2X-8

記憶階層における巨大偏微分方程式の数値計算について

佐藤隆士
(詫間電波高専　情報工学科)

1. はじめに

数値計算において偏微分方程式を解く機会は多い．しかし，2次元以上の高次偏微分方程式では，現在の大型あるいはスーパーコンピュータにおいても，主記憶容量が十分であるとはいえない．最近使用される機会の多くなった，パソコン，ワークステーションにとってはなおさらのことである．したがつて，大型の偏微分方程式を解くためには，補助記憶にたよらなざるを得ない．

スーパーコンピュータにおいては，仮想記憶管理のもとで主記憶外のデータにアクセスすることはできないので，FORTRANのREAD，WRITE文を用いて，高速補助記憶装置のデータにアクセスしている．高速補助記憶装置としては，半導体素子の拡張主記憶などが用いられ，記憶階層間のデータ転送の効率化が計られているが，データ転送はスカラ計算になるため，計算のスーパーコンピュータに対する適合性の指標であるベクトル化率を低下させている．スーパーコンピュータの計算能力は，今後とくにベクトル演算装置の改良により高められることが期待されているが，ベクトル化率の低下により，スーパーコンピュータの処理能力は，かなり敏感に影響を受けることが予測されている．

また，大型汎用コンピュータをはじめワークステーション，パソコンなどにおいては，補助記憶装置のデータ転送スピードがスーパーコンピュータの高速補助記憶装置にくらべ格段に遅いだけに，記憶階層間のデータ転送を伴う数値計算はほとんど不可能となっている．

本稿では，以上のような状況をふまえ，主記憶上では解けない巨大偏微分方程式を記憶階層環境のもとで効率的に解く方法を述べ，さらにパソコンでの実測結果を示す．

2. 定義

本稿では，時間独立の偏微分方程式の数値計算において全格子点データが主記憶に納まらない場合を扱う．

2次元偏微分方程式の数値計算において，$U_{ix,iy}^{it}$ を格子点（ix，iy）の時刻 it のデータとし，これを計算するのに，必要となる1ステップ時間前の格子点データを $\{U_{ix',iy'}^{it-1}\}$（複数データ）とするとき，max{|ix−ix'|，|iy−iy'|}をこの解法の近傍半径（r）という．同様な考え方は，2次元以外の場合にも拡張できる．

偏微分方程式の数値計算において，空間は格子で分割されるが，この際の格子点を空間格子点という．同じ空間格子点でも，時刻によってその値は異なる．空間格子点に時刻を指定したものを時空間格子点という．例えば，2次元偏微分方程式の数値計算において，（ix，iy）は空間格子点であり，これに時刻を加えた，（ix，iy，it）は時空間格子点である．

3. がけくずし法

巨大偏微分方程式を解く際，主記憶，2次記憶からなる記憶階層間のデータ転送を少なくする方法（がけくずし法[1]）を説明する．この方法は，周期的境界条件では使用できないが，比較的単純なアルゴリズムで，わかりやすく，しかも応用範囲も広いものと思われる．以下，2次元の場合について説明するが，他の次元にも容易に拡張できる．

空間格子点を正方形領域に分割する．分割された各領域をブロックといい，主記憶のデータ領域は1ブロック中の空間格子点データの2倍の容量をもつものとする．そして，原点に近いブロックから順にブロック単位に処理する．まず，主記憶上の1ブロック分の初期値をもとに，1時間ステップ分計算する．計算できる領域は，ブロックの境界にあるr個分だけ少なくなっている．次に，

Numerical Calculation of Large Partial Differential Equations on Memory Hierarchy
Takashi SATO
TAKUMA National College of Technology

境界値あるいは以前に処理されたブロックの計算結果を用いることによって，ブロックの座標値の小さい境界の２辺についてｒだけ減少方向に拡大する．同時に，座標値の大きい境界の２辺をｒだけ縮小する．今計算した結果と，拡大された境界のデータをもとに次の時間ステップをする．以下同様なくり返しで，ブロックは処理とともに，ｘ，ｙ軸の減少方向に移動していく．

