Rechnerevaluation und Benchmarking

Vergleich einer MD-Simulationssoftware mit dem SPEC CPU95-Benchmark auf verschiedenen Workstations

Thomas Gössi

9. 9. 1999

Projekt:
Rechnerevaluation und Benchmarking

Vergleich einer MD-Simulationssoftware mit dem SPEC CPU95-Benchmark auf verschiedenen Workstations

Thomas Gössi

9. 9. 1999

1 Einleitung

Bei wissenschaftlichen Simulationsprogrammen ist es wünschenswert, die Resultate innerhalb eines vernünftigen Zeitrahmens zu erhalten. Deshalb ist es wichtig, einen geeigneten Rechner zu finden, der die gestellte Aufgabe möglichst schnell erledigt. Bei der heute erhältlichen Anzahl Workstations wird die Auswahl eines geeigneten Rechners zum Problem, da man oft nicht weiss, ob nun der gewählte Rechner für die verwendeten Algorithmen geeignet ist.
Standard Benchmarks sollen einen Einblick über die Geschwindigkeit der verschiedenen Workstations schaffen. Die Resultate solcher Benchmarks sind auf dem Web veröffentlicht, sodass man nicht jede verfügbare Workstation heranziehen und selber Benchmarks laufen lassen muss. Ein Problem dabei ist jedoch der Bezug zwischen Benchmarkprogrammen und Simulationsalgorithmus. Um den richtigen Rechner auszuwählen sollten nur Benchmarkwerte zur Evaluation herangezogen werden, welche ein ähnliches Abbild der Simulation darstellen, d.h. Werte von Benchmarkprogrammen, welche die Komponenten eines Rechners in ähnlicher Weise beanspruchen wie die Simulation.
Ziel dieses Projekts ist ein Bezug zwischen der Molekulardynamiksoftware GROMOS™ und SPEC CPU95 herzustellen, damit anhand der SPEC CPU95-Resultate eine geeignete Workstation für GROMOS™ gefunden werden kann. SPEC CPU95 wurde als Benchmark gewählt, da es die Geschwindigkeiten von Rechnern anhand wissenschaftlicher Applikationen ermittelt.

2 Testumgebung

Um einen Bezug zwischen der Molekulardynamiksoftware GROMOS™ und SPEC CPU95 herzustellen wurden beide Applikationen auf verschiedenen Workstations laufen gelassen. In diesem Kapitel werden die beiden Applikationen und die verwendeten Workstations kurz beschrieben.

2.1 GROMOS™

GROMOS™ wurde unter der Leitung von Prof. Dr. van Gunsteren entwickelt und ist ein Simulationspaket für Biomolekularsysteme. Eine solche Simulation berechnet die Interaktionen zwischen Proteinen (Solute) und einer Lösung (Solvent), wobei in einem ersten Schritt die Distanzen zwischen den Atomen berechnet, daraus die Kräfte und Energien und am Ende die neuen Positionen der Moleküle ermittelt werden. Da die Kräfte von weit auseinanderliegenden Atomen praktisch keinen Einfluss mehr auf deren Bewegung haben bzw. deren Werte in der Ungenauigkeit der Fliesskommazahlendarstellung verschwinden, müssen diese nicht berechnet werden. Um Rechenzeit einzusparen wird deshalb um jedes Molekül ein Grenzradius (Cutoff) gezogen und alle Moleküle innerhalb dieses Radius in eine Liste (Pairlist) eingetragen. Diese Pairlist enthält dann nur die Atompaare, deren Einfluss in den folgenden Berechnungsschritten für die Simulation relevant sind. Da die Bewegung der Atome pro Zeitschritt langsam ist, reicht eine Neuberechnung der Pairlist auf jeden 5. - 10. Iterationsschritt aus.
Der Simulationsraum kann entweder in kubischer oder in oktaederischer Form sein, wobei dieser periodisch fortgesetzt wird, um Randbedingungen zu vermeiden. Zudem kann in 3 oder 4 Dimensionen gerechnet werden.
In einer Simulation wird die Interaktion von ca 10’000-30’000 Atomen berechnet, wobei etwa 1000 Iterationen bzw. Zeitschritte nötig sind. Die Dauer eines Zeitschritts beträgt 2 Pikosekunden.
Die Simulation hat relativ viele Parameter wie z.B. Anzahl der Solute- und Solvent-Atome, Form und Grösse des Simulationsraumes, Länge des Cutoff-Radius und Periodizität der Pairlistberechnung, welche die Zeitdauer der Berechnung beeinflussen. Die Berücksichtigung aller dieser Freiheitsgrade würde jedoch den Aufwand dieses Projekts sprengen. Von den Chemikern wurden uns deshalb Parameter für zwei Simulationen zur Verfügung gestellt, welche die meisten Anwendungsfälle abdecken sollen. In unserem Fall handelt es sich um eine Thrombin Molekular Topologie in SPC-Wasser, wobei in 3 Dimensionen je einmal im kubischen und einmal im oktaederischen Simulationsraum gerechnet wird.
Man beachte, dass der oktaederische Simulationsraum auf Grund seiner Struktur kleiner ist als der kubische und deshalb in dieser Simulation weniger Solvent-Moleküle verwendet werden.
Die Parameter für die beiden Simulationen sind auf folgender Seite aufgelistet.

Thrombin Molekular Topologie in SPC-Wasser

	kubisch	oktaederisch
Anzahl Solute Atome:	3078	3078
Anzahl Solvent Atome:	32883	16281
Total Atome:	35961	19359
Anzahl Solute Charge Groups:	1285	1285
Anzahl Solvent Charge Groups:	10961	5427
Total Charge Groups:	12246	6712
Dimensionen der Box:	6.9x7.2x7.42	7.4x7.4x7.4
Cutoff Range:	1.4	1.4
Pairlist:	5. Iteration	5. Iteration
Total Iterationen:	100	100

Der Code von GROMOS™ ist in Fortran geschrieben. Für jeden Rechnertyp wurden die optimalen Compileroptionen ermittelt. Diese sind mit den Parametern der verwendeten Rechner im Anhang angegeben.

2.2 SPEC CPU95

SPEC ist die Abkürzung für Standard Performance Evaluation Corporation. Diese non-profit Organisation hat sich zum Ziel gesetzt hat, Standard Benchmarkprogramme anzubieten, welche ein Vergleich verschiedener Hardware Plattformen und Betriebssystemen erlaubt. Die SPEC CPU ist eine solche Benchmarksuite. Sie wird laufend den Veränderungen und dem Fortschritt der Computertechnologie angepasst. Im Turnus von ca 3-5 Jahren wird jeweils eine neue SPEC CPU Benchmarksuite herausgegeben, welche dann von den Computerherstellern verwendet wird, um die Geschwindigkeit ihrer Computer zu messen und die Daten zu veröffentlichen. Die derzeit aktuelle Version ist die SPEC CPU95. Sie unterteilt sich in die zwei Komponenten CINT95 und CFP95, wovon jede aus einer Serie von rechenintensiven Applikationen besteht. Diese Applikationen stammen von herkömmlichen Anwendungen und testen die Performance der CPU und des Speichers unter realen Bedingungen.

CINT95 besteht aus acht Integer Applikationen, die in C geschrieben sind:

Benchmark	Referenz Zeit	Funktion
099.go	4600	Künstliche Intellegenz, spielt “Go”
124.m88ksim	1900	Motorolla 88100 Simulator lässt Testprogramm simulieren
126.gcc	1700	GCC-Compiler generiert SPARC-Code
129.compress	1800	Komprimiert und dekomprimiert Dateien im Speicher
130.li	1900	LISP Interpreter
132.ijpeg	2400	Grafik Kompression und Dekompression
134.perl	1900	Perl Interpreter bearbeitet Strings und Primzahlen
147.vortex	2700	Datenbankprogramm

Tabelle 2.1: CINT95 Applikationen

CFP95 besteht aus zehn Floating-Point Applikationen. Diese sind in FORTRAN geschrieben:

Benchmark	Referenz Zeit	Funktion
101.tomcatv	3700	Vektorisierte Matrixberechnung
102.swim	8600	Simulation eines Shallow Water Modells in einem 1024x1024 Gitter
103.su2cor	1400	Quantenphysik, Monte Carlo Simulation
104.hydro2d	2400	Astrophysik, Simulation von Hydrodynamik mittels Navier Stokes Formeln
107.mgrid	2500	Simulation eines 3D Potentialfeldes
110.applu	2200	Berechnung von parabolischen partiellen Differentialgleichungen
125.turbo3d	4100	Simulation von isotropischen homogenen Turbulenzen in einem Kubus
141.apsi	2100	Simulation des Wettereinflusses auf die Verteilung von Schadstoffen
145.fpppp	9600	Quantenphysik, Folge von Benchmarks
146.wave5	3000	Plasma Physik, Simulation elektromagnetischer Teilchen mittels der Maxwell Formeln

Tabelle 2.2: CFP95 Applikationen

Für jede dieser Applikationen wird die Zeit gemessen und mit einer Referenzzeit verglichen. Letztere ist die Zeit, die eine SUN SPARC Station10/40 (40MHz SuperSPARC ohne L2 Cache) für die Ausführung der Applikationen benötigt. Das Verhältnis von Ausführungszeit und Referenzzeit ergibt dann die SPEC-Ratio, welche ein Mass für die Geschwindigkeit der untersuchten Maschine ist. Um ein gültiges SPEC-Resultat zu erhalten sind für jede Applikation mindestens drei Durchläufe erforderlich. Vom Median der Durchlaufzeiten wird dann für jede Applikation die SPEC-Ratio errechnet. Das geometrische Mittel aller SPEC-Ratios ergiebt dann die SPEC95-Ratio,

wobei für die Integer und die Floating-Point Applikation jeweils eine separate SPEC95-Ratio, die SPECint95- und die SPECfp95-Ratio, ermittelt wird.
Die Applikationen der CPU95 sind alle im Source Code vorhanden und müssen für die Tests zuerst compiliert werden. Dies lässt die freie Wahl von Betriebssystem und Compilerversion zu, womit neben Rechnertypen auch verschiedene Betriebssysteme und Compilerversionen miteinander verglichen werden können.
Um allgemein verwendbare Resultate zu erhalten schreibt SPEC verschiedene Randbedingungen vor. So dürfen weder Source-Code noch Eingabedaten der Applikationen verändert, keine CPU95-Libraries ersetzt und es sollten standard Compiler Libraries verwendet werden. Bezüglich Compileroptionen lässt SPEC zwei Möglichkeiten zu. In der Basis-Variante müssen alle Applikationen mit denselben Optimierungsflags compiliert werden, wobei nur maximal vier Optimierungsflags verwendet werden dürfen. In der zweiten Variante, Peak-Variante genannt, darf jede Applikation einzeln mit entsprechenden Compileroptionen optimiert werden, wobei die maximale Anzahl der Optimierungsflags nicht festgelegt ist. Die Peak-Variante liegt näher an der Praxis, da man auch im Realfall für jede Applikation die dafür geeigneten Compileroptimierungen verwendet. Die Resultate der Basis-Variante werden in SPEC<int/fp>-Base95 und die der Peak-Variante in SPEC<int/fp>-Peak95 oder SPEC<int/fp>95 angegeben.
Zu allen SPEC-Resultaten müssen die Daten des verwendeten Rechners (CPU, Cache, Memory, Disk), die Version des Betriebssystems und der Compiler sowie die verwendeten Compilerflags angegeben werden, damit auch ein Vergleich mit anderen Resultaten gemacht werden kann. SPEC wird vor allem unter den gängigen Betriebssystemen Windows NT und UNIX verwendet, wobei die Resultate einzelner Rechner für jeweils beide Betriebssysteme veröffentlicht sind.

2.3 Workstations

Die verwendeten Rechner sind handelsübliche Workstations, die zu normalen Zwecken verwendet werden können, also keine Supercomputer. Getestet wurde unter UNIX bzw. Linux im Multiuser Modus, wobei darauf geachtet wurde, dass sich während der Tests keine anderen Benutzer einloggten und die Messungen störten.
Um einen Vergleich zwischen GROMOS™ und SPEC CPU95 machen zu können, schafften wir auf den einzelnen Workstations für beide Programme dieselben Bedingungen. Auf jeder Maschine wurden jeweils beide Applikationen neu compiliert, wobei der zum Rechner gehörige Compiler verwendet wurde. Die Optimierungsflags sind für GROMOS™ und SPEC CPU95 pro Rechner dieselben, was bei CPU95 der Basis-Variante entspricht. Auf die Peak-Variante wurde verzichtet, da in dieser Variante für jede CPU95 Applikation eigene Optimierungsflags nötig und somit die Bedingungen bezüglich GROMOS™ nicht mehr dieselben wären.

In der folgenden Tablle sind alle Rechner mit den wichtigsten Parametern aufgelistet, welche wir für unsere Tests verwendet haben:

Kürzel	Workstation	Prozessor	Betriebssystem
PP-200	DEC DE500 PPro/200	Intel Pentium Pro 200MHz	SuSE Linux 5.2 (2.0.35)
PII-233	DEC DE500 PII/233	Intel Pentium II 233MHz	
PII-350	DEC DE500 PII/350	Intel Pentium II 350MHz	
PII-400	DEC DE500 PII/400	Intel Pentium II 400MHz	
AMD-K6	AMD PC	AMD K6 200MHz	
SUNS10	SUN SPARC Station 10/85	Super SPARC II 85MHz	Solaris 2.5.1
SUNU1	SUN Ultra 1/170	Ultra SPARC 170MHz	
SUNU30	SUN Ultra 30/300	Ultra SPARC 300MHz	
A21064	Alpha Station 400 4/233	Alpha 21064 233MHz	DEC Unix 4.0D
A21164	Alpha Personal Workstation 500	Alpha 21164 500MHz	
SGI	SGI Octane	MIPS R10000 rev. 2.7 195MHz	Irix 6.4

Tabelle 2.3: Verwendete Workstations

Zur besseren Übersicht wird in allen Diagrammen und Grafiken das jeweilige Kürzel verwendet. Eine detailierte Aufstellung der Eigenschaften der einzelnen Rechner, der verwendeten Compiler und Optimierungsflags findet sich im Anhang.

3 Resultate

3.1 GROMOS™

3.1.1 Messmethode

Um die Zeitdauer der Berechnung zu messen haben wir Timing-Routinen in den Code von GROMOS™ eingebaut. Beim Start der Simulation wird ein Timer gestartet, dessen Wert an bestimmten Programmpunkten abgelesen wird. Damit lassen sich die Durchlaufzeiten von einzelnen Programmteilen und Prozeduren bestimmen, was eine weitere Aufschlüsselung und Analyse des Programms erlaubt. Wir haben die Durchlaufzeiten folgender Prozeduren ermittelt:

T_{Pl}	Pairlist:	Berechung der Pairlist
T_{Su}	Solute-Kräfte:	Berechnung der Distanzen und Kräfte aller Protein-Moleküle, wobei die Interaktionen von Proteinen mit Proteinen und Proteinen mit Wassermolekülen berücksichtigt sind.
T_{Sv}	Solvent-Solvent-Kräfte:	Berechung der Distanzen und Kräfte zwischen den Wassermolekülen.
T_{In}	Integration:	Aufaddierung aller Teilkräfte der Moleküle.
T_R	Rest:	Zeit aller restlichen Prozeduren, welche nicht in obige Kategorien fallen. Der Rest wird als Differenz zwischen Zeitdauer der Iteration und Summe der Zeiten obiger Prozeduren ermittelt.
T_{It}	Iteration:	Zeitdauer eines Iterationsschritts
T_T	TOTAL:	Gesamtzeit der Simulation

Tabelle 3.1: GROMOS™-Prozeduren

Für jede Iteration wird die Zeitdauer obiger Prozeduren ermittelt und ausgegeben. Die Aufschlüsselung der einzelnen Zeiten eines Iterationsschritts wird so zu:

$$[T_{Pl}] + T_{Su} + T_{Sv} + T_{In} + T_R = T_{It} \tag{3.1}$$

wobei:

$$T_R = T_{It} - ([T_{Pl}] + T_{Su} + T_{Sv} + T_{In}) \tag{3.2}$$

Diese Formeln gelten für jede 5. Iteration, wo auch die Pairlist berechnet wird. Bei den anderen fällt die Zeitkomponente T_{Pl} weg.
Für einen Vergleich interessiert die Zeitdauer der einzelnen Iterationen kaum. Die Laufzeiten der Iterationen sind unterschiedlich und eine Darstellung aller Werte der 100 Iterationen wäre zu detailiert und unübersichtlich. Wichtig ist nur die Gesamtzeit der Simulation, damit man weiss, wieviel Zeit für eine bestimmte Simulation benötigt wird. Hingegen ist die Aufschlüsselung nach einzelnen Prozeduren weiterhin interessant, da z.B. die Pairlistgenerierung mehr von der Speicherbandbreite abhängig ist als z.B. die Kräfteberechnung, die vor allem die CPU stark belastet.

Die Gesamtzeit der Simulation wird dann zu:

$$\sum_{i=1}^{100} T_{Pl[i]} + \sum_{i=1}^{100} T_{Su[i]} + \sum_{i=1}^{100} T_{Sv[i]} + \sum_{i=1}^{100} T_{In[i]} + T_{RT} = T_T \tag{3.3}$$

wobei

$$T_{Pl[i]} = 0 \quad für \quad (i-1) mod\ 5 \neq 0 \tag{3.4}$$

und

$$T_{RT} = T_T - \sum_{i=1}^{100} (T_{Pl[i]} + T_{Su[i]} + T_{Su[i]} + T_{In[i]}) \tag{3.5}$$

T_{RT} steht für den gesamten Rest. Hier kommt zusätzlich zu den restlichen Prozeduren der Iterationen noch die Zeit für die Initialisierung hinzu.
Die Simulation wurde auf jedem Rechner zehnmal laufen gelassen. Von den zehn Resultaten wurde jeweils der Median genommen und ausgewertet.

3.1.2 Berechnungszeiten der GROMOS™-Simulation

In den folgenden Diagrammen sind die Simulationszeiten von GROMOS™ auf den jeweiligen Rechnern dargestellt.

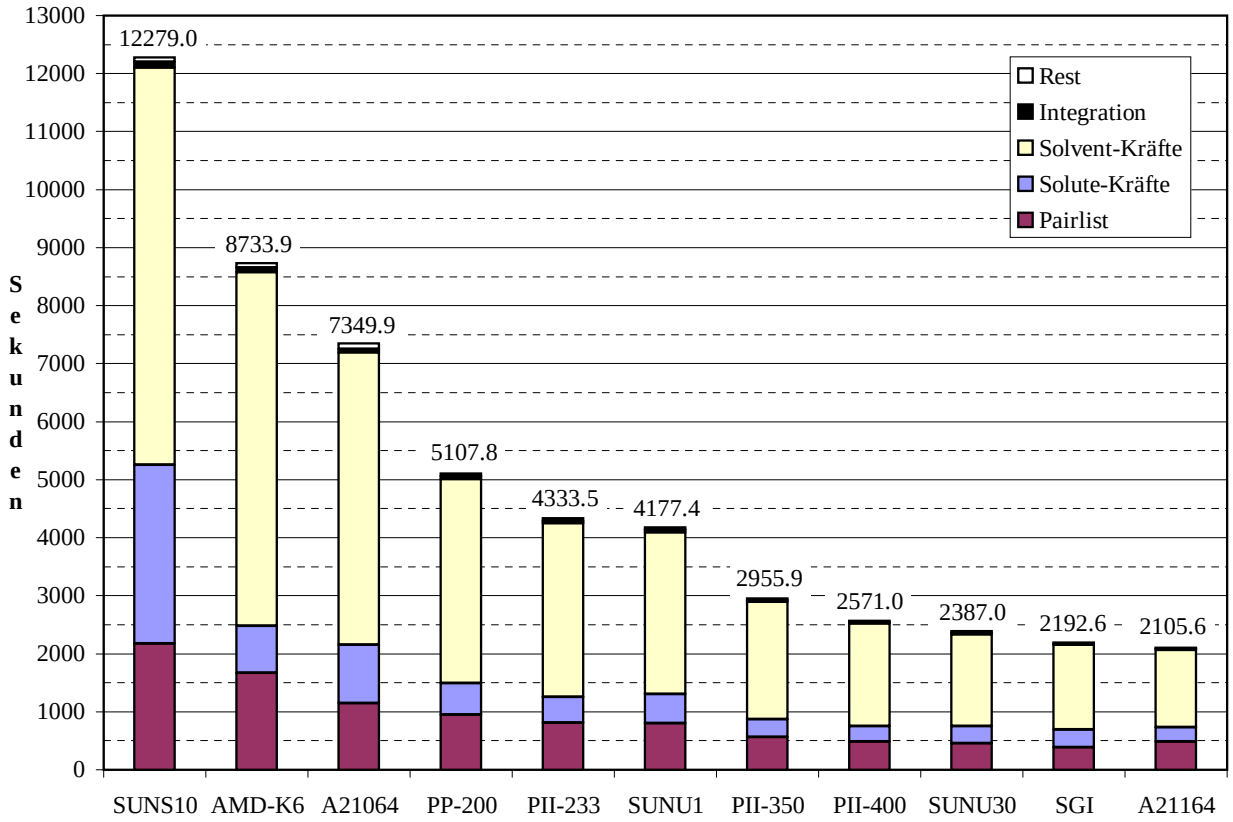


Abb. 3.1: Simulationszeiten der kubischen Thrombin Molekulartopologie

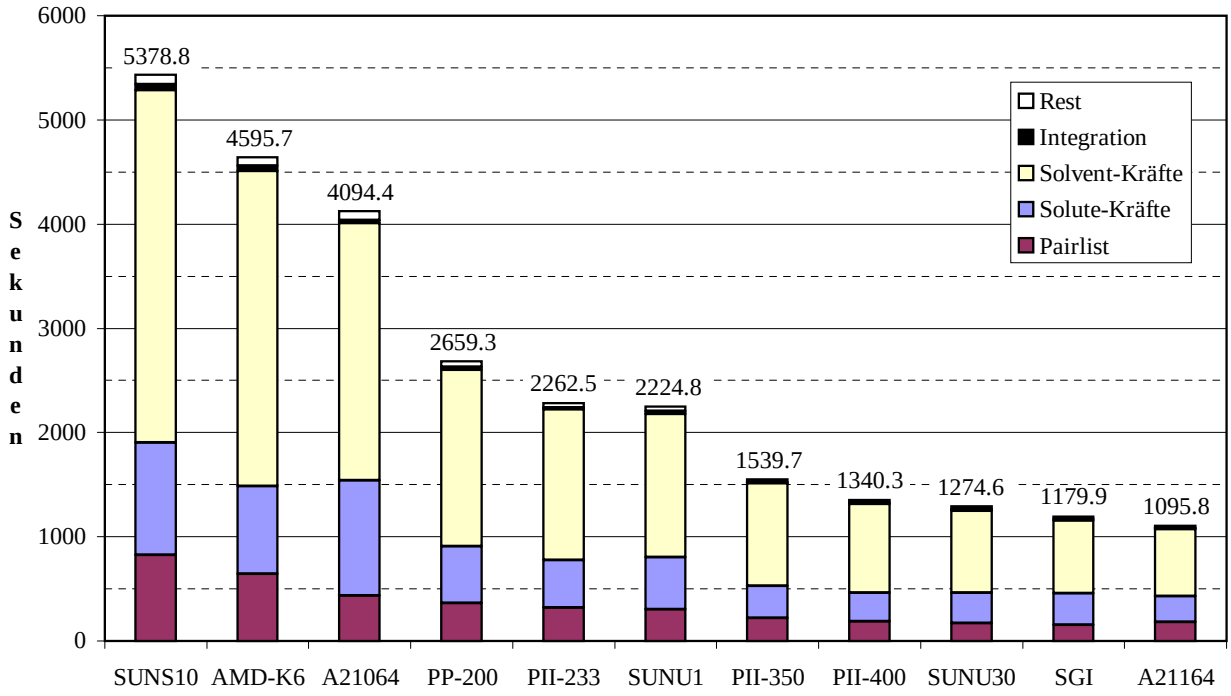


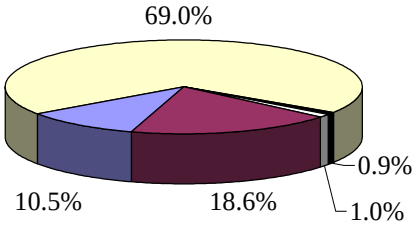
Abb. 3.2: Simulationszeiten der oktaederischen Thrombin Molekulartopologie

Die Rechner wurden nach der jeweiligen Simulationszeit geordnet aufgelistet. Man beachte, dass die kubische Simulation etwas mehr als doppelt so lange wie die oktaederische braucht.
In beiden Abbildungen ist ersichtlich, dass für die Berechnung der Solvent-Kräfte am meisten Zeit benötigt wird. Die Prozeduren für die Solute-Kräfte und die Pairlist benötigen beide etwa gleich lang, wobei zu bemerken ist, dass die Pairlist-Routine nur jeden 5. Iterationsschritt ausgeführt wird. Die Zeiten für die Integration und alle übrigen Programmteile fallen praktisch nicht ins Gewicht.
Interessant ist, dass die relative Zeit der einzelnen Prozduren im Vergleich zur Gesamtzeit der Simulation nicht auf allen Rechnern gleich ist. Betrachtet man Abb. 3.1, benötigt die SPARC10 für die Berechnung der Solute-Kräfte rund drei Mal soviel Zeit wie die AMD-K6, obwohl sie für die gesamte Simulation nur etwa anderthalb Mal langsamer ist.

