グリッドコンピューティングによる 遺伝的アルゴリズムの高速化

遠藤皆生 (福井大学工学部)
白井治彦、高橋勇、黒岩丈介、小高知宏、小倉久和 (福井大学)

1 はじめに

本研究では遺伝的アルゴリズム (以下 GA) をグリッドコンピューティングにより並列計算し処理を高速化することを目標とする。

グリッドコンピューティングでは、複数のコンピュータに処理を分散させて計算させるわけだが、そのとき使用するコンピュータは並列計算を行わせるために性能を揃えたものではなく、普段は別の用途に使用しているコンピュータを用いる。そのため、それぞれのコンピュータの性能は各マシンごとに差があり、また、各マシンはそれぞれ何らかの処理を行っているので、かかっている負荷にも差がある。これらのコンピュータに均等に仕事を割り振った場合、その処理能力の差により結果が返ってくるまでの時間にばらつきがでてしまい計算の効率が悪くなってしまう。

そこで本研究では、GA の並列化においてそれぞれのコンピュータの処理能力及び負荷に応じて処理すべき仕事の量を動的に変化させ、全体の処理時間を短縮する分散処理システムの構築を目指す。

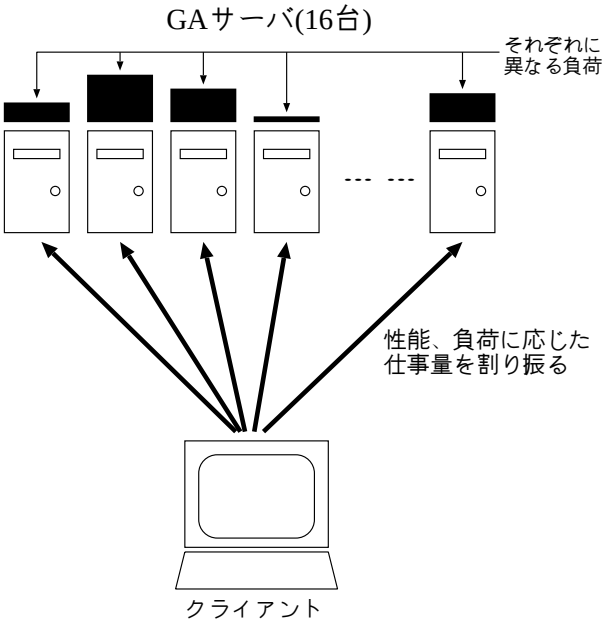

図 1: システムの概念図

2 並列化の方法

GA の並列化の方法としては、突然変異や交差などの遺伝的操作を並列化する方法や、GA 全体を並列化する方法などいろいろあるが、今回は評価の処理の並列化を行う。評価を並列化をする理由としては、評価は他の処理に比べて処理に時間がかかる場合があり、また遺伝子ごとに独立して計算することができ並列化に適しているからである。

本システムの実験環境として 16 台のコンピュータを使用する。それぞれのコンピュータは通常、画像処理やその他様々な処理を行っている。このコンピュータに評価サーバとして、割り振られた遺伝子の計算を行うプログラムを常時走らせておく。システムは各コンピュータに仕事を割り振りながらも常に処理時間を計り、その処理能力から各コンピュータに割り振るべき仕事量を判断し分散処理を行わせる。

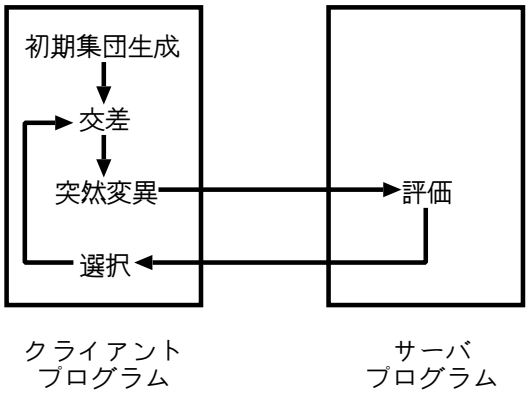

図 2: GA の評価をサーバで行わせた場合の処理の流れ

3 システムの実装

本システムの作成は CORBA(Common Object Request Broaker Architecture) を用いて行う。CORBA とはネットワークを介してオブジェクトを利用するための基盤を提供するものであり、環境や言語に依存しないという特徴を持つ。今回、システムの開発には JDK に標準で付属している CORBA 環境を使用し、

