Vol.2014-MPS-98 No.32
Vol.2014-BIO-38 No.32
2014/6/27

大規模GPUクラスタによるタンパク質ドッキング計算システム

大上 雅史[1,2]　下田 雄大[1]　松崎 由理[3]　石田 貴士[1]　秋山 泰[1,3]

概要：タンパク質ドッキング計算は通常 2 つのタンパク質の複合体構造を予測する手法であるが，3 体以上のドッキング計算やタンパク質間相互作用ネットワークの解析などに応用可能性を持つ．一方でこれらの応用を実現するためには膨大な回数のドッキング計算を要するため，多大な計算リソースと，またそれを充分に活用することのできるシステムの開発が求められていた．本発表では GPU クラスタ上で並列計算が実行可能なタンパク質ドッキング計算システムである MEGADOCK 4.0 を紹介する．本システムの並列性能を，ノードあたり 12CPU コアと 3GPU を備えた TSUBAME 2.5 スーパーコンピュータで測定し，35 ノード実行に対する 420 ノード実行時の強スケーリング値 0.98 を達成した．また，100 万件のタンパク質ドッキング計算が，420 ノードの利用によって約半日で完了することを確認した．

キーワード：タンパク質ドッキング，GPU クラスタ，TSUBAME 2.5，MEGADOCK

Protein–Protein Docking System on Large-Scale GPU Clusters

Masahito Ohue[1,2]　Takehiro Shimoda[1]　Yuri Matsuzaki[3]　Takashi Ishida[1]　Yutaka Akiyama[1,3]

Abstract: The application of protein–protein docking to the large-scale interactome analysis, the treatment of protein flexibility or multiple protein–protein docking problem are current challenges in structural bioinformatics that require huge computing resource. In this work we present MEGADOCK 4.0, an FFT-based docking software which makes extensive use of recent GPU supercomputers and show the powerful scalable performance of over 97% strong scaling with TSUBAME 2.5 supercomputing system. In addition, a million protein–protein docking jobs can be calculated about a half day by using 420 nodes of TSUBAME 2.5.

Keywords: Protein–Protein Docking, GPU Cluster, TSUBAME 2.5, MEGADOCK

1. 序論

タンパク質間相互作用は様々な細胞内プロセスや機能に関わる現象であり，疾病の理解や創薬標的の決定などに相互作用の情報が活用されている．タンパク質間相互作用の大部分はタンパク質同士が直接結合して複合体を形成するものであるが，近年の立体構造の決定技術の発展に伴う構造情報の増加の一方で複合体構造の多くは実験的には決定されていない．そのため，単体のタンパク質構造を利用して複合体構造を予測する**タンパク質ドッキング（protein–protein docking）**が注目されている [1].

タンパク質ドッキングによる複合体構造予測では，予測構造を洗練させるために段階的に処理を行うことが多い．様々な研究グループがそれぞれ独自の予測手法を提案しているが，ほとんどの手法において初期段階として単体の構造を剛体として扱う剛体ドッキングを採用している [2]．特に高速フーリエ変換（fast Fourier transform, FFT）によってグリッド同士の重ね合わせを計算する Katchalski-Katzir ア

[1] 東京工業大学 大学院情報理工学研究科 計算工学専攻
Graduate School of Information Science and Engineering, Tokyo Institute of Technology
[2] 日本学術振興会 特別研究員
Research Fellow of the Japan Society for the Promotion of Science
[3] 東京工業大学 情報生命博士教育院
Education Academy of Computational Life Sciences, Tokyo Institute of Technology

ルゴリズム [3] を用いた剛体ドッキングが広く利用されており，様々なツールが開発されている [4], [5], [6], [7], [8], [9]．我々が開発する MEGADOCK [10], [11] もこれらのツールと同様，FFT に基づくドッキング計算を行うソフトウェアである．

