Vol.2009-ARC-184 No.26
2009/8/5

超並列準汎用計算機GRAPE-DR による 重力多体問題シミュレーションおよびLU 分解

小池 邦昭[†1] 藤野 健[†2] 福重 俊幸[†3]
台坂 博[†4] 菅原 豊[†2] 稲葉 真理[†2]
平木 敬[†2] 牧野 淳一郎[†5]

超並列準汎用計算機 GRAPE-DR は 1 チップに 512 個の演算要素を搭載した SIMD アクセラレータを PC に接続し，これを並列に接続したクラスタシステムである．これは従来の重力多体問題専用計算機 GRAPE の発展形として使用できるように構想されたものであるが，アクセラレータ部分が専用ハードウェアパイプラインではなくプログラム可能な演算器を搭載することでより広い応用が可能であることが大きな特徴である．本論文ではアクセラレータ部で動作する重力相互作用計算と行列積計算ルーチンを実装し，1 ノードでの性能評価をおこなった．現在それぞれのライブラリについて最適化を行っている．現状では重力相互作用計算では 362.6GFlops($N = 262144$)，行列積計算では 635.1GFlops($M = N = 32768, K = 2048$) の演算性能となった．これを用いて High Performance LINPACK(HPL) の加速を行い，演算性能値は 284.3GFlops($N = 34816, NB = 2048$) となった．

Gravitational N-body simulation and LU decomposition with the multi purpose accelerator GRAPE-DR

The multi purpose computer GRAPE-DR is a cluster of PCs computer with the custom-made SIMD accelerator. It was designed as a for the successor of the special purpose computers for N-body simulation, “GRAPE” systems. The GRAPE-DR chip consists of 512 simplified processor cores. It can be used for wider applications than the range of applications of previous GRAPE systems.

We implemented libraries for gravitational N-body simulations and matrix-matrix multiplications(DGEMM) on the GRAPE-DR system, we report the performance of these libraries as of the end of June 2009. we evaluated performances of these libraries on a single PC with a GRAPE-DR accelerator card. The mesared performance was 362.6GFlops($N = 262144$) for the gravitational N-body simulations. and it has achieved and 635.1GFlops for the matrix-matrix multiplications($M = N = 32768, K = 2048$). The performance of LU-decompositions was 284.3GFlops on High Performance LINPACK ($N = 34816, NB = 2048$).

1. はじめに

本論文では，GRAPE-DR[6] への重力多体問題シミュレーションプログラムと，LU 分解アルゴリズムの実装と，実現した性能について述べる．GRAPE-DR は，東京大学の平木を代表とする振興調整費プロジェクト「分散共有型研究データ利用基盤の整備」(2004-2008) で開発された，科学技術計算向けのワンチップ超並列プロセッサである．

GRAPE-DR は，それまで東京大学で開発されてきた重力多体問題専用計算機 GRAPE (GRAvity PipE[10]) の発展として構想されたものである．GRAPE の基本的アーキテクチャを図 1 に示す．汎用のホスト計算機 に，粒子間重力を計算する専用プロセッサである GRAPE プロセッサを接続した構成である．このアーキテクチャには，

- 重力計算専用プロセッサは，専用化することで非常に高い演算性能を実現できる．
- シミュレーションプログラムの殆どの部分はホスト計算機側で実行されるため，プログラム開発，チューニングは容易である．特に，コンパイラ，OS 等を用意する必要がない．

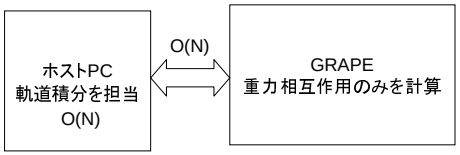

図 1 GRAPE システム

†1 総合研究大学院大学
The Graduate University for Advanced Studies
†2 東京大学
The University of Tokyo
†3 K&F Computing Research Co.
†4 一橋大学
Hitotsubashi University
†5 国立天文台
National Astronomical Observatory of Japan

といった特色がある．このため，1999 年にプロセッサ LSI を開発した GRAPE-6[5] において，30Gflops 相当の演算性能を 10W の消費電力で実現できた．2009 年の最新の CPU や GPGPU に比べるとピーク性能では若干劣るが，電力あたり性能では依然として優位に立っている．

しかし，LSI 開発の初期費用の高騰のため，重力専用のプロセッサを開発するのは現実的ではなくなってきている．一方，近年の汎用プロセッサでは総トランジスタ数に占める浮動小数点演算器部分の割合が下がり続けているため，非常に単純だがプログラム可能な要素プロセッサを多数集積することで，原理的には汎用プロセッサよりも高い性能と電力あたり性能を実現できる．GRAPE-DR (Greatly Reduced Array of Processor Element with Data Reduction) は，このような考え方に基づいて開発されたものである．

GRAPE-DR プロセッサチップは，TSMC 90nm プロセスで製造された約 18mm 角の LSI であり，2 億強のトランジスタで 512 個の要素プロセッサを集積する．要素プロセッサ 1 つはスループット 1 の倍精度浮動小数点加減算器と，スループット 1/2 の倍精度浮動小数点乗算器，32 ワードのレジスタファイル，256 ワードのメモリ，整数 ALU 等からなり，倍精度浮動小数点乗算器は単精度乗算ならスループット 1 で実行できる．

本論文では，この GRAPE-DR プロセッサの実アプリケーションでの性能を，特に重要なアプリケーションである重力多体計算と，密行列の LU 分解の場合について実測した結果を報告する．なお，特に LU 分解については性能向上の余地が多くある，予備的な結果である．本論文の構成は以下の通りである．2 節では GRAPE-DR のアーキテクチャについてまとめる．3 節では重力多体計算での性能について述べる．星団・惑星形成のシミュレーションで広く使われている独立時間刻み法[4] の場合に，粒子数が大きく $N = 262144$ のところでは 362.6 Gflops (動作クロック が 333 MHz の時) が得られ，これは通信を無視した漸近性能の 84.8% である．また，漸近性能の半分が得られる粒子数は $N = 16384$ となった．GRAPE-6 チップを 32 個搭載したプロセッサボード 1 ボードと比較するとピーク性能は 40%ほどになってしまっているが，漸近性能の半分が得られる粒子数は約 1/6 となっており，実用性能は大きく向上した．4 節では，LU 分解，すなわちいわゆる LINPACK ベンチマークの性能について，1 ボードで測定した結果を報告する．LU 分解の演算の殆どを占める行列乗算ルーチン DGEMM については，行列サイズがホスト計算機の主記憶リミットに近い極限で 635.1 Gflops と，理論ピーク性能の 79.3% が得られた．LU 分解については現在のところ 284.3 Gflops と理論ピーク性能の 35.53% に留まっている．5 節では，それぞれのアプリケーションについて，効率を制限している要因と，性能向上の可能性について議論する．

