「高速数値計算の実験」

総合情報学部情報科学科
河野敏行

1. 課題概要

センター実習室の複数の PC を利用して，数値計算を分散させ，大規模な問題を高速に解くことを目標として実験に取り組んだ．昨年度は，学生の利用していない時間帯に，クライアント PC を Linux 系 OS で立ち上げ，その上で，並列計算を試みる実験を考慮し準備を行っていた．この場合，夜間などの学生が利用していない時間を使用することとなるが，多くの問題があり，断念した．そこで，今回は実験室の PC にインストールされている Windows XP の環境で計算環境を構築し，数値計算を行う実験を行った．

全体の構想として，Web サーバを立ち上げ，サーバ上に Web コンテンツとして計算管理システム，計算プログラムを準備しておき，各クライアント PC からサーバに Web アクセスを行うことでブラウザを介して，その PC を 1 クラスタとしてサーバに登録する方法を考えた．管理を行う PC からは計算管理ページを開くことで，各クライアントの状態を管理し，各種の計算の実行・停止・結果の表示などの管理を行う．

計算システム自体は試作段階であるが，簡単な数値実験の結果を示す．

2. 問題設定について

現在，コンピュータの高速化が進み，より複雑な問題を取り扱うことが可能となった．問題が複雑となるに従い，その解析には多くの時間がかかり，1 台の計算機では，解くことが困難となる．特に差分法を用いて得られる次式で示されるような線型方程式を解くことに多くの時間が費やされる．

$$Ax = b$$

ここで，A は $n \times n$ 行列，x は n 次の未知ベクトル，b は n 次の既知ベクトルである．ベクトル b は，真の解 $x^* = (1,1,\cdots,1)^T$ を与え，$b = Ax^*$ を計算することで与え，初期近似ベクトルは $x^{(0)} = (0,0,\cdots,0)^T$ と設定して定常反復法を用いて計算することとする．定常反復法は Gauss-Seidel 法を用い，1CPU の場合とさらに PC クラスタを利用し分散計算をさせた場合とを比較した．一般に，差分法で得られる線型方程式の係数行列は疎な行列となる．すなわち，比較的零要素が多い問題である．今回はテスト問題として，以下に示す A のような行列を扱うこととする．すなわち対角成分をすべて 1.0，上下に 1 ないし 2 つの要素を持つ 3 重または 5 重対角行列とする．

$$A=\begin{bmatrix} 1.0 & p & q & 0 & \cdots & 0 \\ r & 1.0 & p & q & \ddots & \vdots \\ s & r & 1 & p & \ddots & 0 \\ 0 & s & r & 1 & \ddots & q \\ \vdots & \ddots & \ddots & \ddots & \ddots & p \\ 0 & \cdots & 0 & s & r & 1 \end{bmatrix}$$

ここで，p,q,r,s は任意に設定するとする．3 重対角行列の場合は $q=s=0$ とする．

数値実験において扱う係数行列は理論上取り扱いを容易にするために，p,q,r,s を非正とし，$|p|+|q|+|r|+|s|<1$ となるように設定することとする．すなわち，係数行列 A が狭義優対角な Z 行列となる．PC クラスタによる分散処理を行うために係数行列 A を PC クラスタの数によって行の数を分けることにする．そして，それぞれの PC クラスタで得られた近似解を反復ごとにサーバ側で結合し，再度，各 PC クラスタのデータとして与えることとした．すなわち，係数行列 A が n 行，PC クラスタの数が k 台ある場合，1 行目から n/k 行目を 1 番目の PC クラスタが計算することとする．また，割り切れない場合は四捨五入を行い，最後の PC クラスタにおいて計算する行を調整することとした．並列計算の方法としてはさまざまな方法がある[1]が，今回は複雑な処理はせず，毎回の計算で確実に処理できる方法を選んだ．この方法では，サーバと PC クラスタの通信が多く，効率的ではない．また，反復計算に前処理[2]を用いることによって高速化することも可能であるが，今回の実験では省略した．

3.　計算機環境

この研究では Windows 上で分散計算を行う．また，クライアント側に新たにアプリケーションを追加することなく，計算を行うこととしたい．したがって 1 台の Web サーバを立て，そこに Web アクセスする各 PC を PC クラスタとして登録し，分散計算を行い，その結果を管理 PC で集約し管理する．その構成を図 1 に示す．