剛体ドッキングツールは複合体構造の予測に用いられるほか，タンパク質間相互作用の有無の予測 [12], [13]，タンパク質複合体の親和性の予測 [14]，構造アンサンブル同士のドッキング [15] などに応用されている．また多くのツールは通常 2 体のタンパク質の複合体しか扱えないが，2 体のドッキング計算を複数回行って 3 体以上の複合体構造の予測を行う試みもなされている [16]．このような応用例では数万〜数百万のタンパク質ペアを扱うため，ドッキング計算の高速化や並列化が注目されている [17], [18]．我々も CPU クラスタ向けの並列実装 [19] や GPU アクセラレータの利用による高速化 [20] を行ってきたが，近年の主要な並列計算機に見られる GPU を搭載したノードを多数接続したクラスタ（大規模 GPU クラスタ）の活用のためには，GPU を利用しかつ複数ノード間での効率的な並列化を行うことが求められる．

そこで本研究では，さらに大量のドッキング計算を実施可能にすることを目的として，大規模 GPU クラスタ向けの並列実装を行った．

2. 関連研究

2.1 MEGADOCK

我々が開発している MEGADOCK は，複合体構造の評価を行うスコア関数に，形状相補性（rPSC モデル [10]），静電相互作用，脱溶媒和自由エネルギー（RDE モデル [11]）の 3 つの要素を取り入れている．複数の要素をスコア関数に組み入れる場合には通常複数回の FFT を計算する必要があり，例えば PIPER [6] では 22 回の FFT を，ZDOCK [8] では 8 回の FFT を必要としているが，MEGADOCK では 1 回の FFT のみで計算する工夫によって高速化が行われている．

2.2 MEGADOCK-GPU

MEGADOCK は複数の GPU を活用できるよう，既に CUDA による GPU 化が行われている（MEGADOCK-GPU [20]）．ドッキング計算を GPU 化した研究として，Sukhwani らによる PIPER の GPU 実装 [18] が挙げられるが，Sukhwani らは FFT 部分を CUDA の cuFFT ライブラリを用いて GPU 上で計算し，それ以外の処理は CPU で行っていたのに対し，我々はタンパク質のグリッド分割や回転処理，高スコア解の選出といった処理も GPU 上で行うことで，CPU–GPU 間のデータ転送量を抑え，計算時間の削減に成功している．また，Sukhwani らが扱っていなかったマルチ GPU 計算にも対応した．

2.3 MEGADOCK 3.0

前述の GPU 化とは別に，MEGADOCK は複数のタンパク質ペアを計算するための CPU クラスタ上での並列化が行われている（MEGADOCK 3.0 [19]）．MEGADOCK 3.0 では，ノード内のメモリ使用量を抑えるため，計算するタンパク質ペアの配分と監視を行うプロセス（master）と，実際にドッキング計算を行うプロセス（worker）に分けた master/worker 型の実装を行った．master と worker は MPI 通信を行い，worker 同士の通信は行わず，worker プロセス内は OpenMP によるスレッド並列によって 1 つのタンパク質ペアのドッキング計算を並列化するというハイブリッド並列化を採用した．この実装は主に理研 AICS の「京」上で動かすことを想定したものであるが，通常のクラスタマシンでも問題なく動作することを確認している．

3. GPU クラスタ向けの実装

大量のドッキング計算をさらに高速に処理するために，GPU クラスタ向けの並列実装を行った．実装の基本方針は CPU クラスタを利用する場合と同じく master/worker 型であるが，GPU が搭載するメモリ量は通常 CPU に比べて遥かに小さい（NVIDIA Tesla K20X GPU の例では 6 GB）ので，master ノード以外のノード（worker ノード）に 1 プロセスずつドッキング計算を配分し，worker ノード全体で 1 ペアのドッキングを，CUDA による GPU の利用と OpenMP によるスレッド並列で計算するというハイブリッド並列方式をとった．これらの実装を行ったものを，今後 **MEGADOCK 4.0** と呼ぶ．MEGADOCK 4.0 の並列化の概要図を**図 1** に示す．