2. GRAPE-DR

2.1 GRAPE-DR アーキテクチャ

古典的な SIMD アーキテクチャに基づく計算機としては Illiac-IV[1]，Goodyear MPP[9]，CM-1/2[3] 等がある．これらの古典的な SIMD 型の計算機は並列度に比例したデータ供給のバンド幅が必要になる．このことは，これらの計算機が開発された 1960 年代から 80 年代前半までは問題ではなかったが，90 年代になって LSI の集積度があがってくると大きな問題となった．多数の演算器を 1 チップに集積しても，外付のメモリと十分なバンド幅で接続することが不可能になったからである．また，プロセッサ間ネットワークについても，メモリとの接続と同様に，多次元メッシュ結合やハイパーキューブといった，古典的 SIMD 計算機では一般的なネットワークを複数チップで構成するのは現実的ではなくなった．ベクトルプロセッサも，高いメモリバンド幅を実現することは難しい，という古典的 SIMD 計算機と同じ理由で開発が困難になった．

GRAPE-DR では，この困難を解消するために，要素プロセッサのメモリはオンチップにはいる程度の小規模なものに制限し，またプロセッサ間ネットワークも複数チップに容易に拡張できる階層的なツリー型に限った．さらに，外付メモリとの接続バンド幅も，なるべく低く抑えた．これにより，512 個ものプロセッサを 1 チップに集積し，512Gflops (500MHz 動作の場合，単精度) の高いピーク性能を実現したにもかかわらず，消費電力，実装コストの双方を抑えることに成功した．特に消費電力は 60W 程度であり，90nm プロセスで製造されている GRAPE-DR が 2009 年時点で最新の 40nm プロセスで製造された GPGPU に比較しても倍精度演算では高い絶対性能と電力あたり性能をもっている．

もちろん，メモリサイズ，チップ間ネットワーク，外付メモリバンド幅に上のような制限があることは，ある程度応用範囲を制限することになる．GRAPE-DR で高い効率が実現できるのは演算量が入出力のデータ量に比べて大きくなるものである．そのようなアプリケーションは，以下の 3 タイプに分類できる．

(1) 相互作用型
(2) 散乱実験型
(3) 密行列乗算型

(1) の相互作用型のアプリケーションは次のような式 (1) での相互作用が計算の大部分を占めるものである．

$$f_i = \sum_{1 \le j \le N, j \neq i} f(x_i, x_j) \quad (1)$$

ここで扱う系は N この要素からなり，x_i, x_j はそれぞれ i, j 番目の要素を表す変数，$f(x_i, x_j)$ は要素 j から要素 i までの相互作用，f_i は要素 i への系全体からの作用である．このような相互作用形に書けるものとしては従来の GRAPE が使用されてきたような重力相互作用の計算 (衝突系，無衝突系用)，流体力学の粒子法の 1 つである Smoothed Particle Hydrodynamics (SPH)，量子化学計算で用いられる二電子積分の計算などがあげられる．計算できる相互作用が重力だけの従来の GRAPE と違って総当りの組について積算する形の任意の計算が加速できるのが GRAPE-DR の特徴となる．(2) の散乱実験型は多数の小規模な問題を多数のサンプルについて解くような応用になる．この種類の応用例としては連星と微惑星遭遇の問題や多次元のモンテカルロ積分などがあげられる．(3) の密行列型のアプリケーションは問題をブロック化した密行列の積

$$C_{ij} = \sum_k A_{ik} B_{kj} \quad (2)$$

に帰着させるものである．このようなアプリケーションとしては行列積が計算時間のほとんどを占める LINPACK などが存在する．これは (1) の相互作用系と似ているがデータの再利用のされ方が少し違う．

基本的に SIMD 動作する多数の単純なプロセッサを 1 チップに集積する，という GRAPE-DR の基本的アーキテクチャは，最近普及のきざしを見せている GPGPU のアーキテクチャに近いが，GRAPE-DR では，外付メモリのバンド幅を低く抑えることで消費電力を削減すると共に開発・実装を容易にし，ネットワークをツリー型にして演算結果のチップ間での縮約を高速に行えるようにしたことで行列乗算等様々なアプリケーションでの実行効率を大きく向上させると共に，アプリケーションからみたプログラミングを容易にしている．

2.2 GRAPE-DR 演算チップ

GRAPE-DR チップ (SING) は 512 個の単純化した演算要素を 1 チップに集積しすべての演算要素が同じタイミングで同一の動作を行う SIMD プロセッサである. それぞれの演算要素は水平型のマイクロコードで命令を投入し，4 個のデータに対してベクトル命令をパイプライン実行する． 512 個の演算要素を 32 個ごとのブロック 16 個に分け，同一のブロック内の演算要素に放送するためのデータを格納する共有メモリ (放送メモリ，BM) を搭載している. この 32 個ごとの演算要素のブロックを放送ブロックと呼ぶ. またチップからの出力は放送ブロック間で共有メモリの内容を縮約するするための演算器がツリー構造のネットワーク経由でなされる．これによりデータを高速に縮約してから出力することができる (結果縮約ネットワーク，RRN). この放送ブロックと結果縮約ネットワークにより相互作用計算や密行列乗算のアプリケーションで添え字 i だけでなく，j(密行列の場合 k) の方向の並列化もホストの通信量を増加させることなく実現できる．

