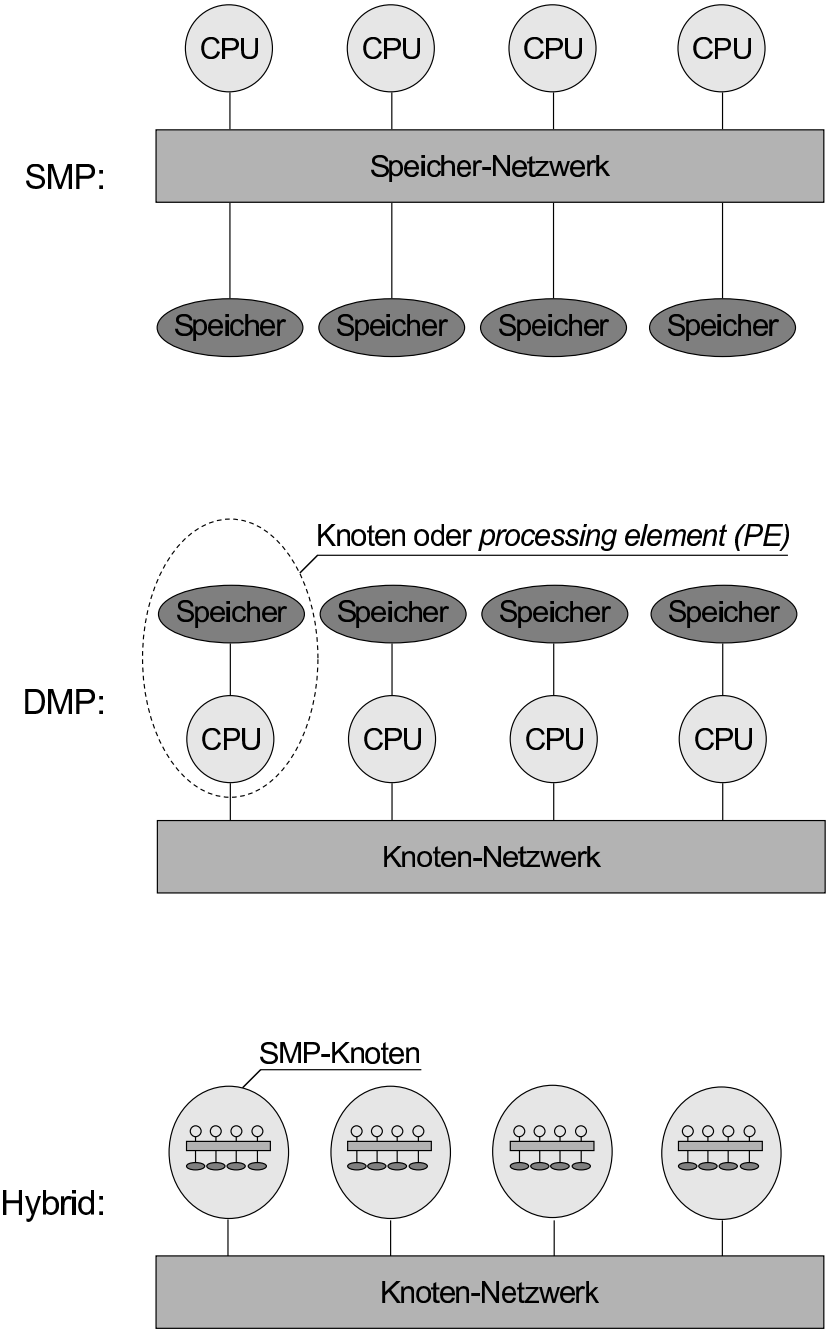

Abbildung 3.1: Strukturdiagramme paralleler Hardware-Architekturen: *Symmetric Multi-Processing (SMP)* mit *shared memory access*, *Distributed Memory Parallel (DMP)* mit *distributed memory access* und Hybride aus DMP verschalteten SMP-Knoten [64].

(DMP) Systeme abgedeckt. In dieser Architektur sind separate Knoten, die *processing elements (PEs)*, bestehend aus einem Prozessor mit lokalem Speicher, durch ein Knoten-Netzwerk verbunden *(distributed memory access)*. Jedes PE bearbeitet nur die Daten in seinem lokalen Speicher. Die Verteilung sowohl der Daten als auch der Datenverarbeitung muß dabei explizit festgelegt werden. Der Datenaustausch zwischen den PEs wird durch Kommunikation via *message passing* realisiert. Dafür stellen die konkreten Schnittstellen, wie das *Message Passing Interface (MPI)* oder die *SHMEM library*, diverse Kommunikationsroutinen bereit [65–67]. MPI ist standardisiert und somit unabhängig vom Hersteller einsetzbar. Die Kommunikation mit MPI erfolgt zweiseitig, da sowohl das sendende als auch das empfangende PE an der Kommunikation teilnehmen. SHMEM-Kommunikation dagegen ist einseitig, da ein PE im/vom Speicher des anderen PE ohne dessen Beteiligung schreibt/liest. Deshalb bietet SHMEM eine höhere Effizienz als MPI, ist jedoch an die Hersteller Cray und SGI gebunden. Mit DMP-Systemen sind Rechenleistungen von $10^2 - 10^3$ GFLOPS erreichbar.

Um die Rechenleistung noch weiter zu erhöhen, werden in Hybrid-Architekturen viele SMP-Knoten, die jeweils aus mehreren Prozessoren, globalem Speicher und entsprechendem Speichernetzwerk bestehen, durch ein Knoten-Netzwerk verbunden. Mit dieser zweistufigen DMP-SMP-Hierarchie kann eine sehr große Anzahl von Prozessoren ($128 \leq P \leq 8192$) zusammengefaßt werden. In der konsequenten Programmierung von Hybrid-Systemen wird die Datenverarbeitung der SMP-Knoten mit OpenMP realisiert. Für die übergeordnete Kommunikation zwischen den Knoten wird MPI verwendet. Hybrid-Systeme markieren mit $10^2 - 10^4$ GFLOPS das gegenwärtig erreichbare Maximum an Rechenleistung.

Die in Kapitel 2 eingeführte, hochdimensionale Quantendynamik eines diatomaren Moleküls auf einer Oberfläche erfordert einen numerischen Aufwand, der durch SMP-Systeme nicht bewältigt werden kann. Nur DMP- oder Hybrid-Systeme werden diesen Anforderungen bezüglich der Rechenleistung und des Speicherbedarfs gerecht. In den folgenden Abschnitten wird eine Strategie zur Parallelisierung der quantendynamischen Simulationsanwendung diskutiert, die auf *message passing* basiert und somit zuallererst für DMP-Systeme geeignet ist. Da alle gängigen *message passing* Schnittstellen durch Emulation der Kommunikation auch auf SMP-Systemen zur Verfügung stehen, ist eine solche Parallelisierung darüber hinaus allgemein anwendbar. Dimensionsreduzierte Simulationen können auf SMP-Systemen durchgeführt werden. Für hochdimensionale Simulationen können neben DMP- auch Hybrid-Systeme genutzt werden. In der vorliegenden Arbeit wird eine vierdimensionale Parallelisierung des FORTRAN90 Simulationscodes in den Koordinaten $\{X, Z, \vartheta, \varphi\}$ mit SHMEM- und MPI-Kommunikation realisiert.

3.2 Datenzerlegung

In der vorgestellten Quantendynamik werden dynamische Kopplungen vollständig innerhalb des pseudospektralen Algorithmus der HAMILTON-Operation vermittelt. Die Analyse dieses Algorithmus bezüglich seiner inhärenten Parallelität führt auf eine typische *Single Program Multiple Data (SPMD)* Struktur: Alle PEs verarbeiten in gleicher Weise äquivalente Datenobjekte in ihrem lokalen Speicher, während der symmetrische Datenaustausch unter den PEs durch *message passing* realisiert wird. Die beiden grundlegenden Operationen sind die Anwendung der verschiedenen diagonalen Anteile des HAMILTON-Operators in der entsprechenden Darstellung und die FBR-DVR-Transformationen für den Wechsel der Darstellung der Wellenfunktion. Die Vektor-Vektor-Multiplikationen der Operatoranwendungen sind lokale Operationen auf jeder Datenverteilung aufgrund ihres elementaren Charakters. Für die FBR-DVR-Transformationen hingegen ist ein globaler Vektor in der Dimension der zu transformierenden Koordinate und somit Kommunikation auf der Datenverteilung erforderlich.

