JOURNAL OF COMPUTATIONAL PHYSICS 67, 28–36 (1986)

An Improved Heat Bath Method for Vectorized Processing

HONG-QIANG DING

Department of Physics, Columbia University,
New York, New York 10027

Received June 20, 1985; revised November 22, 1985

A heat bath combined with a Metropolis method for updating the links of lattice gauge theory is proposed. It is a variation of the method of N. Cabibbo and E. Marinari [*Phys. Lett. B* **119**, 387 (1982)], but is better suited for vectorized programs and parallel processors. It preserves the essential features of that method, but runs at a slightly faster speed. © 1986 Academic Press, Inc.

Recently, calculations of $SU(3)$ gauge theory have been carried out on large size lattices by many groups [1]. All of them used vectorized programs on a mainframe computer or parallel array processors in SIMD mode. Several projects to build a special purpose array processor for fast computations in lattice gauge theory are in progress [2, 9]. There are two commonly used algorithms to update the link variables: the Metropolis [3] method and a heat bath method proposed by Cabibbo and Marinari [4] (hereafter called C–M heat bath). It is known that the C–M heat bath algorithm runs faster than the Metropolis algorithm, and that the correlation of successive sweeps using heat bath updates is about one-half of that of Metropolis updates [4, 10]. Taking this effect into account, the heat bath algorithm runs about three times faster than the Metropolis algorithm.

A problem arises when applying the C–M heat bath method in vectorized processing. In a vectorized program, many links are updated synchronously in order to speed up the update. The C–M method involves an accept–reject procedure which may have to be repeated many times until a trial is accepted. This probabilistic procedure makes the synchronous update inefficient. Let us first describe it in the context of the $SU(2)$ case, since the $SU(3)$ case is similar. The simple and efficient heat bath algorithm, introduced by Creutz [5], generates the random variable a_0 according to the distribution

$$\sqrt{1-a_0^2}\, e^{\beta k a_0}. \tag{1}$$

We have parametrized the $SU(2)$ matrix as $a = a_0\mathbb{1} + i a \cdot \boldsymbol{\sigma}$, where σ^i are the three Pauli matrices, $\beta = 4/g^2$, and k is the determinant of the sum of environment link

0021-9991/86 $3.00

products. This is done in two steps: the first step is to generate a random number x_1 uniformly distributed in the interval [0, 1]. A trial a_0 is obtained by solving

$$x_1 = Z^{-1} \int_{1}^{a_0} e^{\beta k \ell}\, d\ell$$

where Z is the normalization constant. The second step is to accept this a_0 with the probability $\sqrt{1-a_0^2}$. Generate a second random number x_2 in the interval [0, 1]. If x_2 is less than or equal to $\sqrt{1-a_0^2}$, accept this a_0. If not, go back to the first step to try a new a_0. Do this accept reject iteration until a trial of a_0 is accepted. Since the accept–reject iteration is probabilistic, the number of iterations of this accept–reject procedure varies from one link to the next. For a vectorized program or synchronous SIMD processing with a large number of synchronous updates, this means that while one particular update is being iterated many times, all other link processes which have already finished the accept–reject are waiting, doing nothing. This happens almost all the time. For example, for the $SU(3)$ gauge theory with $\beta = 6.0$ and three $SU(2)$ subgroup updates, the average number of iterations is about 3.2. This implies that in a 16-node parallel processor SIMD update, an average number of 10 iterations will be required by the slowest processor (see Appendix A). This significantly slows down the efficient heat bath algorithm. Some suggestions for avoiding this slowdown has been made [6].