４．　実測結果

数値計算は，１次元と２次元の熱伝導方程式について行った．表１，２に入出力データ量と計算時間（経過時間）の実測値を示す．表中，ｎｏｒｍａｌ　ｓｃａｎは転送データ量の減少を意識しない普通のアルゴリズムによるものである．計算機はＮＥＣ　ＰＣ－９８０１（数値演算プロセッサ付）で，２次記憶としてハードディスク装置（アクセス時間８５ｍS）を使用した．

A］　１次元

１次元の場合，主記憶にはいりきらないような巨大偏微分方程式になることはまずありえないが，ここでは意識的に主記憶上の小さい領域を使用して計算を行った．測定パラメータは表１に示したとうりである．ここで，n_xはｘ方向の格子点の分割数，n_tは時間方向の分割数，P_lはブロックの一辺の大きさである．１格子点のデータは８Ｂ（倍精度）とした．

表１　１次元の場合の実測値

パラメータ			normal scan		がけくずし法	
n_x	n_t	P_l	転送量	計算時間	転送量	計算時間
1000	30	100	480kB	７９秒	19kB	１４秒
1000	100	100	1600kB	２７７秒	64kB	３６秒

表２　２次元の場合の実測値

パラメータ			normal scan		がけくずし法	
n_x,n_y	n_t	P_l	転送量	計算時間	転送量	計算時間
100	10	50	1600kB	３３９秒	128kB	４４秒
100	30	50	4800kB	１０９２秒	384kB	１２２秒
100	10	25	1600kB	４９４秒	256kB	６８秒

B］　２次元

２次元の場合，図１に示す　ｉｂ＝ｉｂｘ＋ｉｂｙ　の値が小さい順に処理するアルゴリズムを用いた．２次記憶には同じｉｂ値のブロックについて座標の大きい２辺の境界値をまとめて１つのファイルとした．図の実線と破線について２個のファイルを用意すればよく，ｉｂ値の奇偶によって入出力を切り替えてやればよい．使用計算機では，大配列をとることが困難なため，２次元に対しては主記憶のデータ領域はかなり窮屈なものとなっている．表２にパラメータとともに，転送データ量および実測データを示す．

５．　むすび

記憶階層のもとで，効率よく巨大偏微分方程式を数値計算する方法を述べ，ＮＥＣ　ＰＣ－９８０１で実測した結果を示した．文献[1]にも示したように，２次元の場合主記憶のデータ容量の平方根に比例してデータ転送量が減少するので，大きい計算機ほどより大きな効果が期待できる．

〈参考文献〉

［１］佐藤：巨大偏微分方程式の数値解法において記憶階層間のデータ転送を少なくする方法，情処全大(1986.3)，７Ｅ－３．

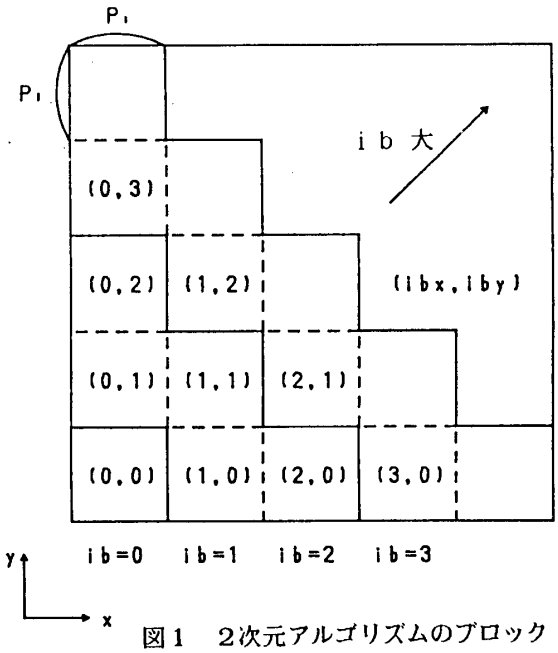

図１　２次元アルゴリズムのブロック