Technische Universität Chemnitz

Sonderforschungsbereich 393

Numerische Simulation auf massiv parallelen Rechnern

L. Grabowsky

MPI–basierte Koppelrandkommunikation und Einfluß der Partitionierung im 3D–Fall

Preprint SFB393/97-17

Zusammenfassung

In der vorliegenden Arbeit wird die Anwendung eines bereits im 2D-Fall benutzten Mechanismus zur MPI–basierten Koppelrandkommunikation [GEW97] auf das 3D–FEM–System SPC PMPo-3D [AMT95] beschrieben. Insbesondere soll der Einfluß der Partitionierung auf die Laufzeit im Vergleich mit den Resultaten für das Originalsystem, für das entsprechende Untersuchungen bereits in [Rei96] durchgeführt wurden, betrachtet werden. Weiterhin wird ein Ausblick auf weitere Optimierungsmöglichkeiten des Verfahrens gegeben.

Preprint-Reihe des Chemnitzer SFB 393

SFB393/97-17 **August 1997**

Inhaltsverzeichnis

Anschrift des Autors:

Lothar Grabowsky
TU Chemnitz
Fakultät für Informatik
D–09107 Chemnitz

`grabowsk@informatik.tu-chemnitz.de`

1 Einführung

Das in [GEW97] vorgestellte Verfahren einer MPI–basierten Koppelrandkommunikation (im folgenden als MPI_CBC bezeichnet) zeigte im 2D–Fall gute Laufzeiteigenschaften. Ein Ziel dieser Arbeit ist es zu untersuchen, inwieweit dies auch im 3D–Fall gilt. Als Vergleichssystem wird dabei eine Programmvariante herangezogen, in der Send–/Recv–Aufrufe sowie kollektive Operationen auf MPI abgebildet werden, die Koppelrandkommunikation aber in der ursprünglichen Form belassen wurde (im folgenden MPI genannt).

In [Rei96] wurde durch verschiedene Partitionierungsverfahren in Verbindung mit der im Ausgangssystem verwendeten Kommunikationsstruktur, zumindest für unregelmäßige Netze, ein deutlicher Effizienzgewinn gegenüber der linearen Verteilung erzielt. Die benutzten spektralen Methoden sind aber relativ aufwendig und daher kaum zur dynamischen Anwendung zur Laufzeit geeignet. In [Rei96] wird davon ausgegangen, daß die Partitionierung lediglich als Preprocessing–Verfahren eingesetzt wird und daher der Aufwand keine größere Rolle spielt. Die Entwicklung effizienter adaptiver Verfahren ist aber eine wichtige Aufgabenstellung. Für diese können aber solche Verfahren nur für die Ausgangsverteilung benutzt werden, im Laufe der Rechnung müssen weniger aufwendige, lokale Verfahren benutzt werden. Aus diesem Grunde sind Kommunikationstechniken erforderlich, die auch mit weniger angepaßten Partitionierungen noch ein effizientes Arbeiten ermöglichen.

Die Beurteilung, ob das vorgestellte Verfahren hierzu bereits geeignet ist oder durch weitere Optimierung eine Basis für ein solches Verfahren darstellen kann, ist die zweite Zielstellung.

2 Die Anwendung des Verfahrens im 3D–Fall

2.1 Spezifika des 3D–Falls

In [GEW97] wurde bereits festgestellt, daß eine Modifikation bezüglich der Numerierung der auszutauschenden Vektoren für den 3–D Fall erforderlich ist.

Dies liegt daran, daß die Identifikation einer Kette hier nicht durch 2 Crosspoint–Nummern, sondern durch (maximal) 4 erfolgt. Damit würde die Umrechnung auf einen einzelnen Wert sehr schnell zu einem Überlauf des verwendeten Integer–Formats führen. Hinzu kommt, daß hier die Anzahl der auszutauschenden Vektoren i.a. ohnehin höher als im 2D–Fall ist, was die Umrechnung auf einen Wert zusätzlich als nicht praktikabel erscheinen läßt.

Aus diesem Grund wurde die Nutzerschnittstelle um einen zusätzlichen Parameter erweitert, der die Länge eines Identifikators bezeichnet. Die Identifikatoren selbst werden dann in Blöcken dieser Länge hintereinander gespeichert.

Ein zusätzliches Problem tritt bei Verwendung des BPX Vorkonditionierers auf. Würden hier wie in den anderen Fällen ausschließlich die Crosspoints zur Bildung des Identifikators benutzt, so wäre dieser nicht mehr lokal eindeutig, was bei der Bildung der Kommunikatoren zu Fehlern führen würde. Hier kann man die Tatsache ausnutzen, daß die Längen der Ketten (wenn diese nicht 0 sind und damit diese Ketten ohnehin nicht aufgenommen werden), die durch identische Crosspoints beschrieben werden, verschieden sind. Somit kann man die

Länge als fünfte Komponente des Identifikators nutzen und damit die lokale Eindeutigkeit sichern.

2.2 Vergleich zur Ausgangsversion

Die folgenden Abschnitte dienen dem Vergleich zu den in [Rei96] gemachten Untersuchungen. Der Aufbau ist daher stark an diese Arbeit angelehnt.

Wie in dieser Arbeit werden als Beispiel die Netze *cube768* (s. Abb. 1) und *spc3-123* (s. Abb. 2) benutzt.

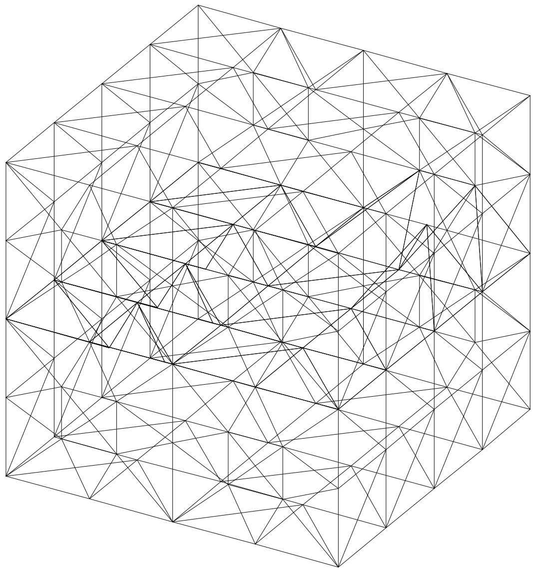


Abbildung 1: FE-Netz *cube768* mit 768 Tetraederelementen

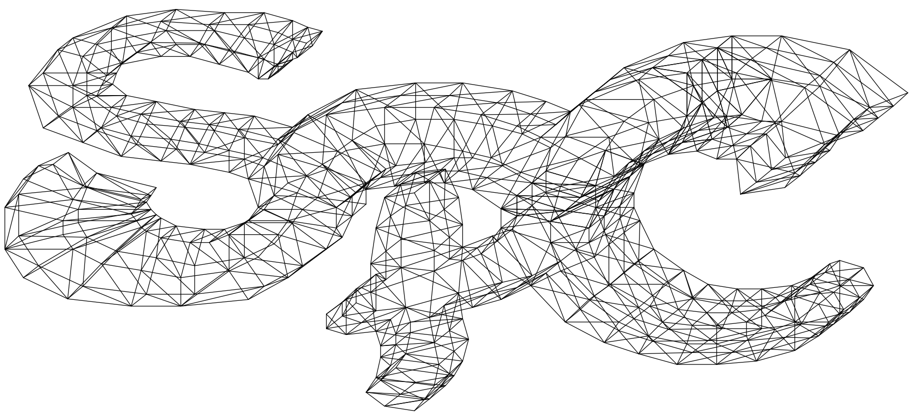


Abbildung 2: FE-Netz *spc3-123* mit 1398 Tetraederelementen

Die verwendeten Partitionierungsverfahren werden hier nicht beschrieben. Dazu sei auf [Rei96] und die darin enthaltenen Literaturangaben verwiesen.

