専用並列計算機による「場の物理」の研究

（研究課題番号 08NP0101）

平成４年度～８年度 科学研究費補助金（創成的基礎研究費）

研究成果報告書

平成９年８月

研究代表者　岩崎洋一

（筑波大学 物理学系 教授）

まえがき

平成元年の学術審議会建議に基づき，平成２年度から「学術の新しい展開のためのプログラム」(略称新プログラム) が発足した．ここに研究成果を報告する「専用並列計算機による「場の物理」の研究」は，その実施テーマの一つとして採択され，平成４年度より５カ年計画で実施されたものである．また，同年４月には，本計画の推進母体として「計算物理学研究センター」が筑波大学に全国共同研究施設として設置された．

本計画の目的は，計算物理学研究に即した高性能超並列計算機 CP-PACS を，計算物理学と計算機工学にまたがる学際的共同研究により開発製作すると共に，それを用いて，素粒子物理学，宇宙物理学，物性物理学に於ける数値的研究を推進することにある．超並列計算機 CP-PACS の開発製作は，新プログラムの財源措置の柱の一つである科学研究費補助金（創成的基礎研究費）を受けて，筑波大学計算物理学研究センターに於いて進められた．

超並列計算機 CP-PACS の開発製作は，平成７年度に，演算要素数 1024 個，最高速度 307GFOPS の性能を持つシステムが完成し，続いて平成８年９月には，演算要素 2048 個，最高性能 614GFLOPS への倍増が行われて最終的な完成をみた．この超並列計算機を用いてリンパック・ベンチマークの性能測定を行い，実効速度として，368.2GFLOPS を達成した．この測定結果はこの時点での世界最高速度であり，米国ピッツバーグで開催された国際会議スーパコンピューティング９６で報告された「世界の高速計算機トップ 500」の第１位に登録された．

計算物理学の応用計算は，1024 個の演算要素から構成されるシステムが完成した時点より開始され，特に格子量子色力学のクエンチ近似でのハドロン質量の本格的な精密計算を行い, 当該分野での従来の結果を飛躍的に向上させた結果を得た．さらに，近似なしの計算の予備計算を実行した．これらの結果は，平成９年３月中旬に計算物理学研究センターで開催された国際会議で報告された．宇宙物理学，物性物理学の分野でも，プログラム開発，性能測定は終了し，予備的ではあるがすでに有望な結果を得ている．

プロジェクト終了にあたり，プロジェクトの全貌，超並列計算機 CP-PACS の詳細， 性能評価，CP-PACS を用いた計算物理学研究の結果を本報告書にまとめた．CP-PACS プロジェクトを幅広く知って頂ければ幸である．

CP-PACS プロジェクトを推進するにあたって，文部省，筑波大学を始め広範囲の関係者の強い支持を頂いた．ここに感謝の意を表したい．

研究代表者
筑波大学物理学系教授・計算物理学研究センター長
岩崎洋一

目 次

第 I 部

研究概要

1　研究目的

素粒子物理学，物性物理学，宇宙物理学の基本的重要問題の多くは，物理系が時空上の場によって記述される「場の物理」の問題である．本計画が構想された時点で，計算機の能力はすでに急速な進歩を見せており，これら「場の物理」の重要な問題において，解析的方法では解くことが困難な問題を数値計算により解決し，実験室では実現不可能な条件を数値実験で実現することが可能な状況が生まれつつあった．しかし，当時のスーパーコンピューターといえども，計算能力，メモリ容量がいまだ足りず，思うような計算ができない問題が数多く残されていた．このため，モデルを簡単化するか，近似を導入するか，計算精度が不十分でも我慢するか，研究者は各々工夫をせまられていた．

これらの問題は，従来は主としてベクトル型汎用スーパーコンピュータを用いて計算されてきたが，この種のスーパーコンピュータの計算能力は技術的限界に近付いていた．一方,「場の物理」の問題は並列計算機に非常に向いた問題である．実際,「場の物理」の問題に適した並列計算機のアーキテクチャを採用すれば，物理空間を計算機アレイに写像でき，近接相互作用の特徴を最大限に活用した高速計算が可能となる．さらに，この問題限定により，計算速度のみならずコストパーフォマンスも飛躍的に向上する．

物理研究の基本的手段として，ベクトル型スーパーコンピュータを凌駕する性能をもつ並列計算機を製作するアプローチは，昭和 62 年より 3 年間，科学研究費補助金（特別推進研究）を受け，筑波大学において行われた QCDPAX の開発によって第一歩が踏み出された．この開発においては，平成 2 年完成時点で，当時の最高速ベクトル型スーパーコンピュータの約 3 倍の速度を持つ並列計算機を製作することに成功し，それを用いて重要な物理結果を導いている．

本プロジェクトは，QCDPAX での経験を生かし,「場の物理」の計算に適した超高性能専用並列計算機 CP-PACS (Computational Physics by a Parallel Array Computer System) を新たに設計・製作し，それを用いて素粒子物理学，物性物理学，宇宙物理学の基本的重要問題に対して，第一原理に即した計算を行なうことを目的としたものである [1]．これにより，基本法則の検証，新しい質の現象に対する原理的な理解，新しい現象の予言などに関して，従来の計算力不足に起因する曖昧さなしに，明快な結論を導くことを目指し，物理学の重要な難問題に対する飛躍的な進歩を図ることを目標とした．

さらに，この専用並列計算機の研究開発を通じて，並列計算機工学の研究を発展させ，併せて，物理学の数値的方法による研究を推進する上で欠くことのできない物理学と計算機工学の共同研究の基盤を確立することも重要な目標であった．

2　本プロジェクトの背景

2.1　計算物理学

近年のコンピュータの能力の急速な進歩によって，科学技術のあらゆる研究分野で，研究方法に構造的な大きな変革が起きている．従来からの研究方法としては，実験及び観測装置を用いた実験的方法と，数学的な手法による理論的方法の 2 つが主要な方法であった．これら 2 つの方法

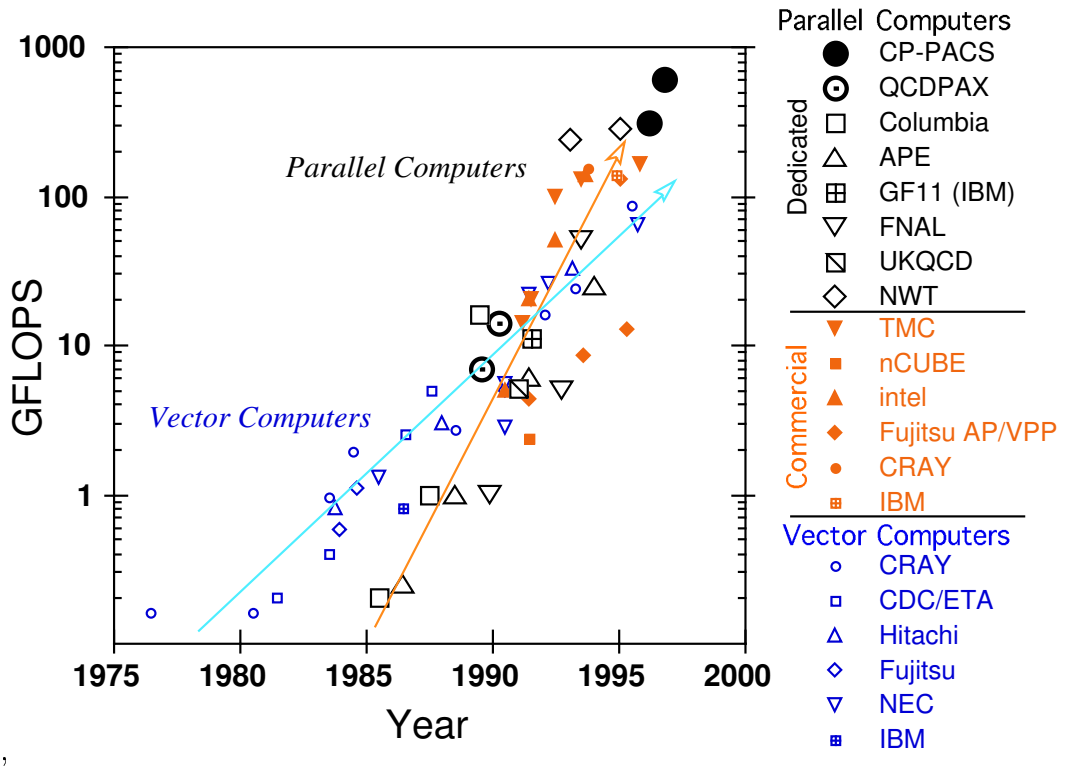


図 1: スーパーコンピュータと並列計算機の最高速度の発展：1 Gflops は 1 秒間に 10^9 回の浮動小数点演算速度.

と並び第 3 の方法として，コンピュータを用いた研究方法が非常に重要になってきている．いままで，技術的，時間的，価格的な困難のため実行出来なかった実験をコンピュータ上で実現し，数学的に解けなかった方程式をコンピュータで解くことが可能となってきた．物理学に於いても同様で，計算機を主たる研究手段として用いて研究する物理を計算物理学という [2].

例を素粒子物理にとれば，クォーク・グルーオンの基本法則と考えられている量子色力学 (QCD) は，本質的に，強結合・非線形理論であり，摂動論を用いて，陽子などハドロンの質量を計算することができない．電子・光子の基本法則である量子電磁気学 (QED) に対しては，摂動論が成功裡に用いられ，“紙と鉛筆” で電子の磁気能率双極子などが計算できたのと対照的である．ハドロンの質量を計算する唯一の方法は，量子色力学を格子上に定義した格子量子色力学を用いた計算機による数値的手法である．また，宇宙初期に於いて温度が約 1 兆度の時に，クォーク・グルーオン・プラズマ状態からハドロン状態へ相転移を起こしたと考えられている．実験室でクォーク・グルーオン・プラズマ状態を実現させるのは容易ではないが，計算機上にその状態を作り出すことが出来る．このように，“紙と鉛筆” で出来ない計算を計算機を用いて計算したり，実験室で実現困難な状態を計算機上に実現することにより，物理学の本質的な進歩が期待出来る．

量子色力学の場合，時間空間が 4 次元であり，1 格子点あたりの自由度も大きいために，計算速度，メモリ容量ともに，格段のものが要求される．スーパーコンピューターといえども満足のいく結果を得るには能力不足である．他の分野でも似た状況である．

さらに高性能なコンピュータをどうやって実現するかが問題である．コンピューターの能力は年々向上していくことが期待できるので，能力の優れたコンピューターが計算機会社によって開発提供されるのを待つのがひとつである．普通はこの方法を取る．一方，素粒子物理の実験の場合，加速器や測定器に出来合いの商品などはもちろんないので，自分たちで設計し製作を発注するこ

とが当たり前である．必要なら製作してしまおうという精神風土，基本法則から出発するのであるから近似は導入したくないという要求，これらが結び付いて，素粒子物理学の分野では，1980年代半ばから専用並列計算機を自ら製作するプロジェクトが，日本，米国，欧州でそれぞれ発足し，新しい流れとなってきた [3].

2.2　スーパーコンピュータから超並列計算機へ

従来は，最高性能の計算機は，いわゆるスーパーコンピュータといわれるベクトル型コンピュータであった．計算速度が 4，5 年に 10 倍になる割合で年々能力が向上してきていた．

しかし，図 1 に示すように，1991 年頃に，最高性能の機種がベクトル型コンピュータから超並列コンピュータに取って代わられた．上記の日本，米国，欧州の素粒子専用並列計算機，及び米国のシンキングマシン社などによる商用並列計算機の出現による．

このクロスオバー現象は，一方で，ベクトル型のコンピュータによる計算速度の高速化がすでに限界に近付いたこと，他方で，半導体技術の急速な進歩によって，CMOS 半導体チップが高速化，小型化，低価格化し，千台規模の超並列計算機が実装でき，全体の性能としてスーパーコンピュータの性能以上のものが製作できるようになったことによる．

ベクトル型のコンピュータの性能向上は，ECL 半導体による演算素子の計算速度を高速化することによって達成されてきた．マシンサイクルは 2 ns 程度である．これは，光が 60 cm 進む間に 1 演算を行なうスピードである．これに見あって，メモリからデータを読みだし，メモリに結果を書き込まなければならない．1 演算要素としてこれ以上格段にスピードを向上させるのは難しい．すでに，スーパーコンピュータでも，複数 (10 程度) の演算要素を並べ，同時に計算が行なえるようになっているし，最近は中央制御装置 (CPU) も複数個 (10 程度) 備えているものが当たり前になってきた．この並列性によって，性能向上を図っている．ECL 半導体は発熱量も多く，電力，冷却の問題で，これ以上大幅な並列化は困難である．

パソコン，ワークステーションなどに使われている CMOS 半導体チップは，ECL 半導体に比べると演算速度はいくらか遅いが，必要電力，発熱量は少なくすみ，千台程度備えたものを実装することが可能である．千台程度以上のものを超並列コンピュータと呼ぶ．これからは，最高性能の計算機は超並列コンピュータであることは衆目の一致するところとなった．

並列計算機のアーキテクチャには， SIMD か MIMD か，メモリ構成は分散か共有か，ネットワークのトポロジー，これらの千差万別の組合せがある．この多様な可能性の中からどれを採用するかが問題である．計算機にさせる仕事の種類は多く，科学技術計算から，事務処理までである．これらどの仕事にも，並列効率のよい並列計算機のアーキテクチャを選択するとなると問題は簡単ではない．しかし，仕事の種類を「場の物理」に限り，実際の応用プログラムで実効速度が充分に出るかを詰めていくことにより，並列計算機のアーキテクチャをしぼっていくことが出来る．この際，物理学の問題の詳細まで熟知している物理研究者と計算機のハードウエア・ソフトウエアの細部まで熟知している計算機工学研究者との共同研究によって初めて，ユーザーにとって使いよい，実効性能のでる並列計算機を製作することができる．

並列計算機の理論的最高速度は，一つの演算要素 (PU) の演算速度 × 台数である．従って，最高速度を向上させるためには，PU の演算速度を向上させ，台数を増やすことが必要である．ただし，実際の計算に於いては，PU 間でデータのやりとりをしなくてはならないので，実効速度を高めるためには，演算速度に見合ったデータ転送速度の向上が必須である．また，実装，価格，電力，信頼性の問題などから，台数にも上限があり，現在の技術では，数千台が限度であると考えられる．

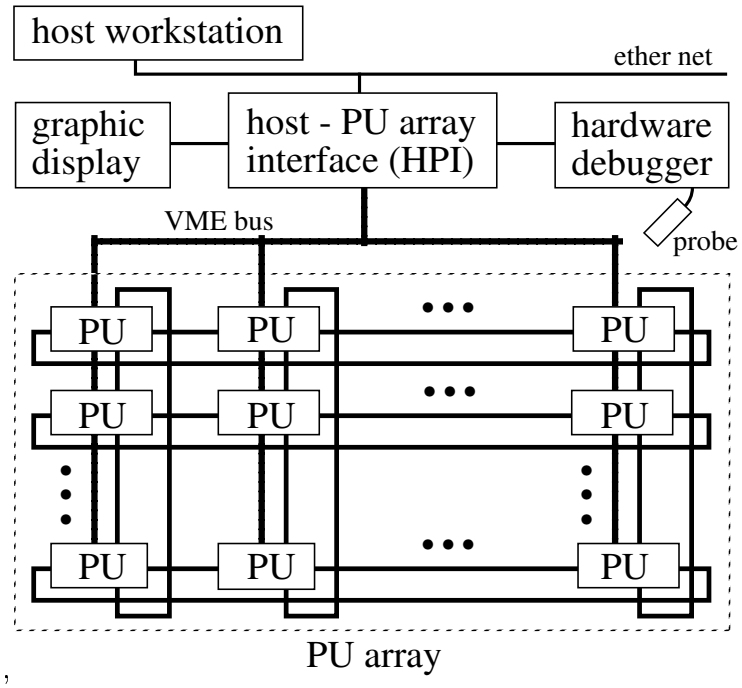


図 2: QCDPAX システム構成図

2.3 並列計算機 QCDPAX

並列計算機 QCDPAX [4] は，このような流れの中で，1987 年から 1989 年までの 3 年間に約 3 億円の科学研究費補助金 (特別推進研究) を受け開発製作された．筑波大学を中心とした素粒子物理学者と並列計算機工学者からなる QCDPAX グループで基本設計を行ない，アンリツ株式会社が製作した．演算要素 (PU) が 480 台から構成されており，2 次元トーラスのメッシュに結合されている．最高速度は約 14 GFLOPS であり，完成当時のスーパーコンピュータの 3 – 4 倍の性能である．それを用いて現在も素粒子物理学の研究を行なっている．完成後約 7 年間フル稼働しており，稼働時間の合計は 6 万時間を越えている．商用機に比べれば，故障も多いし，ソフトウエアも整備されていなくユーザーの苦労も多いが，格子 QCD を用いて，宇宙初期におけるクォーク・グルーオン状態からハドロン状態への相転移の精密計算，ハドロンの質量の計算，クォークの種類数と閉じ込めの関係などに関する結果を導き出している．

QCDPAX は，世界に先駆け 1977 年より筑波大学構造工学系星野力らによって開発が開始された，並列計算機 PAX シリーズの 5 代目に当たる [5]．PAX のアーキテクチャは MIMD ，局所分散メモリ型近接通信，2 次元メッシュトーラス構造であり，格子 QCD に適している．QCD のもつ特徴，局所性，近接性，境界条件が，局所分散メモリ型近接通信，トーラス構造に対応している．また 4 次元を 2 次元写像することは何の問題もなく，ハードウェアの作りやすさ，PU 間の通信の速さなど総合的に判断して，従来の PAX シリーズと同じ二次元トーラス構造を採用した．図 2 に QCDPAX の構成図を示す．

この基本的アーキテクチャに基づき，1 演算要素あたりの計算速度と，通信速度をそれまでの PAX シリーズより格段に速くして，その当時のスーパーコンピュータより高速の計算機を製作した．一台の PU の速度は約 30 MFLPOS であり，全体の PU 台数は 480 台であるので最高速度は約 14 GFLOPS となる．

今回の超並列計算機 CP-PACS の開発においては，QCDPAX の開発・製作過程での経験を生

かして,「場の物理」の計算に適し，かつ使用者にとって使い勝手のよい並列計算機システムの構築を目指した.

2.4 世界における専用並列計算機の開発状況

格子量子色力学のための専用並列計算機を開発するプロジェクトは，米国では，コロンビア大学，フェルミ国立加速器研究所 (ACPMAPS)，IBM のワトソン研究所 (GF11)，欧州ではイタリアのローマ大学を中心として (APE) 進展し，日本の筑波大学における QCDPAX と同程度の計算機が 1992 年前後に製作され，素粒子物理の研究に用いられてきた（表 1 の第 1 グループ）. さらに，日本に於ける CP-PACS プロジェクトに対応する計画として，米国ではコロンビア大学を中心とした QCDSP プロジェクト, 欧州ではイタリアのローマ大学を中心とした APE1000 プロジェクトがある（第 2 グループ）. これら 2 つの計画は現在も進行途上にあり，並列計算機の完成は，それぞれ平成 9 年末，平成 10 年内と予定されている.

表 1: 素粒子物理学分野における専用並列計算機のリスト [3]

プロジェクト	理論最高速度 GFLOPS	稼働（予定）年
Columbia 16	0.25	1985
64	1.0	87
256	16	89
APE 4	0.25	86
16	1.0	88
QCDPAX	14	90
GF11	11	91
ACPMAPS	5	91
APE100	$6 \rightarrow 25$	$92 \rightarrow 94$
ACPMAPS upgraded	50	93
CP-PACS	600	96
QCDSP	400	97
APE1000	~ 1000	99

3 研究計画の概要

本計画の実施にあたっては，(1) 「場の物理」の為の超高性能専用並列計算機 CP-PACS の設計・製作と，(2) それを用いた素粒子物理学，物性物理学，宇宙物理学の重要難問題の研究，の二段階の研究計画を立案した．それぞれの概要は以下のとおりである.

(1) 専用並列計算機の設計・製作に当たっては，物理学研究者と計算機工学者の緊密な共同研究を行ない，理論的ピークスピード 600 GFLOPS（ 1 秒間に 6 千億回の浮動小数点演算）の

計算性能の実現を図る．専用並列計算機として最も重要なことは理論的ピークスピードよりもむしろ「場の物理」の計算に対して実行スピードが充分高速なことである．この目的の為に，物理学研究者は，本研究で解決を図る物理学の諸問題に於いて重要な計算の典型的なプログラムを分析・提示する．計算機工学者は，技術的に実現可能な並列計算機の演算ユニット（演算素子，記憶素子等）の構成とその結合方法等についての検討を基に，これらの計算を超高速に処理するために最も適した並列計算機アーキテクチャを選択し，物理学研究者との討議の上，システム全体の基本設計を纏する．

この基本設計を基に，さらに詳細設計を行なう．詳細設計の各段階においては，設計された並列計算機の物理学の具体的問題に対する実効性能評価のために，典型的な問題の仮想ベンチマークテストを行ない，その結果に応じて，必要な場合は設計の変更を行なう．この詳細設計に基づき，実装設計を行ない，CP-PACS の組立，調整，機能検証，評価を行なった後，本格稼働に入る．

これらと並行して，システムソフトウエア，プログラミング言語，コンパイラの設計・開発，デバッグ機能の開発など，並列計算機を使用する上で重要なソフトウェアを開発・製作する．

(2) 製作された超並列計算機 CP-PACS を用いて，理論的方法と実験的方法では解決困難な「場の物理」の基本的な重要難問題の解決を図る．素粒子物理学に於いては，強い相互作用の基礎理論である量子色力学に基づく素粒子の諸性質の解明，宇宙初期の物質の状態に関する予言，又，ヒッグス粒子の質量等弱い相互作用に絡む未解決の問題の究明を目指し，さらに，物性物理学に於ける高温超伝導機構の解明，宇宙物理学に於ける銀河，星の形成過程の解明，重力崩壊とブラックホールの形成の解明等の研究を行う．

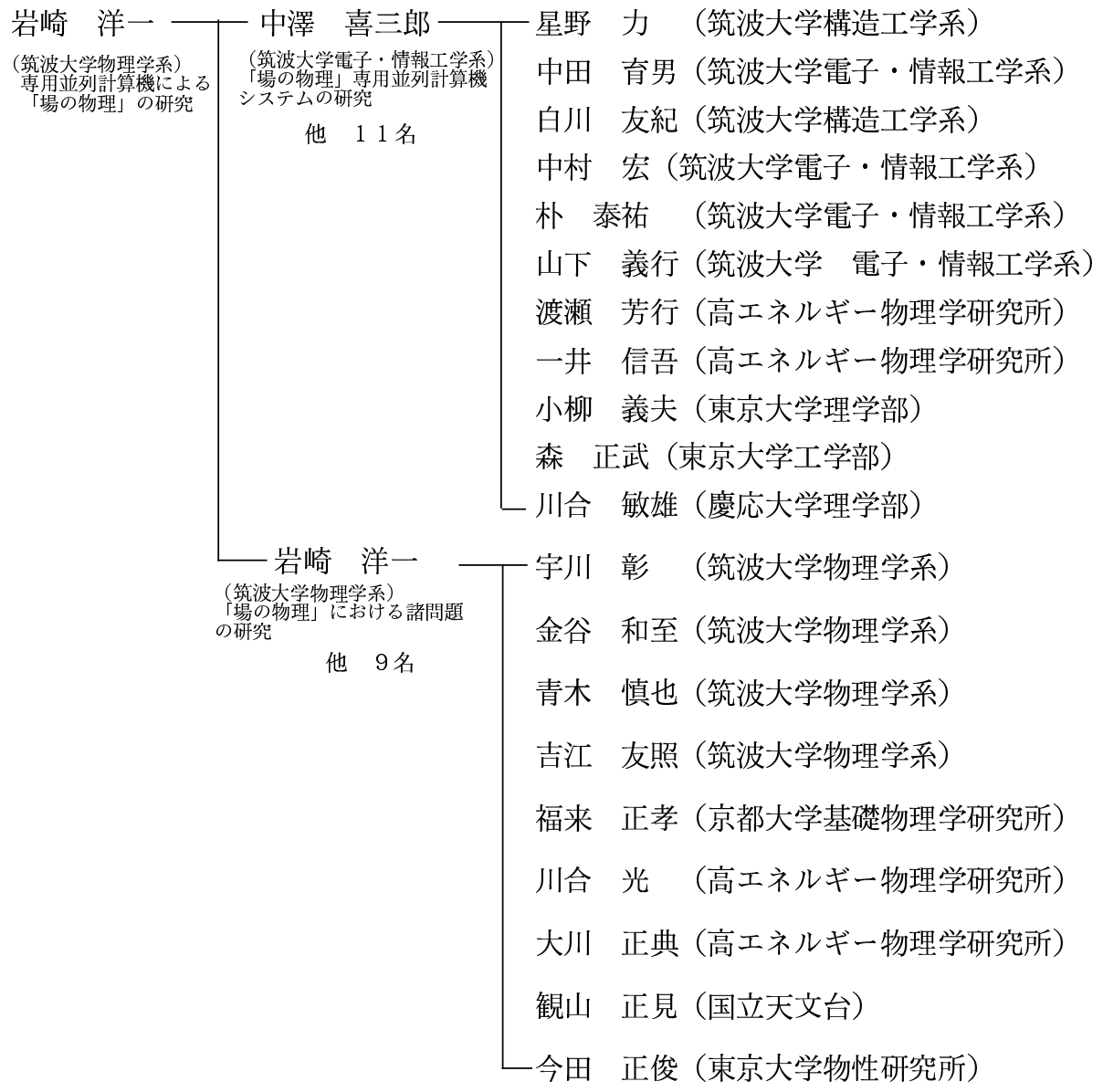
これらの問題について高い計算能力を実現するために，典型的な問題に対して必要とされる計算能力の検討を当初から行なって，超並列計算機 CP-PACS の設計にその結果を反映させると共に，その基本設計・詳細設計の各段階で仮想ベンチマークテストを行なう．ベンチマークテストの結果を参照しながら，並列アルゴリズムの研究と応用プログラムの開発を行い，また CP-PACS のプロトタイプを用いたテスト計算を実行して，応用プログラム・解析プログラムを改良充実する．CP-PACS の本格稼働時迄に以上の計算準備を充分に行ない，本格稼働と同時に物理計算の実行を開始して，格子量子色力学等,「場の物理」の重要問題に対して世界に先進的な結果を得ることを目指す．

4　研究組織

本計画を実施するための研究組織は，上記研究計画の概要の (1) と (2) に対応して,「研究班 1 :「場の物理」専用並列計算機システムの研究」 と「研究班 2 :「場の物理」に於ける諸問題の研究」 の 2 つの班から構成された．図 3 に，平成 4 年度プロジェクト発足当時のの組織構成図を，図 4 に最終年度平成 8 年度の組織構成図を示す．計画参加者数は，最初は，研究班 1 と 2 が各々 10 名，合計 20 名であったものが，最終段階では，各々 15 名，19 名，合計 34 名となった．

さらに，学術振興会特別研究員などの制度を活用して，研究活動を推進した．これら研究員のリストを表 2 に挙げる．

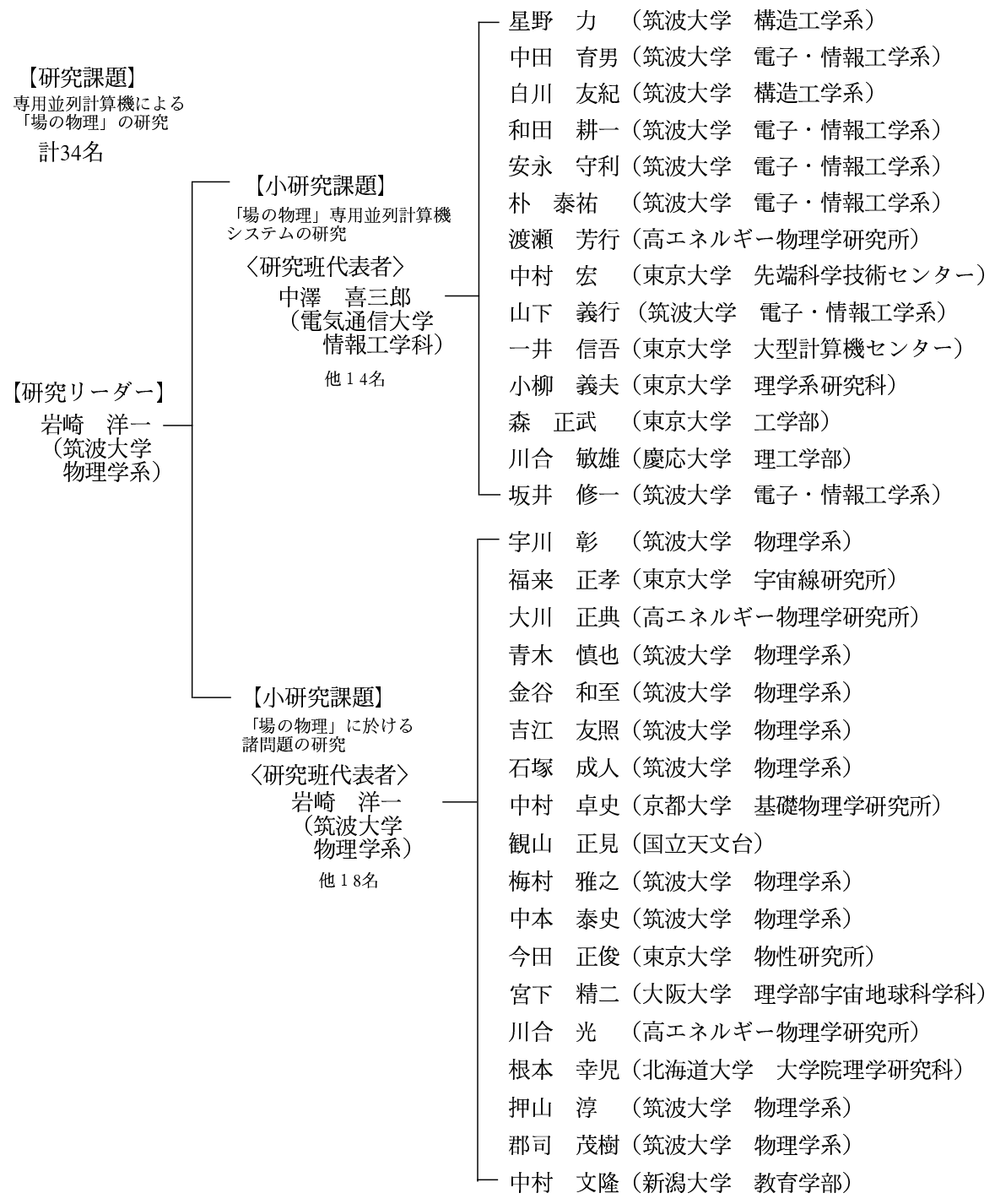
研究組織構成図（平成４年度）



,

図 3: 平成 4 年度研究組織機構図

研究組織構成図（平成８年度）



,

図 4: 平成 8 年度研究組織機構図

	研究員氏名	期間	現職（現住所）
日本学術振興会 特別研究員			
平成 4 年度	菊川芳夫	1992.4.1 - 1993.3.31	京都大学理学部助手
	石塚成人	1992.4.1 - 1993.3.31	筑波大学物理学系助手
平成 5 年度	蔵増嘉伸	1993.4.1 - 1993.3.31	KEK 理論部助手
平成 7 年度	中村文隆	1995.4.1 - 1996.3.31	
平成 8 年度	中村文隆	1996.4.1 - 1997.3.31	新潟大学教育学部助手
	櫃昌吾	1996.4.1 - 1997.3.31	KEK 理論部
日本学術振興会 外国人特別研究員			
平成 7 年度	Wolfgang Bock	1995.10.1 - 1996.9.30	Humbolt University
平成 8 年度	Rudolf Bulkhalter	1996.9.1 - 1997.8.31	Unversity of Tsukuba
	Graham Boyd	1996.11.1 - 1997.10.31	Unversity of Tsukuba
外国人研究員			
平成 7 年度	Philippe de Forcarnd	1996.3.1 - 1996.5.31	ETH，Switzerland
平成 8 年度	Paul B. Macknzie	1996.6.11 - 1996.9.10	Fermi Lab.,USA
	Sergei V. Zenkin	1997.1.16.11 - 1997.3.31	INR，Russia

表 2: 研究員リスト

5 研究経費

本計画の実施，特に超並列計算機 CP-PACS の開発製作は，新プログラムの財源措置の柱の一つである科学研究費補助金（創成的基礎研究費）を受けて，筑波大学計算物理学研究センターに於いて進められた．研究実施 5ヶ年間の研究経費内訳は次のとおりである．

平成 4 年度	30,000 千円
平成 5 年度	222,244 千円
平成 6 年度	625,000 千円
平成 7 年度	745,000 千円
平成 8 年度	580,000 千円
計	2,202,244 千円

表 3: 研究経費

6 計算物理学研究センター設置及びその整備

本計画の研究推進母体として，計算物理学研究センターが平成 4 年 4 月 10 日に全国共同利用施設として筑波大学に設置された．研究部門は計算素粒子物理学，計算物性物理学，計算宇宙物

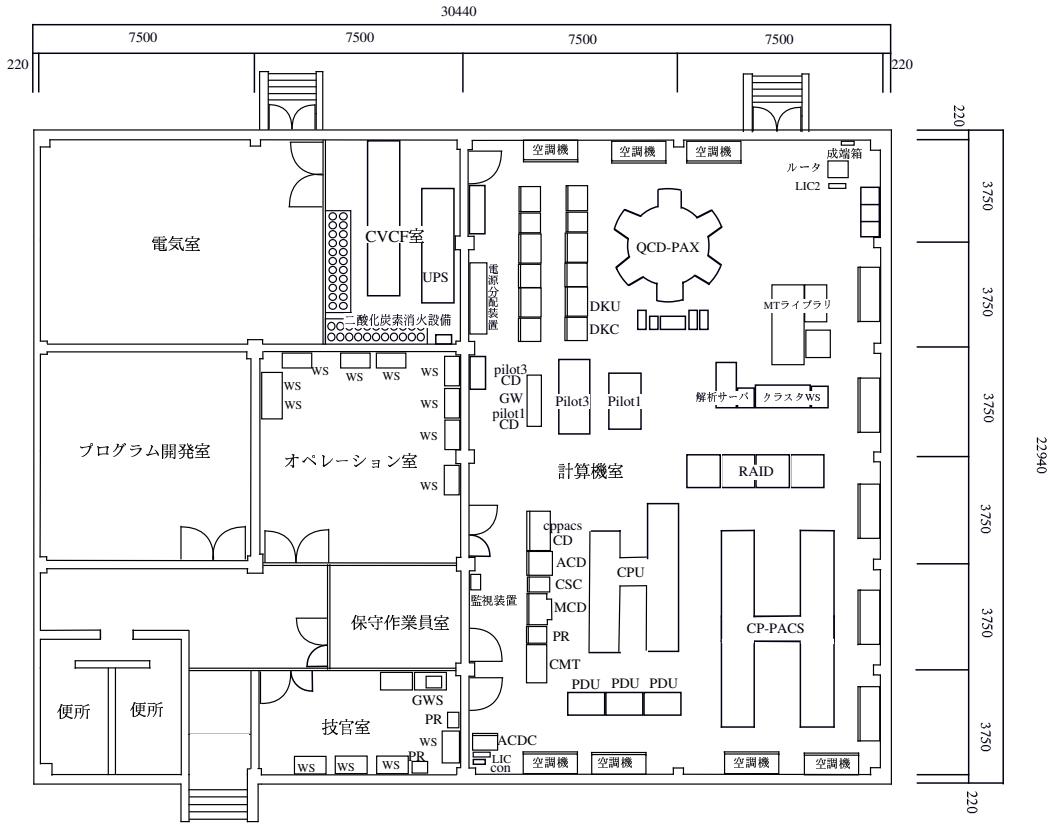
計算物理学研究センター主要日誌	
1992/4/10	計算物理学研究センター設置
7/3	計算物理学研究センター開所式
1993/8/26	センター棟第一期工事竣工
11/18	QCDPAX 移転
1994/3/1	FCS（フロント計算機システム）第一期レンタル開始
4/4 - 4/6	筑波大学物理学関連組織外部評価
8/3	「新プログラム研究」中間評価（現地調査）
1995/3/1	FCS（フロント計算機システム）第二期レンタル開始
5/30	センター棟第二期工事竣工
1996/3/25	CP-PACS(1024PU) 完成設置
5/21	CP-PACS(1024PU) 完成披露式
9/18	CP-PACS(2048PU) 完成設置
11/19	CP-PACS(2048PU) 完成記者会見

表 4: 計算物理学研究センター主要日誌

	平成 5 年度	平成 6 年度	平成 7 年度	全体
電源	KVA	KVA		KVA
計算機用	500	300		800
一般用	100			100
空調用	300	300		600
空調				
台数 (24,000Kcal)	4	2	6	12
無停電装置				
500KVA 5 分間			設置完了	

表 5: 建物付帯主要設備

筑波大学　計算物理学研究センター　計算機システム配置図



,

図 5: 計算物理学研究センター計算機棟平面図

理学及び並列計算機工学の 4 部門から構成される．定員は教官定員 10 人，客員教官 2 人である．

研究の実施にあたっては，プロジェクト参加メンバーをセンター共同研究員に委嘱し，センターを中心に研究活動を行った．センターは，文字通り，新プログラム研究の推進母体として中心的な役割を果たしたと言える．

発足以来のセンターの主要日誌を表 4 に示す．センター建物は計算機棟（図 5 参照）である第 1 期分（ 698 m^2 ）が平成 5 年夏竣工し，研究棟である第 2 期分（ 1070 m^2 ）が平成 6 年度竣工した（表 4 参照）．

建物と並んで重要なセンター設備は，CP-PACS を全国の共同研究者の使用に供するための計算機システムであり，センター経費を用いて構築された．この計算機システムは，図 6 に示したように，CP-PACS，フロント計算機システム，QCDPAX などから構成される．これら計算機システムのための電源，空調，無停電装置などの建物付帯設備も順次整備された（表 5 参照）．また，新規開発した CP-PACS が完成する以前より並列計算機 QCDPAX をセンターに移管し，有効に活用している．

フロント計算機システムは，CP-PACS でのユーザジョブを制御するためのフロントシステムホスト，計算結果保持のための外部記憶装置と磁気テープライブラリ，ワークステーション，プリンタ，グラフィクス系などから構成される．

フロント計算機システムは，第一期分を平成 6 年 3 月より稼働開始し，第 2 期分は，外部記憶装置の増強などを行って，平成 7 年 3 月よりレンタル開始した（表 4 参照）．平成８年３月に

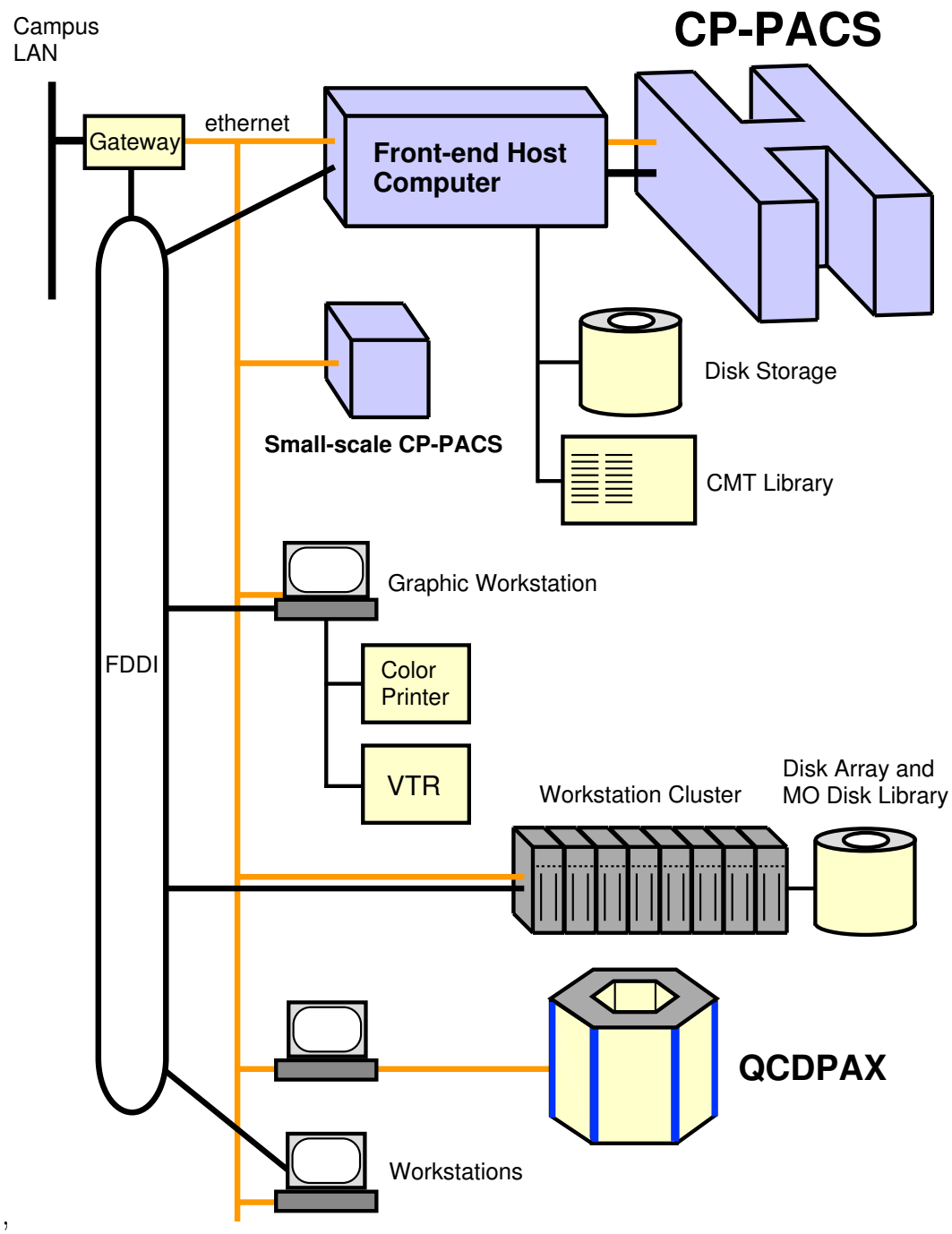


図 6: 計算物理学研究センター計算機システム構成図

1024 個の演算要素から構成される超並列計算機 CP-PACS が完成した時点から，CP-PACS はフロント計算機システムと接続されている．

7　研究推進方法

研究班は 2 つになっているが，物理学者と並列計算機工学者とが実質的に一つのプロジェクトに有機的に取り組んできた．このため研究経費は各班に分配することなく全体で用いた．二つの専門家グループが定期的に会合する場として，センター研究員会議を組織し，ここでの議論を中心に共同研究を進めた．物理学者と計算機工学者が実質的な共同研究を行うことは，応用プログラムに対して高性能かつ信頼性のある並列計算機を開発するには非常に有効である．この伝統は先の QCDPAX プロジェクトから引き継がれたものである．

最先端の技術を採用するためにはメーカーとの協力が欠かせない.「新プログラム研究」の実施テーマの一つとして，専用並列計算機による「場の物理」の研究が内定した時点から，内外の協力の可能性のあるメーカ 14 社に開発協力可能性の打診を始め (表 6 参照)，その回答を基に全体計画書を作成し，それに対する回答要望書を 14 社（内外国系 5 社）に発送し，最終的には平成 4 年度の超並列処理システムの基本設計に関して，一般競争入札によって日立製作所と契約を結んだ．それ以降，以下に記述する種々のワーキンググループを中心として，日立製作所と協力して超並列計算機 CP-PACS の開発にあたった．

超並列処理システム調達主要手続き・日程		
平成 3 年 4 月 2 日	開発協力可能性打診	14 社（内外国系 3 社）
平成 3 年 4 月 20 日	回答締め切り	12 社　回答
平成 4 年 2 月 20 日	全体計画書提出・回答要望書	14 社（内外国系 5 社）
平成 4 年 3 月 2 日	全体計画書説明会	5 社　出席
平成 4 年 4 月 23 日	全体計画書回答	12 社回答
平成 4 年 5 月 14 日	官報公示	
平成 4 年 6 月 12 日	開札・契約	基本設計の契約

表 6: 超並列処理システム調達主要手続き・日程

7.1　センター研究員会議及びワーキング・グループ

先に述べたように，プロジェクトのメンバーは最終段階で 34 名からなり，その中には，汎用大型機及びスーパーコンピュータの開発の中心的な役割をはたしてきた計算機工学者，世界に先駆け並列計算機システム PAX シリーズの開発を推進してきた計算機工学者，並列計算機 QCDPAX を用いて素粒子物理の研究を行なってきたグループ，スーパーコンピュータを用いて研究しているグループなどを擁している．各々がこれまでの経験を生かし，実効速度が速く，ユーザーに使い易く，信頼性の高いシステムを開発製作すべく，センター研究員会議，また日立製作所のメンバーを加えた種々のワーキング・グループなどの議論を中心として，色々な角度から検討を加え，CP-PACS の開発に取り組んだ．

センター研究員会議は 2 週間に 1 度程度，超並列計算機 CP-PACS のハードウエア及びソフ

研究員会議									
平成 4 年度	4/10	4/18	5/9	5/23	6/8	6/20	7/13	7/22	9/5
	10/1	10/24	11/7	12/5	1/9	1/21	2/6	2/18	3/6
	3/22								
平成 5 年度	4/3	4/22	5/8	5/20	6/5	6/17	7/3	7/15	8/5
	8/26	9/16	10/1	10/22	11/6	11/18	12/4	12/16	1/17
	2/5	2/17	3/5	3/17					
平成 6 年度	4/14	5/7	5/24	6/4	6/16	7/2	7/14	9/3	9/16
	10/15	11/5	11/17	12/3	12/15	1/19	2/4	2/16	3/3
	3/20								
平成 7 年度	4/20	5/18	6/3	6/15	7/1	9/2	9/21	10/7	10/19
	11/9	11/22	12/9	12/26	1/18	2/3	2/16	3/9	3/21
平成 8 年度	4/19	5/11	5/24	6/15	7/6	7/22	8/23	9/7	9/20
	10/12	11/2	12/7	12/ 26	1/11	2/1	3/1		
ハードウェア WG									
平成 4 年度	7/9	7/23	8/6	9/5	9/19	10/5	10/27	11/30	12/22
	1/14	1/28	3/1						
平成 5 年度	4/2	5/7	6/17	8/4	9/24	10/28	12/6	1/28	2/28
平成 6 年度	4/11	5/12	6/6	6/17					
ソフトウェア WG									
平成 4 年度	11/6	12/3	1/21	2/23					
平成 5 年度	4/28	5/31	7/5	8/4	8/31	9/16	10/22	12/16	2/7
	3/10								
平成 7 年度	4/14	5/26	7/1						
ハードウェア ソフトウエア合同 WG									
平成 6 年度	7/26	8/30	10/14	11/24	12/27	1/31			
平成 7 年度	4/20	5/22	6/20	7/27	8/30	9/28	10/26	11/21	12/27
	1/26								
平成 8 年度	5/29	6/28	7/19	8/23	10/18	11/22	12/20	1/24	2/20
コンパイラーサブ WG									
平成 6 年度	11/4	12/2	2/3						
平成 7 年度	4/7	5/26	7/21	9/19	11/8	12/25	1/19		
平成 8 年度	4/5	8/7	10/4	12/13	1/17	3/11			
ベンチマークサブ WG									
平成 7 年度	1/11								
平成 8 年度	6/7	8/7	11/11						

表 7: 研究員会議，WG 開催日程

	研究会名	期間
平成 4 年度	超並列計算機による計算物理学	1993.3.16
平成 5 年度	21 世紀への宇宙物理学シンポジウム：天体物理学における輻射輸送過程	1994.3.1 - 3.3
平成 6 年度	並列計算機と計算物理学	1994.12.6 - 12.7
平成 7 年度	滞在型ワークショップ「銀河形成」	1995.12.1 - 12.6
	CCP Workshop on Lattice Filed Theories '96	1996.3.5 - 3.7
平成 8 年度	「クェーサー活動性と銀河形成の物理学的関連」ワークショップ	1997.3.5 - 3.7
	International Workshop "Lattice QCD on Parallel Computers"	1997.3.10 - 3.15
	「並列計算機による物理学」	1997.3.26 - 3.27

表 8: 計算物理学研究センター主催研究会リスト

トウエアのワーキング・グループ (WG) は始めの段階では別々に各々 1 ヶ月に 1 度程度開催した．平成 6 年度の途中より，合同して行う方が有効である段階に差し掛かったとの判断より，以後ハードウエア及びソフトウエアの合同ワーキング・グループを開催して CP-PACS を開発した．センター研究員会議及びワーキング・グループの開催日を表 7 に示した．研究員会議では，超並列計算機 CP-PACS の開発方針，ハードウエア WG 及びソフトウエア WG の報告，CP-PACS のフロント・計算機システムの構成などを議題として通常 2 〜 3 時間程度討論・報告が行なわれた．ハードウエア WG， ソフトウエア WG 及び合同 WG では，CP-PACS の開発に関する方針から技術的な細部にまでわたる議論を毎回 5 〜 6 時間程度かけて行なった．さらに適宜，コンパイラーの問題，ベンチマークの問題のためなどの，サブワーキング・グループを構成し，詳細な議論を重ね研究を進めた．

7.2 研究会開催

表 8 に掲げたように,「新プログラム研究」の進捗状況に応じて，適宜筑波大学計算物理学研究センター主催の研究会を開催した．

初年度， 3 年度，最終年度には，プロジェクト全体及び細部に渡る進捗状況の公開を目的として，国内研究会を開催した．また，センターで行っている宇宙物理学，素粒子物理学の研究分野をテーマに，密度の濃い研究会を各々 2 回， 1 回行った．これらの研究会ほとんど全ての報告集をセンターの研究報告集としてまとめた．

さらに，最終年度には，International Workshop "Lattice QCD on Parallel Computers" を開催した．格子量子色力学 (Lattice QCD) は，CP-PACS の応用の最重要課題であり，本シンポジウムは当該分野をテーマに，国内外の指導的な研究者をほぼ網羅する陣容の参加者を得て開催された（国外 26 人，国内 22 名）．この分野での主要な物理テーマに関する国内外の最新の成果報告と，それに基づく研究の現状の纏め及び将来の方向付けの討論が行われた．特に，CP-PACS を用いた計算結果の最初の報告が行われた．従来にない規模と精度のクエンチ近似でのハドロン質量の結果が，実験値からのずれを明確に示しているが報告された．これはここ 10 年来の懸案を解決する結果である．また，近似なしの計算の試みの結果が報告され，作用の改善によって，近

似なし計算の現実的な可能性が指摘された．これらの物理結果の他に，CP-PACS のアーキテクチャの詳細と実効速度について報告された．これらを通じて，CP-PACS の詳細と現在までの成果を当該分野国内外研究者に広く公開することができた．プロシーディングズは Nuclear Physics B(Proceedings Supplement) の 1 巻として刊行される．

8 研究実施年次経過

超並列計算機 CP-PACS のハードウエア，ソフトウエア開発を中心とした研究開発年次概要は図 7 に示したとおりである．年次毎の詳細を以下に述べる．

8.1 平成 4 年度

「場の物理」の計算に適した専用並列計算機 CP-PACS の基本設計を行った．この目的のために，物理学研究者は，本プロジェクトで解決を図る物理学の諸問題において重要な計算の典型的なプログラムを分析・提示した．計算機工学者は，実現可能な並列計算機の演算ユニット（演算素子，記憶素子等）の構成とその結合方法などについての検討を基に，これらの計算を超高速に処理するために最も適した並列計算機アーキーテクチャを選択し，物理学研究者との討議の上，システム全体の基本設計を纏めた．さらにこれを基に，プロジェクトメンバーと製作メーカーとの合同ワーキンググループにおいて検討を重ね，基本設計の最終案を作成した．具体的な点として特記すべき点は以下のとおりである．

1) PVP-SW 機構による汎用 RISC チップの機能強化．大型科学計算に於ては，汎用 RISC チップのキャッシュが有効に機能せず，最高性能の 1/10 程度の能力しか発揮出来ない．これを解決するために，汎用 RISC チップに擬似ベクトル機能を持たせる PVP-SW (Pseudo Vecor Processor based on Slide Window Registers) 機構を考案し，これによって演算機能を強化することにした．

2) ネットワークとして 3 次元ハイパークロスバーの採用．性能，価格，実装面から，種々のデータ通信ネットワーク構成を比較検討することにより，3 次元ハイパークロスバーが最適との結論に達した．

3) 機能強化 RISC チップのコンパイラー開発の基本方針の決定．上記のような新方式の並列計算機向きのプログラムのコンパイラー・アルゴリズムについても成案を得た．

4) 物理学の応用例に対する実効速度の高速化．格子 QCD のプログラムの中で一番重要な部分に対して，仮想ベンチマークを行ない，上記 1) の機能強化が必要なことが明らかになった．さらに，上記 2) ，3) の成果の結果，実効速度が最高速度の 70 %程度まで向上させ得るとの結論を得た．これは，並列処理システムとしては特記すべきことである．

5) 最高速度 600 FLOPS 実現可能．上記の点を含め，デバッグ，メインテナンス，実装など諸々の点を考察し，最高速度 600 GFLOPS の性能をもつ並列計算機は現状の汎用大型機の上位機種と同程度の大きさ，電力で技術的に製作可能であるとの結論に達した．

8.2 平成 5 年度

基本設計を基に，超並列計算機 CP-PACS の詳細設計を行った．

ハードウエア面では，ノードプロセッサ，結合ネットワーク，システム制御，入出力制御の各部の論理設計を行った．各部の構成素子（演算素子，記憶素子等）の性能，構成方式等の技術的詳細の選択決定にあたっては，それらに基づく並列計算機が「場の物理」の研究に必要な計算性

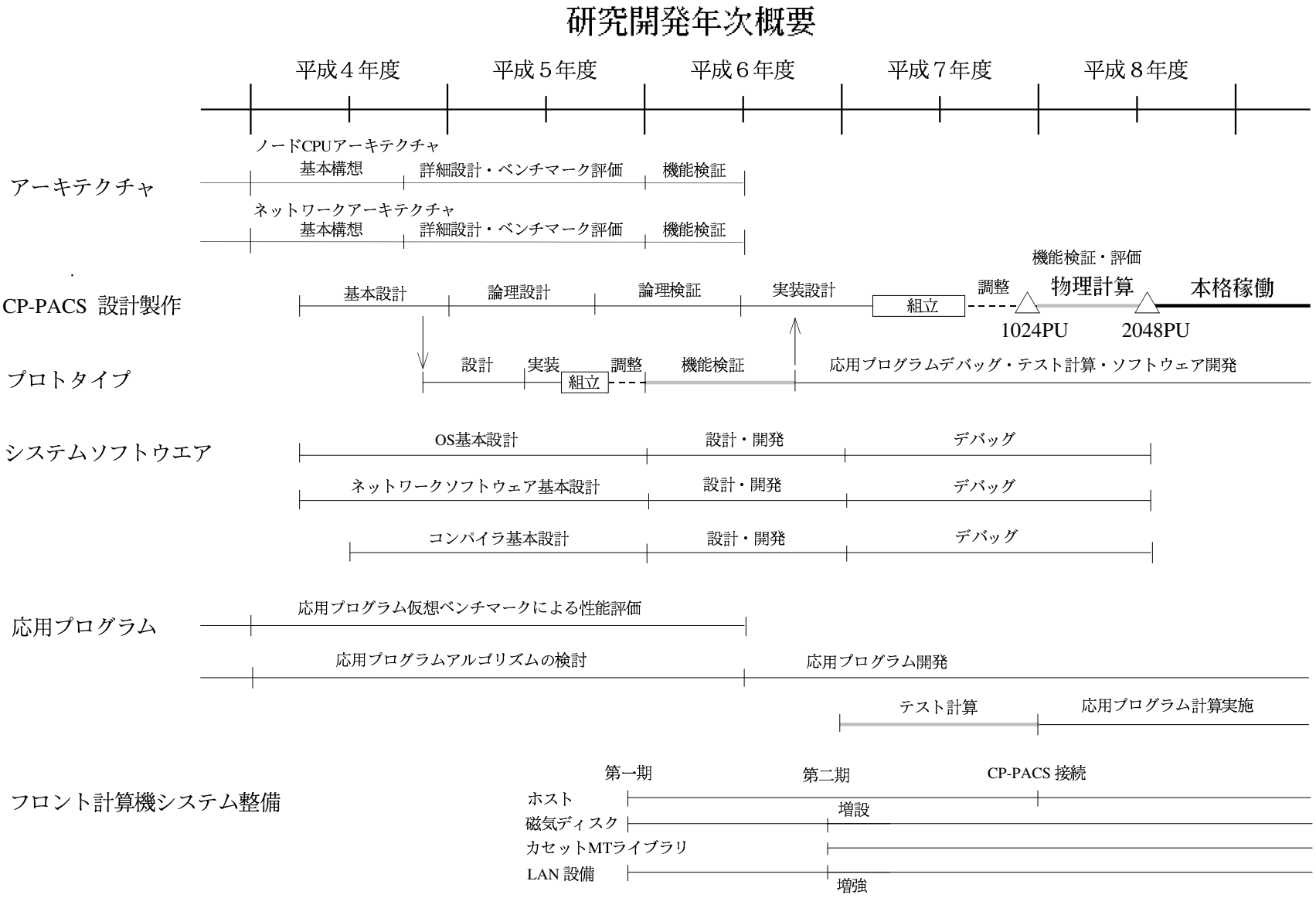


図 7: 研究開発年次計画概要

能を満たすかどうかを，シミュレータによる擬似実行を含む詳細な検討を行い，必要な場合は設計の変更を加えつつ最終案を作成した．さらに，これらの詳細設計の機能動作確認，特にノード間通信性能の確認用に，16 台構成のプロトタイプ (Pilot-1) の部分試作を行った．

ソフトウエア面では，並列処理システム用システムプログラムの基本設計を開始した．特に，ノード間通信の速度は超並列計算機全体の性能の鍵の一つであり，そのためのシステムプログラムについては，その詳細を決定した．また，ノードプロセッサの擬似ベクトル機能に対応するプログラミングアルゴリズムの開発，それに対するコンパイラーの基本設計を推進した．

物理学研究者は，詳細設計の各段階において，設計された超並列計算機の物理学の具体的問題に対する実効速度評価のために，種々の問題の仮想ベンチマークを行った．また，その結果を基に，各種の問題に対して最適なアルゴリズムを検討した．

8.3 平成 6 年度

最高速度 600 GFLOPS を目標に超並列計算機のハードウエア及びソフトウエアの開発を継続した．これと共に，物理学計算の基本パラメターの選択に関する検討を行った．また，既存の計算機を用いて，プログラム開発，計算アルゴリズムの改良を行った．具体的な内容は以下のとおりである．

1) 実装設計を加味した超並列計算機の詳細設計の確定

・主記憶の構成と制御方式：主記憶には DRAM 素子を用いた 16 ウェイのマルチバンク構成を採用することにした．これにより，ノードプロセッサの擬似ベクトル処理機構を最大限活用することができる．

・キャッシュメモリの制御方式：キャッシュメモリはストアスルー方式を採用することとした．この方式とすることにより，キャッシュメモリと主記憶との間のデータの無矛盾性維持が容易となる．

・相互結合ネットワークにおける同期方式：同期操作を行なう制御線を別に用意することで，データ転送用のネットワークを活用して同期操作を実現する方式を採用することとした．

・相互結合ネットワークにおけるデータ転送方式：高速通信機構の転送オーバヘッドを極力抑える制御方式を採用した．

・分割運転時の制御方式：分割形態について，分割数と各分割部の大きさを決定した．また，分割運転時のデータ転送と同期操作についての制御方式の検討も行なった．

・並列プロセッサ全体の実装方式：8 PU (Processing Unit) を 1 つの基板に搭載することにより，分散ディスクとコントローラを含め 1024 PU 全体を 8 筐体程度で実装する方式を採用する方向とした．

２）システムソフトウエアの開発

・並列ファイル入出力方式：ファイル入出力を並列実行できるように，複数のノードへディスク装置を分散させ，複数のプロセスにより複数のファイルへ同時にアクセスする方式を採用した．

・最適化コンパイラのアルゴリズム検討と試作：ノードプロセッサが持つ擬似ベクトル処理機構を最大限活用するための最適化コンパイラのアルゴリズムについて検討した．また，そのコンパイラの雛形の試作を行ない，その有効性を確めた．

・基本ソフトウエアの開発：OS カーネル，言語処理系，ネットワークデータ転送用ライブラリなどの設計仕様をがほぼ固ため，一部はパイロットモデル機での部分的実用評価と合わせて，開発を進めた．プログラミング言語として，Fortran90 と C 言語の使用を決定した．また HPF (High Performance Fortran) についても，HPF から Fortran77 へのトランスレータを用意する予

定とした．

・デバッガ及び性能モニタ：デバッガ，および性能向上のための指標を与える性能モニタの仕様を検討した．

3) シミュレータの開発と性能評価

・シミュレータの開発：ノードプロセッサの擬似ベクトル処理機構，キャッシュと主記憶の構成および制御方式をシミュレートするシミュレータを開発した．

・ノードプロセッサの実効性能：シミュレータを用いて量子色力学計算における主要サブルーチンを実行した時の性能を評価した．1 ノードプロセッサで実行した時，理論的なピーク性能 (300 MFLOPS) の約 65 %以上の性能を達成することが確認出来た．

・パイロットモデルの基本性能評価：既に設置稼働しているパイロットモデル (Pilot-1) の基本性能を評価した．特に，ネットワーク転送のオーバヘッドとスループットに関する基本データを採取し，問題点を整理した．

4) 論理設計・実装設計及び部品の試作

ノードプロセッサ及び主記憶系，並列ネットワーク部，システム制御部・入出力部，各々の論理設計及び実装設計を行い，さらに超並列計算機 CP-PACS を構成するノードプロセッサ用 VLSI，ネットワークボード，入出力系ボード，メモリ系サブユニット部，演算ユニットボード，システム制御ユニット，各々の部品の試作を行った．

5) 物理学研究のためのソフトウエア開発

シミュレーターを用いて物理学計算の仮想ベンチマークテストを行い実効性能を評価すると共に，計算の基本パラメターの選択に関する検討を行った．また，既存の計算機を用いて，計算プログラム開発を行い準備計算を遂行した．その結果を基に，プログラムの検討，計算アルゴリズムの改良を行った．

8.4 平成 7 年度

平成 6 年度までの研究成果に基づき，最高速度 600GFLOPS を目標に超並列計算機のハードウエア及びソフトウエアの開発最終段階を推進した．その結果，平成 8 年 3 月に，1024 PU (Processing Unit) 構成の CP-PACS の製作・組立を完了し，計算物理学研究センターに設置した．これと共に，物理学計算のプログラム開発を行った．詳しくは以下のとおりである．

1) パイロットモデルの問題点の整理：既に設置稼働しているパイロットモデル (Pilot-1) 上で種々のプログラムを実行し，問題点を整理した．

2) CP-PACS の実装，データ転送，運用等の検討

・CP-PACS システム全体実装：1024 PU 構成の場合，及び，2048 PU 構成に拡張した場合の CP-PACS の実装構造，設置面積，消費電力，発熱量について検討した．現在の計算機室に，2048 PU 構成の CP-PACS が設置可能であることを確認した．

・リモート DMA 転送のインタフェース：PU 間のデータ転送を高速に行うためのリモート DMA 転送機構を，Fortran 及び C 言語から呼び出して使用するための手続きの仕様について詳細を検討した．CP-PACS が持つ分散ディスクの入出力能力について検討した．ディスク制御の改良により，典型的な QCD 計算において，分散ディスク入出力に要する時間は悪目に見積もっても全体の実行時間の 20 %弱程度となる見込みが立った．

・CP-PACS 外部入出力：CP-PACS の分散ディスクとフロント計算機システムの外部補助記憶装置との間の，HIPPI チャネルを介した入出力に関して検討した．その結果，当初目標としていた 50MB/sec のスループットを確保できる見通しがついた．