A modified version of the algorithm provides a way out. We argue that the accept–reject procedure can be replaced by a Metropolis-like algorithm. The idea is to use a mixed heat bath Metropolis method [7]. The general setting is the following: write the action as

$$A(U) = A_0(U) + A_1(U)$$

where U is a configuration (or a point in phase space). Let the old configuration be U_0. To generate a new configuration U_1 according to the Boltzmann distribution $\exp(-A(U_1))$ using the mixed method, one first generates a trial U_1 according to $\exp(-A_0(U_1))$ using the heat bath method, and then accepts or rejects this trial U_1 according to $\exp(-A_1(U_1))$ using the Metropolis method; i.e., if $A_1(U_1) - A_1(U_0)$ is less than or equal to 0, accept this U_1 as the new configuration; if $A_1(U_1) - A_1(U_0)$ is greater than 0, accept the U_1 according to the probability $\exp(-(A_1(U_1) - A_1(U_0)))$. This process can be repeated a *fixed* number of times to reduce the correlation between sweeps. The corresponding transition probability $W(U_0 \to U_1)$ can be written as [8]

$$W(U_0 \to U_1) = Z^{\ 1} e^{\ A_0(U)} \cdot \min\{1, e^{-[A_1(U_1) - A_1(U_0)]}\}$$

where $Z = \int e^{-A(U)}\, dU$. It is easy to see that this $W(U_0 \to U_1)$ satisfies the detailed balance equation

$$\frac{W(U_0 \to U_1)}{W(U_1 \to U_0)} = \frac{e^{\ A(U_1)}}{e^{-A(U_0)}}$$

and the normalization condition

$$\sum_{U_1\neq U_0} W(U_0\to U_1)+W(U_0\to U_0)=1.$$

Next let us consider the application of this new method to the case of $SU(2)$ lattice gauge theory. If we treat U as a single $SU(2)$ link variable, $e^{-A(U_1)}\,dU_1$ in Eq. (1) can be written as

$$e^{-A(U_1)}\,dU_1=\sqrt{1-a_0^2}\,e^{\beta\bar{k}a_0}e^{\beta(k-\bar{k})a_0}\cdot da_0\,d\Omega \tag{2}$$

where $U_1=ar^{\dagger}$; $a=a_0\mathbb{1}+i\mathbf{a}\cdot\boldsymbol{\sigma}$ is a new $SU(2)$ matrix to be determined according to the probability of Eq. (2); $d\Omega$ is the differential solid angle of $\mathbf{a}$ with $a_0^2+\mathbf{a}^2=1$; r is a $SU(2)$ matrix obtained from the environment sum $R=kr$; and $\bar{k}$ is a new free parameter. We treat $\sqrt{1-a_0^2}\cdot e^{\beta\bar{k}a_0}$ as $e^{-A_0(U)}$ and $e^{\beta(k-\bar{k})a_0}$ as $e^{-A_1(U)}$. Note that the update is indirect: the old a is not U_0; it is $U_0\gamma$ instead. The new algorithm proceeds as follows: generate a uniform random x_1 between 0 and 1. Next, invert

$$x_1=Z^{-1}\int_{-1}^{a_0}\sqrt{1-t^2}\cdot e^{\beta\bar{k}t}\,dt \tag{3}$$

to obtain a trial a_0. This a_0 has the distribution (2) without the last factor. To include the last factor we do a Metropolis accept–reject: accept this $a_0(\text{new})$ according to the probability P:

$$P=\begin{cases}1, & \text{if}\quad (k-\bar{k})\cdot(a_0(\text{new})-a_0(\text{old}))>0,\\ e^{\beta(k-\bar{k})(a_0(\text{new})-a_0(\text{old}))}, & \text{otherwise.}\end{cases} \tag{4}$$

If $a_0(\text{new})$ is not accepted, we keep the old link. Note that in the case where $a_0(\text{new})$ is rejected, since the probability in Eq. (2) does not depend on the direction of $\mathbf{a}$ in the 2-sphere, we may still update this link by keeping $a_0(\text{old})$ ad randomly changing $\mathbf{a}(\text{old})$ to $\mathbf{a}(\text{new})$ with $a_0(\text{old})^2+a_0(\text{new})^2=1$. Since the process of generating $(a_0, \mathbf{a})$ is repeated a fixed number of times, there is no slowdown due to processor waiting or a large number of rejections. Choosing the free parameter $\bar{k}$ such that $e^{\beta(k-\bar{k})a_0}$ is close to unity and making a lookup table for the factor $\sqrt{1-a_0^2}\,e^{\beta\bar{k}a_0}$ (because k depends on the environment and varies from one link to the other, we cannot make a lookup table for the probability in Eq. (1)), this particular breakup in Eq. (2) can be implemented very efficiently.

