

発行・購読モデルを用いたシミュレーション環境の構築*

井部雅章†

Défago Xavier†

小長谷明彦‡

北陸先端科学技術大学院大学 知識科学研究科‡

1 はじめに

本研究の目的は生化学シミュレーションを広域ネットワーク上で並列実行することにより高速化を行える環境を構築することである。対象とする生化学シミュレーションとしては、並列化が可能な問題としてモンテカルロ法による化学反応系の初期条件探索を採用した。このモデルは計算が互いに独立であり、情報を共有する必要がない。このため、並列化に最適な問題であり、高速化が期待できる。また、初期条件探索は多くの初期変数から多くの結果を導き出し、マクロな統計量を計測するものであるから多くの計算能力と時間を必要とし、これらの問題に並列化が有効であると期待される。

シミュレーション本体以外にも、個々のモデルにあわせた GUI の実装を行えるような設計が必要など多くの課題が存在する。

以上のような状況をふまえて分散環境での生化学シミュレーションの運用を行えるように以下のような環境を設計した。部分的な変更ができるように、要素ごとにコンポーネント化を行う。各要素をつなげるインターフェースには発行・購読モデルを採用する。

発行・購読モデルは拡張性・柔軟性が高く、ローカル・エリア・ネットワーク (LAN) 内だけでなく、Grid などの広域ネットワークにも適している [1]。これを利用し、広域ネットワークでも使えるよう設計を行った。開発言語は JAVA を用いた。

2 並列化の方法

分散した環境を構築するうえで、それぞれの要素をつなげるインターフェースは JMS の発行・購読モデル

を用いた。

発行・購読 (Publish-and-Subscribe) モデルとは 1 つのプロデューサーが 1 つのメッセージを多数のコンシューマーに送信するモデルである。

プロデューサーとコンシューマーの間で依存性がないのが特徴で、トピックと呼ばれる仮想チャンネルを間に置き、プロデューサーはトピックに情報を送り、コンシューマーはトピックに情報を受け取りに行く。コンシューマーはトピックの位置を認識していれば良いだけなので、拡張性が高い。1:m 通信用に提案されたモデルだが今回 m:m の通信に用いた。

プログラムをコンポーネント化するにあたっては、EJB (Enterprise Java Beans) を使う。JMS から各コンポーネントを起動するが、それには EJB2.0 から提案された JMS より起動されるメッセージ起動型 Bean を用いた。

3 設計と実装

システムは計算を行うエンジン、初期条件を入力し計算結果を受け取るクライアント、そして両者の間をとりもつブローカーから成る。

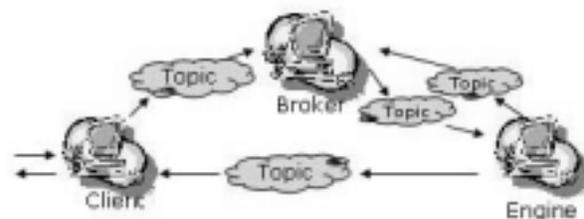


図 1: クライアント / ブローカー / エンジンの関係

動作の流れは、以下の通りである。クライアントから計算要求をブローカーに流し、ブローカーは待機し

* Building Simulation environment by Publish-and-Subscribe model

† Masaaki Ibe, Xavier Défago, Akihiko Konagaya

‡ School of Knowledge Science, JAIST

ているエンジンに初期条件を送る。初期条件を受け取ったエンジンはシミュレーションを行い、クライアントへ送信する。

エンジンの役割はシミュレーションを行うことである。ブローカーから送られてきたシミュレーションの要求に対して、送られてくるメッセージ中の ID 番号を自機のものと同照し、シミュレーションを開始する。また自機の状態を定期的にブローカーに送信する。

クライアントは初期条件を入力し、ブローカーに送信する。その後計算結果を受け取り表示する。

入力はユーザーインターフェースとしてブラウザを利用し、JAVA サブレットで初期条件を受け取る。その初期条件を送信した時点で、クライアントは計算結果を受け取るトピックを必要な数だけ作成する。1 トピックにつき 1 つのエンジンからのみ情報が送られてくるので、計算結果の混同がおきくことを避けることができる。また負荷の高いブローカーへの余計な負荷を避けることができる。

ブローカーは、デザインパターンのオブザーバー・モデルに近い役割を果たす。エンジンとクライアントの間に入り、中継所として依存関係をなくす。

ブローカーはクライアントからの要求を受け取るトピック、エンジンに要求を発行するトピック、エンジンの状態をモニタリングするためのトピックと 3 つのトピックに発行・購読している。

クライアントから送られてきた要求はリストの形でブローカー内に保持される。同時にブローカーはエンジンの状態を別のトピックを通じてモニタリングしており、待機状態のエンジンとクライアントの要求するエンジン数が一致したときエンジンに要求を発行する。

しかしエンジンに指示を送るトピックは 1 つであり発行・購読モデルを利用しているので関係のないエンジンにも命令を送ってしまう。この問題はエンジンに任意の名前をつけ、命令内にエンジン名を照合することにより問題を回避した。

この環境の特徴はこのブローカーを設けたことである。発行・購読モデルを用いたことでクライアントがエンジンを管理する必要がなくなった。そして、エンジン、クライアントともにトピックに接続するだけで自由に追加・終了できる。

4 結果

実装はブローカー用には Pentium3 500MHz Dual、メモリー 512MB の計算機を用い、エンジン用には Pentium3 800MHz Dual、メモリー 1G のマシンを用いた。

JAVA は JDK1.4.1 を利用し、JMS や EJB など J2EE に関しては JBOSS2.4.6[2] を利用した。

実験は約 70 秒かかるシミュレーションを 50 回行い、ブローカーからクライアントまでの時間を計測した。

実験した結果、図 2 のような結果を得た。これにより、オーバーヘッドが少なく、並列化が有効であると言える。

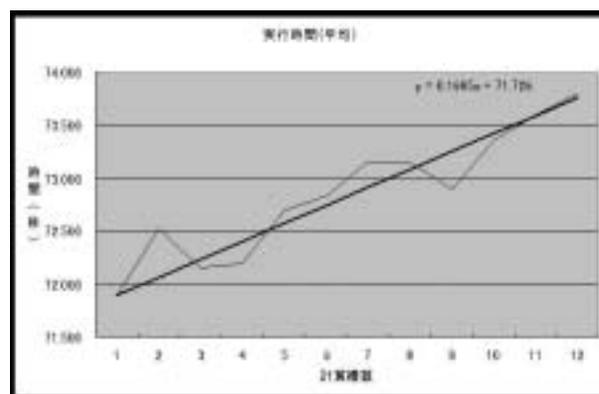


図 2: 並列で動かしたときの実行時間

5 終わりに

本研究により、生化学シミュレーションの分散による並列化が有効であることが証明された。しかしながら、本システムは耐故障性や信頼性の問題を考慮しておらず、改良すべき箇所も多い。またクライアントからの指示に対して待ち行列の適用が可能など発展性も期待できる。

参考文献

- [1] H. Casanova. Distributed computing research issues in Grid computing. *ACM SIGACT News*, 33(3):50-70, September 2002.
- [2] <http://www.jboss.org>.