・ CP-PACS の運用：CP-PACS の運用について，ジョブのクラスと投入方法，ユーザファイルの運用方法，セキュリティを中心に検討し，一応の成案を得た．

・パフォーマンスモニタ：CP-PACS の稼働状況をリアルタイムでワークステーションに表示するパフォーマンスモニタについて検討した．

3) デバッグ用試作機を用いた性能評価

CP-PACS の試作機などを用いて，QCD 計算の主要部の性能予備検討を行った．これまで予想していた性能とほぼ同じ性能を達成できることがわかった．

4) CP-PACS 開発

・ハードウェア関係：擬似ベクトル処理機構を持ったプロセッサ用 VLSI チップおよび，周辺の関連した VLSI チップと一緒に実装したセラミックモジュール，これらを搭載するボード，筐体，電源などの実装設計の製造，テストを終了した後，平成 8 年 3 月に CP-PACS 本体 (1024 PU + 64 IOU) を計算物理学研究センターに設置，システムの確認を行った．

・ソフトウェア関係：言語処理系，OS カーネル，ネットワーク通信用ライブラリを設計・開発した．CP-PACS の持つ処理能力を引き出すために必要不可欠な，擬似ベクトル処理機構を持ったプロセッサ用の最適化コンパイラ，リモート DMA 転送を実現するライブラリ，分散ディスクおよび外部補助記憶装置の入出力制御の開発も行い，これらは CP-PACS に実装された．これと並行し物理学計算のプログラムの開発を行った．

8.5 平成 8 年度

平成 7 年度に製作・組立を完了した 1024 個の演算要素から構成され，最高速度 307 GLFOPS の超並列計算機 CP-PACS の調整・性能評価を行った．それに続いて，物理学，特に格子量子色力学の本格的なクエンチ近似でのハドロン質量の精密計算を開始した．さらに，そこで得られた知見及びバグ対策経験を基に，平成 8 年 8 月から，演算要素を 2048 個，最高性能を 614 GFLOPS に倍増する作業を開始し， 9 月に組み立て設置が完了した．この超並列計算機を用いてリンパック・ベンチマークの性能測定を行い，実効速度として，368.2 GFLOPS を達成した．この測定結果は米国ピッツバーグで開催された国際会議スーパコンピューティング 96 で報告されたトップ 500 の第 1 位に登録された．

10 月よりは，この 2048 個の演算要素から構成される超並列計算機を用いて物理学研究のための稼働を開始した．格子量子色力学のクエンチ近似を呼ばれる近似の中での精密計算をさらに実行した．さらに，近似なしの計算の予備計算を実行した．これらの結果は，平成 9 年 3 月中旬に計算物理学研究センターで開催された国際会議で報告され，当該分野での従来の結果を飛躍的に向上させた結果として非常な注目を浴びた．宇宙物理学，物性物理学の分野でも，プログラム開発，性能測定は終了し，予備的ではあるがすでに有望な結果を得ている．

CP-PACS の調整及び性能評価に関わる個々の詳しい点は以下のとおりである．

超並列計算機 CP-PACS の調整・性能向上として以下の項目を実行した．1) 演算プロセッサ：電圧・周波数マージンの不足チップの交換．2) ノード間ネットワーク：一斉データ転送時のＤＣ電源電圧の変動対策．3) NIA （ネットワーク・インターフェース・アダプター）：電圧・周波数マージンの不足チップの交換．4) 分散ディスク：入出力性能向上のためコトールウエアの改良．5) オペレーティング・システム：OS のメモリ常駐使用量の削減，フラグメンテーション対策．6) コンパイラー：擬似ベクトル化適用範囲拡大・性能向上．7) 通信関数：既存通信関数の性能評価・性能向上，新規通信関数（ソフトウエア・ブロードキャスト関数・ RDMA チェイン関数）の実装．

計算機の環境条件として，空調機・電源設備に関して，床下空気流の吹き出し・室内温湿度の

調整，空調機負荷分散を実施，負荷合計約 420 KVA を確認した．

システム障害発生時，及び停電・火災等の障害発生時の対応体制などの技術的問題を検討，対応策を定めた．

上記の諸問題の解決を図るのと並行して，超並列計算機 CP-PACS の種々の性能測定を行い，最終的に設計とおりの性能が発揮されることを確認した．1) 演算プロセッサ：基本ベクトル演算の大規模計算に対して，擬似ベクトル機構によって高性能が実測された．特に，キャッシュモードと比較し，ベクトル長が長い時に，決定的に性能差が出ることを測定した．2) ネットワーク性能：ピーク性能が 1 秒間に 300 MBytes であり，立ち上がりのハードウエア・ソフトウエアのオーバーヘッドは約 3 マイクロセカンドという高性能を実測した．

これらの基本性能とともに，前述のとおり，性能測定の世界的な標準となっているリンパック・ベンチマークの実効性能として，368.2 GFLOPS を達成した．さらに，格子量子色力学のプログラムを開発し，その性能を測定した．核となる部分はアセンブラコードを用いてプログラムを書き， 1 プロセッサあたり，191 MFLOPS という高性能を実現した．

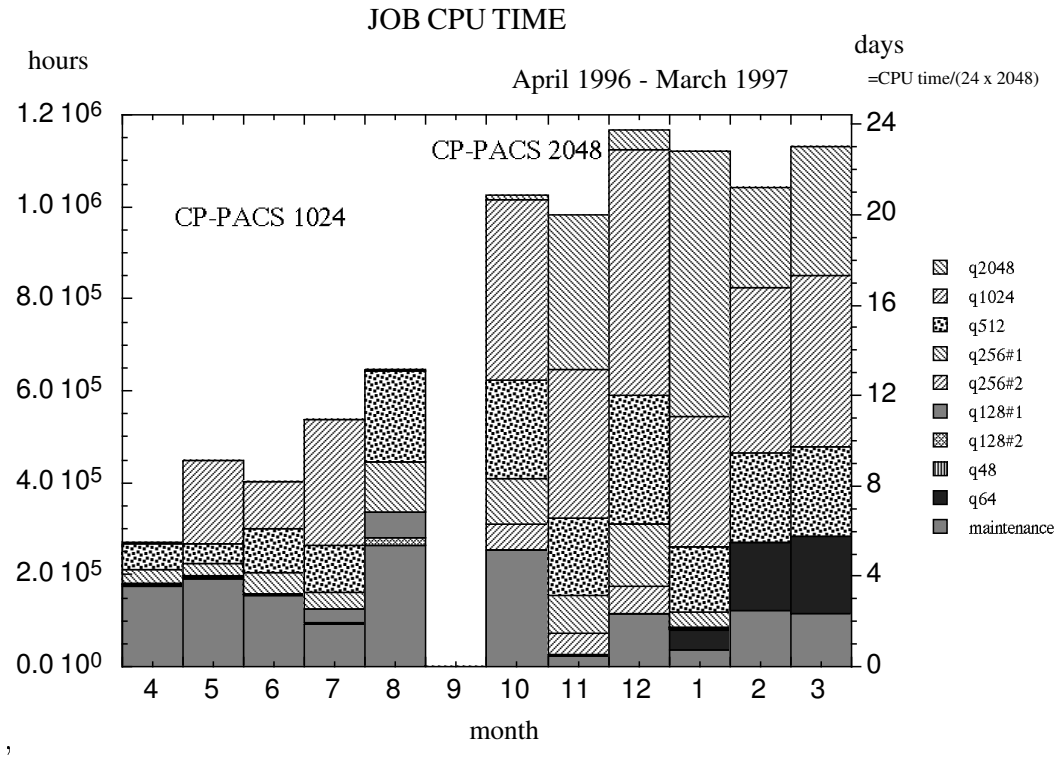


図 8: CP-PACS 稼働状況

9 CP-PACS の稼働状況

1024 個の演算要素から構成される超並列計算機 CP-PACS の製作・組立が完了し，センターに平成 8 年 3 月に設置され 4 月から稼働を開始してからの稼働状況を図 8 に示す．ユーザジョブに使われた時間の演算要素全体にわたる積算時間を，演算要素の数により分別されるジョブクラスにわけ，月ごとに示したものである．9 月分が空白となっているのは 8 月下旬から 9 月にかけて，2048 個の演算要素から構成される CP-PACS への増強作業が行われたためである．メインテナンスに対応する時間帯が始めのうちはかなりの割合を占めていたが，11 月以降は物理計算のた

めにほとんどの時間が使われている．

10　今後の目標

本プロジェクトでは，計算物理学の大きな部分を占める「場の物理」の研究の飛躍的な推進を図るため，当該分野での計算に対し高い効率を発揮しうる超高性能並列計算機 CP-PACS の開発を第一の目標とした．このようなシステムの実現には，プロセッサ単体演算能力の抜本的な強化，ネットワークの柔軟性と高性能の確保，高速 I/O 性能の実現等，超並列計算機を構成する基本部分それぞれにおいて，高度の研究開発が必要であった．本プロジェクト当初から追求した物理学研究者と計算機工学研究者の緊密な共同研究体制，合わせて計算機メーカの協力により，これらの難点を一つずつ解決し，計画最終年度には予定どおりに開発・製作を完了し，物理応用計算のために稼働を開始することができた．

本プロジェクトの第二の目標である，CP-PACS による計算物理学の研究推進ついては，CP-PACS のハードウエアと並行し開発された，コンパイラー，応用プログラムなどのソフトウエアの順調な仕上がりによって， CP-PACS 完成時には直ちに大規模計算を開始することが可能であった．このことにより，プロジェクト終了時点で，すでに約一年間に及ぶ大規模計算の蓄積が行われ，特に素粒子物理学の格子 QCD 計算に関して，従来の結果を飛躍的に発展させる結果を得ている．

CP-PACS は，物理応用計算の実行開始後も適宜行われた調整を経て，極めて安定な稼働状況を示して来ている．計算物理学の面からは，今後数年間は，CP-PACS の計算力を最大限に活用して，格子 QCD をはじめとする素粒子物理学分野での研究，また宇宙物理学，物性物理学分野での重要問題の研究を幅広く且つ強力に推進することが，最も重要な課題である．

CP-PACS は卓越した計算能力を実現したが，その結果として，膨大なデータを処理するための高速な並列 I/O 機能及び並列可視化機能，そしてそれらを超並列計算機と統合する柔軟なマンマシンインターフェースが，超並列計算機の計算能力を生かす上で，解決すべき重要なテーマであることが判明した．計算機工学の面では，このテーマが，ここ数年間に CP-PACS 実機をも援用して研究を進めるべき課題である．

計算物理学は言うに及ばず科学技術分野での計算能力の要求はとどまるところを知らない．将来の 100 TFLOPS から PFLOPS クラスの計算機の実現を念頭に，プロセッサ・メモリ混載 LSI の研究開発等，次世代超並列計算機のために必要な基本アーキテクチャを考察・検証することも，重要なテーマである．

計算物理学をはじめとする科学・技術の多くの分野において，それらの研究の進展が更なる計算機能力の増大を要求し，またその実現によって当該分野が飛躍的に発展するという相互連関は今後もますます顕著となっていくと考えられる．長い目で見れば，先に行われた QCDPAX 計画に続き，本プロジェクトにおいて一層顕著となった貴重な教訓は，このような学際的研究を実施する上で，当該分野の研究者・計算機工学研究者・計算機メーカを併せた共同研究体制が，如何に重要且つ有効であるかを実証した点にある．また，その結果として得られた，計算物理学研究センターを中心とする，計算物理研究者と計算機工学研究者の共同研究体制は，本プロジェクトの生んだ貴重な資産である．

このような資産と経験を生かし，計算物理学と計算機工学両分野に跨る先鋭な学際的研究を一層追求することが今後の大きな目標になっていくものと考えられる．

第 II 部
CP-PACS の開発・製作

11　CP-PACS 全体のアーキテクチャ

本節では，先ず本研究によって開発した超並列計算機 CP-PACS 全体の主要な仕様をまとめて掲げ，次にそのアーキテクチャにつき簡単な説明を行う．さらに主要な項目については，詳細を次節以下に述べる．

11.1　CP-PACS システムの主要な仕様

(1) 全体の理論的なピーク性能：614 GFLOPS (64 ビットデータ，IOU を含めると 652 GFLOPS)

(2) 全体の主記憶容量：128 GB (IOU を含めると 144 GB)

(3) 並列方式：分散記憶型 MIMD 方式

(4) ノードプロセッサ (PU)：2048 ノード +128 IOU ノード，PA-RISC 1.1 仕様のチップに PVP-SW 機能付加 (汎用高性能マイクロプロセッサの機能拡張)

- 物理浮動小数点レジスタ数：128
- クロック周波数：150 MHz
- 理論ピーク性能：300 MFLOPS/PU
- キャッシュメモリ L1：16 KB(I), 16 KB(D), L2：512 KB(I), 512KB(D)
- 主記憶スループット：1.2 GB/sec/PU

(5) 相互結合用ネットワーク: 3 次元 HXB (Hyper-Crossbar network, 8×17×16 の構成)

- 転送ピーク性能：300 MB/sec/link
- 転送方式：wormhole 転送，高速転送プロトコル (RDMA)
- その他機構：バリア同期，放送，ブロックストライド転送

(6) 補助記憶装置：1,059 GB, RAID-5 使用,
IOU (I/O プロセッサ) 経由 並列分散接続

(7) 全体を小部分に分割した運用：各次元方向に独立に 2 分割まで可能

(8) ソフトウェア:

- 言語：assembler, C, C++, Fortran, HPF
- ライブラリ：プロセス間通信通信用 (RDMA 専用ライブラリ, PVM, MPI)，高速ファイル処理用など
- OS：マイクロカーネルをベース

(9) 外部接続:

- FCS (front computer system) との接続：ピーク 100 MB/s の高速接続
- その他：IOU 経由 FDDI, Ethernet

(10) 設置寸法：7.0m(W)×4.2m(D)×2.0m(H)

(11) 消費電力：約 275 KW

(12) 稼働開始：(第 1 期 1024 PU：1996 年 3 月)
第 2 期 2048 PU：1996 年 10 月

(13) 並列計算機システムの信頼度:ハードウェア MTBF $>$ 2000 Hr 目標

11.2 CP-PACS のアーキテクチャ

(1) CP-PACS の目標性能については，計算物理関係者からの要望は表 9 にその例を示すように厳しいものであった．一方，最近の半導体技術の進歩 および 計算機設計技術の進歩により，本研究スタート時点では 1 台の汎用高性能マイクロプロセッサでピーク性能 150 MFLOPS のものが実用の域に達し始め，研究期間中に 300 MFLOPS 以上の性能のものの実現が充分可能と予測された．そこで，PU(ノードプロセッサ) としてはこれを中心に考えることとし，コストパフォーマンス的配慮からも単体の PU 性能を無闇に強化する方策は採らないことにした．

並列機を構成する PU の総台数については，PU 台数も出来る限り大きくしたいが，その反面，

- 設置された時の装置の規模が現実的か否か，
- 装置全体の信頼度が計算物理学の計算に耐えられるものに成り得るか否か，
- 性能と必要とされるコストの問題

なども勘案して台数を決める必要がある．CP-PACS では全体の PU を論理的に 3 次元に配置するイメージから，1024 台構成のものを中心に各種の検討を行った．検討の結果，結合ネットワークの構成方式 (HXB) が明確となり，加えて物理的な実装の可能性と限界も明らかとなり，1024 台構成のシステムを第 1 期の目標仕様とした．実現可能性という意味からは，2048 台以上の構成のものまで可能であることも明らかとなったので，予算措置の進度に合わせ，最終 (第 2 期) は 2048 台の構成のものとした．全体の理論ピーク性能は 614 GFLOPS である．

この他に，後述の IOU(ディスク接続用のプロセッサ) にも，演算用の PU と同一のプロセッサを 128 台使用するので，これを合わせた総ノードプロセッサ台数は 2048+128=2176 台であり，IOU を含めた全体の総理論ピーク性能は 652 GFLOPS となる．

(2) 全体の主記憶容量については，予想される計算物理学の計算に必要となる総記憶容量と，使用可能な半導体記憶素子，各 PU での主記憶装置の構成方法などを勘案して，64 MB/PU としてある．従って PU 全体の主記憶容量は 128 GB である．IOU に関しては，I/O 動作を行う必要性も勘案して，128 MB/PU としてあり，これを含めた総記憶容量は 144 GB である．実際の応用問題では，これ以上の主記憶容量を要求するものがあるが，予算の関係で，ここまでの容量とせざるを得なかったが，これを補うものとして，後述の補助記憶装置 (磁気ディスク) を強化してある．

(3) 並列方式については，物理的な構成から考えて，主記憶装置は各PUに分散して実装する必要があるので，必然的に分散記憶型となる．物理的に分散･論理的に共有メモリ型の並列機の研究が最近行われているが，未だ実用には充分ではない．そこでCP-PACSではsend/receive型のメッセージパッシング (message passing) 方式を採用することとした．

並列処理の方法に関しては，特に初期においては，SPMD (single program multiple data) 方式での処理が多いものと思われ，SIMD型の並列機でも良いのではないかという議論もあったが，SPMDはSIMDとは異なる点もあり，また多種多様な計算物理学上の応用にも応えられ，しかもOLTP (on-line transaction processing)，データベース処理などの並列計算機工学上の問題にも対処できるようにするため，MIMD方式を採用することとしたものである．

(4) 並列計算機を構成するPU (node processor) については，当初汎用の高性能マイクロプロセッサをそのまま使用することを考えていたが，実際の計算物理学の応用での性能を検討した結果，汎用のマイクロプロセッサそのままでは，大規模な計算の場合に充分な性能を発揮できないことが判明した．これは主として汎用マイクロプロセッサは高性能ワークステーションなどへの組み込みを意識したものであって，キャッシュメモリが有効に機能することを前提としているためである．問題によってキャッシュメモリが有効に働かない時には，実効性能はピークの15 %以下にもなってしまうことがある．

そこでCP-PACSとしては，プロセッサアーキテクチャとして次章に述べるようなPVP-SW (Pseudo Vector Processor based on Slide Window, スライドウィンドウ機能による擬似ベクトルプロッセッサ) を新たに考案し，メーカの協力も得て，この機構を採り入れて汎用のマイクロプロセッサの拡張したものを，新たに開発して使用している．理論ピーク性能300 MFLOPS/PUのベクトル処理が可能である．またこれに見合うように主記憶装置のスループットは1.2 GB/sec/PUが用意されている．

この機能の付加により，各PUは実際の問題の計算時にピーク性能の60 %以上の性能を発揮できることが実証され，PVP-SW機能を使用しない場合に比べ，マクロに見ても3〜5倍の実効性能を得ることができ，CP-PACSの実効性能の強化に威力を発揮している．

(5) 相互結合ネットワーク に関しては，隣接したPUの間のデータの転送のみならず，多様な科学・技術計算に適合したものを狙い，各種の結合網につき当初検討を行った．最近では結合網の主要部分は専用VLSIで構成されることになるので，そのVLSIの実現可能性，および相互結合網全体の実装方法と性能との関係を検討し，3次元のHXB (hyper-cross-bar) ネットワーク (図9) を採用した．1本の物理的な転送リンクの転送性能については，2バイト幅のデータのパイプライン的転送 (wave pipeline) を行うことにより，300 MB/secの転送性能を実現した．データ転送方式に関してはウォームホール (wormhole) 方式を採り，経路選択については固定的選択方式により結合網でのデッドロック (dead-lock) を回避する方式を採っている．

計算中のプログラムからのデータ転送要求に対しては，それぞれのPUに存在するOSが関与する通常の転送プロトコルの他に，OSが管理しているI/O領域への (受信側ではI/O領域からの) コピー動作を行うことなく，計算プログラムから直接データ転送を行うことが出来る高速転送プロトコルも設けられており，並列処理のデータ転送に伴うオーバヘッドの低減に大いに役立っている．

また高効率の並列処理を行わせるために，PU間の同期をとるための同期ネットワーク，お

よび多数の PU へ同時に転送を行うための放送用の機能，ブロックストライド転送機能も用意されている．特にブロックストライド転送機能は，後述のように，一定の長さのブロックを単位として，送信側 PU の主記憶上で一定間隔 (送信ストライド) で記憶されている複数ブロックから成るデータを，あたかも連続領域に記憶されているデータの如くに送信し，受信側では，別に指定された一定間隔 (受信ストライド) に従って主記憶に書き込む操作を行うもので，データ転送中にデータの配置換えを行うことが出来，並列処理の効率を高めるのに大きく寄与している．

上記 (3), (4), (5) を基本とする並列処理方式の採用により，CP-PACS では計算物理学上の主要な計算 (例えば QCD 計算) に関しては，全体として実効速度 (sustained speed) がピーク性能の 50 %以上を達成できている．

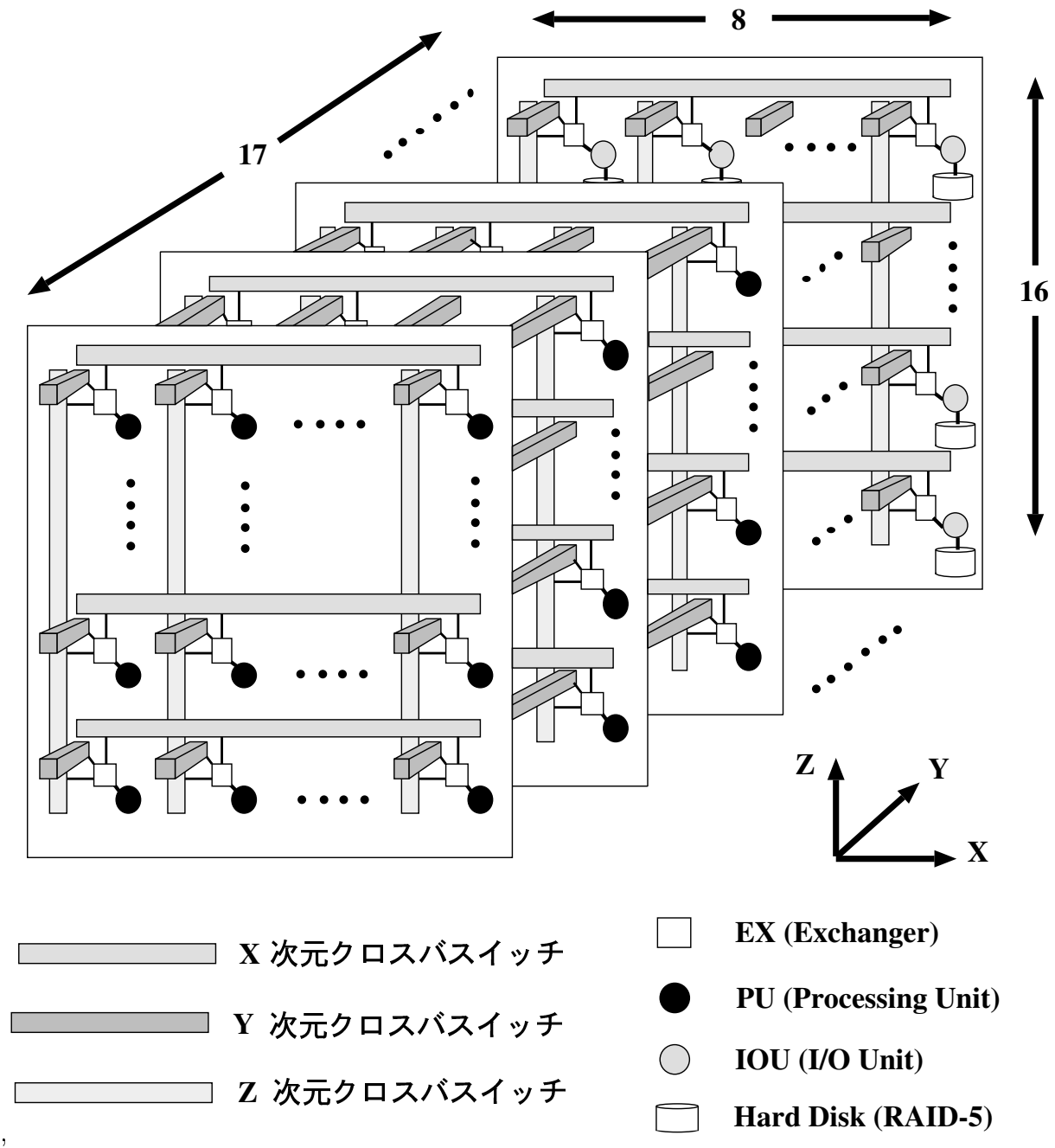
(6) の補助記憶装置については，長時間に及ぶ計算の中間結果などのデータの一時記憶用，バックアップ用，その他のために，並列計算機から直接使用可能な磁気ディスク (disk) 記憶装置を分散して接続することとしてある．必要とされる総記憶容量は大きい程良いには違いないが，当初の検討でも 500 GB 以上と見積もられていた．このような大量の容量のものを実際に設置するにはいくつかの問題がある．

その 1 つはこれだけの容量のものを，コスト/パフォーマンス的に効率の良い 3.5 インチの小型磁気ディスクで実現するとしても，CP-PACS 全体としての総ディスク装置台数は数百台を超えるものとなり，1 台の磁気ディスク装置の寿命が 20 年としても，約 0.5 か月に 1 台の故障 (unreadable error) が発生することになり，補助記憶装置としては大いに問題となる．この点に関しては，当初は他に方法がなく，ユーザは使用時にこのような事を充分注意しながら使用するようにするしかないと思われた．その後この方面の技術が進展し，RAID (redundant array of inexpensive disk, 4 台のディスクに対して 1 台の冗長ディスクを付加することにより 5 台中の 1 台のディスクの故障に対しては自動的に記憶情報の回復が行われるもの) が開発されたので，これを使用することとして信頼性問題は解決した．

もう 1 つの問題は大量のデータの読み書きを行うに要する時間である．この問題は本来読み書きするデータは各 PU に保持されているものであるので，それぞれの PU が並列に同時に読み書きを行うことが出来るようにすることによって，解決される．そのため 3 次元の HXB ネットワークの $n_x \times n_y \times n_z$ 台から成る直方体の 1 つの側面に，もう 1 つの IOU (input/output unit) 用のプロセッサ平面を追加して，$n_x \times (n_y + 1) \times n_z$ の構成とするようにした．CP-PACS の場合，実装上の都合により，$8 \times 16 \times 16 = 2048$ 台の PU に対し，$8 \times 16 = 128$ の X-Z 平面に IOU を配置し，全体を $8 \times 17 \times 16 = 2176$ の構成とし，これらの IOU のそれぞれに，RAID-5 ディスクを接続する分散配置を行うこととした．(図 9 参照).

このようにして，CP-PACS 全体では実効的な I/O 転送能力として 2〜5 MB/sec/DK $\times$128 IOU = 256〜640 MB/sec を用意し，ディスク I/O によるオーバヘッドは総計算時間の 10%以下となることを狙った．

(7) 全体を小部分に分割した運用を可能としたのは，多数の独立な小規模の問題などの場合に，あたかも複数個の並列計算機が設置されているようなイメージで運用することを考慮したものである．このため，3 次元 HXB ネットワークの各次元方向のクロスバスイッチを独立にハードウェア的に 2 分割可能となるようにし，例えば，PU を 1024 台 +512 台 +256 台 +256 台の 4 個の独立な並列計算機としても使用可能なようにしてある．勿論，この他にソ



,

図 9: CP-PACS の全体構成

フトウェア的に分割 (パーティション) を行なって使用する方法も可能である.

(8) ソフトウェア関係については，応用プログラムはユーザが各自作成するのは当然のことである．しかしそのための OS，言語処理系などのシステムソフトウェアを，商用機のように完備したものとして CP-PACS 向きに揃えることは，開発期間から云っても，開発力・予算から云っても到底無理であることが当初から明らかであった．そこで必要最低限のシステムプログラムを開発する方針とした.

最低必要となるものとして，まずプログラミング言語，およびそのコンパイラ (compiler) がある．言語としては Fortran, C, および assembler 言語のみを取り上げることとし，当初は少なくとも超並列計算機を構成する PU のアーキテクチャに追加された擬似ベクトルプロセッサ (PVP-SW) 機能をサポートした 1 台の PU 向きのコンパイラ (およびクロスコンパイラ (cross compiler)) は実現することを目指した．これに対し，現実には後の章で述べるように，PVP-SW 機能をサポートするコンパイラ技術の研究も大いに進展し，メーカの協力もあって，Fortran および C，C++ 言語に関しても PVP-SW 機能サポートのかなり効率の良いコンパイラを開発することができた.

並列処理向きの記述法 (言語仕様などの拡張) については，一般に使用されている PVM (Parallel Virtual Machine) および MPI (Message Passing Interface) のサポートも行なっているが，現在米国などにおいて並列処理ユーザ・メーカ間で広く精力的に仕様が論議されている HPF (high performance Fortran) も取り上げることにしている.

並列向きのプログラム開発用の環境としては，ワークステーション上でのクロスコンパイルとデバッグが行えるようにした他，小規模の並列デバッグ用の並列計算機も設置してある．ワークステーション上でのデバッグのためのシミュレータ (simulator) の用意も行う予定である.

OS に関しては，これも必要最低限の UNIX 系の Mach をベースとした OS カーネル (kernel) とサーバ (server) を備え，メモリ制御，プロセス間通信，I/O ドライバ，ネットワークサーバ，ファイルサーバなどをサポートした．これらと結合ネットワーク通信用のライブラリ (library) 群を用意し，これを逐次整備していくことにしている.

また磁気ディスク補助記憶装置を使用するためのファイル (file) の取扱い手段，FCS (front computer system) との間のファイル転送手段，ジョブ (job) やプロセス (process) の制御手段，運用上必要となる運用・管理ソフトウェアについても一応の用意がなされている.

(9) 超並列計算機 CP-PACS には大容量の補助記憶装置を備えているが，システム全体の構成の章で述べる如く，この補助記憶装置と FCS の間での高速のデータ転送手段が必要とされる．この目的のため，ピーク 100 MB/s 以上の転送を行う高速インターフェースで CP-PACS と FCS 間の接続を行い，ソフトウェアオーバヘッドを考慮しても実効的に 50 MB/s 以上のファイル間転送が可能となるようにした．このため超並列計算機側は IOU の内の 1 台に高速の接続部を設け，FCS 側にも高速転送接続部を特に設けてある.

さらに，超並列計算機を使用するセンターシステムの全体の運用・管理が FCS より一元的に行えるようにし，ジョブの投入・実行制御・結果の取り出しなどの動作の制御を行うために，FCS と超並列計算機の中の 1 台の IOU との間に Ethernet などの LAN も設けてある.

(10) CP-PACS の物理的な諸元に関しては後に述べるが，装置の筐体配置は H 型をしており，設置寸法は 7.0m(W)×4.2m(D)×2.0m(H) である．消費電力については，CMOS 技術の半導

表 9: 計算物理学 大規模問題の所要能力評価例

	仮定した性能	計算例 (主たる解法)	格子数	所要計算時間	所要主記憶容量	所要補助記憶容量
1		ハドロン質量計算				
	sustained:	quenched QCD	48^4	21 日	6 GB	69 GB
	400 GFLOPS	(モンテカルロ法)	64^4	68 日	18 GB	216 GB
	通信/演算:		80^4	163 日	44 GB	527 GB
	20 %	full QCD	24^4	58 日	1 GB	3 GB
	I/O thru-put:	(連立一次方程式)	32^4	244 日	3 GB	9 GB
	300 MB/sec		48^4	1854 日	15 GB	46 GB
2		宇宙流体力学				
	sustained:	格子法 (オイラ)	1000^3	120 hr	80 GB	400 GB
	500 GFLOPS	(P^3M-TVD 法)				
	I/O thru-put:	流体粒子法 (ラグランジ)	200^3	110 hr	4 GB	200 GB
	100 MB/sec	(Tree 法)				
		電磁流体力学 (Roe MHD 法)	800^3	250 hr	136 GB	683 GB
		輻射流体力学	$100^3 \times 1$k	75 hr	176 GB	250 GB

体素子を主体としているので，プログラムの稼働状況によって大きく変動するが，重い負荷の時で約 275 KW である．冷却方式は床下空調による強制空冷方式である．

(11) 稼働開始時期については，基本設計，詳細設計およびパイロットボード (pilot board) などによる検討，実装設計および先行試作，全体装置の製作・調整などの必要な検討を経て，1995 年度末 (1996 年 3 月) に第 1 期 1024 PU のハードウェアの据付けを行い，稼働を開始した．その後並行して準備されていたソフトウェアのデバッグを行い，実使用を開始した．これと並行して第 2 期のための製作を行い，1996 年 10 月から 2048 PU で稼働している．

(12) 並列計算機システムの信頼度については，先に補助記憶装置に関して述べたように，本並列計算機のように大規模なシステムでは信頼性を確保することはかなり難しい問題ではあるが，最近のコンピュータ技術の発展によるスーパコンピュータや大規模汎用システムの実績を考慮して，ハードウェアの MTBF (mean time between failure) は 2000 Hr 以上を目標とすることとした．使用部品の信頼性データを基にした積上げ計算からもこの数字は妥当な線であるが，現時点では他の要因によるトラブルも多少あるので，実績データとして得られるのは，これからであるが，現在の稼働状況からすると，2000 Hr の目標値は充分達成可能であると思われる．

12 ノードプロセッサ・アーキテクチャ

12.1 従来のマイクロプロセッサの問題点

超並列計算機の各 PU のプロセッサとして有力な候補の一つは，一般のワークステーションに用いられているような高速の汎用 RISC マイクロプロセッサである．近年の汎用 RISC プロセッサの処理能力は，集積回路技術の進歩によるクロック周波数の向上と，スーパースカラ方式等に代表される命令レベルの並列性抽出による高速処理方式の実用化により，飛躍的に向上している．

しかし，その高速性は「キャッシュが有効に働く」という仮定が成立する時においてのみ達成される。大規模な科学技術計算においては，データ領域が非常に大きく，またデータの時間的局所性が少ないという性質があるため，データキャッシュのミス率が大きい．このため，通常の汎用RISCプロセッサの性能は，主記憶アクセスレーテンシによるペナルティのために大きく低下する．また，半導体技術の進歩により，プロセッサのクロック周波数は今後も飛躍的に向上することが予想される一方で，主記憶のアクセスレーテンシを決める大きな要因であるメモリ素子のアクセスタイムは，半導体技術の進歩によってもそれほど短縮しない．そのため，主記憶アクセスペナルティによる性能低下は，今後の半導体技術の進歩によっても改善されず，むしろさらに深刻な問題になっていくことが予想される．したがって，大規模な科学技術計算を高速に処理するためには，主記憶アクセスレーテンシを隠蔽する手法が必要不可欠である．

この問題に対する1つの解決法として，プリフェッチ機能をキャッシュに持たせ，必要なデータをあらかじめキャッシュにプリフェッチする手法が考えられる．しかしこの手法には，非連続アドレスアクセス時に同一キャッシュライン中の不要なデータもフェッチされるため主記憶とキャッシュの間に無駄なトラフィックが生ずる，データ参照に時間的局所性がない場合にはキャッシュのトラフィックが非常に厳しくなる，データプリフェッチの際にline conflictにより有効なデータがキャッシュから追い出されてしまう，という3つの大きな問題点がある．

スカラプロセッサとベクトルプロセッサとの大きな相違点は，主記憶のデータ供給能力にある．ベクトルプロセッサでは主記憶アクセスがパイプライン化されておりそのスループットは高い．また，多数のベクトルレジスタを持つことによりベクトルロード／ストア処理のベクトル長を長くでき，チェイニングの機構によりこれらの処理と演算処理とを平行に行うことができる．これにより，レジスタへのプリロード機能を実現でき，主記憶アクセスレーテンシが性能に与える影響をかなり隠蔽できる．しかしながら，スカラプロセッサの主記憶はパイプライン化されておらず，スループットは低い．そしてキャッシュミス時には主記憶アクセスペナルティにより実効性能は大きく低下する．しかしながら，両者とも複数のパイプライン化された演算ユニットを持ち，複数命令同時発行 (スカラ) あるいはチェイニング (ベクトル) といった，複数の演算ユニットを有効に働かせる機構を持つ点では類似している．従ってスカラプロセッサにおいても，十分な主記憶のスループットを確保し、さらにレジスタプリロード機能を実現することで主記憶アクセスペナルティを隠蔽できれば，1つのベクトル命令の処理内容を，複数のスカラ命令によって垂直マイクロプログラミング的に処理することにより，高速にベクトル処理を行うことができると考えられる．

12.2 擬似ベクトルプロセッサ PVP-SW

CP-PACSのプロセッサ・アーキテクチャは，科学技術計算を極めて高い効率で並列処理できるように考えられている．

CP-PACSでは，前節で述べたようなベクトルプロセッサが実現する機能／特徴を，スーパスカラ方式のプロセッサに機能拡張を施して実現することを考える．この処理方式を我々は擬似ベクトル処理 (Pseudo Vector Processing) と呼ぶ．擬似ベクトル処理を実現するためには，前節で述べたスカラプロセッサとベクトルプロセッサの相違点である以下の3点を実現する必要がある．

- 多数の浮動小数点レジスタ
- レジスタへのプリロード機能の強化
- 主記憶アクセスのパイプライン化

我々の設計方針は，ソフトウェアの開発コストを抑えるべく，既存の RISC アーキテクチャとの上位互換性を保ちながらこの 3 つの機能／特徴を実現することである．第 3 の点は，主記憶を多数のバンクに分割し，インターリーブメモリを構成することにより擬似的に実現できる．しかしながら，第 1 の点である多数のレジスタを実現するためには，命令フォーマット中のレジスタ指定フィールドのビット数を増やす必要があるため，通常は命令セットアーキテクチャの互換性は失われる．

そこで，既存の RISC アーキテクチャとの上位互換性を維持しながら，多数のレジスタを利用できる「スライドウィンドウ方式に基づく擬似ベクトルプロセッサ **PVP-SW**(**P**seudo **V**ector **P**rocessor based on **S**lide-**W**indowed registers)」を提案した [6].

PVP-SW のアーキテクチャは既存のスカラプロセッサに対する拡張という形で定義される．拡張点は，浮動小数点レジスタのスライドウィンドウ化と数種類の命令追加である．

スライドウィンドウ構成　図 10 に，浮動小数点レジスタのスライドウィンドウ構成を示す．物理的なレジスタ空間は複数の論理的なウィンドウに分割される． 1 つの論理的なウィンドウ内のレジスタ数は，拡張前のアーキテクチャで定義される数 (図 10 では 32 とした) と等しくなければならない．ウィンドウの位置は **window-offset** で与えられる．各ウィンドウは global 部と local 部にわかれ，global 部のレジスタは全てのウィンドウに共通である．図 10 では，global 部のレジスタは g 個とした．

ある時点ではただ 1 つのウィンドウのみがアクティブであり，このアクティブウィンドウを指定するポインタ **FWSTP**(**F**loating-point **W**indow **ST**art **P**ointer) を導入する．拡張前のアーキテクチャにおける命令は全て FWSTP で指されるアクティブウィンドウ内のレジスタのみを用いる．一方，ソフトウェアで FWSTP の値を変更することにより，このアクティブウィンドウを物理的ウィンドウ空間内で自由に移動 (スライド) することが可能である．ウィンドウ内のレジスタ数は拡張前のアーキテクチャに依存するが，物理レジスタの数 m はハードウェアが許す限り任意の個数実装することが可能である．このようにして，上位互換性を維持しながら多数のレジスタを利用可能となる．

追加命令　以下の命令が追加される．

- **FWSTPset** アクティブウィンドウを指す FWSTP の値を変更し，アクティブウィンドウを切り替える．
- **FRPreload** データをメモリから任意に指定されるウィンドウ内のレジスタにロードする．
- **FRPoststore** データを任意に指定されるウィンドウ内のレジスタからメモリにストアする．

FRPreload 及び FRPoststore 命令は主記憶とレジスタとのデータ転送を行う命令であるが，以下の 3 つの特徴を持つ．

1. 「置いてきぼり処理」により，データ転送の完了を待たずに後続命令は実行される
2. アクティブウィンドウに関係なく，全物理レジスタをデータ転送の対象として指定できる
3. キャッシュミス時には主記憶とレジスタの間で直接データを転送し，キャッシュ内データを変更しない

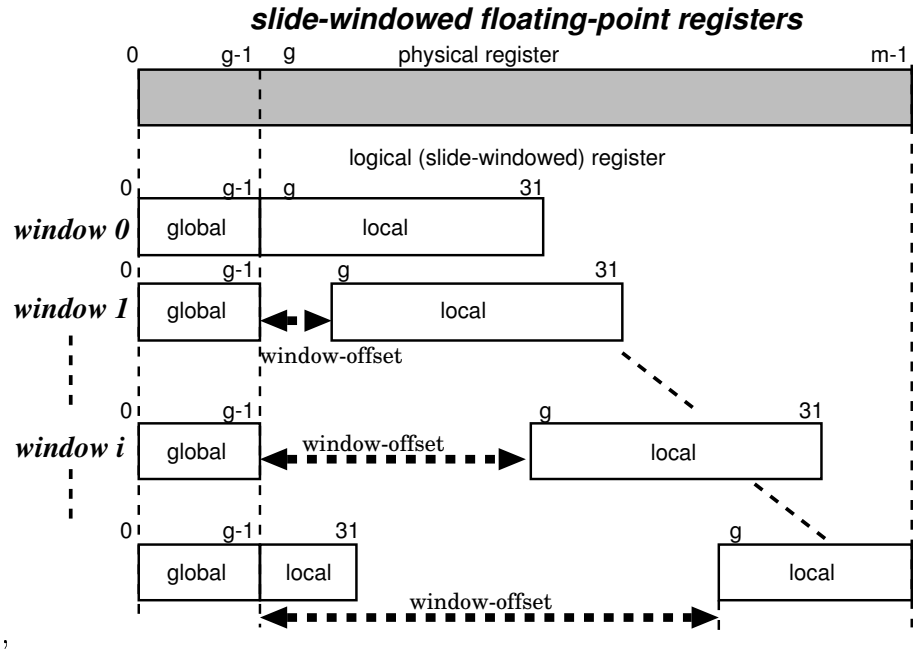


図 10: PVP-SW における浮動小数点レジスタの構成

1 番目の特徴により，主記憶のアクセスレーテンシの影響でデータのロードやストアの実行に長い時間を要したとしても，それらの動作を放っておき，後続命令を実行できる．このため，将来必要となるデータに対するプリロード命令を先行発行することで，他の命令実行を妨げることなく該当データに対する主記憶アクセスレーテンシを隠蔽できる．より長いレーテンシを隠蔽するためにはそれだけプリロード命令を早く発行する必要があり，必要なレジスタ数が増加する．この問題は物理的に多数のレジスタを用意すること，及び 2 番目の特徴で解決する．3 番目の特徴は，キャッシュのスループット不足による性能低下を防ぐため，及びキャッシュコヒーレンシ問題に対処するためである．

12.3 PVP-SW における擬似ベクトル処理例

DAXPY ループ '$Y(i) = a \times X(i) + Y(i)$' を例として，PVP-SW における擬似ベクトル処理を説明する．図 11 にオブジェクトコードを示す．説明を簡単にするため，分岐命令は省略してある．図 11 中の '`FRPreload Y(i+1) -> r30<+2>`' は，現在アクティブであるウィンドウの 2 つ先のウィンドウ (そのウィンドウの window-offset は，アクティブウィンドウの window-offset より 2 大きい) の論理レジスタ`r30` に `Y(i+1)` のデータをプリロードすることを表す．また，最後にある '`FWSTPset +2`' は FWSTP の値を 2 増やすことを表す．

図 11 のオブジェクトコードを実行する時のレジスタ割り当てを図 12 に示す．図 12 ではループ処理が時間の経過と共に展開されて示されている．6cycle ごとの点線は，アクティブウィンドウが切り替る事を表す．影付きの部分は `Y(i+2)` の計算に関係する部分である．

図 12 よりわかるように，ループ毎にウィンドウを切り替えていくことにより各ループは異なるウィンドウ上で処理される．従って同一の論理番号を持つレジスタも，ループ毎に物理的には異なるレジスタに対応することになる．このようにして 32 レジスタしか指定できない演算命令が多数のレジスタを用いて処理を行なうことができる．

```
Loop:  FRPreload      Y(i+1) -> r30 <+2>
       ADD            r29 + r30 -> r29          % aX(i) + Y(i)
       FRPreload      X(i+2) -> r31 <+2>
       MULT           r7*r31  -> r31            % a * X(i+1)
       FRPoststore    r29 <-0> -> newY(i)
       FWSTPset +2    (FWSTP <- FWSTP + 2)
       % branch is omitted
```

図 11: PVP-SW における DAXPY ループのコード例

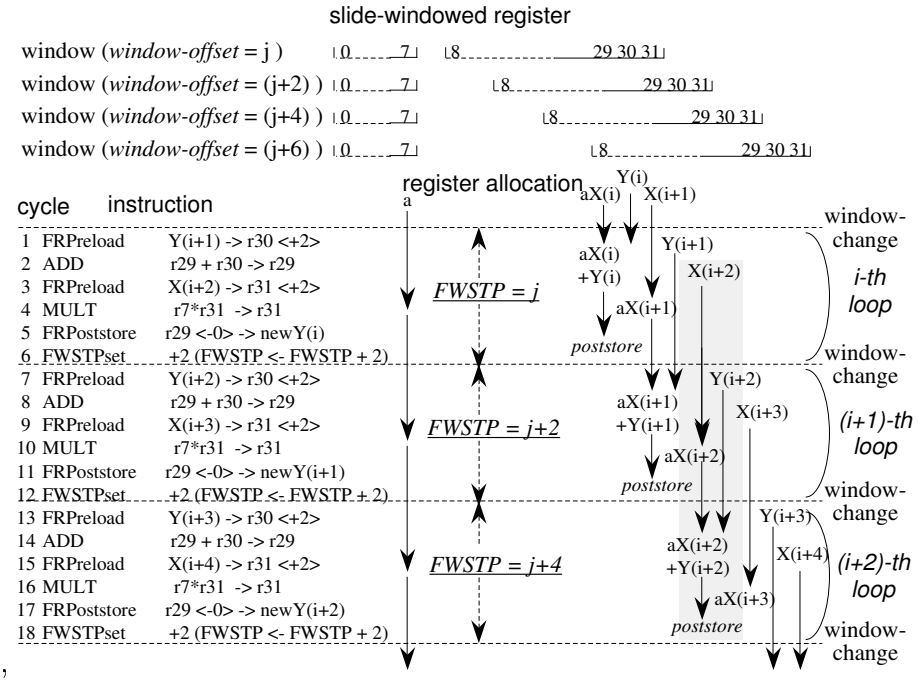


図 12: PVP-SW におけるレジスタ割り当て

主記憶アクセスレーテンシの隠蔽　あるデータに対するロード要求から，そのデータが演算命令により最初に使用されるまでの時間間隔を，そのデータに対する *permitted latency* と呼ぶ．例えば図 12 では `X(i+2)` に対する *permitted latency* は 7cycle である．全てのデータに対する *permitted latency* が主記憶アクセスレーテンシ以上であれば，主記憶アクセスレーテンシによる性能低下を隠蔽することができる．図 12 において，*permitted latency* の最小値は 7cycle であるので，7cycle 以下の主記憶アクセスレーテンシを隠蔽できることになる．より長い主記憶アクセスレーテンシを隠蔽するためには，*permitted latency* を長くする必要がある．そのためにはできるだけ遠くのウィンドウにできるだけ早くデータをプリロードすれば良い．具体的には，図 11 における '`FRPreload Y(i+1) ->r30<+2>`' 及び '`FRPreload X(i+2) ->r31<+2>`' の下線部を例えば，'`FRPreload Y(i+2) ->r30<+4>`' 及び '`FRPreload X(i+3) ->r31<+4>`' に変更すれば良い．この変更では，プリロード命令がさらに 1 ループ先行して発行されることになるため，結果的に *permitted latency* は

$$7(\text{変更前の } permitted\ latency) + 6(1\ \text{ループの実行に要する時間}) = 13cycle$$

になる．しかし，このようにより遠くのウィンドウにデータをプリロードするためには，より多くの物理レジスタが必要になる．

12.4　CP-PACS で採用したプロセッサの諸元

CP-PACS では Hewlett Packerd 社の PA-RISC 1.1 アーキテクチャにこの PVP-SW 機構を追加し，PU の中心となるプロセッサとした．PA-RISC 1.1 は 32 本の倍精度浮動小数点レジスタを持っており，これを拡張して物理レジスタ群とウィンドウ制御回路 (主に論理レジスタ番号と物理レジスタ番号の変換をする) を追加する．実装した物理レジスタの数 128 である．この数は、隠蔽すべき主記憶アクセスレーテンシと実装上の制約から決定した．また，グローバル・レジスタの本数は，ソフトウェアにより，8，12，16 の 3 つの中から最適なものを選択できるようにした．

レジスタのウィンドウ化に伴うハードウェアコストの増加について簡単に触れる．まず，数十の物理的なレジスタの追加だが，レジスタの入出力のポート数を増加する必要がないため，現在の実装技術から考えて (実装に関しては 15.1 で述べる) この程度のレジスタ増加は特に問題はなく，文献 [10] に示されるように、レジスタのウィンドウ化による面積増加は高々5%であった．ウィンドウ制御回路については，チップの面積よりもむしろプロセッサのマシンサイクルに与える影響を考慮すべきであるが，実装されるレジスタ数が 128 本以下であれば論理レジスタ番号と物理レジスタ番号の変換に必要なのは高々7bit の加算器であり，この点もあまり問題ではなかった．

このプロセッサの物理的な諸元については、15.1 で述べる。

13　プロセッサ間接続ネットワーク

本節では CP-PACS のネットワーク・アーキテクチャについて，その概略を述べる．CP-PACS は大きく分けて，データ転送を目的とするデータ転送ネットワークと，PU 間のバリア同期を目的とする同期用ネットワークの 2 種類のネットワークを持つ．以下，各々のネットワークの特徴と，高速データ転送のための特殊な機構について述べる．

13.1 データ転送ネットワーク

超並列計算機向けのネットワーク・トポロジーとしては，Mesh/Torus，Binary n-Cube，多段結合網 (Multistage Interconnection Network)，等いろいろと提案されているが，CP-PACS ではデータ転送ネットワークとして Hyper-Crossbar ネットワークと呼ばれるトポロジを採用した (以下，HXB と略す)[8].

CP-PACS 上で QCD 計算を行う際に必須要求となっているのは，PU を多次元格子上に配置した場合に隣接 PU (Processing Unit) 間でのデータ交換が無衝突で行われることである．すなわち，全 PU が一斉に同一方向の隣接 PU に対してメッセージを転送した場合，それらのメッセージはネットワーク上で互いに衝突することなく，ピーク性能に近いスループットで転送される必要がある．この要件を満たすためには QCD-PAX で用いられた Torus 型ネットワークを用いるのが最も簡単であるが，CP-PACS では単純な隣接交換だけでなく，より複雑なデータ転送パターンにも対応可能な，より柔軟なトポロジを持つネットワークとして HXB を採用する．これは，QCD 以外の広汎な科学技術計算にも柔軟に対応可能な計算機の設計を目指すことにつながる．

さらに，HXB では HXB そのものの物理的なプロセッサの配置と同等な Mesh/Torus だけでなく，総プロセッサ台数が等しく，構成の異なるような Mesh/Torus における隣接転送をも容易にシミュレートすることが可能である．例えば，$8 \times 8 \times 16 = 1024$ プロセッサの 3 次元 HXB 上では，同サイズの 3 次元 Mesh/Torus だけでなく，$4 \times 8 \times 32$ のように各次元のサイズが違うものや，32×32 のように次元数そのものが違うような Mesh/Torus に関しても，その隣接転送を無衝突でシミュレートすることができる．このことは，ユーザがプロセッサの結合形状に合わせて問題を特化させることなく，その性能を最大限に利用できるという利点がある．

HXB のトポロジは n 次元の超立方体の概念を拡張した形としてとらえることができる．ここで次元数 n は任意に選択可能であるが，CP-PACS 実装時の実装技術を想定した結果，3 次元の構成が妥当であると考えた．次に，各次元の PU のサイズを決定する．これは全 PU を 3 次元の直方体の形に配置することに相当し，例えば 1024 PU の場合，典型的な構成は $8 \times 8 \times 16$ という形になる（直方体であるため，各次元のサイズが一致している必要はない[1] ）．実際の CP-PACS では，2048 PU+ 128 IOU という構成のため，全体では $8 \times 17 \times 8$ の構成になっている（図 9 参照）.

一般的な 3 次元 HXB の特徴について，$X \times Y \times Z$ 構成の場合を想定して説明する．まず，全 PU は $X \times Y \times Z$ の 3 次元正方格子状に配置される．そして，1 つの次元方向 (例えば x 次元方向) に並んだ PU 同士をクロスバ・スイッチ (以下，XB と省略する) で完全結合する．例えば x 次元方向の場合，X 入力 X 出力 の XB が必要となる．このようにして全ての次元方向について，その次元方向に並んだ PU 間を XB で結合する．この場合，次元数は 3 なので，各 PU は 3 つの次元方向の XB に対するリンクをそれぞれ持つことになる．

このネットワーク上でのメッセージ転送を考える．以下，座標 (x, y, z) の位置にある PU を $PU(x, y, z)$ と表すことにする．今，$PU(x_s, y_s, z_s)$ から $PU(x_d, y_d, z_d)$ までメッセージを転送するとする．最も簡単な方法としては，まず $PU(x_s, y_s, z_s)$ から $PU(x_d, y_s, z_s)$ までメッセージを送り，続いてそれを $PU(x_d, y_d, z_s)$ へ，そして最後に $PU(x_d, y_d, z_d)$ へ，というように 3 ステップの転送を繰り返して送ればよい．

しかし，この方法では 3 ステップの転送を繰り返すため転送遅延が大きいので，より高速なメッセージ転送を行うため，Wormhole 方式のメッセージ転送方式を行う．これは，中継点の各 PU が一旦メッセージを蓄えるのではなく，その位置の PU を経由せず，ある XB から別の XB へのメッセージの直接転送を行うことにより，上記 3 ステップの転送を一気に行う方法である．このた

[1] 各次元方向のサイズを 定数 m として等しくしたものは base m- n cube として知られている

めに，各 PU と XB を結合する部分に EX (Exchanger) と呼ばれるルータ・スイッチを置く．EX は PU → 任意の XB，ある XB → 別の XB，ある XB → PU というように，メッセージが全ての方向の PU と XB の間を自由に乗り替えられるようにするスイッチで，その制御はメッセージが持つヘッダ情報によって自動化されている．また，メッセージ転送経路の確保は，メッセージのヘッダ部が EX または XB を通過する段階で行われるので，経路確保のための初期オーバヘッド等を小さくすることができる．

Wormhole 方式の HXB には以下のような特徴がある．

- 通信距離が短い．これは短いメッセージを頻繁に転送する際に有利である．
- 同一サイズの正方格子結合 (Mesh/Torus) を直接無衝突でエミュレートできる．さらに，サイズあるいは次元数が異なる場合でも，総 PU 数が等しいか小さければ，それをエミュレートできる．
- Binary-n-Cube ネットワークの部分的なエミュレートが容易である (同ネットワークで無衝突で行なえる転送パターンの多くが，同様に無衝突で実現できる)．
- 各 XB の出力を全開放にすることにより，ブロードキャスト転送が容易に行える (通常の 1 対 1 転送と同じ時間で可能)．
- ランダム転送 (送受信の PU の対が動的にランダムに決定するような通信) におけるスループットが，他のネットワークに比べ非常に高い．

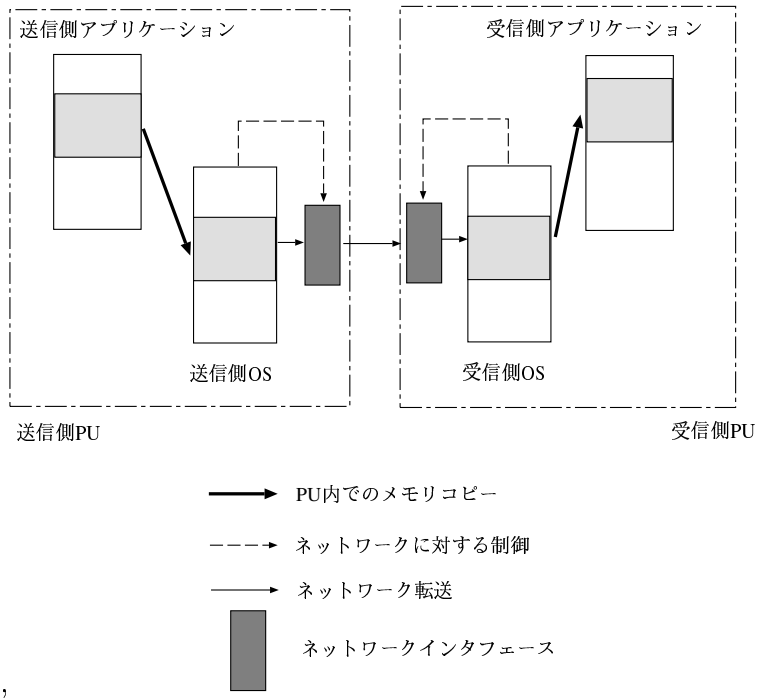
Mesh/Torus ネットワークの隣接転送を無衝突で行える点は，多くの科学技術計算において重要である．さらに，対象とする問題がある程度任意の大きさの場合でもエミュレーションが可能であるため，プログラミングの自由度という点から見ても，ユーザにとっては非常に使い易く，この点は QCD 計算において重要であると思われる．また，ブロードキャストや全対全通信についても，Mesh/Torus ネットワークに比べ非常に高速に処理することができる．

一般に，n 次元 HXB では，最短経路の転送を行う場合，$n!$ 通りの経路が存在する．これらの経路の選択方法には，静的にそれを決定しておくもの (固定ルーティング) と，ネットワークの混雑状況等を評価しながら動的にそれを決定するもの (適応ルーティング) の 2 種類がある．Wormhole 方式の HXB において，単純な適応ルーティングを行った場合，メッセージ同士がネットワーク上でデッドロック (メッセージ同士が互いに相手をブロックし合い，ネットワークが停止してしまう状況) を起こす可能性がある．この問題は仮想的に多重化した転送経路を用意することで解決できるが [12]，CP-PACS 用 HXB ではハードウェア・コストを抑えるため，固定ルーティング方式を用いた．

13.2 データの高速転送のための機構 – Remote DMA

最近の超並列計算機におけるネットワーク性能においては，メッセージの転送スループットだけでなく 1 つのメッセージの転送の立ち上げ時間に対する配慮がなされるのが普通である．この時間をセットアップ時間と呼ぶ．1 つのメッセージ長が非常に長い場合，セットアップ時間の影響は無視することができるが，メッセージ長が短くなるにつれ，メッセージの転送開始から終了までの時間のうちセットアップ時間が占める割合が大きくなってしまう．QCD 計算においては比較的長いメッセージが扱われるが，問題の範囲をより広汎な科学技術計算に広げると，この問題が重要になってくる．

CP-PACS では高速なメッセージ転送立ち上げを実現するために，ユーザ・アプリケーションから直接ネットワークへのメッセージ転送起動が行えるように，特別な機構を用意した．この機構を用いる転送を Remote DMA (Direct Memory Access) 転送モードと呼ぶ．この様子と，通常転送との違いを図 13 及び図 14 に示す．



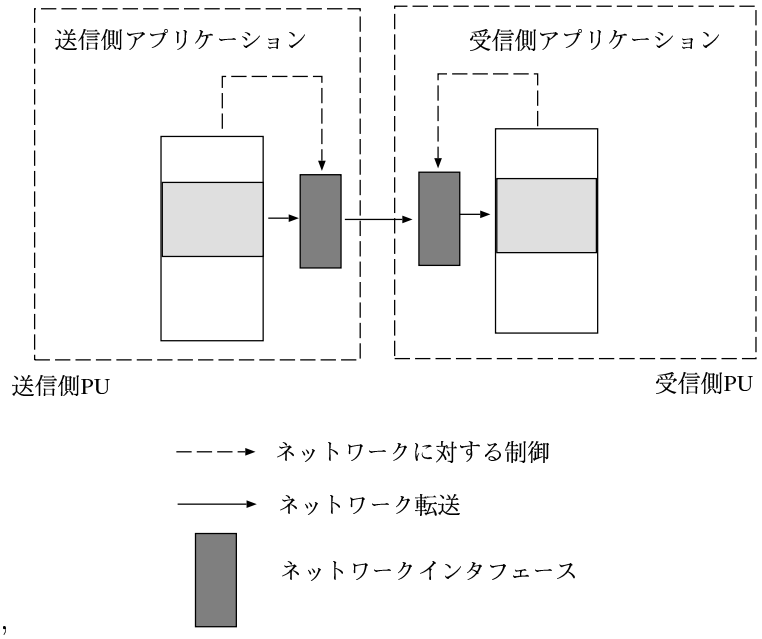
,

図 13: 通常転送モードにおけるデータと制御の流れ

通常の転送では，メッセージ転送要求は各 PU 上のオペレーティング・システムによって処理され，ネットワーク・インタフェース (NIA: Network Interface Adapter) に対する命令の発行やメッセージ到着処理は全てオペレーティング・システムによって行われる．ユーザ・アプリケーションは転送終了まで待たされ，その後で処理が継続される．メッセージ転送はオペレーティング・システムが管理するメモリ上で行われる．まずアプリケーションのメモリからオペレーティング・システムのメモリへデータがコピーされ，それがネットワークを通じて相手 PU のオペレーティング・システムのメモリに転送される．その後，相手側のアプリケーションが再度オペレーティング・システムからデータをコピーし，受信が完了する．この方式では，アプリケーションからオペレーティング・システムに処理が移行する際の処理が非常に重く，結果としてメッセージのセットアップ時間が無視できない大きさになってしまう．

Remote DMA 転送モードでは，オペレーティング・システムを経由せず，アプリケーションが直接 NIA にアクセスし，データをネットワークに送出する．一般に NIA はアプリケーションが誤ったメッセージを送出しないように (例えば処理と無関係な PU への送出や，システム上で転送を禁止されている PU への送出等)，ハードウェアによる保護がなされている．高速転送モードではこの保護を一部解除し，システムに破綻をきたさない範囲でアプリケーションが直接通信を行うことを許可する．オペレーティング・システムを介さないことにより，以下の 2 つの利点が得られる．

- オペレーティング・システムへの処理の移行に伴うオーバヘッドが解消される．
- 送信側アプリケーションのメモリ空間から受信側アプリケーションのメモリ空間に対して直



,

図 14: Remote DMA 転送モードにおけるデータと制御の流れ

接データが転送されるため，データ・コピーの時間が短縮される．

メッセージ長が短い場合は第 1 の利点が，メッセージ長が長い場合は第 2 の利点がそれぞれ活かされ，結果として高速なメッセージ交換が可能となる．

13.3 メッセージのブロード キャスト 転送

並列処理においてしばしば現れる特殊なメッセージ交換の例としてブロードキャスト通信がある．これはある PU から他の全ての PU に対して同一メッセージを一斉に転送するもので，ある数値を集計しその結果を分配する場合等，多くのアルゴリズムで多用される．そのため，CP-PACS では HXB の特性を活かし，ブロードキャスト・メッセージを高速に処理できるような設計がなされている．

HXB においては，ある XB の入力に届いたメッセージを，その XB の全出力に同時に送出することにより，単一次元内でのブロードキャストが容易に実現できる．これを全次元に対して行えば，全 PU に対するブロードキャストが実現できる．CP-PACS では $x \to y \to z$ という固定的な経路決定を用いるので，ある PU から送られたメッセージがまず x 次元方向の XB においてブロードキャストされ，続いてそれらのメッセージが y 次元方向の XB においてブロードキャストされる．そして，最後にそれらのメッセージが z 次元方向にブロードキャストされることにより，全 PU に対するブロードキャストが行われる．

図 15 は，2 次元構成の HXB におけるメッセージ･ブロードキャストの概念を表している ($PU(0,1)$ が転送元)．実際の CP-PACS(3 次元 HXB) におけるブロードキャストも，これをもう 1 ステップ拡張した形で実現される．

ブロードキャストにおいて注意すべき点は，HXB ではその構造上，複数の PU からの別々のブロードキャスト要求を同時に処理することが不可能であるということである．上述のブロードキャスト・アルゴリズムを HXB 上で複数のメッセージに対して実行すると，ネットワーク上でデッドロックを生じる．これに対する対策として，例えばある PU だけがブロードキャスト・メッセー