4. 実験と考察

新たに実装した MEGADOCK 4.0 の並列性能を測定するため，GPU クラスタによる計算機実験を行った．本研究では，タンパク質ドッキング研究で広く用いられている ZLAB benchmark 4.0 [21] というデータセットを用いた．計算はすべて東京工業大学のスーパーコンピュータ TSUBAME 2.5 の Thin ノードで行った．TSUBAME 2.5 の Thin ノードは，1 つのノード内に 12 コアの CPU（Intel Xeon X5670×2）と 3 枚の GPU（NVIDIA Tesla K20X）を搭載している．環境の詳細は**表 1** に示した．

4.1 強スケーリングの測定

ZLAB benchmark 4.0 の全複合体（176 ペア）のデータを用いて，結合相手を入れ替えた全ての組み合わせである $176 \times 176 = 30{,}976$ ペアのドッキング計算を行った．35 ノードで測定した計算時間を基準とし，420 ノードまでの強スケーリング値（strong scaling）を求めた．計算時間は各ノード 5 回ずつ測定した値の平均値を用いた．ここで n ノードでの強スケーリング値 Strong Sc_n は，n ノードで

Vol.2014-MPS-98 No.32
Vol.2014-BIO-38 No.32
2014/6/27

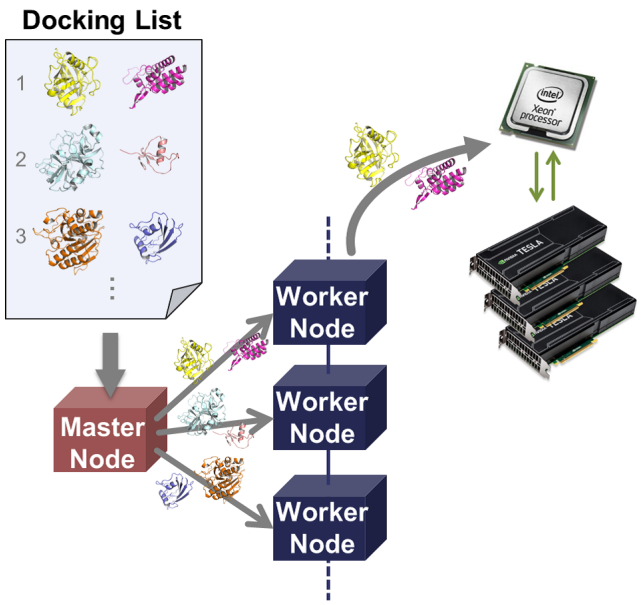

図 1 MEGADOCK 4.0 の GPU クラスタを利用した並列計算の概要図（CPU の図は http://www.intel.com より，GPU の図は http://www.nvidia.com より引用した．）

Fig. 1 The overview of the parallelization of MEGADOCK 4.0 on GPU cluster (the picture of CPU is reprinted from http://www.intel.com and the picture of GPU is reprinted from http://www.nvidia.com).

表 1 TSUBAME 2.5 Thin ノードの詳細.

Table 1 Hardware specification of TSUBAME 2.5 Thin node.

CPU	Intel Xeon X5670 (2.93 GHz) (6 cores) ×2
Memory	54 GB
OS	SUSE Linux Enterprise Server 11 SP1
GPU	NVIDIA Tesla K20X (GK110) ×3
GPU Memory	6 GB/GPU
Compiler	Intel C++ Compiler 14.0.2.144
FFT Lib (CPU)	FFTW 3.2.2
CUDA	CUDA 5.5
FFT Lib (GPU)	cuFFT 5.5

の計算時間 T_n を用いて

$$\text{Strong Sc}_n = (T_{35}/T_n)/(n/35) \quad (1)$$

と表される．

表 2 に測定結果を示す．MEGADOCK 4.0 の強スケーリング値は全体を通して 0.97 を超えており，420 ノードでの計測では 0.980 という高い値を示した．

4.2 大規模計算の実施

ZLAB benchmark 4.0 の中から平均的な大きさ（FFT のサイズが $N = 108$）であるタンパク質を 27 個抽出し，それらの総当たりの計算を複数回行って総計 50 万件および 100 万件のドッキング計算を実施した．この計算は 420 ノードを利用して行った．結果は**表 3** に示す通り，MEGADOCK 4.0 は 100 万件のドッキング計算を約半日で実施可能であることが示された．

表 2 30,976 ペアのドッキング計算の計算時間と強スケーリング.