2.2.1 Allgemeine Testbedingungen

Alle Rechnungen wurden auf dem an der TU Chemnitz installierten GC/PowerPlus ausgeführt. Dabei wurde eine 16MB–Version des Programms mit Grafik benutzt. Als Vorkonditionierer wurde in allen Fällen BPX verwendet.

Für das Netz *spc3-123* wurde von 2 bis 16 Prozessoren mit 2, von 16 bis 64 Prozessoren mit 3, für *cube768* von 2 bis 8 Prozessoren mit 2 und von 8 bis 64 Prozessoren mit 3 Verfeinerungsschritten gerechnet.

2.2.2 Lineare Verteilung und Rekursive Spektralbisektion

Die Ergebnisse sind analog zu den in [Rei96] erzielten. Für das sehr regelmäßige Netz *cube768* führt eine lineare Verteilung zu den besten Ergebnissen. Für *spc3-123* ist dagegen die Spektralbisektion (RSB) die günstigere Variante. Allerdings ist erkennbar, daß der Geschwindigkeitsnachteil durch lineare Verteilung für dieses Netz bei Verwendung von MPI_CBC deutlich verringert werden kann. Insgesamt zeigt sich sowohl für lineare Verteilung als auch für RSB mit MPI_CBC für die meisten Situationen eine geringere Laufzeit.

Die bereits in [Rei96] als extrem ungünstig gekennzeichnete zufällige Verteilung wurde nicht betrachet.

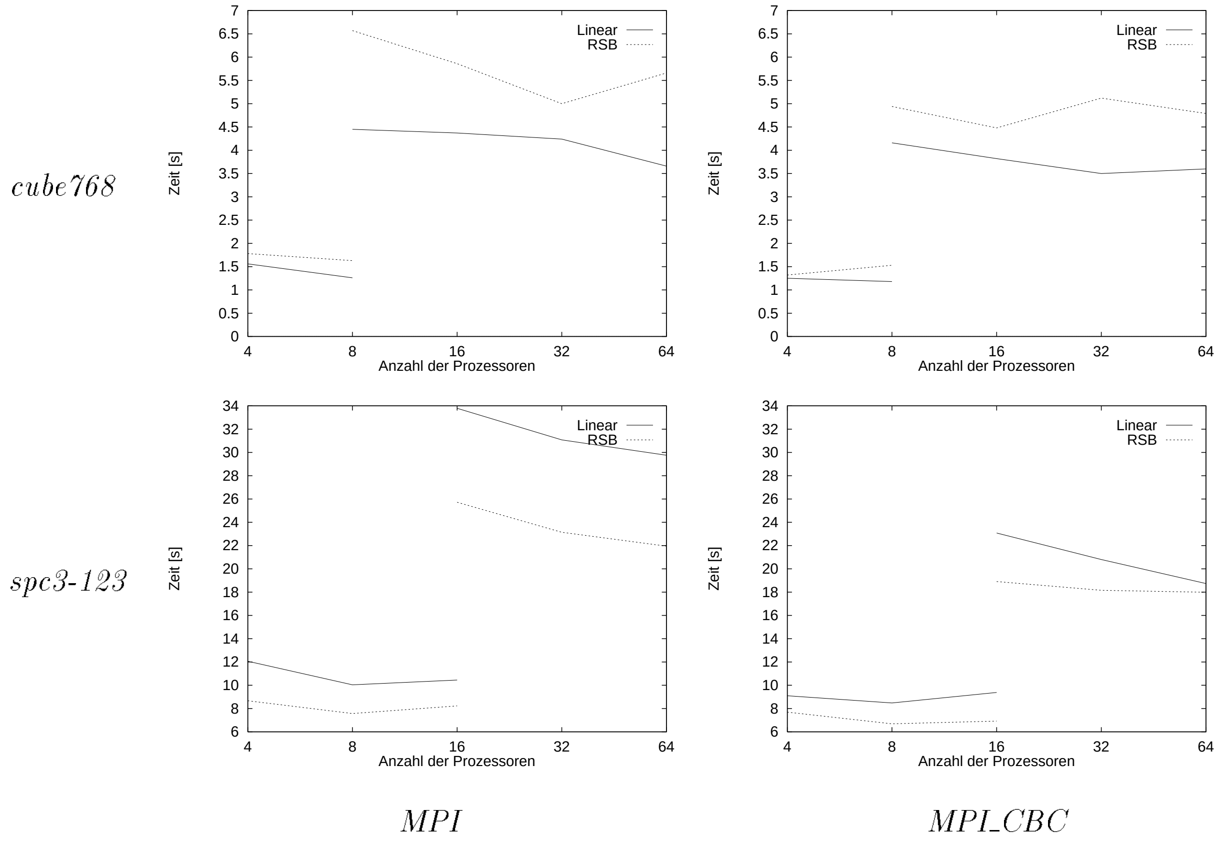


Abbildung 3: Rechenzeiten für lineare Verteilung und RSB

2.2.3 Lokale Nachbesserung mittels KL

Die folgenden Messungen wurden unter Verwendung von Partitionierungen durchgeführt, die aus Ausgangsverteilungen durch RSB entstehen, die mittels der Heuristik von Kerninghan und Lin (KL) nachgebessert wurden.

Die Verbesserung durch KL ist für MPI_CBC nicht in jedem Fall gegeben. Teilweise tritt sogar eine Verschlechterung ein. Für die meisten Fälle hat MPI_CBC aber dennoch laufzeitmäßig Vorteile.