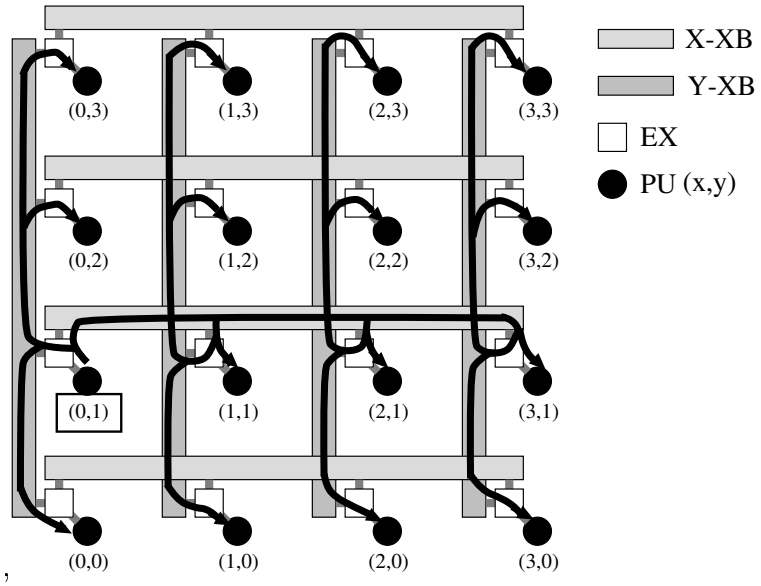


図 15: 2 次元ハイパクロスバ網におけるメッセージ・ブロードキャスト

ジを流すことができるという制限を設け，他の PU からの要求は全て一旦その PU に送ってから改めて処理するという方法が考えられる．この方法ではブロードキャスト要求が逐次処理されるため，ブロードキャスト・メッセージが同時に複数存在することがなくなる．CP-PACS の HXB も基本的にこの手法を元に設計された．

13.4 ストライド転送

一般的なメッセージ転送では，各 PU のメモリ上で 1 次元の連続領域上に配置されているデータが一つのメッセージとして転送される．メモリ上の連続データは，PU が NIA にメモリ上でのデータの開始番地とデータの長さをセットすることにより，自動的にネットワーク上に送出され相手 PU に届く．

しかし，問題空間の形状によって，あるいは問題を解くアルゴリズムによっては，一つの PU に一度に転送したいデータが，必ずしも送信 PU 側のメモリの連続領域に格納されているとは限らない場合がある．逆に受信 PU 側で，送信 PU から送られてきたデータをメモリの連続領域ではなく，飛び飛びの領域に格納したい場合がある．一例として，2 次元空間上のデータ列をメモリ上に配列として配置した場合，どちらか一方の次元方向のデータはメモリ上で連続領域に格納されるが，もう一方のデータは必ず一定間隔を空けた番地に配置される．この時，第二の次元方向で連続なデータ群をある PU に転送しようとすると，メモリ上で (周期性はあるが) 不連続な番地上のデータ群を転送することになる (図 16 参照)．

一般の超並列計算機では，メッセージのストライド転送はユーザのメモリ空間からオペレーティング・システムのメモリ空間にデータをコピーする際にソフトウェア的に行うことができる．すなわち，ユーザのメモリからストライドを意識しながらデータをコピーしパッキングしたものをメッセージとして送出する．しかし，CP-PACS において Remote DMA 転送を用いる場合，データはユーザ空間から直接 NIA にデータが送られるため，ソフトウェア的なパッキング手法が使えない．そのため，NIA がハードウェア的にこのような処理を行う必要がある．もし NIA が単純な連続転送機能しか持っていないとすると，このような転送はデータ一つずつの転送の繰り返しと

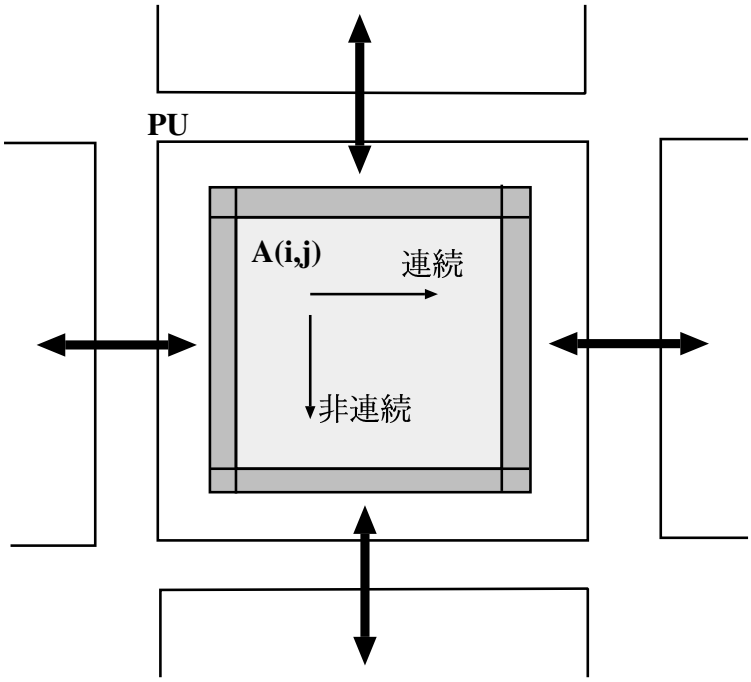


図 16: 2 次元データの隣接 PU 間転送におけるストライド転送の必要性

して処理しなければならず，転送時間のほとんどがそのセットアップによって消費されてしまい，処理効率低減の重大な要因となる．

そこで CP-PACS では，このようなメモリ上で周期性を持つ不連続番地に配置されたデータ群を，一度のセットアップによってネットワーク転送するための機構が NIA に用意されている．このようなデータ転送方式をストライド転送と呼ぶ．ストライドとは不連続なデータの間隔のことである．一般の連続転送では，NIA に対してメモリ上のデータの開始番地とデータ長のみをセットするが，ストライド転送においては，データの開始番地，各データのサイズ，データとデータの間隔 (ストライド)，全転送データ長といったパラメータをセットする．ストライド転送が始まると，NIA は指定された間隔でデータをメモリから取り出し，ネットワークに送出する．この時，ネットワーク上では (元のメモリ上の番地が不連続であっても) データは密に送られるため，データ転送に要する時間は実際に送信するデータ量に対してかかる分だけで済む．

13.5 同期ネットワーク

QCD-PAX においては，隣接 PU 間での共有メモリ方式によって PU 間のデータ交換が行われてきた．このため，全 PU によるバリア同期はアルゴリズム上不可欠なものとなっていた．CP-PACS でもその流れを汲み，同期用ネットワークを用意している．一般的に，メッセージ・パッシング系の PU 間交信では，メッセージの送受信処理そのものが point-to-point の同期操作を同時に実現しているため，その他の同期処理が不要な場合が多い．しかし，現実的には並列プログラムのデバッグ時や，非均質なデータ転送を行った後で一斉転送を行う場合等，ある程度の粒度で PU の同期をとりたい場合が存在する．そこで，CP-PACS でも，このような場合に簡単に全 PU でバリア同期をとることができるような機構を用意した．

CP-PACS における同期ネットワークは，基本的にデータ転送ネットワークを部分的に用いる

ことにより実装される．具体的には，全 PU で一斉同期をとる必要が生じた場合，各 PU は同期用の特殊なメッセージをデータ転送ネットワークに流す．HXB 上の各 XB では，全ての入力からの同期用メッセージの待合わせが行われ，全入力からのメッセージが揃った時点で次の段の XB に送られる．これが繰り返されることにより，最終段 (z 次元方向 XB) にメッセージが揃った段階で全 PU の同期が確認され，それが全 PU に通達される．

このように，同期ネットワークは仮想的にデータ転送ネットワーク上にマップされているため，その動作が本来のデータ転送の妨げにならないよう配慮しなければならない．このため，データ転送ネットワーク上には，同期用メッセージを保持するための専用バッファをデータ保持用のバッファとは別に確保しており，互いの通信が円滑に進むような構造になっている．これにより，少なくともデータ転送メッセージと同期用メッセージがデッドロックを生じることはない．

13.6 TCW チェイン機能

CP-PACS のネットワークでは wormhole 方式によってメッセージを転送するため，一つのメッセージが長時間に渡ってネットワーク上の特定チャネルを占有してしまう可能性がある．また，固定ルーティング方式を採っているため，こうしたチャネル占有状態が続くと，他のメッセージの転送に支障をきたす可能性がある．このため，CP-PACS では，1 つのメッセージ・ブロックの最大長を 64KB までと制限している．このメッセージ・ブロックをパケットと呼ぶ．64KB 以上のメッセージは複数のパケットに分割され，転送される．固定ルーティング方式を採っているため，同一 PU 間でのメッセージ転送において，ネットワーク上でのパケットの追い越しは生じず，パケットの到着順序は保証される．このため，1 つのメッセージが複数のパケットに分割されても問題は生じない．

しかし，複数のパケットに分割されたメッセージを転送する際，ユーザがパケット転送を何度も繰り返すような方式では，パケット送出のオーバヘッドが大きくなってしまい，メッセージ転送の実効スループットの低下につながる．そこで，CP-PACS の NIA では，複数のパケットを自動的に連続転送するような機構が備えられている．

一つのパケットには送信 PU，受信 PU，送信データのアドレス，受信 PU 内でのデータ格納アドレス，データ長，ストライド転送に関する情報といった，各種属性が設けられている．これらの属性は NIA の管理下の特定のメモリに格納され，NIA によって管理されている．これらのメッセージ制御情報を TCW (Transfer Control Word) と呼ぶ．1 つの長いメッセージが複数のパケットに分割される場合，パケット数分だけの TCW が作成される．これらの TCW は互いに論理的に連結されており，各 TCW には次に送出すべき TCW へのポインタがデータ構造として格納されている．NIA は 1 つのパケットの送出処理が終わると，次のパケットの TCW を参照し，もしパケットがあれば，続けてそれをネットワークに送出する，という処理を繰り返す．このように，連続するパケットに対応する一連の TCW 群を自動参照し，それらのパケットを送出する機能を TCW チェイン (TCW Chain) 機能と呼ぶ．

TCW チェイン機能は，単一の長いメッセージを複数パケットに分割した場合だけでなく，各々が 1 パケット分未満の長さであるような複数の短いメッセージを連続送出する場合にも利用可能である．例えば，短いメッセージを複数の PU に個別に送出する場合，まず各メッセージに対応する TCW をそれぞれ用意し，OS に対してそれらの TCW を連結するように指示しておく．そして，先頭のメッセージに対応するパケットの TCW を NIA に与えて送出命令を送ると，NIA はそのパケットの送出を開始する．それが終わると直ちに次のメッセージに対応するパケットの送出が自動的に開始され，以下全てのパケットが順次，ネットワークに送出される．

TCW チェイン機能を用いることにより，ユーザ・アプリケーションからの 1 度だけのメッセージ送出要求により，複数のパケットを自動的に送出できる．これにより，メッセージ転送処理のオーバヘッドを短縮でき，さらに，プロセッサの動作とネットワークへのメッセージ送出処理をオーバラップすることが可能となる．この機能を適当に用いることにより，多数の短かいメッセージを一度に送信するようなアプリケーションにおいて，メッセージ送出効率を大幅に向上することが可能となっている．

13.7 キャッシュ・コヒーレンス制御

並列計算機のプロセッサ間結合ネットワークにおいて常に問題になるのが，ネットワークを通じて送受信されるデータとキャッシュ上のそれとの整合性（コヒーレンシ）問題である．一般的に，NIA はキャッシュではなくメモリ上のデータを送受信する．データを送信する際，キャッシュが write back 方式であり，かつキャッシュが dirty (キャッシュ内容とメモリ内容が一致していない) な場合，何らかの方法でこれをメモリ上にフラッシュした後で送信処理を行なう必要がある．また，データを受信する際，もし受信したデータ番地の内容が既にキャッシュ上に存在する場合，データを読み出す前にキャッシュ内容をパージする必要がある．

これらの解決法には，大別してハードウェアによる方法とソフトウェアによる方法がある．前者の場合，NIA またはその周辺回路が，メモリ上の送受信番地とキャッシュのディレクトリをチェックし，不整合が生じる可能性があれば直ちにフラッシュまたはパージ処理を自動的に行なう．後者の場合，データ送信前にプロセッサがプログラムによって送信データ番地のキャッシュ内容をフラッシュし，また受信前には受信データ番地のキャッシュ内容をパージする．

CP-PACS ではこの両者を選択できるようになっている．まず，いずれの場合でも送信データに関しては問題は生じない．なぜならば，CP-PACS のノード・プロセッサのキャッシュは，1 次・2 次共に write through 方式をとっているため，プロセッサによって書き込まれたデータはキャッシュとメモリとの間で常に整合性が保たれており，フラッシュ処理の必要がないからである．

一方，受信データに関しては整合性をとる必要がある．CP-PACS における Remote DMA 転送モードでは，メッセージ通信の対象となるデータ領域を指定する際，データ受信時の整合性を取るか否かを予め設定するようになっている．この属性はメモリのページ (4KB) 単位で管理され，NIA が外部からデータを受信する際に，この属性に従ってハードウェア制御を行なうか否かが決定される．

一般的には常にハードウェアによる整合性チェックを行なうのが無難であるが，CP-PACS で敢えてこれを行なわないモードを用意したのには以下のような理由がある．通常の RISC プロセッサでは，データのメモリへの格納及びメモリからの読み出しは常にキャッシュを通して行なわれる．従って，キャッシュとメモリとの整合性は常に取れていなければならない．これに対し，CP-PACS で採用している PVP-SW 機構においては，preload/poststore 命令により，キャッシュを介さない直接的なメモリ・アクセスが可能となっている．従って，PVP-SW によって擬似ベクトル処理されるデータに関しては，プログラムによって，それらが決してキャッシュ上に存在しないことが保証される場合があり，そのようなデータの送受信に関しては，NIA のハードウェアによるキャッシュ整合性処理は無駄なものとなる．従って，この処理を意図的に省くことにより，ネットワークからのスムーズなデータ受信が可能となり，実効的なネットワーク・スループットの向上が実現されている．このように，Remote DMA による相手 PU への直接的なデータ転送と，それに続く擬似ベクトル処理は本質的に相性がよく，CP-PACS ではこの性質を最大限に利用し，データ転送と演算処理をスムーズに連結できるようになっている．

14　CP-PACSシステム全体の構成

本節では，超並列計算機 CP-PACS を中心とした計算物理学研究センター全体のシステムについて述べる．

センター計算機システム全体の構成はパート 1 の 6 節の図 6 に，またこれらの機器の計算機棟における配置図は図 5 に，各々示されている．図 17 に各機器の概略仕様と，接続関係を示す．

図 6 において，「超並列計算機 CP-PACS」と「並列プログラム開発用システム」が，本研究によって新たに開発したものである．「並列計算機 QCDPAX」は，本新プログラム研究に先行して筑波大学にて開発されたもので，現在もセンターのネットワークに接続され，一部の計算に使用されている．その他の機器は商用製品を（一部特殊なサポートを要したものもあるが）センターのレンタル費にて導入したものである．

システム全体の構成の基本的な考え方は，以下のようなものである．

- 超並列計算機 CP-PACS には大容量の補助記憶装置を備えているが，全国共同利用センターとしての多数のユーザーのプログラムやデータなどの大量の情報からなるプライベートファイルを，全てこの CP-PACS の補助記憶装置に保持しておくには，なお，容量が不足している．したがって，

- ユーザーのプライベートファイルを保持しておくものには，非常に大容量なものが必要となることが予想されるので，CP-PACS の外部に用意しておく必要がある．

- また，CP-PACS はセンターの貴重な計算資源であるので，大規模な計算のために出来るだけ高い稼働効率で使用したい．このためには，当面は，各ユーザから投入されるジョブをバッチ的に処理する方が望ましいものと考えた．

- このように考えた理由には，シングルジョブでも並列処理そのものに起因する効率の劣化が問題であるのに，超並列計算機でマルチジョブ・マルチプログラミング処理を行った場合にプロセッサ使用効率がどのようなものになるか不明であり，また，主記憶，補助記憶装置の容量が不足することが予想されたこと，さらには，本研究期間中にマルチジョブ・マルチプログラミング環境で効率良く運用できる OS を開発することは無理であるものと予想されたこと，などが挙げられる．バッチ的な処理を行うことに起因する非効率・ターンアラウンド管理などマイナス面も有るが，運用上ある程度はカバー可能である．しかし，最終的には，CP-PACS 内部の補助記憶装置の絶対量不足が問題となるはずと思われる．

- このため，システムとしては CP-PACS の外部に FCS(Front Computer System) を設け，FCS に 大容量磁気ディスク装置，カートリッジ磁気テープライブラリ装置を接続し，さらに FCS から，センター内の LAN(FDDI) を経由してワークステーションクラスタ (ファイルサーバ）を接続，これにディスクアレイや光磁気ディスクをつなげるという形の，階層構造をもった（CP-PACS から見て）外部の記憶装置を設ける構成とした．これらの外部の記憶装置に，ユーザファイルを保管し，その管理はユーザの責任で行うこととしている．

- CP-PACS と FCS 間の大量のデータやプログラムの転送のためには，100 MB/sec の高速インターフェース (HIPPI チャネル）の接続を行っている．

- その他に，一般のワークステーション，グラフィックワークステーション系システムが，各種の作業を行うために設置してある．

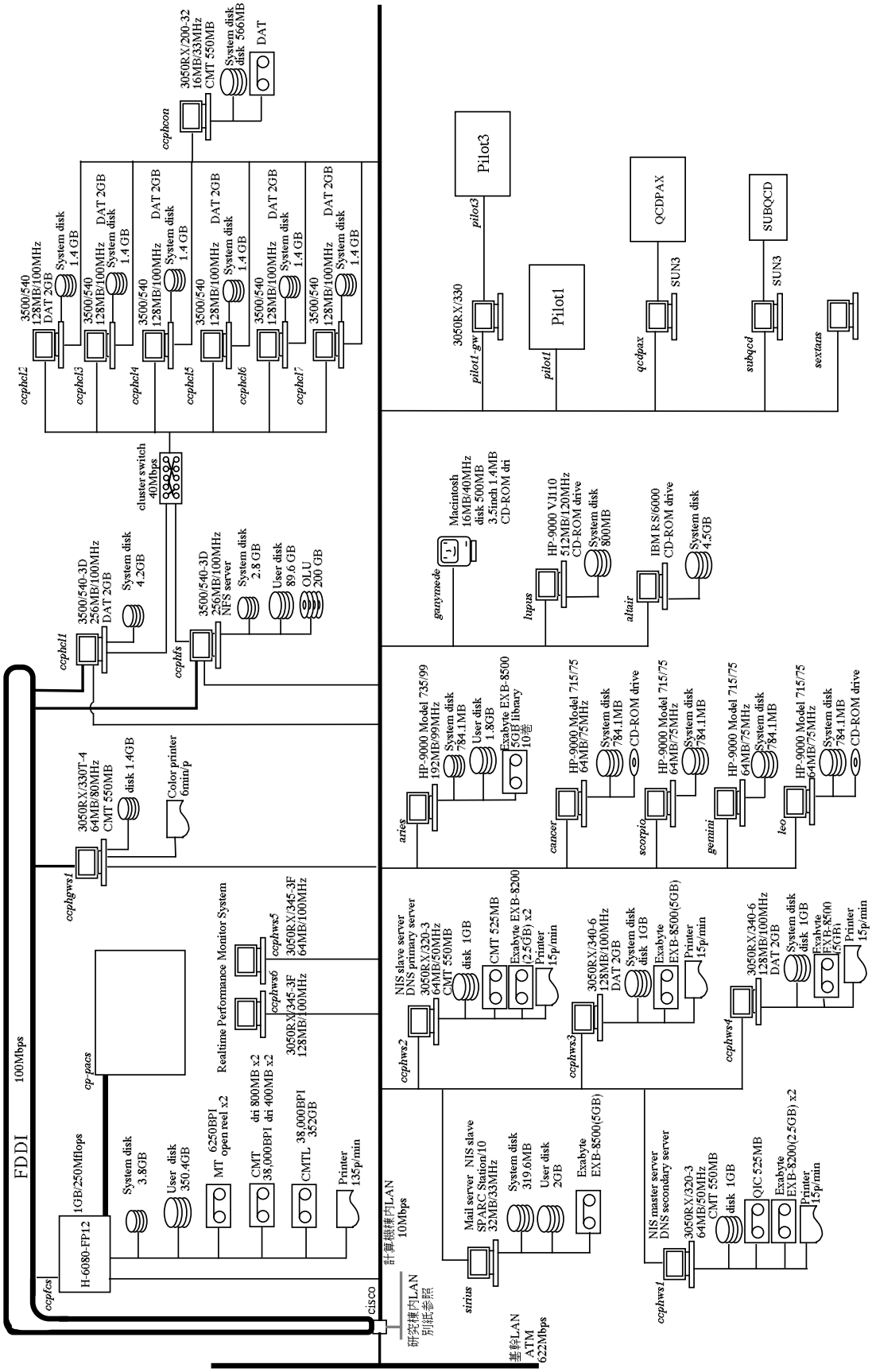


図 17: センター計算機システム接続図

- センター内のネットワークとしては，高速を要するものは FDDI(100 Mbps)，その他のものは Ethernet(10 Mbps) を使用し，センター外部とは学内基幹 LAN ATM(622 Mbps) を使用して接続されている．

ユーザが CP-PACS を使用するときの，概略の使用方法は以下の如くである．

- ジョブ投入に先立ち，ユーザはジョブの実行に必要とされるプログラムおよびデータのファイルを FCS の磁気ディスク上に用意し，それらのリストを用意する．
- このリストを指定してジョブの実行を指示する．投入されたジョブは，FCS のジョブキューに登録され，FCS のジョブキューの制御に従って，CP-PACS での実行が可能となった時点で，プログラムおよびデータのファイルが FCS から CP-PACS の補助記憶装置へ，高速インターフェース (HIPPI チャネル）を経由して送られ，実行が開始される．
- CP-PACS での処理が終了したとき，計算結果は通常は CP-PACS の補助記憶装置に得られ，ここから上記とは逆に FCS の磁気ディスクなどへ転送されることになる．
- 以上は最も単純な場合のジョブの流れであるが，あるジョブの実行途中や，ジョブの切り替わり時などには，必要に応じて FCS/CP-PACS 間でデータや制御情報のやり取りが行われる．

以上のように全体システムを構成したことにより，CP-PACS を中心としたセンターシステムの全体の運用・管理が FCS より一元的に行えるようになっている．

15 CP-PACS のハードウェアの実装

本節では，CP-PACS のハードウェアの物理的な側面，および実装方法などにつき，その概略を述べる．

15.1 ノードプロセッサ・チップ

CP-PACS 全体のアーキテクチャ，および，ノードプロセッサ･アーキテクチャの章で述べた如く，CP-PACS では，汎用の PA-RISC 1.1 仕様のアーキテクチャをベースとして，これに PVP-SW 機構を付加したマイクロプロセッサチップをノードプロセッサとして使用している．

図 18 にノードプロセッサ・チップ（チップ名称：HARP-1E）の外観及びそのフロアプランを示した．

特徴的な事として次のようなことが挙げられる．

- FR(floating point register) 数を物理的に 128 個に増やし，PVP-SW 機構を付加したこと，これによる die size の増加は，約 5%程度であったこと，
- 高速動作を行わせるために，外部接続は CCB(controlled collapsible bonding) によって行うので，CCB 用の I/O pad がチップの中央部分，アクティブ素子の上にも配置されていること，
- process technology : 0.35 μm，CMOS, AL 4 層

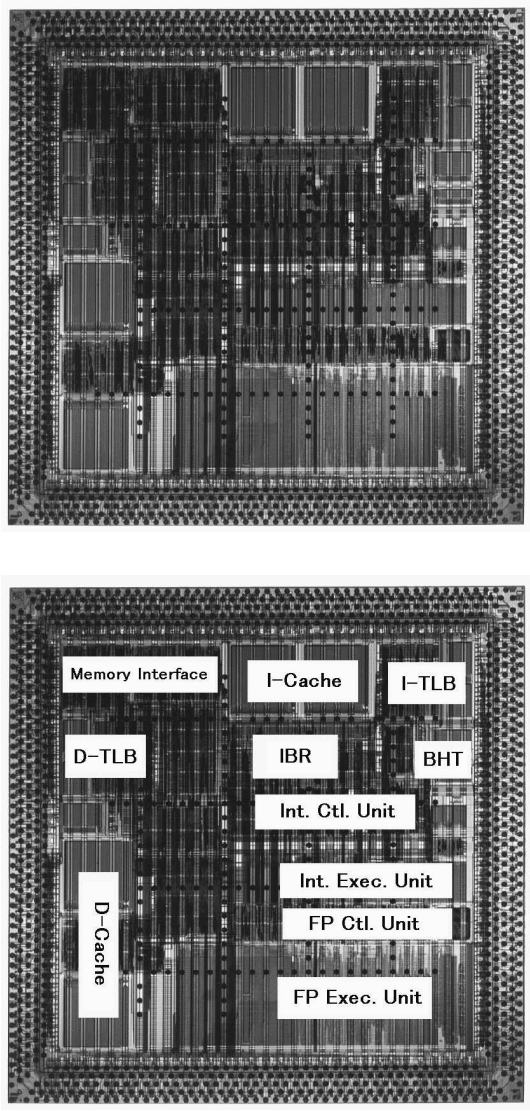


図 18: ノードプロセッサ・チップの外観

- clock frequency : 150 MHz(worst), 200 MHz(typical)
- peak performance : 300 MFLOPS
- transister count : 4.5 M
- cache memory : level 1 (on-chip) 16kB(I) + 16kB(D),
  level 2 (off-chip) 512kB(I) + 512kB(D)
- die size : 15.7 mm × 15.7 mm
- package : MCC(micro-chip carrier), 18.7 mm×18.7 mm
  1670 pin CCB (内 520 signals)
- power supply : 2.5 V
- power consumption : 10 W @ 150 MHz

15.2 プロセッサ・モジュール，主記憶，ネットワークなど

上記ノードプロセッサ・チップを IP として，これに SCU(storage control unit) 用の LSI, NIA/EX(network interface adapter/exchanger) 用の LSI，および level 2 cache 用の SRAM 12 個をまとめて，多層のセラミックキャリアに実装し，プロセッサ・モジュールを形成している．

図 19 にプロセッサ・モジュールの外観を示し，図 20 に論理的なノードプロセッサの構成を示す．

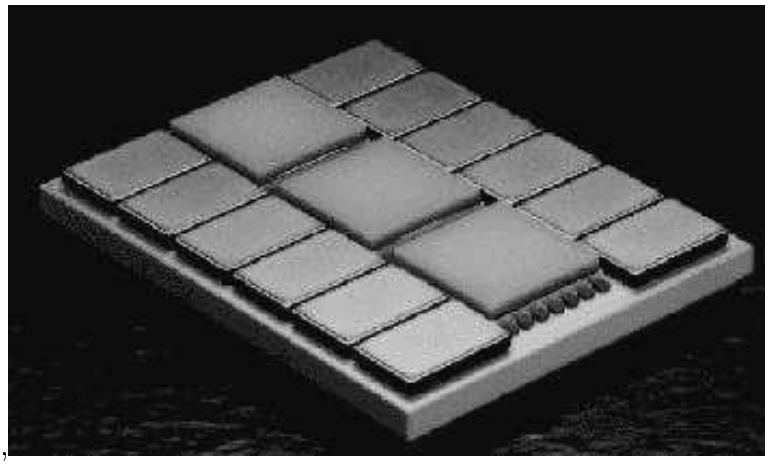
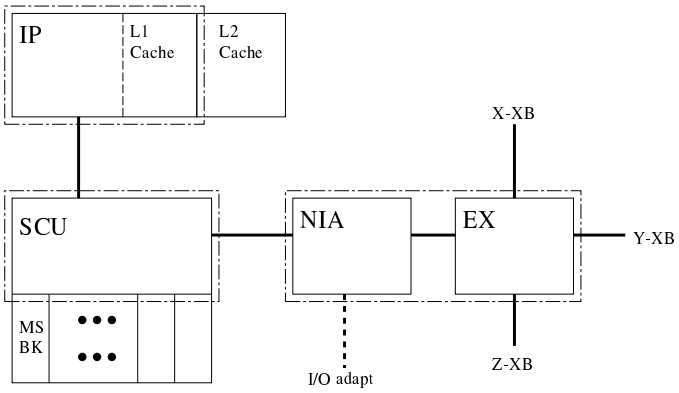


図 19: CP-PACS プロセッサ・モジュール

このモジュールに主記憶装置用の素子を接続して，1 個の PU, IOU などのノードプロセッサが構成される．

HXB(hyper-cross-bar) ネットワークを構成する　XB(cross-bar) 用の LSI は個別の LSI として後述する各階層のボードに実装されている．



IP : instruction processor
SCU : storage control unit
MS BK : main storage, bank
NIA : network interface adapter
EX : exchanger
X-XB : X-direction crossbar switch
Y-XB : Y-direction crossbar switch
Z-XB : Z-direction crossbar switch

,

図 20: CP-PACS ノードプロセッサの構成

- SCU, NIA/EX 用 LSI : 0.5 μm, CMOS GA(gate array), MCC package
- XB switch 用 LSI : 0.5 μm, CMOS GA(gate array), 42 mm×42 mm, ∼1000 pin BPGA package
- PU module base : 72 mm × 57.5 mm, 多層 (∼40) ceramic, 終端抵抗層付き, ∼700 I/O pin
- MS(main storage) : 4 Mbit DRAM-DIMM, 8 B 幅/bank × 16 bank 構成
- MS through-put : peak 1.2 GB/sec
- IP-SCU pass : 8 B 幅 (IP→SCU) ＋ 8 B 幅 (SCU→IP) / 150 MHz
- SCU-NIA/EX pass : 2 B 幅 (SCU→NIA) ＋ 2 B 幅 (NIA→SCU) / 150 MHz
- network interface : 2 B 幅 / 150 MHz, wave pipeline 利用

15.3 PU ボード，プラッタ，筐体実装

ノードプロセッサは，8 個ずつまとめられて PU ボードを構成する．図 21 は PU ボードの外観である．PU ボードに実装されたプロセッサ・モジュールなどの LSI には，強制空冷のための放熱フィンを取り付けてある．

PU ボードには，ノードプロセッサ以外にも X 軸方向の XB switch 用 LSI, clock 分配用 LSI, RAS(保守）用 LSI が搭載されている．

また，各 PU ボードの両サイドには，main edge と sub-edge の connector が用意されていて，Y 軸方向，Z 軸方向の network を形成するために使用されている．IOU の場合には，I/O adapter

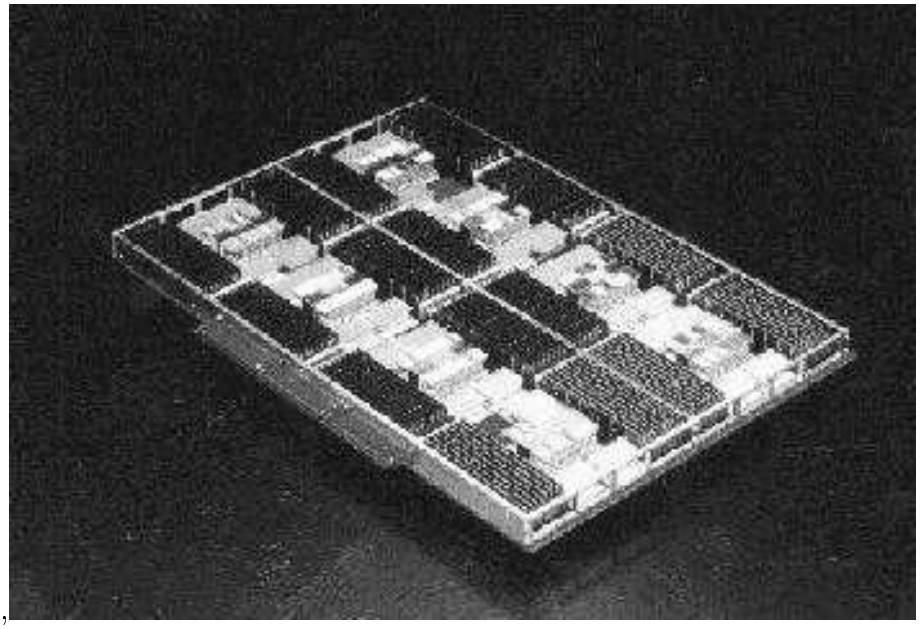


図 21: CP-PACS PU ボード

への接続も sub-edge connector からの cable によって行われる．

図 22 は，CP-PACS の論理的な PU, HXB network を構成する各要素が，筐体にどのように実装されているかを示す概念図である．

各 PU 筐体内には，上下 2 段に PP プラッタ (platter）と呼ばれるマザーボートが装着されていて，PU ボードは main edge connector を経由して，この PP プラッタに挿入される．1 枚の PP プラッタには，17 枚の PU ボードを搭載でき，Y 軸方向の XB switch 用 LSI は，PP プラッタ上に実装されている．

従って，8×17 = 128 PU + 8 IOU の 2 次元の HXB ネットワーク構成の並列計算機は 1 枚の PP プラッタで完結することになる．

Z 筐体内には，Z プラッタが 4 枚実装されており，このプラッタ上に Z 軸方向の XB switch 用 LSI が搭載されている．PU ボードの sub-edge connector からの cable は，Z プラッタの適宜の場所に接続され，Z 軸方向の XB switch を形成している．

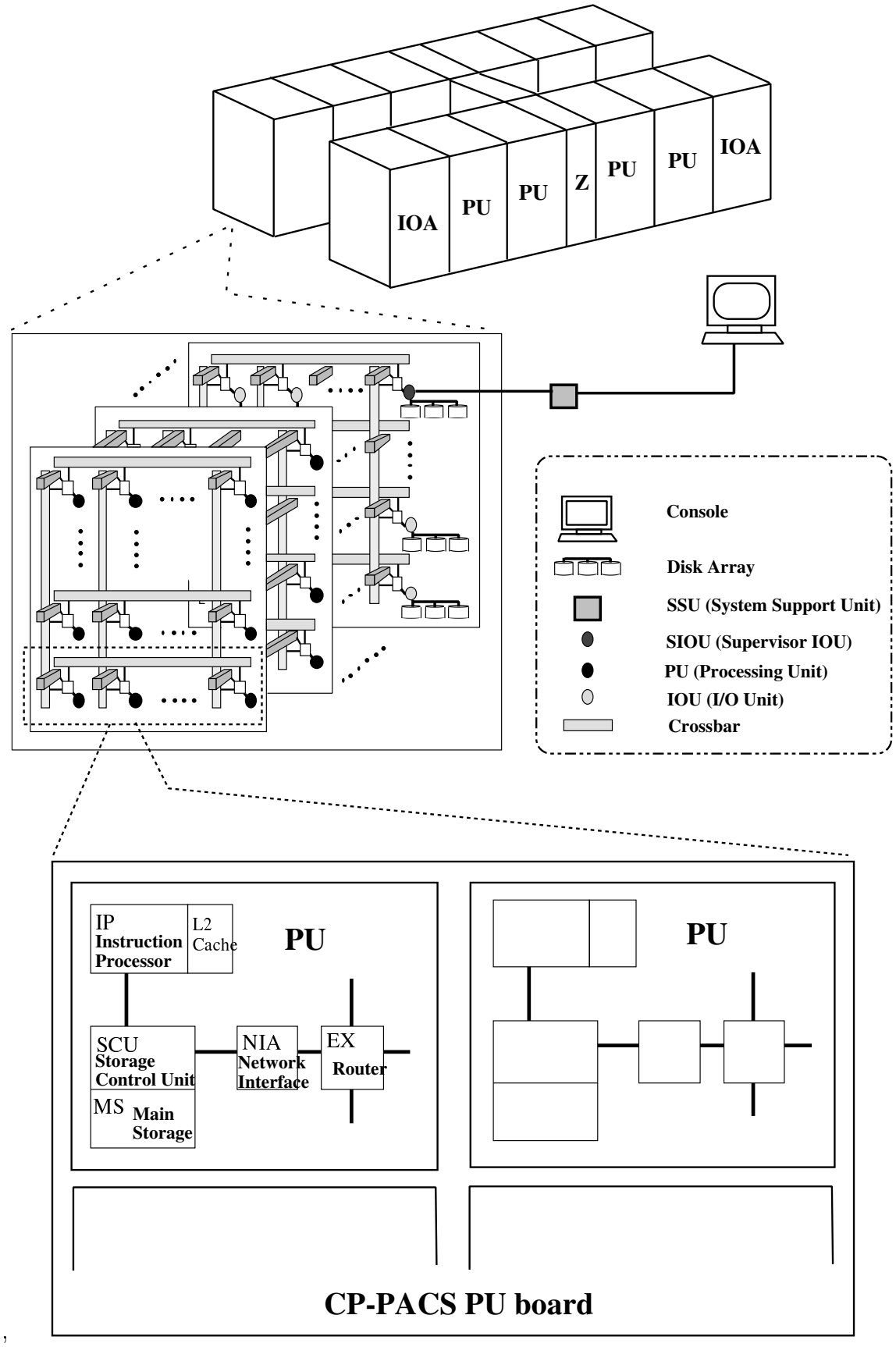


図 22: PU ボード及び匡体実装の概念図

・PU board : 8 個の PU/IOU node processor, X-XB, その他 LSI
406 mm × 625 mm, 多層 (〜40) print 基板,
main edge：250 pin × 3 個,
sub-edge ： 60 pin × 12 個

・PP-platter : 17 枚の PU board, 32 個の Y-XB switch 用 LSI 実装,
128 PU ＋ 8 IOU ＋ Y-XB switch を収納
575 mm × 730 mm, 多層 (〜35) print 基板,

・PU 筐体 : 2 枚の PP-platter, DC 電源，blower/fan 等実装,
256 PU + 16 IOU + Y-XB switch を収納

・Z-platter : 136 個の Z-XB switch 用 LSI, その他 LSI,
接続用の connector を 実装

・CP-PACS 全体 system : size : 7.0 m(W) × 4.2 m(D) × 2.0 m(H)
power : 高負荷稼働時 約 275 kW(約 300 kVA)

IOA 筐体には，主として補助記憶装置（RAID-5 磁気ディスク装置）接続用の I/O adapter board を収納している．RAID-5 接続のインターフェースは，標準の SCSI interface である．RAID-5 仕様の磁気ディスクそのものは，別の独立筐体に実装されている．

各筐体内には，当然，ＤＣ電源，強制空冷用のブロア，ファンも装備されている．冷却方式は，床下空冷方式である．

図 23 が 2048 PU+128 IOU からなる CP-PACS 全体の外観，および扉をはずした内部の写真である．

図 23: CP-PACS 外観

16　ソフトウェア

CP-PACS ソフトウェアシステムの課題は，ハードウェアの持つ性能を最大限に引き出せるようにすることと，使いやすい環境を提供することである．ノードプロセッサの性能を引きだすためには，コンパイラによる最適化が重要である．また，CP-PACS のような分散メモリ型超並列計算機では，データ転送を高速化することが重要である．以下では，基本 OS とプログラミング環境について，どのように設計し，実現したかを述べる [11].

CP-PACS のソフトウェア体系と各ソフトウェアの機能は次のようになっている．

1. OS
   (a) 各 PU の基本 OS
      - UNIX 系の Mach をベース
   (b) 並列実行 OS 機能
      - PU 群、プロセス、ファイルに関する機能
   (c) プロセス間通信機能
      - リモート DMA，PVM，MPI
2. プログラミング環境
   (a) プログラミング言語
      - C, Fortran, HPF, C++, アセンブラ言語
   (b) プログラム開発・デバッグ機能
      - ワークステーションによるコンパイルとデバッグ
      - 標準的な並列プログラムインターフェース
      - パフォーマンスモニタ

以下，このソフトウェア体系の中での重要項目について，設計目標，達成した機能概要などを記す．

16.1　基本 OS

16.1.1　設計目標

科学技術計算向けの分散メモリ型アーキテクチャの並列計算機におけるＯＳの開発方針として，次の 4 点を実現することを主たる目標とした．

1. 1 システムイメージで使えること: 並列計算機をあたかも一つの計算機として扱うことができること．プログラムがどのプロセッサで実行されるかに依存することなく必要な OS インタフェース (ここでは UNIX インタフェース) を利用できること．これにより，並列プログラムの開発が容易になる．
2. 標準的な OS であること: このことと 1. より，従来のプログラムが並列計算機上でほとんどそのまま利用可能となり，他機種のプログラムの移植性も高くなる．

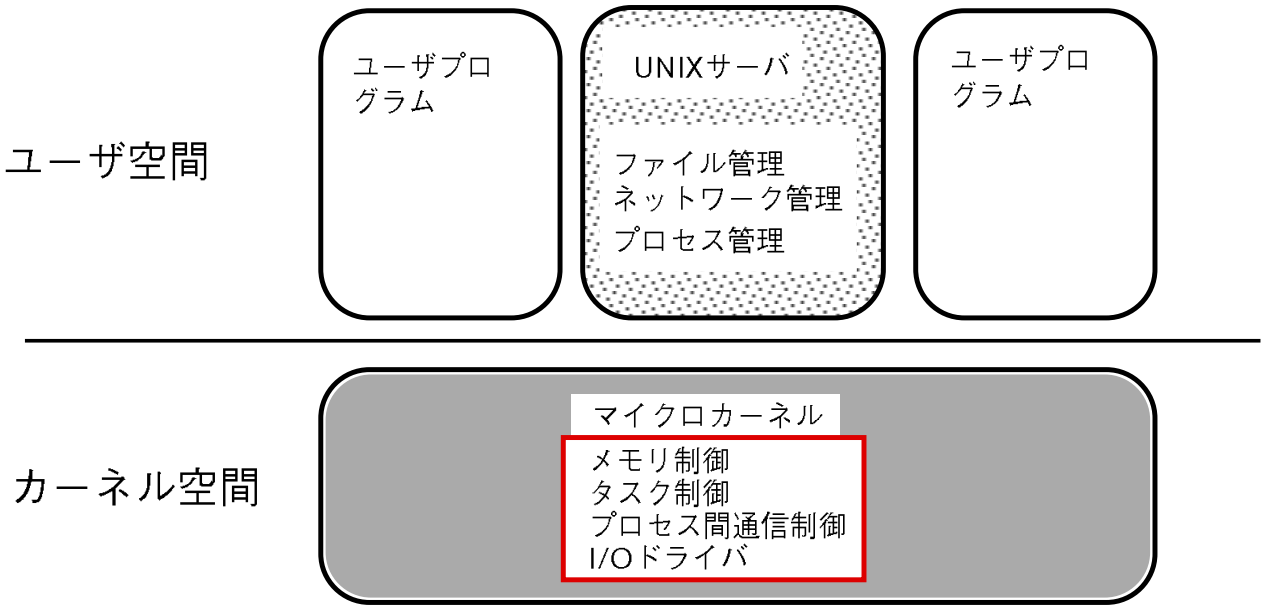


図 24: マイクロカーネル&サーバ構成

3. スケーラビリティが高いこと: プロセッサ数を増加させるのが容易であり，増加したらそれに応じて性能が向上すること．

4. 軽い OS であること: ユーザの使用可能なメモリ容量を大きくするために，OS 占有メモリ容量を小さく抑さえることなどにより，OS の存在や介入による性能低下を最小限にすること．

16.1.2 OS 機能概要

上記の要件を満たすために，マイクロカーネル&サーバ構成の OS とした．マイクロカーネルはカーネギ・メロン大学で開発された Mach 3.0 をベースとしている．構成を図 24 に示す．

カーネル空間上には，OS の機能のうち，特にハードウェアに依存する部分の大きいものを実装している．そのため，メモリ制御機能，タスク制御機能，プロセス間通信制御機能，入出力ドライバといった機能を持つ．ユーザに提供する一般的な UNIX の機能や，並列プログラムの実行制御機能などは，ユーザ空間と同じレベルに実装され，UNIX サーバとして機能する．

図 25 に，これを分散メモリ型のアーキテクチャに適用した場合の従来の方法との比較を示す．

従来の UNIX システムをそのまま分散メモリ型のアーキテクチャに対応させるとすれば，socket などの重い通信機能を用いてプロセッサ間の通信を行う必要があり，しかも，各プロセッサ上には，UNIX システム全体を必要の有無にかかわらず実装する必要があり，メモリ効率もよくない．これに対し，マイクロカーネル&サーバ構成の場合，計算専用のプロセッサ上には OS の最小限の機能であるマイクロカーネルだけを実装し，必要に応じてサーバが実装されているプロセッサと通信することにより UNIX のサービスを受ければよい．これにより，計算専用のプロセッサに於ては，OS 占有メモリを削減することが可能となる．また，サーバ，およびマイクロカーネルを付加的に実装することで，プロセッサ数を容易に大きくすることができ，スケーラビリティを向上させることができる．

図 26 には，実際のハードウェア上にマイクロカーネルおよびサーバをどのように配置しているかを示す．各計算プロセッサ (PU) にはマイクロカーネルのみを，サーバノード (SIOU/IOU)

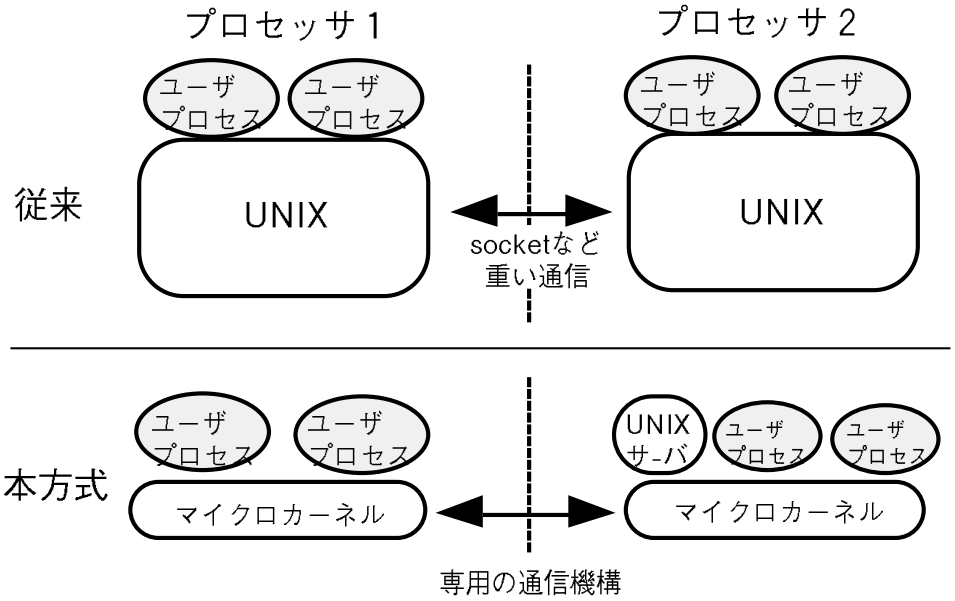


図 25: 分散 OS としてのマイクロカーネル

には，各々UNIX サーバ (プロセスサーバ，ネットワークサーバ，ファイルサーバ) を実装し，各種のサービスを提供する．

以上のような設計の結果，初期のシステムでは計算専用のプロセッサ上のマイクロカーネルを比較的小さくすることはできたが，計算プロセッサの主メモリ容量 64 メガバイトに比較すると必ずしも小さいとは言えず，さらに多くのきめ細かな削減の努力をして，結局マイクロカーネルの占有メモリ量を約 10 メガバイトとすることが出来た．

16.2 並列実行機能

16.2.1 設計目標

複数のユーザが同時に利用可能であり，そのそれぞれの使い方に応じて最適なプロセッサの割り当てが可能で，お互いに他のユーザの影響を受けないようにすることを目標とする．

16.2.2 機能概要

1. ノード群分割機能: 大規模並列計算や小規模並列によるデバッグなどの使用目的別にノード群に分割する機能をもつ．分割したノード群をノードパーティションと呼ぶ．ユーザプログラムはノード群を指定して，その中で実行される．ノードパーティション毎に，それを使えるユーザ（使用可能ユーザ），一つのユーザプログラムに割り当てるノード数，同時実行されるユーザプログラム数などを制限できる．

2. ノード割当て／保護機能: ユーザプログラムはノードパーティション（ノード群）を指定してノード割当を要求し，システムがその時の稼働状況に応じて最適なノードを割当てる（hmpp_nalloc）．その際ノードの絶対アドレスを指定することは出来ない．このことと上記 1. の制限により，不当なノードの使用を防止できる．この割り当てたノードの集まりを

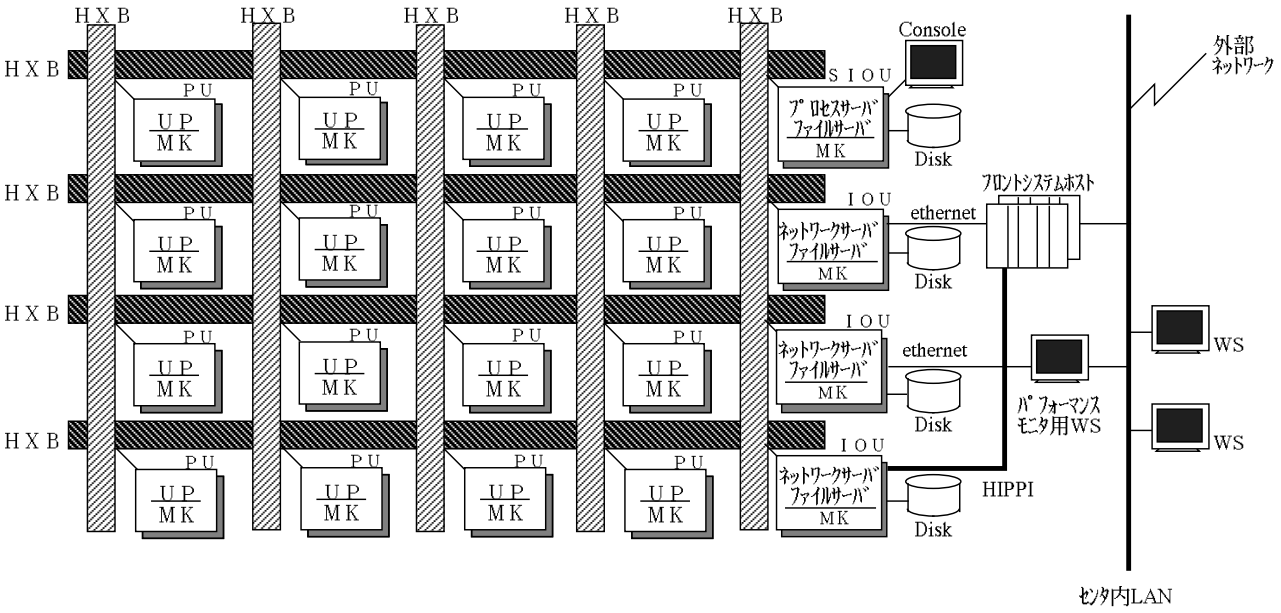


図 26: マイクロカーネル&サーバの実装

ノードグループと呼ぶ.

図 27 にパーティション設定の例と，プログラム実行の様子を示す．2048 台の PU からなる CP-PACS では，標準的なパーティションとしては，下記の 3 種類のパーティションを適宜切り替えて運用している（数字は PU の台数を表し，1 つの数字が 1 つのパーティションをなしている）.

- 2048
- 1024 + 512 + 256 + 256
- 1024 + 512 + 256 + 64 + 64 + 64 + 64

16.3 並列ファイル機能

16.3.1 設計目標

CP-PACS ではファイル入出力操作も並列実行できるようにするため，1024 台ないし 2048 台のプロセッサに対して，64 台ないし 128 台のプロセッサがディスク装置を持つ構成になっている (ディスク装置を持つノードプロセッサを IOU と呼ぶ). その構成を生かすファイルシステムとしては，すべての IOU 上のファイルが全体として 1 つのファイルシステムのように見える必要があり，しかも，複数のファイルへの同時アクセスが並列実行されるだけでなく，一つの論理ファイルへのアクセスも並列実行できるようにする必要がある.

16.3.2 機能概要

上記の目標を以下のようにして実現した.

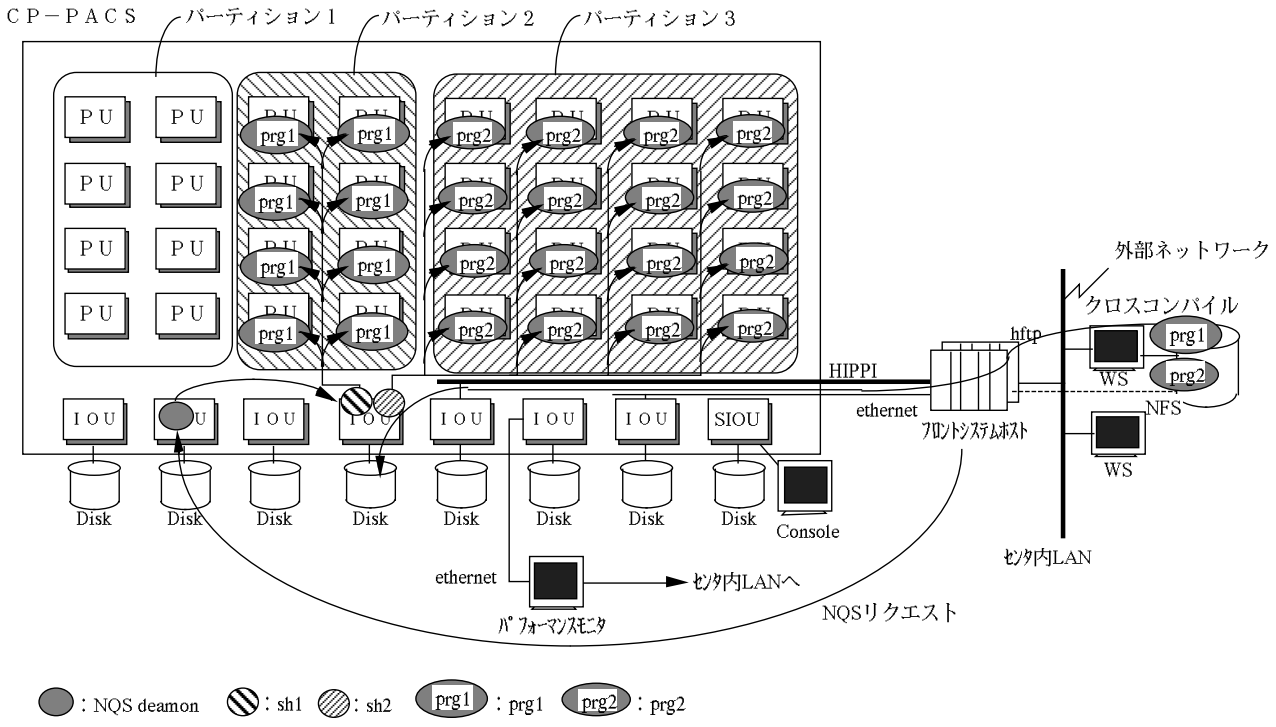


図 27: パーティションの設定と，プログラム実行の様子

1. １ツリーのファイルシステム: 複数のノード（IOU）の記憶媒体上のファイルを全体として一つのツリーに見せるため，各ノード上のファイルシステムは全体のファイルシステムの一つのディレクトリとして扱う．

2. ファイルの分散配置機能: 一つの論理ファイルを複数のノード（IOU）に分散配置出来るようにするため，各ノードディレクトリ（そのノード上のファイルシステムを一つのディレクトリと見たもの）の中で同一のファイル名（path 名）を待つものが一つの論理ファイルを成すものとする．たとえばノード a と b に対応するノードディレクトリ n_a と n_b について，n_a/usr_1/file_1 と n_b/usr_1/file_1 は一つの論理ファイル usr_1/file_1 の一部がノード a と b に分散して配置されているものとみなす．したがって，ファイルを適当な IOU に分散配置することは，適当なファイル名を選択することで実現できる．

16.4 高速ファイル転送機能

16.4.1 概要

CP-PACS では，図 28 に示すようにフロントコンピュータシステム (FCS) にユーザのファイルを置くこととし，CP-PACS と FCS との間では，初期データ，計算結果，中間結果等のファイル転送が行われる．データ量としては、数十メガ〜数十ギガバイトの複数ファイルの転送であり，短時間で大容量のデータ転送するための高速ファイル転送機能が必要である．そこで，以下の特長を持つ高速ファイル転送機能 (HFTP) をもうけることにした．

1. FCS と PPS 間を高速なネットワークで接続: HIPPI(High Performance Parallel Interface) を用いて FCS と PPS を接続する．

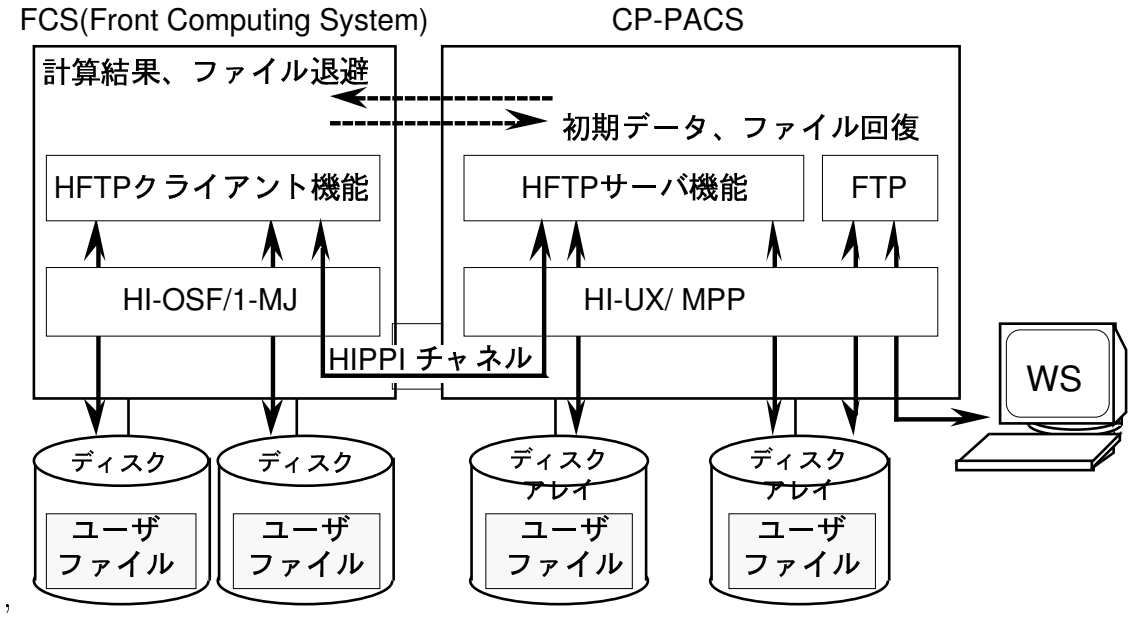


図 28: FCS と CP-PACS の間のファイル転送

2. 複数の大容量のファイルを一括して高速に転送: 複数ディスク装置に分散配置される複数ファイルを一括指定し，同時並列に転送することにより高スループットのファイル転送を実現する．

3. 専用プロトコルによる高速データ転送: データコピーレスの専用通信プロトコルを用いることにより，汎用プロトコル (TCP/IP) では実現できていない実効性能の高いデータ転送を実現する．

4. 標準ファイル転送プログラム (FTP) と同様のユーザインタフェース: HFTP におけるユーザインタフェースは，UNIX システムの標準となっている FTP コマンドを採用することにより，従来システムに近いイメージでの利用が可能になる．

当初の目標性能は，最高で 50 メガバイト/秒であったが，それは達成された．

16.5 高速通信機能

16.5.1 概要

分散メモリ型の超並列マシンによる大規模並列処理では，ネットワークを通じたノード間通信の高速化が最重要課題の一つである．

通常のノード間通信では，確実にデータを転送するため転送データをパケットに分割してヘッダを付加したり，受信したパケットからヘッダを削除してデータを組み立てる処理を行う．これらの処理を行うために，OS のカーネル内で転送データのバッファリングが必要となり，その結果カーネルーユーザ空間間でのデータコピーが発生し，これがノード間通信の性能を低下させる原因となっている（図 29 の通常通信）．そこで，より高速な通信を以下のリモート DMA 通信と呼ぶ機能で実現した．

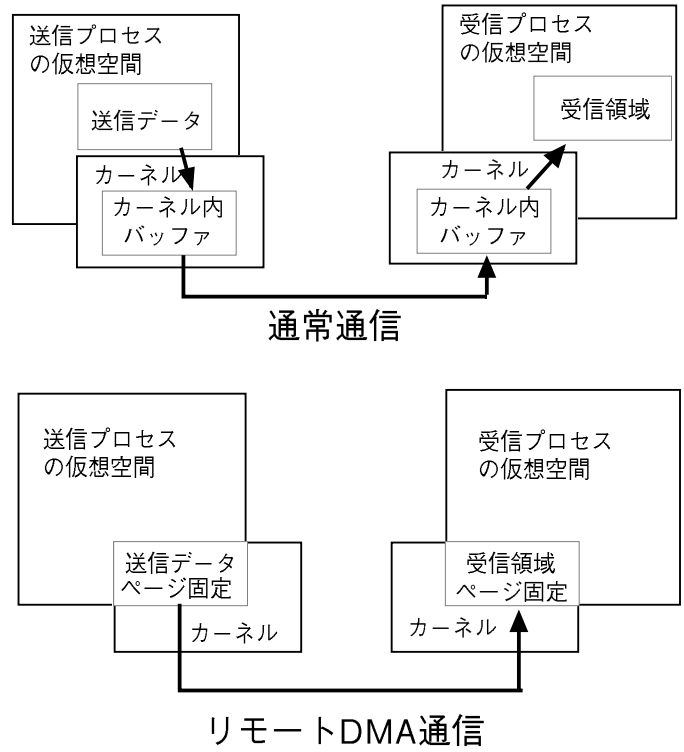


図 29: 通常通信とリモート DMA 通信

リモート DMA 通信は，送信側／受信側ともにユーザの仮想アドレス空間の一部をリアルメモリ上に固定的に割り当てておき，それらのメモリ間でデータ転送を行う高速通信方式である（図 29 のリモート DMA 通信）．異なったノード上のユーザの仮想アドレス空間間で直接データ転送を行うため，カーネルーユーザ空間間でのデータコピーが発生せず，ネットワーク性能をそのままユーザに提供する．このリアルメモリ上に固定的に割り当てた領域のことは Combuf 領域と呼んでいる．

16.6 プログラミング環境

プログラミング環境とは，プログラムの作成，コンパイル，デバッグ，実行，実行結果の表示，など，プログラムの開発を支援するためのソフトウェアシステムである．プログラム開発が困難であるといわれる超並列計算機用としては以下のような種々のシステムによる開発支援が必要になる．

1. セルフ・システムまたは実機システム: 目的とする並列計算機上のもの
2. クロス・システム: ホスト計算機またはワークステーション上のソフトウェアであり，実機上で動くプログラムを生成する．
3. シミュレーション・システム: ワークステーション上で実機での動きをシミュレートするシステム

また，メッセージ通信などを意識するのが苦手な一般ユーザでも比較的容易にプログラム開発が出来ることが必要であるし，マシンの性能を最大限に生かそうとするエキスパートにも使いやす

い環境を提供する必要がある．以下にそれらをどのように実現したか述べる．

16.7 プログラム言語

プログラム言語としては，アセンブラ言語，C，C++，Fortran，HPF(High Performance Fortran) を備えた．超並列計算機用のプログラムを書くのは容易ではない．メッセージ通信などを意識せずにプログラムを書く一般ユーザは，通常の Fortran プログラムにデータの分散配置を指定するだけで並列プログラムが出来る HPF のような言語が必要である．エキスパートには，C や Fortran で並列を意識したプログラムを書き，それに並列に実行されるプロセス間でのメッセージ通信の命令 (手続き呼びだし) を入れられるようにする必要がある．いずれの場合も，並列計算機の性能を最大限に引きだすには，コンパイラによる徹底した最適化が必要であり，各種の最適化の方式を検討した．しかし，コンパイラによる最適化にも限度があるので，アセンブラ言語でのプログラミングも可能とし，そのプログラムのデバッグシステムも考慮する必要がある．以上が，これらの言語を備えることにした理由である．なお，これらの言語については，ワークステーション上のクロスコンパイラと実機上のセルフコンパイラの両方を開発した．

コンパイラによる最適化として CP-PACS で特徴的なのは，擬似ベクトル機能を持ったスーパースカラ型プロセッサであるノードプロセッサのための最適化である． それについては別の節で述べる．

16.8 プログラム開発・デバッグ機能

プログラム開発用のツールとしては，エディタ，コンパイラ，デバッガ，性能モニタなどがある．ここでは，並列計算機で重要なものとなる並列プログラム開発環境，デバッガとパフォーマンスモニタについて述べる．

並列プログラム開発環境としては，CP-PACS 独自のものを開発するよりも，すでに世の中でよく使われているものを提供する方がよいと考え，まず，米国 ParaSoft 社で開発された Parallelware を採用することにした（米国名は Express）．また，パブリックドメインの標準となりつつあった PVM(Parallel Virtual Machine) インタフェースもサポートすることにした．さらに，その後，標準のインターフェースとして開発された MPI（Message-Passing Interface）が広く受け入れられるようになってきたので，それもサポートしている．

