2004－DPS－116　(4)
2004／1／29

大量ゲノムデータの解析速度を基準とした グリッドコンピューティングの有効性の検証

五味悠一郎† 荻島創一† 田中博‡

グリッドコンピューティングは、そのシステム上、疎結合型並列処理向きである。一方、生命情報学分野では解析に多重配列整列が用いられる事が多く、小規模演算が大量に繰り返される事から、グリッドコンピューティングに適していると考えられる。

そこで、グリッドコンピューティング上で多重配列整列処理を行い、データ量やデータ数、スペック等を変更した場合の処理速度を計測した。

その結果をスタンドアロンＰＣ上での処理速度と比較し、有効性の検証をおこなったのでここに報告する。

How grid computing improves using the processing speed of large-scale sequence alignment?

Yuichiro Gomi† Soichi Ogishima† Hiroshi Tanaka

In a Bio Informatics, large-scale sequence alignment is used for analysis in many cases. As for large-scale sequence alignment, much small-scale operation is repeated. Grid computing is for sparse combination type parallel processing. I think it is suitable for Bio Informatics.

In order to measure processing speed, large-scale sequence alignment is performed by grid computing. As compared with the processing speed on Stand-alone PC, validity was verified

1. はじめに

生命情報学の研究分野では、データの相同性を検索したり、多重配列整列を行う事が多い。これらの作業は計算量が膨大になるので、スタンドアロンＰＣ上では、対象とするデータのサイズや比較回数に比例して、長い時間を必要とする。

一般的に、この様な計算を行うときには、ＰＣクラスタが使われる事が多いが、利用しない時間も多く、メンテナンスやアップグレードのことも考えると非経済的である。

2. 目的

我々は、安価に処理の高速化を図る手段として、科学技術計算分野で注目されているグリッドコンピューティングの技術を用い、生命情報学で良く利用されるプログラムを実行させ、その処理時間を計測し、多角的に比較検討する事によって、その有効性を検証する。

3. グリッドコンピューティング

ネットワークに接続された多数のコンピュータの余剰能力を利用し、粒度の高い並列処理で、非常に高いスループットを、安価に実現するアーキテクチャである。ユーザ側はネットワークにマシンを接続するだけで、その先にどんなマシンが存在するか意識することなく、必要なときに必要なだけ利用することができる。

今回使用したグリッドコンピューティングシステムは、NTT データの「cell computing®」である。このシステムは昨年行われた大規模実証実験によって、遺伝子病治療研究プロジェクトや光マイクロプロセッサプラットフォーム実現プロジェクトに大きな貢献をした。今回の研究のために、イントラネット版のクライアント５０台分ライセンス及びサーバを借りる事が出来たので、これを用いる。

アーキテクチャは、図１のように各クライアントＰＣに対してメンバソフトをネットワーク経由で配布し、インストール。各クライアントはサーバから処理すべきデータをもらうと、各クライアント内部で処理を完了し、結果をサーバに戻す。

†東京医科歯科大学　医歯学総合研究科　生命情報学
†graduate scool , Tokyo Medical and Dental University

‡東京医科歯科大学　難治疾患研究所　生命情報学
‡Medical Research Institute , Tokyo Medical and Dental Universitv

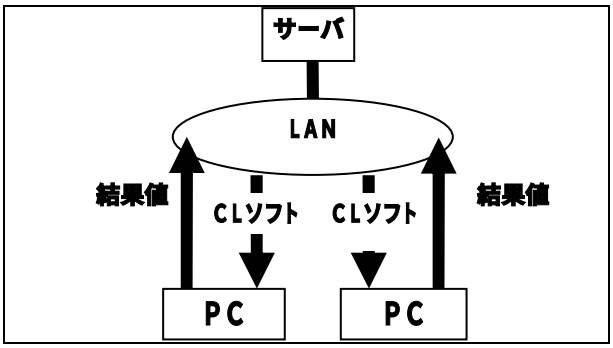

図 1 イントラネット型 cell computing

4. プログラム

生命情報学で良く使われているプログラムであり、グリッド上で使えるようにソースを多少変更する必要があるのでオープンソースであることから、実行させるプログラムは、ClustalW と Hmmer を用いた。

