数理解析研究所講究録
第514巻 1984年 135-149

格子ゲージ理論に現れる大規模連立一次方程式の不完全LU分解CR法とその並列化

筑波大・電子情報　小柳義夫
(Yoshio Oyanagi)

1．量子色力学と格子ゲージ理論

量子色力学（QCD，Quantum Chromodynamics）は、ハドロン（陽子、π中間子など）を支配する基本的な力学であると考えられている。QCDによれば、クォークは香り（Flavor：ストレンジネス、チャーム、ビューティー、トップなど）と色（Color：赤青白または赤青緑）の自由度を持ち、グルオンは色の自由度を運ぶことにより、クォーク間の力を媒介する。QCDは、クォークとグルオンのSU（3）ゲージ理論であり、3種の色の間の時空依存の混合（ユニタリー変換）に対しラグランジアンは全く不変である。ゲージ理論の基本的な性質を知るために、色を2種に減らしたSU（2）ゲージ理論もしばしば研究されている。

これまでの場の理論の計算は、湯川の中間子論や、朝永のくりこみ理論など、すべて摂動論によって行われて来た。摂動論は、結合定数gの冪級数で展開するので、相互作用が弱い場合に有効である。QCDは、QED（量子電磁力学）などとは異なり、漸近的自由という性質を持っている。すなわち近距離ほど相互作用が弱い。したがって、高エネルギー散乱のような近距離（$\sim 10^{-14}$cm以下）の現象に対しては、摂動論による計算が有効である。しかし逆に、クオークがなぜ単独で外に飛び出さないか（”閉じ込め”）とか、ハドロンの質量というような本質的に長距離（$\sim 10^{-13}$cm程度）の現象に

ついては、相互作用がどんどん強くなってしまうので、摂動論は無力である。

K. Wilson は、ゲージ理論を4次元超立方体格子上において定式化することを提唱した。これを格子ゲージ理論という。格子ゲージ理論では、クォークは格子点上に、グルオンはリンク（辺）上におかれる。しかし、格子構造は場の理論を定式化するために導入した道具であって、真の空間はあくまで連続であるから、物理的に意味のある量を求めるには、格子間隔 $a \to 0$ の極限を取らなければならない。この際同時に $g \to 0$ とする。対応する統計力学では低温極限 $T \to 0$ に対応する。低温では相関長 ξ が長い（a を単位として）ので、大きな格子の上でシミュレーションを行わなければならない。例えば、$4^3 \times 8$ というような小さいサイズの格子では、$a \sim 0.3 \times 10^{-13}$ cmの粗さでしか物が見えない。

なぜモンテカルロ法で場の量子論の計算ができるかというと、Feynman の経路積分のよって、量子力学をいわば古典的な軌道の重ね合せで表現することができるからである。場の量を $\phi(x, t)$ で表わすと、ϕ によって定義される任意の物理量 $Q[\phi]$ の期待値は

$$\langle Q[\phi] \rangle = \frac{\int Q[\phi] \exp\left(\frac{i}{g} S[\phi]\right) D[\phi]}{\int \exp\left(\frac{i}{g} S[\phi]\right) D[\phi]}$$

という汎関数積分（経路積分という）によって与えられる。$S[\phi]$ は作用で、ラグランジアン密度を時間空間にわたって積分したものである。

ここでユークリッド化

時間　　$t \rightarrow -i\tau$

$x^2+y^2+z^2-t^2 \rightarrow x^2+y^2+z^2+\tau^2$

を行うと、Qの期待値は

$$\langle Q[\phi] \rangle = \frac{\int Q[\phi]\exp(-S[\phi]/g)\, D[\phi]}{\int \exp(-S[\phi]/g)\, D[\phi]}$$

とかける。他方、温度Tでの統計平衡におけるQの期待値は、

$$\langle Q[\phi] \rangle = \frac{\int Q[\phi]\exp(-H[\phi]/kT)\, D[\phi]}{\int \exp(-H[\phi]/kT)\, D[\phi]}$$

であるから、H（エネルギー）をSと読み直せば、両者は形式的に同値である。すなわち、d次元時空のユークリッド化された場の理論は、d次元空間の統計力学と同値である。また、結合定数gは温度Tに対応している。

統計力学のシミュレーションにおいては、exp(－H／kT)に比例した確率で配位を生成して物理量の期待値を求めるが、場の理論では exp(－S／g)に比例した確率で軌道（全時空におけるϕの値の組）を生成する。