16.8.1 Parallelware

Parallelware は，米国 ParaSoft 社で開発された並列プログラム開発環境ソフトウェアであり (米国での名称は Express)，この種の商用ソフトウェアとしては広く使われているものである．Parallelware では，ユーザプログラムに並列処理用の拡張関数呼び出しを組み込むことによって並列プログラムを可能とする．記述言語は C か Fortran である．

Parallelware には CP-PACS 上のものとネットワークで結合されたワークステーションクラスタ上のものがあるので，後者でデバッグしてから CP- PACS 上で実行させることが出来る．

機能としては次の 3 つの機能がある．

1. パフォーマンスモニタ: プロセス間通信のオーバヘッドや OS のシステムコール，入出力などの占める時間を解析するための CTOOL，OS やユーザのイベント (同期制御，通信など) のモニタリングシステムである ETOOL，サブルーチンごと，ソースプログラム文ごとの

CPU 消費時間を解析するための XTOOL がある．これらの解析は，並列プログラム実行後にワークステーション上で行う．

2. シンボリックデバッガ(NDB): ブレークポイントの設定，シングルステップ単位の実行，変数／配列やレジスタの値の表示などが並列実行時に出来る．

16.8.2 パフォーマンスモニタ

パフォーマンスモニタ（Parallelware の中のパフォーマンスモニタとは別のものである）は CP-PACS ハードウェアの稼働状況をリアルタイムでモニタするものであり，性能チューニングのための情報を出力するものである．アプリケーションプログラムの並列実行時にリアルタイムで各ノードの CPU 使用率，ネットワーク負荷，仮想メモリ使用率などの稼働状況をディスプレイ上（図 26 や図 27 のパフォーマンスモニタと書かれたワークステーション上）にビジュアル表示する．

その機能は下記の 4 つのモニタに分類されている．

1. CPU 稼働状況モニタ: 各ノードのシステム／ユーザ使用時間，使用中ページ総数，ページイン／アウト回数などをモニタリングする．

2. IO 稼働状況モニタ: 各ノードの read ／ write 転送回数，転送容量，サービス時間など，HXB を除く入出力機器稼働状況をモニタリングする．

3. ネットワーク稼働状況モニタ (通信モニタ): 送／受信パケット数，バリア同期回数，Combuf 送／受信回数など，HXB のトラフィックをモニタリングする．

4. ハードウェアモニタ: キャッシュミスヒット率，バスビジー率などプロセッサ性能およびメモリインタフェース性能をモニタリングする．

16.8.3 初期の開発支援ツール

システムが完成してからはあまり使われないが，開発時に有効な幾つかのツールも開発した．それらのうちシミュレータとリモートカーネルデバッガについて触れる．

シミュレータは，CP-PACS 特有の機能であるリモート DMA と擬似ベクトル命令をワークステーションでシミュレートするために開発した．擬似ベクトル命令については，モジュールごとの実行回数・実行命令数の表示，ループごとの実行命令数表示，マシンサイクル数表示，メモリアクセス回数／キャッシュミス率表示，配列の参照箇所のグラフィカルな表示などを行う機能を持たせた．

リモートカーネルデバッガは，ワークステーションなどをホストとして，並列計算機 (ターゲットマシン) のプログラムをデバッグするためのものであり，OS のデバッグも出来るものとした．そのために，ホストマシン上から並列計算機上のプログラムの実行制御をしたり，メモリやレジスタの内容の参照や更新が出来るようにした．

17 擬似ベクトル化コンパイラ

擬似ベクトル・プロセッサに適したオブジェクト・コードを自動生成するコンパイラを擬似ベクトル化コンパイラと呼んでいる．擬似ベクトル化コンパイラによるコード生成/最適化は，ソフト

ウェア・パイプライニングの手法を擬似ベクトル・プロセッサ用に改良したものである．コード生成/最適化の基本アルゴリズムは平成７年度に検討を終了しており，それを CP-PACS 用 Fortran コンパイラおよび C コンパイラに実装し，その有効性は既に確認済みである．

ここではまず，その基本アルゴリズムを紹介し，その後に基本アルゴリズムの問題，課題について触れる．

17.1 基本アルゴリズム

基本アルゴリズムを例を用いて示す．以下のプログラム片

```
for(int i = LOW; i < HIGH; i++){
    tmp = c[i]*(tmp+d[i-1]);
    a[i] = tmp;
    b[i] = d[i];
}
```

は，２命令同時発行可能なスーパスカラプロセッサでは，次のような目的コードへ変換される．ただし，命令を読みやすくするために，通常の機械語表現ではなく，代入文に模した表記を用いた．また，ここには定常的な繰り返し部分 (カーネル部) だけ示してある．最後の命令 CondBranch はループのインデックスを増加し終了判定をする命令である．なお，ここでは簡単のため，遅延分岐は考慮してないが，実際のコンパイラでは勿論それも考慮している．

```
実行サイクル              | 同時に実行される命令
0: b[i-1]:= R(30)       | R(29):= R(28)+R(30)
1: a[i-1]:= R(28)
2: R(125):= c[i+47]     | R(30):= R(31)*R(29)
3: R(124):= d[i+46]
4: Slide += -2
5: CondBranch
```

このプログラムは以下のような考えで生成されている．まず，もとのプログラムのループ１回分の実行命令を考えてみる（図 30 の実線で囲った部分）．最初に，次の２つの preload 命令で，現在のウィンドウでは見えていない遠くのレジスタに c と d をロードする（R(0) から R(31) までが見えているレジスタである）．

```
R(125):= c[i]
R(124):= d[i-1]
```

この後で，ウィンドウをスライドさせる命令「Slide += -2」を１回実行すると，上の命令の対象となっているレジスタの論理番号は R(125)，R(124) から R(123)，R(122) へと変化する．スライド命令を 47 回実行したときに，これらは R(31)，R(30) となってウィンドウの中に入るので「tmp = c[i]*(tmp+d[i-1]);」の演算を実行することが出来る（メモリのレーテンシが相当大きくてもそれまでにはロードが完了しているはずである）．その命令が「R(29):= R(28)+R(30)」と「R(30):= R(31)*R(29)」である．ここでは，加算のレーテンシは２としている．前者の命令の

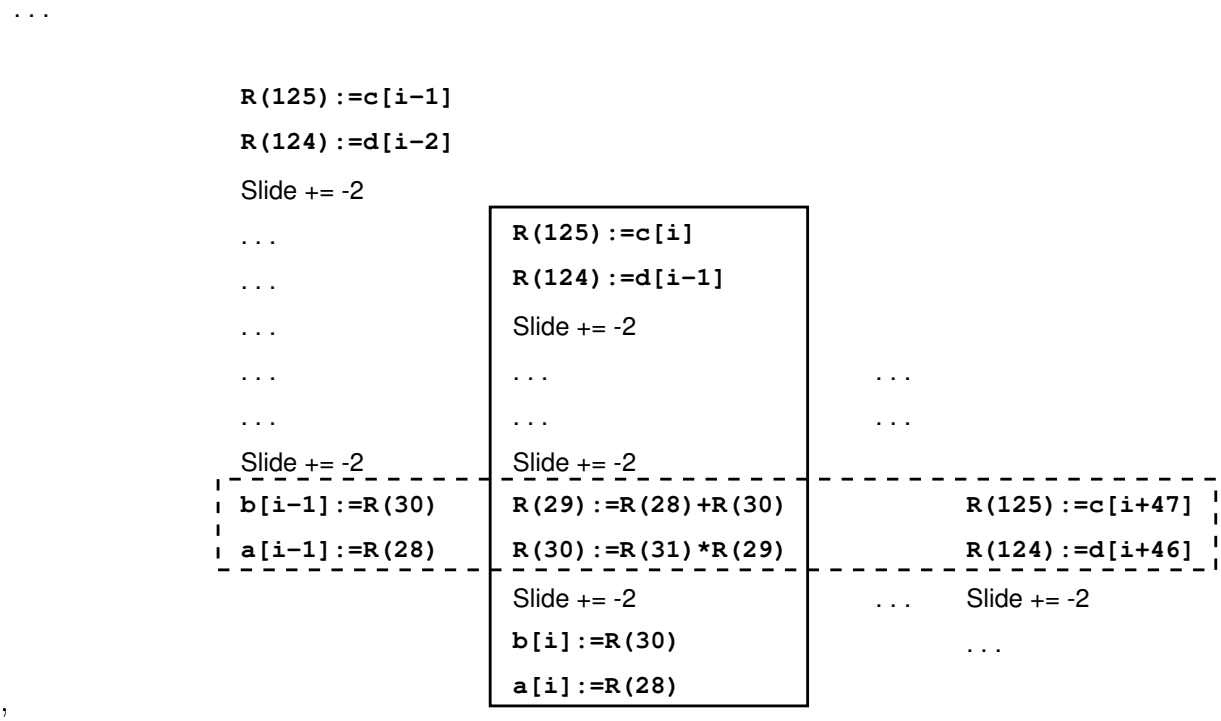


図 30: ループ 1 回分の実行命令と kernel

R(30) は d[i- 1] の値であり，後者の命令の R(30) は tmp のレジスタである．この R(30) はスライド命令を 1 回実行すると R(28) となる（それが前者の命令で R(28)=tmp として使われている）．したがって「a[i] = tmp;」は次のスライド命令を実行してから「a[i]:= R(28)」とすればよい．

以上の命令の繰り返し実行を滞りなく実行するためには，c[i] などに関してこれらの計算をすると同時に，後の繰り返しのために c[i+47] の preload 命令等を発行し，前の繰り返しで計算した tmp の値を a[i-1] に poststore する必要がある．それら（図 30 の点線で囲った部分）を，命令の同時発行の可否，発行のタイミング等を考慮の上，詰め込んだものが前記のプログラムである．

このプログラムを生成する手続きの概略は以下のようなものである．

(1) データ依存グラフ (data dependence graph) の作成

(2) ループ立ち上げ間隔 (loop initiation interval: II と略記される) の最小値決定

(3) リソース予約テーブル (resource reservation tabel) への命令の埋め込み

(4) レジスタ割り付け

データ依存グラフはどの値がどの演算に使われるを示すグラフである．今の例ではループ伝搬 (loop carried) の依存もある．ループ立ち上げ間隔 II は定常的な繰り返し部分のサイクル数であり，上の例では 6 である．その最小値はデータ依存グラフ上の演算の個数とマシンの特性から決まる．リソース予約テーブルとしては，まず II の最小値の長さを持ったプロセッサのリソースの表（たとえば，整数演算と浮動小数点演算とロード/ストアの 3 種類の命令が同時に発行できるプロセッサでは II 行 3 列の表）を作り，そこにデータ依存グラフで示される命令を埋め込んでいく．全部の命令を埋め切れれば良いが，埋めきれなかった場合は II を増やしてやり直す．最後にレジスタ割り付けを行う．

以上は一般的な手続きであるが，CP-PACS の擬似ベクトル処理を実現するコンパイラの手続きは，以下の点で通常のソフトウェアパイプライン処理を実現するコンパイラの手続きと異なる．

まず，繰り返し処理の度に論理レジスタ番号が一定の刻み幅で変化していくから，複数のパイプライン・ステージで使用するレジスタ番号の間の干渉が容易に回避できるという利点がある．いわゆるレジスタの名前替え (register renaming) が実現されているのである．通常のソフトウェア・パイプライニングでは，このレジスタ間の干渉のため，ループ立ち上げ間隔はそのループ処理に含まれる命令の中の最大のレーテンシ以下にはできない．ほとんどの場合そのような命令はロード命令であり，そのレーテンシは相当に大きな値であるため，問題は深刻である．その対策として一般にはループ展開 (loop unrolling および modulo variable expansion) を行い，かつ命令スケジューリングをより巧妙にするが，擬似ベクトル処理ではそのような技巧を凝らす必要がないため，ループ立ち上げ間隔の算出や命令スケジューリングのアルゴリズムがごく自然な発想に基づいて素直に実現できる．ただし， レジスタ番号が変化していくためレジスタ割り当てのアルゴリズムが従来のものとは様変わりしてくる．

基本アルゴリズムは以上の通りである。これはあくまでも基本となるものであり，適用できるループ・プログラムの範囲は限定されていた．そこで以下では，この基本アルゴリズムの問題点とその後の改良方法，今後の技術課題に簡単に触れる．

17.2 基本アルゴリズムの改良

基本アルゴリズムは次のような問題点を持っていた．

1. 効率的なレジスタ割付アルゴリズムが十分解明できていない．
2. 主記憶アクセス・レーテンシの見積もり精度が低く，短ループの実行性能が低い．
3. 多重ループ等で，擬似ベクトル処理の前処理部 (プロローグ部)，後処理部 (エピローグ部) が実行性能を落とす．
4. ループ本体に条件分岐 (if-then-else 文) を含む場合に擬似ベクトル化が難しい．
5. 加算，乗算を同時に発行する複合命令が生成されず，二つの単命令を生成し，実行効率を落とす場合がある．
6. レジスタ数が不足した場合の効率的な対処アルゴリズムが解明されていない．

これらの問題点についてそれぞれ以下のような改良を行った．

レジスタ割付アルゴリズムとして，基本アルゴリズムではレジスタ干渉グラフの彩色による方法を拡張して採用していた．この方法では，スライド・ウインドウによるレジスタ番号の変化が干渉グラフの構造にどのように影響するのか見通しが悪く，レジスタ割付アルゴリズムの改良が困難であった．これに対して，改良法では，干渉グラフとは全く異なる，巡回区間グラフ (cyclic interval graph) による方法を採用した [13]．この方法では，円筒形状の表面に線分を張り付けるというモデル化を行う．円筒がループの繰り返し実行を表し，線分がデータの生存期間を表す．そして，レジスタ割付問題は複数の線分をうまく結合し，より少ない本数の輪を作る問題として表される．スライド・ウインドウを持つ擬似ベクトル・プロセッサでは，レジスタを表す輪は螺旋への拡張をすればよく，従来法からの自然な拡張が可能である．この方法の採用によって見通しのよい議論が可能となった．実際，新アルゴリズムでは，必要レジスタ数が平均 10%程度減少した．また見通しの良さはアルゴリズムの理論的解明も容易にし，必要レジスタ数の上限に関する定理を導くこともできた．この新アルゴリズムは CP-PACS 用コンパイラに実装すべく準備中である．

主記憶アクセス・レーテンシは，キャッシュがヒットしない場合さらにバンク・コンフリクトが発生した場合に極めて大きな値になると予想される．そのため，基本アルゴリズムではスライド・レジスタ・ファイルを使いきるような最大の主記憶アクセス・レーテンシを仮定していた．しかし実際のループ・プログラムの中には必ずしもレーテンシが大きくない場合もある．そのようなループが短ベクトル長の場合，レーテンシを大きく見積もり過ぎたことで，かえって実行性能が落ちてしまう．一般にコンパイラがそれを判断することは難しい．そこで，プログラマ自身がレーテンシ (あるいはそれに代わるパラメータ) をコンパイラに直接指示する機能をコンパイラに付加した．

ソフトウェア・パイプライン化されたオブジェクト・コードでは，プロローグ部とエピローグ部の命令発行パターンが相補的であり，両者を組み合わせるとカーネル部のコードが得られる．二重ループ以上の多重ループでは，最内ループのソフトウェア・パイプライン化されたコードが複数回繰り返し実行されることになるが，ひとつの最内ループのエピローグ部のコードを直後の最内ループのプロローグ部の実行を組み合わせれば，その部分はカーネル部と同等の実行効率を得ることができる [9]．コンパイラにこの機能を実装した．これによって，最大２倍の高速化が可能となった．また多重ループでなくとも，類似したループが並んでいる場合には，ひとつのループのエピローグ部のコードは，次のループのプロローグ部のコードと一部を重ねて実行できる．

以上の項目ではほぼ十分に満足できるアルゴリズムの改良およびコンパイラへの実装が可能であった．これに対して以下に述べる３項目では改良は一部にとどまり，抜本的な改良にはまだ時間がかかると予想される．

ループが条件分岐を含む場合のコード生成には高度なコンパイル技術を要する．原理的には，そのようなループに関してもソフトウェア・パイプライン化技術が適用可能であることが知られている [7] が，条件分岐を含まない場合に比べ，データ依存解析，命令スケジューリング，レジスタ割付の全てのフェーズに格段に複雑なアルゴリズムが必要となる．そのため，現段階ではコンパイラは，then/else 部に現れるロード命令を投機的に実行する処理のみを行っている．

複合命令 FMPYADD(FMPYSUB) は，加算 (減算) と乗算が同時に発行できるため，プログラム実行性能向上には欠かせない命令である．しかし，命令語長 (32bit) の制約のため，乗算は任意のレジスタ間で演算可能であるが，加算は A := A+B　(ここに A，B はレジスタを現す) の形式の演算のみが可能である．このレジスタの指定に関する制約は，レジスタ割付に大きな障害となる．現時点では，この制約のためレジスタ割付に失敗した場合には，その失敗に関連する複合命令を乗算命令と加算命令に分解して実行している．

既に述べたように，レジスタ割付アルゴリズムに関しては大きな改良を行った．しかしながら，その改良は使用レジスタ数を削減する改良であり，使用レジスタ数が不足してしまった場合の効率的な対処法 (いわゆるスピルコードの生成など) についてはまだ十分に検討できていない．

以上，擬似ベクトル化コンパイラの現状を述べた．基本アルゴリズムの検討からこれまでは，実施効果の大きな課題から優先的に取り組んできた．その意味で，残された課題項目 4，5，6 には今後，腰を据えて取り組む必要がある．

18　性能評価

本節では，ノードプロセッサ単体の性能評価，および相互結合網の性能評価を示す．具体的なアプリケーションを用いた性能評価に関しては，21 節で述べる．

18.1 ノードプロセッサの性能評価

12 節で述べた通り，CP-PACS に実装されているノードプロセッサは，擬似ベクトル処理機構が備わっている．この機構は，大規模な科学技術計算でデータキャッシュが効かないために問題となる主記憶アクセスレーテンシを隠蔽するためのものである．具体的には

- 浮動小数点データに関しては，主記憶とスライドウィンドウ化された浮動小数点レジスタの間でデータをパイプライン的に転送することでアクセスレーテンシを隠蔽する．
- 整数データに関しては，汎用レジスタについては特別の追加機構を持っていないので，データキャッシュへデータをプリフェッチすることでアクセスレーテンシを隠蔽する．

という方針を取っている．

この特徴の有効性を示すため，以下の 3 つのカーネルループで性能評価を行なった．

ADD $c(i) = a(i) + b(i)$

ADD5 $c(i) = a(i \times 5) + b(i)$

LIST-SUMUP $s = s + a(l(i))$

ここで，ベクトル a, b, c，およびスカラ s は，倍精度浮動小数点データであり，l は整数データである．ADD は，3 つのベクトルを連続にアクセスするのに対し，ADD5 では，ベクトル a に対するアクセスは stride5 の非連続アクセスになる．LIST-SUMUP では，a の要素の 10% が重複なくアクセスされるように乱数を用いてインデックスベクトル l のデータを生成した．

これらの 3 つのカーネルループに対し，以下の 3 通りのコードを生成し性能を測定した．

scalar キャッシュへのプリフェッチも擬似ベクトル処理も行なわない．

prefetch キャッシュへのプリフェッチのみを行なう．

PVP 擬似ベクトル処理を行なう．リストベクトル処理に関しては整数データであるインデックスベクトルはキャッシュへプリフェッチする．

どのコードを生成するかの切替えは，コンパイル時のオプションとして指定している．すなわち全コードはコンパイラにより生成されており，人手による最適化は行なっていない．但し scalar および prefetch では，データの配置によってはキャッシュで line conflict が頻発し性能が著しく低下するため，この 2 つのコードに対してはデータの配置を人手で最適化している [14]．

上記のコードに対し，同一コードを 2 回実行し，2 回目の実行に要する時間を測定することで性能評価を行なった．この条件では，コードサイズが十分小さいため命令キャッシュは warm start となる．また，データキャッシュを用いる scalar，および prefetch のコードにおいては，ベクトル長が短くデータサイズがキャッシュサイズ (1st:16KB, 2nd:512KB) より小さい時はデータキャッシュも warm start になる．しかしながら，並列計算機においては，他の PU から転送されたデータに対して処理を加えることが一般的である．この場合，コードが命令キャッシュにあるという仮定は自然であるが，処理対象となるデータがデータキャッシュに存在する，という仮定は成立しない．そこで，測定前に，データキャッシュを purge することで，cold start 状態にした場合の性能も測定した．結局，合わせて以下の 5 つの場合の場合の性能を測定した．PVP においては，12 章で述べた通り，メモリをアクセスする preload/poststore 命令がデータキャッシュを用いないため，warm/cold の区別はない．

scalar(cold) ：コードは scalar で，測定前にデータキャッシュを purge する．

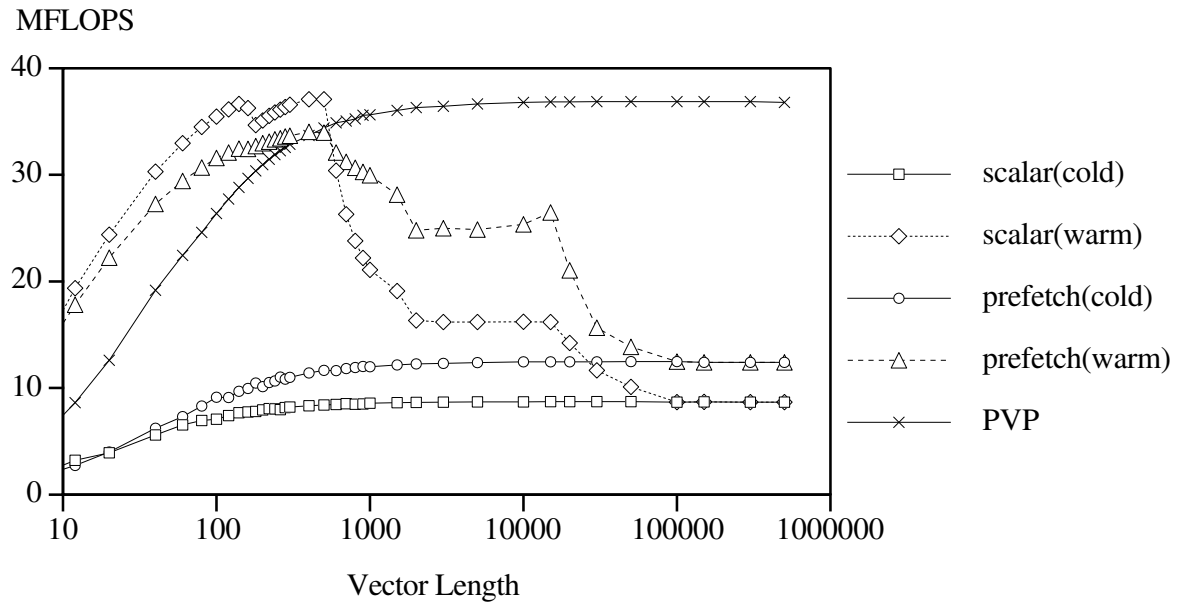
scalar(warm) ：コードは scalar で，測定前にデータキャッシュを purge しない．

prefetch(cold) ：コードは prefetch で，測定前にデータキャッシュを purge する．

prefetch(warm) ：コードは prefetch で，測定前にデータキャッシュを purge しない．

PVP PVP のコードを用いる．

図 31，図 32，図 33 に，ベクトル長を変化させた場合の 3 つのカーネルに対する性能を示す．



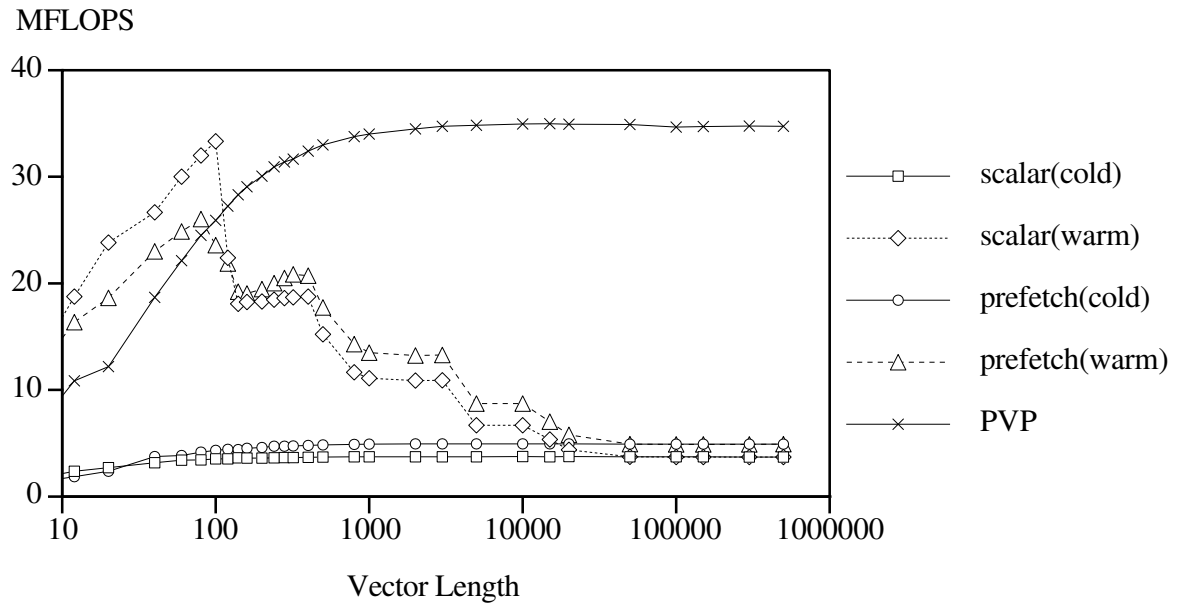
,

図 31: ADD ループの性能

これらのグラフより以下の点がわかる．

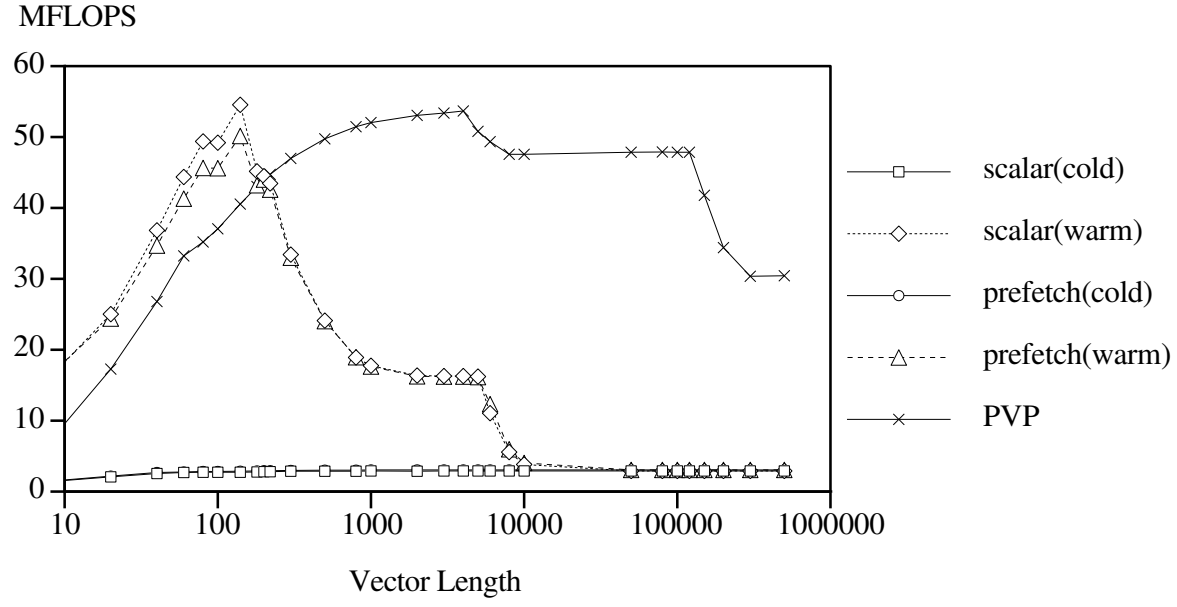
1. 図 31 よりわかるように，ベクトル長が 700 近辺以下で，データサイズがキャッシュ容量より小さい時は，scalar(warm) の性能がもっとも良く，prefetch(warm) の性能は若干低い．この理由は，prefetch においては prefetch 命令が挿入されているため，実行命令数が増えているためである．ベクトル長が短い時に，scalar(warm) や prefetch(warm) と比べて PVP の性能が低いのは，scalar(warm) と prefetch(warm) では短ベクトル長の時はデータがキャッシュに存在するため，データキャッシュのみアクセスすれば良いのに対し，PVP コードではキャッシュを利用しないため，主記憶までデータアクセスする必要があるためである．

   しかし，ベクトル長が長くなりデータサイズがキャッシュ容量よりも大きくなると scalar(warm) 及び prefetch(warm) の性能は急激に低下する．この性能低下が，ベクトル長 700 付近と 20,000 付近の 2 箇所で見られるのは，データキャッシュが 2-level であるからである．また，2 次キャッシュを溢れると prefetch の効果はさほどないこともわかる．それに対し，PVP の性能はベクトル長が長くなっても低下しない．例えば，ベクトル長が 300,000 の時，prefetch は scalar の高々1.43 倍であるが，PVP は scalar の 4.25 倍である．このことは，PVP がメモリアクセスレーテンシを効果的に隠蔽していることを示している．



,

図 32: ADD5 ループの性能



,

図 33: LIST-SUMUP ループの性能

2. 図 31 と図 32 を比較するとわかるように，非連続アクセスを含む ADD5 では，ADD よりも短いベクトル長で，scalar(warm), prefetch(warm) の性能が低下している．これは，ストライドアクセスのため短いベクトル長でデータサイズがキャッシュ容量を越えてしまうからである．また，prefetch の scalar に対する優位性が減少する．例えば，ベクトル長が 300,000 の時，prefetch は scalar の高々1.32 倍である．これに対し，PVP の優位性は増大し，PVP は scalar の 9.36 倍の性能を達成している．これは，cache を用いる scalar/prefetch では，アクセスが非連続である場合，同一ライン中の不要なデータを転送せざるを得ず，無駄なトラフィックを発生するのに対し，PVP ではたとえデータアクセスが非連続であっても必要なデータのみを転送すれば良いからである．

3. 図 33 よりわかるように，ランダムアクセスが入ると，scalar, prefetch の性能はさらに低下し，scalar に対する prefetch の優位性はほとんど無い．例えば，ベクトル長が 300,000 の時，prefetch は scalar のわずか 1.05 倍である．図 33 では，PVP の性能もベクトル長が長くなるにつれ性能低下が 2 回発生しているが，これは，整数データであるインデックス l をキャッシュへプリフェッチしているためであり，l のサイズが 1 次キャッシュ，及び 2 次キャッシュを溢れる時点で性能が低下している．しかし，それでもベクトル長が 300,000 の時，PVP は scalar の 10.5 倍もの性能を達成している．

以上の評価結果より，CP-PACS のノードプロセッサが採用する擬似ベクトル処理機構はキャッシュ容量より大きなベクトルデータを扱う場合に，主記憶アクセスレーテンシを効果的に隠蔽できていることが確認できる．

18.2 相互結合網の性能評価

本節では，様々な転送パターンに対して個々にプログラムを作成し，実測によるネットワークの性能評価を行なう．特に基本的な一対一転送性能を求めるための転送（ピンポン転送）や，実アプリケーションで多様される転送パターンを中心に測定を行ない，その結果を示す．

18.2.1 測定方法

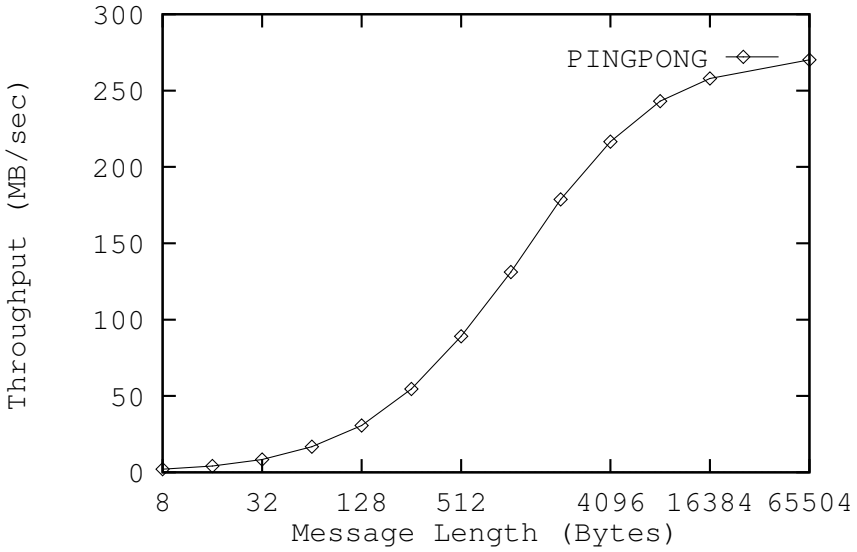
使用するノード数は 256（4 × 8 × 8）台である．時間測定には経過時間をクロックサイクル単位で測定できる関数を用いる．また，ピンポン転送以外の測定では，時間測定を開始する前に全ノードでソフトウェアによるバリア同期をとっている．

送信関数には，プリミティブな関数の中でも最も立ち上げオーバーヘッドの小さい関数を用いる．

18.2.2 一対一転送

まず始めに，基本的な一対一転送であるピンポン転送についての評価を行なう．ピンポン転送とは，2 つのノード間で交互にメッセージの送受信を行なう転送のことである．測定結果を図 34 に示す．

スループットは最大で約 270MB/sec とピーク性能の 9 割を達成できている．これは，システム領域でのデータのバッファリングを排除したことが大きな要因と考えられる．また，通信オーバーヘッドは約 3.5 μ sec，$N_{\frac{1}{2}}$（ネットワークのピーク性能の半分を達成しているメッセージ長）



,

図 34: ピンポン転送における実効スループット

は約 1KB と極めて小さい．このように低オーバーヘッドで済んでいるのは，RDMA 転送が軽いシステムコールで起動できることに因る．

以上より，CP-PACS に実装されている高速通信機構は効率良くハードウェアの性能を引き出していることが分かる．

18.2.3 1 対全 broadcast

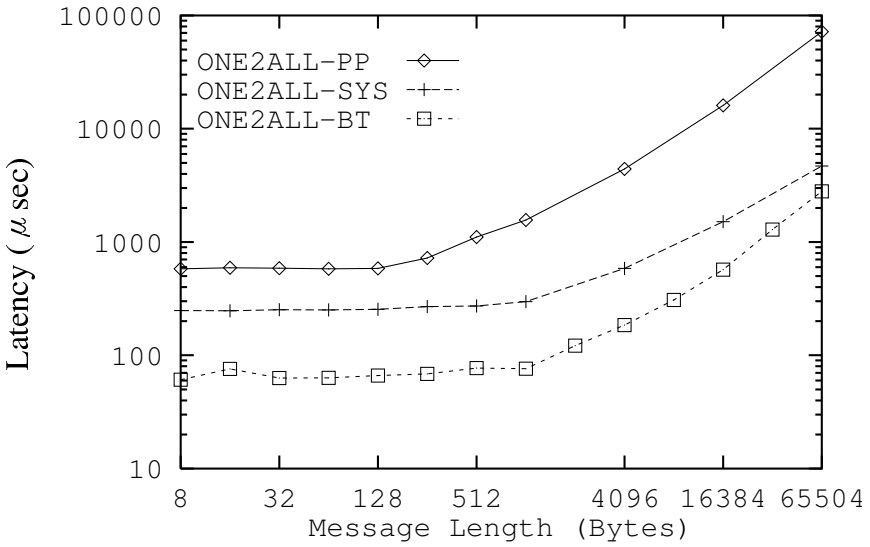
1 対全 broadcast について評価する．1 対全 broadcast とは，あるノードが持っているデータを全てのノードに持たせる転送のことである．

転送アルゴリズムとしては，システム関数を用いる方法（ONE2ALL-SYS），point-to-point 転送によって送信ノードがその他全ての受信ノードにメッセージを送信する方法（ONE2ALL-PP），2 進木状にメッセージを送信する方法（ONE2ALL-BT）の 3 種類について測定する．なお，ONE2ALL-BT では，各ノードが送信するメッセージ全てを TCW チェインにより一回の送信に統合している．

測定結果を図 35 に示す．ONE2ALL-BT がどのメッセージ長においても最短時間で転送を終了している．これは，転送回数が $log_2 P$（P はノード数）で済むことが最大の要因である．さらに，TCW チェインによる送信起動オーバーヘッド低減もかなり効いている．この TCW チェインの効果については，後述の行列の転置のところで考察する．

ONE2ALL-SYS は，同一次元方向に並んだノードに対して TCW チェインを用いて連続的に転送を行なうため，合計 3 回の転送で済む．しかし，OS による割り込み処理が介在するため ONE2ALL-BT よりも遅くなっている．但し，ONE2ALL-BT では転送のために全てのノードが通信に参加しなくてはならないのに対し，ONE2ALL-SYS はユーザの関与を最小限に抑えているので，プロセッサによる内部処理と並行して進めることができる．

また，ONE2ALL-BT は 2 進木に沿って転送を行なうため，ノード数が 2 の冪乗でなくてはならないという制約がある．したがって，ライブラリなどの形で 1 対全 broadcast を関数内に閉じ



,

図 35: 1 対全 broadcast におけるレーテンシ

込めてしまう場合には，システム関数つまり ONE2ALL-SYS を用いた方が良い．

以上より，1 対全 broadcast を行なう場合は，対象となるアプリケーションの性質を考慮した上で ONE2ALL-SYS，ONE2ALL-BT を使い分ける必要があると言える．

18.2.4 Scatter & Gather

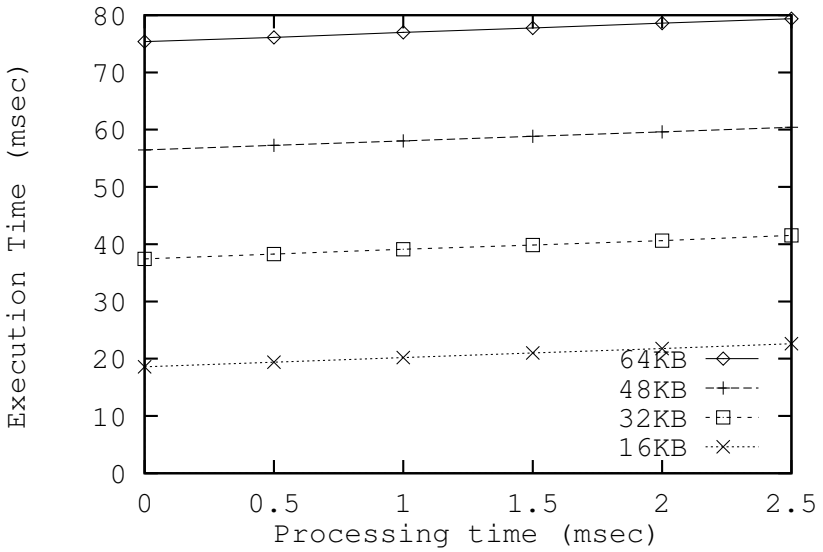
まず，ノード 0 が持っているデータを P 等分（P はノード数）して，それを他のノードへ分配する（Scatter）．各ノードは受けとったデータに対して何らかの処理を施し，その後ノード 0 にデータを送り返す（Gather）．このような処理のことを Scatter & Gather と呼ぶことにする．

測定では，プロセッサ内部処理時間を変えたときに全実行時間がどのようになるかを見てみる．さらに，各ノードに分配するデータ長を 16, 32, 48, 64KB と変えた場合について測定もする．なお，ノード 0 からその他のノードへの送信は，全て TCW チェインにより統合してある．測定結果を図 36 に示す．

スループットを算出すると約 430MB/sec となった．Scatter の際にデータを分配するノード 0 が，まだ全てのノードへのデータ分配を終えないうちに，その他のノードがプロセッサ内部での処理を終えてデータを送り返してくることがある．この場合，NIA は送受信双方向の同時処理に対応しており，またメモリはそれに耐えうるだけのバンド幅を備えているので，最大で 2 倍のスループットを実現できる．したがって，このような高いスループットとなった．

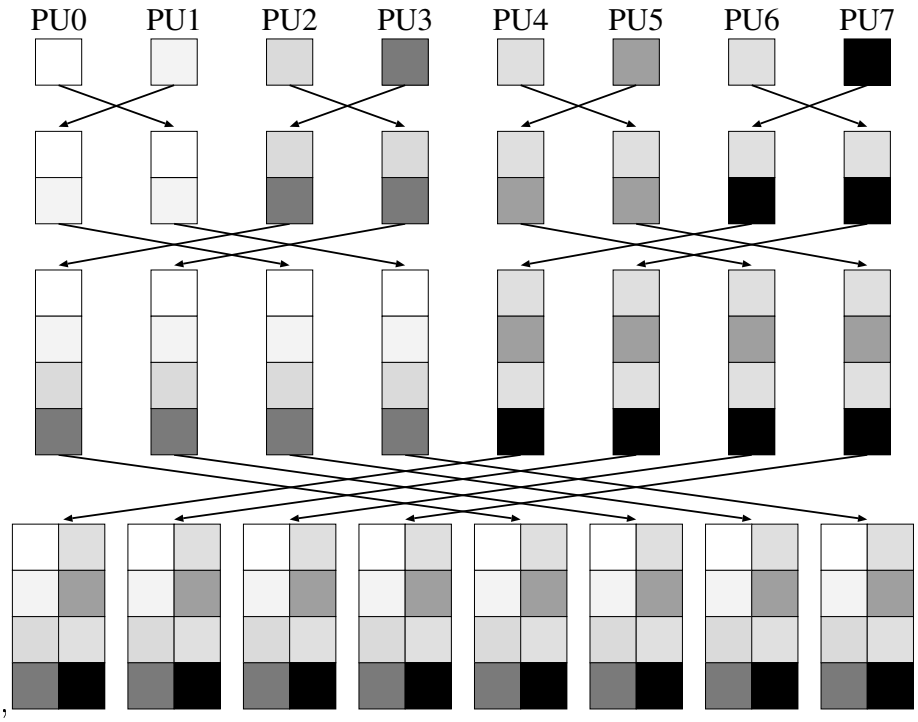
18.2.5 全対全 broadcast

バタフライ・コレクション・アルゴリズム（図 37）を用いて全対全 broadcast の評価を行なう．全対全 broadcast とは，各ノードが持っていたデータを全員が全員分持つようにする転送のことである．また，バタフライ・コレクション・アルゴリズムとは各ノードが隣のノードとのデータ交換からスタートして，送信距離，転送量ともに倍々と増やしていくアルゴリズムである．



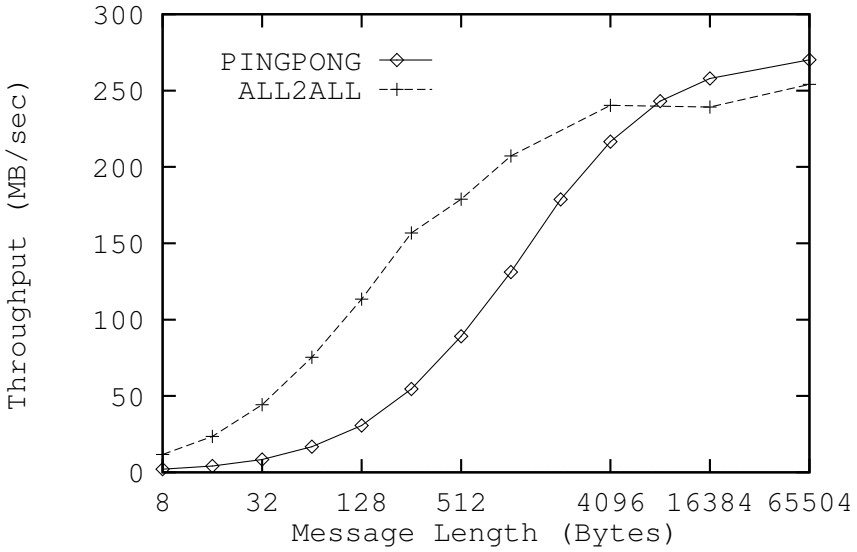
,

図 36: Scatter & Gather におけるレーテンシ



,

図 37: バタフライ・アルゴリズムによる全対全 broadcast



,

図 38: 全対全 broadcast における実効スループット

測定結果を図 38 に示す．全ノードがその他のノードに対して point-to-point 転送でメッセージを送信した場合と比べて，バタフライ・コレクション・アルゴリズムは，総メッセージ転送量は変らないものの，各ノードがメッセージを送信する回数が（$P-1$）回から $log_2 P$ 回（P はノード数，ここでは $P=8$）に減る，効率の良いアルゴリズムである．また，HXB では，バタフライ・コレクション・アルゴリズムによる全対全 broadcast を無衝突で行なうことができる．

図 38 では，メッセージ長が短いときは全対全 broadcast がピンポン転送の性能を越えているように見える．これは，送信一回当たりのメッセージ長がステップをふむ毎に倍々と大きくなるからである．

18.2.6 行列の転置

行列の転置を行なう転送について測定を行なう．行列の転置は多次元の高速フーリエ変換などに使用される．具体例を図 39 に示す．この場合，4×4 の 2 次元 PU 空間に 3 次元配列をマッピングしているので，各 PU は一つの次元方向（図中では N1）に対して連続したデータを持つ．しかし，転置を行なうことによって今度は N2 方向に連続したデータを持つことになる．したがって，転送前に持っていたデータは，分割して他の PU に送られる．この際，転送するデータは転送先によって異なり（個別転送），また分割する PU 数が多いほど メッセージ長は短くなる．

本測定では，3 次元 PU に 3 次元配列をマッピングしたときの転置について測定する．したがって，上例よりもさらに複雑な転送を要する．この際必要となるメッセージ転送について，メッセージ数（#MSG），メッセージ当たりのデータ量（data/MSG）を表 10 に示す．これより，各 PU に 512KB のデータを持たせた場合でもメッセージ長は 128B と $N_{\frac{1}{2}}$ を大幅に下回っているのが分かる．また，転送回数も多い．

このような転送に対し，各転送を順次転送する方法（basic) と，TCW チェインにより各ノードが送信するメッセージを全てまとめて一つのメッセージとする方法（chain）の 2 種類について測定を行なう．

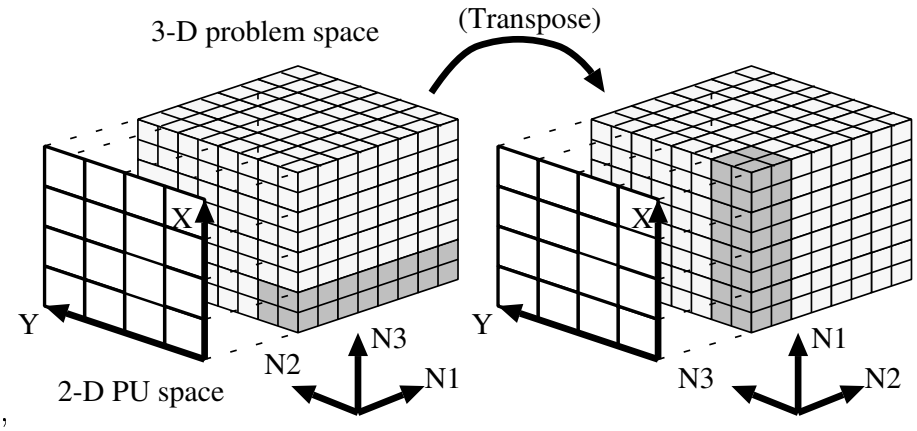


図 39: 2 次元 PU 空間にマッピングされた 3 次元配列の転置

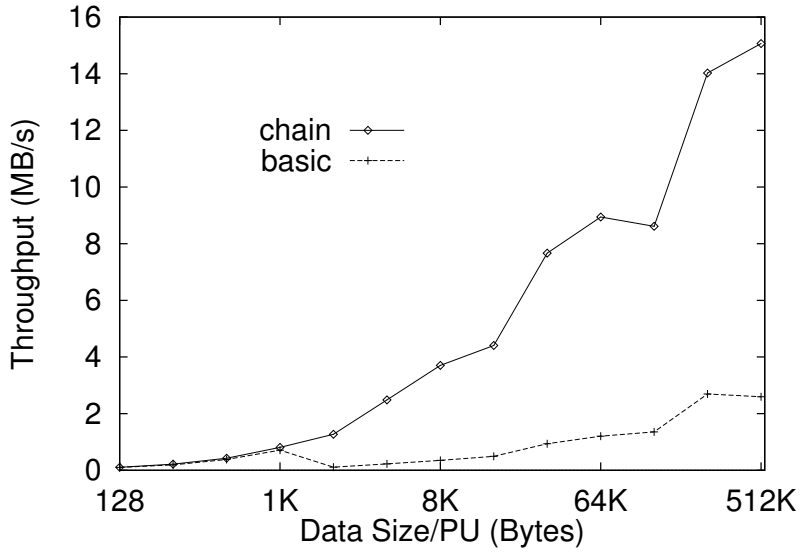


図 40: 転置転送における実効スループット

表 10: 行列の転置に伴う転送

data/PU（Bytes）	data/MSG（Bytes）	#MSG（個）
128	8	16
256	8	32
512	16	32
1024	16	64
2048	16	128
4096	32	128
8192	32	256
16384	32	512
32768	64	512
65536	64	1024
131072	64	2048
262144	128	2048
524288	128	4096

測定結果を図 40 に示す．basic において N 回のメッセージ転送を必要とした場合，chain では，TCW チェインにより 1 回の送信で済むため $(N-1)$ 回分のソフトウェアによる送信起動オーバーヘッドを省くことができる．測定結果を見ても，chain は basic に比べて非常に高い結果が得られている．

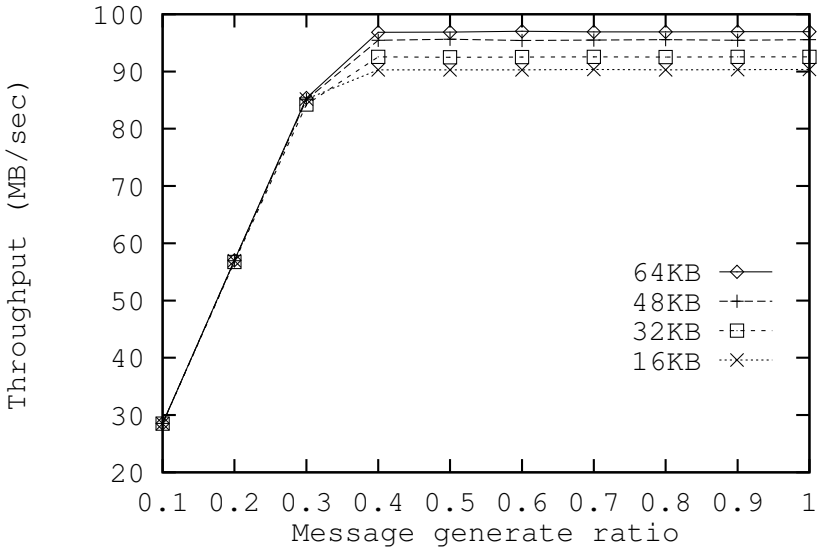
今回のように多数の短メッセージ転送を必要とする場合でも，CP-PACS では TCW チェインを用いることによりある程度性能低下を防ぐことができる．

18.2.7 ランダム転送

これまでは，転送相手が規則正しい転送についての評価を行なった．しかし，実アプリケーションによっては不規則な転送を行なうものもある．そこで，そのような不規則な転送の代表として，ランダム転送の評価を行なう．ランダム転送とは，送信先をランダムに決定する転送である．また，今回の測定ではメッセージ生成率を 0.1〜1.0 に変えて測定する．

メッセージ生成率を上げるにつれて，ネットワーク中のメッセージ数も多くなり，メッセージ同士の衝突が増える．ある一定以上のメッセージ生成率になるとスループットは飽和状態になるが，それはピーク性能の約 1/3 のところであった（図 41）．

これまで我々が行なってきたシミュレーションによるランダム転送の実験 [8, 12] でも，3 次元 HXB の場合はピーク性能の約 1/3 と非常に近い値が出ている．またこのシミュレーションでは，HXB が他のメッシュ／トーラスといった他のネットワーク・トポロジと比べて非常に高いという結果が示されている．



,

図 41: ランダム転送における実効スループット

18.2.8 実アプリケーションへの応用

以上の測定結果より，HXB は様々な転送パターンを効率的にこなせることが分かった．そこで次に，実アプリケーションに対してどれだけ HXB が有効かを考察する．

実アプリケーションでは，必ずしも一種類の転送パターンしか用いないわけではない．例えば NAS 並列ベンチマーク [15] に出てくる CG（Conjugate Gradient）法を例にとると，shuffle 転送，butterfly collection，butterfly summation などといった複数の転送が必要となる [16]．

メッシュやトーラスといったその他のネットワーク・トポロジでも，それぞれの転送パターンに適した各ノードへのデータ割り付けを行なうことにより無衝突もしくはそれなりに効率良くデータ転送を行なうことができるかもしれない．しかし CG 法のようにプログラム中に複数の転送パターンがある場合，どれか一つの転送パターンを効率良く行なえるデータ割り付けであったとしても，その他の転送パターンではうまくいかないことがある．

それに対し，HXB では不規則な転送を含めた各種転送パターンに強いので，どのようなデータ割り付けであってもそれぞれの転送を効率良くこなせる．したがって，対象とするプログラムに最も適したノードへのデータ割り付けを行なった後に，適当な転送パターンを用いてノード間のデータの交換を行なうといったこともできる．

よって，各種転送パターンだけではなく，実アプリケーションにおいてこそ更に HXB の有効性が示されるものと推測される．

第 III 部

CP-PACS による計算物理学

19 CP-PACS 設計仕様の策定に際しての計算物理学からの要求性能の検討

本プロジェクト当初，CP-PACS の設計仕様を定めるにあたっては，素粒子物理学，宇宙物理学，物性物理学の典型的な問題に対して，開発する並列計算機の満たすべき性能を検討した．ここでは，最も詳しく調べた素粒子物理学の格子量子色力学の場合を詳述する．宇宙物理学の計算能力に対する要求は，素粒子物理学と同程度と判断された．ただし，要求する計算能力とメモリ容量との割合が異なるので，その点を主として記述する．物性物理学は，多様なモデルを数多くのパラメータに対して計算を計算を行なうことが多く，計算能力, メモリ容量，ディスク容量に対する要求は素粒子物理学，宇宙物理学と同程度またはそれ以下である．

19.1 素粒子物理学における格子量子色力学からの要求性能

本プロジェクトの主目的の一つは，格子量子色力学，即ち素粒子の強い相互作用の基礎理論を，専用並列計算機を駆使した数値シミュレーションによって解き，その正当性を検証すると同時に，新たな定量的予言を引き出すことにある．本章に於いては，この目的の為必要とされる並列計算機の計算能力に関する検討結果を述べる．

19.1.1 格子量子色力学の計算内容及び計算方法の概要

格子量子色力学 (lattice QCD) は，ハドロンの基本構成粒子であるクォーク（従来から 5 種類が知られていたが，最近 6 種類目の証拠が見い出された）と，それらを結び付けるグルオンを基本的自由度とする四次元時空格子上の場の理論である．これらの自由度を表す基本変数は，四次元時空格子の各格子点 n に定義された 3x4 次元複素グラスマンベクトル $q = q_n^{a\alpha}(a = 1, 2, 3, \beta = 1, 2, 3, 4)$ 及び格子上の最小線分 ℓ 各々に定義された 3 行 3 列の特殊ユニタリ行列 $U = U_\ell^{ab}(a, b = 1, 2, 3)$ により与えられる．前者がクォークの自由度を，後者がグルオンの自由度を表す．変数 q の 3 次元分は変数 U の 3 次元に対応し，カラーの自由度と呼ばれる．残り 4 次元はスピン自由度である．

理論を定める基本的量は，作用と呼ばれる U と q の関数である．格子量子色力学の作用は次式で与えられる．

$$S_{QCD} = \frac{\beta}{6} \sum_p \mathrm{tr}\,(U_{\ell_1} U_{\ell_2} U_{\ell_3} U_{\ell_4}) + \sum_{n,n'} \bar{q}_{n'} D_{n,n'}(U) q_n \tag{1}$$

但し，第一項は，格子上の最小正方形 p 全てについてその四辺上のグルオン変数 U の積の行列トレースを取ることを意味する．β は格子量子色力学の結合定数 g^2 と $\beta = 6/g^2$ なる関係にあるパラメタである．第二項はクォーク作用であり，その取り方には，Wilson fermion 法及び Kogut-Susskind (KS) fermion 法と呼ばれる二つの代表的方法がある．Wilson fermion 法の場合，$D = D_{n,n'}^{a\alpha,b\beta}(U)$ は変数 U について線形の $12V \times 12V$ 次元（V は格子点数）の行列である．本稿では， 特に断らない限り，Wilson fermion 法を中心に評価を行なう．クォーク質量 m_q は行列 D にパラメタとして現れる．

量子色力学の物理量は，作用より定まる重み付きの U と q に関する次の積分平均により与えられる．

$$< \mathcal{O}(U,q) >= \frac{1}{Z} \int \prod_{ell} dU_\ell \prod_n dq_n d\bar{q}_n \mathcal{O}(U,q) \exp S_{QCD}(U,q) \tag{2}$$

但し，Z は $<1>=1$ とするための規格化定数である．

格子量子色力学の数値シミュレーションでは，結合定数とクォーク質量の値を選択した後，適当な確率過程により，積分の配位空間の点（配位）を重み $\exp S_{QCD}$ に従って生成し，各配位上での物理量の値を平均することにより物理結果を得る．

シミュレーションは，配位の生成法により二種類に大別される．「クェンチ近似」 (quenched QCD) は，作用の中の変数 q を無視する近似である．生成法は，変数 U に対する，メトロポリス法・熱浴法等のモンテカルロ法であり，基本演算は，格子の最小正方形各々について，その回りの三辺に沿っての変数 U の行列積 $U_1 \cdot U_2 \cdot U_3$ の計算，その値に基づく残り一辺上の U の変更値の確率的棄却・採用である．

「完全な量子色力学」 (full QCD) ではこのような近似は行なわない．生成法は，ハイブリッドモンテカルロ法を基本とする．配位分布に対する変数 q の効果を取り入れるには，正規乱数を元とする $12V$ 次元ベクトル Y 及び与えられた U の配位に対して，行列 D についての $12V$ 次元連立一次方程式 $Dx = Y$ を解くことが必要となり，これが計算時間の大半を占める．解法としては，共役傾斜法・最小残差法・共役残差法等が用いられ，不完全 LU 分解等の前処理も有効である．基本演算は，$12V$ 次元複素ベクトル p と行列 D の積 $p \to Dp$ である．

生成された配位上での物理量の計算は様々であるが，最も計算時間を要するのは生成された U の各配位上でクォーク伝播関数 $G^{a\alpha,b\beta}_{n,n'} = (D(U)^{-1})^{a\alpha,b\beta}_{n,n'}$ を求めることである．これは連立一次方程式 $Dx = b$ を，12 種類の右辺 b に対して解くことにより得られる．ハドロンの質量や遷移行列要素等，強い相互作用諸量は，このクォーク伝播関数を組み合わせて得られるハドロン伝播関数に，χ^2 フィット等の処理を行なって抽出される．

19.1.2　格子パラメタの選択

格子量子色力学の基本的対象は複数のクォークからなるハドロンである．ハドロンの空間的拡がりは約 $r_H = 2\text{fm}$ 程度（$1\text{fm}=10^{-15}\text{m}$），またクォークの質量は最も軽い u, d クォークの $5 - 10\text{MeV}$ から現在知られている中では最も重い t クォークの 175 GeV 迄広い巾がある．このような系の物理的性質を離散且つ有限な時空格子上での数値シミュレーションにより求めるには，統計誤差に加え，格子を用いたことに起因する各種の系統誤差を小さくしなければならない．このための基本的条件は，(1) 格子間隔 a（最近接格子点間の距離）が r_H に比べて充分小さく，且つ (2) 格子のサイズ La（L は一辺当たりの格子点数）が r_H に比べ充分大きいこと，さらに (3) クォーク質量を現実の値に近く取りうること，の三点である．

条件 (1) と (2) を満たすには一辺あたりの格子点数 L を大きく取ればよいが，格子量子色力学は 4 次元時空上の理論であり，その数値シミュレーションは，必要な記憶容量，基本的演算数，出力データ容量のいずれも L^4 に比例して増大するため，異論の余地のないほど大きな L（例えば $L = 1000$）を取ることは現実的ではない．また，条件 (3) に関しては，軽いクォークに対して前節に述べた連立方程式 $Dx = b$ を解く際の逐次近似解法の必要反復回数が増え，且つ統計的揺らぎが増大するとの困難がある．これは，クォーク質量 m_q が行列 D の定数項として現れ，D の条件数が $1/m_q$ に比例するためである．特に，full QCD シミュレーションに於いては，配位更新の度に連立方程式を解く必要があり，物理量の計算に於いてのみこのことが必要な quenched QCD

	現状			目標		
	La/a(fm)	m_π/m_ρ	格子サイズ	La/a(fm)	m_π/m_ρ	格子サイズ
ハドロン質量	2-3/0.2-0.1	≥ 0.5	$\leq 32^3 \times 64$	3-4/0.1-0.05	$0.4-0.3$	$\geq 64^3 \times 64$
重いクォークの物理	2-3/0.1	≥ 0.9	$\leq 24^3 \times 48$	2-3/0.05-0.03	≥ 0.9	$\geq 64^3 \times 64$
CP 非保存等	2-3/0.2-0.1	≥ 0.5	$\leq 24^3 \times 48$	5-6/0.2-0.1	$0.5-0.4$	$\geq 64^3 \times 64$
有限温度相転移	2-3/0.3-0.2	≥ 0.3	$\leq 16^3 \times 8$	5-6/0.2-0.1	$0.3-0.2$	$\geq 64^3 \times 16$

表 11: 格子量子色力学シミュレーションの現状と目標

に比して，著しく計算量が膨大になる．以上の理由のため，格子間隔 a 及び格子サイズ L をどのように取れば有限格子間隔と有限格子サイズに起因する系統誤差が充分に小さい物理結果を得られるか，またその際クォーク質量をどこまで下げうるかを知ることは極めて重要である．

これらの条件の詳細は問題とする個々の強い相互作用物理量に応じそれぞれ多少異なっている．最も基本的なハドロン質量の計算を例に取れば，現在迄に，$a \approx 0.2-0.1$fm，$La = 2-3$fm（即ち $L \approx 20-30$），またクォーク質量については π 中間子と ρ 中間子の質量で見て $m_\pi/m_\rho \geq 0.5$（現実の値は $m_\pi/m_\rho = 0.18$ であり，これはより軽いクォークに対応する）の領域が，quenched QCD により詳しく調べられている．その結果を基礎として，格子パラメタについては $a \approx 0.1-0.05$fm 及び $La \approx 3-4$fm（即ち $L \geq 60$ ），クォーク質量については $m_\pi/m_\rho \approx 0.3-0.4$ への計算拡張を行なえば系統誤差数%以下の精度が得られると考えられる．

表 11 に，格子量子色力学に於ける中心的課題それぞれについて，現在行なわれているシミュレーションのパラメタ値及び信頼しうる結果を得るために目標とすべきと思われるパラメタ値を示す．

19.1.3 必要とされる計算能力の検討

格子量子色力学の数値シミュレーションは表 11 に挙げたごとく多岐にわたるが，その中で最も基本的課題は，π 中間子，ρ 中間子，核子等軽いクォークからなるハドロンの質量スペクトルの計算及びその実験値との比較による量子色力学の検証である．この計算は必要とされる計算能力の面からも最も大規模にわたるものの一つであり，従って，その検討は，格子量子色力学シミュレーション全般にとり基本的である．以下，quenched QCD と full QCD それぞれについて，計算の各段階について考慮すべき点をまとめた後，(1) 計算速度，(2) 主記憶容量，(3) 中間結果保持のための一時記憶容量，(4) 最終結果保持のための外部記憶容量，についての具体的評価を示す．

quenched QCD の計算手続きは，(1) 変数 U に対するモンテカルロ法による配位の生成，(2) 生成された配位上で，$Dx = b$ を解くことによるクォーク伝播関数の計算，(3) クォーク伝播関数を組み合わせたハドロン伝播関数の計算，(4) 以上の繰り返し，からなる．各ステップでの計算時間を左右するパラメタは次のとおりである．

1. 配位間 sweep 数

   変数 U の生成には通常各最小線分上の U_ℓ に逐次確率的変更を加える．各 U_ℓ に於いて必要な浮動小数点演算数は 5000 程度である．全ての U_ℓ の変更を行なうことを 1 sweep とよ

m_π/m_ρ	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.18
m_q(MeV)	713	563	107	61	37	21	11	3.8
繰り返し回数	200	200	360	630	1060	1830	3470	10130

表 12: red/black 最小残差法の繰り返し必要回数（各要素の精度 0.01 %）($m_\pi/m_\rho = 0.18$ は実験値に対応）.