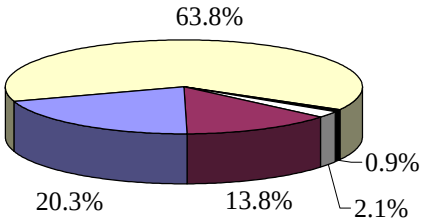
3.1.3 Berechnungszeiten der Prozeduren

Uns interessierte die prozentuale Aufteilung der Zeit auf die Prozeduren pro Rechner, welche in folgenden Grafiken dargestellt ist.

Pentium Pro 200

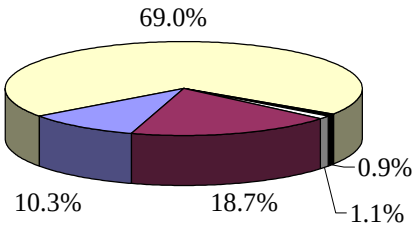


kubische Topologie

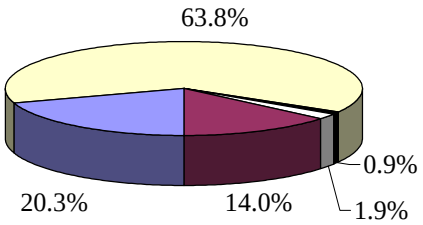


oktaederische Topologie

Pentium II/233

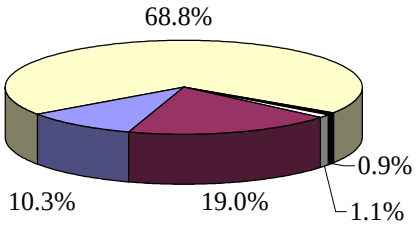


kubische Topologie

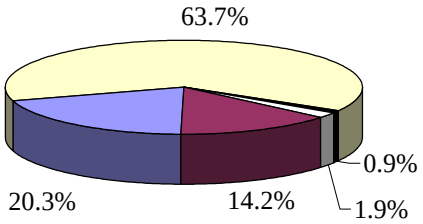


oktaederische Topologie

Pentium II/350

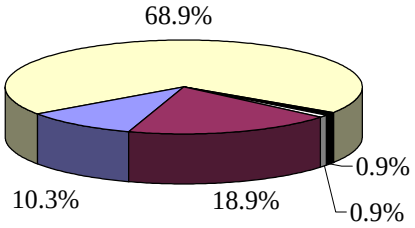


kubische Topologie

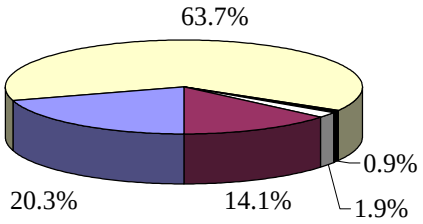


oktaederische Topologie

Pentium II/400

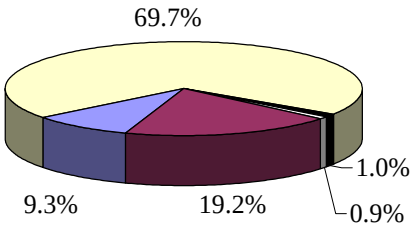


kubische Topologie

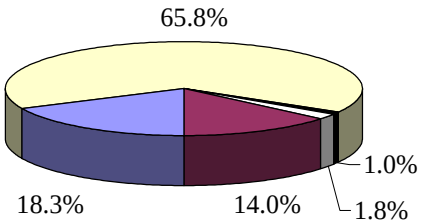


oktaederische Topologie

AMD K6/200

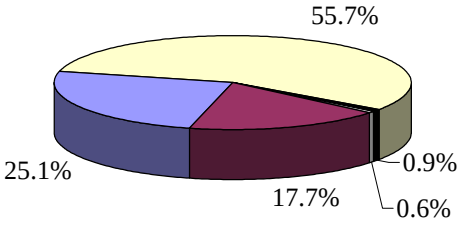


kubische Topologie

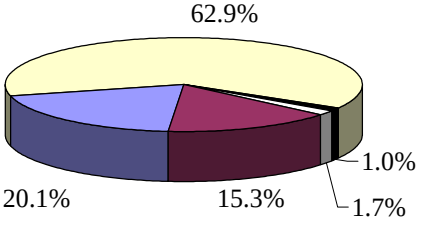


oktaederische Topologie

SUN SPARC Station 10/85



kubische Topologie



oktaederische Topologie

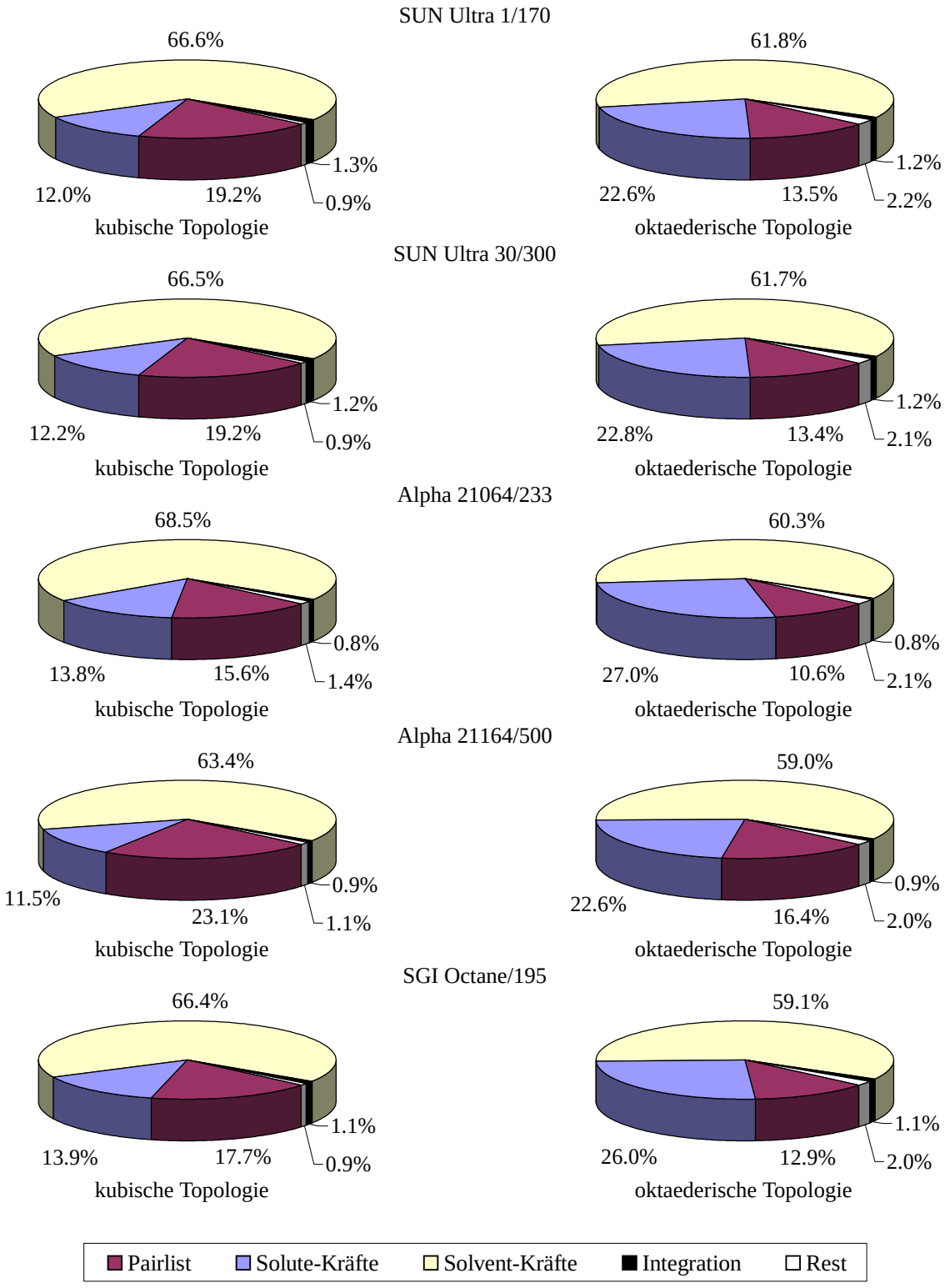


Abb. 3.3: Prozentuale Aufteilung der Zeit auf die einzelnen Prozeduren

Im Wesentlichen bestehen keine grösseren Unterschiede zwischen den Rechnern. Die PCs mit den Intel- und kompatiblen Prozessoren weisen alle dieselbe prozentuale Aufteilung auf. Ähnlich ist es bei den SUNs und der SGI. Sie benötigen für die Berechnung der Solute-Kräfte und der Pairlist ein bisschen mehr Zeitanteil als die PCs, dafür weniger für die Sovent-Kräfte. Nur die SPARC10 benötigt für die Solute-Kräfte der kubischen Topologie fast den doppelten Anteil, dafür wesentlich weniger für die Solvent-Kräfte. Die gesamte Simulation der kubischen Topologie dauert auf der SPARC10 auch 1.5 Mal länger als auf der AMD, wobei die oktaederische Topologie nur etwa 1.1 Mal langsamer ist (siehe Abb. 3.1 und Abb. 3.2).
Die Alpha21064 benötigt weniger Zeit für die Pairlist, dafür mehr für die Solvent-Kräfte. Umgekehrt ist es bei der Alpha21164, wo ein hoher Anteil für die Pairlist verwendet wird, im Gegensatz zur Berechnung der Solvent-Kräfte.
Für die Integration und die übrigen Teile der Simulation werden auf allen Rechnern nur 1-2% Rechenzeit benötigt. Der Anteil der Integration ist für die kubische und oktaederische Simulation gleich gross. Die restlichen Teile benötigen in der oktaederischen Simulation etwa 1% mehr Zeit, was auf die aufwendigere Initialisierung zurückzuschliessen ist.
Vergleicht man die Aufteilung der Prozeduren der beiden Simulationen, sind die Anteile der Pairlistberechnung und der Solvent-Kräfte in der oktaederischen kleiner. Der Grund liegt in der Grösse des Simulationsraumes. In der oktaederischen Simulation wird weniger Lösung verwendet, wobei die Grösse des Proteins in beiden Simulationen gleich bleibt. Somit müssen weniger Solvent-Kräfte berechnet werden und die Zeit dieser Routine fällt entsprechend tiefer aus. Der kleinere Lösungsanteil wirkt sich auch auf die Berechnung der Pairlist aus, wo insgesamt weniger Moleküle berücksichtigt werden müssen. Dadurch verschiebt sich der prozentuale Anteil dieser beiden Routinen auf die Routine für die Berechnung der Solute-Kräfte, welche auf Grund der gleich hohen Anzahl Proteinmoleküle und der aufwendigeren Topologie eher ein bisschen mehr Zeit benötigt.
In den folgenden Grafiken sind die Zeiten der einzelnen Routinen dargestellt. Die Rechner wurden dabei gleich geordnet aufgelistet wie in Abb. 3.1 und Abb. 3.2. Zum Vergleich der beiden Simulationen sind in jedem Diagramm pro Rechner jeweils die Zeiten der Routinen für beide Topologien dargestellt.

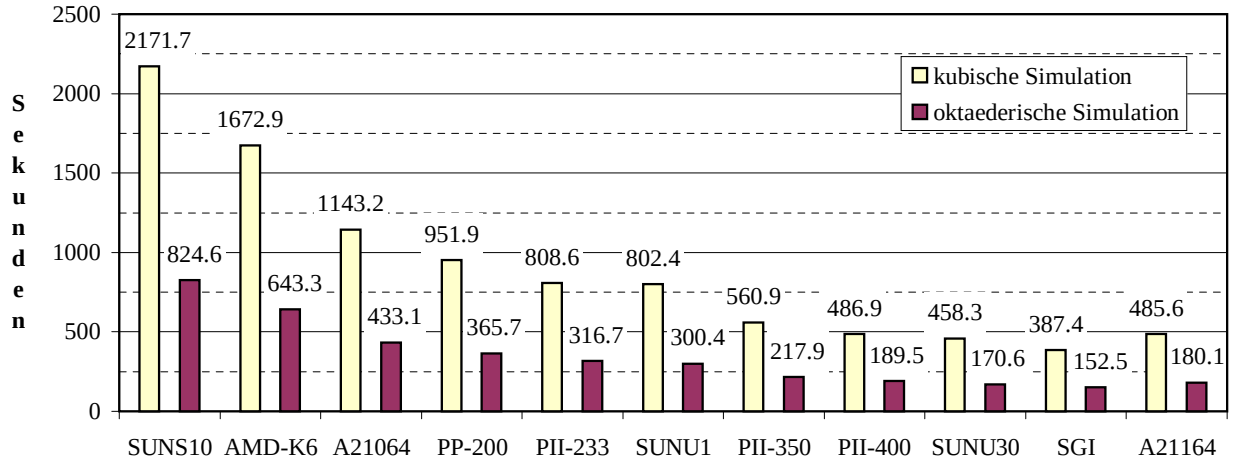


Abb. 3.4: Berechnungszeiten der Pairlist

In Abb. 3.4 sieht man den Einfluss des kleineren oktaederischen Simulationsraumes mit weniger Solvent-Molekülen deutlich. Da in der Pairlistberechnung jedes Molekül mit jedem verknüpft werden muss, hängt der Rechenaufwand quadratisch von der Anzahl Moleküle ab. In der oktaederischen Version werden halbsoviele Solvent-Moleküle verwendet wie in der kubischen, die Anzahl Solute-Moleküle bleibt jedoch gleich. Da die Menge der Solute-Moleküle im Vergleich zur Anzahl Solvent-Moleküle wesentlich kleiner ist, jedoch deren Verknüpfung komplizierter ist, sinkt die Rechenzeit der Pairlist in unserem Beispiel um etwa einen Drittel.
Die Pairlist-Routine ist relativ speicherintensiv, da auf die Daten aller Moleküle zugegriffen werden muss. Deshalb hängt die Berechnungsdauer der Pairlist mehr vom Hauptspeicher als vom Prozessor ab. Die SGI mit ihrem schnellen Speicher ist daher die schnellste Maschine und die Alpha21164 fällt hinter die SUN Ultra30 zurück.

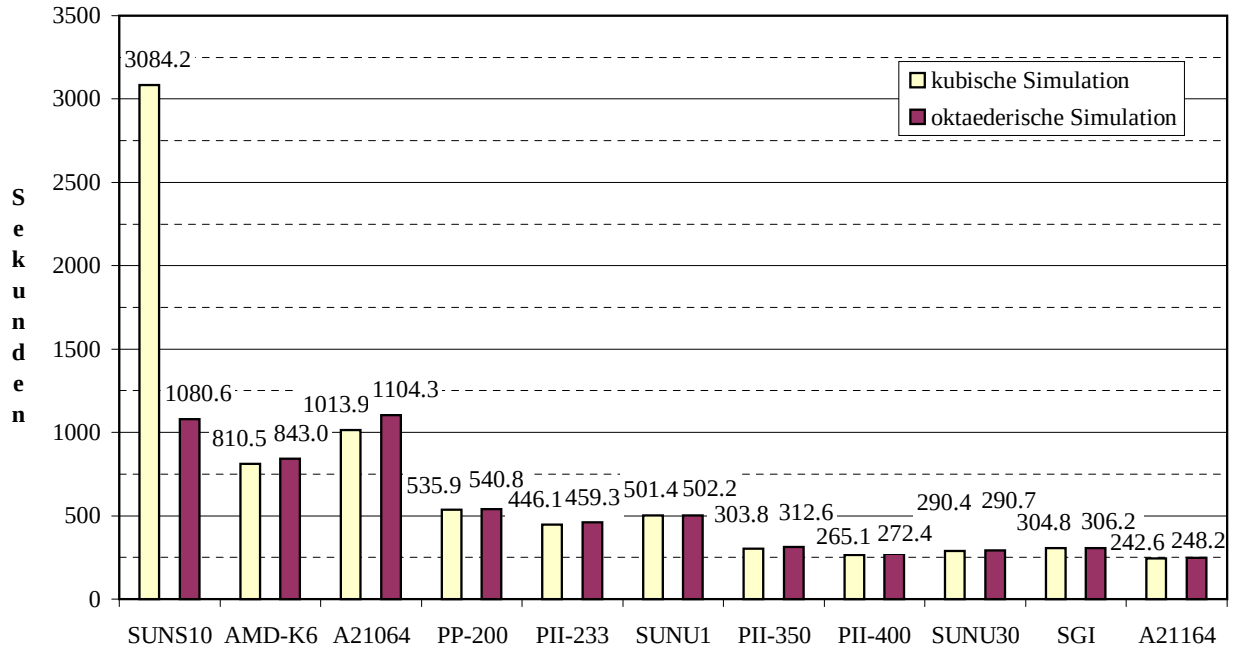


Abb. 3.5: Berechnungszeiten der Solute-Kräfte

Wie oben beschrieben ist die Anzahl der Solute-Atome in beiden Simulationstopologien gleich gross. Der Berechnungsaufwand ist in der komplexeren oktaederischen Topologie nicht wesentlich höher, weshalb die Simulationszeiten der Solute-Kräfte in beiden Topologien praktisch gleich hoch sind. Nur die SUN SPARC10 weicht hier stark von den anderen Rechnern ab.
Die Berechnung der Solute-Kräfte ist ziemlich komplex und weist entsprechend viele Rechenoperationen auf. Dafür sind relativ wenig Atome vorhanden, womit auf wenig Daten zugegriffen werden muss. Bei Rechnern mit grossem Cache passen alle Daten dort hinein und es muss nicht auf den Hauptspeicher zugegriffen werden. Im Gegensatz zur Pairlist-Routine ist die Zeitdauer der Solvent-Berechnung von der Leistung des Prozessors und der Grösse des Caches abhängig.
Im Vergleich zur Gesamtzeit der Simulation berechnen die PCs diese Routine eher schneller, die SUNs sind eher langsamer.

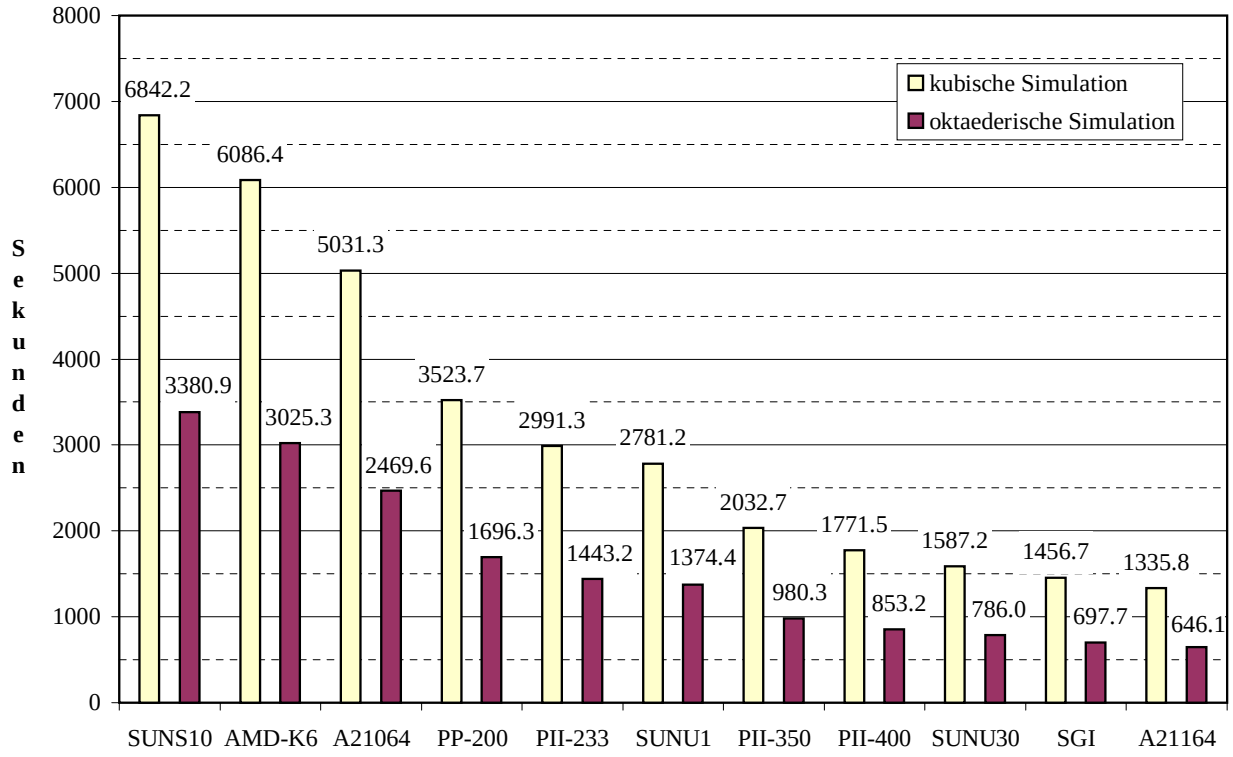


Abb. 3.6: Berechnungszeiten der Solvent-Kräfte

Der Einfluss des kleineren oktaederischen Simulationsraumes wirkt sich natürlich auch bei der Berechnung der Solvent-Kräfte aus. Da die Atome in dieser Routine gemäss Pairlist verknüpft werden, was bedeutet, dass nicht jedes mit jedem Atom verknüpft wird, hängt die Berechnungszeit linear von der Anzahl der verwendeten Atome ab. In der oktaederischen Topologie werden halbsoviele Solvent-Atome verwendet wie in der kubischen. Daher sinkt die Berechnungszeit in der oktaederischen um die Hälfte.
Diese Routine beansprucht die Prozessorleistung und den Hauptspeicher gleichermassen. Es müssen Distanzen, Kräfte und Energien berechnet werden, was entsprechend viele Rechenoperationen erfordert. Dabei muss auf die Daten vieler Atome zurückgergriffen werden, welche nicht alle im Cache Platz haben und deshalb vom Hauptspeicher geholt werden müssen.
Da diese Routine den grössten Anteil der GROMOS™-Simulation ausmacht, ist die Reihenfolge der Rechner nach Geschwindigkeit gleich wie die der gesamten Simulation.

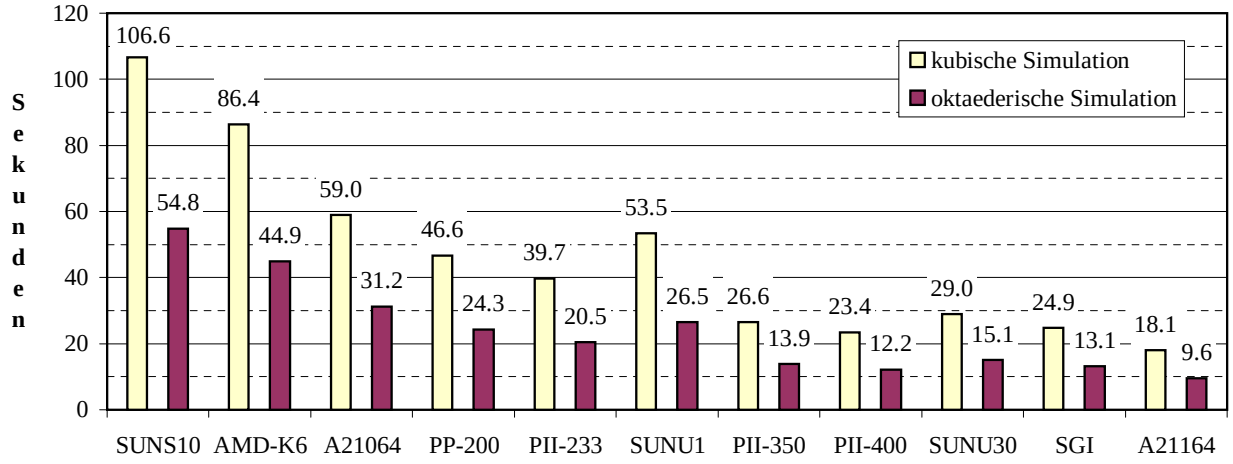


Abb. 3.7: Berechnungszeiten der Integration

Bei halbsovielen Solvent-Atomen werden auch halbsoviele Kräfte berechnet, die in der Integrations-Routine addiert werden. Daher beträgt auch hier die Berechnungszeit in der oktaederischen Simulation knapp die Hälfte im Vergleich zur kubischen.
Die SUNs und die SGI schneiden im Vergleich zu den anderen Rechnern eher schlechter ab.