4.1. Hmmer

HMM(hidden Markov moder)のアルゴリズムを用いて、遺伝子やタンパク質の相同性を検索するプログラムである。HMM とは状態の変化を現在の状態と次の状態との間での遷移確率で表したものである。

今回の実験では、Pfam_ls という広域整列モデルと呼ばれる HMM に対し、ヒトアミノ酸配列をクエリとしてモチーフ検索(hmmpfam)を行い、タンパク質の分類や整理を行った。モチーフとは、複数のタンパク質間で見いだされる、共通のアミノ酸パターンのことである。

4.2. ClustalW

類似性のある複数の遺伝子配列を、類似した配列が重なるように多重に整列配列し、遺伝子配列の類似性の比較から、遺伝子の系統樹を作成するプログラムである。

5. 方法

5.1. システムセットアップ

東京医科歯科大学のコンピュータルームのクライアントＰＣ５０台にメンバソフトのインストールを行い、クライアントＰＣからサーバへ登録を行う。サーバは自動的にクライアントＰＣを認識する。

表１に、今回使用したマシンのスペック、表２に使用したプログラムを示す。なお、ＰＣクライアントは５０台、サーバ１台というシステム構成で、クライアントＰＣとサーバはそれぞれ、100BASE-T の LAN で接続されている。

5.2. Hmmer

用意されている Hmmer を用いてジョブを投入する。必要に応じてパラメータを設定し、データ測定を行う。

大規模なデータベースに対し、クエリのマッチングをかけることが処理の大部分であるので、クエリサイズ別の処理時間を計測するために、同一のファイルを希望するサイズになるまでつなぎ合わせ、実験に必要なクエリを用意した。

一通りジョブが完了したら、シェル上から出力結果を取得し、処理時のログデータをＷｅｂ上から取得した。

5.3. ClustalW

グリッド上で ClustalW を実行させるためには、スタンドアロンＰＣ用のプログラムはそのままでは実行できず、cell computing®用の ClustalW も存在しないので、移植作業を行った。

サーバに ClustalW のタスクを登録し、そのタスクに同じネットワーク上の PC からジョブを投入する。今回の処理では数万のタンパク質について、それぞれ１つの処理を行うデータを投入するので、その作業は Perl スクリプトにより実装する。

なお、処理を行うためのデータは NCBI（米国バイオテクノロジー情報センター）のデータベースから、アミノ酸配列のグループデータベースを作成した。

データの取得は、Hmmer と同様に行う。

表 1 各マシンのスペック

	OS	CPU	Memory
グリッド（SV）	RedHat 7.1	Pentium3 800MHz	128MB
グリッド（CL）	Windows98 SE	Celelon 533MHz	128MB
スタンドアロンa	Windows98 SE	Celelon 533MHz	128MB
スタンドアロンb	Windows XP	Pentium4 2.53GHz	1024MB
スタンドアロンc	Windows 2000	Celeron 1.4G	512MB
スタンドアロンd	Windows 2000	Athlon 900M	512MB

表 2　使用プログラム

プログラム	バージョン
Hmmer	Hmmer2.2g
ClustalW	ClustalW 1.83

5.4.　スタンドアロンＰＣ

最後に、スタンドアロンＰＣ上で、それぞれのプログラムを実行させ、その処理時間を計測した。

6.　結果

Hmmer、ClustalW それぞれにおいて、グリッド上で実行させた場合と、スタンドアロン上で実行させた場合の、作業完了までの時間を比較した。

なお、測定値のバラツキを防ぐために、結果は同じ条件でそれぞれ２回測定し、その平均をとった。

6.1.　Hmmer

まず、100Byte のクエリで計算した。その結果を表３に示す。

表 3　Hmmer の実行結果(100Byte)