Our real interest is in the $SU(3)$ lattice gauge theory. For this gauge group, the update is a little bit more complicated.[1] We need to know what $a(\text{old})$ is. Let U_0 be the old link to be updated, and R the sum of six environment products. From U_0R, take a 2×2 block matrix b from either the upper left corner, or lower right corner, or four corner elements. Write the unitary part of b as kr, where r is a $SU(2)$ matrix, and k is the determinant of the unitary part. Let U_1 be the new link. The Cabibbo–Marinari method [4] is to let

$$U_1=ar^{\dagger}U_0 \tag{5}$$

where a is a $SU(2)$ matrix, and $a u^{\dagger}$ has been converted to $SU(3)$ matrix by reversing the construction of ℓ and adding a diagonal 1 in the missing row and column. a is determined by the transition probability

$$W(U_0 \to U_1) = \sqrt{1 - a_0^2} \cdot e^{2\beta k a_0/3} \tag{6}$$

with a uniformly distributed in the 2-sphere. Since $U_0 = u u^{\dagger} U_0$, the detailed balance Eq. (2) now reads

$$\frac{W(U_0 \to U_1)}{W(U_1 \to U_0)} = \frac{e^{-A(a u^{\dagger} U_0)}}{e^{\;A(u^{\dagger} U_0)}}. \tag{7}$$

Thus the old a in the Metropolis piece is u.[1] The rest of the new algorithm for $SU(3)$ is similar to that for $SU(2)$. Again, we write (6) as

$$W(U_0 \to U_1) = \sqrt{1 - a_0^2} \cdot e^{2\beta \bar{k} a_0/3} \cdot e^{2\beta(k - \bar{k}) a_0/3} \tag{8}$$

and do the heat bath as in (3) and the Metropolis as in (4).

The generation of a_0 according to $\sqrt{1 - a_0^2} \cdot e^{2\beta \bar{k} a_0/3}$ can be done by making a lookup table and interpolation, since $\bar{k}$ is a constant. The optimal choice of $\bar{k}$ is obtained by minimizing the variance of

$$\tfrac{2}{3}\beta(k - \bar{k})(a_0(\text{new}) - a_0(\text{old})) \tag{9}$$

so as to achieve a high acceptance in the Metropolis accept reject test. A simple choice is to set $\bar{k}$ equal to the average of k.

We have done an $SU(3)$ simulation on a 4^4 lattice to check the new algorithm. We chose $\beta = 6.0$ and 5.5. For each β, we first ran the C–M heat bath a few hundred sweeps to estimate the average k, and then made the lookup tables. We use three $SU(2)$ subgroups to update a link and do a single Metropolis hit for the last factor in (8) since the acceptance of a_0 is already high enough (larger than 92%).

At $\beta = 6.0$, the average k is about 3.95; we made one table with $\bar{k} = 4.0$ and 8192 entries, and used a first-order linear interpolation. Both the C–M heat bath method and the new method ran a total of 1400 sweeps. For comparison, we also ran the standard Metropolis method for the same number of sweeps. All three runs start with a cold start; i.e., initial links were all set to be unit matrices. Discarding the first 400 sweeps for each run, the data are shown in Table I. The acceptance of the Metropolis hit in the new method is about 94%, in comparison to an acceptance of 31% for the C–M heat bath. This comes from the fact that the average value of the expression in (9) is about -0.12 when it is negative. The new algorithm runs faster,

[1] See Appendix B for more details.