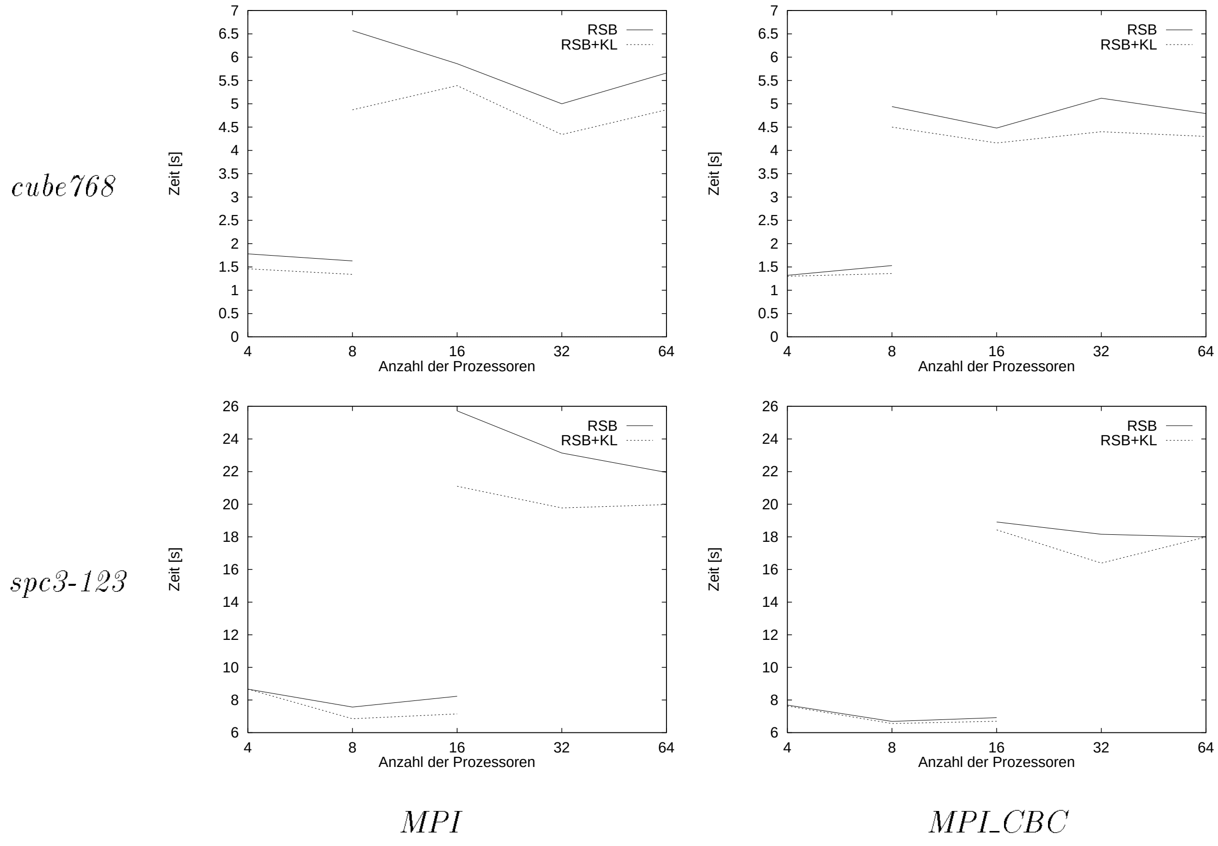


Abbildung 4: Rechenzeiten für RSB und RSB+KL

2.2.4 Quadri– und Oktasektion

Ähnlich zu den in [Rei96] erhaltenen Ergebnissen, konnte durch die Verwendung dieser Verfahren keine nennenswerte Verbesserung erreicht werden. Demgegenüber verschlechterten sich die Ergebnisse in einigen Fällen zum Teil signifikant.

Die Verhältnisse zwischen MPI und MPI_CBC sind dementsprechend ähnlich zu den bisher betrachteten Situationen. Insbesondere für größere Prozessorzahlen und Problemgrößen zeigt MPI_CBC, insbesondere für das Netz *spc3-123*, ein besseres Laufzeitverhalten.

RSO wirkt sich für *cube768* besonders ungünstig auf MPI_CBC aus, hier liegt dessen Laufzeit teilweise höher als für MPI.