Die gewünschte Datenverteilung sollte folglich eine perfekten Lastenausgleich für die lokalen Operationen sicherstellen und eine symmetrische und effiziente Kommunikationsstruktur für die FBR-DVR-Transformationen implizieren. Deshalb wird eine flexible Datenzerlegung in allen Dimensionen favorisiert. Die vierdimensionale (4D) Wellenfunktion $\psi(X, Z, \vartheta, \varphi)$ der globalen Größe $N \equiv N^{\mathrm{DVR}} = N_X \times N_Z \times N_\vartheta \times N_\varphi$ wird gleichmäßig auf $N_{\mathrm{PE}} = N_X^{\mathrm{PE}} \times N_Z^{\mathrm{PE}} \times N_\vartheta^{\mathrm{PE}} \times N_\varphi^{\mathrm{PE}}$ PEs verteilt. Dabei müssen die lokale Größe $N_{\mathrm{local}} = N_X^{\mathrm{local}} \times N_Z^{\mathrm{local}} \times N_\vartheta^{\mathrm{local}} \times N_\varphi^{\mathrm{local}} = N/N_{\mathrm{PE}}$ der resultierenden Wellenfunktionen der einzelnen PEs als auch die lokalen Dimensionsgrößen $N^{\mathrm{local}}_{\{X,Z,\vartheta,\varphi\}} = N_{\{X,Z,\vartheta,\varphi\}}/N^{\mathrm{PE}}_{\{X,Z,\vartheta,\varphi\}}$ ganzzahlige Quotienten sein.[1] Im folgenden wird die konkrete Charakteristik der Datenverteilung in den Dimensionen der kartesischen Koordinaten und der Kugelkoordinaten diskutiert.

3.2.1 Kartesische Koordinaten

Da die Datenobjekte in kartesischen Koordinaten die gleiche Struktur in FBR und DVR haben, wird schon mit einer einfachen Block-Verteilung ein optimaler Lastenausgleich für die lokalen Operationen erreicht. Abb. 3.2 zeigt diese Block-Verteilung eines Datenobjektes in den kartesischen Koordinaten $\{X, Z\}$ für das einfache Beispiel eines 2×2 PE Gitters. Alle PEs verarbeiten die gleiche Datenmenge in ihrem lokalen Speicher. Die Indexalgebra

[1] Alle Grundrechenoperationen in den Formeln des Abschnitts 3.2 sind als Integer-Arithmetik zu verstehen.

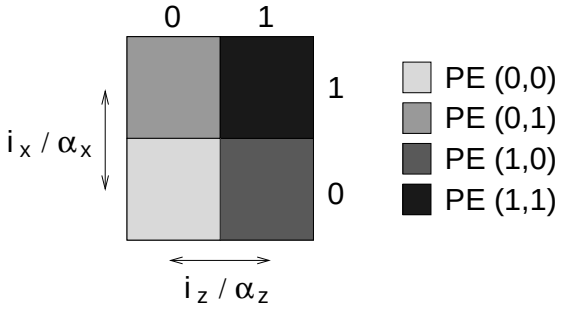

Abbildung 3.2: Datenverteilung für ein Datenobjekt in den kartesischen Koordinaten $\{X, Z\}$ auf einem 2×2 PE Gitter. Daten, die von einem bestimmten PE verarbeitet werden, sind durch farbliche Kennzeichnung unterschieden. Das Farbschema zeigt einen perfekten Lastenausgleich für die lokalen Operationen: Jedes PE verarbeitet die gleiche Datenmenge.

der Zuordnung von globalen und lokalen Datenelementen hat für die Blockverteilung eine einfache Struktur. Ausgehend vom Datenelement mit lokalem Index $i^{\text{local}}_{\{X,Z\}}$ auf dem PE $n^{\text{PE}}_{\{x,z\}}$ ergibt sich der zugehörige, globale Index zu

$$i_{\{X,Z\}} = i^{\text{local}}_{\{X,Z\}} + n^{\text{PE}}_{\{X,Z\}} \cdot N^{\text{local}}_{\{X,Z\}}. \tag{3.1}$$

In der inversen Zuordnung werden für den globalen Index $i_{\{X,Z\}}$ der entsprechende, lokale Index und das PE, auf dem sich das Datenelement befindet, über die Beziehungen

$$\begin{aligned} i^{\text{local}}_{\{X,Z\}} &= \left(i_{\{X,Z\}} - 1\right) \bmod N^{\text{local}}_{\{X,Z\}} + 1 \qquad (3.2)\\ n^{\text{PE}}_{\{x,z\}} &= \frac{i_{\{X,Z\}} - 1}{N^{\text{local}}_{\{X,Z\}}} \end{aligned}$$

bestimmt. Diese Datenverteilung kann leicht auf eine 6D Quantendynamik verallgemeinert werden, indem die resultierenden 3D Datenobjekte in den kartesischen Koordinaten $\{X, Y, Z\}$ auch in der dritten Koordinate analog blockverteilt werden.

3.2.2 Kugelkoordinaten

Für die Winkelkoordinaten ist die Struktur der Datenobjekte abhängig von der Darstellung. Um auch hier einen perfekten Lastenausgleich der lokalen Operationen in FBR und DVR zu erreichen, ist eine kompliziertere Datenverteilung erforderlich. Abb. 3.3 illustriert beispielhaft die Verteilung eines

Datenobjektes in den Winkelkoordinaten $\{\vartheta, \varphi\}$ auf einem 2×2 PE Gitter. Dabei setzt sich das FBR Datenobjekt aus dem zentralen Dreieck und dem oberen Rechteck zusammen. Das Dreieck der Region $j \leq m_{\max}$ resultiert aus der Multiplizitätsrelation $m \leq j$ der Drehimpulszustände j und ihrer Projektion m. Das Rechteck der Region $j > m_{\max}$ hingegen wird von der maximalen Projektion begrenzt und folglich durch die Relation $m \leq m_{\max}$ bestimmt. Im Gegensatz dazu umfaßt das DVR Datenobjekt das gesamte Rechteck $N_\vartheta \times N_\varphi$ einschließlich des rechten und linken Dreiecks. Ist die maximale Projektion gleich dem maximalen Drehimpuls: $m_{\max} = j_{\max}$, verschwindet das obere Rechteck, und die Diskussion der Datenobjekte reduziert sich auf das untere Rechteck aus drei Dreiecken.

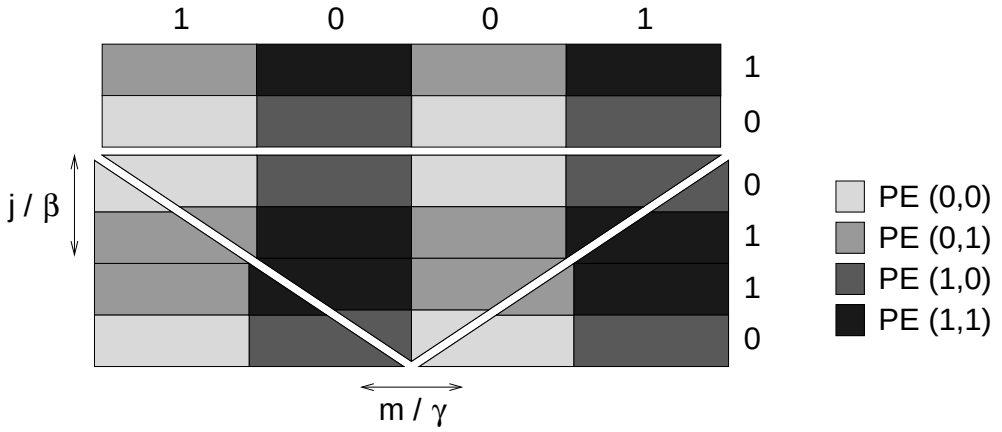

Abbildung 3.3: Datenverteilung für ein Datenobjekt in den Winkelkoordinaten $\{\vartheta, \varphi\}$ auf einem 2×2 PE Gitter. Das FBR Datenobjekt in $\{j, m\}$ besteht nur aus dem zentralen Dreieck und dem oberen Rechteck, das DVR Datenobjekt in $\{\beta, \gamma\}$ hingegen beinhaltet das gesamte Rechteck inklusive des linken und rechten Dreiecks. Daten, die von einem bestimmten PE verarbeitet werden, sind wiederum durch farbliche Kennzeichnung unterschieden. Das Farbschema zeigt einen perfekten Lastenausgleich für die lokalen Operationen: Jedes PE verarbeitet die gleiche Datenmenge sowohl in der FBR als auch in der DVR.