TABLE I

$\beta = 6.0$

Method	Action	Specific heat	Acceptance
C–M heat bath	0.4034 ± 0.0004	3.85 ± 0.18	0.31
New heat bath	0.4038 ± 0.0004	3.53 ± 0.16	0.94
Metropolis	0.4041 ± 0.0007	3.36 ± 0.14	0.61

because there are fewer arithmetic operations and only one trial. The sweep-to-sweep correlation function, defined as

$$C(\ell) = \sum_n (A(\ell + n) - \bar{A}) \cdot (A(n) - \bar{A}) \Big/ \sum_n (A(n) - \bar{A})^2,$$

is shown in Fig. 1. It is clear that the new method has correlations as small as those of the C–M heat bath method and considerably smaller than those of the standard Metropolis method.

At $\beta = 5.5$, with $\ell = 3.4$, we made two tables of the same size, one with x_1 equally divided in the range [1/1024, 1], the other with x_1 in the range [0, 1/1024]. Both the C–M heat bath method and the new method were run a total of 2000 sweeps, starting with the cold start. The Metropolis was also run 2000 sweeps, starting from

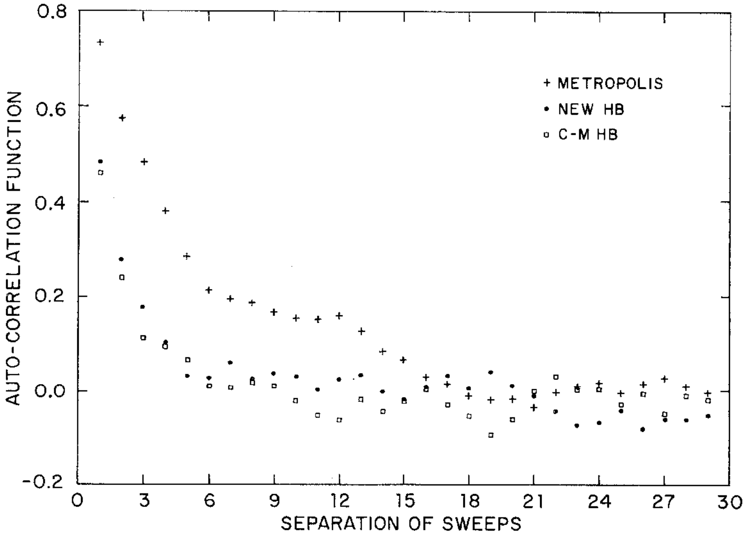

FIG. 1. Comparison of auto-correlation functions for the three methods at $\beta = 6.0$. The correlation function for the Metropolis reaches the noise level (±0.08) at about separation = 15, while that for the two heat bath methods, at about 5.

TABLE II

$\beta = 5.5$

Method	Action	Specific heat	Acceptance
C M heat bath	0.498 ± 0.002	10.56 ± 0.41	0.34
New heat bath	0.498 ± 0.002	10.22 ± 0.39	0.92
Metropolis	0.501 ± 0.003	10.88 ± 0.36	0.63

the old configuration generated by the 2000 sweeps of the C–M heat bath. Discarding the first 600 sweeps for the two heat bath runs and the first 100 sweeps for the Metropolis run, the data are shown in Table II and Fig. 2.

From these data, we can see that (i) the new method is in good agreement with the C M heat bath method and the standard Metropolis method; (ii) these data are in agreement with those of Christ and Terrano [9] and Cabibbo and Marinari [4]; (iii) at both β values, the auto-correlations reach the noise level (about 0.1 or less), about three times faster in the two heat bath methods than in the Metropolis method.