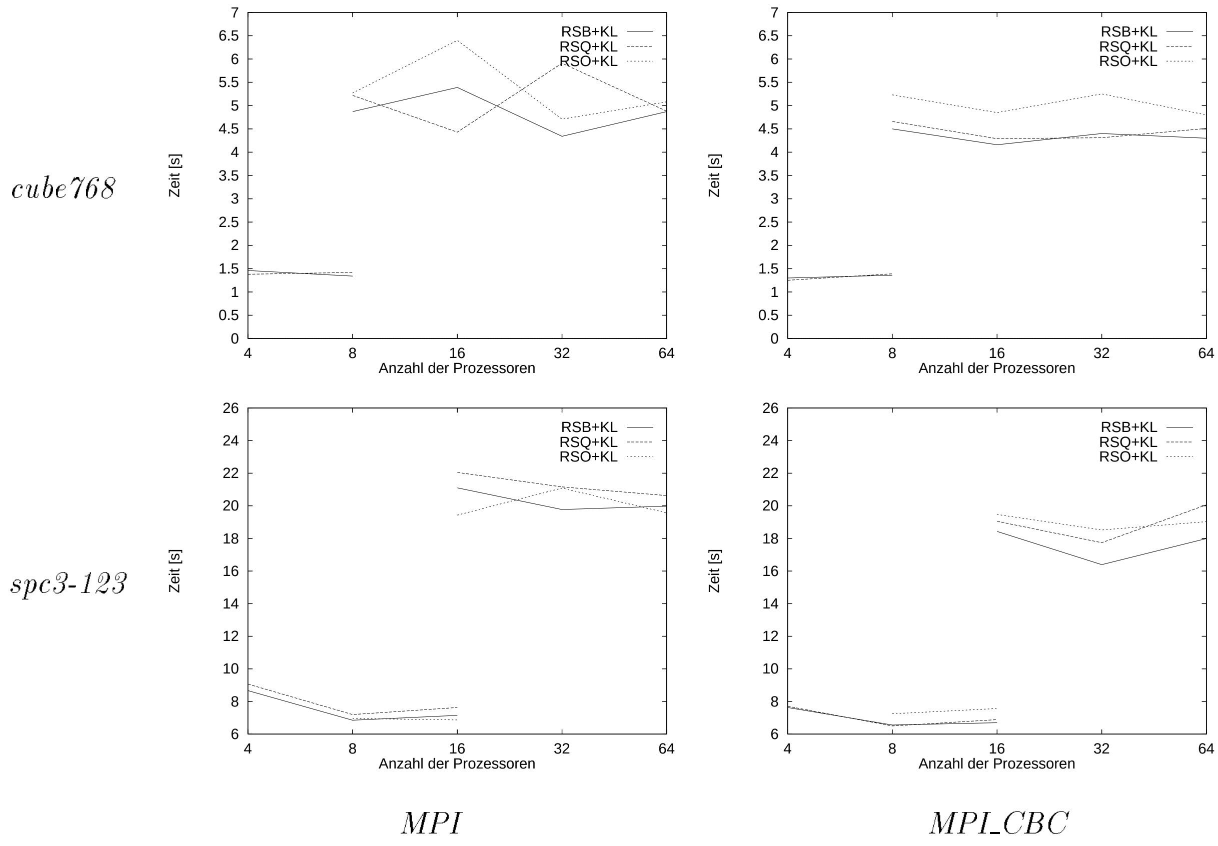


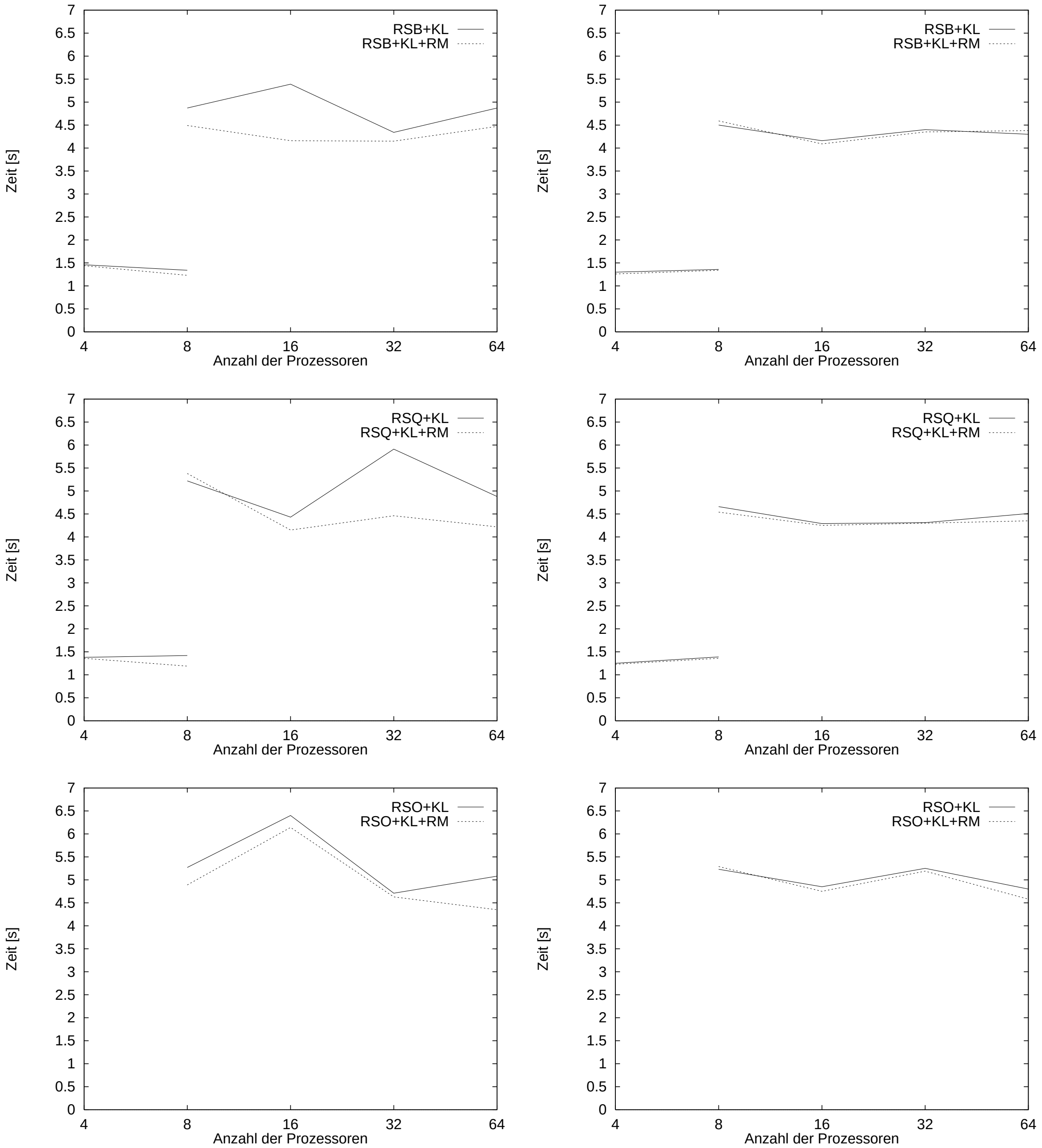
Abbildung 5: Rechenzeiten für RSB+KL, RSQ+KL und RSO+KL

2.2.5 Weitere Optimierungen im Postprocessing

Anpassung der Partitionierung an das Hypercubemodell Für MPI_CBC ist durch dieses Verfahren natürlich keine Verbesserung zu erwarten, da hier nicht auf der Basis des Hypercubes gearbeitet wird. Dennoch ist ein Vergleich interessant, da diese Variante (RM) die Performance des Originalsystems steigert.

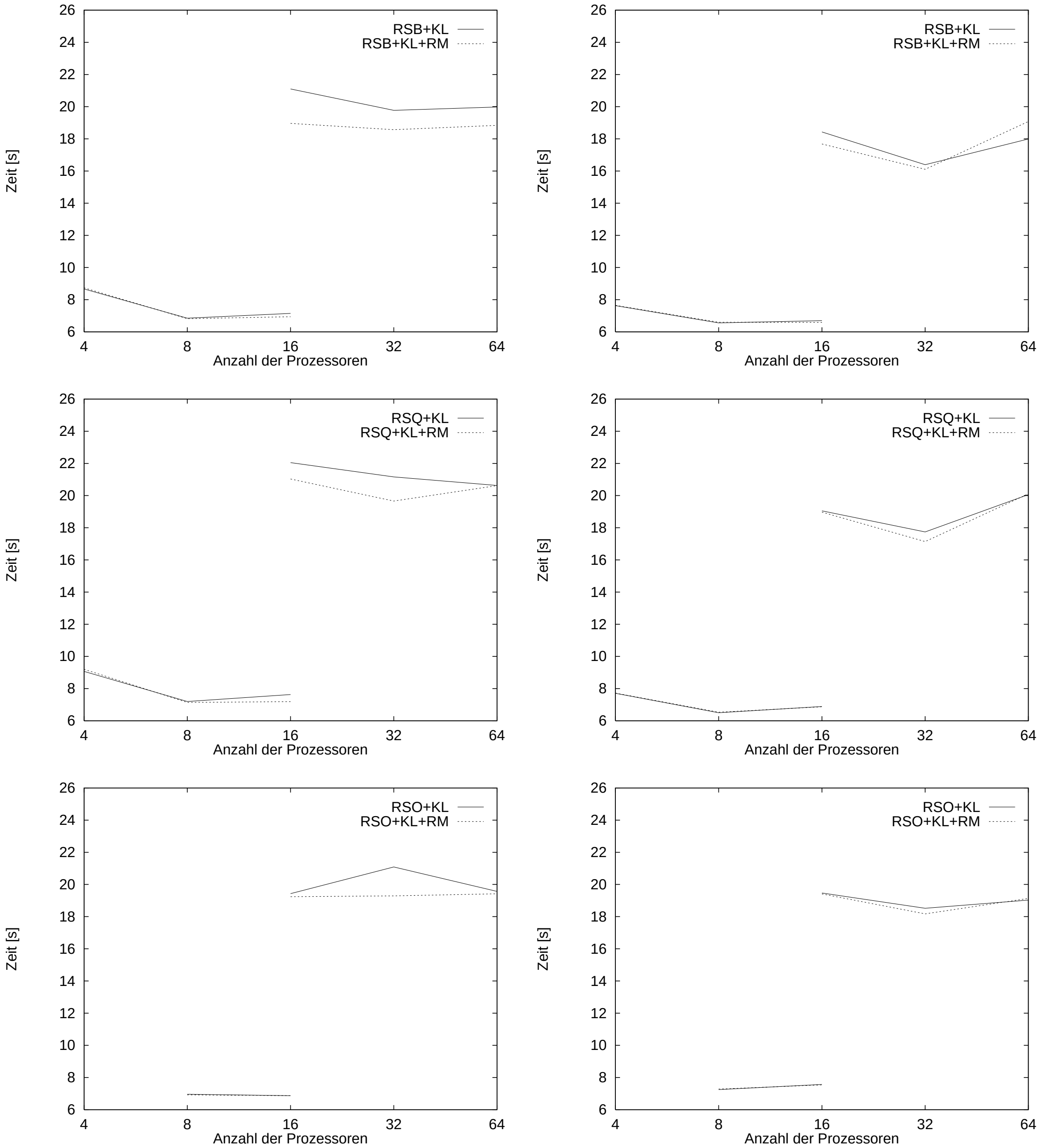
Für das Netz *cube768* bleibt RM nahezu ohne Einfluß auf MPI_CBC, für *spc3-123* führt es jedoch, insbesondere für RSB, zu etwas abweichenden Ergebnissen. Insgesamt lassen sich für diese Verteilungsvariante keine deutlichen Laufzeitvorteile einer der Programmversionen erkennen.

An dieser Stelle wäre zu überlegen, ob mit einem geänderten Postprocessing–Verfahren eine bessere Anpassung an MPI_CBC zu erreichen ist, um dessen Ergebnisse weiter zu verbessern.