ぶ．統計的に独立な配位を得る為に必要な sweep 数 N_{sweep} は，モンテカルロ法の詳細にも依存するが，少なくとも 3000 sweep 以上を実行する必要がある．

2. 配位数

   統計誤差を充分に制御するために必要な独立な配位数 N_{conf} については充分に知られていない．現在迄で最大の $24^4 - 32^4$ 格子・配位数 200 の計算結果によれば，ハドロン質量の統計精度を 1 % におさえるのに必要な配位数は，24^4 格子で 10000 程度以上，64^4 格子で 500 程度以上と考えられる．

3. クォーク伝播関数の計算法

   大行列連立一次方程式 $Dx = b$ の逐次近似解法には様々なものがある．繰り返し一回あたりの基本演算は既に述べたごとく $p \to Dp$ であり，方法によりこれを行なう回数が異なる．効率は，繰り返し一回当たりの計算量と繰り返し回数の積により決まる．ここでは格子量子色力学に適し且つ並列計算機上での実現が容易なものとして red/black 最小残差法を例にとる．この場合，$p \to Dp$ は 2 回必要であり，繰り返し 1 回あたりの浮動小数点演算総数は格子点数を $V = L_x \times L_y \times L_z \times L_t$ として，$1728V$ となる．

   逐次近似解法の必要繰り返し回数 N_{iter} はクォーク質量等格子パラメタに依存する．現在までの経験によれば，繰り返し回数は物理的クォーク質量の逆数に比例し，格子サイズ等他のパラメタにはあまり依存しない．物理的クォーク質量は π 中間子と ρ 中間子の質量比 m_π/m_ρ と直接に関係する．表 12 に最小残差法を用いた場合，解 x の全ての要素が 0.01 % 精度で求まるのに要する繰り返し回数経験値を示す．

4. PU 実効速度と PU 間通信性能

   並列計算機に於ける計算時間は，CPU の実効速度と CPU 相互間の通信速度により決まる．ここでは，実効速度はパラメタとし，通信時間は演算時間の 20 % と仮定する．

5. 主記憶容量

   主記憶容量は，基本変数に計算アルゴリズム上必要な補助変数及び並列計算効率化のためのいわゆる「袖」等を加えたもので決まる．後者は並列計算機の PU 台数と格子サイズの比率等に依存するためここでは考察から除外する．quenched QCD の配位生成に於ける基本変数は U，その次元（実数換算）は $72V$，プログラム上必要な変数総数はその約 2 倍である．クォーク伝播関数の計算においては，基本変数は U 及び $Dx = b$ の解 x（複素 $12V$ 次元），変数総数は逐次近似解法により多少異なり，最小残差法の場合，変数 U の 2 倍である．

6. 一時記憶容量

   ハドロン伝播関数はクォーク伝播関数を組み合わせて得られるため計算時間は大きくないが，クォーク伝播関数は必要な分だけ並列計算機に結合した分散ディスク等一時記憶に保管しておく必要がある．クォーク伝播関数は実数換算で $288V$ 次元の大配列であり，各配位上で計算後一時保管する必要のあるクォーク伝播関数の数 N_q は物理の要請から 5–7 程度に上る．このため，一時記憶容量，一時記憶への書きこみ・読みだし時間は，無視できない要素になる．クォーク伝播関数を計算後，その一時記憶への書き込みは N_q 回，またハドロン伝播関数計算に際しての読みだし回数も N_q である．一時記憶への I/O 時間を規定する I/O スループットについては 300MByte/sec を仮定する．

7. 外部記憶容量

   格子量子色力学計算の基本となるのは変数 U の配位（ゲージ配位）である．ひとたびこれが生成されれば，その上で，表 11 に示した多様な計算を行なうことができる．この意味で，外部記憶装置に保管する必要があるのは，配位生成により得られた N_{conf} 個の配位及びハドロン伝播関数その他の最終データである．クォーク伝播関数は一配位あたりで既にその容量が膨大になるため外部記憶装置への記録は現実的でない．

full QCD に於いては，配位の生成法が，ハイブリッドモンテカルロ法等に変更される．これに付随する変更点は以下のとおりである．

1. 配位生成

   ハイブリッドモンテカルロ法では，変数 U に共役な運動量 X を導入し，作用をポテンシャルとする分子動力学により二つの変数 U と q をある時間 τ だけ時間発展させた後，変数 U の確率的棄却・採用を行なう．これが quenched QCD の 1 sweep に相当し，通常 1 trajectory と呼ばれる．この方法は，quenched QCD の場合のモンテカルロ法と異なり，全ての U_ℓ を同時に更新する．更新の大きさは分子動力学時間発展のステップサイズ $\delta\tau$ により決まり，この各ステップで Y を正規乱数ベクトルとして $Dx = Y$ を解くことが必要になる．

   更新された配位の採用率を 80% 程度以上に保つためには，経験的にステップサイズを $\delta\tau \approx 0.02(8/L)$ 程度に取る必要がある．従って，trajectory 長を $\tau = 1$ とした場合，1 trajectory の生成には，$50(L/8)$ 程度のステップが必要である．$Dx = Y$ の解は基本的にクォーク伝播関数の計算と同じであり，これが「完全な量子色力学」シミュレーションが多大の計算時間を要する原因である．逐次近似解の収束に必要な繰り返し回数はクォーク質量に依存し，最小残差法を用いた場合には表 12 に与えられる．

2. 配位間隔及び配位数

   ハイブリッドモンテカルロ法による配位生成に於いて，trajectory 数を幾つに取れば独立な配位が得られるかについては信頼すべき資料がない．配位生成のための計算時間が膨大であるので，できうる限り多数の配位上で物理量の計算を行なっているのが現状であるが，少なくとも $\tau \approx 5$ 程度（即ち，trajectory 長が 1 ならば 5 trajectory）は間隔を置くべきであると考えられる．

   full QCD の大規模計算としては，現在迄に，$16^4 - 20^4$ 格子上で総 trajectory 数 1000–2000 が行なわれている．その経験を基にすれば，統計誤差を有意味に評価するために必要な総 trajectory 数は，24^4 格子で 10000 程度以上（即ち，5 trajectory 間隔にて，配位数 2000 以上），32^4 格子で 3000 程度以上が一つの目安と考えられる．

3. 主記憶容量

   基本変数は U と共役運動量 X 及び Y，補助変数には $Dx = Y$ を解くためのもの等が含まれる．プログラム上必要な変数総数は U の 5 倍程度である．

格子量子色力学のハドロン質量計算に於いて， 1 配位を処理するために必要とされる浮動小数点演算数及び記憶容量を表 13 にまとめる．但し，ハドロン伝播関数計算部分は，浮動小数点演算数は計算内容に応じ変動が大きいこと，記憶容量はクォーク伝播関数計算の為に必要な容量にて充分であることから省略した．

以上の考察を基に，実効速度 400 GFLOPS を仮定した場合，ハドロン質量計算に要する計算時間及び記憶容量の評価を，幾つかの格子サイズに対して，表 14(a) に示す．前提条件は同表 (b) にまとめてある．また，実効速度を変数とした場合の計算時間についての評価を図 42（quenched QCD）と図 43（full QCD）に与える．

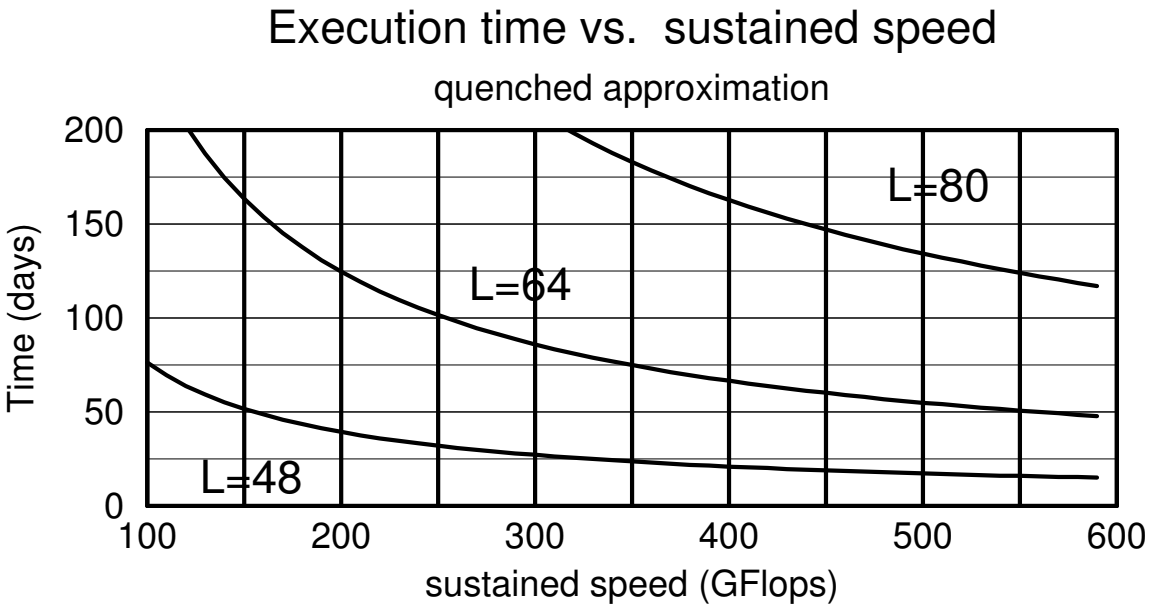


図 42: 格子サイズ L^4 の quenched QCD ハドロン質量計算に必要な CPU 日数の実効速度依存性．前提条件は表 14(b) に同じ．

計算ステップ	配位生成		クォーク伝播関数
	quenched QCD	full QCD	
演算数	$2 \cdot 10^4 V N_{sweep}$	$1728 V N_{iter} \tau / \delta\tau$	$1728 V N_{iter}$
主記憶	$144V$	$368V$	$144V$
一時記憶	–	–	$288 V N_q$
外部記憶	$72 V N_{conf}$	$72 V N_{conf}$	–

表 13: ハドロン質量計算に於ける浮動小数点演算数及び記憶容量（ワード単位）

(a) ハドロン質量計算の計算時間及び必要記憶容量

	quenched QCD			full QCD		
	48^4	64^4	80^4	24^4	32^4	48^4
計算時間（日）	21.1	66.7	163	58.0	244	1854
主記憶（GB）	5.7	18	44	0.91	2.88	14.6
一時記憶（GB）	68.3	216	527	2.85	9.0	45.6
外部記憶（GB）	1420	4500	10990	89	281	1424

(b) 前提条件

全体に関する条件	quenched QCD に関する条件	full QCD に関する条件
演算実効速度=400GFLOPS	$m_\pi/m_\rho = 0.9, 0.8, 0.7, 0.6, 0.5, 0.4^\dagger$	$m_\pi/m_\rho = 0.7, 0.6, 0.5, 0.4^\dagger$
通信時間／演算時間=20%	配位数 500	配位数 500
一時記憶 I/O スループット	配位間 sweep 数 3000	配位間隔 $\tau = 5$
=300MByte/sec		$\delta\tau = 0.02(8/L)$

$^\dagger Dx = b, Y$ の解法は前処理つき最小残差法，収束回数は表 12 と同じ．

表 14: 格子量子色力学に於けるハドロン質量計算の計算時間及び必要記憶容量

19.1.4 乱数及び初等関数

格子量子色力学のシミュレーションに質の良い乱数の効率的生成は必須の要件である．表 15 に熱浴法を用いた quenched QCD での配位生成及びハイブリッドモンテカルロ法による full QCD での配位生成に於いて必要な乱数の種類と個数，さらにその使用に付随して必要となる初等関数の種類を挙げておく．

19.1.5 結論

19.1.1 節「格子量子色力学の計算内容及び計算方法の概要」の表 11 に挙げたように，格子サイズ 64^4，独立配位数 500 の計算は，量子色力学数値シミュレーションに於いて信頼し得る結果を得るための一つの大きな目標である．

quenched QCD に於いては，前節表 14 及び図 42 によれば，実効演算速度 400 GFLOPS を達成できれば，この計算は 60 CPU 日程度で実行することが可能である．格子間隔についての系統誤差の評価の為には，格子の物理的サイズ La を一定に保って，一辺当たりの格子点数 L を 3 種類以上変えた計算が必要である．これについては，最大格子サイズを 64^4 に取り，より小さな格子サイズを用いてより大きな格子間隔の計算を行なうことにより，3 種類の格子間隔に対して総計 180 CPU 日程度で実行することができる．さらに，部分的に 64^4 より大きな格子サイズあるいは配位数 500 より高統計が必要な場合，クォーク質量等のパラメタを限定してその点に集中し

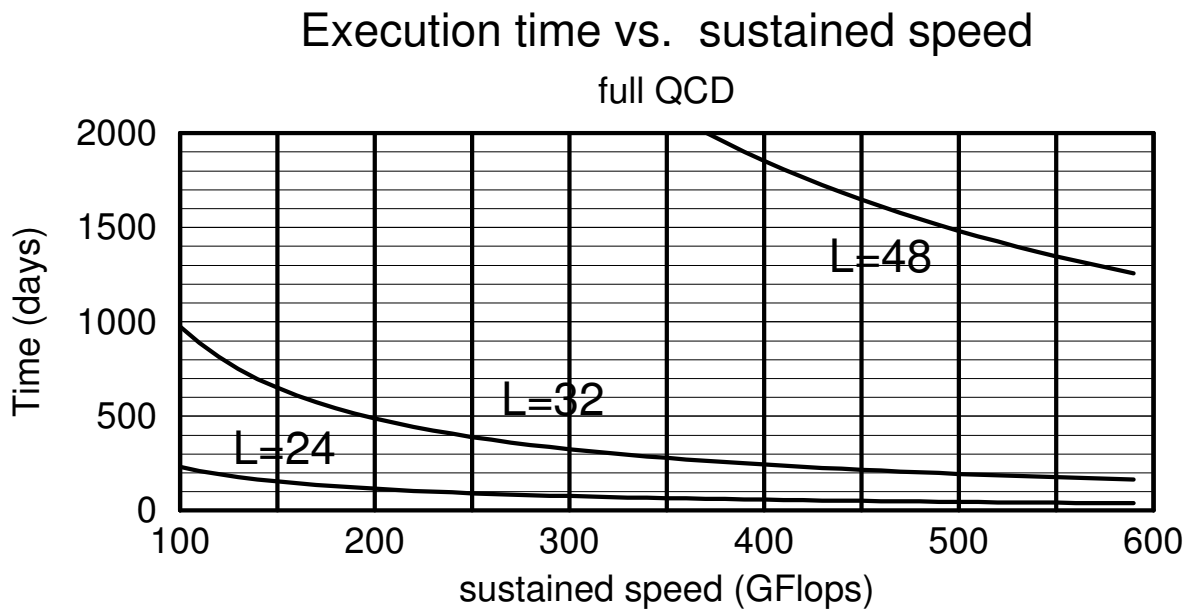


図 43: 格子サイズ L^4 の full QCD ハドロン質量計算に必要な CPU 日数の実効速度依存性．前提条件は表 14(b) に同じ．

	種類	個数	初等関数
熱浴法	$[0,1]$ 一様乱数	$(24N_{hit}+48)V^{\dagger}$/sweep	$\exp, \sqrt{}, \cos, \sin, \log$
ハイブリッド法	正規乱数	$56V$/trajectory	$\tan, \sqrt{}, \cos, \sin$

† N_{hit} は $SU(2)$ 部分行列あたりのヒット回数．

表 15: 量子色力学数値シミュレーションに必要とされる乱数の種類と個数及び使用する初等関数

た計算を行なうことにより，充分実行可能と思われる．必要主記憶容量については，プログラム上必要な「袖」の部分，また上記の部分的により大きな格子サイズの計算の可能性，等を考慮すると，80 GByte 以上が必要と考えられる．また，一時記憶及び外部記憶については，多岐にわたる計算結果を保持する必要性等を考慮すると，一時記憶 1000 GByte 以上，外部記憶 50 TByte 以上が必要条件と考えられる．

full QCD については，red/black 最小残差法により $Dx = b$ を解く限り，実効速度 400 GFLOPS の場合の現実的計算は 32^4 格子程度に留まるが，この場合でも，現在の quenched QCD 計算を凌駕する結果が期待される．また，不完全 LU 分解共役傾斜法など，繰り返し回数の大幅縮減が可能な解法の効率的な並列化が実現できれば，48^4 格子の計算も現実味を帯びる．従って，計算アルゴリズムの開発も並列計算機の有効活用の面から力を注ぐべきテーマである．必要記憶容量に関しては，表 14 からもわかるように，quenched QCD 計算の場合から評価した値が実現されれば充分である．

以上をまとめて，

実効演算速度 400 GFLOPS 以上，
主記憶 80 GByte 以上，
一時記憶 1000 GByte 以上，
外部記憶 50 TByte 以上

を実現することができれば，quenched QCD の枠内では多くの問題について最終的解決に充分の見込があり，full QCD に於いても，現在の quenched QCD 計算以上の精度の達成が可能であると考えられる．

19.2 宇宙物理学からの要求性能

計算宇宙物理学の眼目は，宇宙における様々な非線形現象，例えば銀河形成や星・惑星系の誕生，宇宙大規模構造の発生といった複雑な過程を，数値シミュレーションによって解析し，その本質的物理機構を解明するところにある．この目的のためにこれまで用いられてきた手法は，主として宇宙流体力学である．宇宙流体力学は，様々な数値流体計算法と共に発展してきたが，最近までの計算は，空間的に 1 次元，2 次元が主流であった．これは，計算機の能力が足りなかったためである．しかし，昨今のスーパーコンピュータの発達により，現実的な 3 次元の計算を行なうことも可能になってきている．さらには，ブラックホール周りのガス運動や，銀河中心核の物理を解明するために，流体力学と電磁場を結合させた電磁流体力学を数値的に解く試みもなされ成果を上げ始めている．しかし，これらの 3 次元計算は，空間分解能の点で，まだまだ不十分なものであり，1 桁近く線形分解能を上げることが要求されている．

一方，このような計算機の発達にあっても，いまだに扱えない分野がある．それは，輻射流体力学である．輻射は宇宙に普遍的に存在するものであり，あらゆる宇宙現象に密接に関係している．そもそも，天体の観測は，そこで起こっている物理過程の結果として生まれた輻射を見るものであり，この情報を通してその物理機構を解明していく．すべての天体現象で，輻射場と物質場の相互作用が重要な役割を果たしていることは間違いないが，この相互作用を完全な形で扱うことはこれまで不可能とされてきた．それは，この問題が空間 3 次元，輻射場 3 次元の 6 次元問題になるからである．しかし，超並列計算機を用いれば，その主記憶容量，処理速度の点から言って，この 6 次元問題を解くことが可能になる．(具体的な評価は後に述べる．) また，この問題は輻射場の 3 次元については完全に並列処理を行なうことが可能であり，並列計算に極めて適している．

ここでは，宇宙流体力学，電磁流体力学，輻射流体力学の具体的数値計算法について，現状で行なわれている計算の分解能と，超並列計算機によって目標とする分解能，及びその場合の必要メモリ容量，ディスク容量，標準的な 1 つの計算を行なうのに必要な CPU 時間をまとめ，必要性能を評価する．

19.2.1 宇宙流体力学

基礎となる宇宙流体力学を解く方法は大きく分けて 2 つある．それは，オイラー法とラグランジ法である．オイラー法は，空間に固定グリッドを張りめぐらし，そのグリッドでの物理量の変化を追いかけていく方法で，通常，格子法と呼ばれる．これに対し，ラグランジ法は，流体の運動と共に物理量の変化を見る方法である．空間 3 次元の計算でこれを実現する方法に，流体粒子法がある．この 2 つの方法は，どのような物理変化を見ていきたいかによって使い分けされるべきものである．

宇宙流体力学の特徴は，多くの場合，自己重力系の流体力学であることである．また，しばしばダークマターなどの無衝突粒子系も含んだ 2 成分系の計算になることもある．宇宙流体力学で扱う系は， 2 次元以上の構造を持つことがほとんどである．この場合，自己重力のポアッソン方程式をどう解くかによって計算コストは異なってくる．全ての要素の重力相互作用を直接的に計算するのが最も精度がよいが，これは要素の数の 2 乗の計算コストがかかり非常に重いため，通常は重力計算時間の短縮のための様々な便法が用いられる．

格子法　格子法で用いられる重力計算法の一つは，格子に対して FFT を用いる PM (Particle-Mesh) 法で，計算量は $O(N_G)$ である．ここで，N_G は一辺当たりの格子点数を L として $N_G = L^3$ である．ただし，この場合の重力計算精度は，格子間隔の数倍になることが知られている．そこで，精度を上げる為に P^3M (Particle-Particle Particle-Mesh) 法を用いることがある．P^3M 法は，長距離は FFT で計算し，短距離は直接法で計算するもので，計算コストは $N_G \log N_G$ に比例する．

流体部分の方程式系は双極型の偏微分方程式である．その上宇宙物理では，衝撃波が現れる問題を解くことが多い．このような系を数値的に解く方法として最近は，衝撃波を含んだ解の非線形安定性の条件 (TVD 条件と呼ばれる）を満たす TVD 法が主流である．TVD 法には空間 2 次精度の Roe 法や TVD-ES 法，3 次精度の PPM 法などがある．これまでの計算では，線形空間分解能 ($\equiv 1/L$) で 0.01 程度の計算が限界であったが，この分解能を 1 桁上げることが目標である．この場合，時間発展方程式を安定に解く為の条件 (CFL 条件) を満たなければならないため，時間間隔を分解能に比例してとる必要がある．従って，空間分解能を 1 桁上げることは，1 job のステップ数が 10 倍になり，CPU コストは格子の増加分より 1 桁大きくなる．表 16 に P^3M-TVD 法の現状と目標を，必要メモリ容量，ディスク容量，標準的な 1 つの計算を行なうのに必要な CPU 時間，データ転送時間で示す．この表を見てわかるように，現在の空間分解能を 1 桁上げる為には 100 GB 近い大容量メモリが必要である．また，この計算をいくつかのパラメータについて行なう為には，500 GFLOPS 以上の CPU 性能が必要である．

流体粒子法（SPH）　流体粒子法は流体のラグランジ計算をするため，格子法の場合のような FFT を用いることができない．そこで，粒子法の重力計算には Tree 法と呼ばれる高速計算法が用いられる．Tree 法は，空間をおのおの一つの粒子が占有するテリトリーに分割し，近くは高分解能で計算し，遠くは低分解能で計算する方法で，ダークマターや流体粒子のような粒子系の自己重力計算に適している．Tree 法の演算量は，流体粒子の数を N_S として $O(N_S \log N_S)$ であ

	分解能 $(1/L)$	メモリ (GB)	ディスク容量 (GB)	CPU 時間 (/job)	データ転送時間 (/job/(10MB/s))
現状	0.01	0.08	0.4	6 hr/Gflops	3.7 min
目標	0.001	80	400	120 hr/500Gflops	12.4 hr

表 16: 宇宙流体力学シミュレーション（格子法）の現状と目標

る．分解能を格子法と比較すると，$N_S = O(10) {N_G}^{\gamma}$，$1/2 \leq \gamma \leq 2/3$ の関係がある．従って，$N_G = 10^6$ $(L = 100)$ の格子法と $N_S =$ 数 10^4 の粒子法がほぼ同じ空間分解能の計算ということになる．ここで，流体粒子法の分解能とは線形分解能 ($\equiv 1/\sqrt[3]{N_S}$) である．Tree-SPH 法の現状と目標を表 17 にまとめる．流体粒子法は格子法に比べると，メモリ，ディスク容量共に負荷は小さい．従って，この方法では現在の分解能を 1 桁以上上げることが容易である．

	分解能	メモリ (GB)	ディスク容量 (GB)	CPU 時間 (/job)	データ転送時間 (/job/(10MB/s))
現状	0.05	0.004	0.2	2.6 hr/Gflops	25 sec
目標	0.005	4	200	110 hr/500Gflops	7 hr

表 17: 宇宙流体力学シミュレーション（粒子法）の現状と目標

19.2.2 電磁流体力学

電磁流体力学は，宇宙流体力学と電磁場の方程式を結合したものである．電磁流体力学では，これまで流体計算に関しては非 TVD 法を用いたもの (Lax-Wendolf MHD 法) が多く用いられてきたが，今後は TVD 法を使った Roe MHD 法などの計算が望まれる．ここでは，Lax-Wendolf MHD 法を用いた現状と，Roe MHD 法を用いた目標を表 18 に示す．電磁流体力学では，電磁場の物理量が加わるため，分解能を 1 桁上げようとすると，100 GB を越えるメモリが必要になってくる．また，物理的に意味のある計算を行なうためには CPU 性能は 500 GFLOPS 以上なければならない．

	分解能 $(1/L)$	メモリ (GB)	ディスク容量 (GB)	CPU 時間 (/job)	データ転送時間 (/job/(10MB/s))
現状	1.25×10^{-2}	0.14	0.7	2 hr/Gflops	8.3min
目標	1.25×10^{-3}	136	683	250 hr/500Gflops	19 hr

表 18: 電磁流体力学シミュレーションの現状と目標

19.2.3 輻射流体力学

輻射流体力学は，宇宙流体力学と輻射輸送方程式を結合したものである．輻射流体力学は，空間 3 次元 (L^3)，輻射方向 2 次元 ($N_\theta \times N_\phi$)，輻射エネルギー（波長）空間 1 次元 (N_ν) の 6 次元問題である．このような高次元性のため，輻射流体力学は計算時間よりも主記憶容量の方がまず問題になる．

輻射輸送方程式の解法には，大別して Long Characteristic 法と Short Characteristic 法がある．前者は，計算領域の端から端にわたる長い光線をとり，この光線上で輻射輸送を解く方法である．この場合は，流体の各格子点で，一般に光線の通る位置が異なるため，コーディングが複雑になる上，並列化が難しい．一方，Short Characteristic 法は，各空間メッシュで同等の光線を張ることにより，並列化を可能にする方法である．とはいうものの，流体の物理量と輻射の 6 次元の物理量すべてを計算機のメモリーにおく余裕はない．そこで，次のような方法を考える．流体運動を決める際に必要なのは，各点での輻射強度そのものではなく，これを方向について積分した輻射のエネルギーとフラックスである．輻射輸送を解く際には輻射場の境界条件がわかっていればよいから $O(L^2 N_\theta N_\phi N_\nu)$ の 5 次元分で済むことになる．この境界条件の輻射強度を補助記憶におけば，主記憶では輻射輸送計算に必要な空間格子点での物理量分の容量が必要となる．これは，空間分解能を $L = 100$ にとった場合 172 MB になる．輻射の各方向について並列計算を行なうためには，各 PU にこれだけのメモリがなければならない．また，この計算法では，必要 PU 数は方向分解能の逆数 ($N_\theta N_\phi$) に等しい．表 19 に Short Characteristic 法による輻射流体力学の目標値をいくつかの分解能について示す．ここで，メモリは PU 当たりのメモリ容量，ディスクはシステム全体のディスク容量である．$L = 100$ は現在格子法で行なわれている空間分解能であり，少なくとも空間的にはこの値以上を実現し，これに輻射計算を加えることを目指す．

空間分解能 $(1/L)$	波長分解能 $(1/N_\nu)$	PU 数 $(= N_\theta N_\phi)$	メモリ (MB/PU)	ディスク (GB)	CPU 時間 (/job/500Gflops)	データ転送時間 (/job/(10MB/s))
0.01	0.01	1000	176	250	75 hr	35 hr
0.01	0.01	2000	176	500	150 hr	69 hr
0.01	0.005	2000	176	1000	300 hr	139 hr

表 19: 輻射流体力学シミュレーションの目標

19.2.4 結論

超並列計算機を使用する際に，計算宇宙物理に共通して言えることは，得られる結果の分解能は CPU 性能よりもメモリ容量に強く依存するということである．宇宙流体力学，電磁流体力学では，これまでの分解能を 1 桁上げるためには，全メモリとして 100 GB 以上が必要である．また，CPU 性能は 500 GFLOPS 以上が望まれる．これらの値を実現するためには 2048 台以上の PU が必要である．

また，輻射流体力学では PU 台数だけでなく各 PU のメモリ容量が重要な要素であり，意味のある計算達成のためには PU 当たり 180 MB 以上のメモリが必要である．さらにこの計算では，ディスクと PU 間のデータ転送時間が CPU 時間に匹敵する大きさになる．全計算時間軽減のためには，ディスク-PU 間で 10 MB/sec 以上の実効転送率を実現する必要がある．

20 CP-PACS 設計仕様に基づく計算物理学応用計算の仮想ベンチマーク結果

CP-PACS の設計仕様定は，前節にまとめた計算物理学からの要求性能と計算機工学・技術面からの実現可能性を勘案して定められた．この節では，CP-PACS の設計仕様に基づき，計算物理学の典型的問題の一つである，素粒子物理学の格子量子色力学の応用計算において，どの程度の実効性能が達成されるかを調べる為に行なった，仮想ベンチマークテストの結果を述べる．

20.1 ブロック・ストライド転送

並列計算機全体の高い実効性能を保つ為には，データ転送が PU 性能に比例して十分高速に行なわれる必要がある．一回の転送にかかる転送時間は，転送オーバーヘッドと，転送量を最高転送速度で割った時間との和で与えられる．ここで転送オーバーヘッドはネットワーク経由で相手の PU との接続を確保する為にソフトとハードから要求される時間で，高速転送モードでも数 μ sec 程度必要である．200 MByte/sec の転送速度の場合，これは数百 Byte 分の転送時間に相当する．小量の転送では，転送時間のほとんどが転送オーバーヘッドで決定されてしまうが，転送オーバーヘッドを大幅に削減することは技術的に困難である．従って，転送を高速化する為には，転送回数を減少させ， 1 回当たりの転送量を増大させる必要がある．

最も単純な転送機能は，メモリ上で連続，または一定間隔で並んだデータを，相手 PU のメモリにコピーするというものだが，転送データがメモリ上にどのように分布しているかは解こうとしている問題の性質に依存しており，多くの場合このパターンには当てはまらない．転送データの全ての分布パターンに対応することは，転送プロトコールの肥大化につながりかえってオーバーヘッドを大きくしてしまうが，幸い，QCD 計算をはじめとして多くの応用においてしばしば現れる転送パターンは，一定の長さの連続データが一定の間隔をおいて他数つながるという，規則的な形をしている．

そこで，CP-PACS では，この転送パターンに対応した転送機能を用意し，これを “ブロック・ストライド転送” と命名した．また，同じ形の転送命令が繰り返される時，それをより少ないペナルティーで連続するコマンド・チェイン機能も用意されている．

以下では，QCD の典型的な転送におけるネットワーク性能を評価し，ブロック・ストライド転送の効果を調べる．転送データ量と転送命令数を特徴づけるパラメータを B : 1 命令の転送量，N_{chain} : チェインされる転送命令の個数とおくと，1 回の転送における全転送量は $B_{total} = BN_{chain}$ で与えられ，転送時間 T と転送速度 R は，それぞれ，

$$T = a_s + a_h + B/\nu + (N_{chain} - 1)\{a_c + a_h + B/\nu\} + a_t,$$

$$R = B_{total}/T$$

とあらわされる．ここで，ν : 最大転送速度，a_s : software overhead，a_c : command chain overhead，a_h : hardware overhead，a_t : traveling time である．以下の評価では，$\nu = 200$ MByte/sec，$a_s = 1.5\mu$sec，$a_c = 0.5\mu$sec, $a_h = 0.5\mu$sec を仮定し，a_t は実際は通信相手までのネットワーク上の距離に依存するが $a_t = 0.5\mu$sec と仮定した．

19.1.1 に概説したように，格子 QCD の基本変数は格子の各最小線分上に定義された 3×3 複素行列 U 及び各格子点に定義された 12 次元複素ベクトル q である．従って，QCD 計算では，図 44 に示したように，倍精度複素数が 9 個（または 12 個）連続したデータを一組とし，それがある一定の間隔 (stride) で数個から数百個並んだデータを転送することが多い．(配列の宣言を工

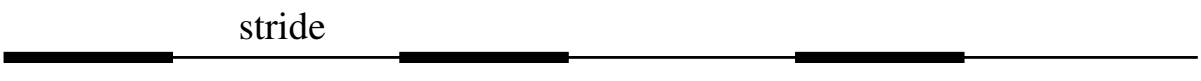


図 44: QCD 計算の典型的な転送パターン

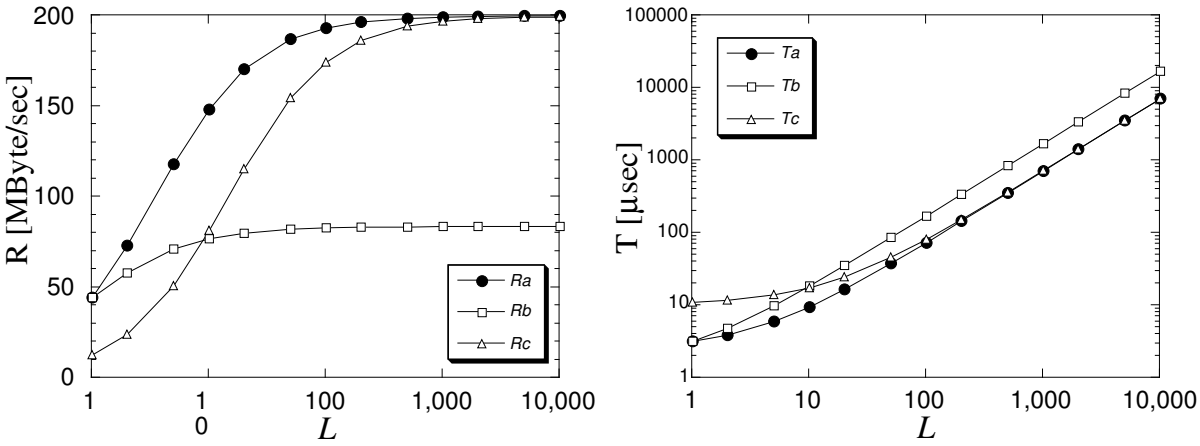


図 45: QCD の典型的な転送における転送速度と転送時間

夫することにより，重い計算の転送パターンをこの形だけにすることが出来る.）ここでは，図 44 のパターンが L 個続いたデータを

(a) そのまま 1 回のブロック・ストライド転送で送る

(b) 8 Byte × 18 の連続データの転送を，コマンド・チェインで L 回連続

(c) 16 Byte のデータが一定間隔（図 44 の stride + 8 Byte × 16）で L 個並んだデータの転送を，コマンド・チェインで 9 回連続

の 3 つの場合について，転送性能を評価する．(b) と (c) はブロック・ストライド転送がサポートされていない場合の，それぞれ L が小さい時と大きい時に最も効率的な転送方法で，(c) では，16 Byte（4 倍精度）までのデータなら一定間隔で並んだものを 1 回で転送する機能が存在すると仮定している．転送データのパラメータはそれぞれ，

(a) $N_{chain} = 1$，$B = B_{total} = 8 \times 18 \times L$

(b) $N_{chain} = L$，$B = 8 \times 18$

(c) $N_{chain} = 9$，$B = 8 \times 2 \times L$

となる．QCD の場合，問題のサイズを固定しても L は広い範囲の値を変化し，例えば，格子サイズが 64^4 で PU 構成が $8 \times 8 \times 16$ の場合には full QCD の計算で $L = 4$ — 512 である．

評価の結果を図 45 に示す．$L > 10$ では (b) より (c) の方が有利だが，ブロック・ストライド転送 (a) と比較すると $L \leq 512$ で 5 — 80 % 遅い．$L < 10$ では (c) より (b) の方が有利だが，(a) よりも 20 — 100 % 遅い．結論として，QCD 計算に最も多く現れる数十から数百程度の L の場合に通信性能の低下を防ぐ為に，ブロック・ストライド転送機能が有効であることが確認された．

20.2 仮想ベンチマーク QCD MULT の評価結果

QCD 計算において最も演算量が多く計算時間がかかる部分は，(行列) × (ベクトル) という構造をしている．即ち，

$$G(n:a) = \sum_{m=1}^{V} \sum_{b=1}^{12} D(n,m:a,b)G(m:b) \tag{3}$$

である．ここで L_i を i 方向の格子サイズとして $V = L_x \cdot L_y \cdot L_z \cdot L_t$ は格子点数であり，a,b はカラーとスピノルの自由度である．QCD の場合，行列 D は ，$n = m$ 及び n と m は隣同士，の二つの場合以外に値を持たない疎行列である．我々は上の演算を QCD MULT（または，単に MULT ）と呼び，これを仮想ベンチマークとしてノードプロセッサやネットワークの性能評価を行なった．以下，CP-PACS の PU 配列として，$8 \times 16 \times 16$ を仮定し，格子サイズは，64^4 を代表例に取って，その結果を報告する．

20.2.1 スライドウィンドウを用いた場合の QCD MULT のノードプロセッサあたり実効速度効率

CP-PACS のノードプロセッサの特徴のひとつはスライドウィンドウ機能である．この機能と preload，poststore 命令を組み合わせることでスカラープロセッサであるにもかかわらずあたかもベクトルプロセッサであるかのように動作させること（疑似ベクトルプロセッサ機能）が可能になる．この機能をフルに活用して， 単体のノードプロセッサ上での QCD MULT のアセンブラ・コードを，日立製作所の協力を得て作成した．ノードプロセッサ単体あたりの実効速度効率に対する結果は以下の通りであり，スライドウィンドウ機能を使うことで 70 % 以上の高い実効速度効率を達成するアセンブラ・コードの作成が可能となった．

(1) 浮動小数点演算数

$$N_{flop} = 1344 \cdot l_x l_y l_z L_t \tag{4}$$

但し，物理的格子 $L_x \cdot L_y \cdot L_z \cdot L_t$ の空間三次元を CP-PACS を構成する PU の三次元配列により分割する．i 方向の PU 台数を n_i とすると，$L_i = n_i \cdot l_i$ $(i = x,y,z)$ という関係がある．

(2) 演算に必要なマシンサイクル数

$$MC_{flop} = (900L_t + 219)l_x l_y l_z + 17 l_y l_z + 17 l_z + 35 \tag{5}$$

ただし，preload 命令を発行してからメモリ上のデータがノードプロセッサに届くまでに必要なマシンサイクル数は 74 以下であると仮定した．

(3) 実効速度効率

ノードプロセッサは 1 マシンサイクルあたり 2 演算可能なスーパースカラーなので，ピーク演算速度と比較したときの実効速度効率は

$$\text{実効速度効率} = \frac{N_{flop}}{2 \cdot MC_{flop}} \tag{6}$$

で与えられる．上記 (1) 及び (2) によれば，有限格子サイズの影響を無視した場合の効率は $1344/(2 \times 900) = 74.7\%$，又 64^4 格子に即して効率を計算すると 74.4 % ($L_t = 64, l_x = 8, l_y = 4, l_z = 4$ の場合)，74.3 % ($L_t = 64, l_x = l_y = l_z = 1$ の場合) となる．

20.2.2 QCD MULT の効率に対するメモリのバンクコンフリクトの影響

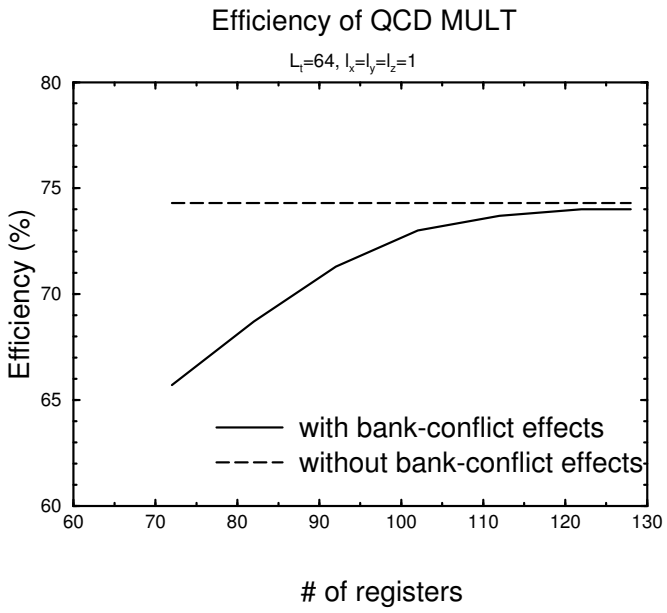


図 46: QCD MULT の効率に対するメモリのバンクコンフリクトの影響とレジスタ数との関係

上の QCD MULT の効率評価では preload 命令を発行してからデータがノードプロセッサに届くのに必要なマシンサイクル数を 74 以下と仮定した．この条件はメモリの連続アドレスからの preload では容易に満たされるが，QCD MULT のように非連続アドレスからの preload の場合，バンクコンフリクトの影響（同じバンクにアクセスしようとすると 2 回目は遅くなること）で，仮定が満たされない場合がある．バンクコンフリクトの影響を採り入れた効率を評価するために，日立製作所作成のメモリの実際の動作を反映するシミュレーターを筑波大学のワークステーション上で動かし，QCD MULT の効率に対するバンクコンフリクトの影響を評価した．シミュレーター作成時に想定したメモリの動作と，現時点で使用する予定のメモリの動作は若干異なるので，シミュレーターの結果は，バンクコンフリクトの影響を評価するだいたいの目安として考える．

図 46 に，$L_t = 64$，$l_x = l_y = l_z = 1$ の場合のシミュレーションの結果を示す．表 20 に，数値を掲げる．点線はバンクコンフリクトの影響のない場合の効率 74.4 % を示し，実線はバンクコンフリクトの影響を採り入れた効率である．横軸はスライドウインドに使用できるレジスタ数である．レジスタ数が多ければ，より” 遠く” に preload できるのでバンクコンフリクトの影響を低減することが可能になる．図 46 の結果からレジスタ数が 110 以上であればバンクコンフリクトの影響による性能低下を 1%以下に押えられることがわかった．

レジスタ数	72	82	92	102	112	122	∞
効率 (%)	65.7	68.7	71.3	73.0	73.7	74.0	74.3

表 20: QCD MULT の効率に対するメモリのバンクコンフリクトの影響とレジスタ数との関係

20.2.3 QCD MULT の効率に対する通信の影響

前述の効率は単体のノードプロセッサの性能であり，並列計算機の性能を評価するためにはノードプロセッサ間のデータ転送のための通信時間を考慮する必要がある．QCD MULT の場合にデータ転送に必要な通信時間を計算し並列計算機全体での QCD MULT の実行性能効率を評価した．結果は以下の通りであり，CP-PACS での量子色力学シミュレーションの重要目標の一つである 64^4 格子に対して， QCD MULT は並列計算機全体で 60 % に近い性能を達成可能と考えられる．

(1) 転送に必要なマシンサイクル数

$$MC_{comm} = 2(192(L_t + 2) + \Delta T)(l_y l_z + l_x l_z + l_x l_y) \tag{7}$$

ここで，ΔT は通信を起動するためのオーバヘッドであり，暫定的に $\Delta T = 1000$ マシンサイクルを仮定する．また，通信ネートワークの転送のスループットは 1 Byte/マシンサイクル とする．

(2) 転送時間と計算時間の比

この比は MC_{comm}/MC_{flop} で与えられる．$L_t = 64$，$l_x = 8, l_y = 4, l_z = 4$ の場合の値は，30.0% である．

(3) 全体の効率

転送時間を含めた全体の実効速度効率は

$$\text{実効速度効率} = \frac{N_{flop}}{2 \cdot (MC_{flop} + MC_{comm})} \tag{8}$$

で与えられる．いろいろな l_i の場合の通信時間を考慮した効率を表 21 に載せる．特に，64^4 格子に対する $L_t = 64$，$l_x = 8, l_y = 4, l_z = 4$ の場合，全体効率は 57.4% である．格子サイズが小さい場合は通信時間の比率が大きくなり効率はかなり低下する．従ってこの場合には PU を分割利用することにより通信時間を低減することが有効である．

L_t	l_x	l_y	l_z	実効速度効率
80	80/8	80/16	80/16	60.4 %
64	64/8	64/16	64/16	57.4 %
48	48/8	48/16	48/16	52.8 %
16	64/8	64/16	64/16	53.2 %
12	48/8	48/16	48/16	47.0 %
8	32/8	32/16	32/16	36.5 %

表 21: 通信時間を含めた QCD MULT の効率の格子サイズ依存性．上半分はハドロン質量計算に，下半分は有限温度シミュレーションに用いる格子サイズの典型例である．

20.3 格子量子色力学シミュレーションに要する計算時間の評価

20.3.1 quenched QCD ハドロン質量計算

格子量子色力学のシュミレーションは 1) ゲージ配位の生成，2) クォーク伝搬関数の計算，3) ハドロン物理量の計算の 3 ステップからなる．クォーク伝搬関数の計算には red/black 最小残差法を用いる．その計算ルーチン QCDMR は，最も計算時間を必要とするサブルーチン MULT に加え，内積等の演算・データ転送・ DISK I/O を含む．各々の実行時間を 20.2 節の評価結果を基礎に評価し，積み上げ式に QCDMR の全実行時間を評価した．その他のステップについては 3 ステップ相互の計算時間比の経験値を用いて，quenched QCD によるハドロン質量計算に必要な全計算時間の評価を行なった．格子サイズは 64^4 を仮定し，各ステップで必要な格子パラメタの値は 19.1 節「素粒子物理学に於ける格子量子色力学」での値を踏襲した．

性能評価の仮定は，以下の通り．

PU 構成: $8 \times 16 \times 16 = 2048$ PU + $8 \times 16 = 128$ IOU
PU 性能: クロック 150 MHz 2 命令スーパースカラー + PVP-SW
ネットワーク 性能: スループット =150 MB/sec レイテンシ = 7 micro sec
ディスク性能: RAID-5 SCSI2 ブロックサイズ = 32KB
クォーク質量: 21 MeV 以上の 6 点 (m_π/m_ρ =0.9,0.8,0.7,0.6,0.5,0.4)
問題規模: 64^4 格子 3000 スイープ 毎に 500 ゲージ 配位
アルゴリズム: Red/Black 最小残差法

表 22(a) に QCDMR 繰り返し一回に要する実行時間を示す．表 22(b) に，一配位当り 6 種類のクォーク質量に対するクォーク伝播関数を QCDMR により求めるのに要する実行時間を示す．表 22(c) に，総計算時間とその内訳を示す．

64^4 格子に対するシュミレーションの総計算時間は 58 CPU 日 であり，これを 3 つの格子間隔に対して繰り返すとすれば，クエンチ近似のハドロン質量計算の総計算時間は 174 CPU 日 である．

20.3.2 full QCD ハドロン質量計算

full QCD のシュミレーションも，quenched QCD 同様 1) ゲージ配位の生成，2) クォーク伝搬関数の計算，3) ハドロン物理量の計算の 3 ステップからなる．full QCD の場合，計算の大部分は ゲージ配位の生成に必要な正規乱数ソースに対するクォーク伝播関数の計算に費やされる．従って，ここでは配位生成時間に主要部分を占める QCDMR の実行時間の評価により，配位生成時間の評価を行なった．

表 23 に $32^3 \times 64$ 格子及び $48^3 \times 64$ 格子の場合の場合の QCDMR 繰り返し一回あたりの実行時間の内訳を示す．

表 24 に，ゲージ配位生成時間を，代表的クォーク質量の各値に対して示す．

20.3.3 有限温度 QCD のシュミレーション

有限温度 QCD のシュミレーションは，空間方向の格子サイズ L_s が 時間方向の格子サイズ L_t よりも大きい格子での full QCD 計算を必要とする．従って，計算時間のほとんどは，ハイブリッ

(a) 格子 QCDMR 繰り返し一回の実行時間.

	# flop/PU	演算時間 (sec)	通信時間 (sec)	実行時間 (sec)
MULT	11010048	0.0495	0.0	0.0495
QCDMR	3145728	0.0123	0.0133	0.0256
合計	14155776	0.0618	0.0133	0.0752
実行時間比		82.3 %	17.7 %	100 %

(b)quenched QCD 一配位当たりのクォーク伝播関数計算時間.

	演算	通信	DISK I/O	合計
時間 (hour)	0.882	0.190	0.373	1.445
比	61.0 %	13.2 %	25.8 %	100 %

(c) quenched QCD ハドロン質量計算必要日数.

	ゲージ配位生成	クォーク伝搬関数	ハドロン物理量	合計
時間 (days)	22.2	30.1	5.7	58.0
比	38.3 %	51.9 %	9.8 %	100 %

表 22: 64^4 格子 quenched QCD ハドロン質量計算に要する総計算時間及び内訳.(flop=浮動小数点演算)

$(a)32^3 \times 64$ 格子

	# flop/PU	演算時間 (sec)	通信時間 (sec)	実行時間 (sec)
MULT	1376256	0.00619	0.0	0.00619
QCDMR	393216	0.00154	0.00333	0.00487
合計	1769472	0.00773	0.00333	0.01106
実行時間比		69.9 %	30.1 %	100 %

$(b)48^3 \times 64$ 格子

	# flop/PU	演算時間 (sec)	通信時間 (sec)	実行時間 (sec)
MULT	4644864	0.02089	0.0	0.02089
QCDMR	1327104	0.00520	0.00750	0.01270
合計	5971968	0.02609	0.00750	0.03359
実行時間比		77.7 %	22.3 %	100 %

表 23: full QCD の場合の QCDMR 繰り返し一回あたりの実行時間

ドモンンテカルロ法でのゲージ配位生成における MULT の実行によるので，ここでは，MULT の全実行時間を評価する．PU 単体は前節と同じくクロック 150 MHz とし，通信スループット 150 MByte/sec，通信オーバーヘッド 1000 マシンサイクルを仮定し，PU 構成は $8 \times 16 \times 16$ とする．

格子サイズを $L_x \times L_y \times L_z \times L_t$ とし，各 PU 内の格子サイズを $l_x \times l_y \times l_z \times L_t$ ($l_x = L_x/8$, $l_y = L_y/16$, $l_z = L_z/16$) とすると，1 回の MULT の演算と通信に要求される計算ステップ数は，それぞれ (5), (7) で与えられる．ハイブリッドモンテカルロ法では，配位を 1 分子動力学時間ステップ $\delta\tau$ だけ進める毎に逆行列計算が 2 回要求される．1 回の逆行列計算に呼び出される MULT の回数 N_{iter} は表 12 を仮定し，ステップ幅は $\delta\tau = 0.02 \times (8/L)$ を仮定した．

表 25 に $\tau = 2000$ だけシミュレーションするのに必要な計算日数をまとめる．配位の蓄積に要

格子サイズ		m_π/m_ρ			
L_t	L_s	0.7	0.6	0.5	0.4
64	48	210	367	618	1067
64	32	46	81	136	234

表 24: full QCD シュミレーション の ゲージ配位生成時間（日）．但し，空間的に等方的な格子をとり，$L_x = L_y = L_z = L_s$ とする．分子動力学ステップサイズは $\delta\tau = 0.02(8/L_s)$ を仮定し，$\tau = 5$ 毎に 500 ゲージ 配位 を生成する場合の評価．

求される Disk I/O 時間などを無視すれば，これが，ひとつのデータを計算するのに必要な時間である．実際のシミュレーションでは，各格子サイズについて，結合定数のパラメーター空間の十点程度でこの計算を繰り返す必要がある．

格子サイズ		m_π/m_ρ			
L_t	L_s	0.7	0.5	0.3	0.18
16	64	81.0	238	781	2279
12	48	21.7	64.0	210	612
8	32	3.7	10.8	35.4	103

表 25: 有限温度 QCD シュミレーションで，$\tau = 2,000$ の計算に要する計算日数．空間的に等方的な格子を考え，$L_x = L_y = L_z \equiv L_s$ とする．

20.4 KS fermion の場合の実効性能評価

20.4.1 QCD MULT の効率

格子 QCD の数値シミュレーションでは，我々が考察してきた Wilson fermion と呼ばれているクォークの作用関数以外にも，KS fermion と呼ばれる別の形のクォークの作用関数も使われている．そこで，我々は KS fermion の場合の QCD MULT の効率も同様に考察した．ここに，その結果を簡単に紹介する．

(1) 浮動小数点演算数

$$N_{flop} = 624 l_x l_y l_z L_t \tag{9}$$

(2) 演算に必要なマシンサイクル数

$$MC_{flop} = 384 L_t l_x l_y l_z \tag{10}$$

ただし，計算の立ち上がりに必要な部分のマシンサイクル数は除いた．

(3) 効率

上記 (1) と (2) から 理論性能は 81.25 % である．

(4) データ転送に必要なマシンサイクル数

$$MC_{comm} = 2(48(L_t + 2) + \Delta T)(l_y l_z + l_x l_z + l_x l_y) \tag{11}$$

ここで，ΔT は前述の，通信の起動のためのオーバヘッドである．

(5) 全体の効率

$L_t = 64$，$l_x = 8$，$l_y = 4$，$l_z = 4$ の場合，

$$\text{全体の効率} = \frac{N_{flop}}{2(MC_{flop} + MC_{comm})} = 67.0\% \tag{12}$$

いろいろな l_i の場合の全体効率を表 26 に載せておく．

L_t	l_x	l_y	l_z	実効速度効率
80	80/8	80/16	80/16	70.0%
64	64/8	64/16	64/16	67.0%
48	48/8	48/16	48/16	62.1%
16	64/8	64/16	64/16	58.9 %
12	48/8	48/16	48/16	50.6 %
8	32/8	32/16	32/16	36.9 %

表 26: KS fermion の場合の QCD MULT の全体効率の格子サイズ依存性．上半分はハドロン質量計算に，下半分は有限温度シミュレーションに用いる格子サイズの典型例である．

20.4.2 KS fermion の場合の QCD シミュレーション計算時間評価

KS fermion を用いた場合の QCD シュミレーションに必要な計算時間の評価をまとめる．quenched QCD の場合に，クォーク伝播関数を計算するのに要する時間をクォーク質量（格子単位）の関数として表 27 に示す．但し，ゲージ配位は既に生成済みとし，その為に必要な時間は含まない．クォーク伝播関数計算には共役傾斜法（CG と略称；繰り返し一回当たりの MULT 呼びだし回数二回）を用い，通信オーバヘッドは $\delta T = 1000$ MC を仮定した．

格子サイズ		m_q			
L_t	L_s	0.02	0.01	0.005	0.0025
64	64	1.8	3.5	7.0	14.1
48	48	0.6	1.2	2.4	4.8

表 27: KS fermion の場合の quenched QCD ハドロン質量計算於けるクォーク伝播関数計算時間（CPU 日）．空間的に等方的な格子を考え，$L_x = L_y = L_z \equiv L_s$ とする．ゲージ配位数は 500，CG 法の収束回数は経験値 $1000(0.01/m_q)$ を仮定．

表 28 に full QCD の場合について，配位生成に要する時間を示す．配位生成法はハイブリッドモンテカルロ法の変形であるハイブリッド R 法を用い，分子動力学ステップサイズは，経験値 $\delta\tau = 0.01(m_q/0.01)$ を仮定する．クォーク逆行列の計算は共役傾斜法により，収束の為の繰り返し回数は経験値 $500(0.01/m_q)$ を用いた．

20.5 結論

以上の仮想ベンチマーク結果は，PU 台数 2048 に対して，19.1 節で評価した必要計算能力がほぼ満足されることを示す．特に，quenched QCD については，より計算時間を要する Wilson fermion 法に対しても CPU 時間 180 日程度を用いることにより，系統誤差と統計誤差双方を充

格子サイズ		m_q			
L_t	L_s	0.02	0.01	0.005	0.0025
64	64	11	40	154	601
64	48	4.9	18	69	268
64	32	1.6	5.9	23	88
16	64	2.1	8.3	33	133
12	48	0.8	3.0	12.2	48.7
8	32	0.2	0.8	3.3	13

表 28: KS fermion の場合の full QCD 配位生成に要する計算時間（CPU 日）．空間的に等方的な格子を考え，$L_x = L_y = L_z \equiv L_s$ とする．表の上半分はハドロン質量計算に典型的格子サイズであり，配位間隔 $\tau = 5$，配位数 500 を仮定．下半分は有限温度の場合であり，$\tau = 2000$ に対する評価値を示す．ステップサイズは $\delta\tau = 0.01(m_q/0.01)$，CG 法の収束回数は $500(0.01/m_q)$ を仮定する．

分に押えた計算が，64^4 格子規模に対して可能である．

full QCD については，KS fermion 作用を用いて場合には，PU 台数 2048 台により, 64^4 規模の計算が CPU 時間 1 カ年程度で可能と考えられる．一方，Wilson fermion 作用の場合は，32^4 格子規模の計算が既に CPU 時間 1 ヶ年程度以上を要する．但し，Wilson fermion 作用を用いた場合，ここでの評価に仮定した red/black 最小残差法の収束回数は 1/2 程度に緩めても結果に影響を及ぼさない可能性がある．また，ILU 法による前処理は収束回数を 1/4-1/5 に低減する効果があることが知られおり，これを効率的に並列プログラムできれば計算時間の一層の削減に継る．以上の諸点の検討により，Wilson fermion 作用を用いた full QCD に対して実行可能な格子規模の拡大を計ることは今後の重要課題である．

21 CP-PACS 上での計算物理学応用プログラムの開発と性能評価

21.1 素粒子物理学における格子量子色力学

CP-PACS で用いるための素粒子物理学格子 QCD プログラムの開発は，平成 7 年秋から本格化した．特に，平成 8 年 4 月に 1024 PU 構成機が稼働を開始した後には，各種の格子 QCD プログラムのデバッグ，基本的な演算及び通信部分の詳細な性能評価とそれに基づくチューニングが集中的に行なわれた．プログラミングは主として FORTRAN90 によりなされたが，QCDMULT 等特に高い実効効率を要する部分については，アセンブラプログラムが作成された．以上の努力の結果，基本的な格子 QCD プログラムについては，平成 8 年秋までに高効率のプログラム群を整備することができた．

表 29 に，本プロジェクトにより開発された主な格子 QCD 用プログラムの性能をまとめる．以下，本節では，格子 QCD 計算に対する CP-PACS 実機の基本性能，開発されたプログラム群の内容，実ジョブ実行時間を述べる．

21.1.1 基本性能

この章では，格子 QCD 計算の重要な部分における CP-PACS の基本性能について，演算・通信・disk I/O の各側面から述べる．特にことわらない限り，FORTRAN90 によるプログラミングである．

3 × 3 複素行列積 quenched QCD のゲージ配位生成計算の中心部分は，3 × 3 複素行列の行列積 `W = U1*U2` の計算である．ループ アンローリングのやり方にも依るが，乗算・加減算・load/store の三種の命令の数が比較的よくバランスしており，CP-PACS の疑似ベクトル機能によって高速に演算できる．但し，現在の CP-PACS の FORTRAN コンパイラを使って効率の良い疑似ベクトル コードを生成するためには，適当なループ アンローリングを行って，ひとつのループ内の load/store の数を 20 個程度以下にしておく必要がある．ここでは，3 × 3 複素行列の行列積の 9 つの要素の計算を，2 要素 ×4+ 1 要素に分割した．load した変数の再利用率やバンクコンフリクトの軽減も考慮に入れ，以下の例のようなループ分割を採用した：

```
      complex*16 W(3,3,*),U1(3,3,*),U2(3,3,*)

      do m=0,Nloop
         W(1,1,m) = U1(1,1,m) * U2(1,1,m)
     *            + U1(1,2,m) * U2(2,1,m)
     *            + U1(1,3,m) * U2(3,1,m)
         W(2,1,m) = U1(2,1,m) * U2(1,1,m)
     *            + U1(2,2,m) * U2(2,1,m)
     *            + U1(2,3,m) * U2(3,1,m)
      enddo
      do m=0,Nloop
         W(1,2,m) = U1(1,1,m) * U2(1,2,m)
     *            + U1(1,2,m) * U2(2,2,m)
     *            + U1(1,3,m) * U2(3,2,m)
         W(2,2,m) = U1(2,1,m) * U2(1,2,m)
     *            + U1(2,2,m) * U2(2,2,m)
     *            + U1(2,3,m) * U2(3,2,m)
      enddo
      do m=0,Nloop
```

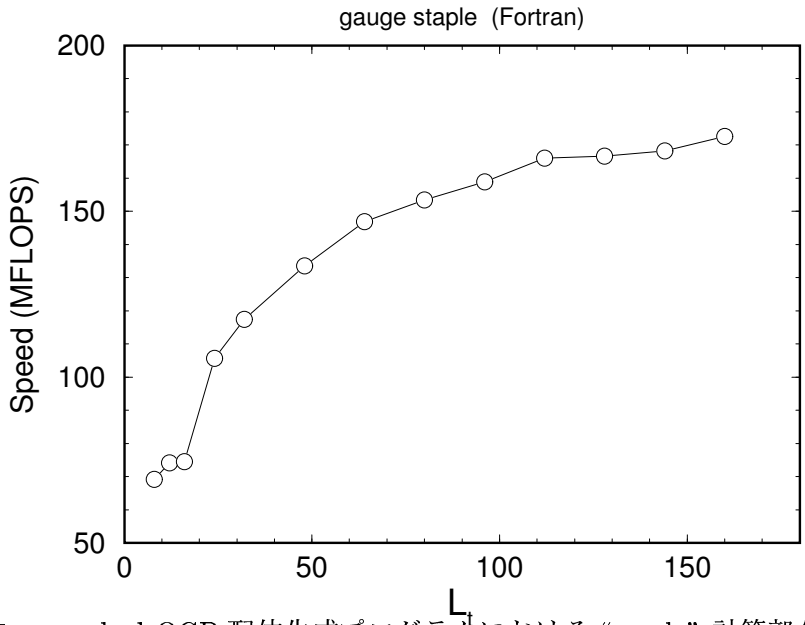


図 47: quenched QCD 配位生成プログラムにおける “staple” 計算部分の性能.

```
     W(3,1,m) = U1(3,1,m) * U2(1,1,m)
*             + U1(3,2,m) * U2(2,1,m)
*             + U1(3,3,m) * U2(3,1,m)
     W(3,2,m) = U1(3,1,m) * U2(1,2,m)
*             + U1(3,2,m) * U2(2,2,m)
*             + U1(3,3,m) * U2(3,2,m)
 enddo
 do m=0,Nloop
     W(1,3,m) = U1(1,1,m) * U2(1,3,m)
*             + U1(1,2,m) * U2(2,3,m)
*             + U1(1,3,m) * U2(3,3,m)
     W(2,3,m) = U1(2,1,m) * U2(1,3,m)
*             + U1(2,2,m) * U2(2,3,m)
*             + U1(2,3,m) * U2(3,3,m)
 enddo
 do m=0,Nloop
     W(3,3,m) = U1(3,1,m) * U2(1,3,m)
*             + U1(3,2,m) * U2(2,3,m)
*             + U1(3,3,m) * U2(3,3,m)
 enddo
```

（実プログラムでは `U1(i,j,m)`, `U2(i,j,m)` は `CONJG(U(j,i,m))` の形で現れることもある.）

このタイプの計算と `W = W + U1*U2` の形の計算を組み合わせて，“staple” と呼ばれる行列 3 個の積 `U1*U2*U3` の和を計算する．図 47 に実プログラムにおける staple 計算部分の PU あたりの性能を示す．横軸は時間方向の格子サイズ L_t で，ベクトル長は even/odd algorithm を採用しているため，その半分 $L_t/2$ である．$L_t > 64$ で PU ピーク性能の 50% を越え，$L_t = 160$ で 173 MFLOPS/PU を達成している.