言語は Java を用いた。

まず、システムの作成にあたり GA のプログラムの作成を行う。このとき、評価の計算を行う部分は独立した 1 つのオブジェクトとしてプログラミングしておく。

つづいて、この評価オブジェクトをネットワークを介して利用するために、CORBA のサービスのひとつであるネーミングサービスを利用する。その方法は、まずサーバ側にある評価オブジェクトへの参照をネームサーバに名前をつけて登録する。登録がなされたら、クライアント側でネームサーバに対して登録してある名前で検索をかけてオブジェクトへの参照を取得する。この参照を用いてローカルにあるオブジェクトと同様に評価オブジェクトを呼び出せば、クライアントとサーバが直接通信を行い評価オブジェクトを利用することができる。このネーミングサービスを利用することにより、クライアント側はネームサーバの場所だけを知っていればよく、利用するオブジェクトの場所は知る必要がない。

GA の評価の計算は遺伝子ごとに独立しているので、あらかじめ評価オブジェクトを実装したサーバを複数用意しておけば、それらに対して遺伝子を分割して同時にサーバに処理させることにより、処理を分散させて行わせることができる。

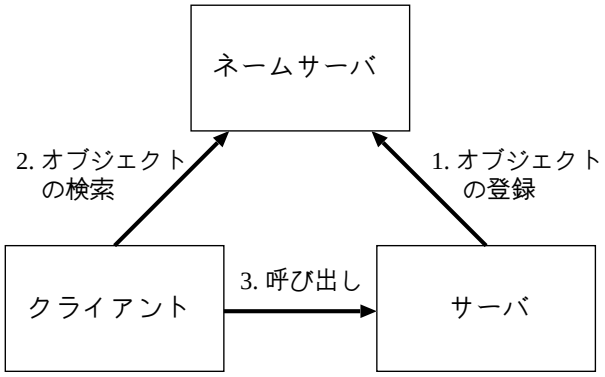

図 3: ネームサービスを利用した通信の手順

4 実験

今回は予備実験として環境の異なった 3 台のコンピュータで、GA の計算を CORBA を用い分散させて行わせた。実験に使用した 3 台のコンピュータの詳細は表 1 のようになる。

実験として、3 台のコンピュータに対して均等に遺伝子を割り振った場合と、性能に応じて遺伝子の量を動的に調整して割り振った場合の 2 種類の方法を行った。その結果を表 2 に示す。

表 1: サーバに使用した各コンピュータの性能

	CPU	OS
A	celeron 366MHz	Linux
B	celeron 700MHz	Linux
C	Athlon 1800	Windows

表 2: GA の処理時間 (10 回の平均)

	処理時間 (sec)
均等に振り分けた場合	51.6
動的に振り分けた場合	27.5

5 考察

今回、分散処理を行わせる実験環境として、それぞれ性能に大きく差があり、また Linux と Windows という異なった OS がインストールされているコンピュータを用意した。実験の結果、このような様々な環境下においても分散処理を行わせることができたので、CORBA を用い Java でシステムを作成することにより、環境に依存しない分散処理システムの構築が可能であることがわかった。また、処理能力に応じて仕事量を変化させた場合とそうでない場合とでは、処理時間に大きな差がみらる。このことから、個々のコンピュータの能力を活かすためにも、能力に応じた仕事量の調整が重要であることがわかった。

今後は今回の結果をもとに、サーバの台数が増えたときの CORBA によるサーバの管理方法や、より効率の良いスケジューリング方法についての検討を行っていきたい。

参考文献

[1] KunYang,XinGuo,AlexGalis:”Towards Efficient Resource on-Demand in Grid Computing”,Oper Syst Rev,Vol.37,No.2, p37-43,(2003)

[2] Jeremy Rosenberger :「例題で学ぶ分散オブジェクト指向 CORBA」ピアソン,(1998)

[3] 小野沢 博文 :「CORBA 完全解説 応用編 -POA を使いこなす-」ソフト・リサーチ・センター,(2000)