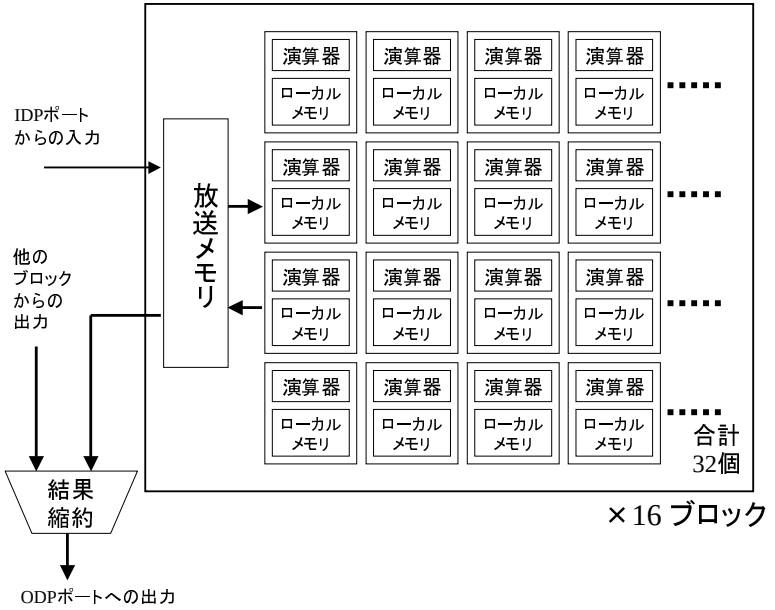

図 2 GRAPE-DR 演算チップ

チップ内の各演算要素は浮動小数点乗算器，浮動小数点加減算器，論理演算器から構成される．扱えるデータフォーマットの基本形は符号 1 ビット，指数部 11 ビット，仮数部 60 ビットの 72 ビットの浮動小数点数である．以下ではこれを 1 長語の浮動小数点数と呼ぶ. また 1 長語の浮動小数点数から仮数部が 24 ビットに短縮された 36bit の浮動小数点数を 1 短語と定義する．このほか 72 ビット，36 ビットの固定小数点数も論理演算器で使用可能である．乗算器は仮数部において 50 ビット入力と 25 ビット入力に対応しており，長語と短語の積を 1 サイクルで行うことが可能である．加減算器は長語と長語の和および差を 1 サイクルで計算する．両方の演算器で長語，短語の型変換は演算命令の一部として実行され

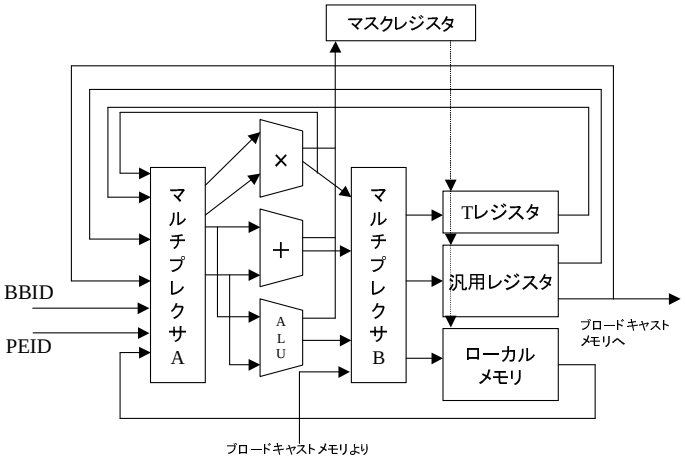

図 3 各演算要素 (PE) の構造

図 4 GRAPE-DR プロセッサボード

る．中間の演算結果を記憶するための領域として，各演算要素に独立に 1 読み出し 1 書き込みのローカルメモリを 256 長語，2 読み出し 1 書き込みの汎用レジスタを 32 長語内蔵する．このほかに共有メモリへのアクセスポートがあり，各演算要素のデータを放送ブロック内の共有メモリに入出力する．

チップ内演算要素 (PE) と外部のデータのやり取りは放送ブロックに搭載されている共有メモリを介して行う．共有メモリはサイクルごとに一語の読み出しまたは書き込みを行う．読み出されたデータは放送ブロック内の全 PE に放送される．書き込みは 1 サイクルには PE1 つのみが行う．また，外部から放送メモリへのアクセスは個別アクセスおよび放送が可能である．また，読み出しは前述の縮約ネットワーク経由となる．縮約ネットワークは浮動小数点加減算器および整数 ALU の全機能をサポートし，また指定した放送ブロック 1 つからの読み出しも可能である．これらを用いて外部から

(1) 全 PE への一斉データ送信
(2) 1 放送ブロック内の PE へのデータ放送
(3) 複数の放送ブロックの同じ PEID を持つ PE へのデータ放送
(4) 個別 PE への書き込み
(5) 個別 PE からの読み出し
(6) 縮約を伴う読み出し

などが実現できる．

2.3 プロセッサボード

GRAPE-DR プロセッサボード (GRAPE-DR Model 1800) は演算プロセッサ (SING プロセッサ)，FPGA で実装された制御プロセッサ，DDR2-SDRAM で実装されたオンボードメモリ (外部メモリ,EM) を 1 セットとした 4 セットを 1 ボード上に搭載して構成される．制御プロセッサは Altera 社製 ArriaGX EP1AGX60EF1152，DDR2-SDRAM は DDR2-533 のメモリを 288MB (32M 長語) 搭載している．また，FPGA 間の縮約を高速に行うための双方向 1GB/s のバンド幅を持つシリアルリンクがリング状に接続されている．

2.4 制御プロセッサ

SING プロセッサ自体は演算に特化した構成を持つため，ホスト側から計算を行わせるには外部に制御回路が必要となる．制御プロセッサはホスト計算機側へのインタフェースをもち，DDR2-SDRAM と SING プロセッサの制御を行い SING プロセッサに計算を行わせる．