QCDMULT　格子 QCD 計算に於いては，多くの CPU 時間が QCDMULT と呼ばれる基本ループの演算に費やされる．full QCD のグルオン配位の生成や，クォーク伝搬関数の計算では，大規模線形方程式 $p = Dq$ を解く必要がある．D はグルオン配位 $\{U\}$ の関数であり，クォーク

プログラム	性能 (MFLOPS/PU)			コード	格子サイズ	PU 数
クォーク solver (red/black 最小残差法) Wilson クォーク	計算	(83%)	191	アセンブラ ＋ FORTRAN	$64^3 \times 112$	2048
	通信	(17%)	–			
	全体		159			
	計算	(80%)	191	アセンブラ ＋ FORTRAN	$48^3 \times 64$	1024
	通信	(20%)	–			
	全体		152			
	計算	(84%)	99	FORTRAN	$48^3 \times 64$	512
	通信	(16%)	–			
	全体		84			
クォーク solver (共役傾斜法) staggered クォーク	計算	(90%)	139	FORTRAN	$32^3 \times 128$	256
	通信	(10%)	–			
	全体		125			
ゲージ配位生成 (heat-bath 法) quenched QCD	計算	(95%)	126	FORTRAN	$64^3 \times 112$	2048
	通信	(5%)	–			
	全体		120			
ゲージ配位更新 (over-relaxation 法) quenched QCD	計算	(89%)	153	FORTRAN	$64^3 \times 112$	2048
	通信	(11%)	–			
	全体		136			
ゲージ配位生成 (hybrid Monte Carlo 法) full QCD	計算	(74%)	151	アセンブラ ＋ FORTRAN	$16^3 \times 32$	256
	通信	(26%)	–			
	全体		112			

表 29: CP-PACS における主要な格子ＱＣＤ応用プログラムの性能．PU あたりのピーク性能は 300 MFLOPS/PU である．

場 q を $12V$ (V:格子点数) 要素の列ベクトルとみた場合，$12V \times 12V$ の疎行列とみる事ができる．線形方程式を反復法によって解く際の行列 D にベクトル q を掛ける演算 Dq を行なうルーチンが，QCDMULT である．

QCDMULT の主要部は，“演算に必要な乗算の数・加減算の数・load/store の数（三種の命令）がよくバランスしている” という特徴を持っている．各格子点での演算は，本質的には， 3 列の複素ベクトルの和の計算と 3 行 3 列の複素行列を 3 列の複素ベクトルに乗ずる演算:

$$y^a = x^a + x'^a \quad ; \quad z^a = \sum_b U^{ab} y^b \qquad (a = 1, 2, 3) \tag{13}$$

が中心であり，異なる配列要素に対して，この種の演算が幾つも必要となる．この演算に対しては，ループ変数 a についてのアンローリングし，作業変数 y はレジスタに置くものとすると，乗算・加減算・load / store 数はいずれも 36 となり，完全にバランスする．CP-PACS のプロセッサは 2 命令スーパスカラで，乗算と加減算とは 1 命令で実行される． 1 命令で load/store を，1 命令で乗算加減算命令を指定することにより，三種の命令を同じ ウエイトで実行することが可能となる．QCDMULT を高速に実行する為には，この様なコードを生成する必要がある.（なお QCDMULT には，三種の命令のバランスが悪い演算が一部含まれている．以下に述べる QCDMULT の効率

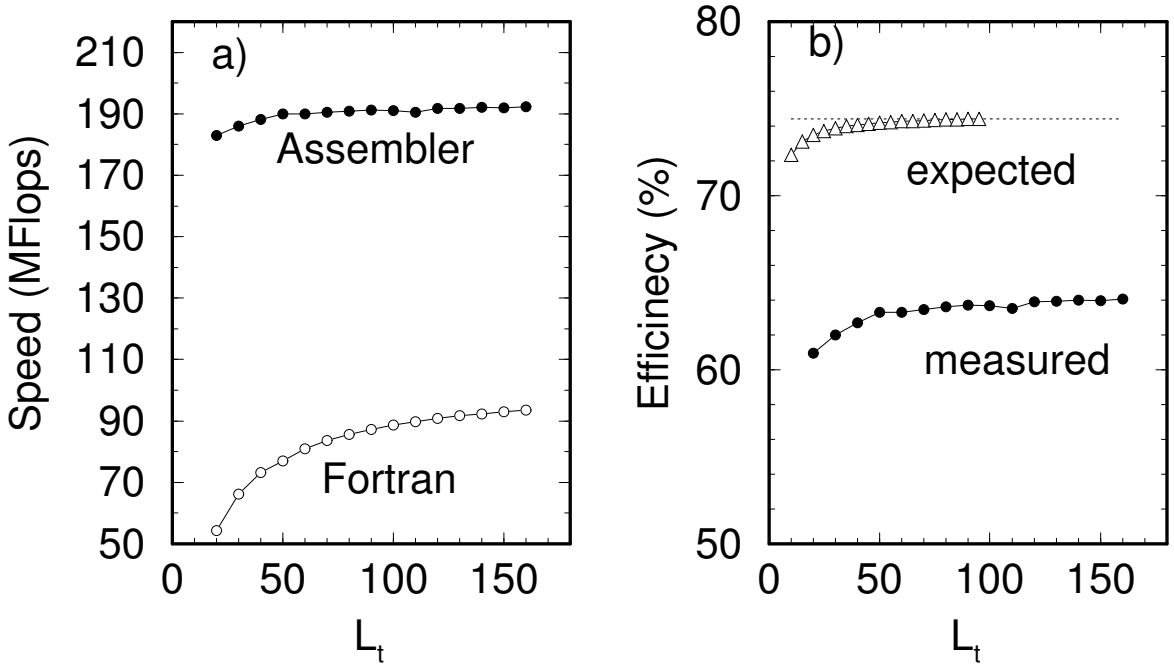


図 48: QCDMULT の (a) 実効速度 と (b) 実効効率．詳細は本文参照.

は，バランスの悪い演算も含んだルーチン全体の効率である.）

我々は，まず，FORTRAN で QCDMULT をコーディングし，チューニングを試みた．式 (13) を a でアンローリングしたものは，演算が複雑で必要となるレジスタ数も多く，現在の FORTRAN コンパイラでは，疑似ベクトル機能を用いた最良のスケジューリングを行うことはできない． 次善の方法として，上記のループを 2 つ組み合わせ，それを 3 つの do-loop に分割した．この為，変数 U の余分な load と作業変数の load/store が生じ，性能低下の原因となる．ループ分割の基本方針は，

- 疑似ベクトル化可能な範囲で，なるべく多くの演算を一つの do-loop で行なう.
- 一つの do-loop 内で，乗算・加算がなるべくバランスする様分割する.
- 余分な load/store を極力避ける.

とした．この方針のもとに試行錯誤を繰り返し，最良のコードを得た．また，バンクコンフリクトを避けるため，グルオン変数の配列・クォーク変数の配列を一つの combuf 領域に取り，配列間に適当なダミー変数を挿入した（作業配列に関しても同様）．バンクコンフリクトを避ける工夫は，演算に比べ load/store が重いループでは極めて重要である．QCDMULT の場合，単純にコーディングした場合 30 MFLOPS/PU 程度の性能しかでなかったが，ダミー変数の挿入だけで，倍以上の性能が得られた.

QCDMULT の 各 do-loop は格子の時間方向に疑似ベクトル化を行ない，残りの 3 つの空間方向に対しては，その外側の do-loop による繰り返しとした．図 48（a） に ○で FORTRAN による QCDMULT の実効速度を示す．横軸は時間方向の格子サイズ L_t であり，ベクトル長は even/odd algorithm を採用しているため，その半分 $L_t/2$ である．ベクトル長が十分長い場合，94 MFLOPS/PU の実効速度を得た．これは，理論 peak speed の 30 % 程度の性能である.

FORTRAN でコーディングした場合の効率は他の商用プロセッサ上の効率と比較して，決して悪いものではないが，CP-PACS の疑似ベクトル機構をフルに生かしているとは言えない．特に，

性能低下の要因	性能	効率
乗算・加算のアンバランス	266 MFLOPS	88 %
Window 切替え	243 MFLOPS	81 %
address 計算等	225 MFLOPS	75 %
memory access latency	193 MFLOPS	64 %

表 30: QCDMULT の性能低下の要因とその要因を考慮に入れた場合の性能と効率.

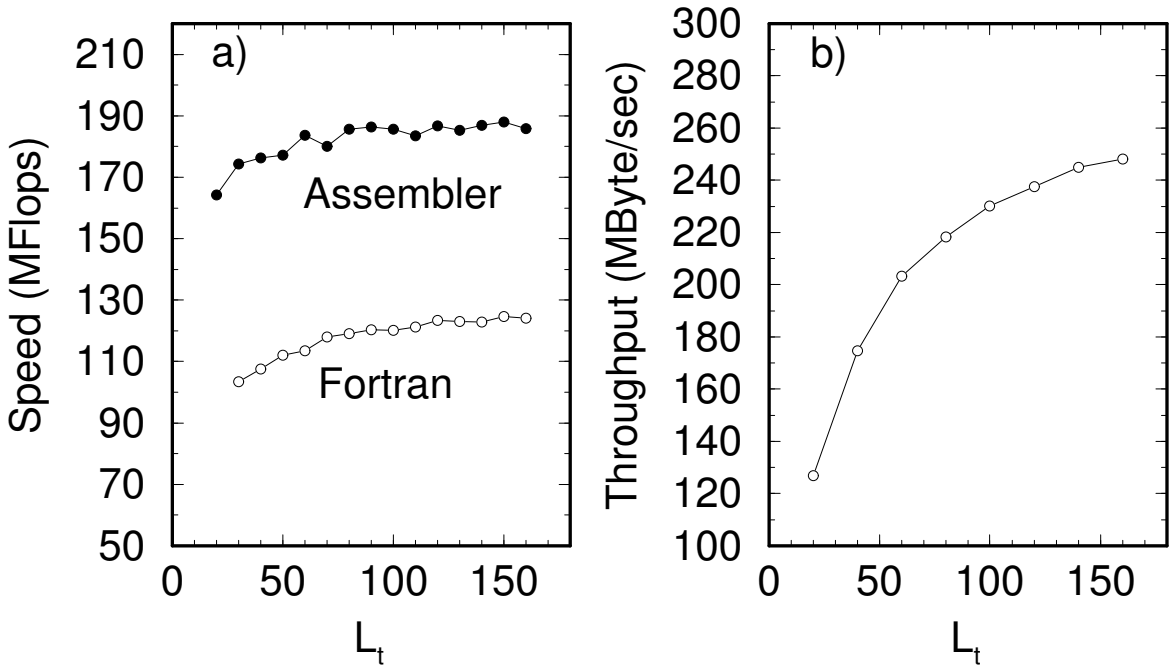
QCDMULT の場合は，ループ中で余分な load/store を行わず，ウィンドウを切り替えながら演算をすべてレジスタ上で行うことによって，高い効率を実現することが可能である．我々は，日立製作所の協力を得てこの様なアセンブラコードを作成し，実ジョブではこれを用いた．ウィンドウは 12 マシンサイクル毎に，10 レジスタづつ更新され，式 (13) の演算は，あらかじめレジスタに preload されたデータを用いて，4 つのウインドウにまたがって計算され， 5 番目のウィンドウで結果が poststore される．同種の演算を入れ子にする事によって，一つのウィンドウで実行される 12 マシンサイクルには，乗算加減算命令と load/store 命令が密に詰められる．図 48 (a) に ●でアセンブラによる QCDMULT の実効速度を示す.

QCDMULT 全体では，前述の三種の命令は完全にバランスしているわけではないので，各マシンサイクルを 3 種の命令で完全に満たすことはできない．ループ全体では，加減算 > 乗算 > load/store の順である．乗算・加減算のアンバランスのみ考慮に入れた場合の理論最高速度は 266 MFLOPS/PU（効率 88 %）であり，実測値は 193 MFLOPS/PU（効率 64 %）である．性能低下の要因には，Window 切り替え，address 計算等の非浮動小数点演算，有限のメモリアクセスレイテンシ（バンクコンフリクトの効果，ストアに 2 マシンサイクル必要である事を含む）がある．表 30 に，性能低下の各要因と，その要因を考慮に入れた場合の性能・効率をまとめた.

図 48（a） に示したように，実効効率のベクトル長依存性はかなり小さく，短いベクトルに対しても高い効率が得られる．これは疑似ベクトル化の特徴である．特に，アセンブラでコーディングした場合，この傾向が著しい．QCDMULT のアセンブラコードは，式 (13) と同種の演算 12 組と，三種の命令が完全にはバランスしない演算 4 組を一つのループにまとめているので，ループ本体が大きく，prologue/epilogue 部が相対的に小さい為，効率の立ち上がりが極めてよくなっている.

QCDMULT は，格子 QCD 計算の最も基本的なルーチンであるので，CP-PACS の設計段階から仮想ベンチマークを繰り返してきた（第 20 章参照）．図 48（b） に第 20.2 節に記載した仮想ベンチマークの結果（△）を再録し，併せて実機による実測値（●）を示す．10% 程度の違いが見られるが，これらは仮想ベンチマーク実施後，プロセッサ及びメモリの仕様に幾つか変更があった事，実測値は even/odd 法を適用した MULT に対するものであるのに対し，仮想ベンチマークはこれを用いない full MULT に対するものであること等による．なお，$L_t = 64$ の場合の full MULT の実測値は 202 MFLOPS/PU（効率 67 % ）であり，even/odd MULT の実測値は 190 MFLOPS/PU（効率 63 %）である.

QCDMR QCDMULT の項で述べた，線形方程式 $p = Dq$ を最小残差法により反復的に解く際の，演算 Dq 以外の部分を QCDMR と呼ぶ．QCDMR の演算量は，QCDMULT の 15 % 程度である．乗算の数・加減算の数は完全にバランスしていて，load/store 数も乗算・加減算の数より幾分小さい．演算は単純なので，FORTRAN でもそこそこの性能を得る事が可能であるが，



’ 図 49: QCDMR の (a) 演算実効速度 と (b) 通信スループット．詳細は本文参照.

よりよい性能を発揮するためにはやはり，アセンブラでのコーディングが必要である．図 49（a）に，QCDMR の演算部の実効速度を示す．ベクトル長が十分長い場合，Fortran では 40 % の，アセンブラでは 62 % の効率が得られた．

アセンブラコードでは，ウィンドウ切り替えを除くと，すべてのマシンサイクルが乗算・加算命令で埋め尽くされている．性能低下の主な要因は，ストアネック（ロード命令が 1 マシンサイクルで実行されるのに対して，ストア命令の実行には実質 2 マシンサイクル必要であること）及びバンクコンフリクトである．メモリスループットが無限大であれば 257 MFLOPS/PU が期待されるところを，実測値は約 190 MFLOPS/PU である．

基本通信性能　QCDMR では，クォーク変数の隣接 PU との境界データの通信が必要となる． 4 次元の時空データのうち空間方向を 3 次元 PU アレイに分割しているので，境界は時間 1 次元，空間 2 次元であり，各 PU には時間方向のすべてのデータが存在する．空間座標一定の時間方向のデータ列は，12 個の複素数データ（8×24 byte）をブロック長とする， 1 回のブロックストライド転送を用いて隣接 PU に転送する事ができる．図 49（b） に，QCDMR のデータ転送スループットを時間方向の格子サイズの関数として示した．

クォーク伝搬関数計算の場合，通信時間を全実行時間の 20 % 程度以下にする為には，ネットワーク性能 “MByte/sec” 値と演算性能 “MFLOPS/PU” 値との比が 1 以上が必要であり，この比は，並列計算機の性能のバランスの良し悪しの一つの指標である．$L_t \sim 64$ の場合，実効演算性能は約 200 MFLOPS/PU であるのに対し，転送スループットは 200 MByte/sec であり，$L_t \sim 64$ の計算を行う quench 近似計算に対しては，演算性能にほぼ見合った通信性能が実現されている．

分散ディスク 基本 I/O 性能　QCDMR で計算されたクォーク伝搬関数は CP-PACS の分散ディスクに一時ストアされた後，次のステップでハドロン伝搬関数を計算するために再び読み込まれる．クエンチ近似のハドロン質量計算の場合，問題規模にも依るが，実行時間の 15–20 % が disk I/O に費やされる．実行時間の短縮の為，ディスク I/O の高速化の多くの工夫を行った． 高速

化に大きな効果のあった工夫は以下の通りである．

- 可能な限り，ディスクへの sequentail access を行う．

  クォークの伝搬関数は計算され次第ディスクへストアされる．一つの IOU には，y 座標が同一の複数 PU からのデータがストアされるが，同時に各 PU から write 命令を発行した場合，IOU ではブロック単位でシリアライズして処理されるので，同一ファイルがディスク上に不連続的に配置される．次回，同一ファイルに書き込みを行う際，タイミングにより処理順が異なるので，ディスクへのランダムアクセスが生じる．一般に，ディスクへのランダムアクセスはシーケンシャルアクセスに比べ極めて遅い．ディスクアクセスを高速化する為に，y 座標が同一の PU からの write 命令をシリアライズし，同一ファイルがディスク上連続的に配置される様にした．write 命令の順を守る限り，ディスクにはシーケンシャルにアクセスされる．

- C の I/O 用関数の利用．

  フォートランの I/O 処理は C のそれに比べて遅いので，フォートランの中から C の fread, fwrite 等の I/O 関数を直接呼び出して利用した．

- 大きなバッファサイズの利用．

  C の I/O 関数を利用する場合，setvbuf を用いて大きなバッファを用いると I/O 性能が大きく改善される．バッファによるメモリフラグメンテーションを避けるため，バッファはユーザーが combuf 領域に cacheable 属性で確保したものを用いた．

その他細かい工夫によって，クエンチ近似のハドロン質量計算のジョブでは，クォークステップの sequentail write で約 5 MB/sec/IOU ，ハドロンステップの random read で約 1.3 MB/sec/IOU という，実効速度が実現された．これは，ほぼ設計値通りである．

21.1.2 応用プログラム

CP-PACS における格子 QCD の応用プログラムは主に FORTRAN90 で書かれており，次の 4 つの部分から構成されている．

1. ゲージ配位生成 (“update”) : 格子作用に従った分布でゲージ場の配位 {U} を Monte Carlo 法を使って生成し，分散ディスクに書き込む．
2. ゲージ固定 : 物理量測定における統計ノイズの低減のため，ゲージ配位をゲージ固定して分散ディスクに書き込む．
3. クォーク solver :（ゲージ固定された）ゲージ配位を読み込み，クォークの伝搬関数を計算し，分散ディスクに書き込む．
4. ハドロン物理量測定 : クォーク伝搬関数を分散ディスクから読みだし，それらを組み合わせてハドロンの物理的観測量を構成し，出力する．

実ジョブでは，上記の 4 つの部分をセットとして，ゲージ配位を受け渡しながら，全体を数百から数千回繰り返し実行する．

ゲージ配位生成部分の計算は，quenched QCD と full QCD とで大きく異なる．また，クォーク solver 部分のプログラムも，格子上のクォークとして Wilson クォークを採るか KS クォークを採るかで大きく異なる．

これらのプログラムでは，4 次元の時空格子 $L_x \times L_y \times L_z \times L_t$ の 3 次元空間部分 $L_x \times L_y \times L_z$ を，CP-PACS の 3 次元の PU 格子 `Ndimx` × `Ndimy` × `Ndimz` に単純マッピングした．従って個々の PU 上ではゲージ場とクォーク場はそれぞれ

$$\texttt{complex*16}\quad \texttt{U}(3,3,0:L_t{+}1,0:N_z{+}1,0:N_y{+}1,0:N_x{+}1,4) \tag{14}$$

$$\texttt{complex*16}\quad \texttt{G}(3,4,0:L_t{+}1,0:N_z{+}1,0:N_y{+}1,0:N_x{+}1) \tag{15}$$

であらわされる．ここで各 PU 上の部分格子は $N_x \times N_y \times N_z \times L_t$ ($N_x = L_x/$`Ndimx` *etc.*) で，配列 `U` と `G` は，隣接 PU 上の対応する場の量のコピーである袖（`ix=0`，N_x`+1`，等）を含んでいる．この袖の部分は，その変数が更新される度に隣接 PU に通信によってコピーされなければならない．

通常のベクトル型計算機では高い性能を得るために大きなベクトル長が要求されており，ベクトル長を長くするするためにしばしば座標変数を 1 次元化したインデックスを導入する．CP-PACS では，前節で述べたように，アセンブラ化されたプログラムのコア部分 QCDMULT でベクトルの立ち上がりが極めて速いので，t 座標に関するループのみをベクトル化した．これにより，プログラムが極めて読みやすいものになり，プログラム開発とデバッグの効率が上がった．

アセンブラ化されていない部分におけるベクトルの立ち上がりは QCDMULT ほど速くはないが，ソフトウエア・パイプライニングによる疑似ベクトル化によって，通常のベクトル型計算機よりかなり速い立ち上がりを実現している．現在まで CP-PACS 上で主に行ってきたゼロ温度 QCD シミュレーションでは，時間方向の格子サイズ L_t が空間方向のサイズ L_x，L_y，L_z よりも大きく，計算に使った L_t で十分効率の良いベクトル化が可能である（前節，及び 21.1.3 節参照）．

今後予定している有限温度 QCD の研究では，時間方向の格子サイズ L_t が小さく 空間方向のサイズ L_x，L_y，L_z が大きい格子をシミュレーションする必要がある．L_t は 4 から最大 12 程度までで，この方向を分割することは通信量を抑えるためには望ましくない．他方，ゼロ温度の研究で現在行っている，3 次元空間格子の 3 次元 PU 格子への単純なマッピングでは，ベクトル長が短くなり過ぎてしまう．しかし CP-PACS では，HXB ネットワークの柔軟性を利用して，現在のプログラムや CP-PACS の運用形態を全く変更すること無しに，この困難を回避することができる．すなわち，HXB ネットワークでは，物理的な 3 次元の PU 格子を論理的に 2 次元格子に組み直しても，隣接通信が無衝突で実現される．そこで，4 次元の時空格子の空間部分のうちの 2 次元を 2 次元 PU 格子にマップして，残りの空間 1 次元＋時間 1 次元の格子の空間部分についてベクトル化すれば，現在行っているシミュレーションと同程度以上のベクトル長を確保できる．

本プロジェクトにより，CP-PACS 用に開発された格子 QCD プログラムは以下のとおりである．

quenched QCD ゲージ配位生成プログラム　quenched QCD におけるゲージ配位生成は heat-bath 法 に従って行い，生成されたゲージ配位間の独立性を高めるために over-relaxation 法 による配位更新を組み合わせた．even-odd アルゴリズムにより，疑似ベクトル化されるループ長は $L_t/2$ である．乱数生成は，並列計算機用の M 系列乱数を用いた．

heat-bath 法，over-relaxation 法，いずれも計算の中心部分は，前節で議論した 3×3 複素行列の行列積 `U1*U2` の計算である．他の類似の計算でも適当なループ アンローリングを行った結果，これらの演算部分では 160 MFLOPS/PU 以上の性能を実ジョブで達成した．heat-bath 法では，その他に exp，log，sin，cos 等の関数をベクトル的に呼び出す部分があり，abs と sqrt 以外はそ

の前後でループ分割が要求される．これら関数周辺のループは，ループ長に比べて演算量が少なく，現状ではそれほど速くない．

full QCD ゲージ配位生成プログラム　我々の full QCD のゲージ配位生成プログラムでは，Hybrid Monte Carlo (HMC) 法を採用した．これは，Wilson クォークでフレーバー数が 2 の場合に厳密な分布を与える方法である．最も重い演算部分はクォーク伝搬関数の計算で，以下に述べる solver と同じである．その他，quenched QCD ゲージ配位生成プログラムと同じタイプの計算が行われる．

近年の格子 QCD の研究から，格子作用として最も簡単な「標準作用」を用いることの限界が明らかになってきた．この問題は，小さめの格子でシミュレーションせざるを得ない full QCD でより深刻である．

よって，full QCD の研究プロジェクトでは,「標準作用」の場合だけでなく，空間的により広がった相互作用を取り入れた「改良された作用」の場合についても計算できるプログラムを開発した．ゲージ部分の作用に関しては，標準の *plaquette* 作用と，長方形の相互作用まで取り入れた *plaquette* + *rectangular* 作用の 2 つを選択できるようになっている．クォーク作用には，標準の Wilson クォーク作用と，Wilson クォークを改良した Clover クォークの 2 つを選択可能である．

ゲージ固定プログラム　ゲージ固定とは $\sum_{n,\mu} \mathrm{Re}[\mathrm{Tr}U(n,\mu)]$ を最大にするように，ゲージ場をゲージ変換する操作である．ここで n は 4 次元時空格子の座標．μ は格子軸の方向で，実ジョブで主に用いた Coulomb ゲージでは 1 から 3 （空間方向のみ），Landau ゲージでは 4 まで（時間方向を含む）の和を取る．QCD の SU(3) 群の場合解析的には解けないので，逐次解法を用いる．我々のプログラムでは， 3 つの SU(2) 部分群に対するゲージ固定を繰り返す方法と美濃英俊（山梨大学工学部）による over-relaxation 法を組み合わせた．

Wilson クォーク solver　与えられたクォーク・ソース B に対してクォークの伝搬関数 $G = D^{-1}(U)\,B$ を計算する．クォークとしては Wilson クォークと，Wilson クォークを改良した Clover クォークの両方から選択可能にしてある．ここでクォーク行列 $D(U)$ は，格子の体積を $V = L_x \times L_y \times L_z \times L_t$ とすると，Wilson 型クォークの場合 $12V \times 12V$ の疎行列で，ゲージ配位に依存している．$D(U)$ の性質から，even-odd 法（red-black 法）により演算量を減らすことができる．

大行列連立 1 次方程式 $G = D^{-1}(U)\,B$ は，逐次近似解法により計算する．その解法には様々なものがあるが，基本演算は前節で議論した QCDMULT で，アセンブラ・コードにより高速に実行される．

我々はまず Minimal Residual (MR) 法（Krylov 次元=1 の共役残差法）による solver を開発した．MR 法の solver では，基本ループ内の QCDMULT 以外の演算についてもほぼ完全にアセンブラ化しており，その結果，前節で示したように極めて速いループの立ち上がりを実現した．

また，$D(U)$ の最小固有値がより小さい領域まで有効な BiCGStab 法の solver も開発した．その他の解法に基づいたプログラムも現在開発・テスト中である．

KS クォーク solver　格子 QCD のクォーク作用には，Wilson あるいはその改善である C lover 作用以外に，Kogut-Susskind（KS）あるいは staggered クォーク作用と呼ばれる種類がある．この作用は現在までのところ実計算には用いていないが，FORTRAN プログラムの性能評価と将来の使用の可能性を考慮し，与えられたソースに対してクォーク伝播関数を求めるプログラムを開発した．計算方法は共役傾斜法である．

KS クォーク作用は Wilson 作用に比べて簡単な構造をしており，FORTRAN によりかなりの性能を引き出すことができる．性能実測値は，時間方向のサイズ $L_t = 256$ の場合で，演算性能 139 MFLOPS/PU，通信を含めて 125 MFLOPS/PU となっている．

ハドロン物理量測定　格子 QCD での最も重要な物理量はハドロンの質量であり，これを求めるにはハドロンの伝搬関数を計算する必要がある．ハドロンの伝搬関数は， 2 つ（メソン），あるいは 3 つ（バリオン）のクォークの伝搬関数を組合わせることによって計算できる．我々は，ハドロンの時間方向への伝搬関数を計算する並列プログラムを開発した．

この場合に時間に関するループを一番外側にしてその中でクォーク伝搬関数を読み込むことにすると，クォーク伝搬関数の大きさは $12 \times 12 \times V_s$ となる．ここで $V_s = L_x \times L_y \times L_z$ は格子の空間体積である．すべてのクォークの種類が異るバリオンを計算するには，このクォーク伝搬関数を 3 本同時にメモリ上に持つ必要があるが，時間 t を固定して計算するようにして同時に必要なクォーク伝搬関数の大きさを減らしたのでそれが可能になった．

我々の quenched QCD プロジェクトでは，重いクォーク 2 種類と軽いクォーク 3 種類の合計 5 種類のクォークを，それぞれ 2 種類のオペレーター（ソース）で計算しているが，メモリ量の制限により，時間を固定してもクォーク伝搬関数 10 本全てを同時にメモリ上に置くことはできない．そこで，異なる軽いクォークの組み合わせに対応するハドロンの計算を断念し，重いクォーク 2 つ，軽いクォーク 1 つの合計 6 本をメモリ上に持つようにして，無駄な I/O を行わない計算を行った．それでもハドロンの計算プログラムは膨大な I/O を必要としており，その性能がこのプログラムの効率を大きく左右しているため，第 21.1.1 節に述べたように，様々の工夫を行なった．

他方，ハドロン物理量の演算部分は，和や積といった単純な計算が主だが，疑似ベクトル化のベクトル長は演算量と比べてやや短めで，実ジョブにおける演算性能は約 100 MFLOPS/PU であった．

21.1.3　実ジョブ実行時間

この節では，前節で紹介したプログラムを使って実行された，格子 QCD の実ジョブにおいて，実際に測定された実行時間について述べる．それぞれのプログラムでは実ジョブ実行時に，主な subroutine の実行時間を FORTRAN の `xclock` 関数 (精度 μsec) を用いて常時監視している．以下にあげるデータは，最近実行されたジョブの平均である．

これらの実ジョブによって得られた物理的成果については，第 22 章で述べる．

quenched QCD　これまでに行われた quenched QCD のシミュレーションを表 31 に示す．クォーク場としては Wilson クォークを用いた．軽いハドロンのスペクトル計算における有限格子サイズ効果を抑えるために空間サイズ $La \approx 3$fm の大きな格子を使い，$m_\pi/m_\rho \approx 0.75$，0.7，0.6，0.5，0.4 に対応するクォーク質量で計算した．$m_\pi/m_\rho \approx 0.4$ という軽いクォークを用いた高統計の計算は世界的にも始めてである．

配位間の独立性を高めるために，これらのシミュレーションでは heat-bath 法による配位更新 1 sweep 毎に 4 sweep の over-relaxation 法による配位更新を行った．このセットを 1 iteration と呼び，$N_{\rm sepr}$ iteration 間隔で配位を生成した．各配位毎に，上記 5 種類のクォークを計算し，それを組み合わせて様々なハドロンの質量や崩壊係数などを計算した．I/O 等も含めた， 1 配位当りの全計算時間を表 31 に示す．また，表 32 にプログラム各部の実行時間と，その全実行時間に対する割合を示す．

格子サイズ	La(fm)	a(fm)	配位数	$N_{\rm sepr}$	PU 数	配位当りの計算時間 (時間)
$32^3 \times 56$	3.2	0.100	800	200	256	3.0
$40^3 \times 70$	3.0	0.076	600	400	512	4.8
$48^3 \times 84$	3.0	0.063	420	1000	1024	6.8
$64^3 \times 112$	3.0	0.047	91	2000	2048	15.6

表 31: CP-PACS における quenched QCD シミュレーション．$64^3 \times 112$ 格子のシミュレーションは進行中．

格子	$32^3 \times 56$	$40^3 \times 70$	$48^3 \times 84$	$64^3 \times 112$
配位生成	0.28 (10%)	0.65 (14%)	1.64 (24%)	4.5 (29%)
heat-bath				[1.5 (10%)]
over-relaxation				[2.8 (18%)]
ゲージ固定	0.13 (5%)	0.22 (5%)	0.33 (5%)	1.8 (12%)
クォーク solver	1.57 (52%)	2.72 (57%)	3.13 (46%)	6.6 (42%)
ハドロン物理量測定	1.01 (33%)	1.18 (24%)	1.67 (24%)	2.6 (17%)
total	2.99	4.76	6.78	15.6
disk I/O	(17%)	(16%)	(19%)	(9%)

表 32: quenched QCD ジョブにおける各部分の 1 配位あたりの実行時間 (時間) と，全実行時間に対する割合 (%)．

quenched QCD における配位生成は速度は通常リンク変数 $U(n,\mu)$ ひとつを更新するのに必要な時間 “link-update time” で表現される．以下では，2048 PU を用いた $64^3 \times 112$ 格子における実ジョブで測定された性能をまとめる．CP-PACS における heat-bath 部の実測値は 0.0232μsec である．heat-bath 更新部分は確率操作や基本関数呼び出しを含むので，この数値を単純に GFLOPS に翻訳することはできないが，格子 QCD 研究者の間では 5700 FLOP/link-update という経験的な数値が広く使われている [17]．それを用いると，CP-PACS における heat-bath 部の速度は 246 GFLOPS である．比較として，heat-bath 配位更新でこれまでに報告されている最高性能は，科学技術庁航空宇宙技術研究所の数値風洞 (Numerical Wind Tunnel) の 128 ノードを使って測定された，0.0317μsec，180 GFLOPS である [17]．出版はされていないが，航空宇宙技術研究所のホームページには 160 ノードを使った 0.0264μsec，216 GFLOPS という結果も報告されている [18]．

CP-PACS における over-relaxation 法による配位更新の link-update time は 0.0109μsec である．over-relaxation 法は確率操作を含まないので，演算数を正確に数えることができ，我々の採用したアルゴリズムでは 3050 FLOP/link である．従って CP-PACS における over-relaxation 部の速度は 280 GFLOPS である．

表 33 に示すように，クォーク solver の演算部分の性能は格子サイズにほとんど依らない．これは，前節で議論したように，Wilson クォークの場合の MR 法によるクォーク solver では，演算のほとんどを占めるループがほぼ完全にアセンブラ化され，ループの立ち上がりが極めて速いからである．2048 PU 全体における solver の性能は，325 GFLOPS である．

格子	$32^3 \times 56$	$40^3 \times 70$	$48^3 \times 84$	$64^3 \times 112$
演算 (MFLOPS,%)	191 (79%)	191 (81%)	191 (80%)	192 (82%)
隣接 PU 間通信 (MB/s,%)	205 (18%)	214 (16%)	221 (17%)	227 (15%)
global sum (%)	(3%)	(3%)	(3%)	(3%)
total (MFLOPS)	152	154	152	157

表 33: Wilson クォーク solver プログラム各部の PU あたりの性能と，実行時間に占める割合．

m_π/m_ρ	0.84	0.69	0.62	0.54	0.41
配位生成	34 (84%)	70 (89%)	165 (95%)	413 (98%)	1110 (99%)
ゲージ固定	1.8 (4%)	2.3 (3%)	1.7 (1%)	1.5 (.4%)	1.9 (.2%)
クォーク solver	3.0 (7%)	4.5 (6%)	5.0 (3%)	3.5 (.8%)	6.2 (.6%)
ハドロン物理量測定	1.8 (4%)	1.7 (2%)	1.8 (1%)	1.6 (.4%)	1.7 (.2%)
total	39	79	174	420	1120

表 34: 典型的 full QCD ジョブにおける各部分の 1 配位あたりの実行時間 (分) と，全実行時間に対する割合 (%)．このジョブでは，ゲージ部分もクォーク部分も改良された作用を用い，格子サイズ $16^3 \times 32$ を 256PU を使ってシミュレーションした．配位生成部分のクォーク逆行列計算は BiCGStab 法により，クォーク solver 部分では $m_\pi/m_\rho > 0.6$ では MR 法で，$m_\pi/m_\rho < 0.6$ では BiCGStab 法で計算した．

full QCD　full QCD は現在予備的研究として，様々な「改良された作用」の比較研究を行っている．クォークとしては Wilson 型のものだけを調べた．この研究では，$8^3 \times 16$ と $12^3 \times 32$ 格子を 64 PU $(4 \times 4 \times 4)$ パーティションで，また $16^3 \times 32$ 格子を 256 PU $(4 \times 8 \times 8)$ パーティションでシミュレーションした．

ジョブの実行時間はゲージ結合定数やクォーク質量などのパラメータの値と，格子のサイズに大きく依存する．典型的な例として，表 34 に，最近行われた $16^3 \times 32$ 格子のシミュレーションにおける 1 配位あたりの実行時間をまとめる．この表に見られるように，full QCD 計算では配位生成が最も大きな計算時間を占めている．

full QCD では，quenched QCD より格子のサイズが小さくならざるを得ない．特に今回の比較研究では，quenched QCD の場合と比較するとかなり小さな格子まで含まれるので，ベクトルループ長がかなり小さくなる．そのため，アセンブラ化されていない部分の疑似ベクトル ループの立ち上がりが問題になる．上記の格子では，重要な演算のループ性能は 80-110 MFLOPS/PU であった．また 1 回の PU 間通信で送られるデータ量も小さくなり，通信性能も $12^3 \times 32$ 格子で 150-160 MB/sec 程度であった．より大きな格子で予定されている本格的な full QCD 計算では，より quenched QCD に近い性能が得られると期待される．

21.1.4　評価

格子 QCD のプログラム開発を通じて得た，CP-PACS の実機性能の評価は次のようにまとめられる．

演算性能については，21.1.1 節の図 48（b）に示したように，格子 QCD プログラムの中心部分である QCDMULT でアセンブラプログラムにより，64 % の効率を引き出すことができた．設計段階における仮想ベンチマークの予想性能を若干下回るとはいえ，これは PVP-SW 方式が RISC プロセッサの能力を引き出す上で極めて有効であることを実証するものである．また，通常の FORTRAN プログラムでも，最低で 30 %，多くのプログラムでは 40 % から 50 % の効率を達成している．これも PVP-SW 方式の有効性の査証を与えると同時に，コンパイラの優秀性を示している．

将来に向けて，演算性能の一層の強化を実現する上で，重要と思われるのは以下の点である．(i) store ネックの改善．現在レジスタから主記憶への poststore に実効的に 2 マシンサイクル掛っている．短縮が望まれる．(ii) バンクコンフリクトの問題．16 バンク構成の主記憶により本格的な対応がなされているものの，この問題により 10 % 程度の性能劣化を生じる場合が多い．プログラムの工夫で対応するには限界がある．(iii) レジスタ・ウインドウのサイズ．現在のサイズ 32 レジスタは QCD によく現れるパターンの計算に対してはやや小さい．このためにループ分割が必要となり，それに付随するウィンドウ・スィッチの回数や load/store の増大が実効性能を下げる要因になっている．(iv) multiply-add 命令生成の強化，数学関数や条件分岐を含んだループの疑似ベクトル化の強化など，コンパイラの性能向上．また，インデックス・ベクトルを用いる場合への対応のために，cache prefetch 命令生成の強化．特に数学関数関係では，abs と sqrt 以外は関数をベクトル的に呼び出す部分の前後で疑似ベクトル化の為にループ分割が要求されるが，それらの分割されたループでは，ループ長に比べて演算量が少ない場合が多く，ループの立ち上がりが遅くなる．演算の順番の調整による対応には限界があり，改善すべき点と思われる．

通信性能に関しては，レイテンシ及びスループットとも，実プログラムにおいてほぼ設計値を実現している．また，リモート DMA のブロック・ストライド転送は，格子 QCD プログラムにおいて極めて重要かつ有効であり，その性能も満足すべきものであった．これにより，前節で示されたように，格子 QCD の問題では，全体の計算時間における通信時間の割合を最大で 20 %以下，多くの場合 10 % 程度の低い値に押さえることができた．

しかし，反省すべき点や改良が望まれる点も幾つか残った．(i) ブロック・ストライド転送において，ストライド・データのエレメント・サイズが 1020 バイト以下，ストライド幅が 65532 バイト以下に制限されているが，QCD の問題に現れるより多くの通信パターンに柔軟に対応するためには，少なくとも 2 桁程度以上大きな数まで許すようことが望まれる．(ii) ブロック・ストライドをエレメントとするブロック・ストライド等，より複雑なパターンへの対応ができるような工夫が必要である．

最後に，分散ディスクとの I/O 性能は，QCD の問題では全体としてやや不足している．21.1.1 節で議論したように，ファイルをディスク上で常に物理的に連続に配置すること，C の I/O 関数の使用等，様々なプログラミング上の工夫により設計性能値を実現しているものの，全計算時間の 10—20 %を I/O に費しているのが現状であり，IOU 及び分散ディスクの性能自身の改善が必要と思われる．

21.2 宇宙物理学における輻射輸送問題

本節では始めに，宇宙物理において CP-PACS を利用する応用プログラムの中でも核となる部分の内容を具体的に紹介する．その計算法は，CP-PACS に良く適合するように私達が新たに考案したものである．そのあと，私達のプログラムの基本計算部分の性能をまとめる．

21.2.1 計算内容

宇宙物理における CP-PACS の応用：多次元輻射輸送問題　利用の立場から見た CP-PACS の主要な特徴は，巨大な PU 資源である．この特徴を十分に生かせる宇宙物理学の重要な問題の一つに，多次元空間内の輻射輸送問題がある．私達は，輻射輸送に関連する問題に焦点を絞って宇宙物理における CP-PACS の応用を進めようと考えている．

輻射は一般に，輻射強度と呼ばれる 6 次元位相空間 (3 次元配位空間 × 3 次元運動量空間) 上のスカラー関数で表現され，輻射輸送は輻射強度の 6 次元空間内の輸送方程式 (輻射輸送方程式) により記述される．輻射輸送が実際の天体現象に関与する仕方は多岐にわたるが，問題を解くに際しては多くの場合，次の定常輻射輸送方程式を解くことが基本となる．

$$\boldsymbol{n} \cdot \nabla I(\boldsymbol{x}, \boldsymbol{n}, \nu) = -\chi(\boldsymbol{x}, \boldsymbol{n}, \nu) \cdot I + \eta(\boldsymbol{x}, \boldsymbol{n}, \nu). \tag{16}$$

ここで，$I(\boldsymbol{x},\boldsymbol{n},\ \nu)$ は輻射強度，$\boldsymbol{x}$ は位置ベクトル (3 次元)，$\boldsymbol{n}$ は輻射の伝播方向を示す単位ベクトル (2 次元)，ν は輻射の振動数 (1 次元)，$\chi(\boldsymbol{x},\boldsymbol{n},\ \nu)$ は単位体積当たりの吸収係数，$\eta(\boldsymbol{x},\boldsymbol{n},\ \nu)$ は単位体積当たりの放射率，である．さらに数値計算においては χ と η を既知関数とみなし，それらを求めること (一般に χ と η は I に依存する) は輻射輸送とは別の問題として切り離して考える．(ほとんどの場合，χ と η を求める計算は式 (16) を解いて I を求める計算に比べて桁違いに少ない計算量である.) 結局，多次元輻射輸送問題の数値計算において核となる計算は，与えられた χ と η に対して式 (16) を積分して，各位置 ($\boldsymbol{x}$) における各方向 ($\boldsymbol{n}$) と振動数 (ν) に対応する輻射強度 I を求めることとなる．

輻射輸送方程式の基本的解法　輻射輸送方程式 (16) において χ と η が既知である場合，方向 ($\boldsymbol{n}$) と振動数 (ν) が異なる輻射強度 I 同士は互いに独立となるので，これらは並列に計算することが出来る．一方空間 ($\boldsymbol{x}$) に関しては，輻射の伝播する下流側は上流側に依存しているので，必ず上流から下流に向かって計算を進めなければならない．実際の計算にあたっては輻射輸送方程式のこのような特性を踏まえつつ，CP-PACS において効率の良い解法を選択する必要がある．

格子法に基づいて多次元定常輻射輸送方程式 (16) を解く基本的な手法は，Short Characteristics 法と呼ばれる方法である．CP-PACS を利用する場合にも，この方法を用いる．ここでその考え方を，簡単のために 2 次元平面内の輻射輸送の場合を例にして説明しておく (図 50 参照)．実際の 3 次元空間内の輻射輸送を計算する場合には，ここでの考え方を 3 次元に拡張したものを用いる．

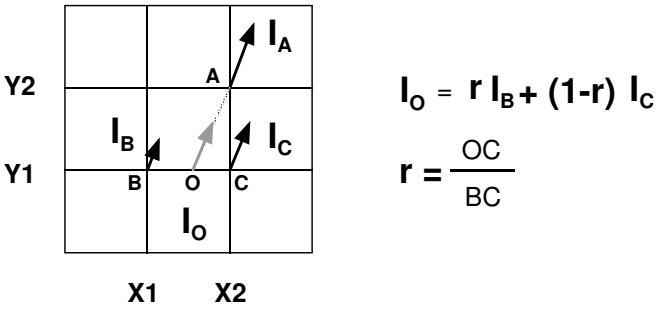


図 50: Short Characteristics 法

式 (16) を解いて点 A における方向 $\boldsymbol{n}$，振動数 ν の輻射強度 I_A を求めるためには，上流の点 O における輻射強度 I_O と，点 A と O における χ_A，η_A，χ_O，η_O を知っている必要がある．このうち χ_A，η_A は既知であるが，点 O は格子上の点ではないので I_O，χ_O，η_O は未知である．

そこで Short Characteristics 法では，点 O でのこれらの量を近傍の格子点 B，C での値から補間により求める．そうやって点 O での I_O，χ_O，η_O が求められれば，線分 OA 上で微分方程式 (16) を積分することにより，点 A での輻射強度 I_A が得られる．

このように，ある点での輻射強度を求めるためにはその点よりも上流側の輻射強度が先にわかっている必要がある．すなわち，空間の格子点の計算順序には注意しなければならない．図 50 の例では，左下から右上へという順序で計算を進める必要がある．

輻射輸送方程式の CP-PACS に適した解法の探求：NWF 法の開発　分散メモリ型並列計算機である CP-PACS で空間的に大きな領域の問題を扱おうとする場合，そのローカルメモリの大きさの制約もあって，配位空間 $(\boldsymbol{x})$ を小領域に分割して各 PU に割り当てるのが並列化のための自然な分割法となる．このような分割法のもとで Short Characteristics 法を用いて輻射輸送方程式を解く場合，一定の方向と振動数に対しては，先の空間格子点の計算順序に対する制約条件と同様の条件が PU 間の計算順序に対して課せられることになる．

図 51 に，2 次元空間の場合の空間分割と各 PU の計算順序の例を示す．この例の場合には，まず PU0 で計算し，次に PU1 と PU2 で計算し，最後に PU3 で計算するという順序で計算を進める．各ステップの間では PU 間通信が行われ，PU 間境界値が上流側 PU から下流側 PU に転送される．各 PU 内では，上流側格子点から下流側格子点に向かって計算を進める．

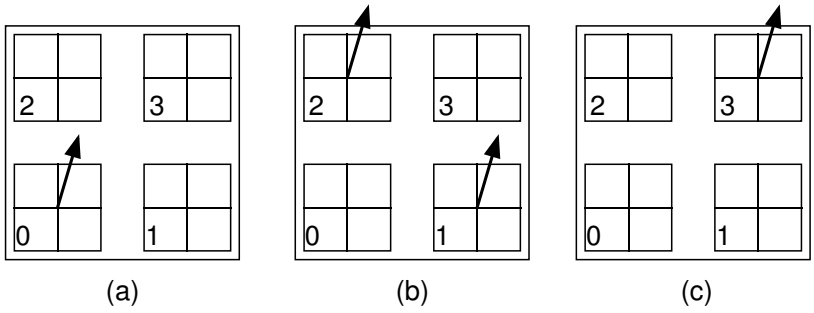


図 51: 空間分割と PU の計算順序：(a),(b),(c) の順に計算を進める．このままでは並列度が非常に低い．

ところがこのままでは，並列化の効率が非常に悪い．(図 51 の (a),(c) の段階では 1 つの PU しか働いておらず，(b) の段階でも 2 つしか働いていない.) そこで並列度を向上させるために，輻射輸送方程式が方向と振動数に対して独立であることを利用する．具体的には，次のように考える．

まず，方向に対する独立性を利用する考え方．ここでの数値計算では，方向 $\boldsymbol{n}$ と振動数 ν も離散化して扱う．それらの方向格子点のうち図 51 では，”右上”方面に向かって伝播するある方向を例にとった．しかし，”右上”方面に伝播する光線はこの方向だけではない．”右上”方面に向かう方向格子点は多数存在する．そこで，そのような”右上”方面に向かう方向の集合に属する方向格子点を，各 PU 内で順に全て計算することにする．方向に関しては独立であるから，各 PU で計算する方向が異なっていてもよいことを利用して並列度を向上させるのである．ただし，ひとつの方向については PU 間の計算順序規則は守られている．このときの具体的な状況を図 52 に示す．

まず最初に，PU0 が方向 $\boldsymbol{n_1}$ の計算をする (図 52(a))．PU0 内の全ての格子点で $\boldsymbol{n_1}$ 方向の計算が終了すると，PU 間境界値が PU1 と PU2 に転送される．そして次に，PU0 では $\boldsymbol{n_2}$ の計算が始まり，PU1 と PU2 が $\boldsymbol{n_1}$ と $\boldsymbol{n_2}$ の計算を進める ((b) の段階)．さらに (c) の段階に進むと，PU0 は $\boldsymbol{n_3}$ を，PU1 と PU2 は $\boldsymbol{n_2}$ を，そして PU3 は $\boldsymbol{n_1}$ を計算する．このようにして計算を進

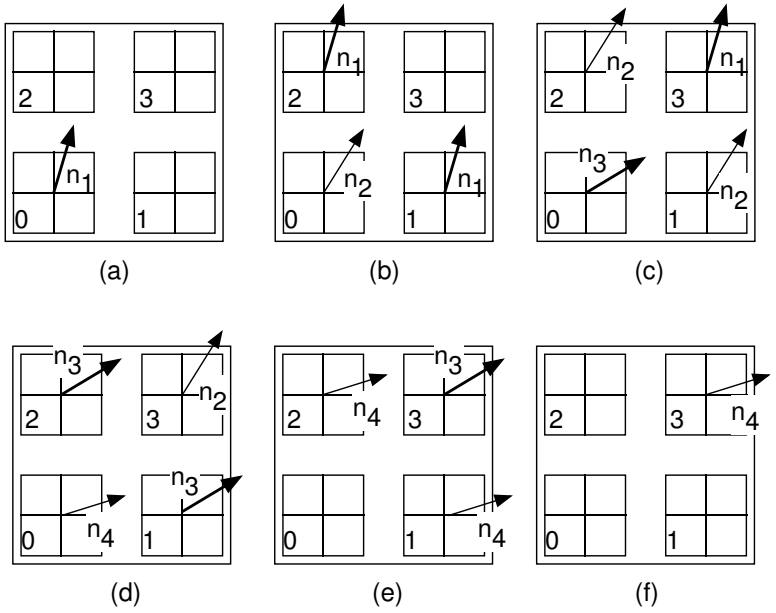


図 52: 方向独立性を利用した並列化率の向上：(a),(b),⋯,(f) の順に計算を進める．(c),(d) の段階では，完全な並列化が実現されている．N_n が十分大きい場合には (c),(d) の段階の継続時間が (a),(b) および (e),(f) の段階の継続時間に比べて十分長くなるので，全体としては高い実効的並列化率が得られる．

めると，(a) の段階では 1 つの PU しか働いていないが，(b) では 3 つの PU が，そして (c) では全 PU が稼働することになる．計算が進むと，やがて”右上”方面に向かう方向格子点がなくなる．計算すべき方向格子点の計算が終了した PU は，他の PU での計算が終了するまで待つ ((e),(f) の段階)．全 PU での計算が終了すると，”右上”方面の方向格子点に対する全空間での計算が完了し，つぎに別の方面 (”左上”など) の計算にとりかかることになる．

図 52 の例では，”右上”方面に向かう方向格子の数が 4 個の場合を示しているが，この数が十分に大きい場合には，(c),(d) の段階は十分に長く続く．すなわち，(a),(b) や (e),(f) の段階での並列度の低下の影響はほとんど無視することが出来る．実際に 3 次元空間内の輻射輸送方程式を，$n_x n_y n_z$ 個の PU ($n_x \times n_y \times n_z$ 構成) を使って解く場合の並列化率 P (全計算時間にわたる平均稼働 PU 数の全 PU 数に対する割合) は，次のように評価できる (PU 間通信やバリア同期待ち等の影響は無視する)：

$$P = \frac{N_n}{N_n + (n_x + n_y + n_z - 3)}. \tag{17}$$

ここで N_n はある方面に向かう方向格子の数 (たとえば”右前上”方面に向かう方向格子の数) である．実際の応用計算では，全方向格子数は 64^2 ないし 128^2 程度を用いる．3 次元の場合には方面の数は 8 であるので，N_n は 500 ないし 2000 程度の値となる．したがってこの計算法では実用計算に対して，95% 程度以上の並列化率が得られることがわかる．

次に振動数の独立性を利用する並列化であるが，これは方向の並列化との組み合わせ方で次の 2 種類の方法が考えられる．

(A) PU 毎に異なる方向を計算しながら，各 PU 内では全振動数を計算する．全振動数格子の計算終了後，PU 間通信を行う．

(B) 全 PU で同じ方向を計算しながら，PU 毎には異なる振動数を計算する．1 つの振動数の計算が終わるたびに，PU 間通信を行う．

これらのどちらがより有効な方法であるかは，振動数格子の数と方向格子の数に依る．実際の応用では，振動数格子数は多くの場合に 10 以下，最も多い場合でも 100 程度が想定されている．このような小さな数の場合には，(A) の方法の方が効率が良い．この場合の並列化率の評価は，式 (17) と全く等しくなる．

ここに記した計算法では，ある一定の方向・振動数を計算している PU (node) 群が node 空間内を平面波状に伝播していく．そして，異なる方向・振動数格子に対応する複数の平面波で node 空間全体を覆い尽くすことにより，並列度が高められているのである．そこで私達はこの計算法を，Node Wave Front (NWF) 法と呼んでいる．この方法は，多次元輻射輸送方程式 (16) を Short Characteristics 法を用いて CP-PACS で効率よく解くために，私達が新たに考案した方法である．一般に NWF 法は，ひとつの自由度に関しては空間依存性があるものの他の多くの自由度とは独立であるような量を，CP-PACS のような並列計算機で計算する際に有効な方法であると思われる．

21.2.2 計算速度

21.2.3 1 PU 内での計算の実際と実効速度

実際に Short Characteristics 法によってある一定の方向・振動数の輻射強度を 1PU 内の全格子点上で求める演算の核心部は，FORTRAN を用いると (原理的に) 次のように表される:

```
         do 1000 iphi=1,Nphi
          do 1010 itheta=1,Ntheta
   c
 1:         do iz=1,NZ
   c
 2:          do iy=1,NY
 3:           do ix=1,NX
 4:            Eta0 =  Eta(ix,  iy,  iz-1)     *C1(itheta,iphi)
       &           + Eta(ix,  iy-1,iz-1)     *C2(itheta,iphi)
       &           + Eta(ix-1,iy,  iz-1)     *C3(itheta,iphi)
       &           + Eta(ix-1,iy-1,iz-1)     *C4(itheta,iphi)
 5:            Chi0 =  log(Chi(ix,  iy,  iz-1))*C1(itheta,iphi)
       &           + log(Chi(ix,  iy-1,iz-1))*C2(itheta,iphi)
       &           + log(Chi(ix-1,iy,  iz-1))*C3(itheta,iphi)
       &           + log(Chi(ix-1,iy-1,iz-1))*C4(itheta,iphi)
 6:            Chi0 = exp(Chi0)
 7:            ratio= Chi(ix,iy,iz)/Chi0
 8:            delta= 1.0d0/(abs(log(ratio)) + eps)
 9:            tau  = L*ChiA*(abs(ratio-1.0d0)+eps)*delta
10:            I_0  =  I(ix,  iy,  iz-1)       *C1(itheta,iphi)
       &           + I(ix,  iy-1,iz-1)       *C2(itheta,iphi)
       &           + I(ix-1,iy,  iz-1)       *C3(itheta,iphi)
       &           + I(ix-1,iy-1,iz-1)       *C4(itheta,iphi)
11:            f1  = exp(-tau)
12:            f2  = (1.0d0-f1)/tau - f1
13:            f3  = 1.0d0 - (1.0d0-f1)/tau
14:            I(ix,iy,iz) = I_0*f1 + f2*Eta0 + f3*Eta(ix,iy,iz)
              end do
15:           end do
   c
16:         end do
   c
```

```
1010  continue
1000 continue
```

ここでは，一定の振動数で，x 軸の正，y 軸の正，z 軸 の正，の方面に向かう方向格子に対応する輻射強度を求める演算を例として示した．さらにそのような方向の中でも特に，主に z 軸の方向に向かっている方向格子に限って示してある．(同じように x 軸の正，y 軸の正，z 軸の正の方面に向かう方向格子の中でも他に，主に x 軸方向に向いているものと，主に y 軸方向に向いているものとがある．結局 3 次元の場合，方向格子は 8 つの方面 (x の正負，y の正負，z の正負の組み合わせ) の中でさらに 3 つずつに細分化され，総計 24 の方向の族に分類される.) 別の族に属する方向の輻射強度に対する計算では，空間積分に対応する行 1，2，3 の `do` ループの順序と上限・下限が入れ替わる (全部で 24 通り)．1 つの PU 内では，空間格子の計算は単純に x 方向，y 方向，z 方向に沿った順で行い，wave front 法は用いない．さらに，複数の振動数を計算する問題では振動数格子に対応するループが加わる．

各演算の意味は，次の通りである: 行 4，5，10 では，近傍の格子点の値から補間により，点 O (図 50 参照) での η_O，χ_O，I_O を計算している．係数 `C1`，`C2`，`C3`，`C4` は，方向毎に決まる一定の係数である．吸収係数 χ (`Chi`) に対して対数をとってから補間を行っているのは，多様な天体現象に対応するための工夫のひとつである．行 9 で，線分 OA 間の光学的厚さ τ (`tau`) を計算する．以上で求めた量を用いると，輻射輸送方程式 (16) を線分 OA 間で積分した解 I_A は，行 14 の演算によって求めることが出来る．行 14 は式 (16) を積分表示し，さらにそれを離散化した表現である．

`do` ループの順序	計算速度 [MFLOPS] (PV)			
	ループの長さ			
	4×4	8×8	16×16	32×32
`do` Z 　`do` Y	71.9 ± 0.1	73.5 ± 1.1	64.0 ± 1.5	63.7 ± 1.4
`do` X 　`do` Z	73.2 ± 0.8	76.9 ± 1.7	65.0 ± 1.2	64.3 ± 0.5
`do` Y 　`do` X	72.6 ± 1.1	77.4 ± 1.7	64.7 ± 1.2	65.8 ± 0.7

コンパイラ: V02-03-/A
コンパイルオプション: -W0,'PVEC(DIAG(1),VOPTION(1)),
　　LANGLVL(PRECEXP(4),H8000),OPT(O(4)),COPTION(1)' -i,E,U,L,P

表 35: 基本 2 重 do ループの計算速度 (1). 疑似ベクトルオプションを指定した場合.

実際の計算における行 2 から行 15 までの 2 重 `do` ループ の計算速度を，表 35 と表 36 にまとめる．この `do` ループの長さは，1 PU 内の空間格子点数`NX`，`NY`，`NZ` に依る．CP-PACS 上での実用計算では，これらは 8 ないし 16 程度の値となる．なお，実際の計算においては精度を維持するため，行 9 の `tau` の値に応じて行 11，12，13 での `f1`，`f2`，`f3` の評価式を換えるように `if` 文が挿入されている．すなわち，最内側 `do` ループ内に `if` 文がある．表 35, 36 中の数値は，`if` 文の分岐が様々になるようないくつかのパラメータで測定した数値を平均したものである．

表 35，36 によれば，ループのベクトル長が 64 $(= 8 \times 8)$ 以下の場合には，スカラー計算の速度と，疑似ベクトル機能を指定したコンパイラが生成したコードの速度はほぼ同じである．これ

do ループの順序	計算速度 [MFLOPS] (Scalar)			
	ループの長さ			
	4×4	8×8	16×16	32×32
do Z do Y	71.2 ± 1.4	74.0 ± 1.8	70.2 ± 0.8	66.0 ± 0.8
do X do Z	73.0 ± 0.7	74.6 ± 1.6	71.8 ± 1.5	71.7 ± 1.4
do Y do X	72.5 ± 1.6	75.3 ± 1.4	70.9 ± 1.4	72.4 ± 1.7

コンパイラ: V02-03-/A
コンパイルオプション: -W0,'LANGLVL(PRECEXP(4),H8000),OPT(O(4)),
COPTION(1)' -i,E,U,L,P

表 36: 基本 2 重 do ループの計算速度 (2). 疑似ベクトルオプションを指定しない場合.