Based on these test runs, it seems that the new algorithm provides a good solution to the asynchronous heat bath accept–reject problem. It runs about four times (taking into account the program speed and the correlation between sweeps) faster than the Metropolis algorithm. In high speed computers or parallel array

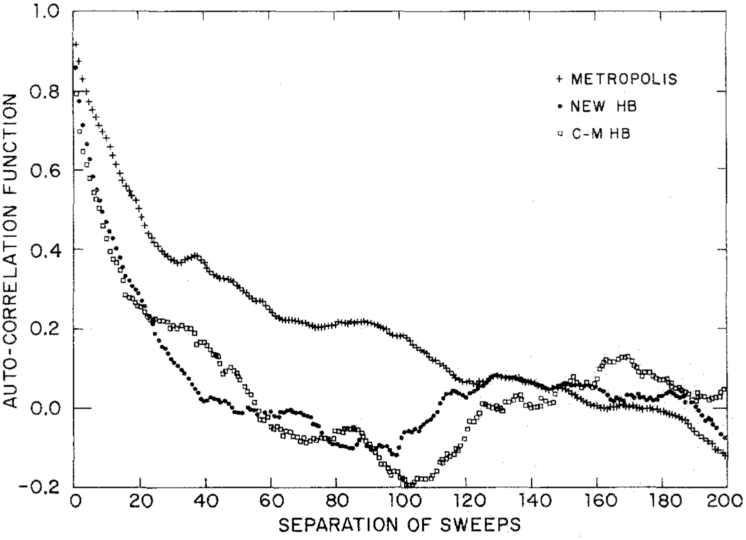

FIG. 2. Comparison of auto-correlation functions for the three methods at $\beta = 5.5$. The correlation function for the Metropolis reaches the noise level (± 0.1) at about separation = 110, while that for the two heat bath methods, at about 40.

processors, where the environment sums of matrix product are very efficiently calculated, the extra speedup, due to the lookup table and fewer repeated trials, may become significant.

The new algorithm was developed mainly to implement the heat bath method in the parallel processor (Columbia group [9]); the "odd" table sizes, $1024 = 2^{10}$ and $8192 = 2^{13}$, came from the consideration of using these tables in the processor.

Appendix A

Let M processors update links synchronously and assume that the acceptance of the accept–reject procedure of the C–M heat bath follows a simple binomial distribution with acceptance probability $\mathscr{p}$. Let $\mathscr{q} = 1 - \mathscr{p}$ be the probability of rejection for one processor in a trial. Then the probability $P(n)$, that only after n trials all the processors accept the $\mathscr{a}_0$'s, is

$$\begin{aligned} &P(1) = r^M, \\ &P(n) = (1-\mathscr{q}^n)^M - (1-\mathscr{q}^{n-1})^M, \qquad n > 1. \end{aligned} \tag{10}$$

Since the probability that the one processor rejects n times in the first n trials is $\mathscr{q}^n$, the probability that one processor accepts at least once in the n trials is $1 - \mathscr{q}^n$. The probability that each of the M processors accepts at least once in n trials is $(1-\mathscr{q}^n)^M$. Therefore, the probability that at least one processor accepts for the first time at the nth trial is $(1-\mathscr{q}^n)^M - (1-\mathscr{q}^{n-1})^M$. This is (10). One can easily see that

$$\sum_{n=1}^{\infty} P(n) = \operatorname*{Lim}_{n\to\infty} (1-\mathscr{q}^n)^M = 1,$$

since $\mathscr{q} < 1$. For $M = 16$ and $\mathscr{p} = 0.31$, the average trial number is $\bar{n} = \sum_{n=1}^{\infty} nP(n) = 9.61$.