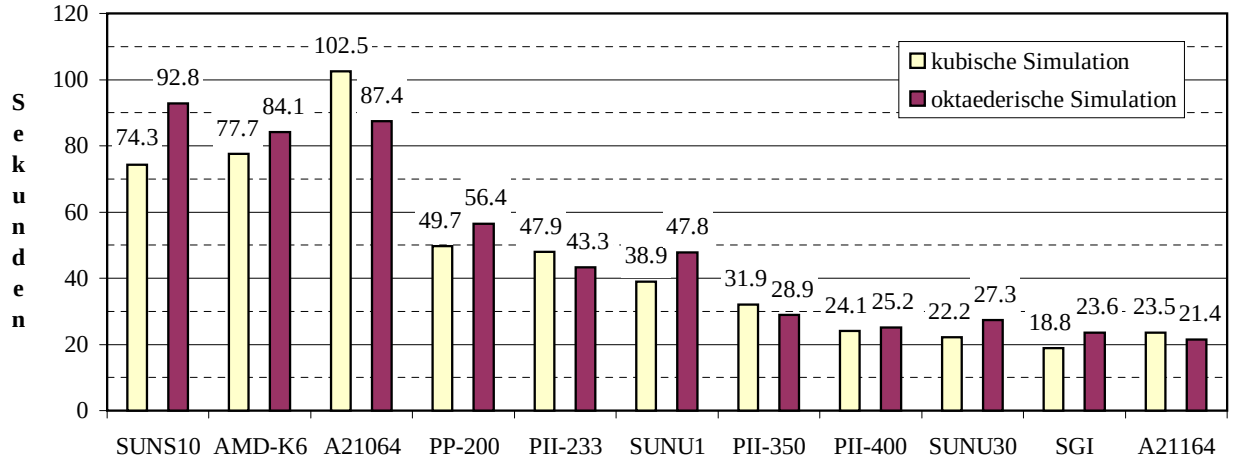


Abb. 3.8: Berechnungszeiten aller übrigen Programmteile

In Abb. 3.8 ist kein Schema mehr erkennbar. Hier spielen viele Faktoren eine Rolle, da diese Zeiten von unterschiedlichen Programmteilen stammen. Der Anteil dieser Routinen am gesamten Programm ist jedoch so klein, dass er für die Simulationszeit praktisch keine Rolle mehr spielt.

3.1.4 Messstreuungen

Die Simulationen wurden auf jedem Rechner zehnmal laufen gelassen (siehe Kapitel 3.1.1). Dabei gab es entsprechende Streuungen bei den Resultaten, welche in den Abb. 3.9 und Abb. 3.10 für die Messungen der Gesamtzeit der GROMOS^{TM}-Simulation dargestellt sind. Die Abweichungen sind relativ zum Median dargestellt, wobei die unterschiedliche Skalierung der beiden Grafi-

ken beachtet werden muss. Die Rechner sind wie in Abb. 3.1 und Abb. 3.2 nach der jeweiligen Simulationszeit geordnet aufgelistet.

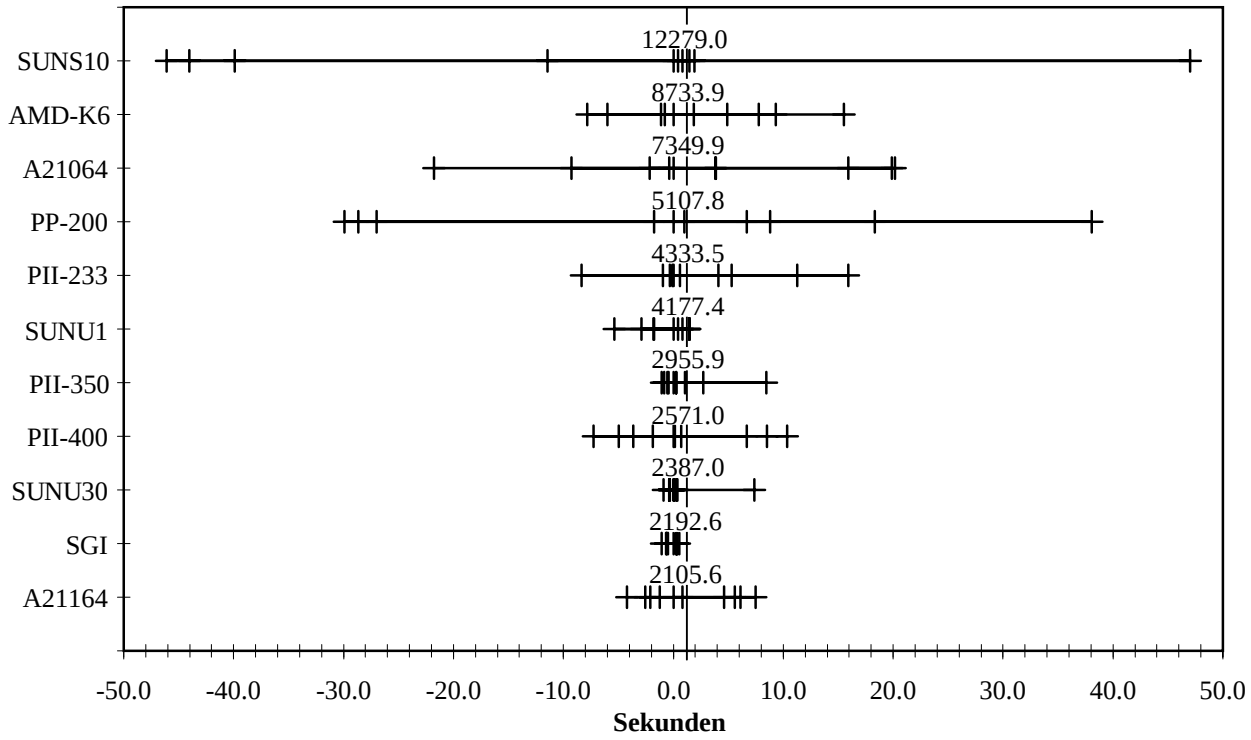


Abb. 3.9: Streuungen der Messresultate der kubischen GROMOS™-Simulation

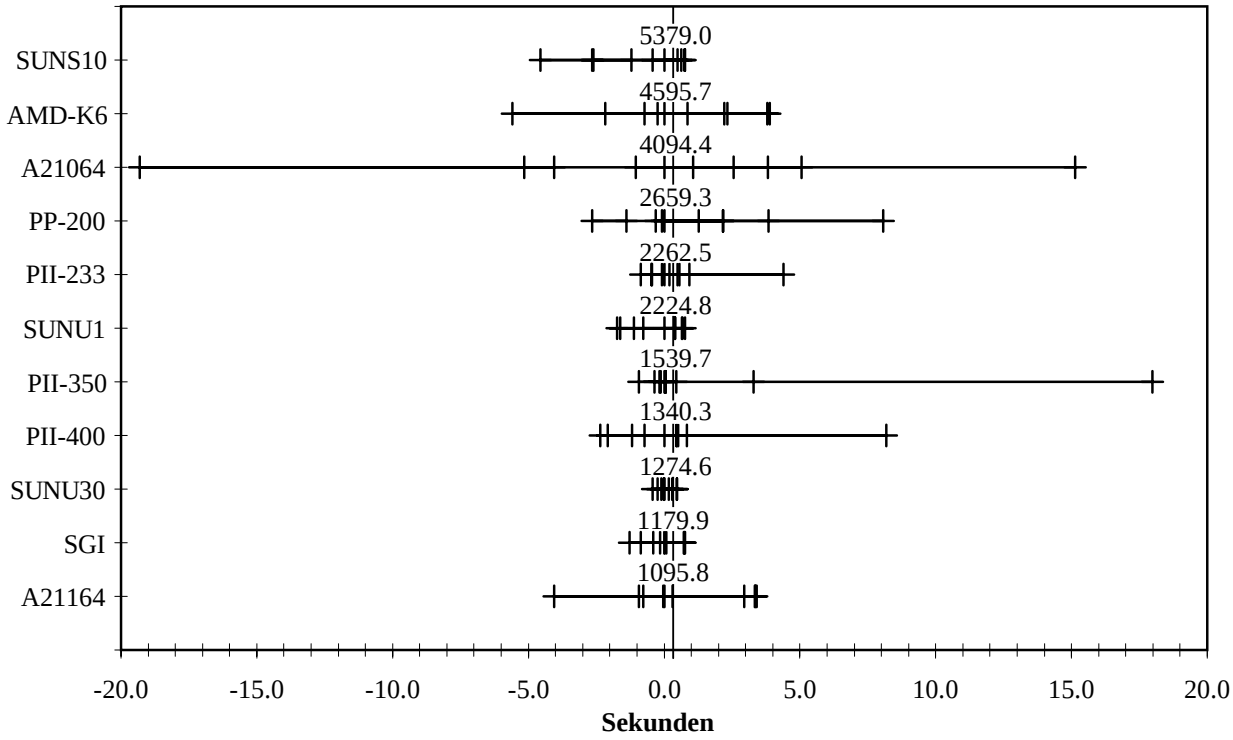


Abb. 3.10: Streuungen der Messresultate der oktaederischen GROMOS™-Simulation

Im Vergleich zur gesamten Simulation liegen die Abweichungen der einzelnen Messwerte für alle Maschinen unter einem Prozent. Die Verteilung ist jedoch unregelmässig. Während die Messwerte bei einzelnen Maschinen gleichmässig über den gesamten Streubereich verteilt sind, weisen andere nur vereinzelt Messwerte mit starker Abweichung auf.
Ein numerisches Mass der Streuung von Messwerten ist die Standardabweichung. In folgender Tabelle sind die Standardabweichungen für beide Simulationen und alle Rechner angegeben:

Workstation	kubische Simulation	oktaederische Simulation
SUNS10	28.3	1.8
AMD-K6	7.1	2.9
A21064	13.2	8.8
PP-200	21.9	3.0
PII-233	6.8	1.5
SUNU1	2.3	1.0
PII-350	2.8	5.7
PII-400	5.9	3.0
SUNU30	2.4	0.3
SGI	0.6	0.7
A21164	4.2	2.5

Tabelle 3.2: Standardabweichungen der Messwerte der GROMOS™-Simulation

Wie die Streuungen sind auch die Standardabweichungen in Tabelle 3.2 im Bezug zur gesamten Simulationszeit klein. Das relative Streuungsmass liegt im Bereich von einigen Promillen.
Die Streuung der Messresultate kann für längere Simulationen grösser werden, da dabei mehr unvorhergesehene Ereignisse wie Kontextwechsel, Garbage Collections etc. auftreten können.
Vergleicht man nun die Zeitdauer der beiden Simulationen mit den Streuungen der Messresultate ist diese Tendenz sichtbar. Während die oktaederische Simulation im Vergleich zur kubischen rund halb so lange dauert (siehe Kapitel 3.1.2), sind die Abweichungen der Messwerte der oktaederischen Simulation (Abb. 3.10) ebenfalls nur etwa halb so gross wie die der kubischen (Abb. 3.9). Auch die Werte der Standardabweichungen in Tabelle 3.2 fallen für die oktaederische Simulation geringer aus.
Zwischen den einzelnen Maschinen ist diese Tendenz jedoch kaum feststellbar, da die Häufigkeit der unvorhergesehenen Ereignisse vom Maschinentyp und der Version des verwendeten Betriebssystems abhängt.
Für die Auswertung der Messresultate verwendeten wir den Median, da dieser weitgehend unabhängig von extrem abweichenden Werten ist und zudem ein reales Resultat darstellt. Trotzdem ist auch der Mittelwert wichtig, da er eine genauere Aussage über die durchschnittliche Zeitdauer der Simulation erlaubt.
Uns interessierte daher die Abweichung zwischen Mittelwert und Median für die beiden Simulationen auf allen Rechnern:

Workstation	kubische Simulation	oktaederische Simulation
SUNS10	9.0	0.9
AMD-K6	2.4	0.4
A21064	3.0	0.2
PP-200	1.4	1.3
PII-233	2.7	0.5
SUNU1	0.6	0.2
PII-350	1.0	2.0
PII-400	0.9	0.4
SUNU30	0.6	0.1
SGI	0.1	0.0
A21164	1.5	0.8

Tabelle 3.3: Abweichung zwischen Mittelwert und Median der GROMOS™-Messresultate

Tabelle 3.3 zeigt, dass Mittelwert und Median praktisch die gleichen Werte ergeben. Im Vergleich zur gesamten Simulationszeit sind die Abweichungen geringer als ein Promill.
Zur Elimination der Messresultate mit den grössten Abweichungen lassen sich sogenannte 10-90 Percentiles der Messreihe bilden. Damit würden in unserer Simulation die beiden äussersten Werte abgeschnitten. Betrachtet man die Grafiken in Abb. 3.9 und Abb. 3.10, könnte bei der SPARC10, der Alpha21064 und dem Pentium Pro 200 eine leichte Verbesserung der Abweichung zwischen Mittelwert und Median erzielt werden. Eine andere Methode wäre die Streichung der zwei Werte mit der grössten absoluten Abweichung zum Mittelwert, was zusätzlich zu den vorhin erwähnten Maschinen noch beim Pentium II/350 und Pentium II/400 eine Verbesserung bewirken würde. Die durch diese Methoden erzielbaren Verbesserungen wäre aber minim und dürften sich angesichts der bereits geringen Abweichungen kaum lohnen.

3.2 SPEC CPU95

3.2.1 Messmethode

Zum Ermitteln der SPEC-Resultate wurde die SPEC CPU95 Suite auf jedem Rechner mit dem jeweiligen Compiler compiliert, wobei dieselben Optimierungsflags wie für GROMOS™-verwendet wurden (Kapitel 2.3). Die pro Rechner verwendeten Optimierungsflags sind im Anhang angegeben. Die SPEC CPU95 Suite wurde jeweils fünf Mal laufen gelassen, wobei das Programm selber die Zeitdauer der Applikationen misst, die Mittelwerte bildet und die Resultate ausgibt (siehe Kapitel 2.2).

3.2.2 Resultate der SPEC CPU95

In den folgenden Diagrammen sind die ermittelten SPEC-Base95-Ratios für die untersuchten Workstations dargestellt.

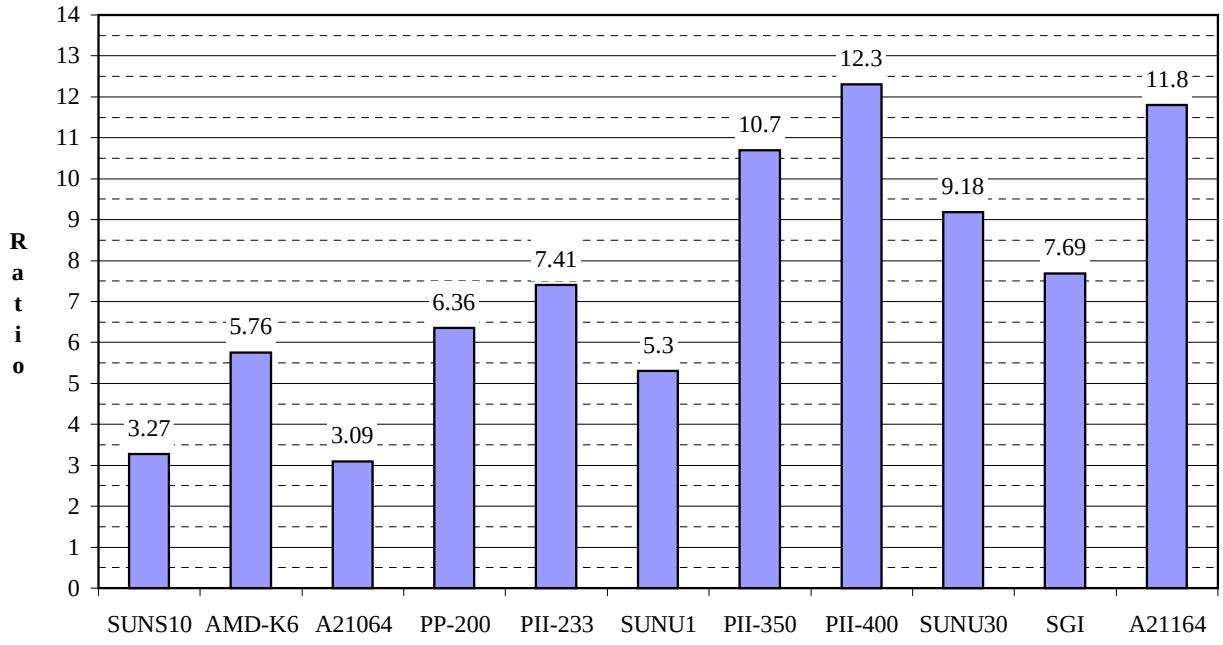


Abb. 3.11: SPECint-Base95-Ratios der verschiedenen Rechner

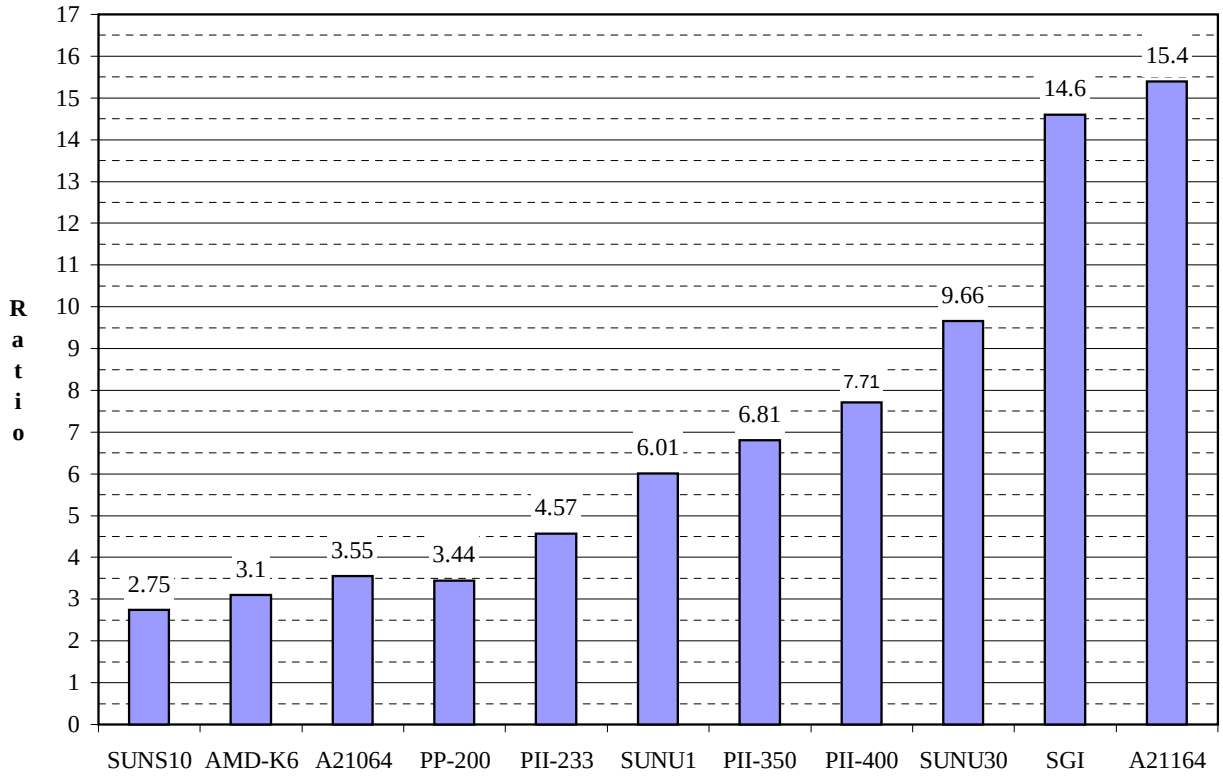


Abb. 3.12: SPECfp-Base95 Ratios der verschiedenen Rechner

Um einen Vergleich mit GROMOS™ machen zu können, sind die Rechner in Abb. 3.11 und Abb. 3.12 in der selben Reihenfolge wie in Abb. 3.1 und Abb. 3.2 aufgelistet. Es war zu erwarten, dass GROMOS™ nicht mit SPEC CINT95 korreliert, da in der GROMOS™-Simulation praktisch ausschliesslich Floating-Point Operationen verwendet werden. Hingegen zeigt die SPEC CFP95 bezüglich Geschwindigkeit auf den einzelnen Rechnern eine ähnliche Tendenz wie GROMOS™.
Die SPEC95-Ratios sind jedoch relative Werte bezüglich einer SUN SPARC Station10/40 als Referenzmaschine (siehe Kapitel 2.2), während die Messungen der GROMOS™-Simulation absolute Zeiteinheiten ergeben. Um einen genaueren Bezug zwischen SPEC CPU95 und GROMOS™ machen zu können, ist es besser, die gemessenen Zeiten der CPU95-Applikationen zu verwenden. Diese Zeiten stellen wie in GROMOS™ absolute Werte dar und werden beim Benchmark zusammen mit den SPEC-Ratios ausgegeben.
Von den Zeiten der einzelnen CPU95-Applikationen haben wir das geometrische Mittel genommen, da das SPECint95 bzw. SPECfp95-Ratio ebenfalls als geometrisches Mittel der einzelnen SPEC-Ratios gebildet wird. In den folgenden Grafiken sind die SPEC CPU95-Zeiten für alle Rechner dargestellt.

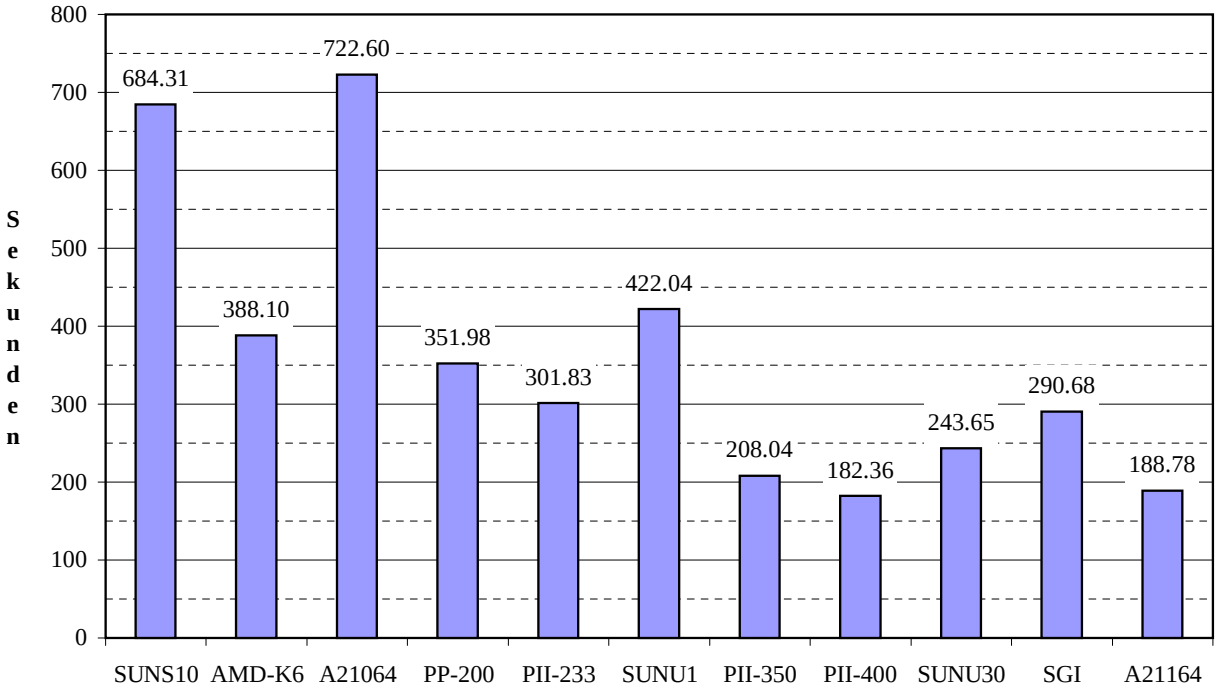


Abb. 3.13: SPEC CINT95 Zeiten der verschiedenen Rechner

Analog zu Abb. 3.11 ist auch in Abb. 3.13 ersichtlich, dass GROMOS™ nicht mit SPEC CINT95 korreliert. Die PCs sind im Vergleich zu den anderen Rechnern im Integerbereich schneller. Die Alpha21064 schneidet sehr schlecht ab und ist sogar langsamer als die SUN SPARC10. Interessanterweise ist auch die SGI im Integerbereich relativ langsam und muss sich vom Pentium II/400, der SUN Ultra30 und der Alpha21164 geschlagen geben, obwohl die SGI eigentlich für Bildverarbeitung konzipiert ist, wo hauptsächlich im Integerbereich gerechnet wird.
Die schnellste Maschine ist der Pentium II/400 dicht gefolgt von der Alpha21164, welche im Floating-Point-Bereich eindeutig am besten abschneidet.