Um die Datenverteilung zu erleichtern, wird durch eine Anpassung der Darstellungen die Symmetrie der Datenobjekte erhöht: In der DVR des azimuthalen Winkels φ wird das Gitter um einen Punkt verkleinert. Mit der somit reduzierten Dimensionsgröße $N_\varphi = 2m_{\max}$ wird in der korrespondierenden FBR die größte Projektion $\pm m_{\max}$ durch nur eine Basisfunktion dargestellt. Dieses Vorgehen beschränkt die Allgemeinheit von konvergierten Si-

mulationen mit hinreichend großen Basissätzen nicht. Zudem wird damit die Wahl der Dimensionsgröße N_φ als Zweierpotenz und somit die optimale Skalierung der FFT im azimuthalen Winkel erreicht.

Für den polaren Winkel ϑ werden die Datenelemente des unteren und oberen Rechtecks unterschiedlich verteilt. Die zwei entsprechenden Beiträge $N_{\vartheta,1}^{\text{local}} = (N_\varphi/2)/N_\vartheta^{\text{PE}}$ und $N_{\vartheta,2}^{\text{local}} = (N_\vartheta - N_\varphi/2)/N_\vartheta^{\text{PE}}$ zur lokalen Dimensionsgröße $N_\vartheta^{\text{local}} = N_{\vartheta,1}^{\text{local}} + N_{\vartheta,2}^{\text{local}}$ müssen ganzzahlige Quotienten sein. Im unteren Rechteck der Region $j \leq N_\varphi/2$ werden die obere und untere Hälfte des globalen Teilvektors paarweise in zunehmender bzw. abnehmender Reihenfolge der Datenelemente auf die lokalen Vektoren verteilt. Der Einfachheit halber werden dabei die resultierenden Datenelementpaare auf fortlaufend ungerade und gerade, lokale Positionen eingeordnet. Im oberen Rechteck der Region $j > N_\varphi/2$ hingegen wird der globale Teilvektor blockverteilt. Die Blöcke werden in den lokalen Vektoren hinter den Daten vom unteren Rechteck abgelegt. Aufgrund der Fallunterscheidungen in dieser Datenverteilung hat die korrespondierende Indexalgebra eine kompliziertere Struktur. Für das Datenelement mit lokalem Index j^{local} auf dem PE n_ϑ^{PE} folgt der globale Index j aus der Relation

$$
\begin{aligned}
&j^{\text{local}} \leq N_{\vartheta,1}^{\text{local}}: && && (3.3)\\
&\quad j^{\text{local}} \text{ ungerade}: && j = && j^{\text{local}}/2 + n_\vartheta^{\text{PE}} \cdot \left(N_{\vartheta,1}^{\text{local}}/2\right) + 1\\
&\quad j^{\text{local}} \text{ gerade}: && j = && N_\varphi/2 - \left(j^{\text{local}}/2 + n_\vartheta^{\text{PE}} \cdot \left(N_{\vartheta,1}^{\text{local}}/2\right)\right) + 1\\
&j^{\text{local}} > N_{\vartheta,1}^{\text{local}}: && j = && N_\varphi/2 + j^{\text{local}} - N_{\vartheta,1}^{\text{local}} + n_\vartheta^{\text{PE}} \cdot N_{\vartheta,2}^{\text{local}}\ .
\end{aligned}
$$

Umgekehrt ergeben sich für den globalen Index j der zugehörige, lokale Index j^{local} und das PE n_ϑ^{PE}, auf dem das Datenelement abgelegt ist, durch die

Gleichungen

$$
\begin{aligned}
&j \le N_\varphi/2: && && (3.4)\\
&\quad j \le N_\varphi/4: && j^{\text{local}} = && 2\cdot(j-1) \bmod \left(N_{\vartheta,1}^{\text{local}}/2\right) + 1\\
&&& n_\vartheta^{\text{PE}} = && \frac{j-1}{N_{\vartheta,1}^{\text{local}}/2}\\
&\quad j > N_\varphi/4: && j^{\text{local}} = && 2\cdot(N_\varphi/2-j) \bmod \left(N_{\vartheta,1}^{\text{local}}/2\right) + 2\\
&&& n_\vartheta^{\text{PE}} = && \frac{N_\varphi/2-j}{N_{\vartheta,1}^{\text{local}}/2}\\
&j > N_\varphi/2: && j^{\text{local}} = && N_{\vartheta,1}^{\text{local}} + (j-N_\varphi/2-1) \bmod N_{\vartheta,2}^{\text{local}} + 1\\
&&& n_\vartheta^{\text{PE}} = && \frac{j-N_\varphi/2-1}{N_{\vartheta,2}^{\text{local}}}\,.
\end{aligned}
$$

Neben den Forderungen an die Größen $N_{\vartheta,1}^{\text{local}}$ und $N_{\vartheta,2}^{\text{local}}$ der lokalen Teilvektoren setzt diese Datenverteilung außerdem eine durch vier teilbare globale Dimensionsgröße N_φ im azimuthalen Winkel voraus. Dies kann aufgrund der vorab durchgeführten Darstellungsanpassung leicht erfüllt werden.

Eine ganz ähnliche Datenverteilung wird für den azimuthalen Winkel φ verwendet. Die obere und untere Hälfte des globalen Vektors werden paarweise in zunehmender Reihenfolge der Datenelemente auf die lokalen Vektoren verteilt. Wiederum werden die resultierenden Datenelementpaare auf fortlaufend ungerade und gerade, lokale Positionen eingeordnet. In der Indexalgebra wird dem lokalen Index m^{local} auf dem PE n_φ^{PE} durch die Beziehung

$$
\begin{aligned}
&m^{\text{local}} \text{ ungerade}: && m = && m^{\text{local}}/2 + n_\varphi^{\text{PE}} \cdot \left(N_\varphi^{\text{local}}/2\right) + 1 && (3.5)\\
&m^{\text{local}} \text{ gerade}: && m = && N_\varphi/2 + m^{\text{local}}/2 + n_\varphi^{\text{PE}} \cdot \left(N_\varphi^{\text{local}}/2\right)
\end{aligned}
$$

der globale Index m zugeordnet. Im Umkehrschluß korrespondiert der globale

Index m durch die Relationen

$$
\begin{aligned}
m \leq N_\varphi/2: \quad m^{\text{local}} &= 2 \cdot (m-1) \bmod \left(N_\varphi^{\text{local}}/2\right) + 1 \\
n_\varphi^{\text{PE}} &= \frac{m-1}{N_\varphi^{\text{local}}/2} \\
m > N_\varphi/2: \quad m^{\text{local}} &= 2 \cdot (m - N_\varphi/2 - 1) \bmod \left(N_\varphi^{\text{local}}/2\right) + 2 \\
n_\varphi^{\text{PE}} &= \frac{m - N_\varphi/2 - 1}{N_\varphi^{\text{local}}/2}
\end{aligned}
\tag{3.6}
$$

mit dem lokalen Index m^{local} auf dem PE n_φ^{PE}. Auch diese Datenverteilung kann leicht auf eine 6D Quantendynamik erweitert werden. Die Struktur der entsprechenden 3D Datenobjekte in Kugelkoordinaten $\{r, \vartheta, \varphi\}$ ist bezüglich der zusätzlichen, radialen Koordinate darstellungsunabhängig. Daher können die Datenobjekte in der radialen Koordinate einfach blockverteilt werden.