MPI *MPI_CBC*

Abbildung 6: Rechenzeiten für RSB, RSQ und RSO mit und ohne spezielle Anpassung der Verteilung an die Hypercube–Topologie für *cube768*



MPI *MPI_CBC*

Abbildung 7: Rechenzeiten für RSB, RSQ und RSO mit und ohne spezielle Anpassung der Verteilung an die Hypercube–Topologie für *spc3-123*

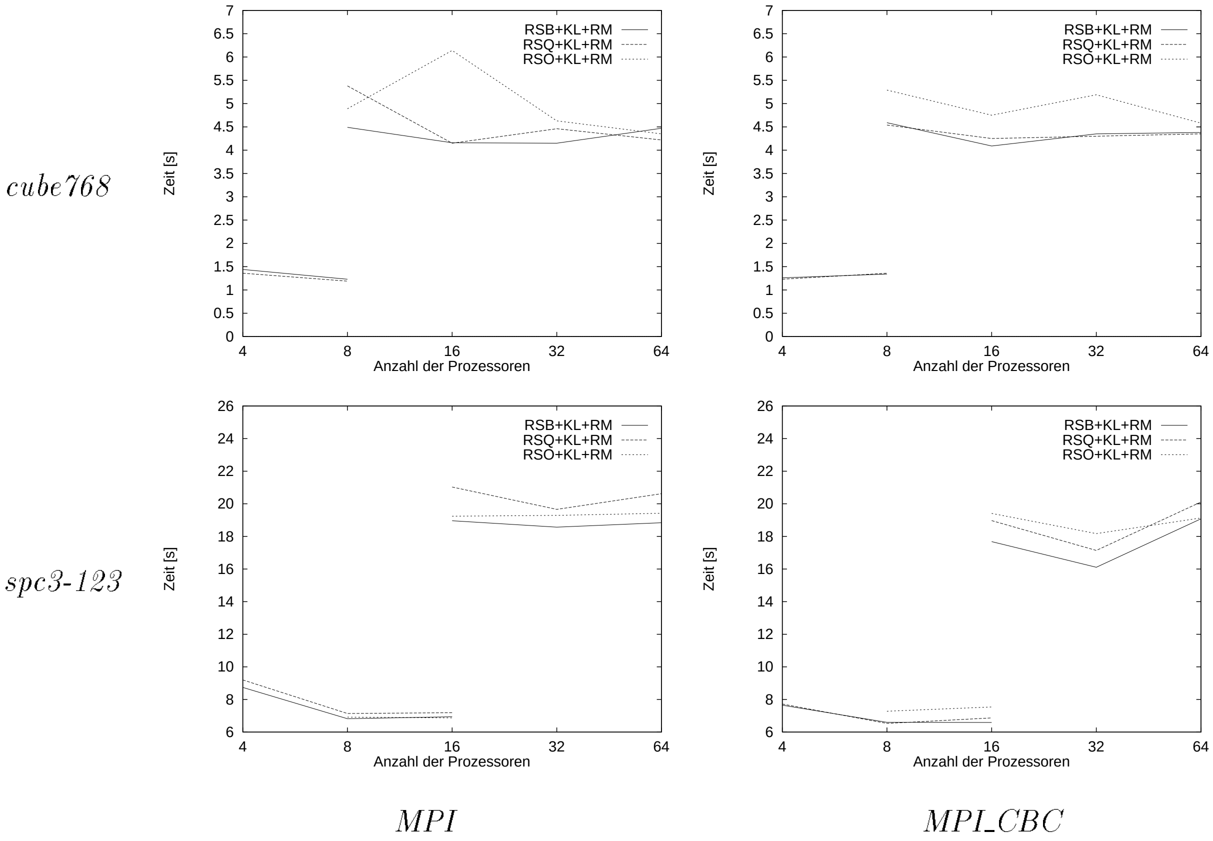


Abbildung 8: Rechenzeiten für RSB+KL+RM, RSQ+KL+RM und RSO+KL+RM

Erhöhung der Anzahl innerer Ecken Die Erhöhung der Anzahl innerer Ecken (IV) wäre für den Fall nützlich, daß in dem entsprechenden Programm eine Überlappung von Rechnung und Kommunikation durchgeführt wird. Dies ist hier nicht der Fall, so daß von IV keine Verbesserung zu erwarten ist.

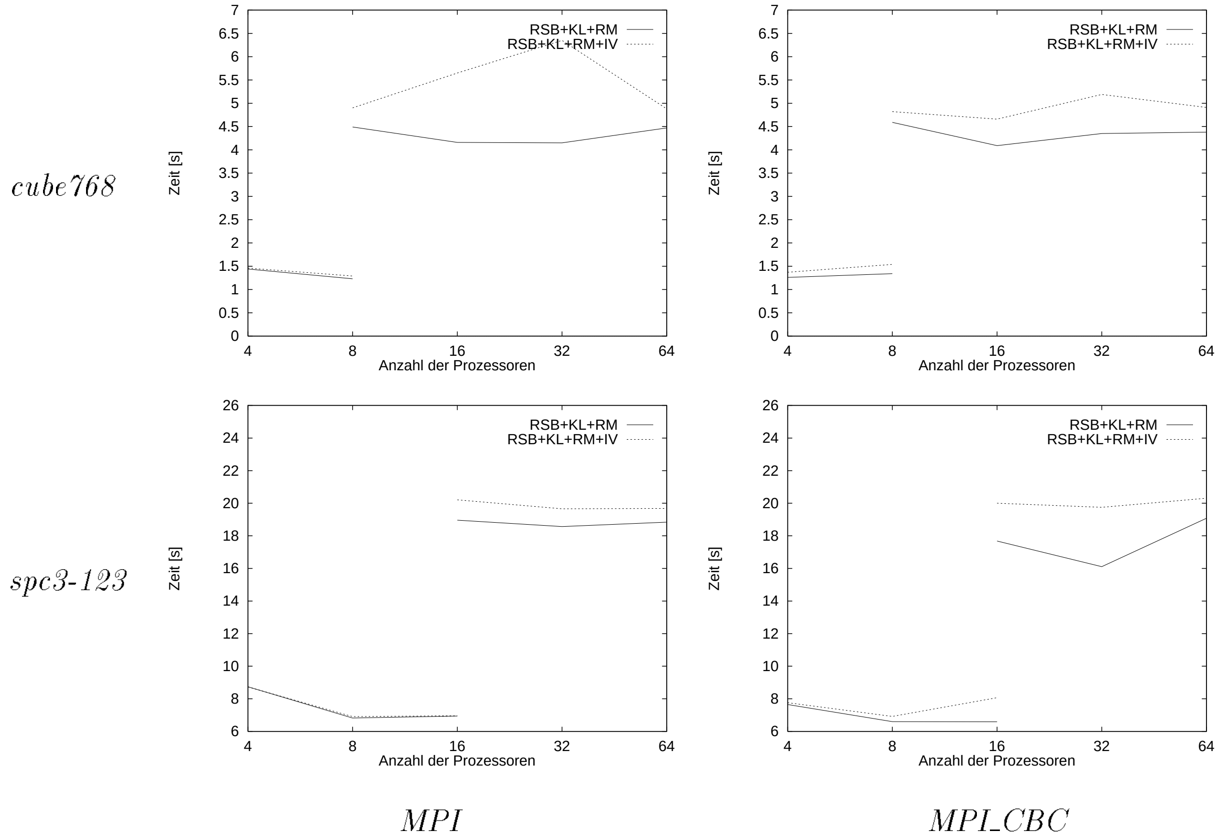
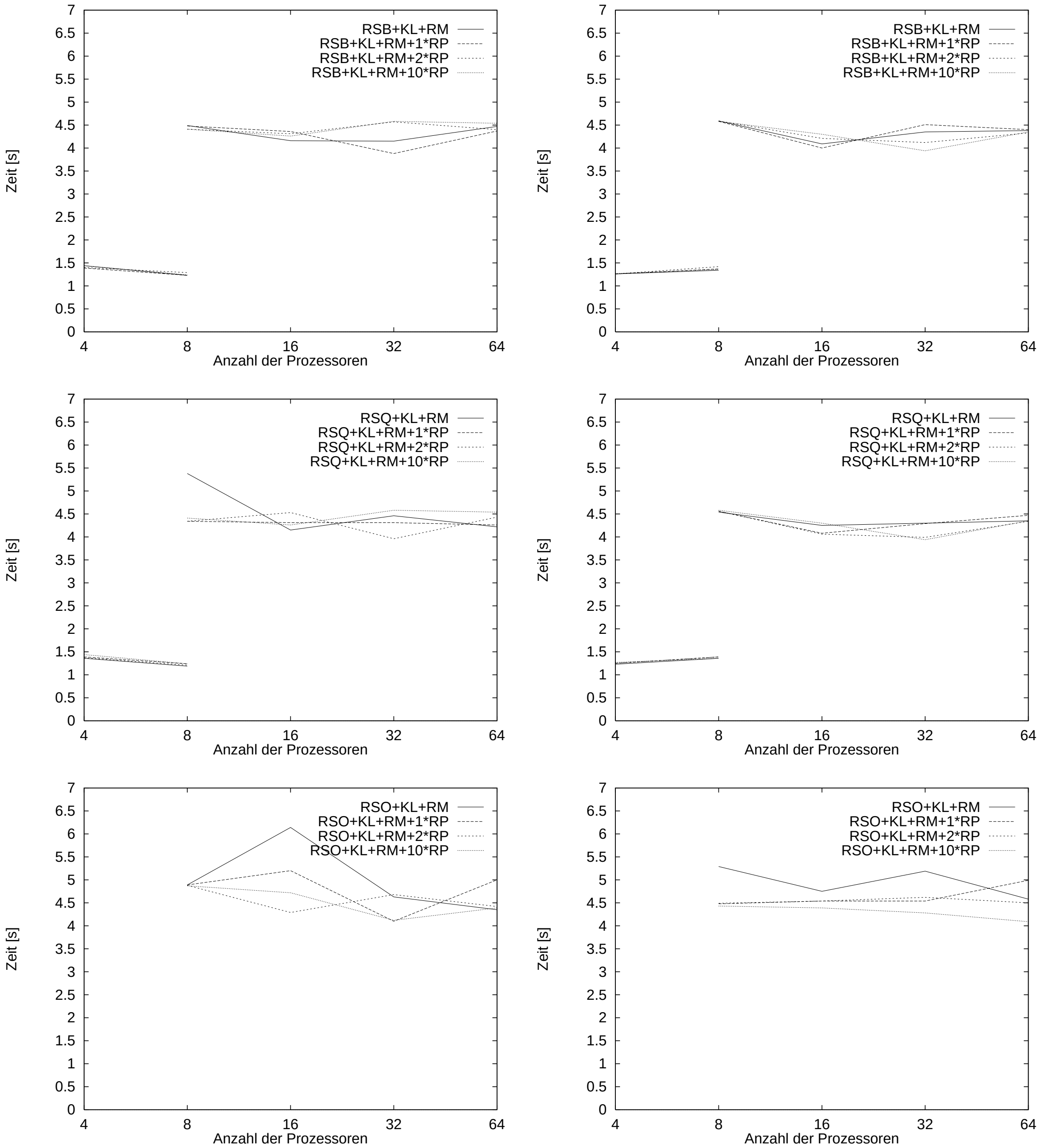