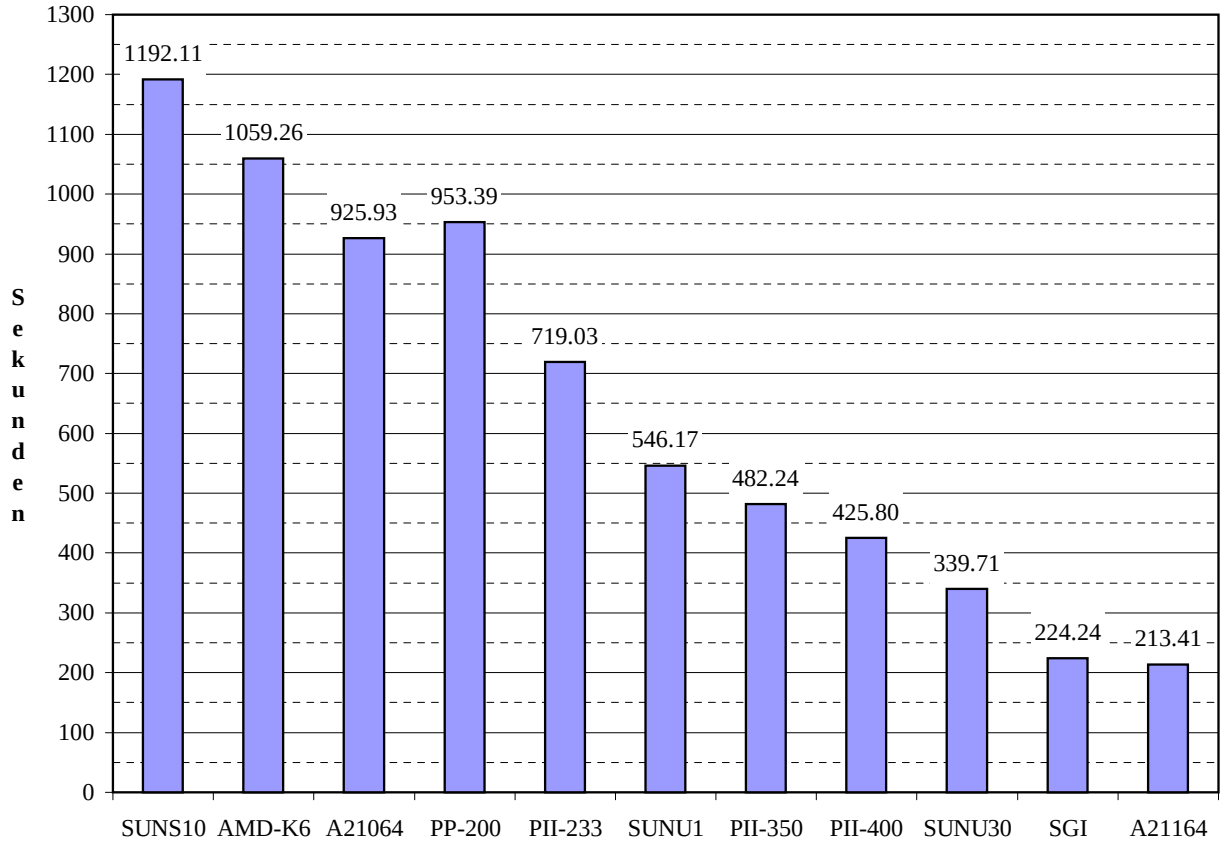


Abb. 3.14: SPEC CFP95 Zeiten der verschiedenen Rechner

Bei der SPEC CFP95 lässt sich eine ähnliche Tendenz wie für GROMOS™ erkennen. Ordnet man die Maschinen nach Geschwindigkeit des CFP95-Benchmarks, ergibt sich die gleiche Rangreihenfolge wie bei GROMOS™. Nur der Pentium Pro 200 tanzt hier aus der Reihe. Er ist geringfügig langsamer als die Alpha21064. In GROMOS™ is der Pentium Pro 200 fast so schnell wie der Pentium II/233.
Vergleicht man das Diagramm in Abb. 3.14 mit denen in Abb. 3.1 und Abb. 3.2, fällt die Zeit der CFP95 bezogen auf die Rechner praktisch linear ab. Bei GROMOS™ ist der Abfall der Simulationszeit zwischen SUN SPARC10 und Pentium Pro 200 grösser als zwischen letzterem und Alpha21164. Konkret verhalten sich die fünf schnellsten Rechner bezüglich Rechenleistung für GROMOS™ ähnlicher als für SPEC CFP95.

3.2.3 Resultate der einzelnen SPEC CPU95-Applikationen

Die SPEC CPU95-Resultate sind Mittelwerte von verschiedenen Applikationen (siehe Kapitel 2.2). Zwischen den Applikationen und Rechnern bestehen Unterschiede in der Laufzeit, da die Rechnerkomponenten in jeder Applikation unterschiedlich ausgelastet werden. In den folgenden Diagrammen sind die Resultate für alle SPEC CPU95-Applikationen dargestellt.

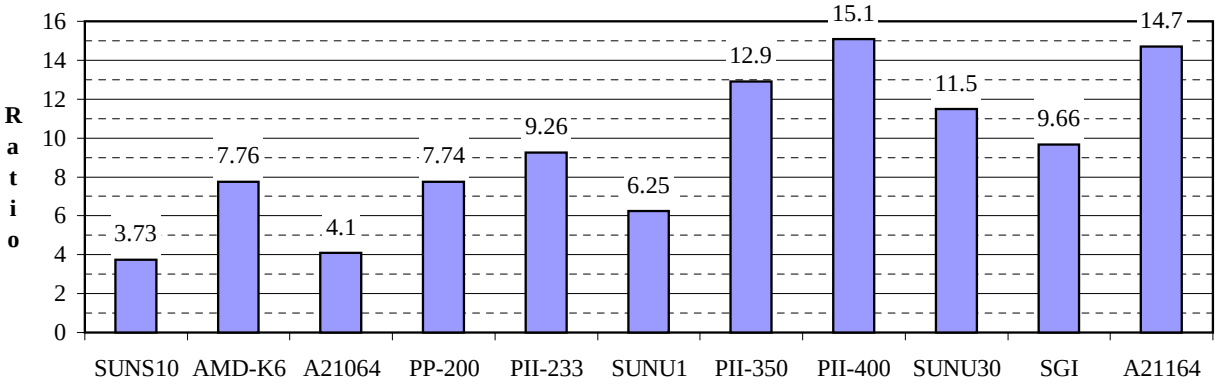


Abb. 3.15: SPEC-Ratios von 099.go

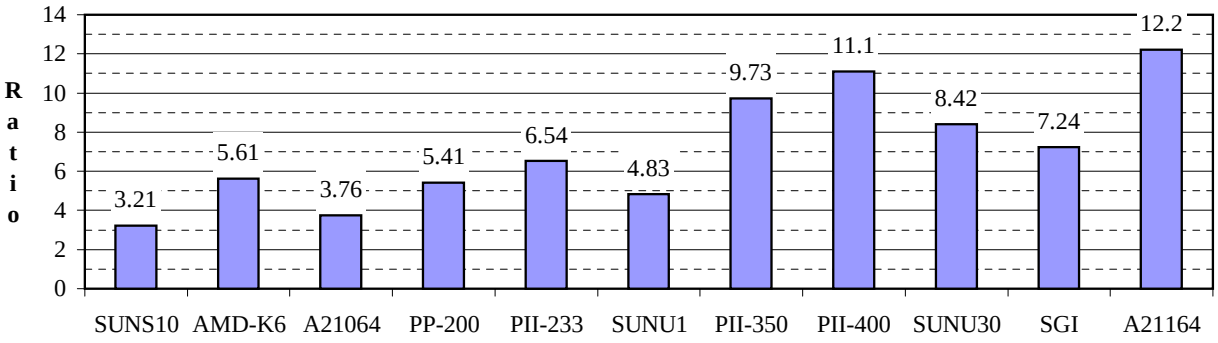


Abb. 3.16: SPEC-Ratios von 124.m88ksim

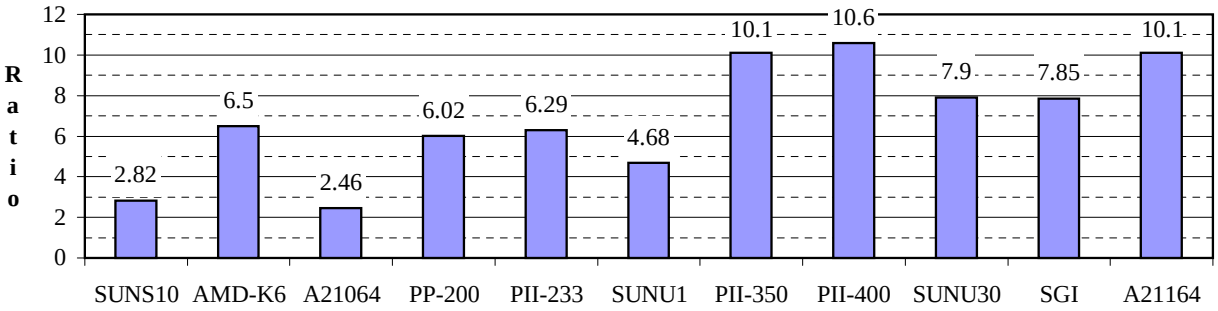


Abb. 3.17: SPEC-Ratios von 126.gcc

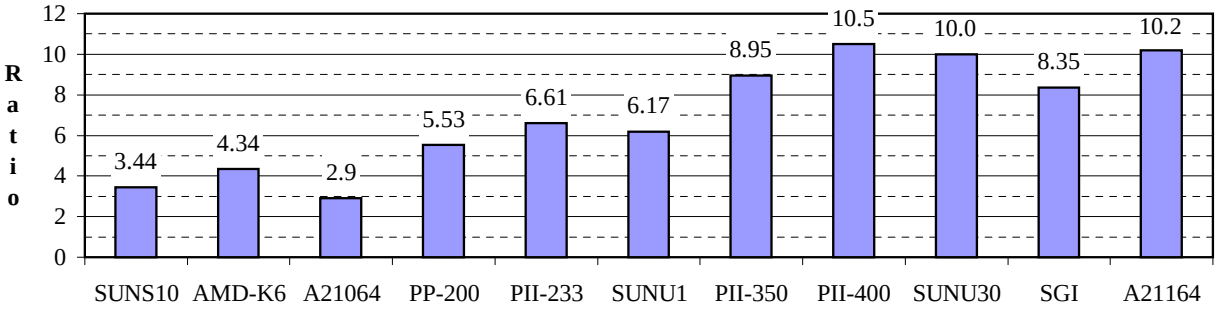


Abb. 3.18: SPEC-Ratios von 129.compress

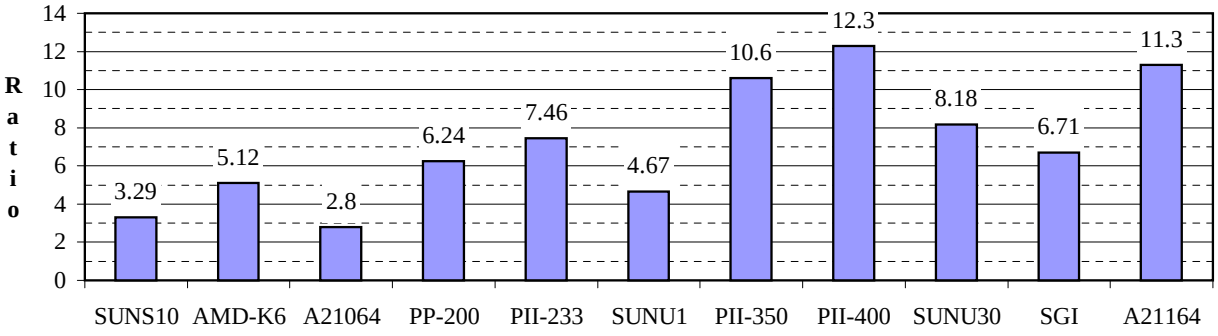


Abb. 3.19: SPEC-Ratios von 130.li

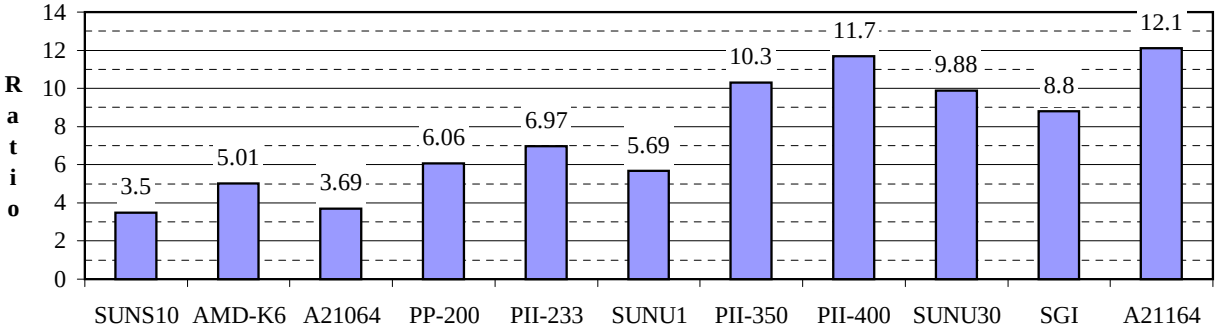


Abb. 3.20: SPEC-Ratios von 132.ijpeg

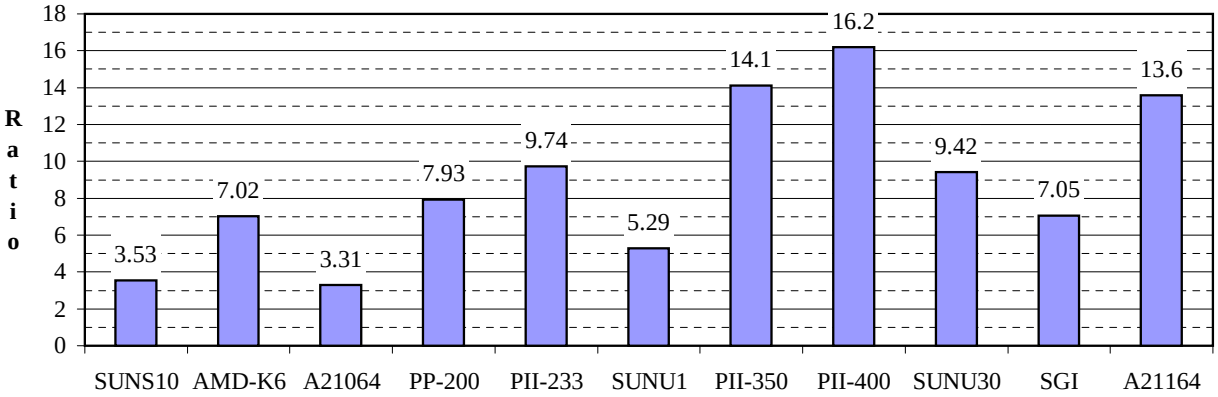


Abb. 3.21: SPEC-Ratios von 134.perl

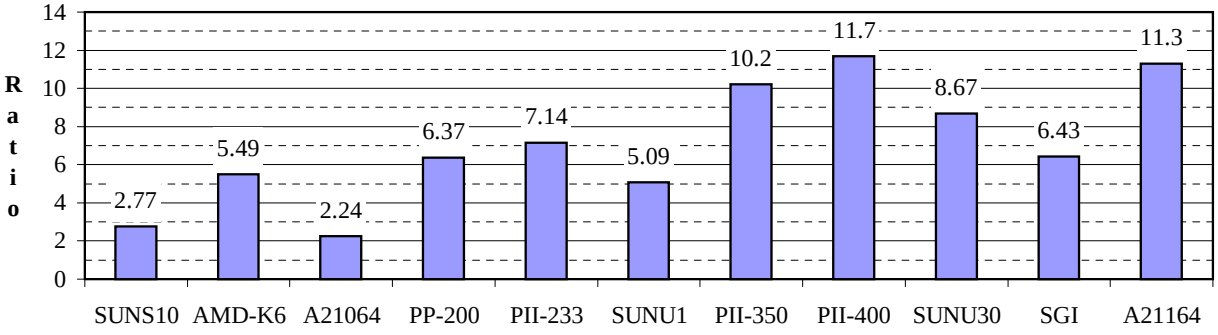


Abb. 3.22: SPEC-Ratios von 147.vortex

Von den SPEC CINT95-Applikationen korreliert keine mit GROMOS™. Die Rechner verhalten sich für die einzelnen CINT95-Applikationen praktisch gleich wie beim Gesamtresultat der SPEC CINT95.
Trotzdem gibt es kleinere Unterschiede. Der Pentium II/400 ist in sechs der acht Applikationen die schnellste Maschine. In 124.m88ksim (Abb. 3.16) und 132.ijpeg (Abb. 3.20) liegt jedoch die Alpha21164 an der Spitze. Die PCs können in 126.gcc (Abb. 3.17) und 134.perl (Abb. 3.21) ihre Stärken ausspielen. In diesen beiden Applikationen haben die beiden Pentium II Rechner mit 350MHz und 400MHz die höchste Performance. Die SUNs sind für Integer-Applikationen nicht besonders geeignet. Einzig in 129.compress (Abb. 3.18) ergeben sich für diese Maschinen höhere Werte. Auch die SGI ist nicht sehr schnell, liegt sie bei allen CINT95-Applikationen an fünfter bis sechster Stelle. Die Alpha21064ist ebenfalls für Integer ungeeignet, liegt sie zusammen mit der SUN SPARC10 überall an letzter oder zweitletzter Stelle.