2．Wilsonのフェルミオン作用

クォークのようなフェルミオン場ψに対するWilsonの作用

は、グルオンの作用をS(U)としたとき、

$$S = S(U) + \Sigma_{ij} \bar{\psi}_i M_{ij} \psi_j$$

と書かれる。ただしi, jは格子点の番号であり、M_{ij}はそれ自身12×12（SU(3)の場合）または8×8（SU(2)の場合）の行列である。

$$\begin{aligned}
M_{ii} &= I \\
M_{ij} &= K(I - \gamma_\mu) U_{ij} \qquad \text{if } j = i + \hat{\mu} \\
M_{ij} &= K(I + \gamma_\mu) U_{ij} \qquad \text{if } j = i - \hat{\mu} \\
M_{ij} &= 0 \qquad\qquad\qquad\qquad \text{otherwise}
\end{aligned}$$

ここで、γはDiracのγ行列（4×4）でスピノルの自由度に対応し、U_{ij}は格子点iとjとを結ぶリンク変数（2×2または3×3の特殊ユニタリー行列）である。すなわち、行列Mは格子座標、スピノルの足、色の自由度の3重のブロック行列である。Kはhopping parameterと呼ばれるもので、クォークの質量に関係し、0-0.25の実数である。クォークが重いほどKは0に近い、すなわち隣の格子点に飛びにくい。行列Mは、隣同志の格子点iとjにたいしてのみ値をもつ。さらにMはエルミート対称ではない。図1にMのパターンを示す。黒い点が1つのブロックを表す。単純な帯行列にならないのは、周期境界条件のためである。