Abbildung 9: Rechenzeiten für RSB+KL+RM und RSB+KL+RM+IV

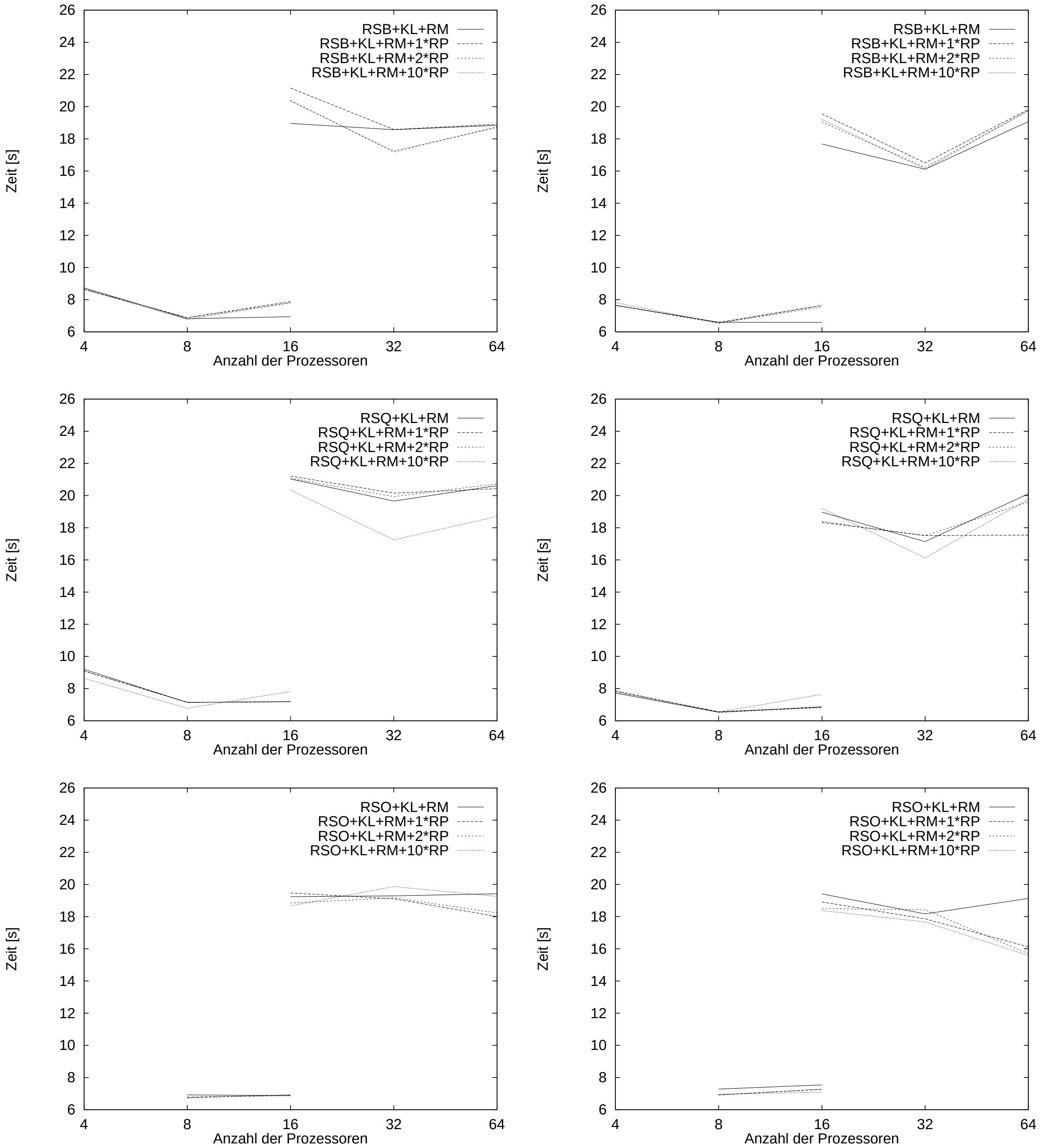
Globale Nachbesserung der Partitionierung Bei diesem Verfahren (RP) wird zunächst das Gewicht der Schnittkanten zwischen jedem Partitionspaar bestimmt. Danach wird in der Reihenfolge von dem Paar mit der größten Grenze zu dem mit der kleinsten zwischen jedem Paar Kernighan–Lin Nachbesserung durchgeführt. Dieser Algorithmus kann in mehreren Iterationsschritten angewendet werden.