Appendix B

In the C–M method, the new link depends on the old link through Eq. (5), and the $\mathscr{k}$ in the transition probability in Eq. (6) depends on both the environment R and the old link U_0. Thus, the *backward* transition in Eq. (7) with $U_0 = \mathscr{a}' r'^{\dagger} U_1$ should be

$$W(U_1 \to U_0) = W(U_1 \to \mathscr{a}' r'^{\dagger} U_1) = \sqrt{1 - \mathscr{a}_0'^2} \cdot e^{2\beta \mathscr{k}' \mathscr{a}_0'/3} \tag{11}$$

where r' and $\mathscr{k}'$ are determined from $U_1 R$, not $U_0 R$. Both $\mathscr{k}'$ and $\mathscr{a}'$ are needed to do the Metropolis accept–reject test. We now show that

$$\mathscr{k}' = \mathscr{k} \qquad \text{and} \qquad r' = \mathscr{a}, \tag{12}$$

so that $r'^{\dagger}U_1 = a^{+}(ar^{+}U_0) = r^{\dagger}U_0$. Hence, from the requirement $U_0 = a'r'^{+}U_1 = a'r^{\dagger}U_0$, we have

$$a(\text{old}) \equiv a' = r$$

and

$$e^{-[A_1(U_1) - A_1(U_0)]} = e^{2\beta(k - \bar{k})(a_0(\text{new}) - a_0(\text{old}))/3}. \tag{13}$$

First, let us look at the *forward* transition $U_0 \rightarrow U_1$, $U_1 = ar^{+}U_0$. For the $SU(3)$ lattice gauge, the action in Wilson form is

$$A(U, R) = \frac{\beta}{3} \operatorname{Re} \operatorname{Tr} UR$$

where U is the link to be updated. R is the environment sum as in the text. For the new link U_1, we have

$$\begin{aligned} A(U_1, R) &= \frac{\beta}{3} \operatorname{Re} \operatorname{Tr}(U_1 R) = \frac{\beta}{3} \operatorname{Re} \operatorname{Tr}(ar^{\dagger}U_0 R) \\ &= \frac{\beta}{3} \operatorname{Re} \operatorname{Tr}(ar^{\dagger}\ell) + \text{terms independent of } a, \end{aligned} \tag{14}$$

where ℓ is defined in the text and the trace in the second line is taken over the 2×2 matrix. Being a general 2×2 complex matrix, ℓ can always be written as

$$\ell = \ell_0 + i\boldsymbol{\ell} \cdot \boldsymbol{\sigma} \tag{15}$$

where ℓ_0 and $\boldsymbol{\ell} = (\ell_1, \ell_2, \ell_3)$ are in general complex numbers. Separate real and imaginary parts of ℓ_0 and $\boldsymbol{\ell}$, extract the normalization factor, and we have

$$\begin{aligned} \ell &= \operatorname{Re} \ell_0 + i(\operatorname{Re} \boldsymbol{\ell}) \cdot \boldsymbol{\sigma} + i[\operatorname{Im} \ell_0 + i(\operatorname{Im} \boldsymbol{\ell}) \cdot \boldsymbol{\sigma}] \\ &= k\gamma + il\delta, \\ k &= \sqrt{(\operatorname{Re} \ell_0)^2 + (\operatorname{Re} \boldsymbol{\ell})^2}, \qquad l = \sqrt{(\operatorname{Im} \ell_0)^2 + (\operatorname{Im} \boldsymbol{\ell})^2} \end{aligned} \tag{16}$$

where γ and δ are $SU(2)$ matrices, e.g.,

$$r = r_0 + i\mathbf{r} \cdot \boldsymbol{\sigma}, \qquad r_0^2 + \mathbf{r}^2 = 1, \qquad r_0 \text{ and } \mathbf{r} \text{ are real.}$$

Note that this r is the same r as in Eq. (5). Now, from Eq. (14), dropping out the terms independent of a, we have

$$\begin{aligned} A(U_1, R) &= \frac{\beta}{3} \operatorname{Re} \operatorname{Tr}[ar^{+}(kr + il\delta)] \\ &= \frac{\beta}{3} \operatorname{Re} \operatorname{Tr} ka = \frac{2}{3} \beta k a_0 \end{aligned} \tag{17}$$

since $\mathrm{Re}[\mathscr{i}\,\mathrm{Tr}(\mathscr{a}r^{\dagger}\mathscr{s})]=0$. Including the factor $\sqrt{1-\mathscr{a}_0^2}$ from the Haar measure of the $SU(2)$ group

$$\frac{1}{2\pi^2}\,d\mathscr{a}_0\,d^3\mathscr{a}\;\delta(\mathscr{a}_0^2+\mathscr{a}^2-1)=\frac{1}{4\pi^2}\sqrt{1-\mathscr{a}_0^2}\,d\mathscr{a}_0\,d\Omega$$

we arrive at the transition probability in Eq. (6) (up to a constant factor, which we did not write out explicitly, as in Eqs. (1) and (2)).