マシン	処理時間
グリッド (CL)	1015sec
スタンドアロンa	305sec
スタンドアロンc	136sec
スタンドアロンd	341sec

Hmmer をグリッド上で実行させた場合、ジョブをいくつかの固まりに分割し、各々をクライアントに渡し、処理をした後サーバに答えを返すという動作をとった。

設定を変更しないで実行させたところ、大幅に処理時間に差が生じた。

このタイプのグリッドは、一般家庭向けＰＣのアイドルタイムを利用する事を想定しているため、ＷＡＮでは不確定要素が多いために冗長性を大きく取っているが、今回の実験の特徴はクエリが小さい事と、PC50 台の LAN なので個々の PC の信頼性は高いだろうという判断から、WU の分割数を当初の 1704 個から 580 個に減らし、Redundancy も２から１に減らした所、734sec で処理を完了する事が出来た。

そこで、この後の実験はこの設定で行う事にした。パラメータ変更後の Hmmer の実行結果を表４に示す。比較しやすいように、グラフにしたのが図２である。

表 4　Hmmer 実行結果(パラメータ変更後)

クエリサイズ [Byte]	グリッド処理時間[s]	スタンドアロンa 処理時間[s]	グリッド利用 PC[台]
1,000	779	1,079	30.5
5,000	2,107	7,677	30
10,000	4,729	16,518	29.5
20,000	8,019	36,565	31
50,000	20,850	94,938	28
100,000	14,895	173,065	31.5
500,000	71,073	765,321	30.5

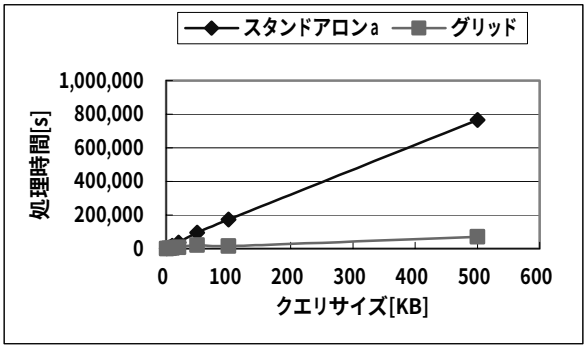

図 2　サイズに対する処理時間

6.2.　ClustalW

実行マシン、処理データサイズ、比較回数を変更し、測定した結果を表５～表７に示す。

データ比較をしやすいようにグラフ化したものが、図３～図１４である。

表 5　AAF46957(1KByte)

データ数[個]	10	50	100
グリッド 処理時間[s]	8	20	140
スタンドアロンa 処理時間[s]	27	54	107
スタンドアロンb 処理時間[s]	1	2	5

表 6　AAF53973(10KByte)

データ数[個]	10	50	100
グリッド 処理時間[s]	32	83	146
スタンドアロンa 処理時間[s]	1380	2761	6420
スタンドアロンb 処理時間[s]	40	203	405

表 7　P16106(23KByte)

データ数[個]	10	50	100
グリッド 処理時間[s]	65	144	425
スタンドアロンa 処理時間[s]	2749	5499	10999
スタンドアロンb 処理時間[s]	105	523	1046

7.　考察

コンピュータルーム内 LAN のスループットを FTP を利用して計測した所、24Mbps 程度であった。処理に用いたデータベース Pfam_ls はサイズが 524MB あるので、単純計算でクライアントへの送信時間が 175sec かかることになる。

これらのことを考慮に入れると、WU や Redundancy の値を変えた事により、1.5 倍程度のスピードアップが図れたことになる。今回の測定ではクエリサイズが小さいため、分割数が多いと遅くなったと考えられるので、クエリが大きい場合は元々の設定値が最適値になると思われる。

比較する回数が少ない場合や、データサイズが小さい場合などはクライアントマシンをスタンドアロン状態で動作させた方が処理時間が短い場合がある。これは、各クライアントへのジョブの振り分け作業に時間がかかるためである。

ジョブ投入から結果が帰ってくるまでの処理速度で考えると、データサイズが大きいほど能力を発揮できると言える。

また、比較回数と処理時間だけでみると、グリッドに特徴はみられない。すなわち、ある一点の値がわかれば、比較回数を変えた場合の処理時間が予測可能であると言える。

8.　結論

今回、一通り cell computing®を使ってみたが、１世代前のマシンが余っているが、ある程度能力のあるマシンも欲しい、という所に適しているシステムと言える。

問題点としては、まだ発展途上にあるシステムのため、利用したいプログラムがある場合には、移植作業などで多くの時間を必要としてしまい、ＷＡＮで利用する事を考えると、セキュリティを高くしなければならないため、さらに時間がかかってしまう所だろう。