経路積分では、ψはGrassmann数であるが、2次形式なので積分ができ、

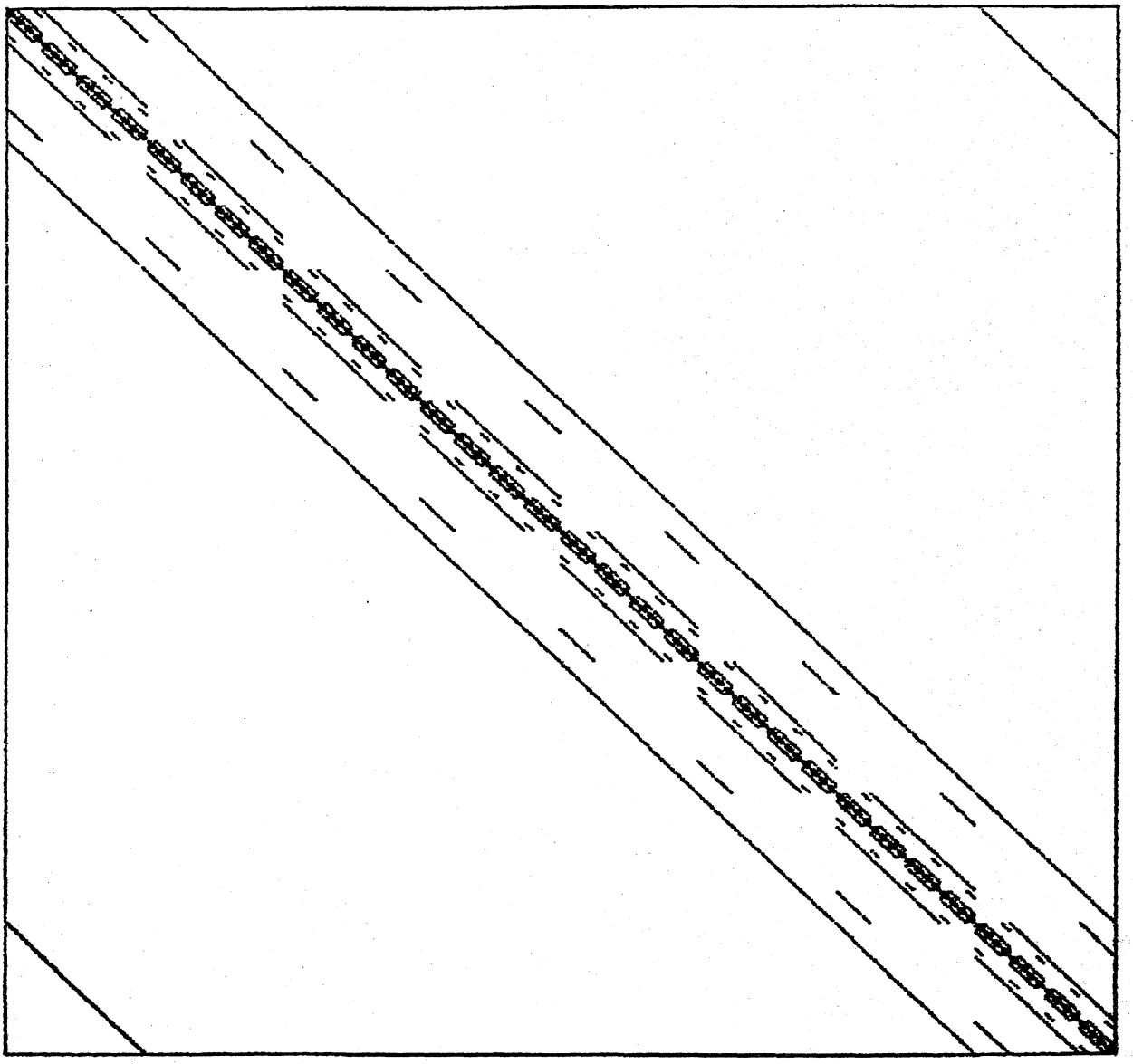

Fig. 1 The pattern of non-zero blocks of matrix M.
The lattice size is $4^3 \times 8$.

$$\int \exp(-S)\, d[\psi]\, d[\bar{\psi}] = \det(M)\exp(-S(U))$$

となる。よく知られているように、ψが通常の数の場合は2次形式を積分するとdet(M)$^{-1}$がでるが、Grassmann数ではdet(M) そのものとなる。

しかし、行列式の値を計算することは大変なので、普通はクェンチ近似(Quenched approximation)を用いる。これはdet(M) = 1 とおく近似であり、物理的にはクォークの生成を無視したことに対応している。 det(M)をKで展開すると 1 + O(K^4) となるので、Kが小さいときにはよい近似であると考えられる。

π中間子や、陽子の質量を求めるためには、クォークの伝幡関数

$$\langle \psi_i \bar{\psi}_j \rangle = \int d[U] (M^{-1})_{ij} \exp(-S(U))$$

を計算しなければならない。しかし、Mの逆行列を全部求めるのは不可能なので、実際はi = 1に対してのみ計算する。すなわち12行の右辺について、連立1次方程式を解く。これから、ハドロンの質量を計算する。

従来は Gauss-Seidel法、SOR法などが用いられてきた。これに対し、われわれのグループは、1次方程式を微分方程式

$$dx(t)/dt = b - Mx(t)$$

に置き替え、これを2次のRunge-Kuttaで解く方法を開発し、かなりの成果を収めてきた。しかし行列Mの固有値の実部が0に近づくと、収束が遅くなる。hopping parameter Kが大きい(すなわちクォークの質量が軽い)ところまで計算する

には、別の方法が必要である。

3．ブロック不完全ＬＵ分解

この節では、行列Ｍの特異な性質により、Ｍの不完全ＬＵ分解が、何ら余分な計算なしに得られることを示す。ただし格子の一辺の長さは３より大きいとする。

ブロック下三角行列Ｌと、ブロック上三角行列Ｒ（Ｕはリンク変数に使ったのでＲを用いる）を次のように定義する。

$$L_{ij}\begin{cases} = M_{ij} & (i \geq j) \\ = 0 & (i < j) \end{cases}$$

$$R_{ij}\begin{cases} = M_{ij} & (i \leq j) \\ = 0 & (i > j) \end{cases}$$

すると行列Ｍは

$$M = LR - N$$

と分解される。ただしＮはＬＵ分解の誤差である。

このＬＵ分解が隣同志の格子点 i,jについては正確であること、つまり

$$N_{ii} = 0 \quad \text{かつ}$$

$$N_{ij} = 0$$

であることを証明する。証明のポイントは、３つの格子点ｉ，ｊ，ｋが同時に隣同志にはなれないということ、すなわちもしｉとｊとが隣ならば、必ず$M_{ik} = 0$または$M_{kj} = 0$の少なくとも一方が成立していることである。

まずｉとｊは隣同志かつｉ＜ｊとすると、

$$(LR)_{ij} = \sum_{k=1}^{i-1} M_{ik}M_{kj} + M_{ii}M_{ij} = M_{ij}$$

逆に i > j なら

$$(LR)_{ij} = \sum_{k=1}^{j-1} M_{ik}M_{kj} + M_{ij}M_{jj} = M_{ij}$$

また i = j なら、

$$(LR)_{ii} = \sum_{k=1}^{i-1} (I + \gamma_\mu) U_{ik} (I - \gamma_\mu) U_{ik}^{+} + M_{ii}M_{ii} = I$$