Now for the backward transition, $U_1\to\mathscr{a}'r'^{\dagger}U_1$, repeat a similar procedure, using the fact that $U_1=\mathscr{a}r^{\dagger}U_0$ and $\mathscr{b}'=U_1R=\mathscr{a}r^{\dagger}U_0R=\mathscr{a}r^{\dagger}\cdot(\mathscr{k}r+\mathscr{i}\mathscr{l}\mathscr{s})=\mathscr{k}\mathscr{a}+\mathscr{i}\mathscr{l}\mathscr{a}r^{\dagger}\mathscr{s}$, we have $A(\mathscr{a}'r'^{\dagger}\cdot U_1, R)=(\beta/3)\,\mathrm{Re}\,\mathrm{Tr}(\mathscr{a}'r'^{\dagger}\mathscr{k}\mathscr{a})$. By definition, $\mathscr{k}'\cdot r'$ = the unitary part of $U_1R=\mathscr{k}\mathscr{a}$, so that

$$\mathscr{k}'=\mathscr{k},\qquad r'=\mathscr{a}.$$

With $A(\mathscr{a}'r'^{\dagger}U_1, R)=\frac{2}{3}\beta\mathscr{k}'\mathscr{a}_0'$, we have arrived at Eq. (11) with the relation Eq. (12).

Acknowledgments

I thank Professor Norman Christ for suggesting this problem to me and for many clarifying and fruitful discussions. I thank Olivier Martin for many helpful discussions. I thank Professor Anthony Terrano for programming assistance.

References

1. D. Barkai, K. J. M. Moriarty, and C. Rebbi, *Phys. Rev. D* **29** (1984), 1207; N. Christ and A. Terrano, *Phys. Rev. Lett.*, in press; A. D. Kennedy, J. Kuti, S. Meyer, and B. J. Pendleton, *Phys. Rev. Lett.* **54** (1985), 87; O. Martin and S. Samuel, Columbia University Preprint CU-TP-300; S. Otto and J. Stack, *Phys. Rev. Lett.* **52** (1984), 2328.
2. J. Beetem, M. Denneau, and D. Weingarten, *The GF11 Supercomputer*, IBM Yorktown Report, 1985; N. Cabibbo *et al.*, Universita di Roma "La Sapienza" Report IFUP-TH84/40, 1984. Italy.
3. N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, *J. Chem. Phys.* **21** (1953), 1007.
4. N. Cabibbo and E. Marinari, *Phys. Lett. B* **119** (1982), 387.
5. M. Creutz, *Phys. Rev. D* **21** (1980), 2308.
6. A. D. Kennedy, J. Kuti, S. Meyer, and B. J. Pendleton, Santa Barbara Preprint NSF-ITP-84-62.
7. B. Freedman, P. Smolensky, and D. Weingarten, *Phys. Lett. B* **113** (1982), 481; J. Sexton, Ph.D. thesis, Columbia University, 1985.
8. K. Binder, *Monte Carlo Methods in Statistical Physics*, edited by K. Binder (Springer-Verlag, Berlin, 1979), p. 1.
9. N. Christ and A. Terrano, *Nucl. Instrum. Methods* **222** (1984), 534; *IEEE Trans. Comput. C* **33** (1984), 344; Columbia University Preprint CU-TP-298.
10. K. Bowler and B. Pendleton, *Nucl. Phys. B* **230** (1984), 109.