制御プロセッサの主な機能は

- ホスト側との PCI-Express インタフェース
- DDR2-SDRAM インタフェース

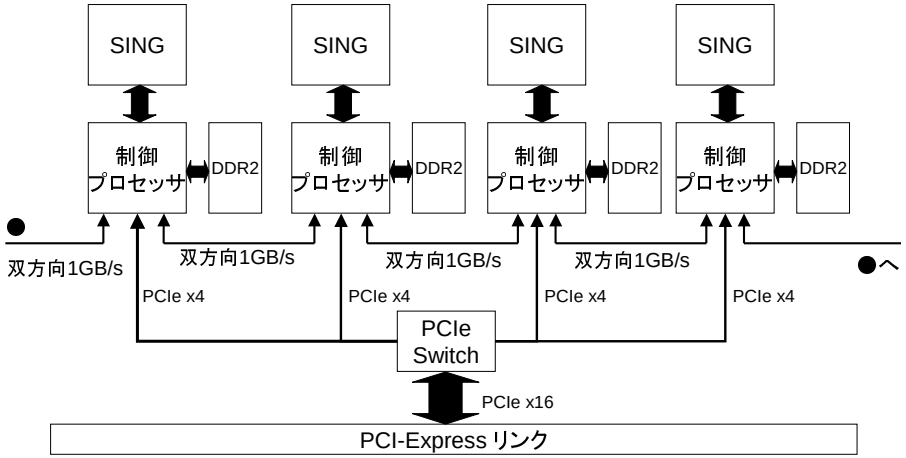

図 5 GRAPE-DR プロセッサボードのブロック図

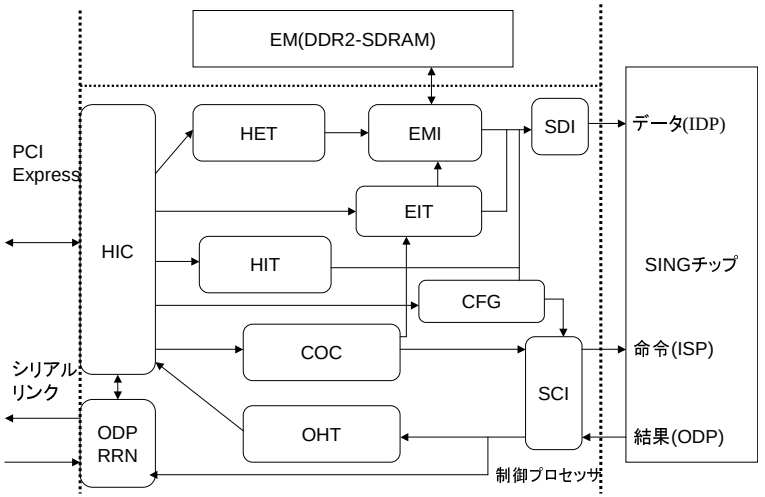

図 6 制御プロセッサのブロック図

- SING プロセッサへの命令転送
- データ形式変換
- 隣接 FPGA との通信を用いた結果縮約

である．これを実現するために制御プロセッサは

- 外部メモリ (EM)-IDP 転送シーケンサ (EIT)
- ホスト-EM 転送回路 (HET)
- コード出力回路 (COC)
- 出力データポート (ODP)-ホストインタフェース転送回路 (OHT)
- EM インタフェース (EMI)
- SING-IDP インタフェース (SDI)
- ホストインタフェース (HIC)
- SING チップインタフェース (SCI)
- SING チップ初期化回路 (CFG)
- チップ間シリアルインタフェース (ODPRRN)

の各モジュールから構成される．図 6 に各ブロックの接続を示す．

EIT はホスト側からの外部メモリから SING プロセッサへのデータ転送指令を受けとり，それをもとにして外部メモリの読み出しアドレスを生成し，EMI に読み取り指令を発行させる．またコード出力回路からの転送指令を受けつけ，外部メモリのデータを SING プロセッサに転送する．HET はホスト側からの 64 ビットの浮動小数点・固定小数点のデータを受け取り SING プロセッサで使用される 72 ビットの浮動小数点・固定小数点形式に変換し，変換したデータを EMI 経由で外部メモリに書き込む．COC はホスト側から SING プロセッサに送り込む PE に対する命令列・データ回収命令を受け取り，ホスト側からの起動命令で計算を実行する．OHT は SING プロセッサからの結果を回収して 72 ビットの形式から 64 ビットの形式に変換する．EMI は HET から出力された外部メモリへの書き込みデータを受け取り外部メモリへの書き込みを行い，EIT から発行された読み込み指令を受け取り外部メモリからのデータ読み込みを行う．SDI および SCI は SING プロセッサと他のモジュールとのデータのやり取りおよびクロック変換を行う．CFG は SING プロセッサがリセット時に必要な設定を行う．ODPRRN はチップ間のシリアルインタフェースを用いて 4 チップから出力されたデータを加算する．

2.5 プログラミングモデル

GRAPE-DR のプログラミングは演算プロセッサと制御プロセッサの動作をアセンブラによって記述することで行われる．現在，GRAPE-DR 用に高級言語でのプログラミングを

可能にしているコンパイラである Sakura-C，LSUMP[7] が存在している．また，OpenMP に類似した記述を可能にするコンパイラ (goosecc) も利用可能である．それらを用いても GRAPE-DR のプログラミングは可能となっている．

ユーザーは GRAPE-DR アセンブリ言語を用いて演算プロセッサ上で計算される初期化・ループコードおよびホスト側からのデータ転送指令を記述する．これをアセンブラは解釈し，アセンブラで記述した処理に対応する C 言語のインタフェースライブラリ関数を自動的に生成する．データの転送回収および計算ループの記述法にはユーザーの利便性を考慮して 2 種類あり，Type-I と Type-II が定義されている．Type-I は相互作用計算に特化した形の記述タイプとなっており，ユーザーはループコードの記述を行い，演算プロセッサへのデータ転送，回収コードは自動的に生成される．Type-II はユーザー側からデータ転送，回収のタイミングを指定することでさらに柔軟な記述が可能である．ユーザーはアセンブラが生成したライブラリ関数をユーザープログラムから呼び出すことで GRAPE-DR 上での転送および計算を行う．

3. 重力多体計算

3.1 計 算 方 法

GRAPE-DR における重力相互作用計算は独立時間刻みを用いたエルミート積分法[4] に対応するため，粒子系の重力加速度 (式 3) および重力加速度の時間微分 (jerk)(式 4) を計算する．

$$r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$$

$$\mathbf{a}_i = \sum_{j \neq i} -Gm_j \frac{\mathbf{r}_i - \mathbf{r}_j}{r_{ij}^3} \tag{3}$$

$$\dot{\mathbf{a}}_i = \sum_{j \neq i} -Gm_j \left[\frac{\mathbf{v}_{ij}}{r_{ij}^3} - \frac{3(\mathbf{v}_{ij} \cdot \mathbf{r}_{ij})}{r_{ij}^5} \right] \tag{4}$$

GRAPE-DR1 チップでの重力計算の演算手順は以下のようになる．

(1) 全 j 粒子を外部メモリに格納する．
(2) i 粒子 128 個をチップに送る．1 つの PE は 4 個の i 粒子を格納し，1 つの放送ブロックの中の 32 個の PE はそれぞれ別々の i 粒子をもつ．16 個の放送ブロックが持つ i 粒子 128 個はすべて同じある．
(3) (4)-(5) を外部メモリに格納した j 粒子がなくなるまで繰り返す．
(4) 外部メモリから j 粒子 16 個を読み出し，放送ブロック 1 個に 1 粒子ずつ配る．