Table 2 Benchmarking results (strong scaling) of 30,976 docking jobs.

#Nodes n	35	70	140	280	420
Time T_n (min)	264.4	133.3	67.4	33.1	22.5
Strong Sc_n†	-	0.991	0.973	0.988	0.980

† Strong scaling value from 35 nodes
(Strong $\text{Sc}_n = (T_{35}/T_n)/(n/35)$)

表 3 平均サイズのデータセットに対し TSUBAME 420 ノードで計算したときのドッキング計算時間.

Table 3 Benchmarking results (computation time) of averaged set with TSUBAME 420 nodes.

#Protein pairs	500,000	1,000,000
Docking time (hour)	5.71	11.51

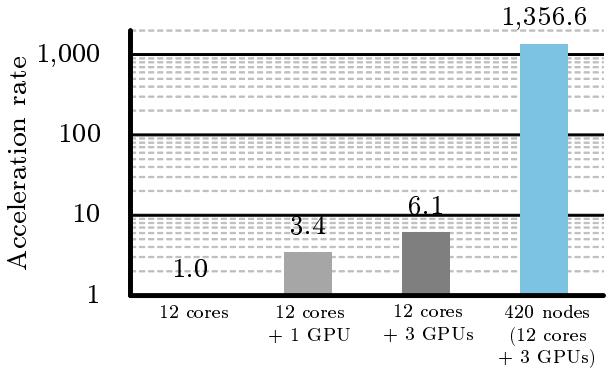

図 2 単一ノードとマルチノードでの高速化率．値は 12 cores を 1 としたときの高速化率である．

Fig. 2 The acceleration rate on single node and multiple nodes. The rates are calculated based upon the speed up when the “12 cores” is 1.

4.3 単一ノード実行との比較

単一ノードを利用した逐次計算とマルチノードによる計算を比較した．4.2 節で用いたデータセットのタンパク質を用いて 10,000 件のドッキング計算を行い，その計算時間から高速化率を求めた．単一ノードの 12 CPU コアでの計算時間を 1 としたときのそれぞれの高速化率を**図 2** に示す．

12 CPU コア利用に比べて，GPU を 1 枚利用した場合は 3.4 倍，3 枚利用した場合は 6.1 倍の高速化が達成されているが，420 ノードの全資源を利用した場合には約 1,360 倍の高速化が達成された．12 CPU コアと 3 GPU を利用した場合同士を比べると，420 ノードと単一ノードとの間は約 220 倍の高速化に留まっているが，これは master ノードを 1 つ確保する必要があること，worker ノードの監視が必要なことなど，単一ノードでの逐次実行の場合とは異なる，大規模実行に向けた処理が含まれているためである．

5. 結論

我々が開発したタンパク質ドッキングソフトウェアである MEGADOCK を，大規模 GPU クラスタ向けに並列化し，その計算性能を確認した．新たに並列実装を行った MEGADOCK 4.0 は，アクセラレータを搭載したヘテロジニアスな環境においても高いスケーラビリティを保ち，大量のドッキング計算を短時間に実施可能であることを示した．現在は GPU とは別の主要なアクセラレータである，Intel Xeon Phi に代表される MIC アーキテクチャ向けの並列実装，および MIC 搭載クラスタ向けの並列化も検討中である．電力効率等の面から，今後もアクセラレータを多数搭載したスーパーコンピュータは増え続けるものとみられ，MEGADOCK はそのような計算機環境をフルに活用できるツールとして，大規模なインタラクトーム解析やアンサンブルドッキング等に活用されることが期待される．