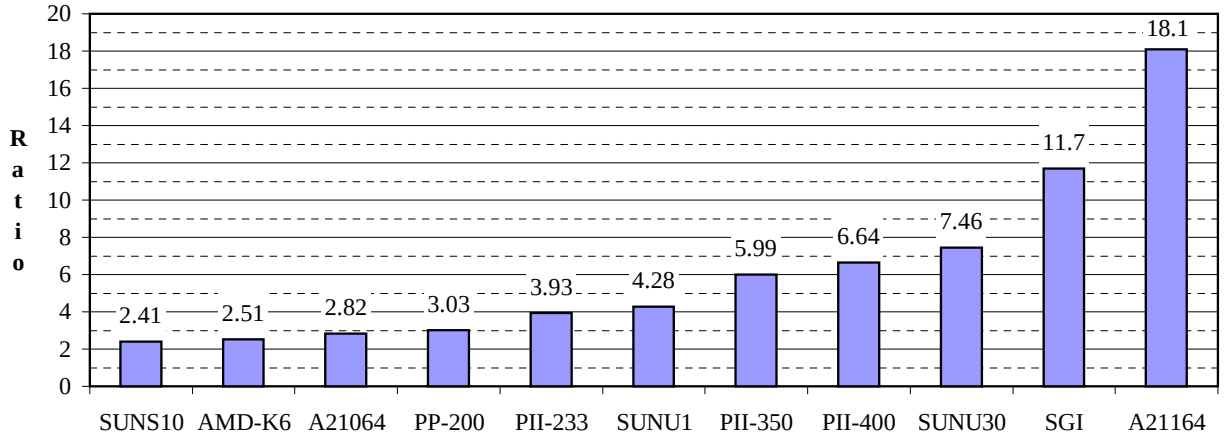


Abb. 3.23: SPEC-Ratios von 125.turb3d

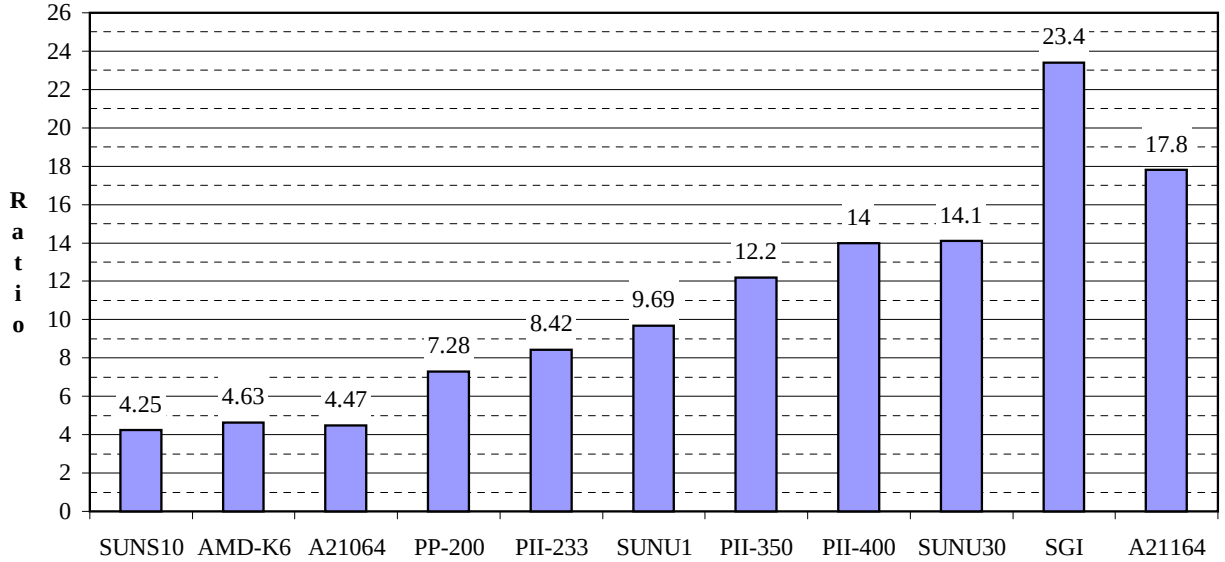


Abb. 3.24: SPEC-Ratios von 101.tomcatv

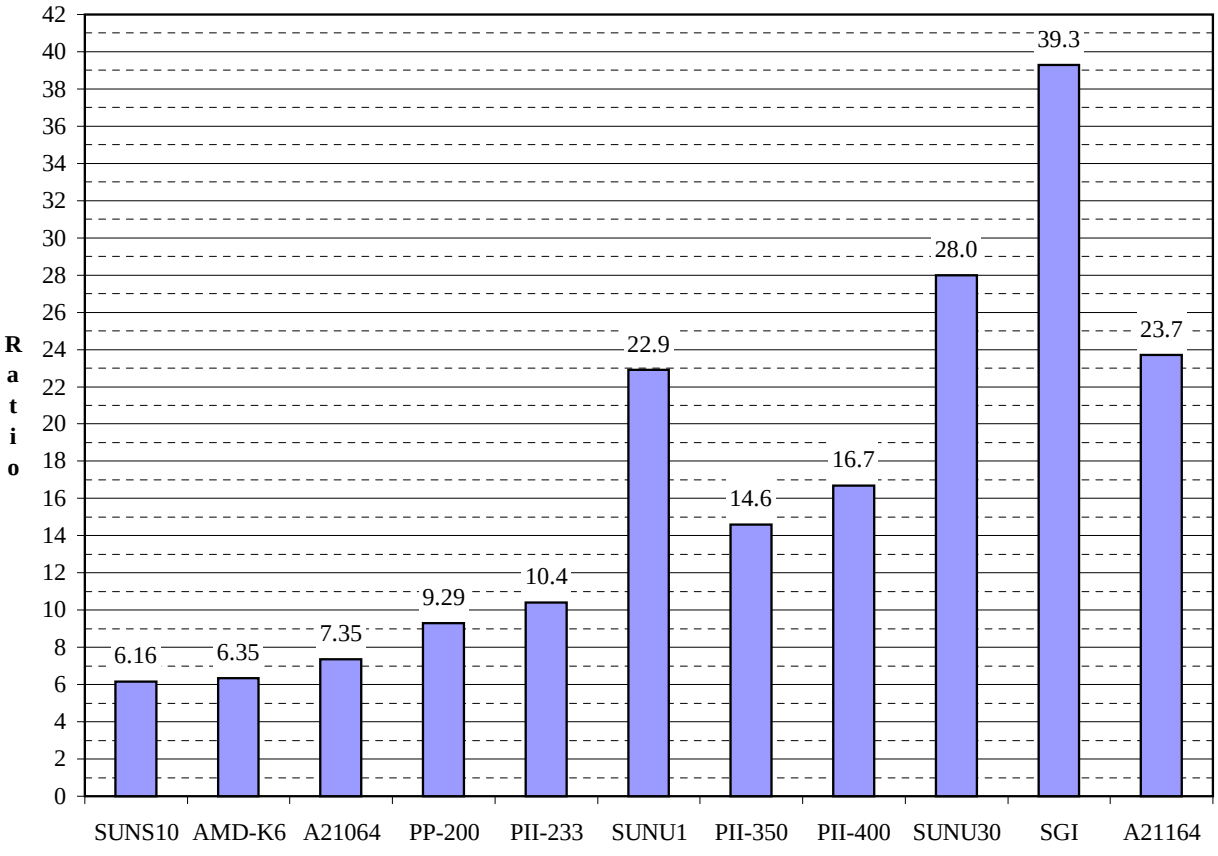


Abb. 3.25: SPEC-Ratios von 102.swim

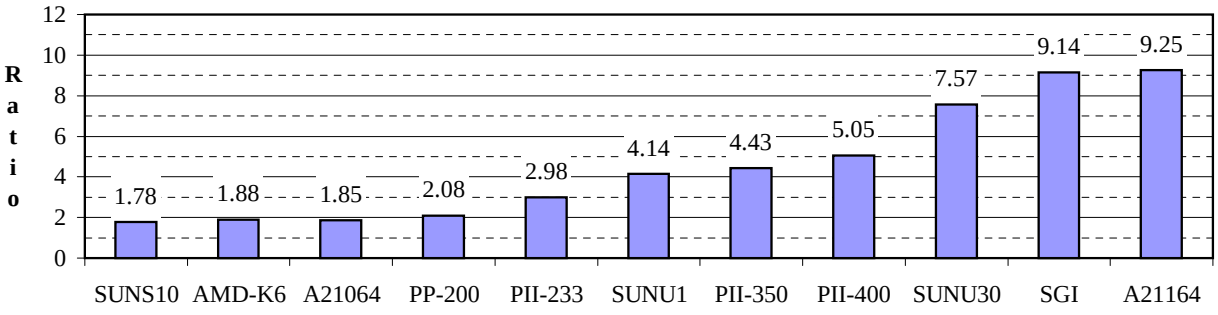


Abb. 3.26: SPEC-Ratios von 103.su2cor

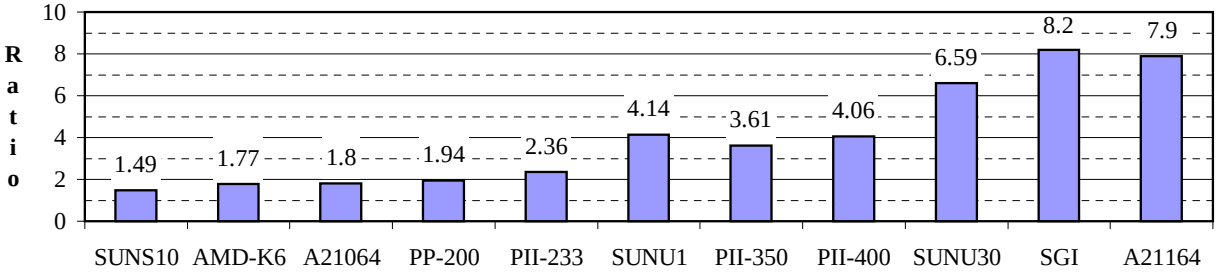


Abb. 3.27: SPEC-Ratios von 104.hydro2d

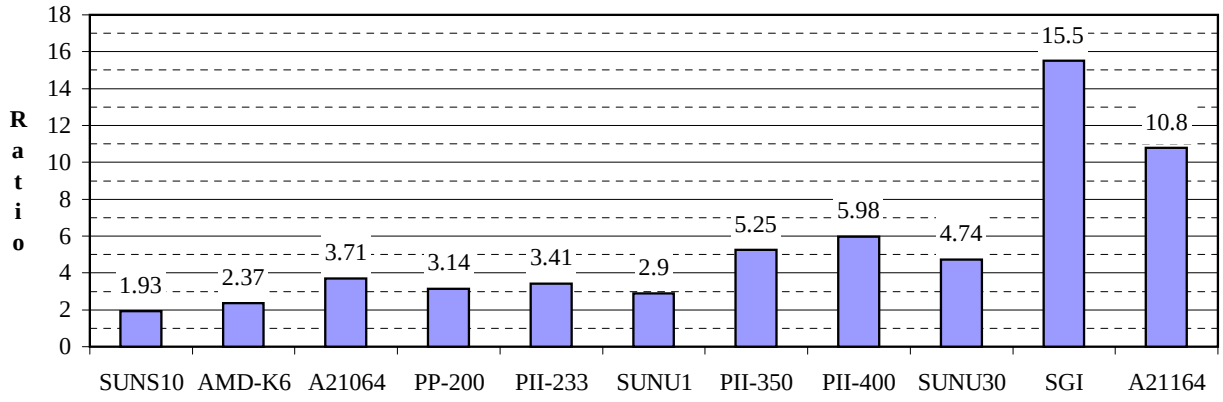


Abb. 3.28: SPEC-Ratios von 107.mgrid

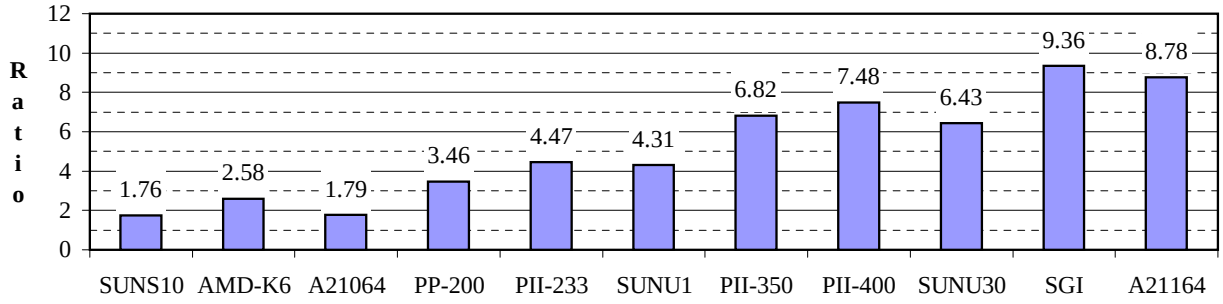


Abb. 3.29: SPEC-Ratios von 110.applu

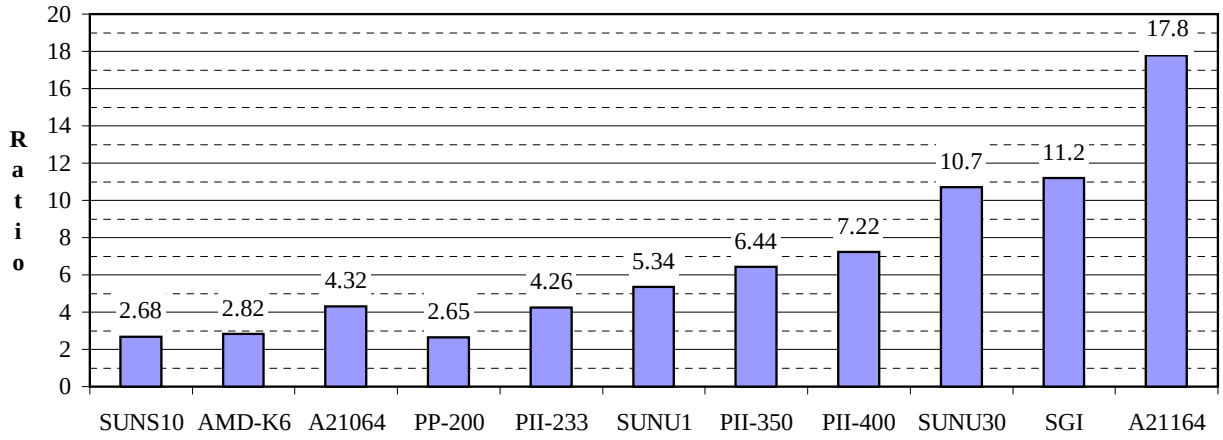


Abb. 3.30: SPEC-Ratios von 141.apsi

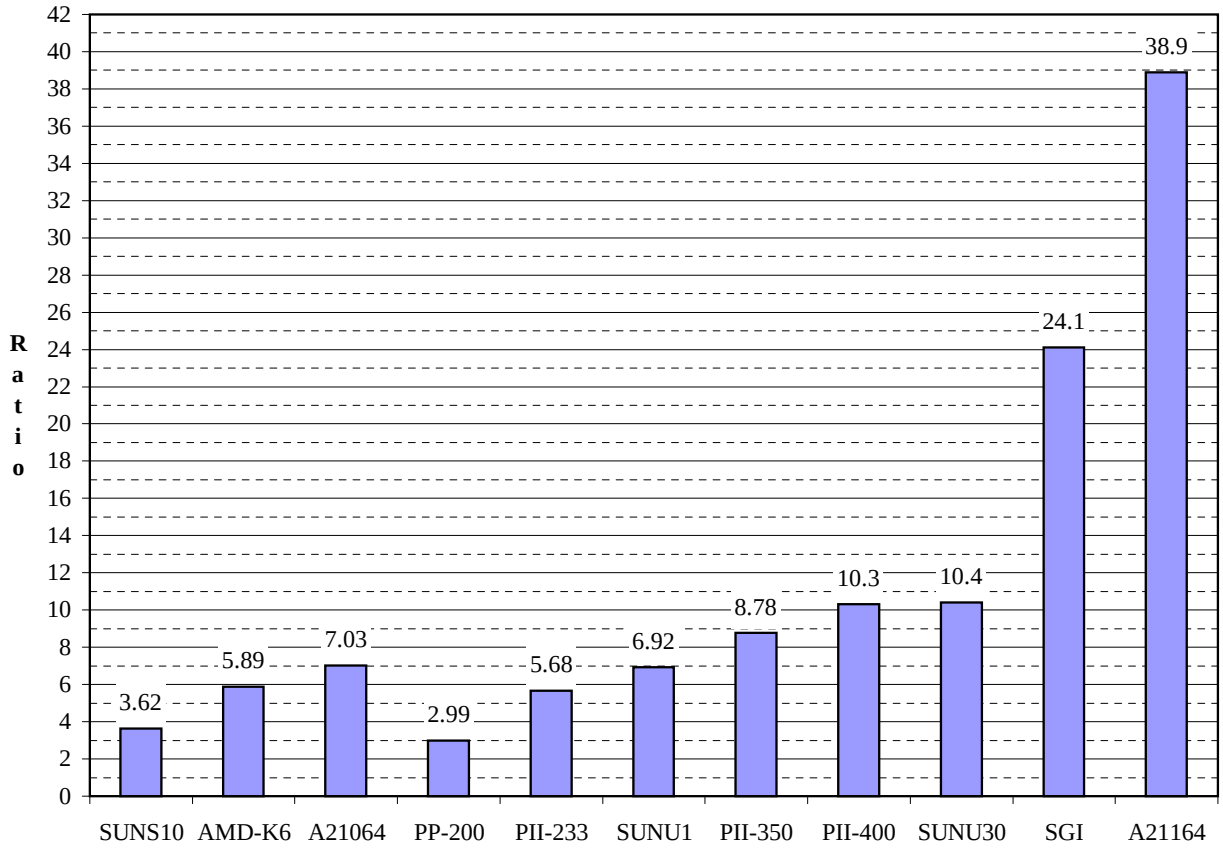


Abb. 3.31: SPEC-Ratios von 145.fpppp

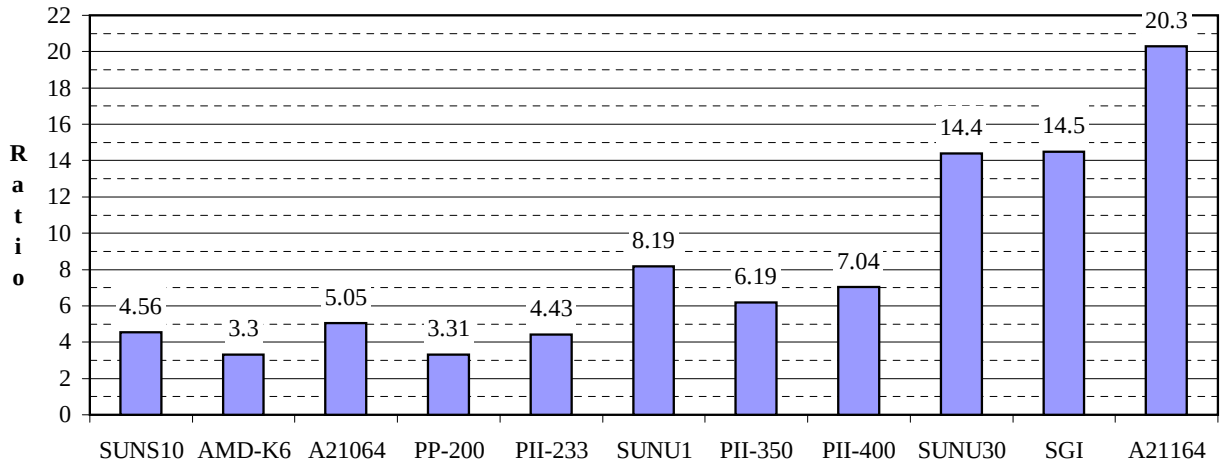


Abb. 3.32: SPEC-Ratios von 146.wave5

Die SPEC CFP95-Applikationen verhalten sich zum Teil sehr unterschiedlich im Vergleich zum Gesamtresultat. In allen Grafiken ist eine steigende Tendenz von der SUN SPARC10 zur Alpha21164 erkennbar, wobei die Rangreihenfolge nach Geschwindigkeit bezogen auf GROMOS™ nur in 125.turb3d (Abb. 3.23) gleich ist. In allen anderen CFP95-Applikationen weichen eine oder mehrere Maschinen davon ab.

In der Hälfte der Applikationen kann die SGI ihre Stärke ausspielen und ist von allen Maschinen an der Spitze. Dies ist in 101.tomcatv (Abb. 3.24), 102.swim (Abb. 3.25), 104.hydro2d (Abb. 3.27), 107.mygrid (Abb. 3.28) und 110.applu (Abb. 3.29) der Fall. In allen anderen liegt die Alpha21164 klar vorne. Interessant ist das Verhalten der SGI im Zusammenhang mit den SUNs. Diese Maschinen weisen im Vergleich zu den anderen Rechnern in 102.swim und 104.hydro2d bessere Werte auf, hingegen liegen sie in 107.mygrid und 110.applu unter dem Schnitt. Ausser in 101.tomcatv weichen die SUNs in den gleichen Applikationen wie die SGI vom Durchschnitt ab.
Die PCs weisen in 110.applu (Abb. 3.29) bessere Werte auf. In 146.wave5 (Abb. 3.32) fällt ihre Geschwindigkeit im Vergleich zu den übrigen Maschinen ab. Ein interessanter Effekt tritt in 141.apsi (Abb. 3.30) und 145.fpppp (Abb. 3.31) auf. In diesen Applikationen bricht die Geschwindigkeit des Pentium Pro ein, sodass die AMD schneller ist, in 145.fpppp ist die AMD sogar schneller als der Pentium II/233.
Die Alpha21064 verhält sich anders als ihre Schwester, die Alpha21164. In 107.mygrid (Abb. 3.28), 141.apsi (Abb. 3.30) und 145.fpppp (Abb. 3.31) weist sie im Vergleich zu den anderen Rechnern bessere Werte auf. Die Alpha21164 liegt jedoch in 107.mygrid hinter der SGI, ist also eher etwas langsamer als der Durchschnitt. In den anderen beiden Applikationen liegt sie an der Spitze. Das bedeutet, dass die beiden Alphas ihre Stärken nicht in denselben Applikationen ausspielen können.
Vergleicht man die CFP95-Applikationen untereinander, sind gewisse Parallelen in 103.su2cor (Abb. 3.26) und 125.turb3d (Abb. 3.23) sowie 141.apsi (Abb. 3.30) und 145.fpppp (Abb. 3.31) feststellbar. In den ersteren ist ein kontinuierlicher Anstieg von der SUN SPARC10 zur Alpha21164, mit Ausnahme der Alpha21064 in 103.su2cor, erkennbar. In den beiden letzteren weisen SPARC10, AMD und Alpha21064 bessere Werte auf als der Pentium Pro. An dieser Stelle fällt der Anstieg zurück und wächst dann kontinuierlich an bis zur Alpha21164.
Im Vergleich zu GROMOS™ ist am ehsten eine Korrelation mit 103.su2cor und 125.turb3d zu erwarten. In den folgenden Grafiken sind die Zeiten dieser beiden Applikationen dargestellt, um einen genaueren Vergleich mit GROMOS™ machen zu können.

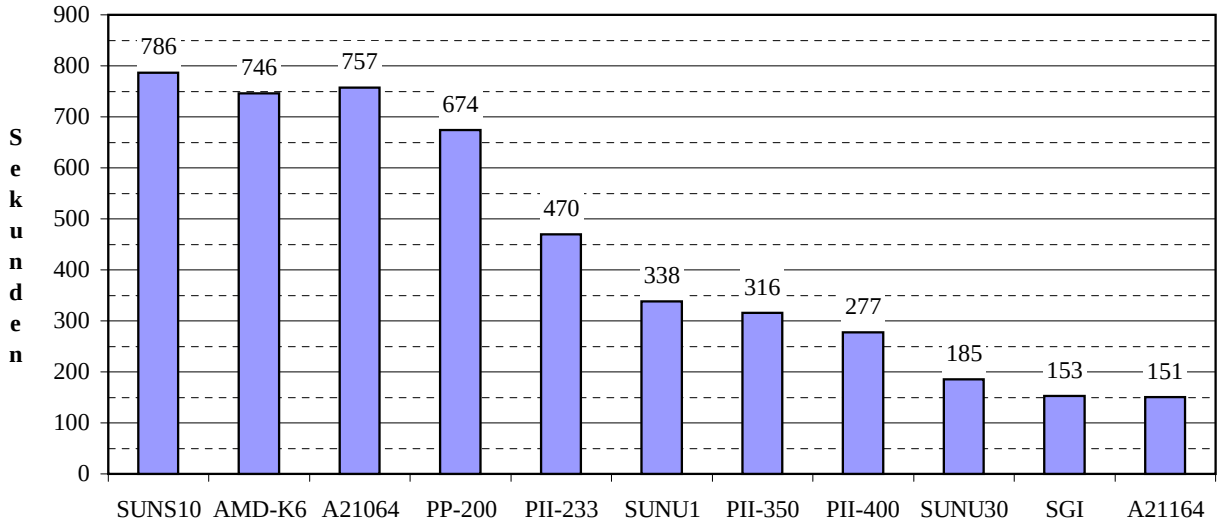


Abb. 3.33: Zeiten von 103.su2cor

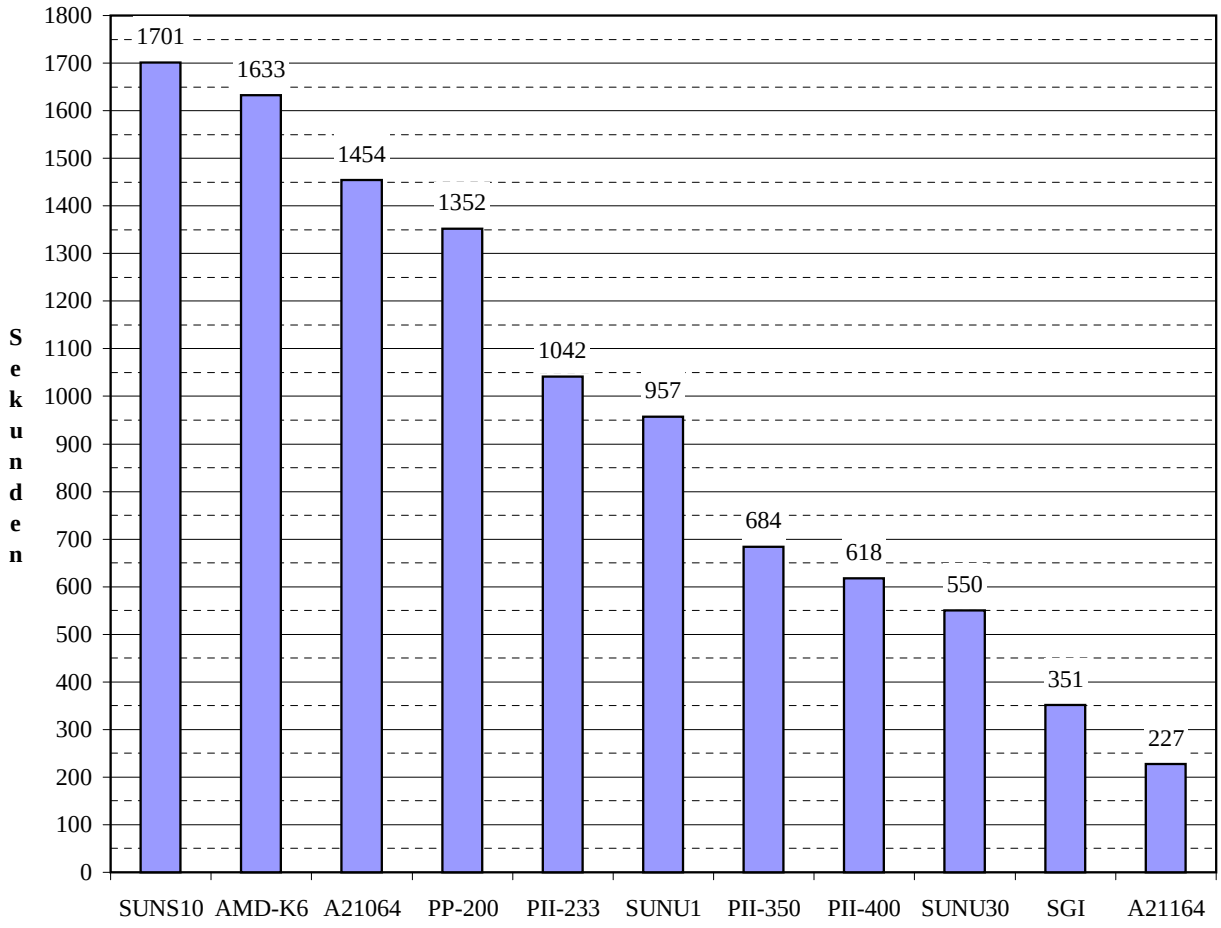


Abb. 3.34: Zeiten von 125.turb3d

In 103.su2cor (Abb. 3.33) ist der Abfall der Zeit bezogen auf die Rechner bei den schnellsten und den langsamsten Rechnern ziemlich schwach, nur im Mittelfeld zwischen Pentium Pro und SUN Ultra1 fällt die Zeit stark ab und reduziert sich dort um die Hälfte.
Der Zeitabfall in 125.turb3d (Abb. 3.34) ist wie beim Gesamtresultat der CFP95 in Abb. 3.14 praktisch linear, im Gegensatz zu GROMOS™, wo der Zeitabfall bei den langsamen Rechnern stärker und bis zu den schnellsten Rechner kontinuierlich stärker wird (Abb. 3.1 und Abb. 3.2).
Dies deutet darauf hin, dass im Mittelfeld grössere Abweichungen zu erwarten sind, wenn man GROMOS™ in Relation zur SPEC CPU95-Suite oder zu einer der CPU95-Applikationen setzt. Es ist jedoch schwierig, nur anhand der Grafiken eine Aussage über die Korrelation der beiden Softwarepakete zu machen.

3.3 Vergleich von GROMOS™ und SPEC CPU95

In diesem Kapitel wird der Zusammenhang zwischen GROMOS™ und der SPEC CPU95 genauer untersucht und aufgezeigt, ob eine Korrelation zwischen beiden besteht, bzw. wie stark die beiden voneinander abweichen.

3.3.1 Methode

Um festzustellen, ob zwei Software- oder Benchmarkprogramme miteinander korrelieren gibt es folgende zwei Methoden:

- Eine der Maschinen wird als Referenzmaschine genommen, worauf die Ausführungszeiten der Programme als Referenzzeiten gelten. Auf jeder Maschine wird jeweils für jedes Programm separat das Verhältnis zwischen Ausführungszeit und Referenzzeit gebildet. Dieses Verhältnis ist wie bei den SPEC-Benchmarks ein relatives Geschwindigkeitsmass oder Ratio bezüglich der Referenzmaschine. Korrelieren die beiden Programme, ergeben sich auf den gleichen Maschinen dieselben Ratios.
- Auf jeder Maschine wird das Verhältnis der Ausführungszeiten beider Programme gebildet. Bei einer Korrelation muss dieses Verhältnis auf allen Maschinen mit einer bestimmten Toleranz identisch sein.

Die erste Methode hat den Vorteil, dass für jedes Programm ein separater Geschwindigkeitsvergleich der Maschinen gemacht werden kann. Bei der Untersuchung der Korrelation zweier Programme können jedoch Probleme auftreten, da die Ratios ein relatives Geschwindigkeitsmass bezüglich einer Referenzmaschine darstellen. Korrelieren die Programme nur auf einem Teil der untersuchten Rechner, jedoch nicht auf der Referenzmaschine, ist dies nicht ersichtlich, da die Ratios auf den korrelierenden Maschinen nicht identisch sind.
Für die Untersuchung der Korrelation verwendeten wir deshalb die zweite Methode, wo dieses Problem nicht auftreten kann. In dieser Methode werden die Programme und nicht die Maschinen zueinander in Relation gesetzt. Damit können die Maschinen unabhängig voneinander betrachtet werden.
Für jede Maschine ergiebt sich dabei:

$$v_m = \frac{t_{p1}}{t_{p2}} \tag{3.6}$$

Der Index $\boldsymbol{m}$ steht für die jeweilige Maschine, $\boldsymbol{t_{p1}}$ und $\boldsymbol{t_{p2}}$ sind die Ausführungszeiten der beiden Programme auf der Maschine.
Die Verhältnisse $\boldsymbol{v_m}$ sagen nichts über die Performance der Rechner aus. Sie liefern nur den Faktor, um den sich die Zeiten der beiden Programme unterscheiden. Bei unterschiedlichen Programmen, wie in dieser Untersuchung, sind die absoluten Werte der Verhältnisse uninteressant. Nur die relativen Werte interessieren für die Untersuchung der Korrelation. Je mehr die Programme korrelieren, desto weniger weichen die Verhältnisse voneinander ab.
Die Genauigkeit der Korrelation ist zudem von der absoluten Grösse der Verhältnisse abhängig. Unterscheiden sich z.B. zwei Programme um den durchschnittlichen Faktor von 8, ist eine Abweichung von 0.5 auf einem Rechner weniger schlimm als wenn die Programme sich nur um den Faktor 4 unterscheiden.
Deshalb müssen die Verhältnisse normiert werden, um ein vergleichbares Mass der Abweichungen auf den verschiedenen Rechnern zu erhalten. Damit kann die Korrelation von mehreren Programmen auf den verschiedenen Rechnern verglichen werden.
Als Normierungsfaktor dient der Mittelwert der Verhältnisse.