Im Gegensatz zu konventionellen Konzepten werden in der vorgestellten Datenverteilung die Datenobjekte in allen Freiheitsgraden zerlegt, um sehr große Systeme ($N > 10^9$) quantendynamischen Studien zugänglich zu machen. Gewöhnliche Strategien zur Parallelisierung quantendynamischer Beschreibungen sind auf eine Datenzerlegung in nur einer Koordinate beschränkt [68–71]. Ein solches Vorgehen ist praktikabel für die Untersuchung von Systemen mittlerer Größe, die nur die Datenverteilung auf eine geringe Anzahl von PEs erfordert. Sollen hingegen Studien eines sehr großen Systems durch die Nutzung von vielen PEs bewältigt werden, sind die Grenzen einer Datenverteilung in nur einem Freiheitsgrad erreicht: Die lokale Größe der verteilten Dimension wird extrem klein. Folglich sollte diese Dimension gemäß der Speicherstruktur von FORTRAN Feldern[2] dem letzten Index des Datenfeldes zugeordnet werden, um einen optimalen, lokalen Datenzugriff zu gewährleisten. Für den globalen Datenzugriff in der verteilten Dimension wird dadurch jedoch ein maximaler Umordnungsaufwand notwendig, um die Kommunikation von nichtzusammenhängenden Daten zu vermeiden [71]. Durch eine eindimensionale Datenverteilung wird daher entweder der lokale oder der globale Zugriff auf nichtzusammenhängende Daten in besonders nachteiliger Weise erzwungen. Der Überlegenheit der eingeführten, multidimensionalen Datenzerlegung liegt zum einen im optimalen, lokalen Datenzugriff auf Datenobjekte mit ausgeglichenen, lokalen Dimensionsgrößen. Zum

[2]Die lineare Speicherstruktur eines multidimensionalen FORTRAN Feldes ist von der ersten zur letzten Dimension geordnet. Vektoren der ersten Dimension sind zusammenhängend, während den Vektoren der weiteren Dimensionen die Datenschrittweite der vorhergehenden Dimensionen auferlegt ist.

anderen vereinfacht sich im globalen Datenzugriff das Kommunikationsmuster ganz entscheidend, wie im nächsten Abschnitt aufgezeigt wird.

3.3 Kommunikationsalgorithmus

Um den Zugriff auf globale Daten in den verschiedenen Dimensionen für die FBR-DVR-Transformationen zu realisieren, wird ein skalierbares Kommunikationsschema entwickelt, das zwei grundsätzliche Probleme in der Parallelisierung pseudospektraler Algorithmen auf multidimensionalen Gittern berücksichtigt: Zum einen führt der schrittweise Zugriff auf nichtzusammenhängende Daten *(strided data access)* zu einer mangelhaften Ausnutzung des Cache-Speichers auf cache-basierten Architekturen und der verfügbaren Speicherbandbreite im allgemeinen. Dieser Nachteil wird bei der Bewältigung großer, multidimensionaler Felder in quantendynamischen Studien zu einem rechentechnisch relevanten Problem. Mit der Datenzerlegung in allen Koordinaten wird für die lokalen Operationen die ineffiziente Verarbeitung nichtzusammenhängender Daten minimiert. Dementsprechend soll der Kommunikationsalgorithmus den globalen Zugriff auf Daten in den hinteren Dimensionen ohne die besonders aufwendige Kommunikation nichtzusammenhängender Daten ermöglichen.

Desweiteren implizieren die FBR-DVR-Transformationen ein komplexes Kommunikationsmuster. Da die Transformationen alle Datenelemente koppeln, wird das globale Datenobjekt auf einmal benötigt. Vor allem die FFTs können nur mit sehr großem Aufwand in wiederum gekoppelte Subalgorithmen zerlegt werden [72]. Diese nichtabbaubare, globale Datenkopplung führt auf die aufwendigste Kommunikation vom Typ *all-to-all*: Jedes PE sendet und empfängt unterschiedliche Daten zu bzw. von jedem anderen PE. Im Vergleich dazu kann die globale Kopplung einer Matrix-Vektor-Multiplikation unterteilt und der Datenaustausch auf eine Ringkommunikation reduziert werden [70]. Auch approximative *Finite Difference (FD)* Methoden würden infolge der Kopplung nächster Nachbarn in die einfache Ringkommunikation resultieren [73]. Die optimierten, minimalen Darstellungen, die sehr gut skalierenden Transformationen und die hohe Genauigkeit pseudospektraler Algorithmen haben somit aufgrund der hohen Informationsdichte der Beschreibung die komplexe Kopplung bzw. Kommunikation als Konsequenz. Das auszuführende Kommunikationsschema soll daher zumindest die unumgängliche *all-to-all* Kommunikationsstruktur bis auf das physikalisch notwendige Kopplungsmuster reduzieren. Im folgenden wird aufgezeigt, daß diese zwei Zielsetzungen für den Kommunikationsalgorithmus gerade durch die vorteilhafte Nutzung der Multidimensionalität des Systems umgesetzt werden.

Die erforderliche Kommunikation auf der eingeführten Datenverteilung besteht in der Zusammenstellung und Rückverteilung von globalen Daten vor bzw. nach den FBR-DVR-Transformationen. Um die Wellenfunktion in einer bestimmten Koordinate zu transformieren, werden globale Vektoren in der zugehörigen Dimension von allen beteiligten PEs gesammelt *(gathering)*. Dabei wird die Parallelität der Datenverarbeitung durch eine entsprechende, zusätzliche Zerlegung der Wellenfunktion in einer anderen Koordinate aufrechterhalten. Nach erfolgter Transformation wird die ursprüngliche Datenzerlegung durch Verteilen der globalen Vektoren unter den PEs wiederhergestellt *(scattering)*.

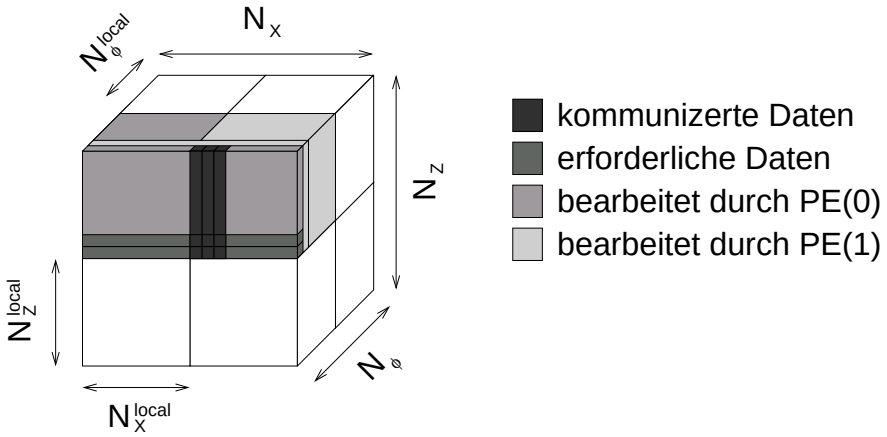

Abbildung 3.4: Kommunikationsalgorithmus für FBR-DVR-Transformationen in den Koordinaten der hinteren Dimensionen: Exemplarisch wird ein 3D Ausschnitts $\{Z, X, \varphi\}$ der 4D Wellenfunktion $\psi(Z, X, \varphi, \vartheta)$ auf einen Würfel aus $2 \times 2 \times 2$ PEs verteilt.[4] Das globale Feld der Größe $N_Z \times N_X \times N_\varphi$ wird in lokale Felder der Größe $N_Z^{\text{local}} \times N_X^{\text{local}} \times N_\varphi^{\text{local}}$ zerlegt. Für eine FFT in der Koordinate X sind kommunizierte und erforderliche Daten sowie die beteiligten PEs durch ein Farbschema gekennzeichnet. Eine detaillierte Beschreibung des Kommunikationsalgorithmus wird im Text gegeben.