Im Gegensatz zu den Ergebnissen ohne globale Nachbesserung ergibt sich hier gegenüber RSB eine z.T. deutliche Verbesserung mit RSO, insbesondere für *spc3-123* und größere Prozessorzahlen.



MPI *MPI_CBC*

Abbildung 10: Rechenzeiten für RSB, RSQ und RSO mit unterschiedlichen Iterationszahlen globaler Nachbesserung für *cube768*



MPI *MPI_CBC*

Abbildung 11: Rechenzeiten für RSB, RSQ und RSO mit unterschiedlichen Iterationszahlen globaler Nachbesserung für *spc3-123*

2.2.6 Terminal Propagation

Dieses Verfahren soll hier nicht weiter beschrieben werden, hierzu sei wiederum auf [Rei96] verwiesen. Da auch hier eine optimale Verteilung an die Hypercube–Topologie das Ziel ist, sind wiederum für MPI_CBC keine Verbesserungen zu erwarten.

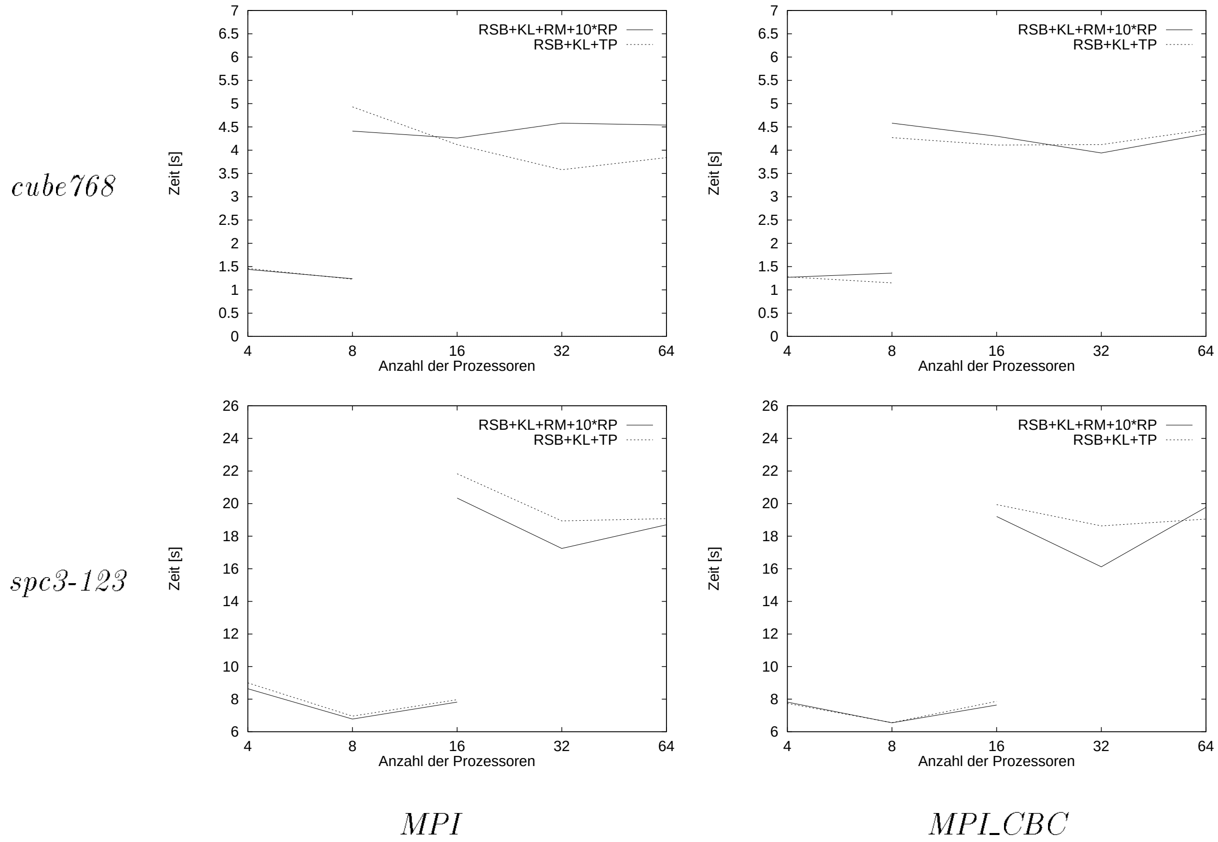


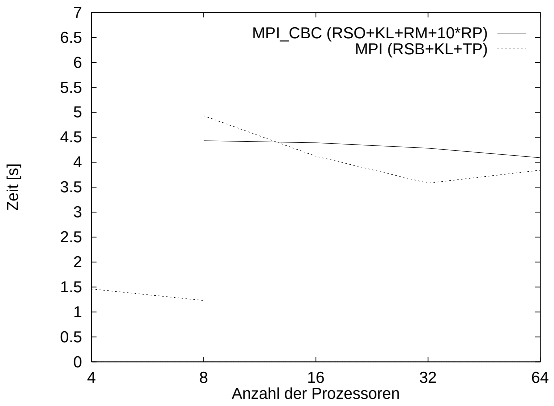
Abbildung 12: Rechenzeiten für RSB+KL+RM+10*RP und RSB+KL+TP

3 Zusammenfassung

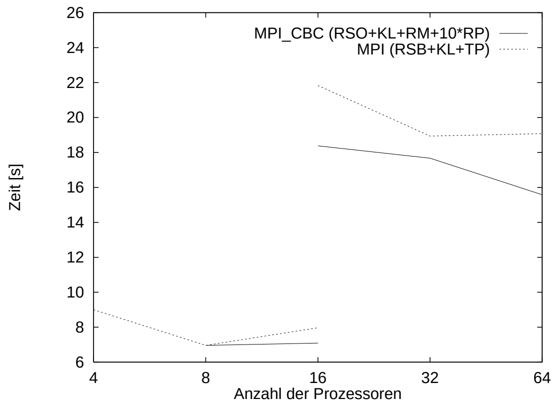
Betrachtet man beide Netze sowie die hier verwendeten Verfahren, so kann für die Originalvariante die Verteilung mit Terminal Propagation als günstigster Fall angesehen werden. Für MPI_CBC lieferte eine globale Nachbesserung der Partitionierung mit 10 Iterationsschritten die besten Ergebnisse.

Die folgende Abbildung zeigt die Ergebnisse für diese Verteilungsvarianten im Vergleich.

Wie in der Abbildung zu erkennen ist, führt MPI_CBC bei *spc3-123* zu einer erheblichen Effizienzsteigerung. Für *cube768* wird die Leistung des Originalsystems nicht erreicht. Gründe hierfür sind zum einen die durch die regelmäßige Struktur des Netzes sehr gute Anpassung an den Hypercube und die relativ große Anzahl serieller Kommunikationsschritte von MPI_CBC für dieses Netz. Der Grund hierfür liegt im Vorhandensein vieler verschiedener Kommunikationsbeziehungen je Knoten, die im gegenwärtigen Verfahren nicht zusammengefaßt werden können.



cube768



spc3-123

Abbildung 13: Rechenzeiten für MPI_CBC und MPI bei der jeweils günstigsten Verteilungsvariante

Die Ermöglichung einer solchen Zusammenfassung und damit die Einsparung serieller Kommunikationsschritte ist Gegenstand gegenwärtiger Arbeiten.