Das normierte Verhältnis für jeden Rechner ergiebt sich so zu:

$$\tilde{v}_m = \frac{v_m}{\bar{v}} \tag{3.7}$$

Die Differenz zwischen normiertem Verhältnis und dem Mittelwert aller normierten Verhältnisse ergibt eine Aussage, wie stark die Korrelation zweier Programme auf einem Rechner vom Durchschnitt abweicht. Da der Mittelwert aller normierten Werte eins ist, erhält man die Abweichung bzw. Korrelation auf einem Rechner mit:

$$\tilde{c}_m = \tilde{v}_m - 1 \tag{3.8}$$

Ein Mass für die Genauigkeit der Korrelation für alle Rechner wird mit der Standardabweichung der normierten Verhältnisse erhalten.

$$c = \sqrt{var(\tilde{v}_m)} \tag{3.9}$$

Der Ausdruck ***var(v)*** steht dabei für Varianz von ***v***.
Man kann sehr leicht nachvollziehen, dass die Standardabweichung der normierten Verhältnisse mit dem relativen Streuungsmass der nicht normierten Verhältnisse übereinstimmt.
Damit gilt:

$$c = \frac{\sqrt{var(v_m)}}{\bar{v}} \tag{3.10}$$

Mit Formel 3.10 kann damit die Genauigkeit der Korrelation zweier Programme auf einer Serie von Rechnern bestimmt werden, während Formel 3.8 die Korrelation der zwei Programme auf einem einzelnen Rechner ergibt.

3.3.2 Verknüpfung von GROMOS™ und SPEC CPU95

Die Formeln in Kapitel 3.3.1 werden nun gebraucht, um die Korrelation von GROMOS™ und SPEC CPU95 zu bestimmen. Die Resultate der SPEC CPU95-Benchmarks sind sogenannte SPEC-Ratios, welche das Verhältnis zwischen den Ausführungszeiten der untersuchten Maschine und einer Referenzmaschine angeben (siehe Kapitel 2.2).
Gemäss Formel 3.6 wird dann das Verhältnis für jede Maschine zu:

$$v_m = \frac{t_{mGROMOS}}{t_{mCPU95}} \tag{3.11}$$

Eine Relation zwischen GROMOS™-Zeit und CPU95-Zeit kann auch unter Verwendung der SPEC-Ratios hergestellt werden. Betrachtet man die CPU95-Ratio als

$$R_{mCPU95} = \frac{t_{Ref95}}{t_{mCPU95}} \tag{3.12}$$

kann das Verhältnis für die einzelnen Maschinen auch geschrieben werden als:

$$v_m = t_{mGROMOS} \cdot R_{mCPU95} \tag{3.13}$$

Ersetzt man nun $\boldsymbol{R_{mCPU95}}$ in Formel 3.13 mit dem Wert in Formel 3.12 ergiebt sich:

$$v_m = \frac{t_{mGROMOS}}{t_{mCPU95}} \cdot t_{Ref95} \tag{3.14}$$

Die Verhältnisse in den Formeln 3.11 und 3.14 unterscheiden sich um den Faktor $\boldsymbol{t_{Ref95}}$. Betrachtet man, dass $\boldsymbol{t_{Ref95}}$ die Ausführungszeit der Referenzmaschine ist und damit für alle Maschinen konstant bleibt, kann nachgewiesen werden, dass der Faktor $\boldsymbol{t_{Ref95}}$ bei der Normierung verschwindet und für die Bestimmung der Korrelation keine Rolle spielt.
Der Mittelwert aller $\boldsymbol{v_m}$ bei $\boldsymbol{n}$ Maschinen ist:

$$\bar{v} = \frac{1}{n}\sum_{m}^{n} v_m = \frac{1}{n}\sum_{m}^{n} \frac{t_{mGROMOS}}{t_{mCPU95}} \cdot t_{Ref95} = \frac{t_{Ref95}}{n}\sum_{m}^{n} \frac{t_{mGROMOS}}{t_{mCPU95}} \tag{3.15}$$

Setzt man dies und den Wert aus Formel 3.14 in Formel 3.7 ein, wird das normierte Verhältnis zu:

$$\tilde{v}_m = \frac{\frac{t_{mGROMOS}}{t_{mCPU95}} \cdot t_{Ref95}}{\frac{t_{Ref95}}{n}\sum_{m}^{n} \frac{t_{mGROMOS}}{t_{mCPU95}}} = \frac{\frac{t_{mGROMOS}}{t_{mCPU95}}}{\frac{1}{n}\sum_{m}^{n} \frac{t_{mGROMOS}}{t_{mCPU95}}} \tag{3.16}$$

Dabei kann der Faktor $\boldsymbol{t_{Ref95}}$ weggekürzt werden.
Bei $\boldsymbol{n}$ Maschinen wird die Standardabweichung aller $\boldsymbol{v_m}$ zu:

$$\sqrt{var(v_m)} = \sqrt{\frac{1}{n-1}\sum_{m}^{n} (v_m - \bar{v})^2}$$

$$= \sqrt{\frac{1}{n-1}\sum_{m}^{n} \left(\frac{t_{mGROMOS}}{t_{mCPU95}} \cdot t_{Ref95} - \frac{t_{Ref95}}{n}\sum_{m}^{n} \frac{t_{mGROMOS}}{t_{mCPU95}} \right)^2} \tag{3.17}$$

$$= t_{Ref95} \cdot \sqrt{\frac{1}{n-1}\sum_{m}^{n} \left(\frac{t_{mGROMOS}}{t_{mCPU95}} - \frac{1}{n}\sum_{m}^{n} \frac{t_{mGROMOS}}{t_{mCPU95}} \right)^2}$$

Unter Verwendung der Formel 3.10 wird dann das relative Streuungsmass zu:

$$c = \frac{t_{Ref95} \cdot \sqrt{\frac{1}{n-1}\sum_{m}^{n}\left(\frac{t_{mGROMOS}}{t_{mCPU95}} - \frac{1}{n}\sum_{m}^{n}\frac{t_{mGROMOS}}{t_{mCPU95}}\right)^2}}{\frac{t_{Ref95}}{n}\sum_{m}^{n}\frac{t_{mGROMOS}}{t_{mCPU95}}} \tag{3.18}$$

$$= \frac{\sqrt{\frac{1}{n-1}\sum_{m}^{n}\left(\frac{t_{mGROMOS}}{t_{mCPU95}} - \frac{1}{n}\sum_{m}^{n}\frac{t_{mGROMOS}}{t_{mCPU95}}\right)^2}}{\frac{1}{n}\sum_{m}^{n}\frac{t_{mGROMOS}}{t_{mCPU95}}}$$

In den Formeln 3.16 und 3.18 wird gezeigt, dass t_{Ref95} weggekürzt werden kann. An Stelle der SPEC CPU95-Zeiten können daher für die Festlegung der Korrleation auch die SPEC-Ratios verwendet werden.

3.3.3 Korrelation zwischen GROMOS™ und SPEC CPU95

Im folgenden Diagramm ist die Korrelation zwischen der kubischen und oktaederischen GROMOS™-Simulation und der SPEC CINT95 und CFP95 Benchmarks für alle Rechner dargestellt. Gemäss Kapitel 3.3.2 verwendeten wir die SPEC<int/fp>-Base95-Ratios für den Vergleich:

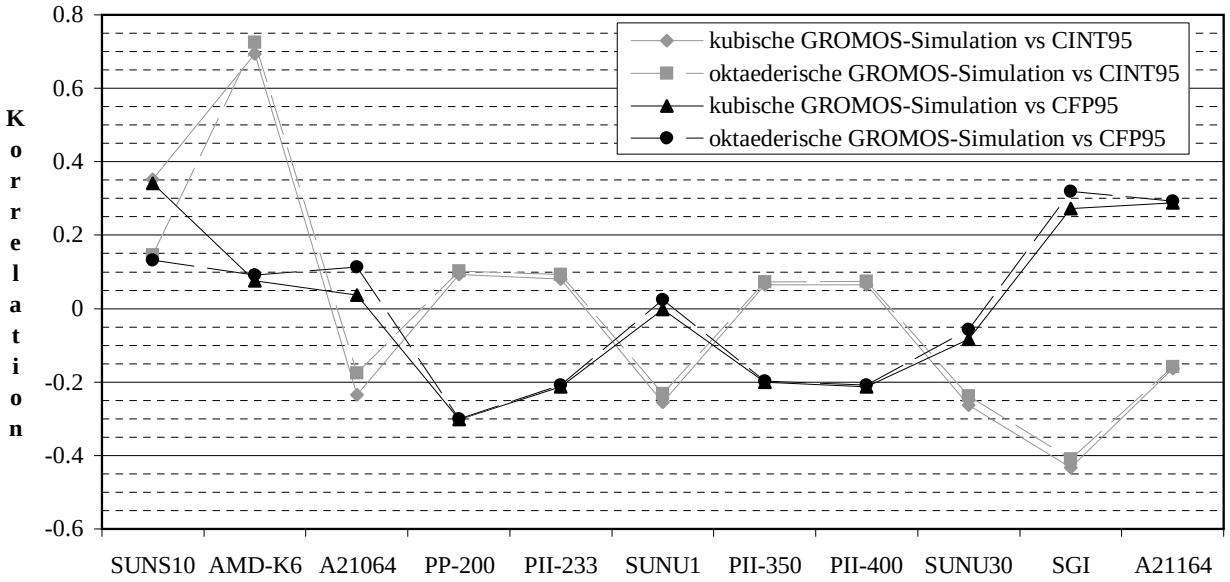


Abb. 3.35: Korrelation zwischen GROMOS™ und SPEC CPU95

Folgende Tabelle zeigt die Genauigkeit der Korrelation für die vier Vergleiche. Je kleiner der Wert, desto genauer die Korrelation.

Vergleich	kubische GROMOS™-Simulation	oktaederische GROMOS™-Simulation
CINT95	0.32	0.30
CFP95	0.23	0.21

Tabelle 3.4: Genauigkeit der Korrelationen zwischen GROMOS™ und SPEC CPU95

Wie schon in Kapitel 3.2.2 festgestellt wurde, korreliert GROMOS™ eher mit SPEC CFP95 als mit CINT95. Die kubische und oktaederische Version von GROMOS™ verhalten sich auf den einzelnen Rechnern praktisch gleich (Abb. 3.35), wobei die oktaederische Version noch ein bischen besser mit SPEC korreliert.
Die Relationen auf den einzelnen Rechnern weichen teilweise sehr stark voneinander ab. Dabei ist festzustellen, dass die Programme einerseites auf den PCs, andererseits auf den SUNs und teilweise den Alphas korrelieren. In Abb. 3.35 weisen alle PCs ausser die AMD praktisch die gleichen Werte auf. Genauso ist es bei den SUNs, wobei in den Vergleichen mit CINT95 zusätzlich die Alphas ähnliche Werte wie die SUNs aufweisen.
Die Genauigkeit der Korrelation ist aber immer noch zu gering. Wollte man anhand der SPEC-Ratios die Simulationsdauer von GROMOS™ abschätzen, würde die berechnete Simulationszeit bei Verwendung der SPECfp-Base95-Ratios durchschnittlich um 22% von der gemessenen Ausführungszeit abweichen.

3.3.4 Korrelation zwischen GROMOS™ und den SPEC CPU95-Applikationen

Um eine genauere Korrelation zwischen GROMOS™ und SPEC CPU95 zu finden, verglichen wir GROMOS™ mit den einzelnen Applikationen der CPU95. Da in der GROMOS™-Simulation praktisch nur Floating-Point Operationen verwendet werden, korreliert diese schlecht mit den CINT95-Applikationen, wie folgende Tabelle zeigt:

Vergleich	kubische GROMOS™-Simulation	oktaederische GROMOS™-Simulation
099.go	0.34	0.33
124.m88ksim	0.33	0.31
126.gcc	0.42	0.41
129.compress	0.26	0.21
130.li	0.33	0.31
132.ijpeg	0.26	0.22
134.perl	0.38	0.37
147.vortex	0.35	0.34

Tabelle 3.5: Genauigkeit der Korrelationen zwischen GROMOS™ und der SPEC CINT95-Applikationen

Bei keiner Applikation ist eine Abweichung von weniger als 20% zu erwarten, sodass sich eine weitere Untersuchung der Korrelation mit den CINT95-Applikationen erübrigt.

Im folgenden werden deshalb nur noch die CFP95-Applikation betrachtet.

Vergleich	kubische GROMOS™-Simulation	oktaederische GROMOS™-Simulation
101.tomcatv	**0.16**	**0.14**
102.swim	0.30	0.31
103.su2cor	0.23	0.21
104.hydro2d	0.24	0.23
107.mygrid	0.36	0.37
110.applu	**0.14**	**0.13**
125.turb3d	0.32	0.30
141.apsi	0.29	0.29
145.fpppp	0.50	0.51
146.wave5	0.40	0.36

Tabelle 3.6: Genauigkeit der Korrelationen zwischen GROMOS™ und der SPEC CFP95-Applikationen

GROMOS™ korreliert am besten mit 101.tomcatv und 110.applu. Hier liegt die durchschnittliche Abweichung bei rund 15%. Interessanterweise liegt die Abweichung bei 125.turb3d bei 30%, obwohl wir für diese Applikation gemäss Kapitel 3.2.3 eine gute Korrelation vermuteten. Auch bei 103.su2cor liegt die Abweichung über 20%. 145.fpppp und 146.wave5 korrelieren am schlechtesten mit GROMOS™. Die Abweichungen sind hier über 40%.
Eine genauere Information wie die beiden Programme auf den einzelnen Rechnern korrelieren liefern folgende Diagramme:

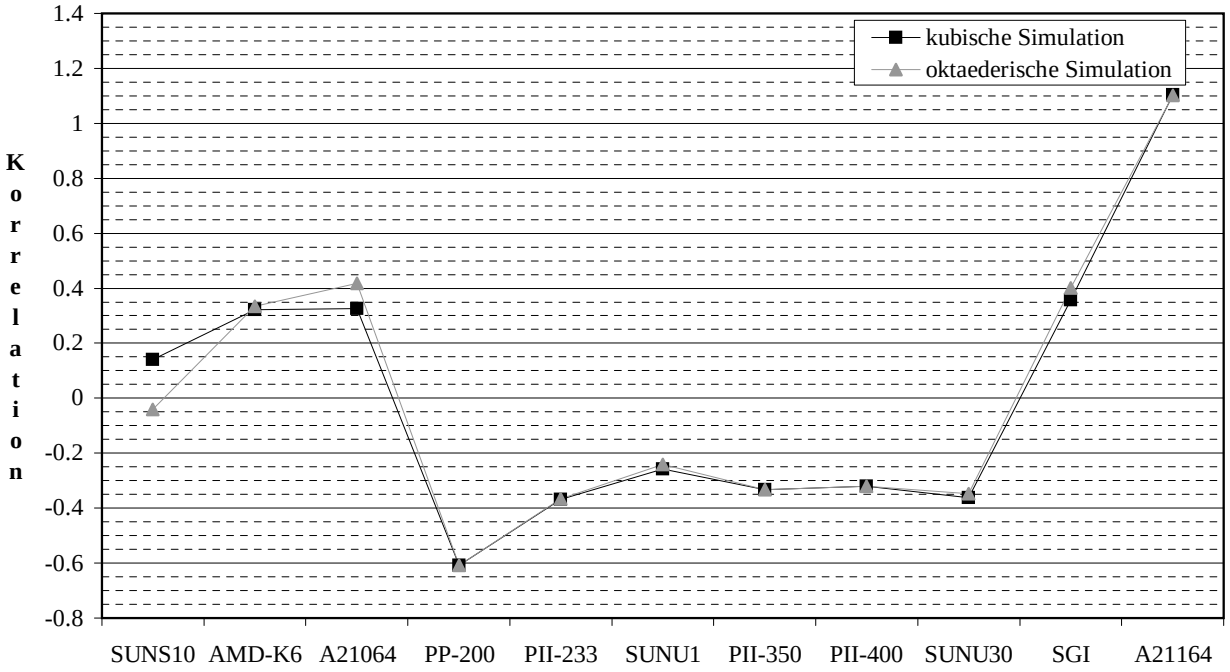


Abb. 3.36: Korrelation zwischen GROMOS™ und 145.fpppp

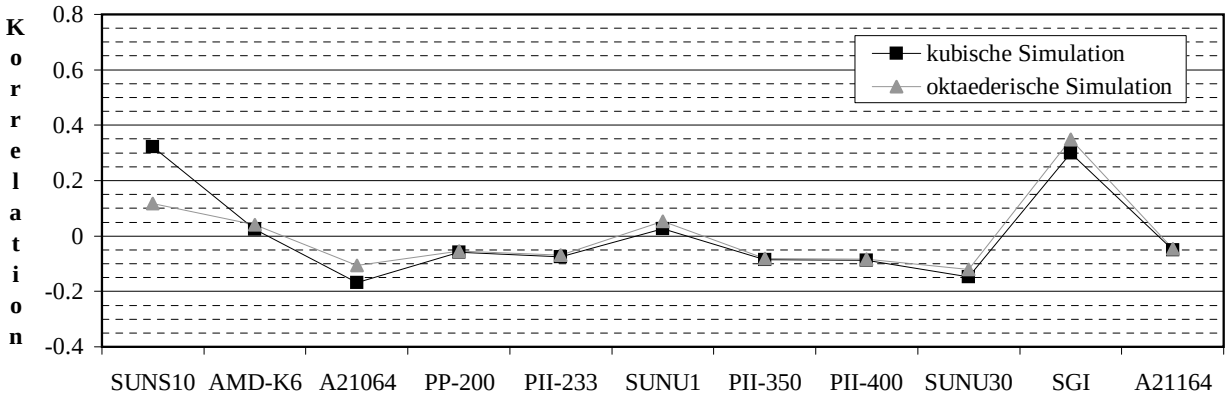


Abb. 3.37: Korrelation zwischen GROMOS™ und 101.tomcatv

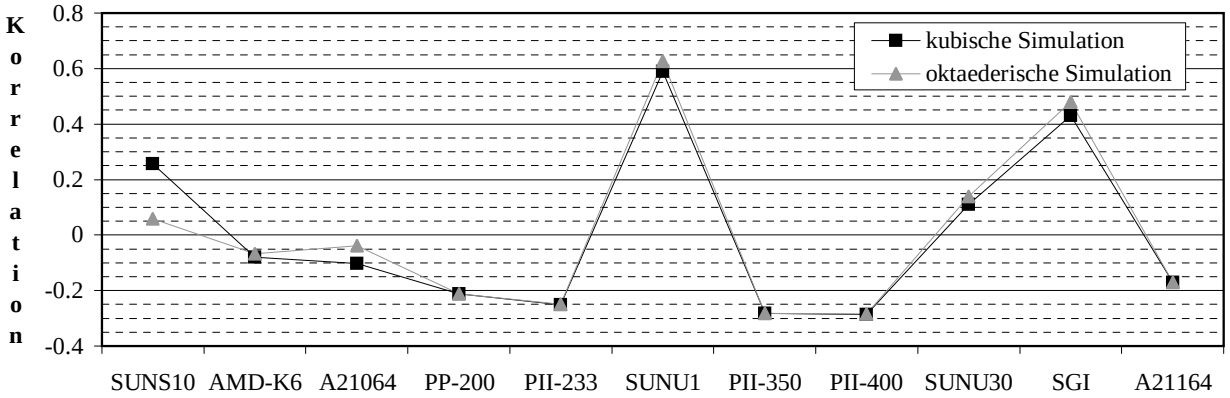


Abb. 3.38: Korrelation zwischen GROMOS™ und 102.swim

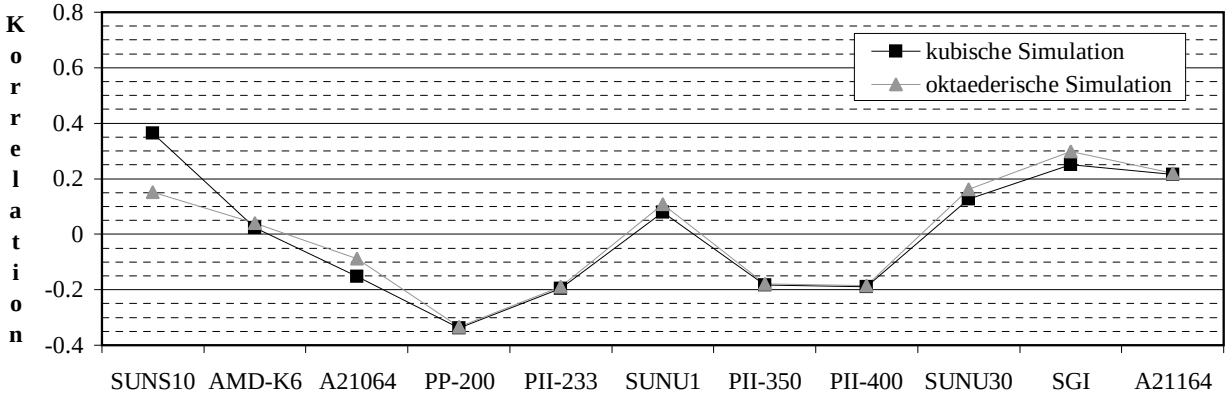


Abb. 3.39: Korrelation zwischen GROMOS™ und 103.su2cor

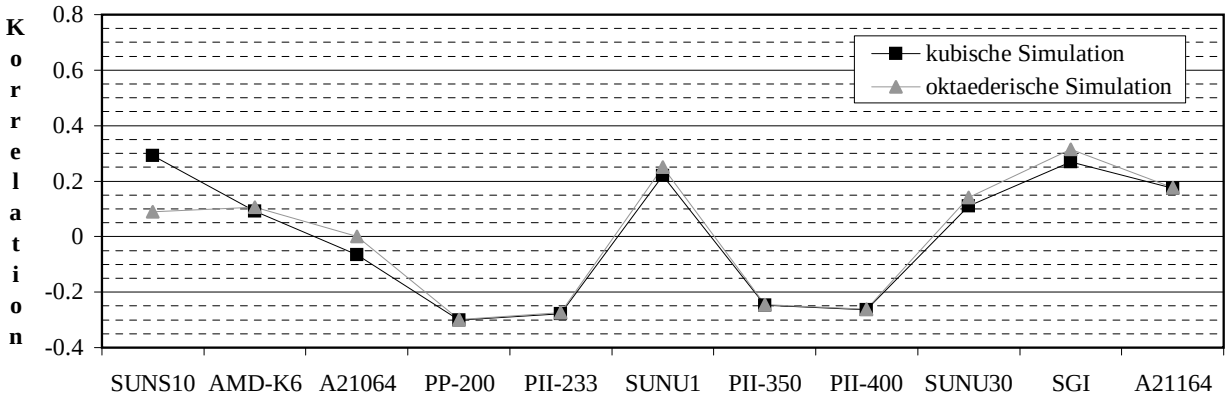


Abb. 3.40: Korrelation zwischen GROMOS™ und 104.hydro2d

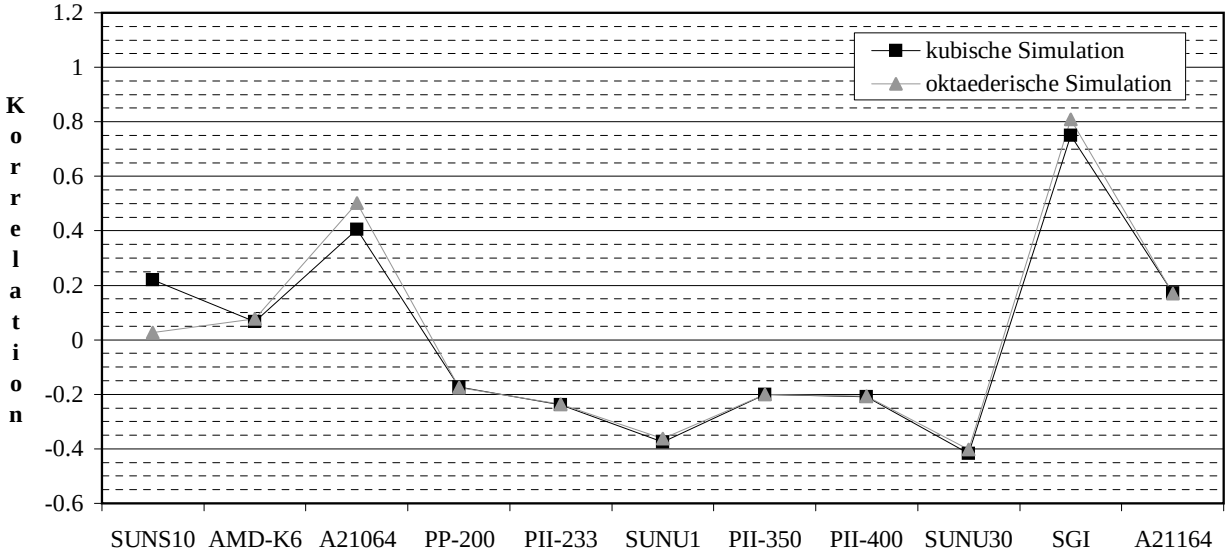


Abb. 3.41: Korrelation zwischen GROMOS™ und 107.mgrid

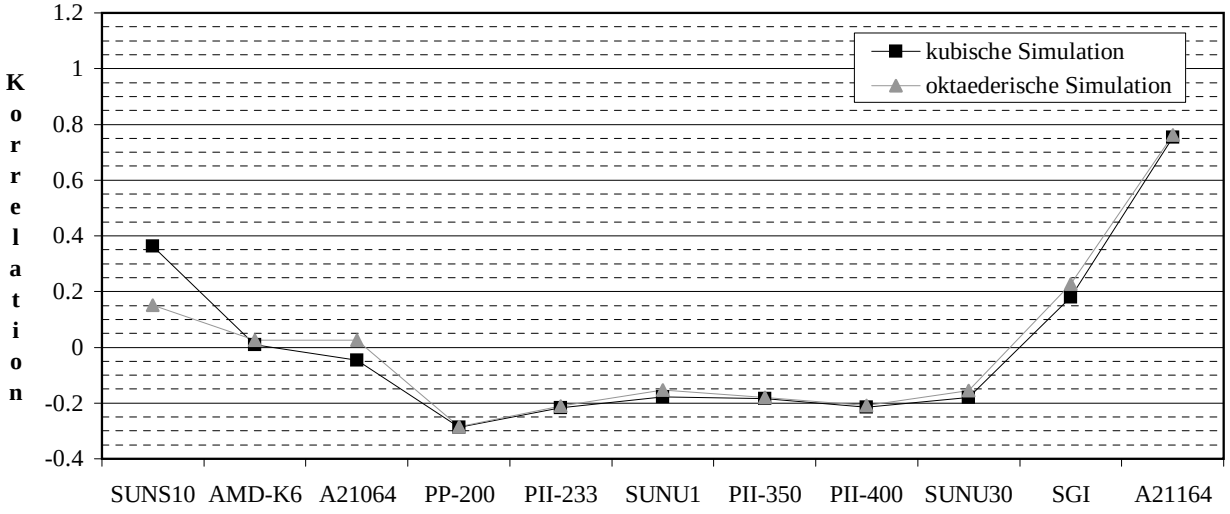


Abb. 3.42: Korrelation zwischen GROMOS™ und 125.turb3d

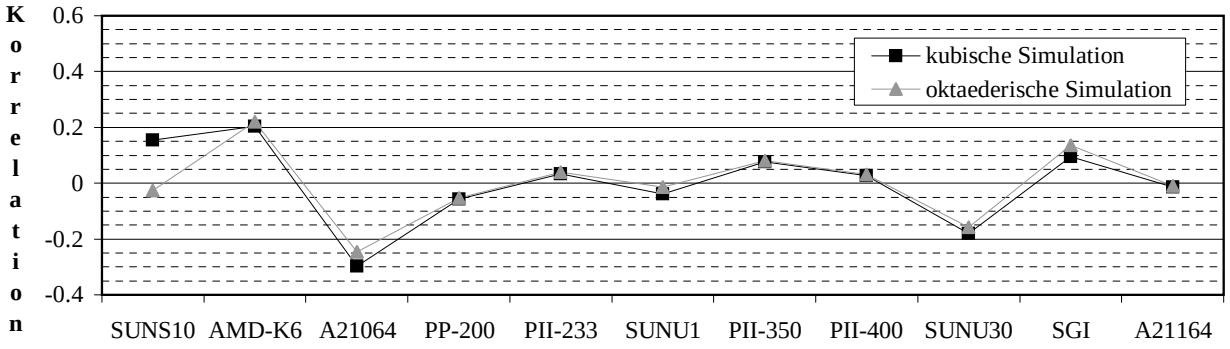


Abb. 3.43: Korrelation zwischen GROMOS™ und 110.applu

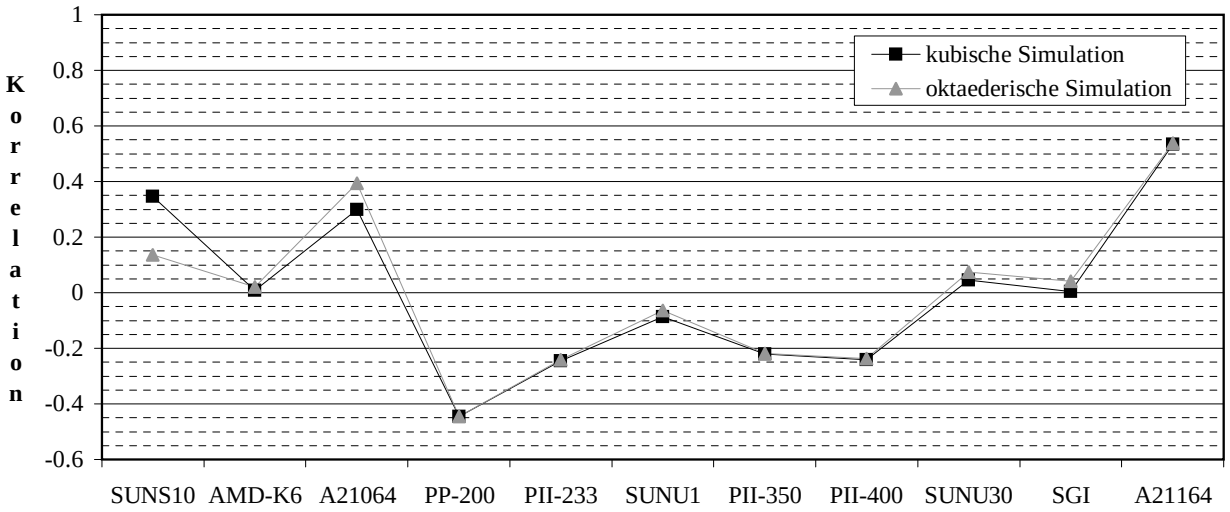


Abb. 3.44: Korrelation zwischen GROMOS™ und 141.apsi

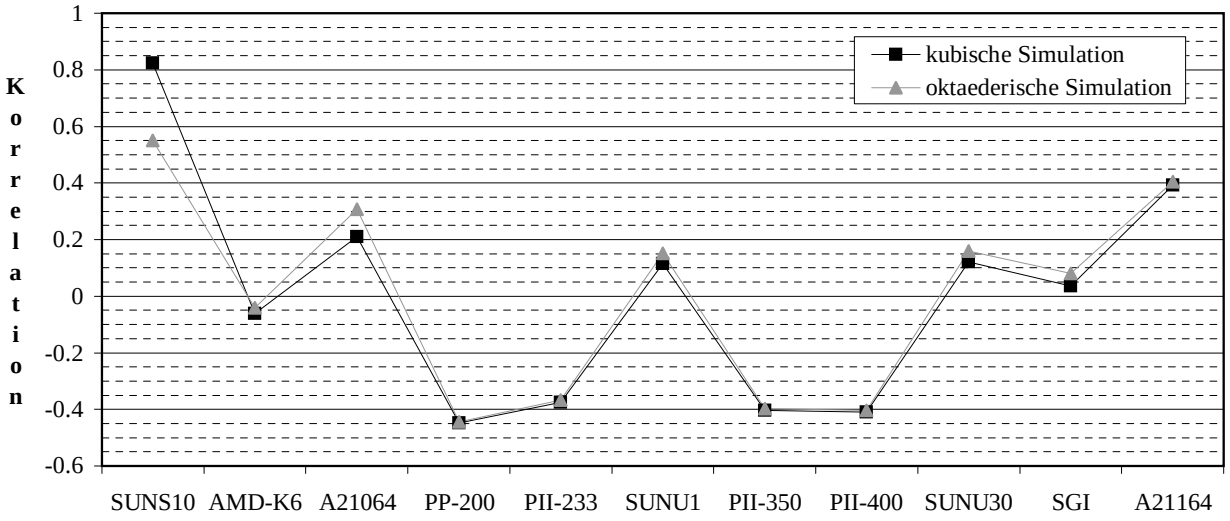


Abb. 3.45: Korrelation zwischen GROMOS™ und 146.wave5

Wie beim Vergleich von GROMOS™ mit dem Gesamtresultat der SPEC CFP95 in Kapitel 3.3.3 verhält sich die Korrelation für die beiden Simulationstopologien praktisch gleich. Die grössten Abweichungen zwischen kubischer und oktaederischer Simulation weisen die SUN SPARC10 und die Alpha21064 auf.
Die Korrelation der SUN SPARC10 weicht vor allem für die kubische Simulation bei praktisch allen Applikationen stark vom Mittelwert ab. Nur bei 107.mgrid (Abb. 3.41) und 145.fpppp (Abb. 3.36) liegt sie in einem vernünftigen Rahmen. Ähnlich ist es bei der SGI, wo die Korrelation bei der Hälfte der CFP95-Applikationen eine hohe Abweichung vom Mittelwert aufweist. Bei 125.turb3d (Abb. 3.42), 141.apsi (Abb. 3.44), 145.fpppp (Abb. 3.36) und 146.wave5 (Abb. 3.45), wo die Abweichung für die SGI klein ist, weist hingegen die Korrelation der Alpha21164 grosse Werte auf. Nur in 110.applu (Abb. 3.43) liegen SGI und Alpha21164 im Mittelfeld. Diese Applikation korreliert auch mit GROMOS™ am besten. Die PCs verhalten sich für die einzelnen Applikationen analog wie im Vergleich mit dem Gesamtresultat der CFP95 in Kapitel 3.3.3. Ausser der AMD weisen alle PCs ähnliche Abweichungen auf. Die Werte der beiden SUN Ultra Maschinen sind mit wenigen Ausnahmen in allen Applikationen praktisch gleich. Nur in 102.swim (Abb. 3.38) ist ein grösserer Unterschied festzustellen.
Die Korrelation verhält sich für einzelne Applikationen sehr ähnlich. So weisen 141.apsi (Abb. 3.44) und 146.wave5 (Abb. 3.45) praktisch dieselben Kurven auf. Genauso ähnlich verhalten sich 103.su2cor (Abb. 3.39) und 104.hydro2d (Abb. 3.40). Sieht man vom Verhalten der SUN Ultra1 einmal ab, kommen zu den beiden letzteren Applikationen noch 101.tomcatv (Abb. 3.37) und 102.swim (Abb. 3.38) hinzu. Die Kurven von 125.turb3d (Abb. 3.42) und 145.fpppp (Abb. 3.36) weisen im Vergleich zu denen der anderen Applikationen am wenigsten Ähnlichkeit auf.
In der Kurve von 125.turb3d wird nun auch die Feststellung von Kapitel 3.2.3 (Seite 29) sichtbar. Da der Zeitabfall bei GROMOS™ bei den langsamen Rechnern stärker ist als bei 125.turb3d, sinkt auch die Kurve von 125.turb3d ab. Erst bei den schnellen Rechnern steigt die Kurve wieder an, da dort das Umgekehrte der Fall ist.
Vergleicht man die Kurven der einzelnen Applikationen mit den Werten in Tabelle 3.6, sind teilweise einzelne Rechner für eine hohe mittlere Abweichung der Korrelation verantwortlich. Einzelne Applikationen korrelieren besser, wenn man die SGI, die Alpha21164 oder auch die SUN SPARC10 bei der Untersuchung weglässt. In 125.turb3d liegt die Genauigkeit der Korrelation dann bei 13%. In 101.tomcatv kann die Abweichung der Korrelation sogar auf 8% verringert werden, wenn man nur die SGI weglässt.
Insgesamt ist die Korrelation bei allen CFP95-Applikationen eher schlecht. Bei Betrachtung der einzelnen Applikationen lässt sich aber eine bessere Korrelation mit GROMOS™ erzielen als bei Verwendung des Gesamtresultats der SPEC CFP95.

3.3.5 Korrelation zwischen den GROMOS™-Prozeduren und den SPEC CFP95-Applikationen

Da sich die GROMOS™-Prozeduren im Vergleich zur Gesamtsimulation auf den einzelnen Rechnern teilweise unterschiedlich verhalten, kann eventuell eine bessere Korrelation gefunden werden, wenn man die einzelnen Prozeduren von GROMOS™ mit den CFP95-Applikationen und deren Gesamtresultat vergleicht.
Da sich die Korrelation für die kubische und oktaederische Simulation praktisch gleich verhält (siehe Kapitel 3.3.3 und Kapitel 3.3.4), haben wir in diesem Kapitel nur die kubische Topologie betrachtet.

In der folgenden Tabelle sind die mittleren Abweichungen aller Vergleiche der GROMOS™-Prozeduren mit den CFP95-Applikationen sowie mit dem Gesamtresultat der CFP95 aufgelistet. Für jede Prozedur sind die genausten Korrelationen fett abgedruckt:

Vergleich	Pairlist	Solute-Kräfte	Solvent-Kräfte	Integration	Rest
101.tomcatv	**0.17**	0.54	**0.13**	0.25	**0.13**
102.swim	0.30	0.57	0.29	0.46	0.27
103.su2cor	0.26	0.58	0.18	0.31	0.21
104.hydro2d	0.27	0.55	0.21	0.34	0.24
107.mygrid	0.34	0.61	0.35	0.39	0.44
110.applu	0.18	**0.44**	0.15	**0.20**	0.15
125.turb3d	0.41	0.61	0.27	0.29	0.38
141.apsi	0.35	0.60	0.26	0.28	0.41
145.fpppp	0.59	0.63	0.48	0.46	0.61
146.wave5	0.42	0.82	0.33	0.41	0.40
CFP95	0.26	0.57	0.19	0.27	0.26

Tabelle 3.7: Genauigkeit der Korrelationen zwischen den GROMOS™-Prozeduren und den CFP95-Applikationen

Die Pairlist-Routine, die Berechung der Solvent-Kräfte sowie die übrigen Teile der Simulation korrelieren alle am besten mit 101.tomcatv. Die Integration korreliert am genausten mit 110.applu. Dieses Verhalten ist analog zur Gesamtsimulation von GROMOS™, welche ebenfalls am besten mit diesen beiden CFP95-Applikationen korreliert (siehe Kapitel 3.3.4). Die Routine für die Berechnung der Solute-Kräfte korreliert mit keiner der CFP95-Applikationen. Der beste Wert liegt hier im Vergleich mit 110.applu bei 44%.
Am schlechtesten ist die Korrelation der drei CFP95-Applikationen 107.mygrid, 145.fpppp und 146.wave5 mit allen Prozeduren. Die Abweichungen liegen hier überall über 30%. Auch 102.swim, 125.turb3d und 141.apsi korrelieren relativ wenig mit den GROMOS™-Prozeduren, liegen die Abweichungen alle immer noch über 25%.
In den folgenden Diagrammen ist die Korrelation der CFP95-Applikationen für die einzelnen Rechner dargestellt. Auf die graphische Darstellung der Korrelationen von 102.swim, 107.mygrid, 141.apsi, 145.fpppp und 146.wave5 wurde aus den vorhin erwähnten Gründen verzichtet. Zusätzlich wurden noch die Korrelationen der GROMOS™-Prozeduren mit dem Gesamtresultat der CFP95-Applikationen grafisch dargestellt.