Wird dieses Vorgehen auf die Transformation in Koordinaten der hinteren Dimensionen angewandt, stellt sich die Frage nach der Vermeidung

[4]Die Ordnung der Koordinaten in den einzelnen Dimensionen richtet sich nach dem optimalen, lokalen Datenzugriff auf FORTRAN Felder: Von schnell transformierenden Koordinaten mit großer Basisdarstellung in den vorderen Dimensionen zu weniger schnell transformierenden Koordinaten mit kleiner Basisdarstellung in den hinteren Dimensionen.

der Kommunikation von nichtzusammenhängenden Daten. Gewöhnlich wird dieses Problem lokal gelöst: Durch Transponierung der lokalen Wellenfunktion wird die entsprechende Dimension in die erste Position gebracht [71]. Diese Methode ist jedoch für große Felder in hochdimensionalen Studien aufgrund des schrittweisen, lokalen Datenzugriffs vollkommen ungeeignet. Deshalb werden die Zugriffe auf nichtzusammenhängende Daten effizient in äußeren Schleifen „versteckt“ und die auftretenden Datenschrittweiten durch die Datenverarbeitung in den inneren Schleifen neutralisiert. Abb. 3.4 illustriert dieses Konzept anhand eines 3D Auschnittes $\{Z, X, \varphi\}$ der 4D Wellenfunktion $\psi(Z, X, \varphi, \vartheta)$.[4] Durch die beispielhafte Verteilung des globalen Feldes der Größe $N_Z \times N_X \times N_\varphi$ auf einen Würfel aus $2 \times 2 \times 2$ PEs resultieren lokale Felder der Größe $N_Z^{\text{local}} \times N_X^{\text{local}} \times N_\varphi^{\text{local}}$. Für die FFT in der Koordinate X ist die Bildung globaler Vektoren in der zweiten Dimension mittels direkter Kommunikation unvorteilhaft, da die zu kommunizierenden Daten die Datenschrittweite der ersten Dimension aufweisen. Deshalb werden 2D Felder, lokal in Z und global in X, aufgebaut, indem lokale Vektoren der ersten Dimension kommuniziert werden. Um die Datenverarbeitung gleichmäßig zu verteilen, erzeugt und transformiert jedes PE ein anderes 2D Feld bezüglich der Koordinate φ in der dritten Dimension: Für das einfache Beispiel von nur zwei Kommunikationspartnern bearbeiten PE(0) und PE(1) jeweils 2D Felder mit ungeradem bzw. geradem Index in φ. Nachdem die FFTs der 2D Felder in X abgeschlossen sind, wird die originale Datenverteilung durch entsprechend inverse Kommunikation wiederhergestellt.

Dieser vollständig symmetrische Kommunikationsalgorithmus garantiert einen perfekten Lastenausgleich für die FBR-DVR-Transformationen: Alle PEs bewältigen den gleichen Kommunikationsaufwand und verarbeiten die gleiche Datenmenge. Unter Nutzung der Multidimensionalität der Datenobjekte wird beim Zugriff auf globale Daten in den hinteren Dimensionen die Kommunikation von nichtzusammenhängenden Daten vermieden. Die gesamte Kommunikation erfolgt in der ersten Dimension, die globalen Daten werden in der benötigten Dimension aufgebaut, und die Datenverarbeitung wird in einer weiteren Dimension verteilt. Die schrittweisen, lokalen Datenzugriffe werden auf die Transponierung der temporär erzeugten 2D Felder vor und nach der Transformation reduziert. Desweiteren können in dem Kommunikationsschema Winkelkoordinaten mit ihrer komplizierten Datenverteilung leicht in den hinteren Dimensionen berücksichtigt werden. Die aus der verwickelten, lokalen Anordnung der Datenelemente resultierenden Datenschrittweiten treten dann nur in den äußeren Schleifen auf. Der Kommunikationsalgorithmus gestaltet sich für den globalen Datenzugriff in den anderen Dimensionen ähnlich. Im Falle der Koordinate Z in der ersten Dimension fallen kommunizierte und erforderliche Dimension zusammen. Für die Win-

kelkoordinaten φ und ϑ in den hinteren Dimensionen ändern sich entsprechend die benötigte Dimension und die Dimension, in der die Datenverarbeitung verteilt wird. Dabei werden in der angeforderten Dimension nur die relevanten FBR Datenelenente berücksichtigt. Analog kann das Kommunikationsschema durch Hinzufügen der Koordinaten Y und r in zwei weiteren Dimensionen auf eine 6D Quantendynamik verallgemeinert werden.

Im Vergleich mit der Kommunikation infolge einer konventionellen, eindimensionalen Datenzerlegung realisiert der vorgestellte Kommunikationsalgorithmus ein reduziertes Kommunikationsmuster auf der Datenverteilung in allen Freiheitsgraden: Für eine Datenzerlegung in nur einer Koordinate sind die FBR-DVR-Transformationen in den nichtverteilten Koordinaten lokale Operationen. Die gesamte *all-to-all* Kommunikation wird jedoch für die Transformation in der verteilten Koordinate notwendig. Folglich besteht das nachteilige *all-to-all* Kommunikationsmuster für die triviale Prozessgruppe mit allen PEs [68, 69, 71]. Dabei erfordert die HAMILTON-Operation zwei Kommunikationszyklen, in denen die Wellenfunktion jeweils einmal kommuniziert wird. Die Transformationen in den nichtverteilten Koordinaten werden im voraus durchgeführt. Danach erfolgt der erste Kommunikationszyklus und die Transformation in der verteilten Koordinate. Die lokalen Operationen in der korrespondierenden Darstellung werden sofort auf der geänderten Datenverteilung ausgeführt. Nach der Rücktransformation in der verteilten Koordinate stellt der zweite Kommunikationszyklus die anfängliche Datenverteilung wieder her. Schließlich folgt mit den Rücktransformationen in den nichtverteilten Koordinaten der Wechsel in die ursprüngliche Darstellung.

Das Kommunikationsschema auf der Datenverteilung in allen Dimensionen reduziert dagegen die *all-to-all* Kommunikationsstruktur auf die Kopplungen der FBR-DVR-Transformationen in den verschiedenen Dimensionen: Die Transformation in der Koordinate X erfordert *all-to-all* Kommunikation zwischen nur N_X^{PE} PEs, die in jeweils einer von $N_Z^{\mathrm{PE}} \times N_\varphi^{\mathrm{PE}} \times N_\vartheta^{\mathrm{PE}}$ äquivalenten Prozessgruppen zusammengefaßt sind. Da diese Prozessgruppen disjunkt sind, werden die korrespondierenden *all-to-all* Kommunikationen parallel ausgeführt. In analoger Weise gestaltet sich die Transformation in den anderen Koordinaten. Folglich resultieren aus der Datenzerlegung in allen Freiheitsgraden stark verkleinerte, parallele Kommunikationsräume. Für ein d-dimensionales System erhöht sich dadurch die zu kommunizierende Datenmenge um den Faktor $2d$ im Vergleich zur Kommunikation auf der eindimensionalen Datenverteilung. Die 4D HAMILTON-Operation erfordert bei einer vollständigen Datenzerlegung sechzehn Kommunikationszyklen. Vier Kommunikationszyklen werden in der Datenverarbeitung einer Koordinate durchgeführt, da die originale Datenverteilung sofort nach der Transformation und der Rücktransformation wiederhergestellt werden muß. Dieser Mehr-

aufwand an Kommunikation wird jedoch durch die Dimensionalität des Systems fixiert. Deshalb überwiegen in hochdimensionalen Studien mit großen Basisdarstellungen klar die Vorteile des reduzierten Kommunikationsmusters, durch das der Kommunikationsalgorithmus ein extrem gutes Skalierungsverhalten zeigt (siehe Abschnitt 3.4).

Abschließend werden die numerischen Umsetzungen des präsentierten Parallelisierungskonzeptes mit SHMEM und MPI auf einer CRAY T3E verglichen. Die in Abschnitt 3.2 dargelegte Indexalgebra der Datenzerlegung wird in beiden Implementationen mittels schneller FORTRAN90 Modulfunktionen ausgeführt. In der SHMEM-Umsetzung erfolgt die Kommunikation zwischen zwei PEs mit `GET/PUT` Routinen. Dabei sind lokale Vektoren der ersten Dimension als kommunizierte Datenobjekte vollkommen ausreichend, da die verfügbare Kommunikationsbandbreite schon mit vergleichsweise kleinen Datenlängen ausgeschöpft wird. Im Gegensatz dazu werden in der MPI-Implementation effiziente, kollektive `GATHER/SCATTER` Routinen für die Kommunikation in den Prozessgruppen verwendet. Um die optimale Kommunikationsbandbreite zu erreichen, sind lokale 2D Felder in den ersten zwei Dimensionen als kommunizierte Datenobjekte erforderlich. Dazu wird der vorgestellte Kommunikationsalgorithmus von drei auf vier Dimensionen erweitert: Die global benötigte Dimension wird durch Kommunikation von lokalen 2D Feldern der ersten beiden Dimensionen aufgebaut, während die Datenverarbeitung in einer weiteren Dimension verteilt wird. Diese Anpassung unterstreicht die Flexibilität einer multidimensionalen Parallelisierung bezüglich der Anforderungen der Kommunikationsschnittstelle sowie der verwendeten Hardware. Die FFTs werden in beiden Implementationen mit der CRAY spezifischen Routine `CCFFT` für eine komplexe 1D FFT realisiert. Die Matrix-Vektor-Multiplikation der GLT wird wegen der starken Verknüpfung von Kommunikation und Datenverarbeitung direkt ausgeführt, da sich die Nutzung von BLAS Routinen wegen der zusätzlichen Aufbereitung der Datenobjekte als ineffizient erweist.