は，コンパイラの"臨界ベクトル長"が 64 と 256 の間に設定されているためであろうと思われる．すなわちベクトル長が 64 以下の場合には，コンパイラの指定の如何にかかわらず，いずれの場合もスカラー用のコードが生成されているからであろう．一方，ベクトル長が 256，1024 の場合では，スカラー計算の方が速い．これは，ベクトル長がまだ十分には長くないため，疑似ベクトル機能の恩恵を受けていないためであると思われる．

全 PU での計算時間　次に，NWF (Node Wave Front) 法を用いて実際に 3 次元空間内の輻射輸送方程式を CP-PACS で解いた場合の計算時間を，図 53 に示す．計測に使用した PU 数は，2，8，64，512，2048 の 5 種類である．問題規模は，空間格子数と方向格子数をそれぞれ 16，32，64，128，256 の 5 段階に変化させた．振動数の格子数は 1 に固定した．計測した時間は，1 つの振動数に対して全空間で全方向の輻射強度を計算し終える時間である．これには，PU 間通信やバリア同期待ち等の時間も全て含まれる．

同じ問題規模で比較した場合，計算時間は PU 数にほぼ反比例して減少している．これは，NWF 法が有効に働いていることを示している．PU 数が大きくなった場合に若干効率が悪くなる傾向があるが，これは PU 間通信やバリア同期の回数が増えることと，式 (17) にあるように並列度がわずかに低下するためであろうと思われる．一方，PU 数を同じにして比較した場合，問題規模が $32(= 2^5)$ 倍になっても計算時間はそれほど伸びてはいない．実際には最大で 32 倍であり，多くは 20 倍前後になっている．

なお，図 53 に示した計測結果は 1997 年 3 月時点での値である．この数値については，今後まだ多少向上するものと思われる．各 PU 内での計算の最適化，PU 間通信の最適化，バリア同期の取り方の最適化，等々の改善の余地がまだ残っているからである．

21.2.4　まとめ

宇宙物理学における CP-PACS の応用として，輻射輸送に焦点を絞って応用プログラムの開発を進めている．これまでのところ，その核となる部分の開発がほぼ完了した．それを利用した実

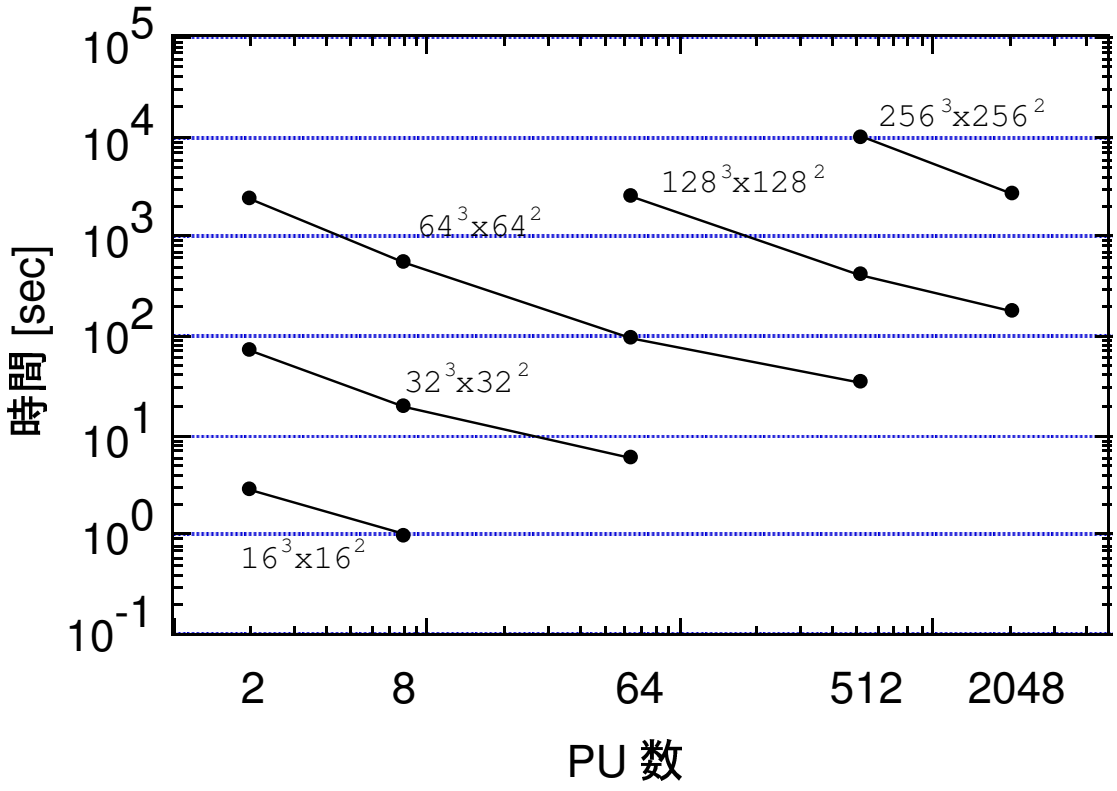


図 53: PU 数に対する計算時間の変化．問題規模は，(空間格子数 × 方向格子数) = $(16^3 \times 16^2)$, $(32^3 \times 32^2)$, $(64^3 \times 64^2)$, $(128^3 \times 128^2)$, $(256^3 \times 256^2)$ の 5 段階を用いて調べた．

際の宇宙物理問題への応用も進められている．

開発したプログラムは，当初想定していたアルゴリズムとは異なり，CP-PACS においてより効率的に実行できるように新たに考案したアルゴリズムによる．テスト計算により，この新しい計算法 (NWF 法) が実際に有効であることも確認できた．

プログラムは基本的に完成しているが，今後は各部の最適化を進める予定である．実際の多くの応用計算では，輻射輸送計算の計算時間が他の部分に比べて桁違いに大きいので，この部分の高速化は，実用計算の計算時間の短縮に極めて有効に働くからである．

21.3 物性物理学における応用プログラムの開発と性能評価

21.3.1 計算内容

物性物理学では，量子モンテカルロ法・厳密対角化法等による量子スピン系や強相関電子系の研究，第一原理計算による物質の構造及びその電子状態の決定等が，大規模計算を必要とする分野である．CP-PACS においては，この内第一原理計算のためのプログラム開発と性能評価が行なわれた [19]．ここでは，その内容と結果をまとめる．

第一原理法は，物質の構造と電子状態をできる限り物理学の第一原理に基づいて定めようとするものである．実際上は，電子を量子力学的に波動関数で記述し，一方原子核は古典的にその空間座標で表して，これらの自由度から全エネルギーが計算される．この時，電子波動関数は，与えられた原子配置に対するエネルギーが極小値をとるように求められる．そして，そのような電子波動関数を用いて，原子核に働く力が計算され，この力を用いてニュートン方程式を解けば分子動力学が実現され，力がゼロになるように原子を移動させることにより物質の安定構造が求まる．計算コストの多くを占める部分は，共役傾斜法により，与えられた原子配置に対して電子波動関数を自由度として最適化するプロセスである．その中でも，波動関数の運動量空間表示と座標空間表示相互の変換のための高速フーリエ変換，並列分割軸を変更するための転置転送，複数のプロセッサーに分割配置されたベクトルの内積，総和 (global sum) などが計算時間の大部分を占める．

21.3.2 性能評価

表 37 に，CP-PACS 向けに開発された高速フーリエ変換プログラムの単体性能と，日立によるライブラリを用いた場合の性能の比較を示す．

格子サイズ	改善前	改善後	hzft6（日立ライブラリ）
128^3	7.5 sec	3.8 sec	3.9 sec
64^3	0.68 sec	0.4 sec	0.45 sec
32^3	0.092 sec	0.07 sec	0.10 sec

表 37: 高速フーリエ変換及び逆変換一回に要する単体 CPU 時間（秒）[19]

サイズ 128^3 に対して，VPP500 単体 (1.6GFLOPS．CP-PACS の PU に比べ約 5 倍) での計算時間は 0.5 sec，又 DEC ワークステーション（400MHrz）では 12 sec である．

また，Si 16 個と水素 1 個からなるユニットセルに対するプログラム全体の PU 単体による計算時間は，プログラムの種々の改善により，40000 秒から 14000 秒に短縮された．この計算は VPP500 上で 1800 秒，DEC ワークステーション上で 17500 秒を要する．

これらの比較に用いた計算機の単体性能と CP-PACS の PU の性能比を考慮すれば，以上の結果は PVP-SW によるベクトル化機構が有効に働いていることを示す．

次に，並列化効率についてであるが，第一原理計算法の並列効率を決める重要な因子として global sum 性能があげられる．本プログラムでは，開発当初から 2 進木アルゴリズムを採用していたが，十分な性能が得られないという問題があった．しかし，この問題は，転送ステージの制御にバリア同期を用いていたことと，データ転送関数として標準的な lcwrite 関数を用いていたことが原因であることが判明した．これらにかわり，転送ステージの制御には shuttle アルゴリズムを用い，転送関数としてもっとも立ち上がり時間が短い，TCW 再利用型関数を利用することにより，ほぼ理論ピーク値に近い性能を得ることに成功している．(以下のグラフ参照)

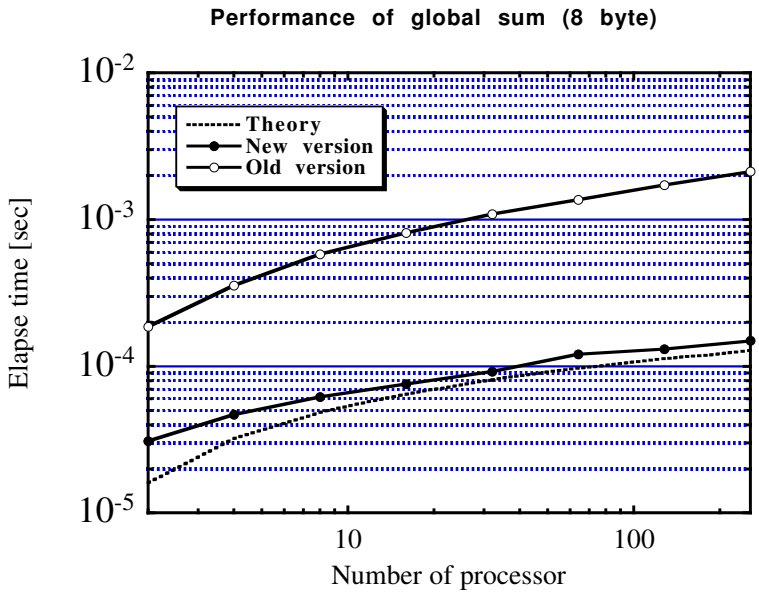


図 54: 8 byte data の global sum 一回に要するシステム経過時間（秒）

また，大きなデータ (例えば $8 * 128^3$ byte) を数百 node にわたって global sum をとる場合には，転置転送アルゴリズムの方が 2 進木アルゴリズムより数倍高速であることが理論的に予測されるが，開発当初は 2 進木アルゴリズムのみを利用していた．現在は電子密度の総和をとる部分にこのアルゴリズムを採用している．また，その実測性能は理論予測と良い一致を示している．

この様に並列最適化した結果，プログラム全体としてもまずまずの並列効率を実現した．現在採用している並列アルゴリズムでは，最大並列加速率が Nband/2 (Nband: 単位胞中の電子数の半分) であり，これに対し，Si 64 原子 (Nband=128) で 128 PU を利用した場合に約 45 倍 (理論最大値 64 倍) というの加速率を得た．

現在残された課題としては，並列分割方法を，実行時に自由に変更できるようなプログラムにすることである．第一原理計算においては，計算対象とするシステムによって，最適な並列化の仕方が変わってくる．このような問題を，ユーザ自身が Remote DMA を用いてプログラム自身を変更することにより対応するのは非常に困難であり，並列効率が十分に引き出せて，なおかつ移植性の高い並列言語の開発が望まれる．

22 CP-PACS による計算物理学の成果

22.1 素粒子物理学

22.1.1 Quenched QCD におけるハドロン質量スペクトル計算

ハドロンの質量スペクトルを第一原理である QCD から導く事は，格子 QCD の試金石と考えられている．第 19.1 章で述べたように，近似無し QCD（full QCD） のシュミレーションには膨大な計算時間が必要となるので，今日まで，動的クォーク効果を無視する quench 近似 の範囲内で精度の高い結果を求める事に，多くの努力が払われてきた．quench 近似は，種々の物理的な考察によって，多くの物理量に対して 10% 程度の誤差で正しい結果を与えると考えられている．quench 近似内で，数% 以下の精度でスペクトラムを計算し，結果が 10% 程度の範囲で実験値を再現することを示し，さらに，この近似による実験値とのズレを明らかにする事が，quenched QCD 計算の目標である．これを達成する為，我々は Wilson クォーク作用を用いた大規模な数値シュミレーションを行った．[20, 21]

今日までの研究で，quench 近似を用いても，数% の精度でスペクトラムを計算することは予想以上に困難である事がわかってきている．計算はモンテカルロ法によるので，統計誤差を小さくする為充分なサンプル数が必要になるのは言うまでもないが，信頼できる結果を得るためには，格子 QCD 計算に特有の系統誤差が小さくなるようにシュミレーションを行い，誤差をコントロールしなければならない．系統誤差を生じる原因は次の 4 つである．

1. 空間方向の格子サイズが有限である効果
2. 時間方向の格子サイズが有限である効果（励起状態からの系統誤差）
3. クォーク質量に関する内挿・外挿の誤差
4. 格子間隔が有限である事による効果

我々は，この 4 つの系統誤差が，今日までの計算に比較して格段に小さくなるよう，パラメータを慎重に選び，また，種々の工夫を行ったシュミレーションを実行している．これまでに行なった quenched QCD のシミュレーションパラメータ及び計算時間を 表 38 に示した．

まず，格子サイズが有限である事に起因する系統誤差に関しては，今日までの研究によって比較的よく分析されている．quench 近似計算では，2.5 fm 程度のサイズがあれば系統誤差は充分小さい事が予想されている．我々は安全の為，3 fm 以上の格子を用いて計算を行った．

格子サイズ	β	La(fm)	a(fm)	配位数	PU 数	計算時間 (時間)
$32^3 \times 56$	5.90	3.2	0.100	800	256	3.0
$40^3 \times 70$	6.10	3.0	0.076	600	512	4.8
$48^3 \times 84$	6.25	3.0	0.063	420	1024	6.8
$64^3 \times 112$	6.47	3.0	0.047	91	2048	15.5

表 38: Quench 近似でのハドロン質量スペクトル計算のパラメータ，及び 1 配位当りの計算時間（ I/O 等を含む）．β は，格子間隔 a に対応したパラメータ，La は空間方向の物理的な格子サイズ．$64^3 \times 112$ 格子のシミュレーションは進行中．

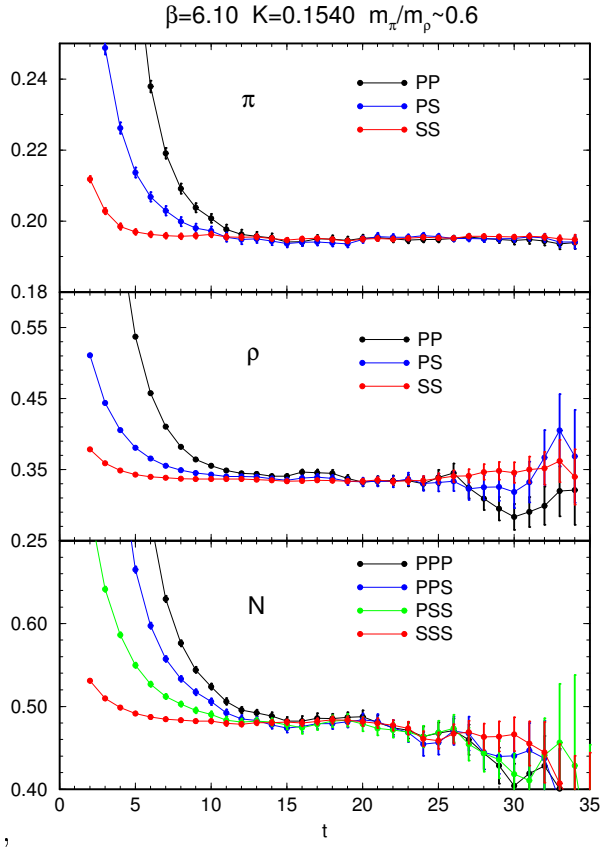


図 55: effective mass の典型的な結果．effective mass の quark source 依存性を示す．P は point quark source, S は smeared quark source を表す．すべての quark に対して smeared source を用いた場合に，effective mass は t が最も小さいところからフラットになる．上から順に，パイ中間子，ロー中間子，核子の effective mass であり，データは $40^3 \times 70$ 格子，$\beta = 6.1$，$m_\pi/m_\rho \approx 0.6$ のもの．

ハドロンの質量は，メソンに対しては 2 つ，バリオンに対しては 3 つのクォーク伝搬関数を組み合わせて構成されるハドロンの伝搬関数 $G_H(t)$ の時間 t の大きいところの振る舞いから決定される．具体的には，次式に示すように，$G_H(t)$ を指数関数でフィットして，時間に関する減衰率から質量を得る．

$$\begin{aligned} G_H(t) &= \sum_{\vec{n}} G_H(n) \\ &\sim A_0 e^{-m_0 t} + A_1 e^{-m_1 t} + \cdots \\ &\to A_0 e^{-m_0 t} \quad (t \to \infty) \end{aligned}$$

ここに m_0 は，求めたいハドロンの基底状態の質量であり，m_i $(i = 1, 2, \cdots)$ は，その励起状態の質量である．時間方向の格子サイズが十分でないか，そうであっても t の大きいところでの伝搬関数が十分の精度で計算できない場合，得られたハドロンの質量に励起状態からの誤差が含まれる．我々は今日までの経験を生かし，時間方向の格子サイズを約 5 fm に取り，励起状態からの寄与の小さいソース（smeared quark source）を採用することによって，励起状態からの寄与が，時間 t の小さいところでも小さくなる様工夫した．励起状態からの寄与の程度の指標は，effective

mass $m_{\rm eff}(t)$

$$m_{\rm eff}(t) = \log(G_H(t)/G_H(t+1)) \ \rightarrow \ m_0$$

である．effective mass は，励起状態からの寄与が十分小さくなれば，t の関数として一定になる．図 55 に effective mass の典型的な結果を示した．図に示した様に，effective mass には，広い t の領域に渡る plateau （effective mass がある精度の範囲で一定である領域）が見られる．これは，この領域の伝搬関数をフィットすることによって，励起状態の寄与がほとんど無視できる結果が得られる事を意味する．今日までの多くの計算では，t の広い領域に渡る plateau は，クォーク質量がかなり大きい場合を除いて，得られておらず，誤差の大きな原因となっていた．高い統計と quark source の工夫によって，軽いクォークに対しても精度の高いハドロンの質量が得られたことは，今回の計算の著しい結果である．

現実世界の u,d クォークは 5 MeV 程度の，s クォークは 140 MeV 程度の質量をもつ．軽いクォークの伝搬関数計算の計算量は，クォークの質量に逆比例して大きくなる．u,d クォーク質量に対応した計算を行うことは現実的に不可能であるので，より大きい質量のクォークのシュミレーションの結果を外挿して，u,d クォークから構成される軽いハドロン（陽子，Δ 粒 子等）の質量を求める．クォーク質量に関する外挿による系統誤差は，quench 計算において，最も定量的評価が困難な誤差である．今日までは，計算能力の不足により，クォーク質量 m_q がおおむね 50 MeV 程度以上の計算を行うのが精一杯であった．我々は，この誤差をより小さくするため，$m_q \approx 30$ MeV までのシュミレーションを行った．具体的には，$m_\pi/m_\rho \approx 0.75,\ 0.7$，0.6，0.5，0.4 に対応するクォーク質量（おおむね，$m_q \approx 155$，120，75，45，30 MeV に相当） の 5 点でのシュミレーションを行なった．最初の 2 つは，現実世界の s クォーク質量程度であるり，各々 $s1, s2$ と呼ぶ．現実の s クォークに対する結果は，この 2 点の内挿によって求めた．残りの 3 つは u_1, u_2, u_3 と呼び，u,d クォークへの外挿に用いた．また，u,d クォーク s クォークが混在するハドロンの質量を求めるため，メソンに対しては $s_i u_j$ 型の，バリオンに対しては $s_i s_i u_j$，$s_i u_j u_j$ 型のハドロンの質量の計算も行った．

ハドロンの質量は，第一近似では，それを構成するクォークの質量の和に比例すると考えられる．（パイ中間子は例外で，パイ中間子の質量の 2 乗がクォークの質量の和に比例する．）今日までの計算では，s クォーク程度以下の軽いクォークに対しては，おおむねこの性質が満たされている事が示されていた．しかし，我々の計算によって，この性質はより軽いクォークに対しては，必ずしもよい近似になっていない事が判明した．図 56 に，ハドロン質量のクォーク質量依存性の典型的な結果を示す．図に示すように，核子，及び Λ 粒子の質量はクォーク質量に関して凸関数である事が判った．これに対して，他の粒子の質量は，良い近似で，クォーク質量に比例することが確認された．核子と Λ 粒子に対する結果は，本シュミレーションによって，より軽いクォークに対する精度の高い結果が得られたことと，質量の異なるクォークから構成されるハドロンの質量が得られた事による，全く新しい知見である．

核子の質量がクォークの質量に対して凸関数であると言う結果は，quench 近似のスペクトラム計算にとって大きな意味がある．今日までの多くの計算では，核子の質量を現実世界の u,d クォーク質量まで外挿して得られる現実世界の核子の質量の予言値は，実験値に比べて幾分大きいという結果が得られており，quench 近似による系統誤差の一つの現れであるとも考えられていた．この結果は，比較的重い（$m_\pi/m_\rho \geq 0.5$，$m_q \geq 50$ MeV）クォークに対するシュミレーションで得られた質量を，クォーク質量の関数としての凸の曲がりを考慮に入れないで外挿して得られた結果である．本計算によって，より軽いクォークに対して，核子の質量が急激に小さくなる事が示されたので，核子の質量は，以前の予想よりかなり小さくなる．

最後の系統誤差は，格子間隔が有限である事に起因する誤差である．この誤差を小さくする為，

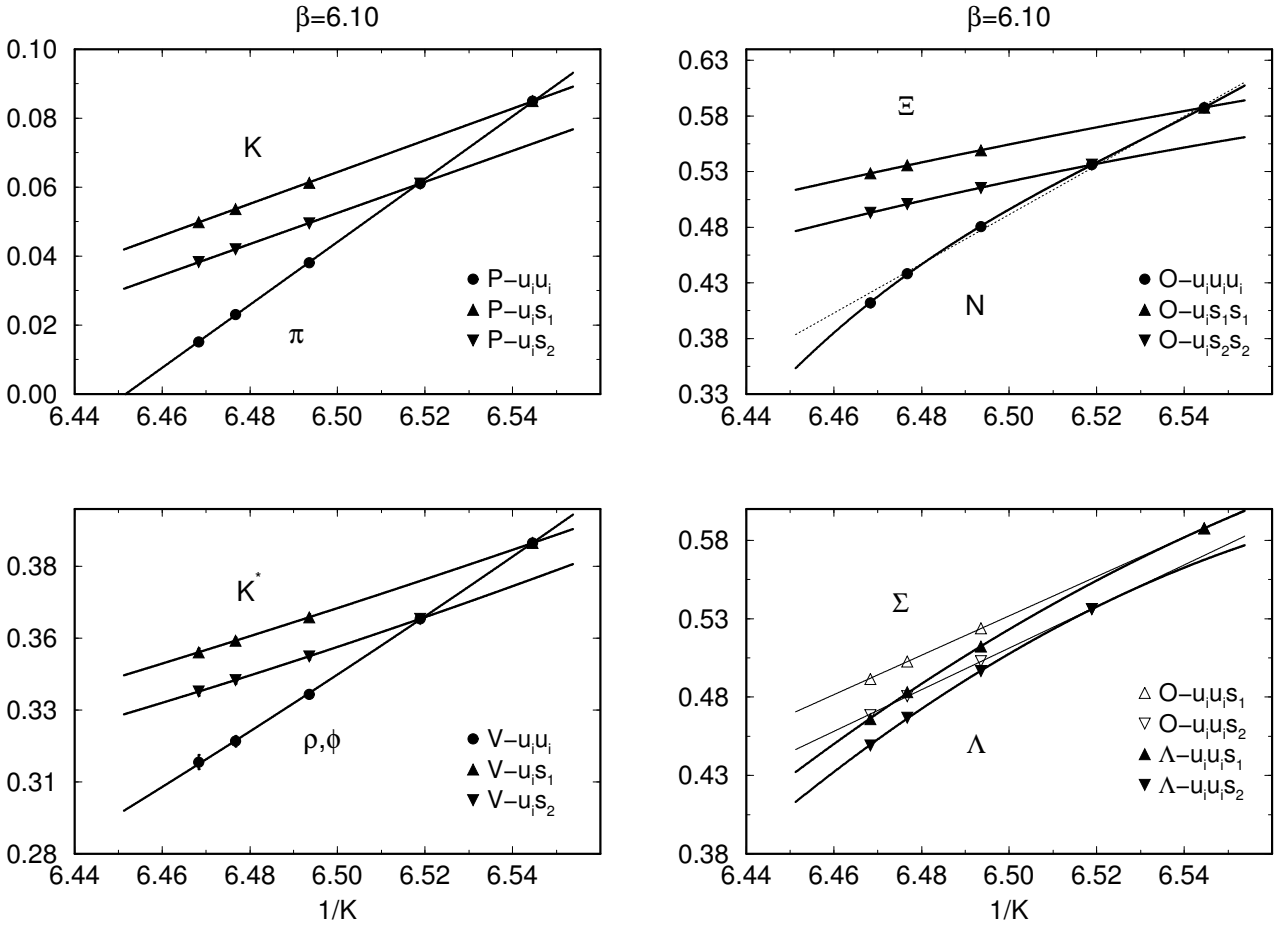


図 56: ハドロン質量のクォーク質量依存性の例．ウィルソンクォークの場合，クォーク質量 m_q は hopping parameter K の関数であり，おおむね $m_q \propto (1/K - 1/K_c)/2$ （K_c はある定数）の関係にある．図の横軸は $1/K$ で，クォーク質量（＋定数）と思って差し支えない．

格子間隔を 4 つ選んでシュミレーションを行い（表 38 参照），結果を格子間隔ゼロへ外挿した．我々が採用したウィルソンクォークの場合，格子間隔が充分小さければ，格子間隔有限の効果は格子間隔に比例することが理論的に予想されている．図 57 に例を示す様に，我々の計算した格子間隔の範囲では，ハドロン質量はよく格子間隔に比例するので，この関係式を仮定して外挿を行った．

以上の様に， 4 つの系統誤差に細心の注意を払ってシュミレーション及びデータ解析を行った．質量の信頼できる結果を得るためにはもちろん，高い統計も必要であり，現時点では，表 38 に示したサンプル数での結果が得られている．最終結果に対する統計誤差は，メソンに対しては 0.5% 程度，バリオンに対しては 1–3% 程度であり，quench 近似で予想される系統誤差 10% に対して充分小さい．図 58 に現時点でのハドロンの質量の最終結果を示した．

詳細になるので本稿では述べないが，ハドロン質量計算と共に，u, d, s クォークの質量，メソンの崩壊定数の計算も行った．

本計算で得られた結果は膨大であるが，重要な結果は，以下の様に要約できる．

1. quench 近似のハドロンの質量には，実験値との間に最大で 10σ にのぼる明白な差が見られるが，その大きさは最大で 10% 程度であり，quench 近似に対する系統誤差として理論的に

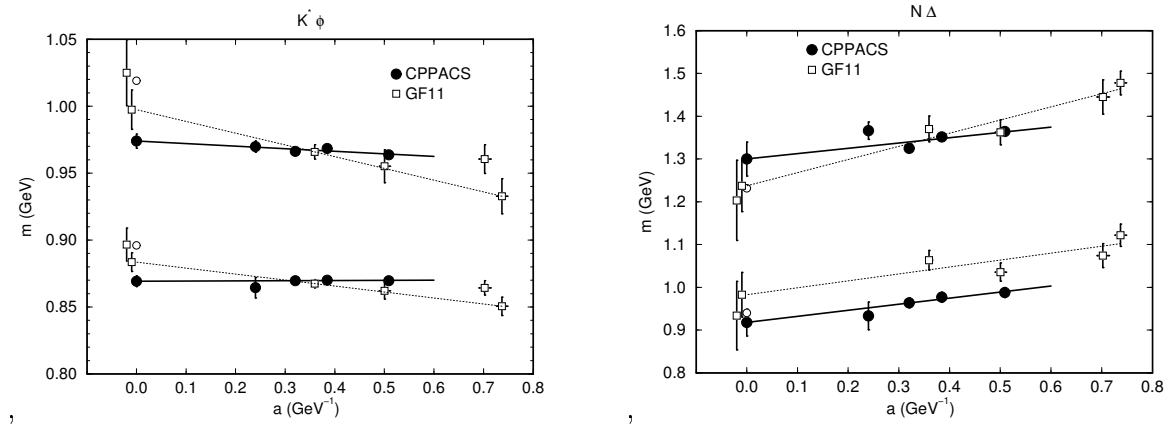


図 57: ハドロン質量の格子間隔に関する外挿の例．● が CP-PACS の結果．$a = 0$ の ○ は実験値を表す．参考のため GF11 グループの結果を | | で示した．s クォーク質量は K メソンの質量の実験値をインプットして決めた．

予想された範囲内である．

2. quench 近似による実験値との差異には，以下の特徴がある．

   (a) メソンの hyper-fine splitting（$K - K^*$ の質量差）は，実験値より 10% 程度小さい．

   (b) 8 重項のバリオンの質量に関し，核子の質量は実験値とコンシステントであるが，Λ,Σ, Ξ 粒子の質量は実験値より小さい．ただし，$\Sigma - \Lambda$ の質量差は実験値とコンシステントである．

   (c) 10 重項のバリオンの質量に関し，Δ は実験値より重く，Σ^* はコンシステント，Ξ^* は幾分軽く，Ω はかなり軽い．言い替えれば，10 重項間の質量差は，実験値より小さい．

3. ベクターメソン (ρ，ϕ) の崩壊定数は実験値とコンシステントである．

4. π 中間子の崩壊定数 (f_π) は実験値とコンシステントか，幾分小さいが，K 中間子のそれ (f_K) は，実験値よりかなり小さい．特に，$f_\pi/f_K - 1$ は，実験値よりかなり小さく，この結果は，カイラル摂動論による理論予測と矛盾しない．

5. クォーク質量の結果は，摂動的定義と非摂動的定義で，連続極限で同じ値を与える．これは，ウィルソンクォークのカイラル対称性が連続極限で回復するという理論的期待と矛盾しない．クォーク質量自身は，これまでの予測値より，かなり大きい．

22.1.2 full QCD 計算

上記の quenched QCD の研究により，格子 QCD で計算されるハドロンの物理量に，quenched 近似そのものに由来すると考えられる実験値からのずれが存在することが明らかになった．full QCD 計算を行う必要がある．また，有限温度 QCD や，U(1) 問題，核子シグマ項の計算など，本来 quench 近似では研究出来ないテーマも多く存在する．これらの状況から，格子 QCD の研究は，いよいよ quench 近似を取り除いた full QCD に努力を傾ける時代が到来したと言うことができる．

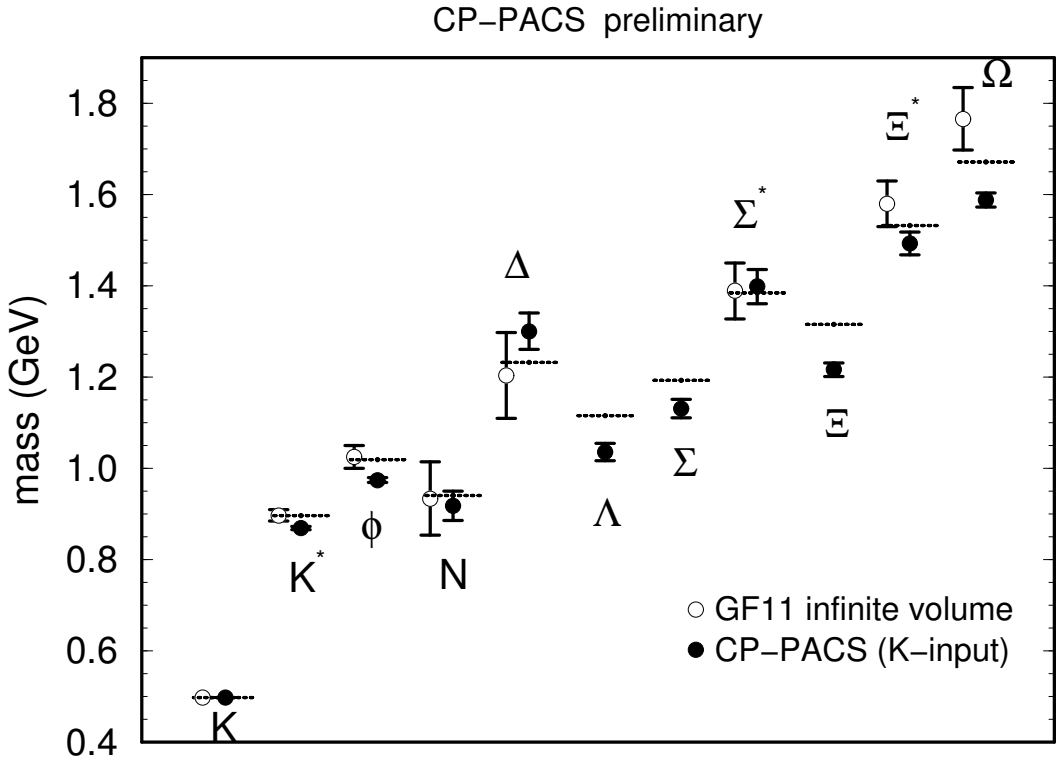


図 58: ハドロン質量の結果．● が CP-PACS の結果．破線は実験値．参考までに，GF11 グループの 結果を ○ で示した．

有限な格子で得られた様々な数値結果から，現実世界における物理量を計算するためには，物理的な格子サイズを十分大きくしながら，同時に格子間隔をゼロにする極限（連続極限）を取る必要がある．quenched QCD における軽いハドロンのスペクトル計算の経験から，有限サイズ効果を抑えて意味のある結果を得るためには，格子の物理的サイズが少なくとも $La >$ 2-2.5 fm でなければならない事が知られている．また，標準的に用いられている格子作用では，連続極限を取り得るためには格子間隔が $a \lesssim 0.1$fm でなければならない．CP-PACS による quench 近似計算は，まさにこのような条件をみたすように行われた計算である．しかしながら，同様な計算を full QCD で行おうとすると，full QCD では，ゲージ配位生成部分の計算量が quenched QCD より少なくとも 100 倍から 1000 倍以上多くなるため，計算量があまりに膨大になりすぎるという困難がある．

他方，この数年「改良された作用」の研究が世界的に急速に進められてきた．これは，空間的により広がった相互作用を取り入れ，その結合定数を調整しすることにより,「標準作用」より大きな格子間隔から連続極限の性質を実現するように改良した格子作用である．こうした改良により標準作用より粗い格子から連続極限を取り得るようになれば，上記の full QCD 研究の困難が解決されると期待できる．

様々な「改良された作用」が提案されているが，ハドロン スペクトルなどの非摂動論的な物理量に関して改良の効果が期待どうりのものであるかどうかは，実際にシミュレーションを行って確認しなければならない．これまでの研究では主に quenched QCD における改良の効果しか調べられていない．そこで我々は，CP-PACS における full QCD 研究の予備的研究として，full QCD における改良された作用の比較研究を行った．[22, 23]

改良はゲージ作用部分とクォーク作用部分の両方で可能であり，それぞれ性質の異なる効果が

	β	size	K	C_{SW}	$N_{\rm conf}$	m_π/m_ρ	a^{-1}[GeV]
P-W	4.8	$12^3\times 32$	.1846		222	.828(2)	1.00(3)
			.1874		200	.773(4)	
			.1891		200	.701(3)	
	5.0	$12^3\times 32$	.1779		300	.847(2)	1.08(4)
			.1798		301	.791(3)	
			.1811		301	.712(6)	
R-W	1.9	$12^3\times 32$	.1632		200	.899(2)	1.15(5)
			.1688		200	.804(3)	
			.1713		200	.688(4)	
	2.0	$12^3\times 32$	.1583		300	.898(2)	1.29(5)
			.1623		300	.827(3)	
			.1644		305	.736(4)	
P-C	5.0	$12^3\times 32$	.1590	1.0	100	.826(2)	0.97(5)
			.1610		100	.785(4)	
			.1630		100	.722(4)	
			.1415	1.855 (MF)	200	.810(2)	0.85(5)
			.1441	1.825	200	.750(4)	
			.1455	1.805	200	.711(4)	
	5.2	$12^3\times 32$	.1390	1.69 (MF)	248	.834(3)	1.3(1)
			.1410	1.655	232	.794(4)	
			.1420	1.64	200	.734(8)	
	5.25	$12^3\times 32$	.1390	1.637 (MF)	198	.834(4)	1.5(1)
			.1410	1.61	194	.756(7)	
R-C	1.9	$16^3\times 32$	.1370	1.55 (pMF)	203	.843(2)	1.15(2)
			.1400		198	.780(3)	
			.1420		202	.691(4)	
			.1430		212	.618(4)	
			.1435		163	.537(4)	
			.1440		34	.410(12)	
		$12^3\times 32$	.1370	1.55 (pMF)	267	.844(2)	1.0(1)
			.1400		214	.777(3)	
			.1420		268	.693(5)	
	2.0	$12^3\times 32$	.1420	1.0	100	.877(3)	1.15(5)
			.1450		100	.828(4)	
			.1480		100	.701(6)	
			.1300	1.505 (pMF)	100	.911(2)	1.3(2)
			.1370		90	.792(4)	
			.1388		90	.711(8)	
			.1300	1.54 (MF)	201	.902(1)	1.25(10)
			.1340	1.529	200	.862(3)	
			.1370	1.52	200	.790(4)	
			.1388	1.515	200	.705(8)	

表 39: full QCD の「改良された作用」の比較研究のための主要なシミュレーション．表の格子間隔の逆数 a^{-1} の値において，1GeV $\simeq$ $(0.197\text{fm})^{-1}$ である．C_{SW} は Clover クォークにおける改良項（Clover 項）の係数.

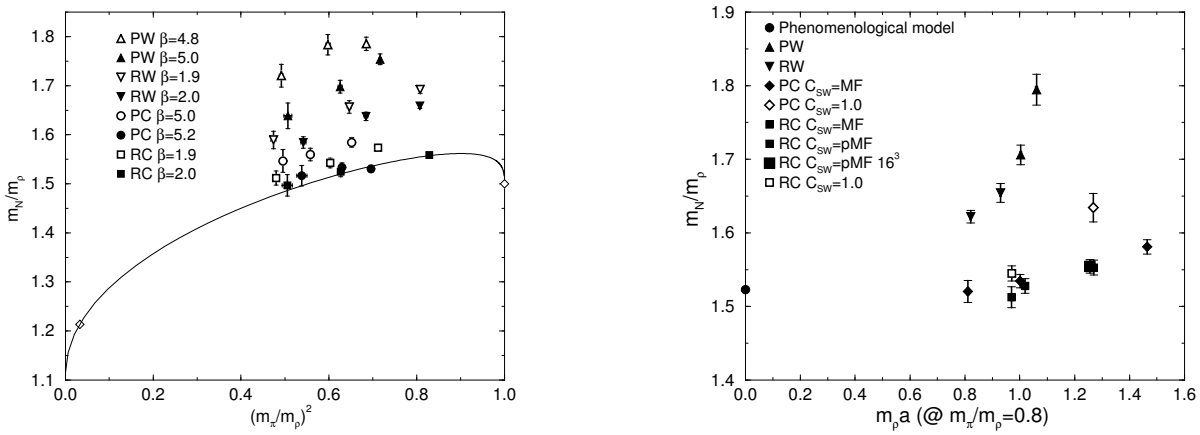


図 59: 様々な格子作用を用いた full QCD のハドロン スペクトル．格子サイズは $12^3 \times 32$．(a) 核子質量とロー中間子質量の比 m_N/m_ρ の $(m_\pi/m_\rho)^2$ 依存性．横軸はクォーク質量に比例する．ダイアモンド記号は実験値で，実線は現象論的モデルの結果．(b) $m_\pi/m_\rho = 0.8$ における m_N/m_ρ を格子間隔の関数として比較したもの．● は現象論的モデルの予言値．

期待されている．一般に，相互作用に より大きな空間的広がりを導入する程，より大きな改良を期待しうる．他方，空間的にあまり広がった相互作用は，並列プログラムの効率を阻害する．よって,（通常用いられている範囲内で）出来るだけ広がりを抑えた相互作用が，それぞれどの程度の改良効果を持っているのかを知る必要がある．

ここでは，ゲージ部分の作用に関しては，標準の *plaquette* 作用（**P**）と，長方形の相互作用まで取り入れた *plaquette* + *rectangular* 作用（**R** の 2 つ，クォーク作用として，標準の Wilson クォーク作用（**W**）と，Wilson クォークを改良した Clover クォーク（**C**）の 2 つを比較した．よって，**P-W**，**P-C**，**R-W**，**R-C** の合計 4 種類の作用に対するシミュレーションを実行し，ハドロンのスペクトル，重いクォークのポテンシャル等の物理量でそれぞれの改良の程度を調べた．この研究では，$8^3 \times 16$ と $12^3 \times 32$ 格子を 64PU $(4 \times 4 \times 4)$ パーティションでシミュレーションした．また，改良の効果を明確に見るために格子間隔は比較的粗い $a \simeq 0.2$fm とし，クォーク質量は $m_\pi/m_\rho \simeq 0.7$-0.9 をとなるように，格子のパラメータを選んだ．表 39 に，これまでに実行された主要なシミュレーションのパラメータをまとめる．

図 59(a) に，$12^3 \times 32$ 格子で得られたハドロン質量の結果をまとめる．図の実線は，現象論的にハドロン質量の実験値を良く再現するモデルから得られた結果で，格子 QCD も連続極限に近づけばそれに近い値を出すであろうと期待されている．この図から明らかなように，作用の改良は粗い格子で連続極限の性質を再現する上で極めて有効である．各作用で格子間隔が多少違うので，より精密な比較を行うために，図 59(b) に，$m_\pi/m_\rho = 0.8$ における m_N/m_ρ を格子間隔の関数として比較したものを示す．これから，m_N/m_ρ 等のハドロンのスペクトルに関する性質に関しては，クォーク作用の改良が主要な寄与をしていることが明らかになった．また，Clover 作用によってハドロン スペクトルに十分大きな改良効果が得られることも示された．同様の結論は，デルタ バリオンの質量からも導かれる．また，中間子伝搬関数の運動量依存性の研究からも Clover 作用の改良効果が確認できた．

重いクォークのポテンシャルの研究からは，ゲージ作用の改良が，格子化で壊された回転対称性の回復に有効である事が示された．図 60 に，$a \sim 0.2$fm の格子におけるポテンシャルの結果を示す．図で異なるシンボルは格子上の異なる方向に対応する．**P-W** と **P-C** の場合にデータがひ

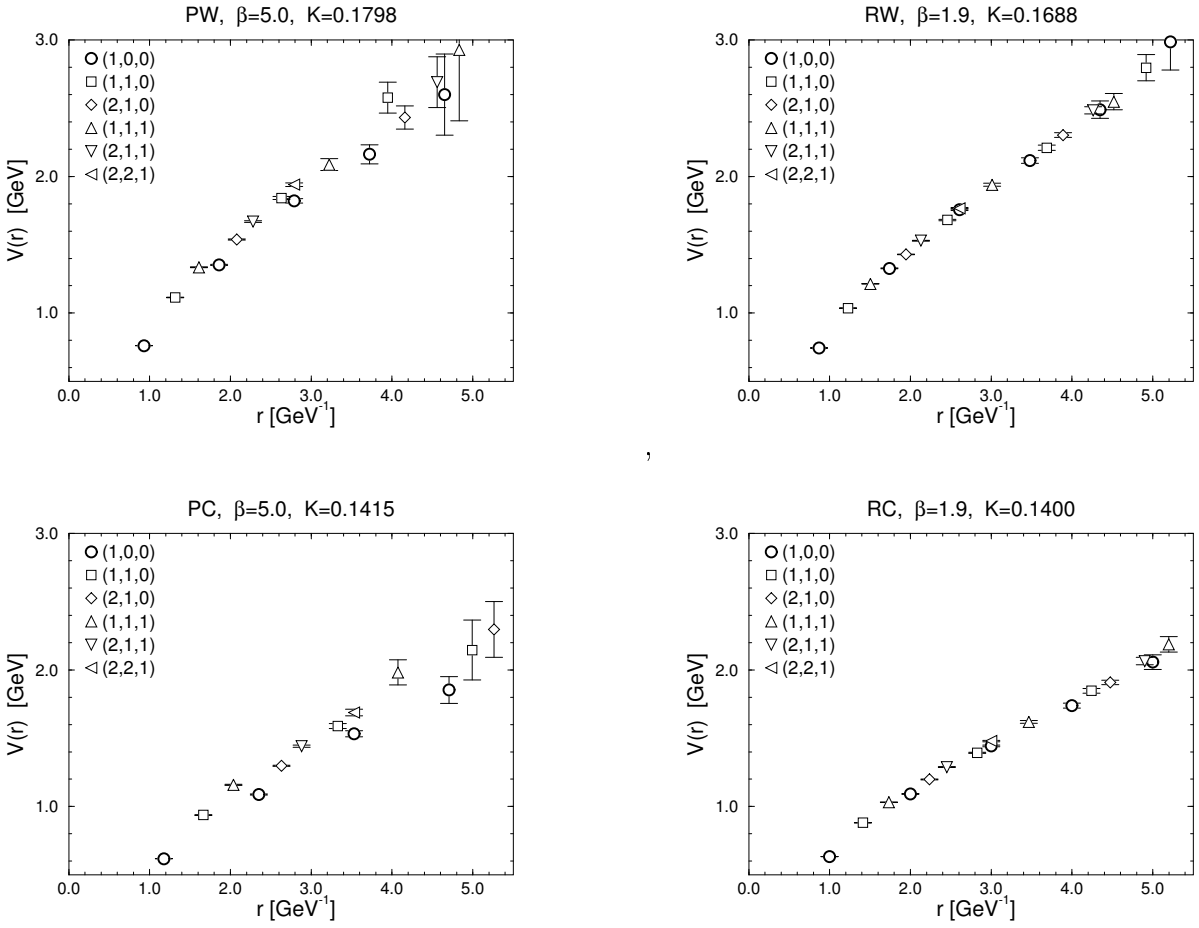


図 60: full QCD における重いクォークのポテンシャル．格子サイズは $12^3 \times 32$ で，格子間隔が $a \sim 0.2$fm になるようにパラメータを調節した．

とつの曲線上に揃わないのは，連続極限で回復すべき回転対称性が $a \simeq 0.2$fm の粗い格子上ではまだ有意に壊れている事を意味する．ポテンシャルにおける回転対称性の破れは，弦張力などの物理量の計算に不定性と誤差をもたらす．他方，**R-W** や **R-C** の場合には，$a \simeq 0.2$fm の粗い格子でもすでに回転対称性が非常に良く回復している．これから，回転対称性の回復にはゲージ作用の改良が主に寄与しており，**R** 作用がそれに極めて有効である事が示された．

様々な物理量の連続極限を研究するためにはスペクトルも回転対称性も重要であり，上記の結果は，今後のシミュレーションではゲージ作用とクォーク作用の両方を改良する必要があることを意味する．同時に，ゲージ作用としては，標準の *plaquette* 作用に長方形相互作用を加えた *plaquette* + *rectangular* 作用（**R**）によって，また，クォーク作用としては，標準の Wilson クォーク作用に Clover 項を加えた Clover 作用（**C**）を用いることにより，それぞれ極めて大きな改良効果が見られた．従って，効率的に並列化しうるこれらの最小限の改良によっても，十分効果的に連続極限に近づきうることが確認された．

現在，最も有望な **R-C** の場合のより詳細なパラメータ依存性を調べるために，$16^3 \times 32$ 格子を 256 PU $(4 \times 8 \times 8)$ パーティションでシミュレーション中である．

22.1.3 今後の計画

前節に述べた quenched QCD の大規模計算は，quench 近似の限界を明らかにし，研究重点の full QCD への移行の必要性を強調するものである．計算規模の観点から，これは決して容易なことではないが，続いて行われた full QCD における「改良された作用」の比較研究により，作用の最小限の改良でも連続極限へのスムースな外挿に極めて有用であることが示された．さらに，現在シミュレーション中の $16^3 \times 32$ 格子では，$m_\pi/m_\rho \sim 0.4$ というクォーク質量もテストしており，その結果，BiCGStab 法などのアルゴリズムの改良を同時に行えばこのような軽いクォークも full QCD でシミュレーション可能であることがわかってきた．これらは，CP-PACS を用いることにより，full QCD による現実的な計算を十分行い得る可能性を示している．今後はこの方向を重点的に押し進め，full QCD シミュレーションによるハドロンの諸性質の解明と予言が中心目標となる．特に，軽いハドロンのスペクトルや崩壊定数のみならず，U(1) 問題などの full QCD 固有の問題，クォークグルオンプラズマ相転移係など有限温度 QCD の研究，さらには重いクォークや CP 非保存等弱い相互作用の長年の懸案に関係する問題など，研究対象を順次拡大していくことを計画している．

22.2 宇宙物理学

宇宙物理における CP-PACS の応用として我々は，輻射輸送に関連した問題に焦点を当てている．特に，3 次元空間での輻射輸送は重要な問題であるにもかかわらず，計算量・メモリ量ともに膨大であるために従来はほとんど行われていなかった．このような計算も，CP-PACS を利用することで実現できるようになってきた．宇宙物理における CP-PACS を利用した最初の応用計算の成果を，以下に簡単に報告する．

22.2.1 宇宙初期の電離状態

宇宙は，およそ 140 億年前にクエーサーが誕生したことによって非常に高く電離されたことが観測によりわかっている．宇宙の電離は矮小銀河形成の阻害など銀河形成史に大きな影響を及ぼしたと考えられる．我々は，宇宙の電離に伴う様々な現象を調べるために，冷たいダークマター (CDM) 宇宙モデルに基づき，密度揺らぎを持った宇宙の中での電離紫外線 (UV) の輻射輸送を解くことを試みた．この計算の新しい点は，輻射輸送を初めて 3 次元空間で解いた点である．この計算は，空間 3 次元，方向 2 次元，そして振動数 1 次元の計 6 次元の問題である．最初の 5 次元分については，64^5 ないし 128^5 の格子を置いた．振動数依存性については，半解析的前処理をすることによって振動数空間を 6 格子で近似する方法を開発し計算に用いた．

具体的な計算は次のようにして行った．まず，ダークマターの宇宙密度パラメーターを $\Omega_{\rm CMD} = 0.95$，バリオン物質の密度パラメーターを $\Omega_{\rm baryon} = 0.05$ とし，冷たいダークマター宇宙の標準モデルに基づいて 3 次元空間内での密度分布を与える．この際，密度ゆらぎの振幅を COBE 規格化条件を満たすように設定し，Zel'dovich 近似により密度ゆらぎを生成する．この密度分布の中で，水素とヘリウムからなる原始組成ガスを考え，振動数依存の UV 輻射輸送を解くことにより電離度を計算する．電離度の計算と輻射輸送の計算は，これらが収束するまで交互に繰り返し，定常問題の解として self-consistent な電離度と輻射場を求める．計算領域の境界から入射する UV 輻射は，その強度と振動数依存性をパラメータとして与えた．

この計算によって得られた主な結果は次のようなものである．

(1) 中性雲の形成 — 平均 UV 強度が弱い場合，self-shielding によって，密度の高い領域で中性雲が形成される．また，この中性雲の近傍では，中性雲による shadowing 効果が見られる．

(2) 力学効果 — 密度分布により輻射場の非等方性が生じるため，隣接した雲に対し UV 輻射力による引力が働く．

(3) 中性水素密度分布の統計 — 一様な UV 輻射強度を仮定する場合に比べて，高い中性水素柱密度を持つ雲の割合が増える．

(4) 矮小銀河形成の阻害 — 赤方偏移 $2 \le Z \le 10$ において，境界から入射させる紫外線の Lyman limit での平均強度 J_{21} ($\equiv J/10^{-21}$ erg cm^{-2} s^{-1} Hz^{-1} str^{-1}) を $10^{-3} \le J_{21} \le 1$ の範囲で変化させて電離度を計算した結果，ある UV 強度以上になると矮小銀河に進化できるような中性雲は急激に減少することがわかった．この効果は低赤方偏移ほど重要である．

計算結果の一例を図 61 に示す．これは，一辺 8 Mpc $= 2.4 \times 10^{24}$cm の立方体の内部の水素ガスの電離状態を求めたものである．図 61 には，全水素粒子に占める中性水素粒子の割合 X_{H1} を示した．赤方偏移は $Z = 4$，外部からの入射紫外線の強度 J_{21} は 0.1 とした．この図の計算は，$64^3 \times 64^2 \times 1$ 格子を利用した単色光近似のものである．上記 (1) に記したように，比較的密度の高い領域で中性水素雲 ($X_{H1} > 0.1$ の領域) が形成されることがわかる．また，中性度の高い領域 (将来，銀河になりうると思われる領域) は水素ガス雲の深い部分に埋もれていることもわかる．

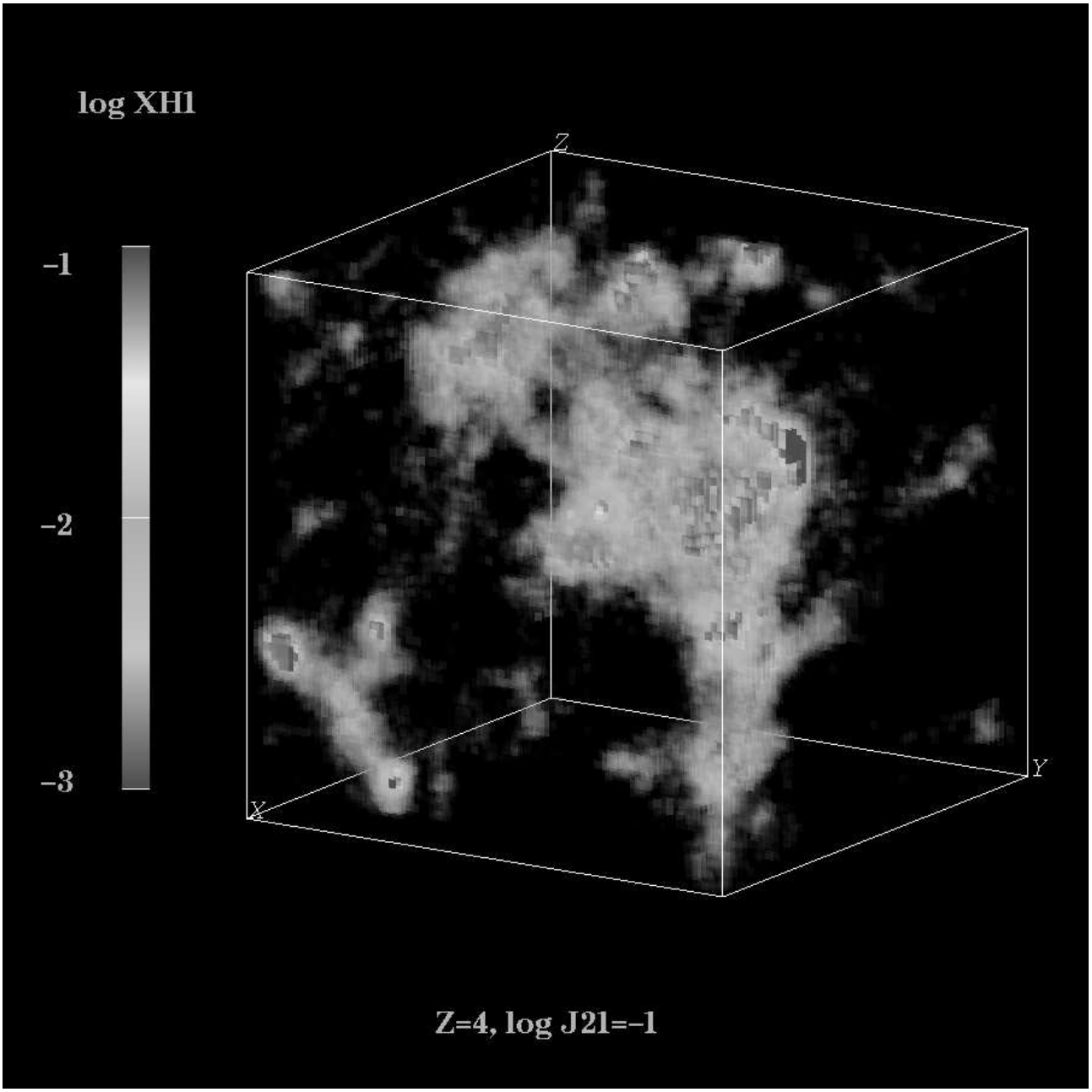


図 61: 宇宙初期の水素ガスの電離状態

22.2.2　今後の計画

輻射が関係する多次元問題では輻射輸送を解くことが最も重い計算になっていたが，この部分が CP-PACS を利用することによって現実的な時間で実行可能になり，多くの問題の解決への足掛かりが出来つつある．今後は輻射輸送に関連する多様な問題，すなわち，宇宙初期の電離状態のほか，銀河の形成に関連する問題，星の形成に関連する問題，惑星の形成に関連する問題，などへの応用を進めていく予定である．

参考文献

[1] CP-PACS プロジェクトに関する総合報告は次に掲載された.

特集『計算物理学と超並列計算機 – CP-PACS 計画 –』, 情報処理 37 (1996):

中村 宏, 「特集『計算物理学と超並列計算機 – CP-PACS 計画 –』の編集にあたって」, p.10.

岩崎 洋一, 宇川 彰, 梅村 雅之, 「計算物理学と CP-PACS 計画」, p.11-17.

中澤 喜三郎, 中村 宏, 朴 泰祐, 「超並列計算機 CP-PACS のアーキテクチャ」, p.18-28.

中田 育男, 山下 義行, 小柳 義夫, 「超並列計算機 CP-PACS のソフトウェア」, p.29-37.

青木 慎也, 金谷 和至, 吉江 友照, 「超並列計算機 CP-PACS の計算物理学分野における実効性能の予測」, p.38-42.

国際会議等におけるプロジェクトの年次進行状況の報告は以下のとおりである.

Y. Oyanagi, "New parallel computer project in Japan dedicated to computational physics", in Proceedings of Lattice '92 (Amsterdam, The Netherlands, 15-19 Sept., 1992), Nucl. Phys. B (Proc. Suppl.) 30 (1993) 229-232.

Y. Iwasaki, "Computers for lattice field theories", in Proceedings of Lattice '93 (Dallas, USA, 12-16 Oct., 1993), Nucl. Phys. B (Proc. Suppl.) 34 (1994) 78-92.

A. Ukawa, "Status of the CP-PACS Project", in Proceedings of Lattice '94 (Bielefeld, Germany, 27 Sept.-10 Oct., 1994), Nucl. Phys. B (Proc. Suppl.) 42 (1995) 194-200.

中澤 喜三郎, 「計算物理学研究用並列計算機: CP-PACS」, 応用数理, 第 6 巻, (1996) 17-28.

Y. Iwasaki, "Status of the CP-PACS Project", in Proceedings of Lattice '96 (St. Louis, USA, 4-8 June, 1996), Nucl. Phys. B (Proc. Suppl.) 53 (1997) 1007-1009.

Y. Iwasaki, "The CP-PACS Project", in Proceedings of the International Workshop "Lattice QCD on Parallel Computers" (Tsukuba, March 10-15, 1997), Nucl. Phys. B (Proc. Suppl.), to appear.

T. Boku, H. Nakamura, K. Nakazawa, Y. Iwasaki, "The Architecture of Massively Parallel Processor CP-PACS", in Proceedings of 2nd Aizu International Symposium on Parallel Algorithm/Architecture Synthesis (1997) 31-40.

T. Boku, K. Itakura, H. Nakamura, K. Nakazawa, "CP-PACS: A massively parallel processor for large scale scientific calculations", in Proceedings of ACM International Conference on Supercomputing '97 (1997) 108-115.

[2] 日本物理学会編: 計算物理学 (培風館, 1991).

[3] 素粒子物理分野における専用並列計算機に関する総合報告は以下を見られたい.

N. Christ, Nucl. Phys. B (Proc. Suppl.) 9 (1989) 549-556.
R. Tripiccione, Nucl. Phys. B (Proc. Suppl.) 17 (1990) 137-145.
N. Christ, Nucl. Phys. B (Proc. Suppl.) 20 (1991) 129-137.
D. Weigarten, Nucl. Phys. B (Proc. Suppl.) 26 (1992) 126-136.

E. Marinari, Nucl. Phys. B (Proc. Suppl.) 30 (1993) 122-135.
Y. Iwasaki, Nucl. Phys. B (Proc. Suppl.) 34 (1994) 78-92.
J.C. Sexton, Nucl. Phys. B (Proc. Suppl.) 47 (1996) 236-247.

[4] QCDPAX 計画
Y. Iwasaki, T. Hoshino, T. Shirakawa, Y. Oyanagi and T. Kawai, "QCDPAX: A parallel computer for lattice QCD simulation", Comp. Phys. Comm. 49 (1988) 449-455.
Y. Iwasaki, K. Kanaya, T. Yoshie, T. Hoshino, T. Shirakawa, Y. Oyanagi, S. Ichii and T. Kawai, "Status of QCDPAX", Nucl. Phys. B(proc. Suppl.) 17 (1990) 259-262.
T. Shirakawa, T. Hoshino, Y. Oyanagi, Y. Iwasaki, T. Yoshie, K. Kanaya, S. Ichii and T. Kawai, "QCDPAX - An MIMD array of vector processors for the numerical simulation of quantum chromodynamics", Proceedinds of Supercomputing '89, Reno, USA, Nov. 13-17, 1989, 495-504.

[5] PAX 計画
星野力編著:「PAX コンピュータ」(オーム社, 1985).

[6] H. Nakamura, H. Imori, K. Nakazawa, T. Boku, I. Nakata, Y. Yamashita, H. Wada, Y. Inagami, "A Scalar Architecture for Pseudo Vector Processing based on Slide-Windowed Registers", in Proceedings of ACM International Conference on Supercomputing '93 (1993) 298-307.

[7] 山下義行, 中田育男, 「ループ中に条件分岐を含む場合の最適なソフトウェア・パイプライニング」, 並列処理シンポジウム JSPP'94 論文集, 17-24.

[8] 朴 泰祐, 板倉 憲一, 曽根 猛, 三島健, 中澤 喜三郎, 中村 宏, 「ハイパクロスバ・ネットワークにおける転送性能向上のための手法とその評価」, 情報処理学会誌 第 36 巻, 7 号 (1995) 1610-1618.

[9] 山下義行, 中田育男, 「ソフトウェア・パイプライニングにおける多重ループの最適化」, 並列処理シンポジウム JSPP'95 論文集 (1995) 185-192.

[10] K. Shimamura, S. Tanaka, T. Shimomura, T. Hotta, E. Kamada, H. Sawamoto, T. Shimizu, K. Nakazawa, "A Superscalar RISC Processor with Pseudo Vector Processing Feature", in Proceedings of ICCD'95 (IEEE International Conference on Computer Design: VLSI in Computers and Processors) (1995) 102-109.

[11] 中田育男, 山下義行, 小柳義夫, 「超並列計算機 CP-PACS のソフトウェア」, 情報処理, 第 37 巻 (1996) 29-37.

[12] 曽根 猛, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「ハイパクロスバ・ネットワークにおける Virtual Channel の動的選択による適応ルーティング」, 情報処理学会誌 第 37 巻, 第 7 号 (1996) 1409-1418.

[13] 萩川友宏, 添野元秀, 山下義行, 中田育男, 「スライドウインドウを考慮したレジスタ割付」, 日本ソフトウェア科学会第 13 回大会論文集 (1996) 201-204.

[14] Preeti Ranjan Panda, Hiroshi Nakamura, Nikil D. Dutt, Alexandru Nicolau, "A Data Alignment Technique for Improving Cache Performance", in Proceedings of International Conference on Computer Design (ICCD '97), (Austin, USA) (1997), to appear.

[15] D. Bailey, E. Barszcz, J. Barton, D. Browning, R. Carter, R. Fatoohi, S. Fineberg, P. Frederickson, T. Lasinski, R. Schreiber, H. Simon, "The NAS Parallel Benchmarks", NAS, RNR-94-007, 1994.

[16] 板倉 憲一, 松原 正純, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「超並列計算機 CP-PACS における NPB Kernel CG の評価」, 並列処理シンポジウム JSPP'97 論文集 (1997) 5-12.

[17] M. Yoshida, A. Nakamura, M. Fukuda, T. Nakamura, and S. Hioki, in Proc. 1995 ACM-IEEE Supercomputing, San Diego, 1995, CD-ROM.

[18] `http://www.nal.go.jp/www-e/facility/ns2/home.html`.

[19] 荻津 格, 常行 真司, 「CP-PACS による物性物理：第一原理計算」, 筑波大学計算物理学研究センター研究会「並列計算機による物理学」（平成 9 年 3 月 26 日- 27 日）

[20] CP-PACS Collaboration: S. Aoki, G. Boyd, R. Burkhalter, N. Ishizuka, Y. Iwasaki, K. Kanaya, Y. Kuramashi, M. Okawa, A. Ukawa, and T. Yoshié, "CP-PACS results for quenched QCD spectrum with the Wilson action", in Proceedings of International Workshop "Lattice QCD on Parallel Computers" (Tsukuba, March 10-15, 1997), Nucl. Phys. B (Proc. Suppl.), to appear.

[21] CP-PACS Collaboration: S. Aoki, G. Boyd, R. Burkhalter, N. Ishizuka, Y. Iwasaki, K. Kanaya, T. Kaneko, Y. Kuramashi, M. Okawa, A. Ukawa, and T. Yoshié, "CP-PACS results for the quenched light hadron spectrum", in Proceedings of Lattice '97 (Edinburgh, Scotland, 22-26 July, 1997), Nucl. Phys. B (Proc. Suppl.), to appear.

[22] CP-PACS Collaboration: S. Aoki, G. Boyd, R. Burkhalter, N. Ishizuka, Y. Iwasaki, K. Kanaya, T. Kaneko, Y. Kuramashi, M. Okawa, A. Ukawa, and T. Yoshié, "Full QCD simulation on CP-PACS", in Proceedings of International Workshop "Lattice QCD on Parallel Computers" (Tsukuba, March 10-15, 1997), Nucl. Phys. B (Proc. Suppl.), to appear.

[23] CP-PACS Collaboration: S. Aoki, G. Boyd, R. Burkhalter, N. Ishizuka, Y. Iwasaki, K. Kanaya, T. Kaneko, Y. Kuramashi, M. Okawa, A. Ukawa, and T. Yoshié, "Hadron spectroscopy and static quark potential in full QCD: A comparison of improved actions", in Proceedings of Lattice '97 (Edinburgh, Scotland, 22-26 July, 1997), Nucl. Phys. B (Proc. Suppl.), to appear.

第IV部
研究発表

23 論文及びプロシーデングス

23.1 CP-PACS 計画全般

1. Y. Oyanagi, "New parallel computer project in Japan dedicated to computational physics", in Proceedings of Lattice '92 (Amsterdam, The Netherlands, 15-19 Sept., 1992), Nucl. Phys. B (Proc. Suppl.) 30 (1993) 229-232.

2. Y. Iwasaki, "Computers for lattice field theories", in Proceedings of Lattice '93 (Dallas, USA, 12-16 Oct., 1993), Nucl. Phys. B (Proc. Suppl.) 34 (1994) 78-92.

3. A. Ukawa, "Status of the CP-PACS Project", in Proceedings of Lattice '94 (Bielefeld, Germany, 27 Sept.-10 Oct., 1994), Nucl. Phys. B (Proc. Suppl.) 42 (1995) 194-200.

4. 特集『計算物理学と超並列計算機 – CP-PACS 計画 –』, 情報処理 37 (1996):
   中村 宏, 「特集『計算物理学と超並列計算機 – CP-PACS 計画 –』の編集にあたって」, p.10,
   岩崎 洋一, 宇川 彰, 梅村 雅之, 「計算物理学と CP-PACS 計画」, p.11-17,
   中澤 喜三郎, 中村 宏, 朴 泰祐, 「超並列計算機 CP-PACS のアーキテクチャ」, p.18-28,
   中田 育男, 山下 義行, 小柳 義夫, 「超並列計算機 CP-PACS のソフトウェア」, p.29-37,
   青木 慎也, 金谷 和至, 吉江 友照, 「超並列計算機 CP-PACS の計算物理学分野における実効性能の予測」, p.38-42.

5. 中澤 喜三郎, 「計算物理学研究用並列計算機: CP-PACS」, 応用数理, Vol.6, (1996) 17-28.

6. Y. Iwasaki, "Status of the CP-PACS Project", in Proceedings of Lattice '96 (St. Louis, USA, 4-8 June, 1996), Nucl. Phys. B (Proc. Suppl.) 53 (1997) 1007-1009.

7. Y. Iwasaki, "The CP-PACS Parallel Computer Project", in Proceedings of International Conference "Multi-Scale Phenomena and Their Simulation", World Scientific (1997) 80-90.