Nは隣の隣の格子点の間にO(K²)のオーダーの要素を持つ。

4. ILU分解を用いた種々の反復解法

ブロック不完全LU分解を用いた解法は色々考えられる。

a. 単純修正法

最も簡単なのは、残差を不完全LUで修正する方法であるすなわち、

$r_0 = b - Mx_0$; i = 0

repeat until converge

$x_{i+1} = x_i + (LR)^{-1} r_i$

$r_{i+1} = b - Mx_i$

i = i + 1

主な演算は、Mx, $L^{-1}x$, $R^{-1}x$である。三角方程式の解法は、重ね書きできるので、必要な記憶領域は、2個のベ

クトルxとrである。

ただし収束の保証は無い。実際、Kが小さい時には収束するが、Kが大きいと発散してしまう。

b．ＩＬＵＭＲ法（ILUCR(0)）

発散を防ぎ、収束を安定化するには、xの修正量を、残差が最小になるように選べばよい。すなわち、

$r_0 = b - Mx_0$；　$i = 0$

repeat until converge

$p_i = (LR)^{-1} r_i$

$\alpha = (Mp_i, r_i) / (Mp_i, Mp_i)$

$x_{i+1} = x_i + \alpha p_i$

$r_{i+1} = r_i - \alpha Mp_i$

$i = i + 1$

必要な記憶領域はx，r，p，Mp。前の２倍である。

残差のノルムは増大しないが、αが０になると、無限ループに入る。実際Kが臨界値になると、αは０に近くなる。一般にαは複素数であるが、この問題では対称性により実数である。計算誤差により虚数がでてくるが、αの計算では実部だけとっている。

主な演算は、Mx，$L^{-1}x$，$R^{-1}x$である。すなわち、単純修正法とほぼ同じである。

c．ＩＬＵＣＲ（ｋ）法

これは、Saadの共役残差法に、不完全ＬＵ分解によるPre-

conditioningを施したものである。

$r_0 = b - Mx_0$;　$i = 0$

repeat until converge

$p_i = (LR)^{-1} r_i + \beta_1 p_{i-1} + \cdots + \beta_k p_{i-k}$

$\alpha = (Mp_i, r_i) / (Mp_i, Mp_i)$

$x_{i+1} = x_i + \alpha p_i$

$r_{i+1} = r_i - \alpha Mp_i$

$i = i + 1$

$\beta_1 \cdots \beta_k$は、p_iが$p_{i-1} \cdots p_{i-k}$とM^+Mに関して共役になるように、きめる。

必要な記憶領域は、r，xと，（k＋1）組のpとMp。

主な演算は、やはりMx，$L^{-1}x$，$R^{-1}x$で、演算の量はkにはあまり依らない。

ＩＬＵＣＲ（１）の典型的な計算時間

（HITAC　M200H（KEK）による。）

○サイズ　$4^3 \times 8$　　ＳＵ（２）　　（４０９６）

０．８３　秒／反復

○サイズ　$10^3 \times 12$　ＳＵ（２）　　（９６０００）

２１．１　秒／反復

括弧内は、方程式の次元数。

最も重要なのはKが大きく臨界値に近い場合なので、実用的なのはＩＬＵＣＲ（１）である。ｋを増やせば収束は早まるかも知れないが、記憶容量の点で非現実的である。

５．ベクトル化

まず元のプログラム（ＹＯと呼ぶ）を示す。このプログラムでは、冗長を避けるために、行列の掛け算Ｍｘと、三角方程式の解法$L^{-1}x$（前進代入），$R^{-1}x$（後退代入）を一本化してある。すなわち、模式的に書けば、

```
      SUBROUTINE MULT(Y,U,X)   掛け算
      LFB=0; KK=-K; GO TO 1
C
      ENTRY      LINV(Y,U,X)   前進代入
      LFB=1; KK=K;  GO TO 1
C
      ENTRY      RINV(Y,U,X)   後退代入
      LFB=-1; KK=K; GO TO 2
C
1     N4B=1; N4E=NT; N3B=1; N3E=NZ; NI=1
      N2B=1; N2E=NY; N1B=1; N1E=NT; GO TO 3
C
2     N4B=NT; N4E=1; N3B=NZ; N3E=1; NI=-1
      N2B=NY; N2E=1; N1B=NX; N1E=1; GO TO 3
C
```

```
3     DO 80 N4=N4B,N4E,NI; DO 80 N3=N3B,N3E,NI
      DO 80 N2=N2B,N2E,NI; DO 80 N1=N1B,N1E,NI
      NN=(((N4-1)*NZ+N3-1)*NY+N2-1)*NX+N1
      Y(NN)=X(NN)
C     L=1                              x 方向の処理
      NR=NN+NUPSFT(N1,1)               右の格子点番号
      NL=NN-NDNSFT(N1,1)               左の格子点番号
      L1=4*NN-3                        右のリンク番号
      L2=L1-4*NDNSFT(N1,1)             左のリンク番号
      IF(LFB*(NN-NR) .GE. 0)
     1    Y(NN)=Y(NN)+KK*U(L1)*X(NR)
      IF(LFB*(NN-NL) .GE. 0)
     1    Y(NN)=Y(NN)+KK*U(L2)*X(NL)
以下同様の式がL=2,3,4 について続く
80    CONTINUE
```