また、デフォルトの設定のままでは、能力を十分に発揮できないことと、最低でも１日以上はかかる計算でなければ、本来の能力を発揮することは難しい点に気をつけなければならない。

今回はバイオインフォマティクス系のツールを使うために、移植作業や最適なパラメータの模索、ジョブを投入する際にもシェルで入ったりＷｅｂから指示を出したりと、測定以外の所で時間の大半を使った。

大規模計算で使われるツールがほぼ網羅され、自動的に最適なパラメータを見つけ出し、操作も１つのコンソールから全て行える様になれば、かなり使えるツールとなるであろう。

準備に多少時間がかかる事を除けば、古いマシンを５０台程度集めれば３年前の機種でも、現行のマシンと同等に扱う事が出来るので、有効性はあると言える。

9.　まとめ

今回、マクロな視点からジョブ投入時の cell computing®の処理能力を調べた。

この研究の成果を用いれば、今後グリッド上で生命情報学系のツールを利用する際に、処理完了時間の目安や、効率的なジョブの投入方法が分かり、大規模計算が必要な研究に対して貢献できたと言える。

また、まだまだ使いやすい形にはなっていないが、ClustalW を移植する事により、今後利用するユーザに対し貢献できた。

今後は、利用クライアント台数や LAN 回線のスループットを変えた場合のスタンドアロンＰＣとの性能比や、各種パラメータを変えたときの計算タスクの分配効率などについて比較を行い、さらに詳しく検証をして行きたい。

参考文献

[1]United Devices, Inc.(http://members.ud.com/)
[2]SETI@home(http://setiathome.ssl.berkeley.edu/)
[3]grid.org(http://www.grid.org/)
[4]cell computing(http://www.cellcomputing.jp/)
[5]五味他;グリッドコンピューティングの有効性の検証;情報処理学会第 65 回全国大会 pp.3-259-3-260(2003)
[6]広川貴次,美宅成樹「できるバイオインフォマテックス」（中山書店,2002）