8. Y. Iwasaki, "The CP-PACS project", in Proceedings of the International Workshop "Lattice QCD on Parallel Computers" (Tsukuba, March 10-15, 1997), Nucl. Phys. B (Proc. Suppl.), to appear.

9. A. Ukawa, "The CP-PACS Parallel Computer", in Proceedings of CHEP'97 (Berlin, April 7-11, 1997) 595-600.

10. T. Boku, H. Nakamura, K. Nakazawa, Y. Iwasaki, "The Architecture of Massively Parallel Processor CP-PACS", in Proceedings of 2nd Aizu International Symposium on Parallel Algorithm/Architecture Synthesis (1997) 31-40.

11. T. Boku, K. Itakura, H. Nakamura, K. Nakazawa, "CP-PACS: A massively parallel processor for large scale scientific calculations", in Proceedings of ACM International Conference on Supercomputing '97 (1997) 108-115.

23.2 CP-PACS のアーキテクチャとソフトウエア

【平成 4 年度】

1. 中村 宏, 位守 弘充, 伊藤 元久, 中澤 喜三郎, 「レジスタウィンドウとスーパスカラ方式による擬似ベクトルプロセッサの提案」, 並列処理シンポジウム JSPP'92 論文集 (1992) 367-374.

2. K. Nakazawa, H. Nakamura, H. Imori, S. Kawabe, "Pseudo Vector Processor based on Register-Windowed Superscalar Pipeline", in Proceedings of Supercomputing '92 (1992) 642-651.

【平成 5 年度】

1. 位守 弘充, 中村 宏, 朴 泰祐, 中澤 喜三郎, 「スライドウィンドウ方式による擬似ベクトルプロセッサ」, 情報処理学会論文誌, 第 34 巻, 第 12 号 (1993) 2612-2623.

2. H. Nakamura, H. Imori, K. Nakazawa, T. Boku, I. Nakata, Y. Yamashita, H. Wada, Y. Inagami, "A Scalar Architecture for Pseudo Vector Processing based on Slide-Windowed Registers", in Proceedings of ACM International Conference on Supercomputing '93 (1993) 298-307.

3. 中村 宏, 中澤 喜三郎, 李 航, 位守 弘充, 朴 泰祐, 「スライドウィンドウ方式に基づく擬似ベクトルプロセッサ」, 情報処理学研究報告 93-ARC-101-11 (1993) 81-88.

【平成 6 年度】

1. 山下 義行, 中田 育男, 「ループ中に条件分岐を含む場合の最適なソフトウェア・パイプライニング」, 並列処理シンポジウム JSPP'94 論文集 (1994) 17-24.

2. 中澤 喜三郎, 朴 泰祐, 中村 宏, 中田 育男, 山下 義行, 岩崎 洋一, 「CP-PACS のアーキテクチャの概要」, 情報処理学会研究報告 94-ARC-108-9 (1994) 57-64.

【平成 7 年度】

1. 山下 義行, 中田 育男, 「ソフトウェア・パイプライニングにおける多重ループの最適化」, 並列処理シンポジウム JSPP'95 論文集 (1995) 185-192.

2. 斎藤 拡二, 橋本 眞宏, 澤本 英雄, 熊谷 多加史, 山縣 良, 釜田 栄樹, 松原 健二, 柏山 正守, 磯部 敏子, 堀田 多加志, 中野 哲夫, 清水 照久, 中澤 喜三郎, 「擬似ベクトル機構を有する並列コンピュータ向け RISC プロセッサ」, 電子情報通信学会技術研究報告 DSP95-104/ICD95-153 (1995) 1-6.

3. K. Shimamura, S. Tanaka, T. Shimomura, T. Hotta, E. Kamada, H. Sawamoto, T. Shimizu, K. Nakazawa, "A Superscalar RISC Processor with Pseudo Vector Processing Feature", in Proceedings of ICCD'95 (IEEE International Conference on Computer Design: VLSI in Computers and Processors) (1995) 102-109.

4. K. Saito, M. Hashimoto, H. Sawamoto, R. Yamagata, T. Kumagai, E. Kamada, K. Mastubara, T. Isobe, T. Hotta, T. Nakano, K. Nakazawa, "A 150 MHz Superscalar RISC Processor with Pseudo Vector Processing Feature", in Proceedings of Hot Chips VII('95), (1995) 197-205.

【平成 8 年度】

1. 秡川 友宏, 添野 元秀, 山下 義行, 中田 育男, 「スライドウインドウを考慮したレジスタ割付」, 日本ソフトウェア科学会第 13 回大会論文集 (1996) 201-204.

2. 秡川 友宏, 添野 元秀, 山下 義行, 中田 育男, 「スライドウインドウを考慮したレジスタ割付」, 情報処理学会第 54 回全国大会講演論文集 (分冊 1), (1997) 191-120.

23.3 CP-PACS の性能評価

【平成 4 年度】

1. 斎藤 哲也, 森本 貴之, 位守 弘充, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「超並列計算機のネットワークの実現可能性と性能評価」, 情報処理学会研究報告 92-ARC-95-4 (1992) 25-32.

2. 位守 弘充, 中村 宏, 朴 泰祐, 中澤 喜三郎, 「擬似ベクトルプロセッサによるリストベクトル処理とその評価」, 情報処理学会研究報告 92-ARC-96-17 (1992) 117-124.

【平成 5 年度】

1. 中村 宏, 位守 弘充, 中澤 喜三郎, 「レジスタウィンドウ方式を用いた擬似ベクトルプロセッサの評価」, 情報処理学会論文誌 第 34 巻, 第 4 号 (1993) 669-680.

2. 朴 泰祐, 斉藤 哲也, 板倉 憲一, 中澤 喜三郎, 中村 宏, 「ハイパクロスバ・ネットワークの性能評価」, 電子情報通信学会技術研究報告 CPSY-93-40 (1993) 41-48.

【平成 6 年度】

1. H. Nakamura, K. Nakazawa, H. Li, H. Imori, T. Boku, I. Nakata, and Y. Yamashita, "Evaluation of Pseudo Vector Processor based on Slide-Windowed Registers", in Proceedings of Hawaii International Conference on System Sciences 27 (1994) 368-377.

2. 朴 泰祐, 曽根 猛, 三島 健, 板倉 憲一, 中澤 喜三郎, 中村 宏, 「ハイパクロスバ・ネットワークにおける転送性能向上のための手法とその評価」, 並列処理シンポジウム JSPP'94 論文集 (1994) 121-128.

3. 曽根 猛, 三島 健, 板倉 憲一, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「ハイパクロスバ・ネットワークにおけるバッファの利用法と転送性能について」, 電子情報通信学会技術研究報告 CPSY-94-53 (1994) 97-104.

4. H. Nakamura, T. Wakabayashi, K. Nakazawa, T. Boku, H. Wada, and Y. Inagami, “Pseudo Vector Processor for High-speed List Vector Computation with Hiding Memory Access Latency”, in Proceedings of IEEE TENCON’94 (1994) 338-342.

5. 板倉 憲一, 廣野 哲, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「ハイパクロスバ・ネットワークにおける NAS ベンチマークの評価」, 情報処理学会研究報告 94-HPC-52-20 (1994) 119-126.

【平成 7 年度】

1. 朴 泰祐, 曽根 猛, 三島 健, 板倉 憲一, 中村 宏, 中澤 喜三郎, 「ハイパクロスバ網における適応ルーチングの導入とその評価」, 電子情報通信学会論文誌 第 J78-D-I 巻, 第 2 号 (1995) 108-117.

2. 板倉 憲一, 服部 正樹, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「超並列計算機 CP-PACS における NAS-PB の仮想評価」, 情報処理学会研究報告 95-HPC-55-7 (1995) 49-56.

3. 廣野 哲, 上野 幸樹, 中村 宏, 朴 泰祐, 中澤 喜三郎, 「マルチバンクメモリ上における擬似ベクトルプロセッサ PVP-SW の性能評価」, 情報処理学会研究報告 95-ARC-111-2 (1995) 9-16.

4. 曽根 猛, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「ハイパクロスバ・ネットワークにおける virtual channel の動的選択による適応ルーティング」, 並列処理シンポジウム JSPP’95 論文集 (1995) 249-256.

5. 朴 泰祐, 板倉 憲一, 曽根 猛, 三島健, 中澤 喜三郎, 中村 宏, 「ハイパクロスバ・ネットワークにおける転送性能向上のための手法とその評価」, 情報処理学会論文誌 第 36 巻, 第 7 号 (1995) 1610-1618.

6. K. Itakura, M. Hattori, T. Boku, H. Nakamura, and K. Nakazawa, “Preliminary evaluation of NAS Parallel Benchmarks on CP-PACS”, Proceedings of PERMEAN’95 (International Workshop on Performance Evaluation and Analysis), (1995) 68-77.

7. 服部 正樹, 板倉 憲一, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「CP-PACS パイロットモデルにおける NAS 並列ベンチマークの評価」, 情報処理学会研究報告 95-HPC-57-8 (1995) 43-48.

【平成 8 年度】

1. T. Yoshié, “Benchmark test of CP-PACS for lattice QCD”, in Proceedings of International Workshop “QCD on Massively Parallel Computers” (Yamagata, Japan, March 16-19, 1995) Prog. Theor. Phys. Suppl. 122 (1996) 8-24.

2. 曽根 猛, 服部 正樹, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「CP-PACS パイロットモデルにおける LINPACK ベンチマークの高速化」, 情報処理学会研究報告 96-ARC-117-15 (1996) 83-88.

3. 添野 元秀, 山下 義行, 中田 育男, 「スライドウインドウを考慮したレジスタ割り付け」, 情報処理学会第 52 回全国大会講演論文集 (5), (1996) 13-14.

4. 栗元 貴文, 山下 義行, 中田 育男, 「データ並列型言語 NCX における通信コストと仮想プロセッサマッピング」, 情報処理学会第 52 回全国大会講演論文集 (6), (1996) 117-118.

5. 曽根 猛, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「ハイパクロスバ・ネットワークにおける Virtual Channel の動的選択による適応ルーティング」, 情報処理学会論文誌 第 37 巻, 第 7 号, (1996) 1409-1418.

6. 松原 正純, 服部 正樹, 板倉 憲一, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「超並列計算機 CP-PACS における PVM の実装」, 情報処理学会研究会報告 96-ARC-119-3 (1996) 13-18.

7. 田井 秀樹, 山下 義行, 中田 育男, 「データ並列言語 NCX の分散メモリ MIMD 並列計算機用コンパイラ」, 情報処理学会第 53 回全国大会講演論文集 (1), (1996) 333-334.

8. 廣野哲, 中村宏, 朴泰祐, 中澤喜三郎, 「擬似ベクトルプロセッサにおける高速リストベクトル処理」, 情報処理学会論文誌 第 37 巻, 第 10 号, (1996) 1850-1858.

9. 板倉 憲一, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「超並列計算機 CP-PACS の CG ベンチマークによる性能評価」, ハイパフォーマンス コンピューティング 63-6, (1996) 31-36.

10. 村上 祥基, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「VHDL によるハイパクロスバ網用ルータチップの設計」, 情報処理学会研究会報告 96-ARC-121-3/96-DA-82-3 (1996) 17-24.

11. L. S. Yang, H. Machidori, T. Shirakawa, "BEM and BEM with SOR on the parallel computer QCDPAX", Engineering Analysis with Boundary Elements, Vol.18 (1996) 231-237.

12. Y. Abei, K. Itakura, T. Boku, H. Nakamura, K. Nakazawa, "Performance Improvement for Matrix Calculation on CP-PACS Node Processor", in Proceedings of HPC Asia'97 (1997) 672-677.

13. K. Itakura, T. Boku, H. Nakamura, K. Nakazawa, "Performance evaluation of CP-PACS on CG benchmark", in Proceedings of HPC Asia'97 (1997) 678-683.

14. 板倉 憲一, 松原 正純, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「超並列計算機 CP-PACS における NPB Kernel CG の評価」, 並列処理シンポジウム JSPP'97 論文集 (1997) 5-12.

15. M. Guo, Y. Yamashita, I. Nakata, "An Efficient Data Distribution Technique for Distributed Memory Parallel Computers", 並列処理シンポジウム JSPP'97 論文集 (1997) 189-196.

16. 星野 健一, 中田 育男, 山下 義行, 「疎行列演算プログラムの並列化の一方法」, 情報処理学会第 54 回全国大会講演論文集 (分冊 1), (1997) 65-66.

17. 過 敏意, 山下 義行, 中田 育男, "An Efficient Data Distribution Technique for Distributed Memory Parallel Computers", 情報処理学会第 54 回全国大会講演論文集 (分冊 1), (1997) 339-340.

18. 田井 秀樹, 松本 正義, 酒寄 保隆, 山下 義行, 中田 育男, 「超並列計算機用 NCX 言語処理系の試作」, 情報処理学会第 54 回全国大会講演論文集 (分冊 1), (1997) 345-346.

19. 糸賀 裕弥, 山下 義行, 中田 育男, 「条件分岐を考慮したループ並列化の 1 手法」, 情報処理学会第 54 回全国大会講演論文集 (分冊 1), (1997) 351-352.

20. 萩川 友宏, 添野 元秀, 山下 義行, 中田 育男,「レジスタ割付からみたスライドウインドアーキテクチャの優位性について」, 情報処理学会第 55 回全国大会論文集分冊 1, (1997) 16-17.

21. 板倉 憲一, 安部井 嘉人, 松原 正純, 朴 泰祐, 中村 宏, 中澤 喜三郎,「超並列計算機 CP-PACS の基本性能評価」, 計算機アーキテクチャ 123-4, (1997) 19-24.

22. 板倉 憲一, 安部井 嘉人, 松原 正純, 朴 泰祐, 中村 宏, 中澤 喜三郎,「超並列計算機 CP-PACS における分子動力学シミュレーション」, ハイパフォーマンス コンピューティング 66-2, (1997) 7-12.

23. 板倉 憲一, 安部井 嘉人, 松原 正純, 朴 泰祐, 中村 宏, 中澤 喜三郎,「超並列計算機 CP-PACS のネットワーク性能評価」, ハイパフォーマンス コンピューティング 67-10, (1997) 55-60.

23.4 CP-PACS による計算物理学

【平成 8 年度】

1. CP-PACS Collaboration (S. Aoki *et al.,*), "CP-PACS results for quenched QCD spectrum with the Wilson action", in Proceedings of International Workshop "Lattice QCD on Parallel Computers" (Tsukuba, March 10–15, 1997), Nucl. Phys. (Proc. Suppl.), to appear.

2. CP-PACS Collaboration (S. Aoki *et al.,*), "Full QCD results from CP-PACS", in Proceedings of International Workshop "Lattice QCD on Parallel Computers" (Tsukuba, March 10–15, 1997), Nucl. Phys. (Proc. Suppl.), to appear.

3. CP-PACS Collaboration (S. Aoki *et al.,*), "CP-PACS results for the quenched light hadron spectrum", in Proceedings of Lattice '97 (Edinburgh, Scotland, 22-26 July, 1997), Nucl. Phys. (Proc. Suppl.), to appear.

4. CP-PACS Collaboration (S. Aoki *et al.,*), "Hadron spectroscopy and static quark potential in full QCD: A comparison of improved actions on the CP-PACS", in Proceedings of Lattice '97 (Edinburgh, Scotland, 22-26 July, 1997), Nucl. Phys. (Proc. Suppl.), to appear.

5. M. Umemura, "Radiation Hydrodynamics on a Massively Parallel Supercomputer", in Proceedings of "Numerical Astrophysics Using Supercomputers", (1996) 2-7.

6. T. Nakamoto, M. Umemura, H. Susa, "Photoionization of a Clumpy Universe", in Proceedings of International Symposium on Supercomputing "New Horizon of Computational Science" (1997), to appear.

24 口頭発表

24.1 CP-PACS 計画全般

【平成 5 年度】

1. 岩崎 洋一, 「超並列計算機 CP-PACS」, 日本物理学会第４９回年会（福岡工業大学, 福岡, 3 月, 1994）.

【平成 6 年度】

1. 吉江 友照, 「CP-PACS による計算物理学 計算性能評価」, 筑波大学計算物理学研究センター研究会「並列計算計算機と計算物理学」（筑波大学, つくば, 1994 年 12 月 6–7 日）.
2. 青木 慎也, 「CP-PACS による計算物理学 素粒子物理研究計画」, 筑波大学計算物理学研究センター研究会「並列計算計算機と計算物理学」（筑波大学, つくば, １９９４年１２月）.
3. 吉江 友照,「超並列計算機 CP-PACS」, スーパーコンピュータ高度利用のための研究会（高エネルギー物理学研究所, つくば, 1995 年 3 月 1–3 日）.

【平成 7 年度】

1. 岩崎 洋一,「CP-PACS project」, 筑波大学計算物理学研究センター研究会「CCP Workshop on Lattice Field Theories '96」（筑波大学, つくば, １９９６年３月５–７日）.
2. 金谷 和至, 「CP-PACS プロジェクト」, KEK ワークショップ「並列計算機による計算物理の進展」（高エネルギー物理学研究所, つくば, １９９６年２月８–９日）.

【平成 8 年度】

1. 岩崎洋一, “The CP-PACS Project and Computational Physics”, International Symposium on Parallel Computing in Engineering and Science (Science and Technology Agency, Tokyo, Japan, Jan. 27 -28, 1997).

24.2 CP-PACS による計算物理学

【平成 7 年度】

1. 吉江 友照, 「Light Hadron Spectrum 」, 筑波大学計算物理学研究センター研究会 「CCP Workshop on Lattice Field Theories '96」（筑波大学, つくば, １９９６年３月５–７日）.
2. 梅村 雅之, 「輻射流体力学による宇宙物理学 I」, 数値シミュレーションによる天文学シンポジウム（１９９６年１月）.
3. 釣部 通, 梅村 雅之, 「輻射流体力学による宇宙物理学 II:散乱の取り扱い」, 数値シミュレーションによる天文学シンポジウム（１９９６年１月）.
4. 中本 泰史, 「輻射流体力学による宇宙物理学 III : 星・原始惑星系円盤の形成」, 数値シミュレーションによる天文学シンポジウム（１９９６年１月）.

【平成 8 年度】

1. 吉江友照, 「Lattice QCD ハドロンスペクトロスコピー計算」, 筑波大学計算物理学研究センター研究会「並列計算機による物理学」（筑波大学, つくば, １９９７年３月２６–２７日）.

2. 金谷和至, 「CP-PACS による full QCD 計算」, 筑波大学計算物理学研究センター研究会「並列計算機による物理学」(筑波大学, つくば, １９９７年３月２６–２７日).

3. 吉江友照, 「Quenched QCD Hadron Spectrum on CP-PACS」, 日本物理学会年会（名城大学, 名古屋, １９９７年３月）.

4. 金児隆志, 「Numerical simulations in lattice QCD with improved actions on CP-PACS」, 日本物理学会年会（名城大学, 名古屋, １９９７年３月）.

5. 梅村 雅之, 「宇宙流体力学による可視化」, 情報処理学会ハイパフォーマンスコンピューティング研究会, (１９９６年５月).

6. 大越智 幸司, 中本 泰史, 「２次元軸対称の輻射流体力学」, 日本天文学会（１９９６年１０月）.

7. 中本 泰史, 「差分法による３次元輻射流体力学計算」, 日本天文学会（１９９６年１０月）.

8. 梅村 雅之, 「輻射流体力学の展望」, 数値シミュレーションによる天文学シンポジウム II（１９９６年１２月）.

9. 大越智 幸司, 「２次元軸対称の輻射流体力学」, 数値シミュレーションによる天文学シンポジウム II（１９９６年１２月）.

10. 中本 泰史, 「３次元輻射流体力学計算」, 数値シミュレーションによる天文学シンポジウム II（１９９６年１２月）.

11. 大越智 幸司, 中本 泰史, 「多次元輻射流体力学計算コードの開発」, 日本天文学会（１９９７年３月）.

12. 梅村 雅之, 「CP-PACS による計算物理学：輻射流体力学計算」, 筑波大学計算物理学研究センター研究会「並列計算機による物理学」, (１９９７年３月).

25 関連論文及び学会発表等

25.1 計算機工学

【平成 5 年度】

1. 中田 育男, 山下 義行, 「再帰的下向き構文解析における演算子順位構文解析」, 情報処理学会論文誌, 第 34 巻, 第 2 号, (1993) 239-245.

2. 佐々 政孝, 石塚 治志, 中田 育男, 「1 パス型属性文法に基づくコンパイラ生成系 Rie」, コンピュータソフトウェア, 第 10 巻, 第 3 号, (1993) 20-36.

【平成 6 年度】

1. H. Morimoto, K. Yamazaki, H. Nakamura, T. Boku, K. Nakazawa, “Superscalar Processor Design with Hardware Description Language AIDL”, in Proceedings of 2nd Asia Pacific Conference on Hardware Description Language (1994) 51-58.

【平成 7 年度】

1. 山下 義行, 中田育男, 「時相属性文法によるグラフィカル・ユーザーインターフェースの記述」, コンピュータソフトウェア, vol.12, no.2, (1995) 76-93.

2. M. Sassa, H. Ishizuka, I. Nakata, "Rie, a Compiler Generator Based on a One-pass-type Attribute Grammar", Software-Practice and Experience, Vol.25(3), (1995) 229-250.

3. 中井 央, 山下 義行, 中田 育男, 「インクリメンタルな LR 構文解析の一方式の提案とその評価」, 情報処理学会第 50 回全国大会講演論文集 (5), (1995) 41-42.

4. 中川 裕之, 金谷 英信, 星野 秀之, 中田 育男, 山下 義行, 「拡張 1 パス型属性文法によるコンパイラ生成系の実現」, 情報処理学会論文誌, 第 36 巻, 第 4 号, (1995) 902-912.

5. 中田 育男, 山下 義行, 「正規右辺属性文法の一提案」, 情報処理学会論文誌, 第 36 巻, 第 6 号, (1995) 1415-1421.

6. T. Boku, T. Harada, T. Sone, H. Nakamura, and K. Nakazawa, "INSPIRE : A general purpose network simulator generating system for massively parallel processors", in Proceedings of PERMEAN'95 (International Workshop on Performance Evaluation and Analysis), (1995) 24-33.

7. 原田 智紀, 曽根 猛, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「並列処理用ネットワークのための性能評価用シミュレータ生成系 INSPIRE」, 情報処理学会研究報告 ARC-95-113-9 (1995) 65-72.

8. 山崎 浩太, 森本 貴之, 中村 宏, 朴 泰祐, 中澤 喜三郎, 「SFL を用いた superscalar 及び VLIW プロセッサの設計とその比較」, 第 7 回パルテノン研究会予稿集 (1995) 11-18.

9. 下平 文彦, 小林 覚, 白川 友紀, 田村 義保, 「統計データ解析の並列処理」, 計算機統計学, 第 8 巻, 第 1 号 (1995) 15-25.

10. I. Nakata, "Generation of Pattern-Matching Algorithms by Extended Regular Expressions", Advances in Software Science and Technology, Vol.5, (Dec.,1995) 1-9.

【平成 8 年度】

1. 中井 央, 山下 義行, 中田 育男, 「インクリメンタルな LR 構文解析の一方式の提案とその評価」, 情報処理学会論文誌, 第 37 巻, 第 3 号, (1996) 371-383.

2. 平見 知久, 山下 義行, 中田 育男, 「1 パス型属性文法におけるバックパッチ処理の自動生成」, 情報処理学会第 52 回全国大会講演論文集 (5), (1996) 7-8.

3. 川口 さおり, 山下 義行, 中田 育男, 「演算子順位を利用した再帰的下向き構文解析器生成系の実現」, 情報処理学会第 52 回全国大会講演論文集 (5), (1996) 9-10.

4. A. Murata, T. Boku, H. Amano, "The MDX (Multi-Dimensional X'bar): A Class of Networks for Large Scale Multiprocessors", IEICE Trans. on Information and Systems, Vol.E79-D, (1996) 1116-1123.

5. 三島 正博, 板倉 憲一, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「並列計算機の仮想性能評価システム VIPPES」, 情報処理学会研究報告 96-HPC-62-5 (1996) 27-32.

6. 中田育男, 「初期のコンパイラの開発と最適化」, 夏のプログラミングシンポジウム「コンピューティングの歴史」報告集, (1996) 55-62.

7. 秋川 友宏, 添野 元秀, 山下 義行, 中田 育男, 「スライドウインドウを考慮したレジスタ割付」, 日本ソフトウェア科学会第 13 回全国大会講演論文集, (1996) 201-204.

8. 中井 央, 佐々政孝, 山下 義行, 中田 育男, 「LR 属性文法に基づいたインクリメンタルな属性評価」, 日本ソフトウェア科学会第 13 回全国大会講演論文集, (1996) 289-292.

9. 森本 貴之, 斉藤 一志, 中村 宏, 朴 泰祐, 中澤 喜三郎, 「方式レベル記述言語 AIDL を用いた高性能プロセッサ設計支援」, 情報処理学会研究報告 96-ARC-121-8 (1996) 57-64.

10. A. Murata, T. Boku, T. Harada, H. Amano, "The MDX (Multi-Dimensional X'bar): A class of networks for large scale multiprocessors", in Proceedings of 9th ISCA International Conference on Parallel and Distributed Computign System (PDCS96), (1996) 296-303.

11. 中井 央, 佐々 政孝, 山下 義行, 中田 育男, 「LR 属性文法に基づいたインクリメンタルな属性評価」, 情報処理学会論文誌, 第 37 巻, 第 12 号, (1996) 2254-2265.

12. T. Boku, M. Mishima, K. Itakura, H. Nakamura, K. Nakazawa, "VIPPES: A performance pre-evaluation system for parallel processors", HPCN Europe'96 (Brussels, Bergium, Apr. 1996).

13. T. Morimoto, K. Saito, H. Nakamura, T. Boku, K. Nakazawa, "Advanced Processor Design Using Hardware Description Language AIDL", in Proceedings of Asia and South Pacific Design Automation Conference 1997 (ASP-DAC'97), (1997) 387-390.

14. 山下 義行, 「一般相対論的 4 次元時空における幾何モデリングの汎用的手法」, 情報処理学会第 54 回全国大会講演論文集 (分冊 4), (1997) 221-222.

15. 服部 正樹, 松原 正純, 板倉 憲一, 朴 泰祐, 「超並列計算機 CP-PACS における分子動力学法シミュレーション」, 情報処理学会研究報告 97-HPC-66-2 (1997) 7-12.

16. 松原 正純, 板倉 憲一, 朴 泰祐, 中村 宏, 中澤 喜三郎, 「超並列計算機 CP-PACS のネットワーク性能評価」, 情報処理学会研究報告 97-HPC-67-10 (1997) 55-60.

17. P. R. Panda, H. Nakamura, N. D. Dutt, A. Nicolau, "A Data Alignment Technique for Improving Cache Performance", in Proceedings of International Conference on Computer Design (ICCD '97), (1997), to appear.

18. 過 敏意, 山下 義行, 中田 育男, "Optimal Implementation of Multi-dimensional Array Redistribution", 日本ソフトウェア科学会第 14 回全国大会講演論文集, (1997) 65-68.

19. 立堀 道昭, 千葉 滋, 中田 育男, 「Java 言語のための新たな自己反映機構の提案」, 日本ソフトウェア科学会第 14 回全国大会講演論文集, (1997) 201-204.

20. 山下 義行, 「構造主導による属性文法の自動変換」, 日本ソフトウェア科学会第 14 回全国大会講演論文集, (1997) 569-572.

21. 中井 央, 佐々政孝, 山下 義行, 中田 育男, 「解析木を用いないインクリメンタルな属性評価」, 日本ソフトウェア科学会第 14 回全国大会講演論文集, (1997) 637-640.

22. 亀山 裕亮, 中井 央, 山下 義行, 中田 育男, 「属性文法に基づいたインクリメンタルな Pascal-S コンパイラ」, 情報処理学会第 55 回全国大会講演論文集 (分冊 1), (1997) 281-282.

25.2 計算物理学

【平成 4 年度】

1. Y. Iwasaki, K. Kanaya, S. Sakai, and T. Yoshié, "Quark confinement and number of flavors", in Proceedings of Lattice '91 (Tsukuba, Japan, 5-9 Nov., 1991), Nucl. Phys. B (Proc. Suppl.) 26 (1992) 311-313.

2. K. Kanaya, Y. Iwasaki, T. Yoshié, T. Hoshino, T. Shirakawa, Y. Oyanagi, S. Ichii, and T. Kawai, "Deconfining transition of SU(3) gauge theory on $N_t = 6$ lattices", in Proceedings of Lattice '91 (Tsukuba, Japan, 5-9 Nov., 1991), Nucl. Phys. B (Proc. Suppl.) 26 (1992) 302-304.

3. T. Yoshié, Y. Iwasaki, K. Kanaya, S. Sakai,T. Hoshino, T. Shirakawa, and Y. Oyanagi, "Quenched hadron spectrum on a $24^3 \times 54$ lattice", in Proceedings of Lattice '91 (Tsukuba, Japan, 5-9 Nov., 1991), Nucl. Phys. B (Proc. Suppl.) 26 (1992) 281-283.

4. Y. Iwasaki, K. Kanaya, T. Yoshié, T. Hoshino, T. Shirakawa, Y. Oyanagi, S. Ichii, and T. Kawai, "Finite temperature phase transition of SU(3) gauge theory on $N_t = 4$ and 6 lattices", Phys. Rev. D 46, (1992) 4657-4667.

【平成 5 年度】

1. Y. Iwasaki, K. Kanaya, S. Sakai, and T. Yoshié, "Quark confinement and number of flavors in strong coupling lattice QCD", Phys. Rev. Lett. 69 (1992) 21-24.

2. K. Kanaya, "QCD calculations on QCDPAX", in Proceedings of "Large Scale Computational Physics on Massively Parallel Computers" (Jülich, Germany, 14-16 June, 1993), Intnl. J. Mod. Phys. C 4, (1993) 1221-1232.

3. Y. Iwasaki, K. Kanaya, S. Sakai, and T. Yoshié, "Quark confinement in multi-flavor quantum chromodynamics", in Proceedings of Lattice '92 (Amsterdam, The Netherlands, 15-19 Sept., 1992), Nucl. Phys. B (Proc. Suppl.) 30 (1993) 327-330.

4. Y. Iwasaki, K. Kanaya, S. Sakai, T. Yoshié, T. Hoshino, T. Shirakawa, and Y. Oyanagi, "Contamination of excited states in quenched QCD hadron propagators", in Proceedings of Lattice '92 (Amsterdam, The Netherlands, 15-19 Sept., 1992), Nucl. Phys. B (Proc. Suppl.) 30 (1993) 397-400.

【平成6年度】

1. Y. Iwasaki, K. Kanaya, S. Sakai and T. Yoshié, "Quantum chromodynamics with various number of flavors", in Proceedings of Lattice '93 (Dallas, USA, 12-16 Oct., 1993), Nucl. Phys. B (Proc. Suppl.) 34 (1994) 314-316.

2. Y. Iwasaki, K. Kanaya, S. Sakai, T. Yoshié, T. Hoshino, and T. Shirakawa, "High statistics calculations of quenched QCD spectrum using various quark sources", in Proceedings of Lattice '93 (Dallas, USA, 12-16 Oct., 1993), Nucl. Phys. B (Proc. Suppl.) 34 (1994) 354-356.

3. Y. Iwasaki, K. Kanaya, Leo Käkkänen, K. Rummukainen and T. Yoshié, "Interface tension in quenched QCD", Phys. Rev. D49 (1994) 3540-3545.

4. S. Aoki and Y. Kikukawa, "Fermion mass in the Wilson-Yukawa approach for chiral Yukawa theory", Int. J. of Mod. Phys. A9 (1994) 4565-4580.

5. S. Aoki, H. Hirose and Y. Kikukawa, "Charged fermion states in the quenched U(1) chiral Wilson-Yukawa model", Int. J. of Mod. Phys. A9 (1994) 4129-4148.

6. Y. Aoki and K. Kanaya, "Interface tension in SU(3) lattice gauge theory at finite temperatures on an $N_t = 2$ lattice", Phys. Rev. D50 (1994) 6921-6930.

7. Y. Kuramashi, M. Fukugita, H. Mino, M. Okawa and A. Ukawa, "η' meson mass in lattice QCD", Phys. Rev. Lett. 72 (1994) 3448-3451.

8. Y. Kuramashi, M. Fukugita, H. Mino, M. Okawa and A. Ukawa, "An Exploratory Study of Nucleon-Nucleon Scattering Length in Lattice QCD", Phys. Rev. Lett. 73 (1994) 2176-2179.

9. Y. Kuramashi, M. Fukugita, H. Mino, M. Okawa and A. Ukawa, "Lattice QCD calculation of hadron scattering lengths", in Proceedings of Lattice '93 (Dallas, USA, 12-16 Oct., 1993), Nucl. Phys. B (Proc. Suppl.) 34 (1994) 117-122.

10. M. Umemura, J. Fukue, "Cosmological Spherical Accretion via External Radiation Drag", Publ. Astron. Soc. Japan, 46 (1994) 567-574.

11. T. Tsuribe, J. Fukue, M. Umemura, "Unsteady Disk Accretion via External Radiation Drag", Publ. Astron. Soc. Japan, 46 (1994) 579-603.

12. J. Fukue, M. Umemura, "Accretion Disks driven by External Radiation Drag - β Disks", in Proceedings of "The New Horison of X-Ray Astronomy", Universal Academy Press, (1994) 617-618

13. T. Nakamoto, Y. Nakagawa, "Formation, Early Evolution, and Gravitational Stability of Protoplanetary Disks", Astrophysical Journal, 421 (1994) 640-650.

14. T. Nakamoto, "Growth of Protoplanetary Disks around Young Stellar Objects", in Proceedings of "The 27th ISAS Lunar and Planetary Symposium" (1994) 91-94.

【平成7年度】

1. Y. Iwasaki, "Phase diagram of QCD at finite temperatures with Wilson fermions", in Proceedings of Lattice '94 (Bielefeld, Germany, 27 Sept.-10 Oct., 1994), Nucl. Phys. B (Proc. Suppl.) 42 (1995) 96-102.

2. Y. Iwasaki, K. Kanaya, S. Kaya, S. Sakai, and T. Yoshié, "Nature of the finite temperture transition in QCD with strange quark", in Proceedings of Lattice '94 (Bielefeld, Germany, 27 Sept.-10 Oct., 1994), Nucl. Phys. B (Proc. Suppl.) 42 (1995) 499-501.

3. Y. Iwasaki, K. Kanaya, S. Sakai, and T. Yoshié, "Finite temperature transition in two flavor QCD with renormalization group improved action", in Proceedings of Lattice '94 (Bielefeld, Germany, 27 Sept.-10 Oct., 1994), Nucl. Phys. B (Proc. Suppl.) 42 (1995) 502-504.

4. K. Kanaya and S. Kaya, "Critical exponents of a three dimensional O(4) spin model", Phys. Rev. D51 (1995) 2404-2410.

5. K. Kanaya, "Deconfining chiral transition in QCD on the lattice", in Proceedings of "From Hadronic Matter to Quark Matter: Evolving View of Hadronic Matter" (Kyoto, Japan, Oct. 30-Nov. 1, 1994), Prog. Theor. Phys. Suppl. 120 (1995) 25-36.

6. M. Fukugita, Y. Kuramashi, H. Mino, M. Okawa and A. Ukawa, "Hadron Scattering Lengths in Lattice QCD", Phys. Rev. D 52 (1995) 3003-3021.

7. M. Fukugita, Y. Kuramashi, M. Okawa and A. Ukawa, "Lattice QCD Solution to the U(1) Problem", Phys. Rev. D51 (1995) 3952-3954.

8. T. Tsuribe, M. Umemura, J. Fukue, "Self-Similar Spherical Accretion via External Radiation Drag", Publ. Astron. Soc. Japan, 47 (1995) 73-79.

9. A. Takahashi, J. Fukue, K. Sanbuichi, M. Umemura, "Dynamical Stability of Cosmological Accretion Disks Embedded in External Radiation Fields", Publ. Astron. Soc. Japan, 47 (1995) 425-428.

10. J. Fukue, M. Umemura, "Accretion Disks driven by External Radiation Drag around Central Luminous Sources", Publ. Astron. Soc. Japan, 47 (1995) 429-437.

11. T. Nakamoto, Y. Nakagawa, "Growth of Protoplanetary Disks around Young Stellar Objects", Astrophysical Journal, 445 (1995) 330-336.

12. 福江 純, 梅村 雅之, 「宇宙の巨大ブラックホール」, 日本物理学会誌「butsuri」, 50 (1995) 562-565.

【平成8年度】

1. S. Aoki, "On the Phase structure of QCD with Wilson fermions", in Proceedings of Japan-German Seminar "QCD on Massively Parallel Computers" (Yamagata, Japan, March 16-19, 1995), Prog. Theor. Phys. Supplement 122 (1996) 179-186.

2. S. Aoki, A. Ukawa and T. Umemura, "Fate of the critical line and chiral transition in finite temperature lattice QCD with the Wilson quark action", in Proceedings of Lattice '95 (Melbourne, Australia, 11-15 July, 1995), Nucl. Phys. B (Proc. Suppl.) 47 (1996) 511.

3. Y. Iwasaki, K. Kanaya, S. Kaya, S. Sakai, and T. Yoshié, "Finite temperature QCD with Wilson quarks: A study with a renormalization group improved action", in Proceedings of Lattice '95 (Melbourne, Australia, 11-15 July, 1995), Nucl. Phys. B (Proc. Suppl.) 47 (1996) 515-518.

4. K. Kanaya, "QCD phase transition with two flavors of Wilson quarks using a RG improved action", in Proceedings of Japan-German Seminar "QCD on Massively Parallel Computers" (Yamagata, Japan, March 16-19, 1995), Prog. Theor. Phys. Suppl. 122 (1996) 115-122.

5. K. Kanaya, "Finite temperature phase transition in QCD with strange quark: study with Wilson fermions on the lattice", in Proceedings of International RCNP Workshop on Color Confinement and Hadrons "Confinement '95" (Osaka, Japan, 22-24 March, 1995), World Scientific, (1996) 119-126.

6. K. Kanaya, "Finite temperature QCD on the lattice", in Proceedings of Lattice '95 (Melbourne, Australia, 11-15 July, 1995), Nucl. Phys. B (Proc. Suppl.) 47 (1996) 144-159.

7. Y. Iwasaki, K. Kanaya, S. Sakai, and T. Yoshié, "Chiral phase transition in lattice QCD with Wilson quarks", Z. Phys. C71 (1996) 337-341.

8. Y. Iwasaki, K. Kanaya, S. Kaya, S. Sakai, and T. Yoshié, "QCD phase transition with strange quark in Wilson formalism for fermions", Z. Phys. C71 (1996) 343-346.

9. Y. Iwasaki, K. Kanaya, T. Yoshié, T. Hoshino, T. Shirakawa, Y. Oyanagi, S. Ichii, and T. Kawai, "Hadron masses and decay constants with Wilson quarks at $\beta = 5.85$ and 6.0", Phys. Rev. D53 (1996) 6443-6464.

10. Y. Iwasaki, K. Kanaya, S. Kaya, S. Sakai, and T. Yoshié, "Finite temperature transition in lattice QCD with Wilson quarks — chiral transitions and the influence of the strange quark —", Phys. Rev. D54 (1996) 7010-7031.

11. S. Sasaki, M. Umemura, "Reionization of the Universe due to Early-Formed Massive Black Holes", Astrophysical Journal, 462 (1996) 104-109.

12. T. Tsuribe, M. Umemura, "Radiation-Hydrodynamical Evolution of the Cosmological Accretion Disks", Cosmological Constant and the Evolution of the Universe, (1996) 307-308.

13. M. Umemura, "Cosmological Accretion Disks driven by Radiation Drag", in "Basic Physics of Accretion Disk", Gordon and Breach Science Publishers, (1996) 203-208.

14. T. Tsuribe, M. Umemura, "Rapid Mass Accretion by Background Radiation Force", in "Basic Physics of Accretion Disk", Gordon and Breach Science Publishers, (1996) 215-218.

15. T. Tsuribe, M. Umemura, "Radiation-Hydrodynamical Evolution of the Cosmological Accretion Disks", in "Numerical Astrophysics Using Supercomputers", (1996) 8-10.

16. T. Nakamoto, "Radiation Hydrodynamics for Star and Protoplanetary Disk Formation", in "Numerical Astrophysics Using Supercomputers", (1996) 11-13.

17. S. Mineshige, M. Umemura, "Self-Similar, Self-Gravitating Viscous Disk", Astrophysical Journal Letters, 469 (1996) L49-L51.

18. F. Nakamura, T. Hanawa, T. Matsumoto, "Dynamical Collapse of Magnetized Molecular Cloud Cores; Formation of Self-similarly Contracting Disks", in "Low Mass Star Formation from Infall to Outflow", (1996) 232-234.

19. F. Nakamura, T. Hanawa, "Nonaxisymmetric Evolution of Dynamically Contracting Disks and Formation of Binary Stars", in "Low Mass Star Formation from Infall to Outflow", (1996) 235-237.

20. T. Hanawa, T. Matsumoto, F. Nakamura, "Gravitational Collapse of a Rotating Cloud to form star and disk system", in "Low Mass Star Formation from Infall to Outflow", (1996) 212-214.

21. Y. Iwasaki, K. Kanaya, S. Kaya, and T. Yoshié, "Scaling of chiral order parameter in two-flavor QCD", Phys. Rev. Lett. 78 (1997) 179-182.

22. Y. Iwasaki, K. Kanaya, T. Kaneko, and T. Yoshié, "Scaling in SU(3) pure gauge theory with a renormalization group improved action", Phys. Rev. D56 (1997) 151-160.

23. S. Aoki, A. Ukawa and T. Umemura, "Phase structure of lattice QCD with Wilson quark action", Phys. Rev. Lett. 76 (1996) 873-876.

24. S.Aoki and K. Nagai, "Domain-wall fermions with U(1) dynamical gauge fields", Phys. Rev. D53 (1996) 5058.

25. S. Aoki and K. Nagai, "Chiral zero modes on the domain-wall model in 4+1 dimensions", Phys. Rev. D56 (1997) 1121-1130.

26. S. Aoki, M. Doui, T. Hatsuda and Y. Kuramashi, "Tensor Charge of the Nucleon in Lattice QCD", Phys. Rev. D56 (1997) 433-436

27. S. Aoki, K. Nagai and S. Zenkin, "Domain wall fermions with Majorana couplings", hep-lat/9705001.

28. Y. Aoki, "Four-dimensional Simulation of the Hot Electroweak Phase Transition with the SU(2) Gauge-Higgs Model", Phys. Rev. D, to appear.

29. Y. Iwasaki, K. Kanaya, T. Kaneko, and T. Yoshié, "Scaling of the critical temperature and quark potential with a renormalization group improved SU(3) gauge action", in Proceedings of Lattice '96 (St. Louis, USA, 4-8 June, 1996), Nucl. Phys. B (Proc. Suppl.) 53 (1997) 429-431.

30. Y. Iwasaki, K. Kanaya, S. Kaya, S. Sakai, and T. Yoshié, "Phase structure of QCD for general number of flavors", in Proceedings of Lattice '96 (St. Louis, USA, 4-8 June, 1996), Nucl. Phys. B (Proc. Suppl.) 53 (1997) 449-455.

31. K. Kanaya, "Simulations of the finite temperature QCD phase transition on the lattice", in Proceedings of International Symposium "Origin of Matter and Evolution of Galaxies in the Universe" (Jan. 18-20, 1996, Atami, Japan), (1997) 343-353.

32. A. Ukawa, "Finite-temperature QCD on the lattice", in Proceedings of Lattice '96 (St. Louis, USA, 4-8 June, 1996), Nucl. Phys. B (Proc. Suppl.) 53 (1997) 106-119.

33. S. Aoki, T. Kaneda, A.Ukawa and T. Umemura, "Finite-temperature phase structure of lattice QCD with the Wilson quark action for two and four flavors", in Proceedings of Lattice '96 (St. Louis, USA, 4-8 June, 1996), Nucl. Phys. B (Proc. Suppl.) 53 (1997) 438-441.

34. S. Aoki and K. Nagai, "Domain wall fermions with U(1) dynamical gauge fields in (4+1)-dimensions", in Proceedings of Lattice '96 (St. Louis, USA, 4-8 June, 1996), Nucl. Phys. B (Porc. Suppl.) 53 (1997) 635.

35. Y. Aoki, "Four-dimensional Simulation of the Hot Electroweak Phase Transition with the SU(2) Gauge-Higgs Model", in Proceedings of Lattice '96 (St. Louis, USA, 4-8 June, 1996), Nucl. Phys. B (Proc. Suppl.) 53 (1997) 609-611.

36. K. Kanaya, "Order of the Finite Temperature QCD Phase Transition on the Lattice", in Proceedings of International Conference on Physics and Astrophysics of Quark-Gluon Plasma "ICPA-QGP '97" (Jaipur, India, March 17-21, 1997), to appear.

37. M. Umemura, J. Fukue, S. Mineshige, "Radiative Avalanche: Starburst Induced Fuelling to AGNs", Astrophysical Journal Letters, 479 (1997) L97-L100.

38. S. Mineshige, M. Umemura, "Self-Similar Collapse of Self-Gravitating Viscous Disk", Astrophysical Journal, 480 (1997) 167-172.

39. E. L. Turner, M. Umemura, "Very Strong Microlensing of Distant Luminous Stars by Relic Massive Black Holes", Astrophysical Journal, 483 (1997) 603-607.

40. T. Tsuribe, M. Umemura, "Angular Momentum Transport in Early-formed Objects by Cosmic Background Radiation: Radiation-Hydrodynamical Approach", Astrophysical Journal, 486 (1997) 48-59.

41. S. Mineshige, K. Nakayama, M. Umemura, "Self-Similar Viscous Collapse of a Self-Gravitating, Polytropic Gas Disk", Publ. Astron. Soc. Japan, 49 (1997) 439-443.

42. F. Nakamura, M. Umemura, "Collapse and Fragmentation of Filamentary Primordial Gas Clouds due to H_2 Cooling", Numerical Astrophysics Using Supercomputers II (1997) 80-84.

43. T. Matsumoto, T. Hanawa, F. Nakamura, "Gravitational Contraction of Rotating Clouds: Formation of Self-similarly Collapsing Disks", Astrophysical Journal, (1997), to appear.

44. F. Nakamura, T. Hanawa, "Nonaxisymmetric Evolution of Dynamically Contracting Disks and Its Implication for Binary Formation", Astrophysical Journal, (1997), to appear.

25.3 口頭発表

【平成 6 年度】
【国際学会】

1. 金谷 和至, "Finite temperature deconfining chiral transition in QCD with Wilson quarks for $N_F = 2$, 3, and 2+1 ", European Research Conference "Complex Systems in Subatomic Physics: Thermodynamics of Quarks and Hadrons" (Seeheim, Germany, 19-25 Sept., 1994).

【国内学会】

1. 青木 慎也, 「Determination of the Strong Coupling Constant from Lattice QCD」, 場の理論研究会（京都, 12 月, 1994）.

2. 金谷 和至, 「QCD with various number of flavors」, 日本物理学会第 49 回年会（福岡工業大学，福岡, 3 月, 1994）.

3. 青木 慎也, 「素粒子物理学における計算物理」, 日本物理学会第４９回年会, 福岡工業大学, 福岡, 3, 1994 .

4. 榧 昌吾, 「FLV=3,degenerate quark の有限温度相転移」, 日本物理学会分科会, 山形大学, 山形, 10, 1994.

5. 金谷 和至, 「QCDPAX による素粒子物理学」, 筑波大学計算物理学研究センター研究会「並列計算計算機と計算物理学」（筑波大学, つくば, 1994 年 12 月 6 – 7 日）.

6. 梅村 雅之, E. L. Turner, 「Exteme Microlensing due to Cosmic Massive Black Holes」, 日本天文学会（１９９４年５月）.

7. 釣部 通, 梅村 雅之, 福江 純, 「Non-Steady Gas Accretion due to Compton Drag」, 日本天文学会（１９９４年５月）.

8. 三分一 清隆, 福江 純, 嶺重 慎, 梅村 雅之, 「中心天体の輻射場中の降着円盤の時間変化」, 日本天文学会（１９９４年５月）.

9. 高橋 敦, 三分一 清隆, 福江 純, 梅村 雅之, 「外部輻射抵抗を受けた降着円盤の安定性」, 日本天文学会（１９９４年５月）.

10. 中本 泰史, 「原始星周囲の円盤の質量増加について」, 日本天文学会（１９９４年５月）.

11. 横野 安則, 中本 泰史, 観山 正見, 千葉 博嗣, 「T Tauri 型星のフラットスペクトルを説明するモデル」, 日本天文学会（１９９４年５月）.

12. 梅村 雅之, 「宇宙論的流体力学と輻射輸送問題」, 天文・天体物理若手夏の学校（１９９４年７月）.

13. 梅村 雅之, 「論文雑誌受理のシステム」, 天文・天体物理若手夏の学校（１９９４年７月）.

14. 中本 泰史, 「理論は天文観測と隕石研究の接着剤となれるか？— 原始惑星系円盤を中心とした星・惑星系形成の理論的研究 —」, 惑星科学夏の学校（１９９４年７月）.

15. 中本 泰史, 「Growth of Protoplanetary Disks around Young Stellar Objects」, 宇宙科学研究所 第 27 回 Lunar and Planetary Symposium （１９９４年８月）.

16. 釣部 通, 梅村 雅之, 中本 泰史, 「宇宙論的輻射輸送と大質量ブラックホールの形成」, 日本天文学会（１９９４年１０月）.

17. 三分一 清隆, 福江 純, 嶺重 慎, 梅村 雅之, 「中心天体の輻射場中の降着円盤 2」, 日本天文学会（１９９４年１０月）.

18. 福江 純, 梅村 雅之, 釣部 通, 「ベータ降着円盤を伝わる波」, 日本天文学会（１９９４年１０月）.

19. 釣部 通, 福江 純, 梅村 雅之, 「外部輻射抵抗を考慮したガスの重力不安定性」, 日本天文学会（１９９４年１０月）.

20. 村上 泉, 梅村 雅之, 「Dynamical Heating and Metallic Gas Supply by Elliptical Galaxies in Rich Clusters」, 日本天文学会（１９９４年１０月）.

21. 中本 泰史「原始星周囲の原始惑星系円盤の非線型成長」, 日本惑星科学会（１９９４年１０月）.

22. 中本 泰史, 観山 正見「T Tauri 型星のフラットスペクトルを説明するモデル (II)」, 日本天文学会（１９９４年１０月）.

23. 梅村 雅之, 「宇宙流体力学」, 計算物理学研究センター研究会（１９９４年１２月）.

24. 梅村 雅之, 「H_0 と λ, クェーサーとは何か」, 理論天文学懇談会シンポシウム (１９９４年１２月)

25. 梅村 雅之, 「筑波大 CP-PACS のアーキテクチャとサイエンス」, 理論天文学懇談会シンポシウム (１９９４年１２月)

26. 梅村 雅之, 「銀河形成論の方法論」, 計算物理学研究センターワークショップ (１９９５年３月)

27. 釣部 通, 梅村 雅之, 中本 泰史, 「宇宙論的輻射輸送と天体の形成」, 日本天文学会（１９９５年３月）.

28. 中本 泰史「原始星周囲の物質分布」, 日本惑星科学会（１９９５年３月）.

【平成７年度】
【国際学会】

1. 中本 泰史, 中川 義次, “Growth of Protoplanetary Disks around Protostars” Gordon Research Conference “Origins of Solar Systems” (NH, USA, June, 1995).

【国内学会】

1. 宇川 彰, 「格子上の量子色力学」, 京都大学基礎物理学研究所統合記念シンポジウム「基礎物理学の現状」（京都大学数理解析研究所, 京都, １９９５年１２月）.
2. 宇川 彰, 「有限温度格子量子色力学」, 筑波大学計算物理学研究センター研究会「CCP Workshop on Lattice Field Theories '96」（筑波大学, つくば, １９９６年３月５–７日）.
3. 宇川 彰, 「有限温度格子量子色力学の現状」, 日本物理学会年会 素粒子論分科シンポジウム講演（金沢大学, 金沢, １９９６年３月）.
4. 宇川 彰, 「Phase structure of Lattice QCD with the Wilson Quark Action」, 日本物理学会年会（金沢大学, 金沢, １９９６年３月）.
5. 青木 保道, 「Nature of the Hot Electroweak Phase Transition in the SU(2) Higgs Model」CCP Workshop on Lattice Field Theories '96 （筑波大学, つくば, １９９６年３月）.
6. 梶 昌吾, 「繰り込み群によって改良された作用を用いた有限温度格子 QCD のカイラル相転移」, 日本物理学会年会（中部大学, １９９５年９月）.
7. 金児 隆志, 「くりこみ群により改良された作用を用いた SU(3) 格子ゲージ系の有限温度相転移」, 日本物理学会分科会（中部大学, 名古屋, １９９５年９月）.
8. 梶 昌吾, 「Finite temperature transition with the Wilson quark action and a RG improved gauge action」, 筑波大学計算物理学研究センター研究会 「CCP Workshop on Lattice Field Theories '96」（筑波大学, つくば, １９９６年３月５–７日）.
9. 梶 昌吾, 「Finite temperature QCD with various number of flavors」, 日本物理学会年会（金沢大学, １９９６年３月）.
10. 吉江友照, 「格子ゲージ理論と非摂動論的 QCD」, 大阪大学核物理学研究センター夏の学校 '96（大阪大学, 大阪, １９９６年８月５–９日）.
11. 金児 隆志, 「Finite temperature phase tansition of SU(3) pure gauge systemwith R.G. improved action 」, 筑波大学計算物理学研究センター研究会 「CCP Workshop on Lattice Field Theories '96」（筑波大学, つくば, １９９６年３月５–７日）.
12. 金児 隆志, 「くりこみ群により改良された作用を用いた SU(3) 格子ゲージ系の有限温度相転移２」, 日本物理学会年会（金沢大, 金沢, １９９５年３月）.
13. 長井 敬一, 「Domain-wall fermion with dynamical gauge fields」, 日本物理学会年会（金沢大学 , 金沢, １９９６年３月）.
14. 金田 知之, 「Quenched lattice QCD における Wilson quark action の critical line と有限温度相転移」, 日本物理学会年会（金沢大学, 金沢, １９９６年３月）.

15. 梅村 雅之, 「QSO/AGN の形成」（レビュー）, 京都大学基礎物理学研究所短期研究会「銀河形成」（１９９５年6月）.

16. 釣部 通, 梅村 雅之, 中本 泰史, 「宇宙論的輻射輸送：層状ガスと背景輻射場の相互作用」, 日本天文学会（１９９５年１０月）.

17. 吉田 宏, 梅村 雅之, 吉井 譲, 「銀河の光度関数と宇宙論的パラメータ」, 日本天文学会（１９９５年１０月）.

18. 中村 文隆, 花輪 知幸, 「動的に収縮する円盤の分裂と連星系の形成」, 日本天文学会（１９９５年１０月）.

19. 中村 文隆, 花輪 知幸, 「磁気雲の重力収縮について」, 日本天文学会（１９９５年１０月）.

20. 松本 倫明, 花輪 知幸, 中村 文隆, 「回転しているガス円盤の相似的収縮」, 日本天文学会（１９９５年１０月）.

21. 中村 文隆, 「磁気雲の重力収縮と動的に収縮する円盤の進化」（レビュー）, 滞在型研究会「星の形成」（１９９５年１０月）.

22. 中村 文隆, 「動的に収縮する円盤の分裂」, 重点領域研究「星間物質とその進化」大研究会 (１９９５年１０月）.

23. 中本 泰史, 「冷たいガス円盤における自己重力不安定 (I) 計算法」, 日本惑星科学会（１９９５年１１月）.

24. 梅村 雅之, 「QSO/AGN 物理過程の概観」, 筑波大学計算物理学研究センター研究会「滞在型ワークショップ・銀河形成」, （１９９５年１２月）.

25. 釣部 通, 梅村 雅之, 「QSO/AGN の物理」, 筑波大学計算物理学研究センター研究会「滞在型ワークショップ・銀河形成」, （１９９５年１２月）.

26. 梅村 雅之, 「銀河形成の物理」（レビュー）, 第８回理論天文学懇談会シンポジウム「宇宙の階層構造」（１９９５年１２月）.

27. 釣部 通, 梅村 雅之, 「Radiation-Hydrodynamical Evolution of the Cosmological Accretion Disks」, 第８回理論天文学懇談会シンポジウム「宇宙の階層構造」（１９９５年１２月）.

28. 中本 泰史, 「星周円盤の形成と円盤の重力不安定性」, 筑波大学計算物理学研究センター研究会「滞在型ワークショップ・銀河形成」, （１９９５年１２月）.

29. 中本 泰史, 「原始惑星系円盤の形成と円盤の自己重力不安定性」, 総研 B 研究会「星の誕生」（１９９５年１２月）.

30. 中本 泰史, 「ガス円盤における重力不安定 — 局所近似・非線型計算」, 第８回理論天文学懇談会シンポジウム「宇宙の階層構造」（１９９５年１２月）.

31. 中村 文隆, 「動的に収縮するガス円盤の分裂と連星系の形成」, 第 8 回理論天文学懇談会シンポジウム「宇宙の階層構造」(１９９５年１２月）.

32. 中村 文隆, 「回転しながら動的に収縮するガス円盤の分裂と連星系の形成」, NRO ワークショップ「星の誕生」大研究会 (１９９５年１２月).

33. 中村 文隆, 「動的に収縮するガス円盤の分裂と連星系の形成」, 数値シミュレーションによる天文学シンポジウム (１９９６年１月).

34. 梅村 雅之, 「輻射流体力学の定式化と線形波動」(レビュー), 京都大学基礎物理学研究所短期研究会「宇宙における構造形成と輻射輸送」, (１９９６年２月).

35. 嶺重 慎, 釣部 通, 梅村 雅之, 「ハイ z クェーサーディスクの構造」, 日本天文学会 (１９９６年３月).

36. 中本 泰史, 「ガス円盤における非軸対称自己重力不安定 — 局所近似・非線型計算 —」, 日本天文学会 (１９９６年３月).

37. 松本倫明, 花輪知幸, 中村文隆, 「収縮する回転ガス円盤の振動とそのメカニズム」, 日本天文学会 (１９９６年３月).

38. 中村 文隆, 花輪 知幸, 「回転しながら動的に収縮する円盤の進化と連星系の形成」, 日本天文学会 (１９９６年３月).

【平成 8 年度】
【国際学会】

1. 宇川 彰, “Thermodynamics and the Lattice - Overall View - ”, RHIC’96 Theory Workshop (BNL, USA, July 8-19, 1996)

2. 宇川 彰, “Recent Progress in Lattice QCD”, 1996 UK Theory Institute (Durham, August 25-September 14, 1996).

【国内学会】

1. 金谷和至, 「Phase structure of QCD on the lattice – T and Nf-dependence of the vacuum –」, 基礎物理学研究所研究会「非摂動論的量子色力学–ＱＣＤの真空構造–」(京都大学, 京都, １９９６年１２月１７–２０日).

2. 青木慎也, 「Phase structure of Lattice QCD with Wilson fermion at finite temperature」, 筑波大学計算物理学研究センター研究会「並列計算機による物理学」(筑波大学, つくば, １９９７年３月２６–２７日).

3. 石塚成人, 「 Lattice calculation of $K \to \pi\pi$ decay amplitude 」, 筑波大学計算物理学研究センター研究会「並列計算機による物理学」(筑波大学, つくば, １９９７年３月２６–２７日).

4. 榧 昌吾, 「Phase structure of QCD with general number of flavors」, 日本物理学会 (佐賀大学, 佐賀, １９９６年１０月６–９日).

5. 榧 昌吾, 「$N_F = 2$ 及び 3 の有限温度ＱＣＤの相構造」, 日本物理学会年会 (名城大学, 名古屋, １９９７年３月).

6. 金児隆志, 「Finite temperature transition in the SU(3) pure gauge theory with a renormalization group improved action」, 日本物理学会（佐賀大学, 佐賀, １９９６年１０月６–９日）.

7. 青木保道, 「格子 SU(2) ヒッグス模型における電弱理論の高温相転移」, 日本物理学会 （佐賀大学, 佐賀, １９９６年１０月６–９日）.

8. 青木保道 , 「Hot Electroweak Phase Transition with the SU(2) Gauge-Higgs model on the Lattice」, 第 5 回 KEK 理論研究会「CP 非保存とその起源」（KEK, つくば, １９９７年３月６–７日）.

9. 中村 文隆, 梅村 雅之, 「原始銀河雲における星形成: H_2 冷却による円筒状ガス雲の動的収縮と分裂」, 日本天文学会（１９９６年１０月）.

10. 釣部 通, 嶺重 慎, 梅村 雅之, 「乱流粘性と輻射抵抗による角運動量輸送と宇宙論的 AGN 形成」, 日本天文学会（１９９６年１０月）.

11. 中山 薫二, 嶺重 慎, 梅村 雅之, 「自己重力円盤の自己重力崩壊」, 日本天文学会（１９９６年１０月）.

12. 釣部 通, 犬塚 修一郎, 佐野 孝好, 永井 智哉, 増永 浩彦, 「Mesh 内部不連続点を考慮した新しい移流の計算法」, 日本天文学会（１９９６年１０月）.

13. 佐野 孝好, 犬塚 修一郎, 釣部 通, 永井 智哉, 増永 浩彦, 「磁気流体力学的数値計算法について」, 日本天文学会（１９９６年１０月）.

14. 中村 文隆, 梅村 雅之, 水素分子の冷却によるフィラメント状ガス雲の動的収縮と分裂」, 数値シミュレーションによる天文学シンポジウム II（１９９６年１２月）.

15. 釣部 通, 「輻射粘性の数値計算」, 数値シミュレーションによる天文学シンポジウム II（１９９６年１２月）.

16. 梅村 雅之, 「LMSA による銀河／クェーサー形成の探究」, 第９回理論天文学懇談会シンポジウム「大型観測装置時代における理論天文学の役割」（１９９６年１２月）.

17. 梅村 雅之, 「活動銀河核とスターバーストの物理的関連：輻射性なだれモデル」, 京都大学基礎物理学研究所短期研究会「銀河形成の物理過程」,（１９９７年２月）.

18. 釣部 通, 「粘性降着による原始活動銀河中心核種の成長過程」, 京都大学基礎物理学研究所短期研究会「銀河形成の物理過程」,（１９９７年２月）.

19. 梅村 雅之, 「銀河中心核への質量降着の新しいメカニズム：輻射性なだれ」, 筑波大学計算物理学研究センター研究会「クェーサー活動性と銀河形成の物理的関連」,（１９９７年３月）.

20. 中本 泰史, 「ダストガス雲からの星形成」, 筑波大学計算物理学研究センター研究会「クェーサー活動性と銀河形成の物理的関連」,（１９９７年３月）.

21. 梅村 雅之, 福江 純, 嶺重 慎, 「Radiative Avalanche: 銀河中心核への質量降着の新しいメカニズム」, 日本天文学会（１９９７年３月）.

22. 嶺重 慎, 梅村 雅之, 福江 純, 「Radiative Avalanche II: Induced Accretion Flow」, 日本天文学会（１９９７年３月）.

23. 川口 俊弘, 嶺重 慎, 梅村 雅之, E. L. Turner, 「活動銀河核の可視光ゆらぎの起源」, 日本天文学会（１９９７年３月）.

24. 米原 厚憲, 嶺重 慎, 福江 純, 梅村 雅之, E. L. Turner, 「Microlens Mapping of AGN Disk」, 日本天文学会（１９９７年３月）.

25. 福江 純, 梅村 雅之, 嶺重 慎, 「Radiative Avalanche Driven by Spherical Starbursts」, 日本天文学会（１９９７年３月）.

26. 中村 文隆, 「動的に収縮するフィラメント状ガス雲の自己相似解」, 日本天文学会（１９９７年３月）.

27. 中本 泰史，大越智 幸司, 「多次元輻射流体力学計算」, 地球惑星科学関連学会合同大会（１９９７年３月）.

28. 釣部 通, 「粘性降着による原始星成長と原紙惑星系円盤の形成」, 日本天文学会（１９９７年３月）.

29. 釣部 通, 「回転原始星の成長と星周円盤の形成を表す自己相似解」, NRO ワークショップ「星・惑星系形成研究の新展開」, （１９９７年３月）.

26 著書

1. 中澤 喜三郎, 「計算機アーキテクチャと構成方式」, 朝倉書店 (1995).

第Ⅴ部

新聞・雑誌等の記事