An dieser Stelle muß noch angemerkt werden, daß die Partitionierungen durch die Anpassung an den Hypercube für die Kommunikationsstruktur des Ausgangssystems optimiert sind und daher MPI_CBC in gewisser Weise benachteiligt ist. Für nicht an den Hypercube angepaßte Partitionierungen, wie etwa lineare Verteilung oder Verteilung durch rekursive Spektralbisektion, zeigte MPI_CBC deutliche Laufzeitvorteile. Ein weiterer Schwerpunkt aktueller Arbeiten ist daher die Untersuchung von für MPI_CBC optimierten Partitionierungs– bzw. Postprocessing–Strategien, von denen weitere Effizienzsteigerungen zu erwarten sind.

Ein für den GC/PowerPlus nicht erkennbarer Vorteil des Verfahrens liegt noch in einer anderen Tatsache. Die Koppelrandkommunikation wird hier nicht über explizite Send–/ Recv–Aufrufe, sondern über kollektive MPI–Operationen ausgeführt. Damit eröffnet sich überhaupt erst die Möglichkeit, innerhalb von MPI optimierte kollektive Operationen zu nutzen, insbesondere für heterogene Systeme eine Voraussetzung für effizientes Arbeiten.

Literatur

[AMT95] Th. Apel, F. Milde, and M. Theß *SPC-PM Po 3D Programmers Manual*, Preprint-Reihe der Chemnitzer DFG–Forschergruppe "Scientific Parallel Computing" 95_34, TU Chemnitz, 1995.

[GEW97] L. Grabowsky, Th. Ermer, and J. Werner, *Nutzung von MPI für parallele FEM–Systeme*, Preprint-Reihe des Chemnitzer SFB 393 97-08, TU Chemnitz, 1997.

[Rei96] U. Reichel, *Partitionierung von Finite–Elemente–Netzen*, Preprint-Reihe des Chemnitzer SFB 393 96-18, TU Chemnitz, 1996.

Other titles in the SFB393 series:

96-01 V. Mehrmann, H. Xu. Chosing poles so that the single-input pole placement problem is well-conditioned. Januar 1996.

96-02 T. Penzl. Numerical solution of generalized Lyapunov equations. January 1996.

96-03 M. Scherzer, A. Meyer. Zur Berechnung von Spannungs- und Deformationsfeldern an Interface-Ecken im nichtlinearen Deformationsbereich auf Parallelrechnern.
March 1996.

96-04 Th. Frank, E. Wassen. Parallel solution algorithms for Lagrangian simulation of disperse multiphase flows. Proc. of 2nd Int. Symposium on Numerical Methods for Multiphase Flows, ASME Fluids Engineering Division Summer Meeting, July 7-11, 1996, San Diego, CA, USA. June 1996.

96-05 P. Benner, V. Mehrmann, H. Xu. A numerically stable, structure preserving method for computing the eigenvalues of real Hamiltonian or symplectic pencils. April 1996.

96-06 P. Benner, R. Byers, E. Barth. HAMEV and SQRED: Fortran 77 Subroutines for Computing the Eigenvalues of Hamiltonian Matrices Using Van Loans's Square Reduced Method. May 1996.

96-07 W. Rehm (Ed.). Portierbare numerische Simulation auf parallelen Architekturen. April 1996.

96-08 J. Weickert. Navier-Stokes equations as a differential-algebraic system. August 1996.

96-09 R. Byers, C. He, V. Mehrmann. Where is the nearest non-regular pencil? August 1996.

96-10 Th. Apel. A note on anisotropic interpolation error estimates for isoparametric quadrilateral finite elements. November 1996.

96-11 Th. Apel, G. Lube. Anisotropic mesh refinement for singularly perturbed reaction diffusion problems. November 1996.

96-12 B. Heise, M. Jung. Scalability, efficiency, and robustness of parallel multilevel solvers for nonlinear equations. September 1996.

96-13 F. Milde, R. A. Römer, M. Schreiber. Multifractal analysis of the metal-insulator transition in anisotropic systems. October 1996.

96-14 R. Schneider, P. L. Levin, M. Spasojević. Multiscale compression of BEM equations for electrostatic systems. October 1996.

96-15 M. Spasojević, R. Schneider, P. L. Levin. On the creation of sparse Boundary Element matrices for two dimensional electrostatics problems using the orthogonal Haar wavelet. October 1996.

96-16 S. Dahlke, W. Dahmen, R. Hochmuth, R. Schneider. Stable multiscale bases and local error estimation for elliptic problems. October 1996.

96-17 B. H. Kleemann, A. Rathsfeld, R. Schneider. Multiscale methods for Boundary Integral Equations and their application to boundary value problems in scattering theory and geodesy. October 1996.

96-18 U. Reichel. Partitionierung von Finite-Elemente-Netzen. November 1996.

96-19 W. Dahmen, R. Schneider. Composite wavelet bases for operator equations. November 1996.

96-20 R. A. Römer, M. Schreiber. No enhancement of the localization length for two interacting particles in a random potential. December 1996. to appear in: Phys. Rev. Lett., March 1997

96-21 G. Windisch. Two-point boundary value problems with piecewise constant coefficients: weak solution and exact discretization. December 1996.

96-22 M. Jung, S. V. Nepomnyaschikh. Variable preconditioning procedures for elliptic problems. December 1996.

97-01 P. Benner, V. Mehrmann, H. Xu. A new method for computing the stable invariant subspace of a real Hamiltonian matrix or Breaking Van Loan's curse? January 1997.

97-02 B. Benhammouda. Rank-revealing 'top-down' ULV factorizations. January 1997.

97-03 U. Schrader. Convergence of Asynchronous Jacobi-Newton-Iterations. January 1997.

97-04 U.-J. Görke, R. Kreißig. Einflußfaktoren bei der Identifikation von Materialparametern elastisch-plastischer Deformationsgesetze aus inhomogenen Verschiebungsfeldern. March 1997.

97-05 U. Groh. FEM auf irregulären hierarchischen Dreiecksnetzen. March 1997.

97-06 Th. Apel. Interpolation of non-smooth functions on anisotropic finite element meshes. March 1997

97-07 Th. Apel, S. Nicaise. The finite element method with anisotropic mesh grading for elliptic problems in domains with corners and edges.

97-08 L. Grabowsky, Th. Ermer, J. Werner. Nutzung von MPI für parallele FEM-Systeme. March 1997.

97-09 T. Wappler, Th. Vojta, M. Schreiber. Monte-Carlo simulations of the dynamical behavior of the Coulomb glass. March 1997.

97-10 M. Pester. Behandlung gekrümmter Oberflächen in einem 3D-FEM-Programm für Parallelrechner. April 1997.

97-11 G. Globisch, S. V. Nepomnyaschikh. The hierarchical preconditioning having unstructured grids. April 1997.

97-12 R. V. Pai, A. Punnoose, R. A. Römer. The Mott-Anderson transition in the disordered one-dimensional Hubbard model. April 1997.

97-13 M. Thess. Parallel Multilevel Preconditioners for Problems of Thin Smooth Shells. May 1997.

97-14 A. Eilmes, R. A. Römer, M. Schreiber. The two-dimensional Anderson model of localization with random hopping. June 1997.

97-15 M. Jung, J. F. Maitre. Some remarks on the constant in the strengthened C.B.S. inequality: Application to h- and p-hierarchical finite element discretizations of elasticity problems. July 1997.

97-16 G. Kunert. Error estimation for anisotropic tetrahedral and triangular finite element meshes. August 1997.

97-17 L. Grabowsky. MPI-basierte Koppelrandkommunikation und Einfluß der Partitionierung im 3D-Fall. August 1997.

The complete list of current and former preprints is available via
`http://www.tu-chemnitz.de/sfb393/sfb97pr.html`.