最後の下線を付けた二式は、本当は各々8本（SU(2)）または12本（SU(3)）からなり、右辺ももうすこし複雑である。

このプログラムをベクトル化するには、種々の困難がある。一つの問題は、NRやNLなどがN1..N4の線形関数ではなく、いわばランダムであることである。この点については、HITAC S/810 や FACOM VP/200 などでは、リストベクトルを用いることができるので、解決できる。もう一つの問題は、三角方程式の場合、配列YとXは同一であり、直前の計算結果が右辺に現れていることである。これは、本質的な問題であり、アルゴリズムを根本的に書き変える必要がある。

この問題の場合、アルゴリズムの構造は偏微分方程式の差

分法と同一であるので、超平面法によってリストベクトル化が可能になる（後氏の示唆による）。すなわち、格子点のx, y, z, t座標をそれぞれN1, N2, N3, N4としたとき、N1+N2+N3+N4=一定　の点は、互いに独立であり、並列に計算できる。従って、プログラムは次のようになる。

```
1     IB=4; IE=NSUM; II=1; GO TO 3
2     IB=NSUM; IE=4; II=-1; GO TO 3
C
3     DO 80 IS=IB,IE,II        但し IS=N1+N2+N3+N4
*VOPTION VEC
      DO 80 J=NB(IS-1)+1,NB(IS)
      NN=NLIST(J)              前のNNより1少ない。
      N4=NN/(NX*NY*NZ)+1
      N2=MOD(NN/NX,NY)+1
      N1=MOD(NN,   NX)+1
      N3=IS-N1-N2-N3
      NN=NN+1

      以下は前と同じ
```

こうして修正したプログラム（Y2と呼ぶ）を日立のベクタイザーにかけて解析した。$4^3 \times 8$格子、SU(2)、K=0.12, 0.13の場合では、

DOループ率	99.620%
ベクトル化可能	99.214%
ベクトル化不能	0.406%

という、驚くべき結果を得た。この場合、超平面法のループの平均長は３０である。

ベクタイザーはよかったのであるが、このプログラムをＦＯＲＴ７７／ＨＡＰのコンパイラーにかけたところ、最内側ループが長すぎてベクトル化してくれない。複素変数、３次元配列、かつリストベクトルで、最内側ループがラインプリンター５枚というのは、コンパイラーにとってちょっと難物のようである。ただし、基本的には、１文毎にベクトル化すればよいので、将来の改善に期待する。

そこでやむを得ず、最内側ループを手で５６個のＤＯループに細分し、やっとベクトル化することができた（Ｙ４と呼ぶ）。同一の問題（$4^3\times8$格子、ＳＵ（２）、Ｋ＝0.12...,0.165）に対する計算時間（CPUおよびSPU,単位秒）は：

計算機	Ｙ０	Ｙ２	Ｙ４
M200H （KEK）	153.16	181.09	254.53
S810/20（東大）	121.58	120.14	24.93

Ｓ８１０の時間で比較すると、約５倍の高速化を達成した。しかしＭ２００Ｈの時間を見ると、ＤＯループを細分化したことによるオーバーヘッドがかなりあるものと思われる。もしＹ２が完全にベクトル化できれば、18秒程度まで改善されるのではないかと期待している。また、実際的な問題では格子のサイズは$10^3\times20$（平均ループ長 400）以上であるから、高速化率はさらに上がるであろう。これは、２４万元の複素方程式である。

以上のように、ＩＬＵＣＲ法は格子ゲージ理論のフェルミ

オン問題に極めて有効であり、スーパーコンピューターの能力を最大限に発揮しうるアルゴリズムである。

本講演にあたり、福来正孝氏、金子敏明氏はじめ、格子ゲージ理論モンテカルログループの多くの方々の協力を得た。なお、この研究の一部は、東京大学大型計算機センターからの「スーパーコンピュータS－810／20の試用依頼」に基づいて提供された計算時間を利用した。