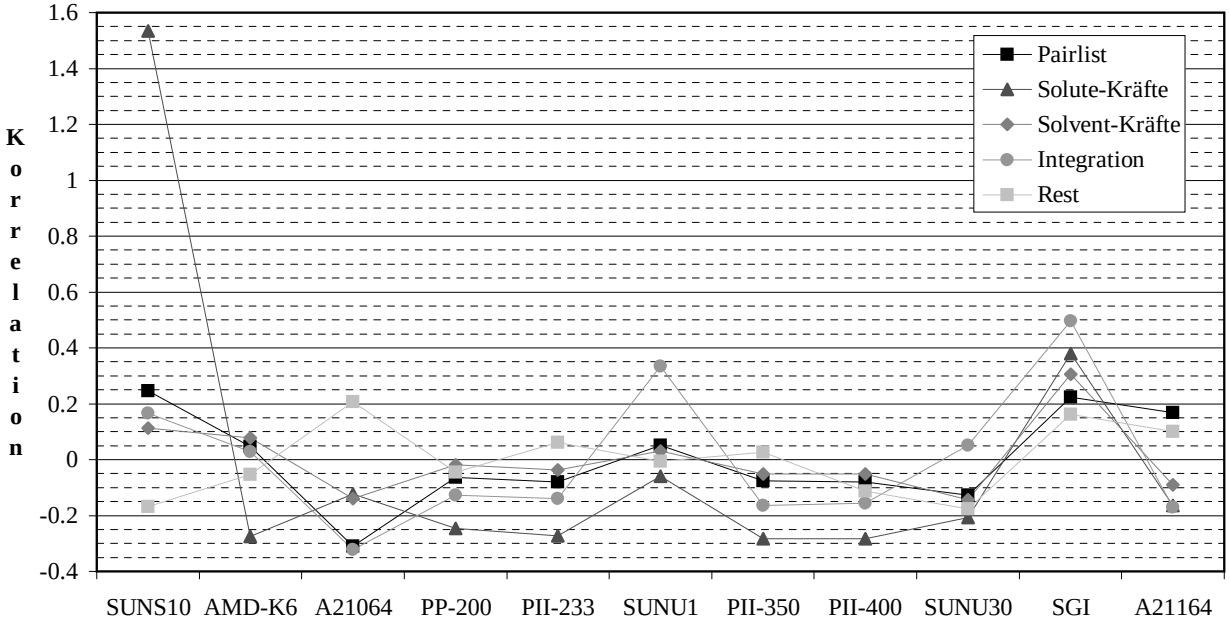


Abb. 3.46: Korrelation zwischen den GROMOS™-Prozeduren und 101.tomcatv

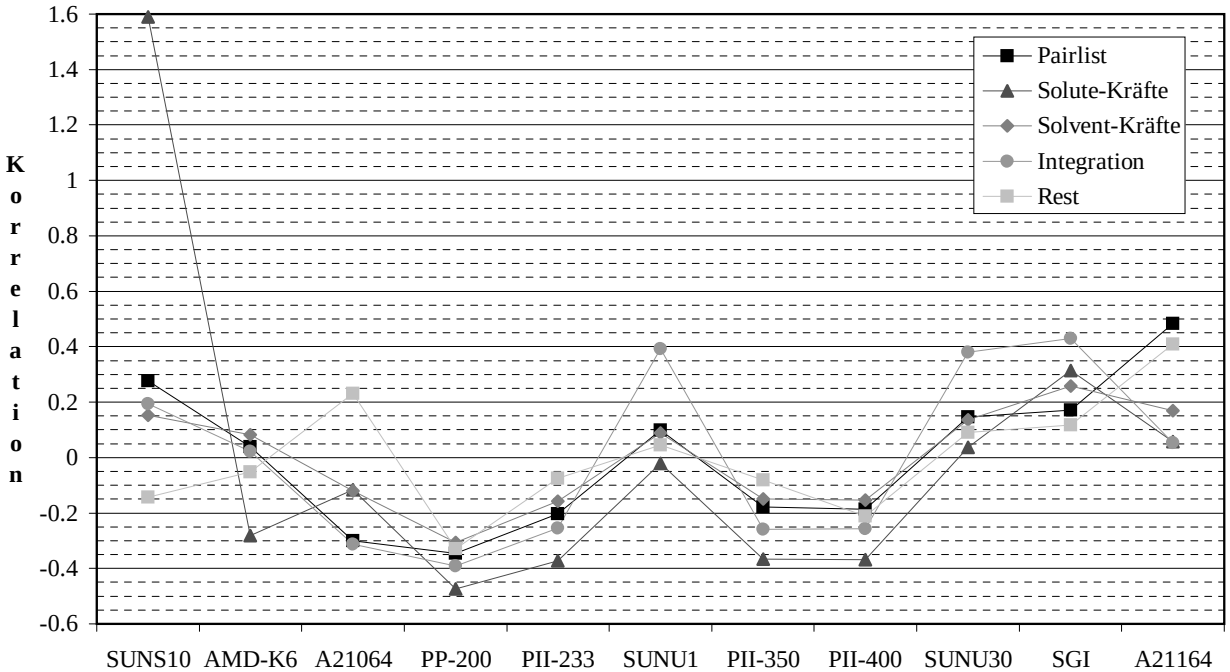


Abb. 3.47: Korrelation zwischen den GROMOS™-Prozeduren und 103.su2cor

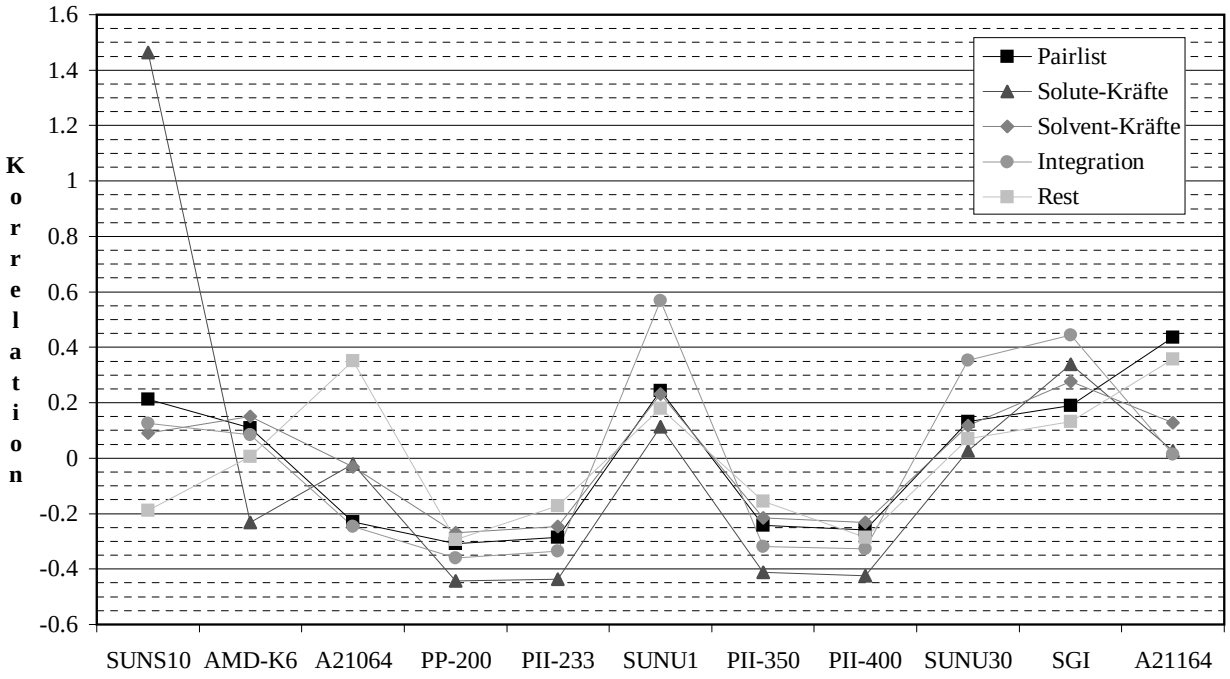


Abb. 3.48: Korrelation zwischen den GROMOS™-Prozeduren und 104.hydro2d

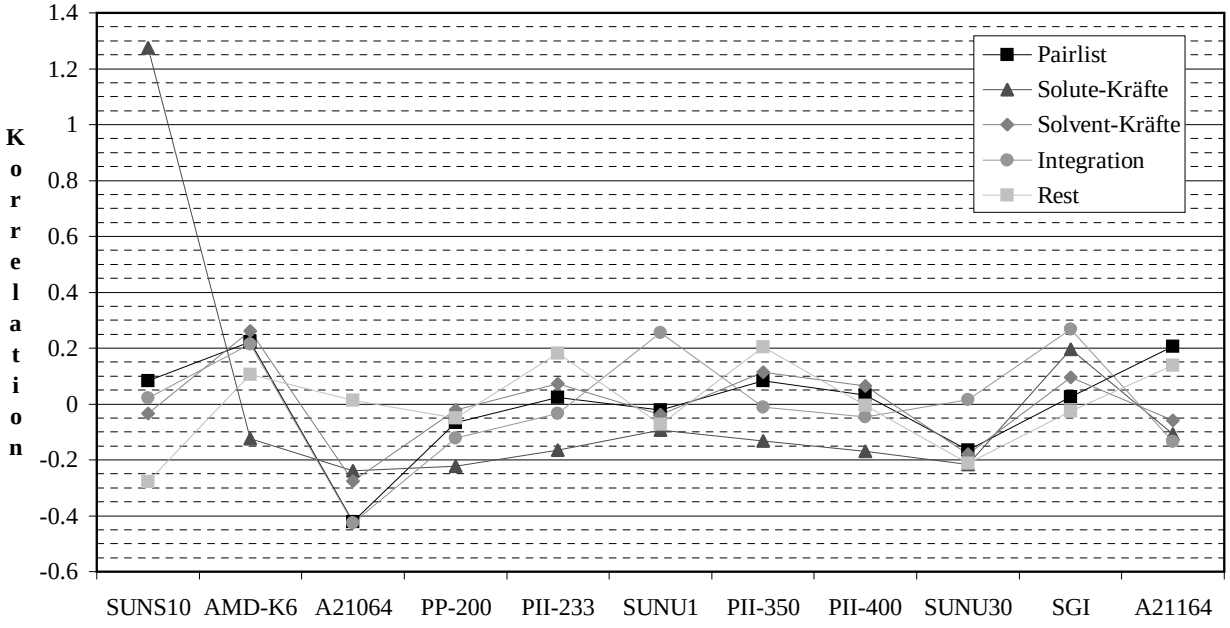


Abb. 3.49: Korrelation zwischen den GROMOS™-Prozeduren und 110.applu

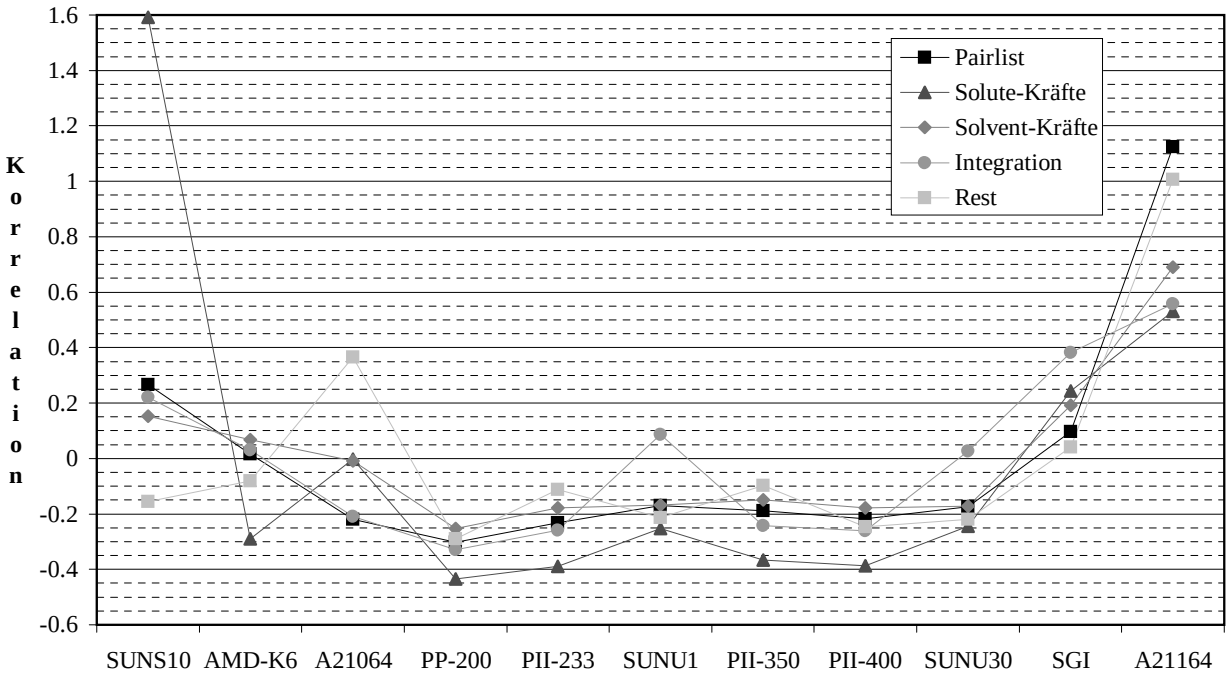


Abb. 3.50: Korrelation zwischen den GROMOS™-Prozeduren und 125.turb3d

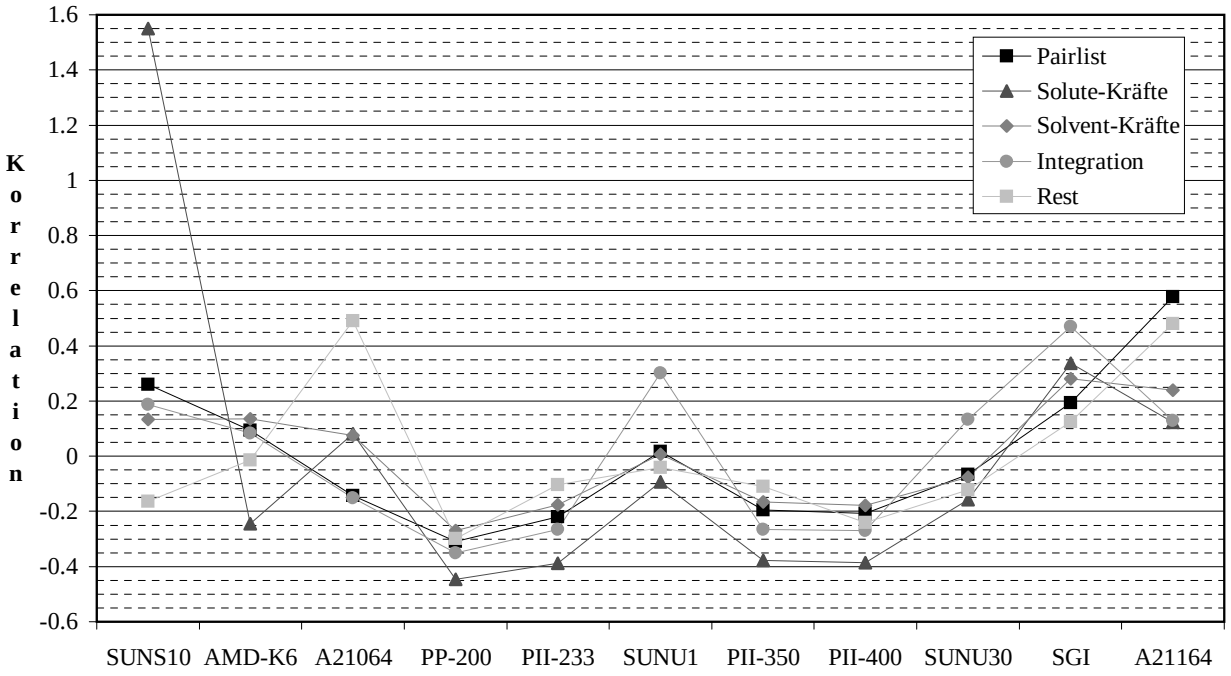


Abb. 3.51: Korrelation zwischen den GROMOS™-Prozeduren und SPEC CFP95

Beim Betrachten der Grafiken sind nun die hohen mittleren Abweichungen der Korrelation für die Solute-Kräfte-Routine erklärbar. Für diese Prozedur weicht der Wert für die SUN SPARC10 überall sehr stark vom Mittelwert ab. Weitere hohe Abweichungen der SUN SPARC10 sind für die rest-

lichen Routinen in 110.applu (Abb. 3.49) sowie für alle Routinen in 125.turb3d (Abb. 3.50) ausser den übrigen Prozeduren festzustellen. Die beiden SUN Ultra Maschinen weisen für alle Prozeduren in 103.su2cor (Abb. 3.47), 104.hydro2d (Abb. 3.48), 125.turb3d (Abb. 3.50) und der gesamten CFP95 (Abb. 3.51) sehr ähnliche Werte auf. In 101.tomcatv (Abb. 3.46) weichen die Werte der SUN Ultra1für alle Prozeduren ausser dem Rest und in 110.applu für die Integration stärker vom Mittelwert ab. Die Alpha21064 weist für die restlichen GROMOS™-Routinen überall ausser in 110.applu hohe Abweichungen auf. In 125.turb3d weisen SGI und Alpha21164, in 101.tomcatv nur die SGI für alle Prozeduren hohe Abweichungen vom Mittelwert auf. Die PCs weisen mit Ausnahme der übrigen Routinen praktisch überall dieselben Werte auf.
Einzelne GROMOS™-Prozeduren verhalten sich im Vergleich zu den CFP95-Applikationen sehr ähnlich. Die Kurve der Pairlist-Routine ist für 103.su2cor (Abb. 3.47) und 104.hydro2d (Abb. 3.48) praktisch gleich. Die Solute-Kräfte-Routine zeigt sogar für 101.tomcatv (Abb. 3.46), 103.su2cor, 104.hydro2d und die CFP95 (Abb. 3.51) ein ähnliches Verhalten. Bei der Solvent-Kräfte-Routine sind die Kurven einerseits in den Vergleichen mit 103.su2cor und 104.hydro2d praktisch gleich, andererseits weisen die Vergleiche mit 101.tomcatv (Abb. 3.46), 110.applu (Abb. 3.49) und CFP95 (Abb. 3.51) ähnliche Kurven auf, wenn man die Alpha21164 bei der CFP95 nicht betrachtet. Die Integrations-Routine zeigt für alle betrachteten CFP95-Applikationen ausser 125.turb3d (Abb. 3.50) ein ähnliches Verhalten und die Kurven der restlichen Routinen sind für 103.su2cor und 104.hydro sowie für 101.tomcatv und 125.turb3d ausser der Alpha21164 sehr ähnlich. Zusammengefasst zeigen daher die beiden CFP95-Applikationen 103.su2cor und 104.hydro3d im Vergleich zu allen GROMOS™-Prozeduren dasselbe Verhalten.
Vergleicht man die Korrelationen der einzelnen GROMOS™-Prozeduren untereinander, ergiebt sich für alle betrachteten CFP95-Applikationen folgendes Bild. Die Kurve der Pairlist-Routine hat eine ähnliche Tendenz wie die Solute-Kräfte-Routine, die Solvent-Kräfte-Routine und die Integrations-Routine. Abweichungen ergeben sich bei der Solute-Kräfte-Routine auf der SUN SPARC10, der Alpha21064 und der SGI, bei der Solvent-Kräfte-Routine auf der SGI und den beiden Alphas und bei der Integrations-Routine auf der SUN Ultra1 und der Alpha21164. Vergleicht man die Solute-Kräfte-Routine mit der Solvent-Kräfte-Routine zeigen diese nur im Mittelfeld der Rechner zwischen Pentium Pro 200 und SUN Ultra30 dieselbe Tendenz. Ähnlichkeiten gibt es noch zwischen der Solute-Kräfte-Routine und der Integrations-Routine ausser auf der SUN SPARC10 und der Alpha21064.
Da vereinzelte Rechner sehr starke Abweichungen der Korrelation vom Mittelwert aufweisen, wird die durchschnittliche Korrelation in einigen Vergleichen sehr schlecht. Wie oben schon erwähnt führt die hohe Abweichung der SUN SPARC10 bei der Solute-Kräfte-Routine auf die schlechten Resultate in Tabelle 3.7.
Im Folgenden wird nun versucht, durch Ausschliessen einzelner Maschinen die Genauigkeit der Korrelation zu verbessern. Dabei haben wir in einem ersten Schritt für jeden Vergleich der Programmroutinen die Maschine mit der grössten Abweichung zum Mittelwert ermittelt und anschliessend neu die durchschnittliche Abweichung aller restlichen Maschinen, d.h. ohne die Maschine, für welche die Programme am schlechtesten korrelieren, berechnet.
In der folgenden Tabelle sind die in diesem Kapitel genauer betrachteten Vergleiche aus den Abb. 3.46 bis Abb. 3.51 mit den jeweils schlechtesten korrelierenden Maschinen und deren Abweichungen aufgelistet. Zusätzlich kommt noch der Vergleich zwischen den jeweiligen CFP95-Prozeduren und der Gesamtsimulation von GROMOS™ hinzu.

Vergleich	Pairlist	Solute-Kräfte	Solvent-Kräfte	Integration	Rest	GROMOS™
101.tomcatv	A21064 -0.31	SUNS10 1.54	SGI 0.30	SGI 0.50	A21064 0.21	SUNS10 0.32
103.su2cor	A21164 0.48	SUNS10 1.59	PP-200 -0.31	SGI 0.43	A21164 0.41	SUNS10 0.36
104.hydro2d	A21164 0.44	SUNS10 1.46	SGI 0.28	SUNU1 0.57	A21164 0.36	PP-200 -0.30
110.applu	A21064 -0.42	SUNS10 1.27	A21064 -0.28	A21064 -0.42	SUNS10 -0.28	A21064 -0.30
125.turb3d	A21164 1.13	SUNS10 1.59	A21164 0.69	A21164 0.56	A21164 1.01	A21164 0.75
CFP95	A21164 0.58	SUNS10 1.55	SGI 0.28	SGI 0.47	A21064 0.49	SUNS10 0.34

Tabelle 3.8: Maschinen mit der höchsten Abweichung der Korrelation vom Mittelwert

Es fällt sofort auf, dass die SUN SPARC10 und die Alpha-Maschinen meistens die höchste Abweichung vom Mittelwert aufweisen. Zudem hat in den Vergleichen der verschiedenen CFP95-Applikationen mit derselben GROMOS™-Routine häufig auch dieselbe Maschine die höchste Abweichung vom Mittelwert. In der Pairlist-Routine ist das eine der beiden Alpha Maschinen, in der Solute-Kräfte-Routine ist es immer die SUN SPARC10 und in den restlichen meistens auch eine der beiden Alphas. In allen Vergleichen von 125.turb3d mit den GROMOS™-Routinen weist praktisch immer die Alpha21164 die grösste Abweichung vom Mittelwert auf. Weitere Maschinen mit den höchsten Abweichungen sind die SGI für jeweils zwei Vergleiche mit der Solvent-Kräfte-Routine und der Integration, bei zwei Vergleichen der Pentium Pro200 und einmal die SUN Ultra1.
Im Folgenden wurde nun für alle Vergleiche aus Tabelle 3.8 nochmals die Genauigkeit der Korrelation berechnet, wobei jeweils die Maschine mit der höchsten Abweichung aus Tabelle 3.8 nicht mehr für die Mittelwertbildung berücksichtigt ist.

Vergleich	Pairlist	Solute-Kräfte	Solvent-Kräfte	Integration	Rest	GROMOS™
101.tomcatv	0.13	0.24	**0.09**	0.23	**0.12**	0.14
103.su2cor	0.23	0.30	0.12	0.31	0.17	0.21
104.hydro2d	0.25	0.33	0.20	0.32	0.22	0.22
110.applu	**0.11**	**0.14**	0.12	**0.14**	0.13	**0.11**
125.turb3d	0.20	0.38	0.17	0.25	0.21	0.22
CFP95	0.20	0.31	0.18	0.25	0.23	0.21

Tabelle 3.9: Genauigkeit der Korrelationen von GROMOS™ und CFP95 ohne Maschinen mit der grössten Abweichung

Durch das Weglassen der am schlechtesten korrelierenden Maschine wird keine wesentliche Verbesserung erzielt. Nur in den Vergleichen mit der Solute-Kräfte-Routine werden die mittleren Abweichungen wesentlich verringert, da in diesen die SUN SPARC10 als einzige Maschine eine

wesentliche Abweichung im Vergleich zu den anderen Maschinen aufweist (siehe Abb. 3.46 - Abb. 3.51). In den anderen liegt die Verbesserung bei einigen Prozenten.
Auch wenn jeweils die am schlechtesten korrelierende Maschine beim Vergleich weggelassen wird, kann keine durchschnittliche Abweichung von weniger als 10% erreicht werden. Am besten korreliert noch 101.tomcatv mit der Solvent-Kräfte-Routine mit einer durchschnittlichen Abweichung von 9%.
Vergleicht man die besten Abweichungen der Vergleiche der einzelnen GROMOS™-Routinen mit dem Vergleich von 110.applu und der gesamten GROMOS™-Simulation, ist die Abweichung in letzterem mit 11% teilweise besser als in den Vergleichen mit den einzelnen GROMOMS™-Routinen. Hier stellt sich die Frage, ob man bei Verwendung von 110.applu mit direkter Abschätzung der Simulationszeit von GROMOS™ nicht bessere Resultate erreicht als mit Vorausberechnen und Aufsummieren der Zeiten der einzelnen Routinen. Folgende Abschätzung zeigt, dass letztere Methode ein bisschen genauere Resultate ergiebt als erstere.
Beim Aufsummieren der geschätzten Zeiten der einzelnen Routinen werden deren Abweichungen gemäss ihren Prozentanteilen bezüglich der Gesamtsimulation relativiert. Folgende Tabelle zeigt die über alle Rechner gemittelten Prozentanteile der GROMOS™-Prozeduren:

Pairlist	Solute-Kräfte	Solvent-Kräfte	Integration	Rest
18.8%	12.7%	66.6%	1.0%	1.0%

Tabelle 3.10: Mittlere Prozentanteile der GROMOS™-Prozeduren an der Gesamtsimulation

Bei einer Aufsummierung der Abweichungen ergiebt sich dann:

$$0.11 \cdot 18.8\% + 0.14 \cdot 12.7\% + 0.09 \cdot 66.6\% + 0.14 \cdot 1.0\% + 0.12 \cdot 1.0\% = 0.101$$

Mit der Vorausberechnung der Zeiten der einzelnen GROMOS™-Prozeduren und deren anschliessender Aufsummierung kann die Zeit der Gesamtsimulation ein bisschen genauer vorausgesagt werden. Ob die abgeschätzte Zeit nun aber um 10.1% oder 11% daneben liegt ist bei der Höhe der Ungenauigkeit nicht relevant. Insgesamt betrachtet ist eine Abweichung von 10% für die Voraussage der Laufzeit eines Programmes immer noch zu hoch.

4 Zusammenfassung

Die Untersuchungen in Kapitel 3.3 zeigen, dass GROMOS™ mit SPEC CPU95 eher nicht korreliert. Die mittleren Abweichungen der Korrelationen sind alle höher als 20%, was bedeutet, dass man bei einer Voraussage der Simulationszeit von GROMOS™ anhand der SPEC CPU95-Benchmarks rund 20% oder mehr daneben liegt. Zudem ist nicht bekannt, wie eine neuere Maschine allenfalls auf GROMOS™ und SPEC CPU95 reagiert. Die Abweichung einer Voraussage anhand der ermittelten Daten kann dann noch viel höher liegen.
Durch die Untersuchung der Prozeduren von GROMOS™ und der einzelnen Applikationen der SPEC CPU95 konnten bei den Vergleichen insgesamt bessere Resultate erzielt werden. Aber auch bei den genausten Korrelationen liegt die Abweichung immer noch bei rund 15%, zu ungenau für eine zuverlässige Voraussage. In einem nächsten Schritt wurden deshalb bei jedem Vergleich die Maschinen entfernt, für welche die Programme am schlechtesten korrelieren, mit dem Resultat, dass eine mittlere Abweichung von rund 10% erreicht werden kann. Dies ist jedoch immer noch zu ungenau. Zudem kann im Vergleich von 110.applu mit der Gesamtsimulation von GROMOS™ ebenfalls eine mittlere Abweichung von nur 11% erreicht werden.
Es wäre nun durchaus möglich, weitere Maschinen aus der Untersuchung auszuklammern, um die Resultate noch zu verbessern. Die Untersuchung soll jedoch ein breites Spektrum an Maschinen abdecken, weshalb eine solche Vorgehensweise wenig Sinn macht. Eine andere Möglichkeit, die Korrelation zu verbessern, wäre die Maschinen in Gruppen einzuteilen, z.B. alle PCs und alle übrigen Maschinen. Dabei taucht jedoch die Frage auf, welcher Gruppe man einen neuen Rechner zuteilen soll. Die Rechner und deren Prozessoren werden dauernd weiterentwickelt und es kann gut sein, dass z.B. ein neuer PC eher zusammen mit den SUNs und den Alphas für GROMOS™ und SPEC CPU95 korreliert als mit den alten PCs, da Intel für die Prozessoren vielleicht Arithmetikeinheiten der Alphas übernimmt und verwendet.
Ein Grund für die schlechte Korrelation zwischen GROMOS™ und der SPEC CPU95 ist das Fehlen einer Applikation für die Molekulardynamik. Die SPEC CFP95 weist praktisch nur Applikationen für Physiksimulationen auf. Nur 102.swim ist eine Chemiesimulation. Einige der Applikationen berechnen ähnliche Strukturen wie GROMOS™ oder beinhalten Teile, welche auch in GROMOS™ vorkommen, z.B. 101.tomcatv, welches Matrizen berechnet, die ebenfalls in GROMOS™ vorkommen oder 104.hydro2d, wo Algorithmen der Astrophysik berechnet werden, die mit der Molekulardynamik verwandt ist. Insgesamt sind die Applikationen aber zuwenig ähnlich mit GROMOS™, um damit zu korrelieren.
Die Untersuchung könnte nun beliebig ausgedehnt werden. Z.B. könnten weitere Rechner in die Untersuchung miteinbezogen oder andere Compiler und Optionen verwendet werden. Eine Kombinationen zweier oder mehrerer CFP95-Applikationen und deren Vergleich mit GROMOS™ könnten ebenfalls gemacht werden. Dies würde aber den Rahmen dieses Projektes sprengen und zudem wären damit wohl kaum bessere Resultate zu erzielen.

5 Ausblick

Wie in Kapitel 4 erwähnt, beinhaltet die SPEC CFP95-Suite keine Molekulardynamiksimulation. Es soll jedoch bald eine neue Version der SPEC-Benchmarksuite erhältlich sein. Genannt wird diese dann SPEC CPU2000. In dieser sollen einerseite mehr Applikationen als in der CPU95 vorhanden sein und andererseits soll diese in der Tat eine Molekulardynamiksimulation enthalten. Bei einer entsprechenden Untersuchung von GROMOS™ mit der neuen SPEC-Benchmarksuite sind möglicherweise bessere Resultate zu erwarten.

Workstation *[Abkürzung]*	SUN SPARC Station10/85 [SUNS10]	SUN Ultra 1/170 [SUNU1]	SUN Ultra 30/300 [SUNU30]
Prozessr *1st Level Cache* *2nd Level Cache* *RAM*	Super SPARC II / 85MHz 20KB(I)+16KB(D) intern 192KB extern 64MB	Ultra SPARC / 167MHz 16KB(I)+16KB(D) intern 256KB extern 192MB	Ultra SPARC / 296MHz 16KB(I)+16KB(D) intern 256KB extern 256MB
Betriebssystem	Solaris 2.5.1	Solaris 2.5.1	Solaris 2.5.1
Compiler für CINT95 *[Optimierungsflags]*	SUN CC 4.2 [-xO4 -xunroll=4 -xdepend -native]	SUN CC 4.2 [-xO4 -xunroll=4 -xdepend -native]	SUN CC 4.2 [-xO4 -xunroll=4 -xdepend -native]
Compiler für CFP95 *[Optimierungsflags]*	SUN Fortran77 4.2 [-O4 -unroll=4 -depend -native]	SUN Fortran77 4.2 [-O4 -unroll=4 -depend -native]	SUN Fortran77 4.2 [-O4 -unroll=4 -depend -native]

Dell GXPro Pentium Pro 200 [PP-200]	Vobis Highscreen Pentium II/233 [PII-233]	Dell GX1 Pentium II/350 [PII-350]	Dell Precision 410 Pentium II/400 [PII-400]
Intel Pentium Pro 200MHz 8KB(I)+8KB(D) intern 256KB extern 64MB	Intel Pentium II / 233MHz 16KB(I)+16KB(D) intern 512KB extern 64MB	Intel Pentium II / 350MHz 16KB(I)+16KB(D) intern 512KB extern 64MB	Intel Pentium II / 400MHz 16KB(I)+16KB(D) intern 512KB extern 128MB
SuSe-Linux 5.2 (2.0.35)	SuSe-Linux 5.2 (2.0.35)	SuSe-Linux 5.2 (2.0.35)	SuSe-Linux 5.2 (2.0.35)
Gnu CC 2.7.2.1 [-O5 -ffast-math -fcse-follow-jumps]	Gnu CC 2.7.2.1 [-O5 -ffast-math -frerun-cse-after-loop]	Gnu CC 2.7.2.1 [-O5 -ffast-math -frerun-cse-after-loop]	Gnu CC 2.7.2.1 [-O5 -ffast-math -frerun-cse-after-loop]
Gnu Fortran77 0.5.19.1 [-O5 -ffast-math -fcse-follow-jumps]	Gnu Fortran77 0.5.19.1 [-O5 -ffast-math -frerun-cse-after-loop]	Gnu Fortran77 0.5.19.1 [-O5 -ffast-math -frerun-cse-after-loop]	Gnu Fortran77 0.5.19.1 [-O5 -ffast-math -frerun-cse-after-loop]

AMD PC [AMD-K6]	SGI Octane [SGI]	Alpha Station 400 4/233 [A21064]	Alpha Personal Workstation 500 [A21164]
AMD K6 / 200MHz 32KB(I)+32KB(D) intern 512KB extern 128MB	MIPS R10000 195MHz, Chip Rev. 2.7 32KB(I)+32KB(D) intern 1MB extern 128MB	Alpha 21064 / 233MHz 16KB(I)+16KB(D) intern 512KB extern 96MB	Alpha 21164 / 500MHz 8KB(I)+8KB(D)+96(I+D) intern 2MB extern 128MB
SuSe-Linux 5.2 (2.0.35)	Irix 6.4	DEC Unix 4.0D	DEC Unix 4.0D
Gnu CC 2.7.2.1 [-O2 -ffast-math -unroll=4]	MIPSpro C Compiler 7.2.1.2m [-O3 -LNO:ou_max=4]	DEC C Compiler 5.6-071 [-O5 -fast -unroll 4]	DEC C Compiler 5.6-071 [-O4 -fast -unroll 4 -u]
Gnu Fortran77 0.5.19.1 [-O2 -ffast-math -unroll=4]	MIPSpro Fortran77 7.2.1.2m [-O3 -LNO:ou_max=4]	DEC Fortran77 4.0 [-O5 -fast -unroll 4]	DEC Fortran77 4.0 [-O4 -fast -unroll 4 -u]