3.4 Speedup und Skalierung

Um die Leistungsfähigkeit der umgesetzten Parallelisierung zu untersuchen, wird die Implementation einer Laufzeitanalyse auf *Massively Parallel Processing (MPP)* Systemen unterzogen. Diese DMP-Systeme zeichnen sich durch sehr schnelle Knoten-Netzwerke hoher Verfügbarkeit und Kommunikationsbandbreite aus, auf denen die komplexe Kommunikationsstruktur der Implementation besonders gut bewältigt werden kann. Für die Rechnungen werden die Ressourcen des Rechenzentrums der Max-Planck-Gesellschaft Gar-

ching, eine CRAY T3E 600 mit 512 PEs, und des Zentralinstitutes für Angewandte Mathematik am Forschungszentrum Jülich, eine CRAY T3E 1200 mit 512 PEs, genutzt. Jedes PE der MPP-Systeme CRAY T3E 600/1200 besteht aus einem DEC Alpha EV5 (21164) Prozessor mit 300/600 MHz Taktfrequenz und 128/512 MB lokalem Speicher.

In der Analyse wird die Propagation einer gegebenen Wellenfunktion unterschiedlicher Größe auf einer variierenden Anzahl von PEs ausgeführt. Die Wellenfunktion wird dabei in der Splitnäherung propagiert. Da sowohl der signifikante Rechenaufwand als auch die gesamte Kommunikation auf die FBR-DVR-Transformationen der HAMILTON-Operation entfallen, sind die Ergebnisse dieser Splitpropagation auch charakteristisch für andere Propagationsschemen mit iterativer Anwendung des HAMILTON-Operators wie beispielsweise die CHEBYSHEV-Propagation. Die Abhängigkeit der resultierenden Ausführungszeit $t_{\mathrm{run}}(N, N_{\mathrm{PE}})$ von der globalen Größe N und der Anzahl der verwendeten PEs N_{PE} wird durch sogenannte *speedups* beschrieben [74]. Der *(fixed size) speedup*

$$S = \frac{t_{\mathrm{run}}(N, 1)}{t_{\mathrm{run}}(N, N_{\mathrm{PE}})} \tag{3.7}$$

ist das Verhältnis der Laufzeiten auf einem PE und N_{PE} PEs bei fester Problemgröße N. Im idealen Fall folgt eine lineare Abhängigkeit $S(N_{\mathrm{PE}}) = N_{\mathrm{PE}}$: Die Lösung desselben Problems benötigt auf der verdoppelten Anzahl der genutzten PEs die halbe Laufzeit, da der anfallende Kommunikationsaufwand vernachlässigbar klein ist. Neben dieser Charakterisierung der Kommunikationseffizienz wird im *scaled speedup*

$$S_{\mathrm{scale}} = \frac{N_{\mathrm{PE}}\, t_{\mathrm{run}}(N, 1)}{t_{\mathrm{run}}(N_{\mathrm{PE}}N, N_{\mathrm{PE}})} \tag{3.8}$$

noch das Skalierungsverhalten der Implementation mit der Problemgröße einbezogen. Unter Berücksichtigung der Komplexität des parallelisierten Algorithmus wird dabei die Ausführungszeit für ein Problem der Größe N auf einem PE zur Laufzeit für ein N_{PE}-mal größeres Problem auf N_{PE} PEs ins Verhältnis gesetzt. Die Gl. (3.8) unterstellt aufgrund des Faktors N_{PE} im Zähler einen linear skalierenden Algorithmus. Die ideale Umsetzung realisiert diese lineare Skalierung mit vernachlässigbarem Kommunikationsaufwand, was erneut auf die lineare Abhängigkeit $S_{\mathrm{scale}}(N_{\mathrm{PE}}) = N_{\mathrm{PE}}$ führt.

Die Implementation der parallelisierten 4D Quantendynamik wird auf der CRAY T3E 600 analysiert. Eine Betrachtung des *scaled speedup* bezüglich der gesamten Systemgröße erweist sich als nicht sinnvoll, da der pseudospektrale Algorithmus der HAMILTON-Operation kein einheitliches Skalierungsverhalten zeigt: Die FFTs in Z, X und φ skalieren semilinear, die GLT in ϑ

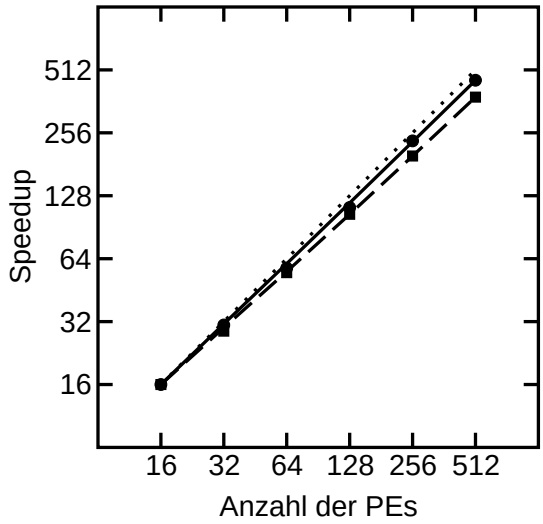

Abbildung 3.5: *Speedup* für ein 4D System $Z \times X \times \varphi \times \vartheta$ der festen, globalen Größe $N = 128 \times 128 \times 64 \times 32$. Die parallele Leistung der SHMEM- (durchgezogen) und MPI- (gestrichelt) Implementation wird mit dem idealen *speedup* (gepunktet) verglichen.

hingegen skaliert quadratisch mit der Dimensionsgröße. Deshalb wird der *speedup* für die Propagation eines 4D Systems $Z \times X \times \varphi \times \vartheta$ der festen, globalen Größe $N = 128 \times 128 \times 64 \times 32 \equiv 512\,\mathrm{MB}$ bestimmt, um die Kommunikationseffizienz der Implementation zu untersuchen. Abb. 3.5 zeigt den Verlauf dieses *speedups* ausgehend von 16 bis hin zu 512 genutzten PEs in doppelt logarithmischer Darstellung. Die Speicheranforderung für die gewählte Systemgröße kann erst von 16 PEs der CRAY T3E 600 bewältigt werden, während die maximal mögliche Verteilung auf 512 PEs in noch nicht zu kleine, lokale Datenobjekte resultiert. Unabhängig von der benutzten Kommunikationsschnittstelle wird ein nahezu linearer *speedup* beobachtet, der ein sehr hohes Verhältnis der erbrachten Rechenleistung zum bewältigten Kommunikationsaufwand dokumentiert. Ein nichtlinearer Fit $f(x) = x^e$ führt für die SHMEM- und MPI-Implementation auf den Exponenten $e = 0.98$ bzw. $e = 0.95$. Die maximal verfügbaren 512 PEs werden mit einer parallelen Effizienz von 89% mittels SHMEM-Kommunikation genutzt, via MPI-Kommunikation werden nur 74% erreicht. Diese ausgezeichnete Kommunikationseffizienz demonstriert das Potential der entwickelten Parallelisierungsstrategie, eine sehr große Anzahl von PEs für hochdimensionale Studien zu nutzen. Die leicht geringere parallele Leistung der MPI-Umsetzung ist auf den zusätzlichen Aufwand durch den Übergang von einseitiger SHMEM- zu zweiseitiger MPI-Kommunikation zurückzuführen.