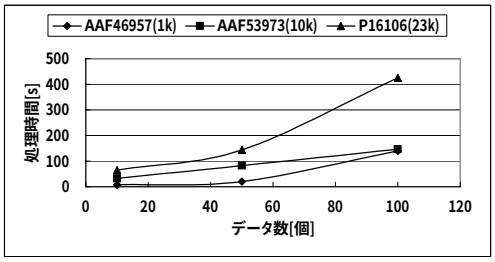

図3　データ数に対する処理時間（グリッド利用）

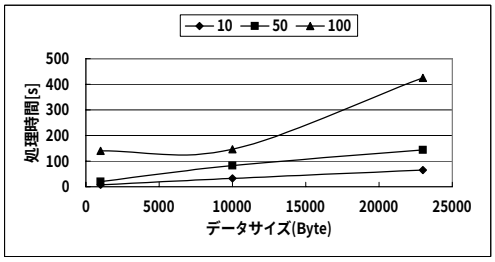

図9　データサイズに対する処理時間（グリッド利用）

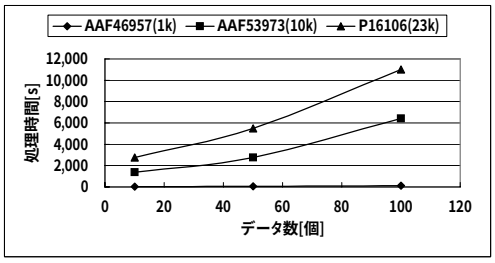

図4　データ数に対する処理時間（スタンドアロンa利用）

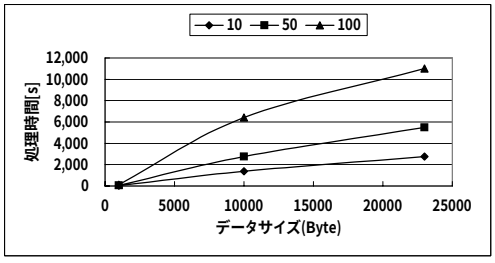

図10　データサイズに対する処理時間（スタンドアロンa利用）

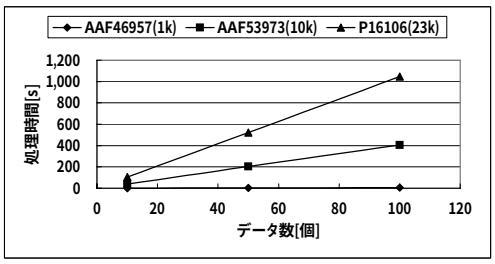

図5　データ数に対する処理時間（スタンドアロンb利用）

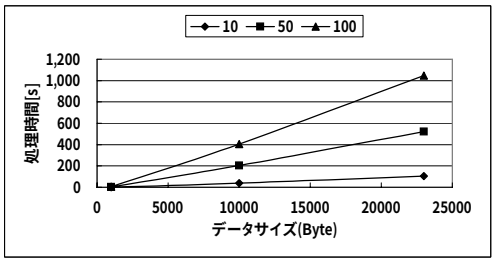

図11　データサイズに対する処理時間（スタンドアロンb利用）

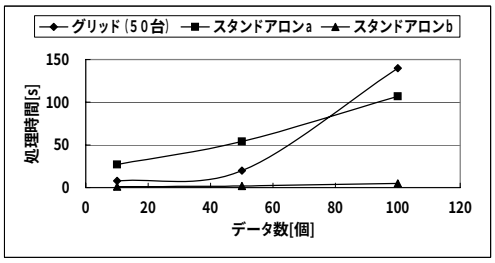

図6　データ数に対する処理時間（データサイズ1KB）

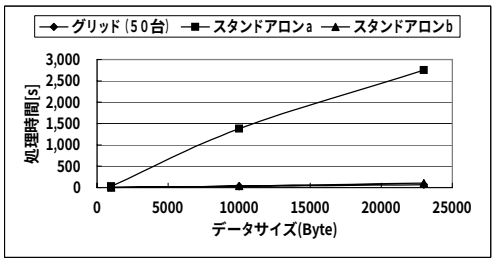

図12　データサイズに対する処理時間（データ数10）

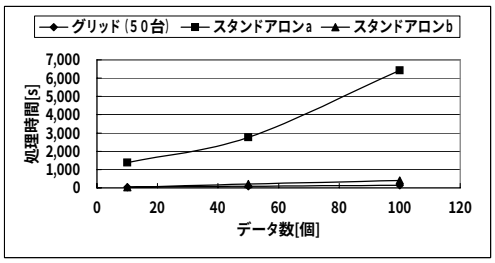

図7　データ数に対する処理時間（データサイズ10KB）

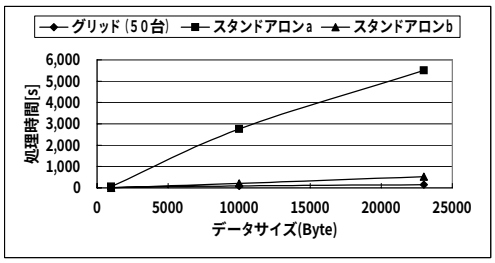

図13　データサイズに対する処理時間（データ数50）

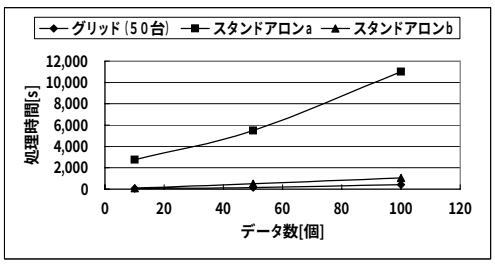

図8　データ数に対する処理時間（データサイズ23KB）

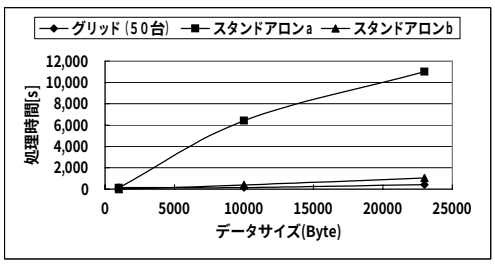

図14　データサイズに対する処理時間（データ数100）