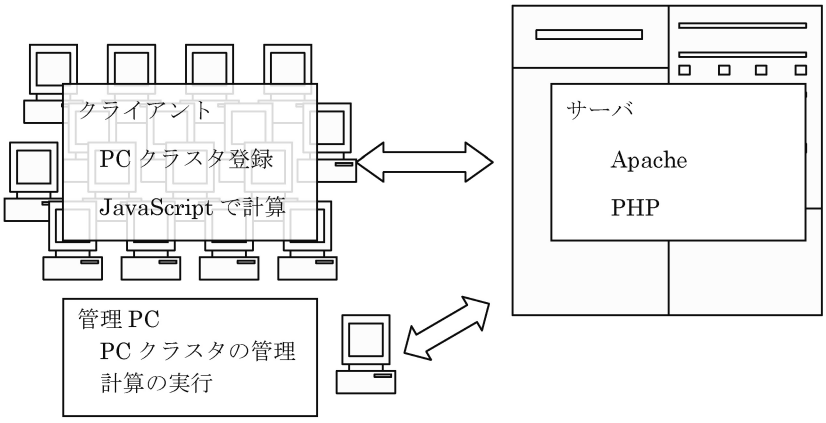

図 1　サーバとクライアント，管理 PC

図 1 において，サーバには Web サーバとして Apache を利用し PHP で各プログラムを作成した．クライアント側では JavaScript で記述されたプログラムで計算が行われるが，これらはサーバ側で PHP によって各 PC クラスタに合わせて生成されたプログラムである．通信は通常の Web 接続（80 番ポート）を利用し，サーバに保存された数値データを用いて計算を行っている．

以下に計算の流れを箇条書きで示す．

○サーバ及び管理 PC 上

- 全体のデータを処理
- アクセスした PC に自動的に番号を割り振る
- 計算プログラムを配信する
- ①各データを配信し，計算をさせる
- ②計算結果を集約し，データを更新する
- ①②を収束するまで繰り返す．
- 結果を表示

○　PC クラスタ

- サーバにアクセス
- 計算プログラムを受け取る
- データを受け取り，計算した結果を返す
- 終了まで繰り返す

サーバにおいて各 PC クラスタの計算結果を受け取る際に，各 PC の処理能力や通信のタイミングなどの影響で，割り当てた PC の順序ごとには最新の結果が得られないため，計算の済んだ PC から順に値を受け取るようにした．現在はデータの受け渡しのタイミングにおいて余裕を持たせるように処理に間隔を空けるための時間制御のパラメータを設定している．そのため，計算時間として余分にかかってしまうがデータへの無駄なアクセスを減らすことが期待できるはずである．このパラメータは環境によって，最適に設定することで，計算時間を短縮することが可能であるが，この設定は大変難しい問題であり，今回は適当な値を与えている．

4.　計算実験

11 号館 6 階第 1 実習室において，4 台の PC を利用し計算プログラムの実験を行った様子を図 2 に示す．図 2 は 4 台の PC において研究室にインストールされたサーバへアクセスし，クライアント登録を行って実験を行った様子である．左の PC では計算の管理画面を開き，管理 PC としても登録された状態である．拡大したものを図 3 に示す．

図 2　実験の試験段階の様子

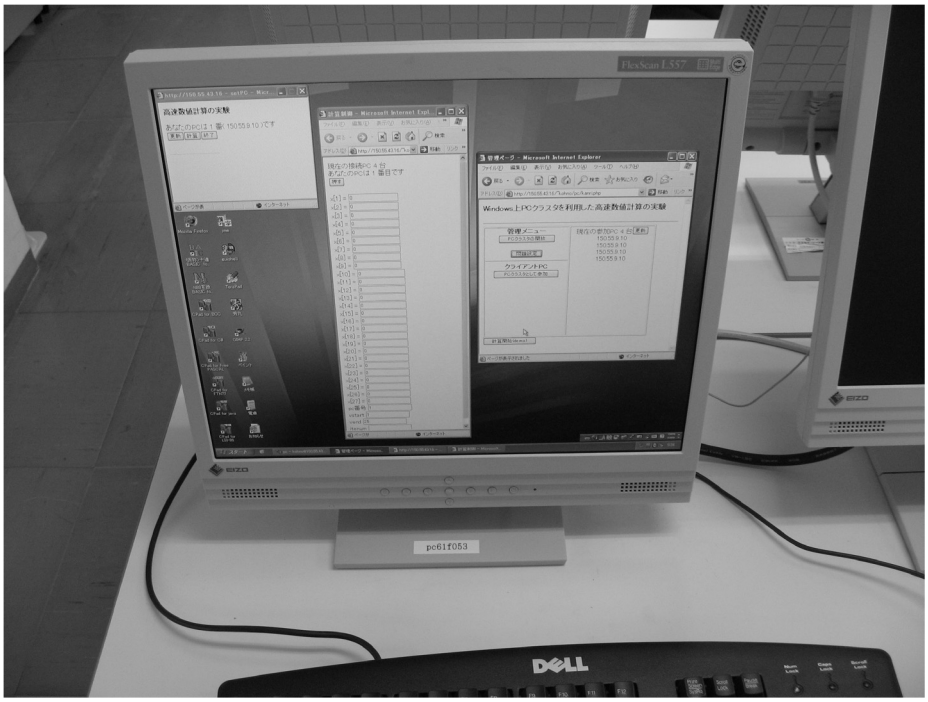

図 3　クライアントにおける PC クラスタ画面と管理画面

図 3 において中央に表示した縦長のウインドウは計算過程をチェックするために計算で得られた近似値を表示している様子である．左側に PC クラスタのプログラムが開いており，右側に管