Desweiteren wird das Skalierungsverhalten der Implementation mit der Bestimmung eines *scaled speedup* in den verschiedenen Dimensionen charakterisiert. Ausgehend von der Propagation eines 4D Systems der festen, lokalen Größe $N_{\text{local}} = 64 \times 64 \times 32 \times 16 \equiv 32\,\text{MB}$ werden die Anzahl der verwendeten PEs und die Größe der betreffenden Dimensionen fortlaufend verdoppelt. Der entsprechende *scaled speedup* wird mit der Gl. (3.8) gebildet. Folglich wird für die semilinear skalierende HAMILTON-Operation ein Algorithmus mit streng linearer Skalierung vorausgesetzt (vgl. Abschnitt 2.4.3). In Abb. 3.6 ist die

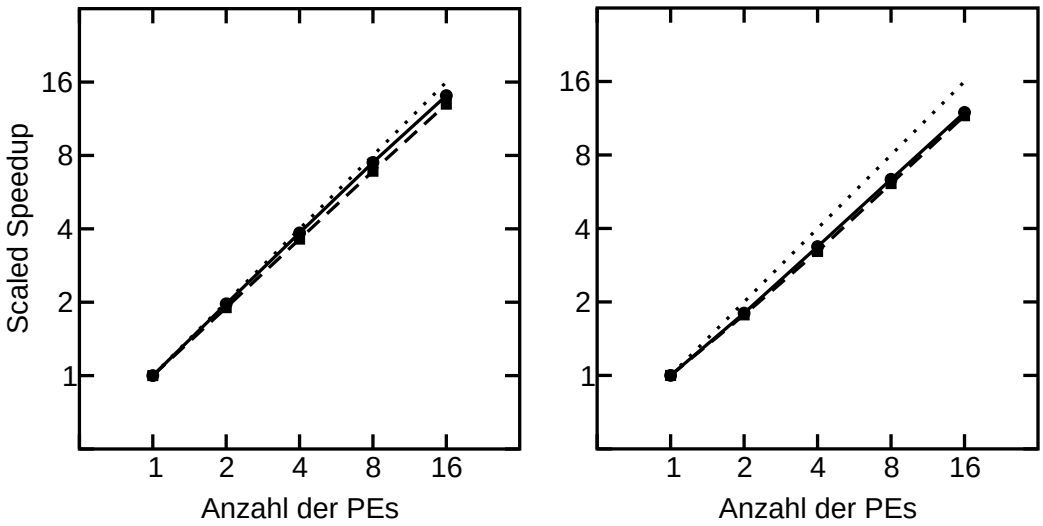

Abbildung 3.6: *Scaled speedup* in den kartesischen Koordinaten Z (links) und X (rechts) für ein 4D System $Z \times X \times \varphi \times \vartheta$ der festen, lokalen Größe $N_{\text{local}} = 64 \times 64 \times 32 \times 16$. Die Größe der betreffenden Dimension wird von 64 auf 1024 Datenelemente skaliert. Die parallele Leistung der SHMEM- (durchgezogen) und MPI- (gestrichelt) Implementation wird mit dem idealen *scaled speedup* (gepunktet) verglichen.

resultierende Abhängigkeit des *scaled speedup* von der Anzahl der genutzten PEs für die kartesischen Koordinaten Z und X doppelt logarithmisch illustriert. Dabei wird die Dimensionsgröße zwischen 64 und 1024 Datenelementen variiert. Alle Kurven zeigen ein fast lineares Verhalten entsprechend dem semilinearen Skalierungsgesetz der FFTs in den kartesischen Koordinaten. Die parallele Leistung in Z (Abb. 3.6, links) ist dem idealen *scaled speedup* am nächsten aufgrund des optimalen, lokalen Zugriffs auf zusammenhängende Daten in der ersten Dimension. Der leicht geringere Anstieg für die MPI-Implementation wird durch die Umordnung von lokalen Vektoren der ersten Dimension in den temporären 2D Feldern verursacht. Im Vergleich dazu wird für die Koordinate X in der zweiten Dimension (Abb. 3.6, rechts)

wegen der Transponierung der temporären 2D/3D Felder eine etwas geringere parallele Leistung erreicht. Da dieser Umordnungsaufwand unabhängig von der eingesetzten Kommunikationsschnittstelle ist, liefern die SHMEM- und MPI-Implementation nahezu gleiche Ergebnisse.

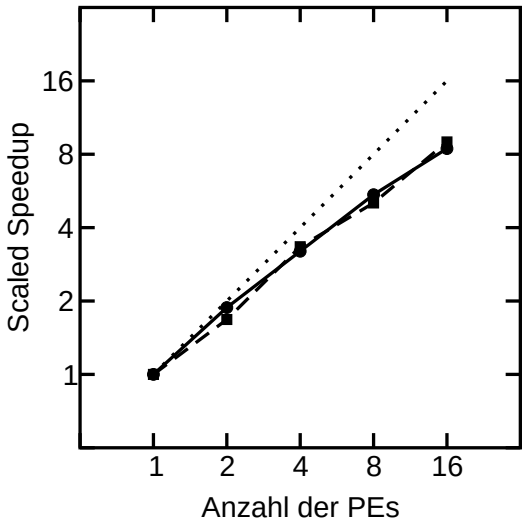

Abbildung 3.7: *Scaled speedup* in den Winkelkoordinaten $\{\varphi, \vartheta\}$ für ein 4D System $Z \times X \times \varphi \times \vartheta$ der festen, lokalen Größe $N_{\text{local}} = 64 \times 64 \times 32 \times 16$. Die Größe der betreffenden Dimensionen wird von 32×16 auf 128×64 Datenelemente skaliert. Die parallele Leistung der SHMEM- (durchgezogen) und MPI- (gestrichelt) Implementation wird mit dem idealen *scaled speedup* (gepunktet) verglichen.

Bei den Winkelkoordinaten sind die FFT in φ und die GLT in ϑ durch die Darstellung in einem nichtdirekten Produkt sehr eng miteinander verbunden. Darüber hinaus haben die beiden Transformationen verschiedene Skalierungsgesetze. Um die entsprechende Konkurrenz von semilinearer und quadratischer Skalierung darzustellen, wird ein gemeinsamer *scaled speedup* in den Winkelkoordinaten $\{\varphi, \vartheta\}$ betrachtet. Die Größe der Dimensionen variiert dabei zwischen 32×16 und 128×64 Datenelementen. Abb. 3.7 zeigt doppelt logarithmisch den Verlauf des so bestimmten *scaled speedup* in Abhängigkeit von der Anzahl der verwendeten PEs. Für beide Implementationen wird ein ganz ähnliches Verhalten beobachtet. Der anfängliche, fast lineare *scaled speedup* wird durch die semilinear skalierende FFT bestimmt. Mit steigender Anzahl der PEs und zunehmenden Dimensionsgrößen konkurriert die quadratisch skalierende GLT stärker. Dies führt zu einem langsamen Abfall der Kurve von dem idealen *scaled speedup* des vorausgesetzten, linear skalierenden Algorithmus. Im Vergleich der Implementationen ist der geringfügig kleinere

scaled speedup der MPI-Umsetzung für zwei bzw. acht PEs auf die unterschiedliche Ausführung der Kommunikation für die GLT zurückzuführen. In der SHMEM-Kommunikation werden mit den `GET/PUT` Routinen nur die für das FBR Datenobjekt relevanten Datenelemente berücksichtigt. Mit den kollektiven `GATHER/SCATTER` Routinen hingegen wird das gesamte DVR Datenobjekt kommuniziert, auch wenn die unnötig kommunizierten Datenelemente nicht weiterverarbeitet werden. Die kollektive Kommunikation wirkt sich dabei für zwei bzw. acht PEs nachteilig aus, da die entsprechenden Datenlängen keine Potenzen von Zwei sind. In der Perspektive kann dieser Mehraufwand durch eine ausschließliche Kommunikation mit `SEND/RECV` Routinen vermieden werden. Der numerische Aufwand der GLT erreicht den der FFT erst für die Dimensionsgrößen 128×64. Somit dominiert die quadratische Skalierung der GLT für typische Basisentwicklungen ($N_\vartheta \propto 10^1$) nicht einmal die Transformation der Winkelkoordinaten, und die gesamte 4D HAMILTON-Operation folgt dem in Abschnitt 2.4.3 vorweggenommenen, semilinearen Skalierungsgesetz.