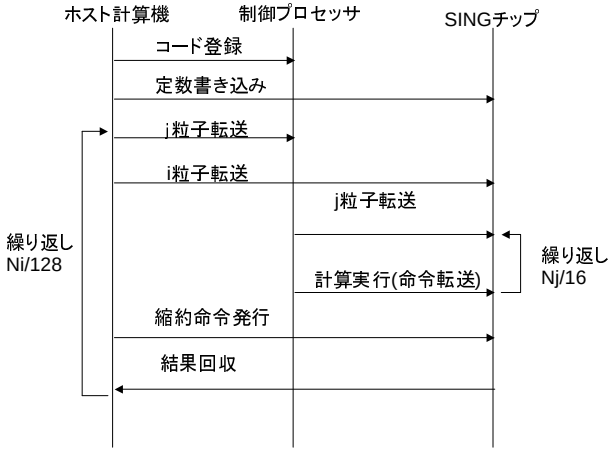

図 7 重力計算時の処理の流れ．Ni は i 粒子数，Nj は j 粒子数を示す．

(5) すべての j 粒子からの力の計算が終了したら，縮約ネットワークを通して i 粒子 128 個の重力を読み出す．
(6) まだ i 粒子が残っているならば，(2) にもどる．

図 7 に計算の流れを示す．実際には (5) の計算とここの繰り返しのための j 粒子の転送 (4) は並行して行われる．

SING チップで実行される計算ループ中では独立時間刻み法に対応するため各 j 粒子の予測子を計算する．時刻 t_0 の重力加速度の 2 回時間微分 $\mathbf{a}_0^{(2)}$，重力加速度の 1 回時間微分 $\dot{\mathbf{a}}$，重力加速度 $\mathbf{a}_0$，速度 $\mathbf{v}_0$，位置 $\mathbf{r}_0$ を用いて時刻 t の予測子は次のように書ける．

$$\Delta t = t - t_0$$

$$\mathbf{r}_p = \frac{\Delta t^4}{24}\mathbf{a}_0^{(2)} + \frac{\Delta t^3}{6}\dot{\mathbf{a}}_0 + \frac{\Delta t^2}{2}\mathbf{a}_0 + \Delta t \mathbf{v}_0 + \mathbf{x}_0 \tag{5}$$

$$\mathbf{v}_p = \frac{\Delta t^3}{6}\mathbf{a}_0^{(2)} + \frac{\Delta t^2}{2}\dot{\mathbf{a}}_0 + \Delta t \mathbf{a}_0 + \mathbf{v}_0 \tag{6}$$

次に予測子を用いて式 3 と式 4 を評価する．このとき，粒子間の距離 r_{ij} を計算する際にはニュートン反復を用いている．

また現在のところ重力計算では制御プロセッサに実装されたチップ間の縮約ネットワークはまだ利用していない．このため 4 チップに j 粒子は分割して転送し，同じ i 粒子への部分力を計算させ，回収した 4 つの部分力をホストで合計している．また，j 粒子転送に関して

も同時に 1 チップにのみ書き込みがされないので j 粒子転送速度が遅くなっている．このため，今回報告する実測性能は縮約ネットワークを利用し，j 粒子をバッファリングした場合に比べて通信オーバーヘッドが大きい予備的なものである．今後，これらについても順次実装する．

3.2 性能評価

4 チッププロセッサボード (GRAPE-DR Model 1800) を用いて重力相互作用の計算を行った．使用した環境は CPU:Intel Core i7(2.67GHz), メモリ:DDR3-1066 12GB, チップセット:Intel X58 である．SING チップは動作周波数 333MHz で動作させている．

粒子数を $N = 2048$ から 262144 まで変化させ，相互作用計算のみの計算性能を測定した．図 8 にその結果を示す．ここで 1 相互作用の演算数は 60 として計算している． この結果，重力相互作用のみの計算では $N = 262144$ のときに 377.0Gflops の値となった．粒子数を無限大にした極限を考え，通信を無視し相互作用計算ループのみでの理論ピーク性能は 426.4GFlops である．ここから 88.4%の効率であることがわかる．またこのときに重力相互作用の i 粒子通信，j 粒子通信，計算および回収の各段階でかかった時間を図 9 に示す．

図 9 より，

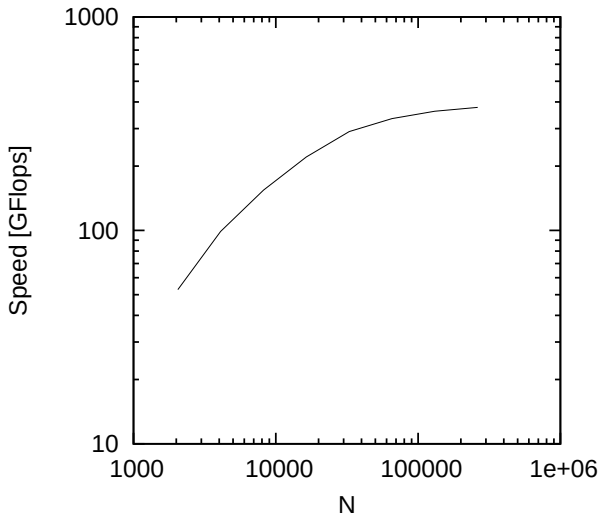

図 8 重力相互作用の計算性能と粒子数の関係

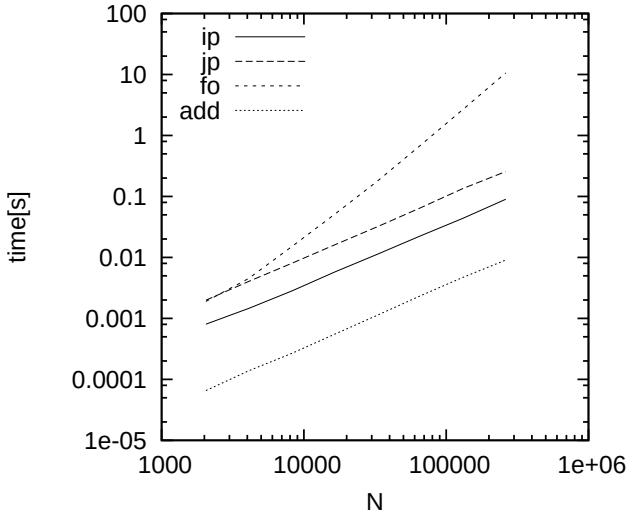

図 9 重力相互作用の各段階における計算時間と粒子数の関係．ip は i 粒子転送にかかった時間，jp は j 粒子転送にかかった時間，fo は計算および結果回収にかかった時間を示す，add はホスト側での部分力の加算にかかった時間である．