理メニューを表示している．管理メニューでは，PC クラスタの開放，問題の設定，計算開始のボタンなどを用意している．このように，PC クラスタと管理プログラムを 1 台の CPU で起動することも可能である．

実験としては小さな問題しか扱っていないが，所属する情報科学科の実験室を用いて表 1 の環境の下，表 2 の結果を得た．

表 1　計算環境

実験問題	次数　100 3 重対角行列 $p=-0.4, q=0, r=-0.3, s=0$ 収束判定条件　$\left\|x^{*}-x^{(k)}\right\|_2 \le 0.0001$
計算 PC	Celeron 1.7GHz　Windows XP

表 2　計算結果

CPU 数	計算時間(秒)	備考
1	0.2124	サーバ上で PHP によりかかれたプログラムで計算
4	0.1848	PC クラスタ 4 台，管理 1 台をサーバにアクセス
5	0.1671	PC クラスタ 5 台，管理 1 台をサーバにアクセス

サーバ機には同スペックの PC に Apache_2.0.53　win32 版をインストールし，設定を行った．表 2 において，CPU 数 1 台の場合，PHP プログラムのみで実行されており，PC クラスタを利用した結果ではない．したがって，比較としては適切ではないが参考までに示した．次に，CPU 数を 4 から 5 に増やした場合，クライアントでは JavaScript で計算がされ，サーバ側で PHP において処理されるという同じ環境である．4 台の場合は 1 台あたり 25 行，5 台の場合は 1 台あたり 20 行の計算を各クライアントが計算している．この場合，計算時間が減少した結果が得られた．しかしながら，同様の実験を繰り返し行った場合に，計算時間は変化することがある．おそらくはネットワーク通信上の遅延が原因であると思われるが，より詳細な実験が必要であると考える．CPU の台数を増やした場合，通信エラーが起こすことがあり，プログラムの修正が必要であることがわかった．

5.　考察

今回の実験において，プログラムの試作段階まで行うことができた．プログラムは PHP と JavaScript で作成し，現在のところ Internet Explorer 6 上での動作確認をおこなった．PHP はサーバ側で動作し，JavaScript はクライアント側のブラウザ上で動作する．OS に依存する部分が少ない環境での分散計算環境が構築できた．さらに，最近ではガジェットの利用も増えている

ので，今後，ガジェットとしての開発も考えられる．より多くの PC に起動と同時にアプリケーションを登録しておくことで, サーバからの計算命令で各 PC の余剰している CPU パワーを利用して計算を行うことが可能であると考える．

OS に依存することなく，分散処理を行うプロジェクトとしては，BOINC が知られている[3]. BOINC とはボランティア・コンピューティングとデスクトップ・グリッド・コンピューティングのためのオープンソース・ソフトウェアである．データとしては膨大な量を扱うために，1 台や数台の PC でその PC 間の通信のロスが計算のロスにつながるというものでは無く，インターネット上に存在する膨大な数の PC，もちろん Windows，Mac などの OS に依存しない PC を利用して計算を行い，何百年もかかってしまう計算を短期間で行うものである．このような分散環境で計算を行うためには，解くべき問題を適切に処理できる形に最適化を行う必要がある．

今後の課題として，より複雑な問題を領域分割など行い簡略化しその部分ごとの計算を各クラスタが行うような計算環境を構築することが必要であるといえる．また，各計算を高速化するために，前処理の適用を今回は出来なかったが，前処理の導入と，さらにはこのような分散処理に合わせた前処理を導出することが今後の課題といえる．プログラムにおいては，データの更新においてデータ量を最小限に抑えることや，計算のタイミングの調整など多くの課題が残されている．

参考文献

[1] 小国力訳，J.J.Dongarra, I.S.Duff, D.C.Sorensen, H.A. van der Vorst，コンピュータによる連立一次方程式の解法，丸善株式会社,1993.

[2] 河野敏行，適応型前処理付反復法について，日本応用数理学会論文誌 15,3(2005)235-243.

[3] BOINC, http://boinc.berkeley.edu/