Um abschließend das gesamte Skalierungsverhalten zu charakterisieren, wird die Parallelisierungsstrategie auf ein quantendynamisches 6D Modellsystem aus allgemeinen FOURIER-Koordinaten angewandt. Der korrespondierende HAMILTON-Operator

$$\hat{H} = \hat{T}(\hat{\mathbf{p}}) + \hat{V}(\mathbf{q}) = \sum_{d=1}^{6} \frac{\hat{p}_d^2}{2m_d} + \hat{V}(\mathbf{q}) \tag{3.9}$$

ist die Summe der Operatoren der kinetischen Energie $\hat{T}(\hat{\mathbf{p}})$ und der potentiellen Energie $\hat{V}(\mathbf{q})$, die von den kanonisch konjugierten Impulsoperatoren $\hat{\mathbf{p}} \equiv \{\hat{p_d}\}$ bzw. Ortskoordinaten $\mathbf{q} \equiv \{q_d\}$ abhängen. Die zeitabhängige Wellenfunktion dieses Modellsystems kann analog zum Abschnitt 2.4.1 in einer FBR Basis aus ebenen Wellen bzw. DVR Basis aus diskreten DIRACschen δ-Funktionen dargestellt werden. Wiederum sind die konjugierten Darstellungen, FBR und DVR, durch FFTs miteinander verbunden. Folglich zeigt die Anwendung des 6D HAMILTON-Operators auf die Wellenfunktion eine einheitliche, semilineare Skalierung mit der gesamten Systemgröße. Auf der Basis von SHMEM-Kommunikation wird in Analogie zum Abschnitt 3.2.1 das Parallelisierungskonzept mit einer blockweisen Datenverteilung in allen Freiheitsgraden realisiert. Die so erhaltene Implementation der parallelisierten 6D Quantendynamik wird auf der CRAY T3E 1200 analysiert. Wegen des uniformen, semilinearen Skalierungsgesetzes der HAMILTON-Operation kann nun ein gesamter *scaled speedup* gemäß Gl. (3.8) bestimmt werden. Die Anzahl der PEs, auf denen die Propagation des 6D Systems erfolgt, wird fortlaufend von 1 bis auf 512 verdoppelt, während die globale Größe N von

$32 \times 16 \times 16 \times 16 \times 16 \times 8 \equiv 256\,\mathrm{MB}$ bis $128 \times 64 \times 64 \times 32 \times 32 \times 16 \equiv 128\,\mathrm{GB}$ entsprechend skaliert wird. Das resultierende Verhalten des so ermittelten

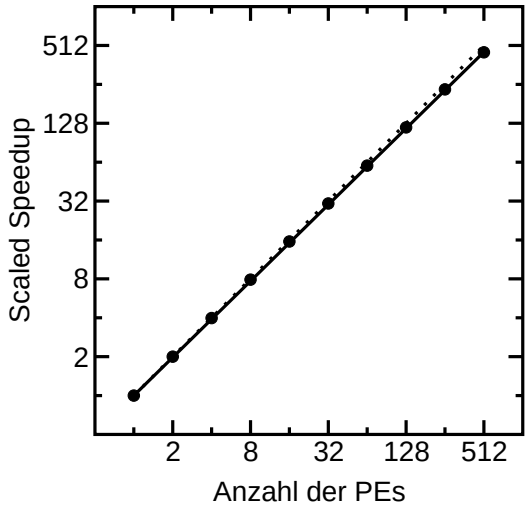

Abbildung 3.8: *Scaled speedup* für ein 6D System $\prod_{d=1}^{6} q_d$ aus FOURIER-Koordinaten. Die globale Größe N wird zwischen $32 \times 16 \times 16 \times 16 \times 16 \times 8$ und $128 \times 64 \times 64 \times 32 \times 32 \times 16$ Datenelementen skaliert. Die parallele Leistung der SHMEM-Implementation (durchgezogen) wird mit dem idealen *scaled speedup* (gepunktet) verglichen.

scaled speedup ist in Abb. 3.8 doppelt logarithmisch gezeigt. Der nahezu lineare Abhängigkeit von der Anzahl der verwendeten PEs beweist eindeutig die Überlegenheit der multidimensionalen gegenüber der eindimensionalen Parallelisierungsstrategie: Obwohl das kommunizierte Datenvolumen bei einer vollständigen Datenzerlegung ab 64 PEs im Vergleich zur Kommunikation auf einer eindimensionalen Datenverteilung $2d = 12$ mal so groß ist, realisiert die Implementation die semilineare Skalierung der HAMILTON-Operation fast ideal: Der nichtlineare Fit $f(x) = x^e$ ergibt einen Exponenten $e = 0.98$, und die maximal verfügbaren 512 PEs werden mit einer parallelen Effizienz von 89% genutzt. Somit wird auch für sehr große Systeme eine ausgezeichnete Kommunikationseffizienz erreicht. Diese hevorragende Skalierbarkeit prädestiniert das multidimensionale Parallelisierungskonzept für den Einsatz in hochdimensionalen Studien mit großen Basisdarstellungen.

Durch das einheitliche Skalierungsgesetz der 6D HAMILTON-Operation kann schließlich mit der Bestimmung des dimensionsaufgelösten *scaled speedup* die Charakteristik des globalen Datenzugriffs in den einzelnen Dimensionen analysiert werden. Bei der Propagation eines 6D Systems der festen, lokalen Größe $N_{\mathrm{local}} = 16 \times 16 \times 16 \times 16 \times 16 \times 16 \equiv 256\,\mathrm{MB}$ werden da-

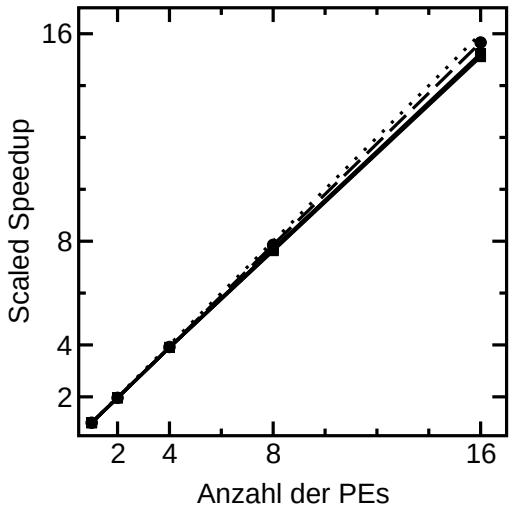

Abbildung 3.9: *Scaled speedup* in den einzelnen FOURIER-Koordinaten $\{q_d\}$ für ein 6D System $\prod_{d=1}^{6} q_d$ der festen, lokalen Größe $N_{\text{local}} = 16 \times 16 \times 16 \times 16 \times 16 \times 16$. Die Größe der betreffenden Dimension wird von 16 auf 256 Datenelementen skaliert. Die parallele Leistung der SHMEM-Implementation in der ersten Dimension (gestrichelt) und den hinteren Dimensionen (durchgezogen) wird mit dem idealen *scaled speedup* (gepunktet) verglichen.

zu wiederum sowohl die Anzahl der genutzten PEs als auch die Größe der betrachteten Dimension fortlaufend verdoppelt. Dabei wird die Dimensionsgröße zwischen 16 und 256 Datenelementen variiert. Abb. 3.9 illustriert die erhaltene Abhängigkeit des *scaled speedup* von der Anzahl der verwendeten PEs für die verschiedenen Dimensionen. Im Gegensatz zu den vorherigen Abbildungen wird auf eine doppelt logarithmische Darstellung verzichtet, da hier nicht die folgerichtige Realisierung des semilinearen Skalierungesetzes diskutiert werden soll. Stattdessen wird eine lineare Darstellung gewählt, um in dem fast linearen Verhalten des *scaled speedup* die sehr geringen Unterschiede bezüglich der einzelnen Dimensionen aufzulösen. Erwartungsgemäß kommt die parallele Leistung in der ersten Dimension dem idealen Verlauf am nächsten wegen des optimalen, lokalen Datenzugriffs. Das wichtige Ergebnis dieser Analyse ist jedoch die ausgezeichnete, parallele Leistung in allen hinteren Dimensionen: Der Kommunikationsalgorithmus realisiert einen einheitlich schnellen, globalen Zugriff selbst auf lokal nichtzusammenhängende Daten in den hinteren Dimensionen ungeachtet der auftretenden Datenschrittweiten. Der geringfügig kleinere Anstieg des *scaled speedup* in den hinteren Dimensionen ist auf die zusätzliche Transponierung der temporären 2D Felder zurückzuführen.