$$t_{ip} = aN + b$$
$$t_{jp} = cN + d$$

として，i 粒子と j 粒子の転送にかかるスループットとレイテンシを測定するため得られた転送時間を上記の式に最小二乗法でフィットした．それぞれの係数は $a = 3.42309 \times 10^{-7} \pm 1.93 \times 10^{-9}$， $b = -3.0245 \times 10^{-5} \pm 0.0002065$， $c = 9.83288 \times 10^{-7} \pm 1.007 \times 10^{-8}$， $d = 0.000423242 \pm 0.001078$ となった．これより，i 粒子は 1 秒間に 2.921×10^6 粒子，j 粒子は 1.017×10^6 粒子転送できることがわかる．粒子数が大きい領域で測定したため，レイテンシについてはこの測定からは測定できなかった．現状の i 粒子転送では i 粒子データは 72 バイトであるため i 粒子転送速度は 210.3MB/s となってしまっている．これは現状では同一の i 粒子データを 4 チップ同時に送信しているため PCI Express の帯域を有効に活用できていないためである．今後チップ間のブロードキャストネットワークを使用して，4 チップにそれぞれ別の i 粒子データを送信しチップ間で i 粒子データをやり取りすることで，PCI Express の帯域を有効に活用できるようにする．また j 粒子データは 1 データにつき 160 バ

イトなので j 粒子の転送速度は 162.7MB/s となってしまっている．現状のライブラリでは 1j 粒子は同時に 1 チップにのみ書かれるので PCI Express の 16 リンクのうち 4 リンクしか動作していないことになる．また，ホストインタフェースは Write Combined に対応したダブルバッファを持っている．現在は 1j 粒子ごとにダブルバッファを切り替える動作が入るために実効転送速度が落ちてしまっている．今後はライブラリ側でバッファリングを行い，4 チップ同時に j 粒子を送信しかつ WriteCombined 用バッファの切り替えを最小限にするようにする．また，j 粒子データに使用している重力加速度の 2 回微分 $\mathbf{a}_0^{(2)}$，重力加速度の 1 回時間微分 $\dot{\mathbf{a}}$ の精度を減らすことで，j 粒子データの時間あたりの転送量を増大させる．

また，計算回収の部分に関しては

$$t_{fo} = eN^2 + fN + g \tag{7}$$

として計算ループでの漸近性能と起動オーバヘッドについて求めた．ここの fo にはホスト側での縮約演算は入っていない．図 9 より，$e = 1.525 \times 10^{-10} \pm 8.029 \times 10^{-14}$，$f = 3.725 \times 10^{-7} \pm 2.114 \times 10^{-8}$，$g = 0.00157 \pm 0.0007264$ となった．これより，実測での粒子数が無限大の場合漸近性能は 393.4GFlops となり，現状の重力計算ループの行数から計算される漸近性能 426.4GFlops から 7.7%ほどの低下が見られる．この原因については現在調査中であるがボード上外部メモリ (EM) のリフレッシュ動作により SING チップの命令の実行が一時中断される可能性が考えられる．1 計算ループは 128i 粒子ごとに起動される．この起動にかかる時間とホストへの回収にかかる時間は $128f = 4.768 \times 10^{-5}$s となった．また，粒子回収については回収する 1 粒子データは 64 バイトなので，128i 粒子分回収すると合計で 8192 バイトとなる．この長さデータでのホストへの DMA の速度は 1 チップあたり 541MB/s となるのでデータ回収にかかる時間は $\frac{64\times128}{541\times10^6} = 1.514 \times 10^{-5}[s]$ となる．それを除くと計算起動にかかるオーバーヘッドは 32.5μs になる．現在の実装では 4 チップに計算を起動する際に 4 スレッドで 1 スレッドにつき 1 チップの計算の起動を行うのでスレッドの切り替えコストと同期コストが発生するためそのオーバーヘッドが大きくなってしまっている．今後，4 チップの計算を起動するのを非同期に行い，スレッドの切り替えをしないような実装をすることでの高速化を図る．

次に独立時間刻み法を用いて $N = 2048$ から 262144 までの粒子数の Plummer モデルの運動を $t = 0.0$ から $t = 1.0$ まで計算し，計算性能を測定した．図 10 に結果を示す．使用したソフトニングパラメータ $\epsilon = 0.01$，タイムステップを決定する精度パラメータ $\eta = 0.1$ として計算した．粒子数 $N = 262144$ のところでは実効性能で 362.9GFlops の演算性能となった．

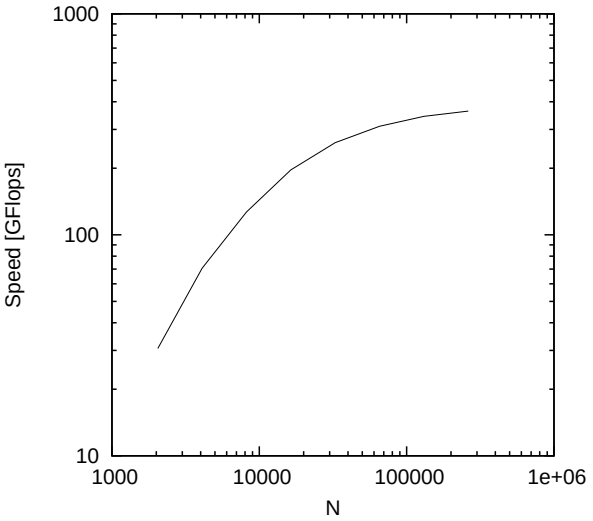

図 10 独立時間刻みで Plummer モデルの運動を計算した場合の計算性能と粒子数の関係

4. LU 分 解

ブロック化した LU 分解では行列積が計算量の大部分を占める．このため，行列積を GRAPE-DR ボード上で加速すればブロック化した LU 分解の加速が可能になる．この節では GRAPE-DR を用いた行列積の計算方法を 4.1，それを用いて行列積の性能評価を 4.2 に示す．これを用いて High Performance LINPACK(HPL)[8] を加速し性能評価をおこなった．結果を 4.3 に示す．