謝辞

本研究は，科学研究費補助金（特別研究員奨励費 23·8750, 26·30002）および HPCI システム利用研究課題（hp140173）の支援を受けて行われたものである．

参考文献

[1] Stein, A. *et al.* (2011) Three-dimensional modeling of protein interactions and complexes is going 'omics. *Curr. Opin. Struct. Biol.*, 21, 200–208.
[2] Lensink, M.F., Wodak, S.J. (2013) Docking, scoring and affinity prediction in CAPRI. *Proteins*, 81, 2082–2095.
[3] Katchalski-Katzir, E., *et al.* (1992) Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques. *Proc. Natl. Acad. Sci.*, 89: 2195–2199.
[4] Gabb, H.A. *et al.* (1997) Modelling protein docking using shape complementarity, electrostatics and biochemical information. *J. Mol. Biol.*, 272, 106–120.
[5] Chen, R. *et al.* (2003) ZDOCK: an initial-stage protein-docking algorithm. *Proteins*, 52, 80–87.
[6] Kozakov, D. *et al.* (2006) PIPER: an FFT-based protein docking program with pairwise potentials. *Proteins*, 65, 392–406.
[7] Cheng, T.M.-K. *et al.* (2007) pyDock: electrostatics and desolvation for effective scoring of rigid-body protein-protein docking. *Proteins*, 68, 503–515.
[8] Pierce, B.G. *et al.* (2011) Accelerating protein docking in ZDOCK using an advanced 3D convolution library. *PLOS ONE*, 6, e24657.
[9] Roberts, V.A. *et al.* (2013) DOT2: Macromolecular docking with improved biophysical models. *J. Comput. Chem.*, 34, 1743–1758.
[10] Ohue, M. *et al.* (2014) MEGADOCK: An all-to-all protein–protein interaction prediction system using tertiary structure data. *Protein Pept. Lett.*, 21. (epub ahead of print)
[11] Ohue, M. *et al.* (2012) Improvement of the protein–protein docking prediction by introducing a simple hydrophobic interaction model : an application to interaction pathway analysis. *Lect. Notes Comput. Sci.*, 7632, 178–187.
[12] Wass, M.N. *et al.* (2011) Towards the prediction of protein interaction partners using physical docking. *Mol. Syst. Biol.*, 7, 469.
[13] Ohue, M. *et al.* (2013) Highly precise protein–protein interaction prediction based on consensus between template-based and de novo docking methods. *BMC Proc.*, 7, S6.
[14] Fleishman, S.J. *et al.* (2011) Community-wide assessment of protein-interface modeling suggests improvements to design methodology. *J. Mol. Biol.*, 414, 289–302.
[15] Król, M. *et al.* (2007) Flexible relaxation of rigid-body docking solutions. *Proteins*, 68, 159–169.
[16] Karaca, E., Bonvin, A.M.J.J. (2011) A multidomain flexible docking approach to deal with large conformational changes in the modeling of biomolecular complexes. *Structure*, 19, 555–565.
[17] Ritchie, D.W., Venkatraman, V. (2010) Ultra-fast FFT protein docking on graphics processors. *Bioinformatics*, 26, 2398–2405.
[18] Sukhwani, B., Herbordt, M.C. (2009) GPU acceleration of a production molecular docking code. In *Proc. of GPGPU-2*, 19–27.
[19] Matsuzaki, Y. *et al.* (2013) MEGADOCK 3.0: a high-performance protein–protein interaction prediction software using hybrid parallel computing for petascale supercomputing environments. *Source Code Biol. Med.*, 8, 18.
[20] Shimoda, T. *et al.* (2013) MEGADOCK-GPU: acceleration of protein–protein docking calculation on GPUs. In *Proc. of ACM-BCB'13*, 883–889.
[21] Hwang, H. *et al.* (2010) Protein–protein docking benchmark version 4.0. *Proteins*, 78, 3111–3114.