4.1 行列積の計算方法

GRAPE-DR Model 1800 上では 4 チップを用いて，A 行列 256 行 2048 列，B 行列 2048 行 2 列を単位とした行列の積の計算を行う．その際，1 チップでは 256 行 512 列の行列 A と 512 行 2 列の行列 B の積になるように分解する (図 11)．

行列乗算の順序は以下のようになる．

(1) 行列 B 全体を外部メモリに転送する．
(2) A 行列より 256 行をホストからローカルメモリに転送する．これは PE のローカルメモリを利用して入る最大の行列サイズである．PE1 つには 32×8 の行列が入る．

(3)　B 行列 1 行を外部メモリから読み出し，16 分割して放送メモリに格納する．
(4)　B1 行と A32 × 8 の行列・ベクトル積を各 PE で計算する．
(5)　放送ブロック間で和をとりながら，演算結果の 256 × 2 要素を出力する．

実際にはステップ (3)-(5) は同時並行的に進む．さらにステップ (2) についても次のループでの A 行列データを (3)-(5) の計算中にホストから転送し，外部メモリに格納している．現在の実装では次のループでの A 行列を格納するバッファ領域が 1024 長語までのみなので，オーバーラップできる行列のサイズに制限がある．このように以下に示す性能値は最適化の余地のある暫定的な性能値である．

4.2　行列積ルーチンの性能評価

行列積ライブラリを 4 チップボードである GRAPE-DR Model 1800 に実装し，性能の評価を行った．使用したホスト構成は 3.2 で使用したものと同一である．SING の動作周波数は 400MHz で動作させている．

行列行列積 (DGEMM) の行列サイズと演算性能の関係を図 12 に示す．ここで使用した行列サイズは A 行列の列数=B 行列の行数=$K = 2048$ であり，A 行列の行数 (M)=B 行列の列数 (N) を 2048 から 32768 まで変化させ，計算性能を評価した． $M = N = 32768, K = 2048$ のとき DGEMM の実効性能値は 635.1GFlops となった．400MHz 駆動の SING プロセッサの倍精度ピーク性能は 200GFlops となるので DGEMM の実効性能はピーク性能に対して 79.3%になっていることがわかる．

4.3　GRAPE-DR による High Performance LINPACK の加速

High Performance LINPACK(HPL)[8] は並列ブロック LU 分解を行うライブラリとベンチマークプログラムのパッケージである．通信ライブラリに MPI，線形演算ライブリに BLAS を用いている．このためプラットフォームに応じて最適化された BLAS を用意することでさまざまな環境で高い性能を得られるようになっている．特にブロック化 LU 分解の計算の大部分を占める行列のブロック更新 $A - LU$ での計算性能が全体の計算性能に影響するため，BLAS の DGEMM ルーチンをどれだけ高速に実行できるかが最も重要になる．今回の実装ではホスト PC 側の BLAS には GotoBLAS[2] を使用し，DGEMM の部分を行列サイズに応じてホスト PC と GRAPE-DR 上で振り分けて実行する．

GRAPE-DR を用いた HPL では行列積を GRAPE-DR ボードで計算し，BLAS の DGEMM ルーチンを加速する．4 チップの演算チップを搭載したプロセッサボード (GRAPE-DR Model 1800) を用いるので，HPL のブロックサイズは仕様上 2048 に固定される．このように大きなブロックサイズを用いると，ホスト PC 上で計算される

(1)　再帰的パネル分解 (pfact)
(2)　右上部分行列 (U 部分行列) の変形 (DTRSM)

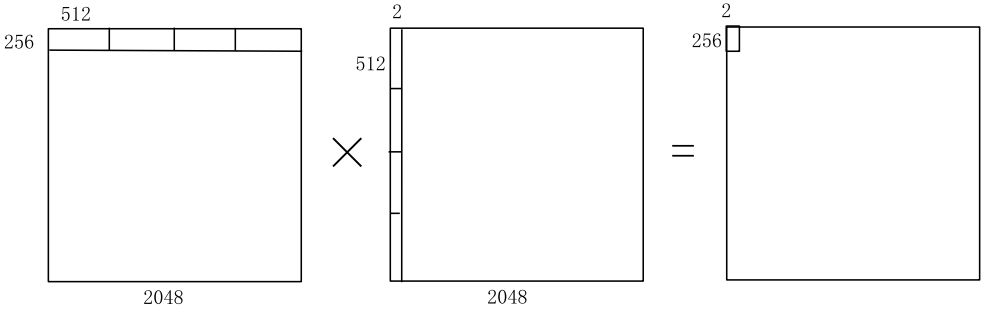

図 11　4 チップで行う場合の行列積の行列の分割方法

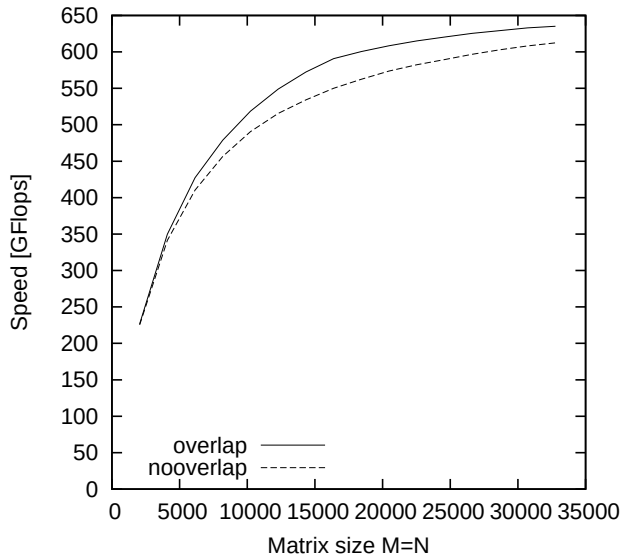

図 12　オーバーラップ時の行列行列積 (DGEMM) の性能と行列サイズの関係．DGEMM のパラメータのうち A 行列の列数は $K = 2048$ に固定し，A 行列の行数 M と B 行列の列数 N を $M = N$ として，横軸にとった．overlap はオーバラップをさせた場合の計算性能を，nooverlap はオーバーラップさせない場合の計算性能を示す．

マシン種別	$\frac{\|Ax-b\|_\infty}{\epsilon\|A\|_1 N}$	$\frac{\|Ax-b\|_\infty}{\epsilon\|A\|_1\|X\|_1}$	$\frac{\|Ax-b\|_\infty}{\epsilon\|A\|_1\|X\|_\infty}$
GRAPE-DR	0.0142381	0.0139567	0.0024497
Intel Core i7	0.0132599	0.0129979	0.0022814

表 1 GRAPE-DR とホスト計算機 (Intel Core i7) のみを用いた場合の HPL での残差の比較．行列サイズおよびブロックサ イズは $N = 34816, NB = 2048$ とした．

マシン種別	計算性能 [GFlops]	パネル分解 [s]	更新 [s]	行交換 [s]	後退代入 [s]
GRAPE-DR	284.2	30.06	64.51	4.06	0.34
Intel Core i7	28.7	52.64	907.36	18.04	0.34

表 2 GRAPE-DR を用いた場合とホスト計算機のみの HPL での各処理段階での実行時間. 行列サイズおよびブロックサイズはそれぞれ表 1 と同様の値を用いた．

の計算量が大きくなるという問題がある．このような問題に対処するため，(1) についてはDGEMM のパラメータで行列積については K=512 まで GRAPE-DR 上で計算を行い，(2) についてはDTRSM を再帰的にブロック化し行列積の形で書くことで GRAPE-DR 上でも計算が行われるようにした．現状のライブラリでは 1024 長語までのオーバーラップにのみ対応しているので，$M = N = 16384, K = 2048$ よりも小さい行列については通信と計算が隠蔽できなくなってしまっている．

GRAPE-DR 上で DGEMM を実行し，$N = 34816, NB = 2048$ の条件下で 1 ノードで HPL を実行した．HPL の実行時間は 98.99 秒となり，計算性能値は 284.2GFlops となった．表 1 に HPL での精度チェックで得られた残差を示す．また，表 2 に HPL での各処理段階で消費された時間をホスト計算機のみとの比較で示す．

5. まとめ・課題

GRAPE-DR 用の重力計算ライブラリと行列積ライブラリを実装をおこなった．現状では通信部分の最適化が不十分である．

現在の実装では重力計算をするための i 粒子，j 粒子転送ともに PCI Express の帯域を活用できていない欠点がある．1 回の 8192 バイトの PIO 転送でチップあたりの PCI Express は 538MB/s と実測されている．128 個の i 粒子転送を行う際は 9216 バイトの通信が発生するので，実効の i 粒子転送でも同じ程度のバンド幅が期待できる．このため 4 チップの帯域を活用することで約 2.1GB/s のバンド幅が期待できる．このためには i 粒子転送に関しては隣接チップ間のブロードキャストリングを用いることで 4 つのチップに別々のデータを送り PCI Express の 16 リンクの帯域を十分に使えるような実装を行う．さらに i 粒子転送時の SING チップのローカルメモリへの転送のオーバーヘッドの解析を詳しく行い，1 チップあたりの転送速度についても PCI Express の実効速度が出せるような最適化を行う．これらを行い，現状の i 粒子転送の実効値である 210.3MB/s を約 10 倍加速するのが今後の課題となる．

j 粒子転送では現在の実装では 1j 粒子の更新ごとに 1 チップに書き込みの転送が発生している．このため 160 バイトごとの転送で WriteCombined 用のバッファをスワップしてしまうために転送効率が落ちている．今後重力計算ライブラリ側でバッファリングを行い，4 チップ同時に書きまたダブルバッファのスワップ回数を減らすような実装を行う．現在の j 粒子の実効転送値である 162.7MB/s を PCI Express 16 リンクの転送実効値の 2.1GB/s に近づける．また，高次微分の精度を削減することで 1j 粒子データのデータ量を 160 バイトから 136 バイトに削減できれば 1 秒間に 1.54×10^7 個の j 粒子転送量が期待できる．

また，行列積ライブラリでは制御回路がオーバーラップ用のバッファが 1024 長語までしか実装されていない．このような状況では $M = N = 16384, K = 2048$ よりも小さい行列では転送と計算が隠蔽できなくなってしまう．より大容量の 8192 長語以上のオーバーラップバッファを実装して $M = N = K = 2048$ のような行列サイズまで通信と転送を完全に隠蔽するようにし，HPL での実効性能を向上させる．

謝辞 本研究は振興調整費プロジェクト「分散共有型研究データ利用基盤の整備」(2004-2008) によって援助をうけた．また K&F Computing Research の川井敦博士には PCI Express インタフェースについて有益な助言と協力をいただいた． 会津大の中里直人准教授には GRAPE-DR 用のコンパイラ LSUMP の提供をしていただいた．

参 考 文 献

1) W.J. Bouknight, S.A. Denenberg, D.E.McIntyre , J.M. Randall, A.H. Sameh, and D.L. Slotnick. The illiac iv system. *Proc. IEEE*, Vol.60, No.4, pp. pp. 369– 388, April 1972.
2) K.Goto. *GotoBLAS*. Texas Advaced Computing Center, THE UNIVERSITY OF TEXAS, http://www.tacc.utexas.edu/resources/software/.
3) W.D. Hillis. *The Connection Machine*. MIT Press, Cambridge, Massachusetts,., 1985.
4) J.Makino and S.J. Aarseth. On a Hermite integrator with Ahmad-Cohen scheme for gravitational many-body problems. *PASJ*, Vol.44, pp. 141–151, April 1992.
5) J.Makino, T.Fukushige, M.Koga, and K.Namura. GRAPE-6: Massively-Parallel Special-Purpose Computer for Astrophysical Particle Simulations. *PASJ*, Vol.55,

Vol.2009-ARC-184 No.26
2009/8/5

pp. 1163–1187, December 2003.

6) Junichiro Makino, Kei Hiraki, and Mary Inaba. Grape-dr: 2-pflops massively-parallel computer with 512-core, 512-gflops processor chips for scientific computing. In *SC '07: Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, pp. 1–11, New York, NY, USA, 2007. ACM.

7) N.Nakasato, J.Makino, H.Matsubara, and T.Ebisuzaki. A compiler for high performance adaptive precision computing. In *SACSIS 2008*, 2008.

8) A. Petitet, R. C. Whaley, J. Dongarra, and A. Cleary. *HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers*. http://www.netlib.org/benchmark/hpl/, 2004.

9) J.L. Potter. *The Massively Parallel Processor*. Cambridge, Massachusetts, 1985.

10) D. Sugimoto, Y. Chikada, J. Makino, T. Ito, T. Ebisuzaki, and M. Umemura. A special-purpose computer for gravitational many-body problems. *Nature*, Vol. 345, pp. 33–35, May 1990.