Mem. Fac. Sci. Shimane Univ.,
24, pp. 139–148 Dec. 24, 1990

A Parallel Algorithm to Calculate Second Derivatives of Conformational Energy in Proteins with Any Branched Sidechains

Heihachiro HARA, Takashi KAYANO, Kazuhiko KANEHIRO*
and Nobuhiro Go**
Department of Information Science, Shimane University, Matsue 690, Japan
(Received September 5, 1990)

A protein molecule can be regarded as a set of rigid units and rotatable bonds. We present here a parallel algorithm of calculations for second derivative matrix with respect to dihedral angles around bonds for proteins with any branched side chains. The basic idea of the algorithm for processor array type computers is the introduction of the nearest outside units and related quantities. A method of initial setting and transfer of bond data is also presented in simple mathematical expressions which are easily coded in the computer program.

§1. Introduction

We assume here the bond lengths and the bond angles in the protein molecule as fixed and treat only the dihedral angles as independent variables. The second derivative matrix of conformational energy with respect to dihedral angles is an important physical quantity in the stage of minimization of energy and also in the stage of conformational fluctuation computations.

The calculation of the second derivative matrix is, however, a time consuming step. A straightforward approach needs n^4 operations for the second derivative matrix of conformational energy where n is the number of dihedral angles. This rapid increase of the number of operations makes this approach prohibitive for large molecules. By using a conventional (sequantial processing) computer, fast algorithms have been developed to calculate the second derivatives (cf. [1], [8]). These algorithms require a number of operations proportional to n^2. A recent increase in speed has been also achieved on the pipe-line type supercomputer by modifying the algorithms to be well vectorized (cf. [9]). The pipe-line type supercomputer is an SIMD (single instruction stream / multiple data stream) computer.

Another option is the use of processor array type (or parallel) computers which are generally superior in high performance compared with cost. ILLIAC-IV (Univ. of Illinois) is the most famous prototype of the processor array type computer, but it

* Tamano Laboratory, Mitsui Engineering & Shipbuilding Co. Ltd., Tamano, Okayama 706, Japan
** Department of Chemistry, Kyoto University, Kyoto 606, Japan

is specified as SIMD. After the pioneering project of ILLIAC-IV, various type processor array computers have been developed for scientific applications: BSP (Burroughs), DAP (ICL), MPP (NASA), HEP (Denelcor) and MiPAX. MiPAX was originally developed in Univ. of Tsukuba [6], and is now being improved for commercial use by Mitsui E&S [4], [5], [7]. MiPAX has a nearest-neighbor-connected array and is specified as MIMD (multiple instruction stream / multiple data stream).

In our previous papers [2], [3], a new parallel algorithm was presented to calculate the second derivatives by using the processor array type computer which has the nearest-neighbor-connected array. The basic idea of the algorithm is the introduction of the concepts of the nearest outside units and related quantities. In the previous papers, however, the mathematical expressions were restricted to proteins without side chains for easy understanding. The purpose of this paper is to present the general expressions for the proteins with any branched side chains.

§2. Second derivative matrix

We treat here only dihedral angles as independent variables. As a result of this assumption a protein molecule can be regarded as a set of rigid units and rotatable bonds. The units are rigid in the sense that there are no rotatable bonds within them. Each unit consists of one or more atoms.

The second derivative F_{ij} of the conformational energy E with respect to the dihedral angle θ_i around bond i and the dihedral angle θ_j around bond j is specified as follows (cf. [1]):

$$F_{ij} \equiv \frac{\partial^2 E}{\partial\theta_i \partial\theta_j}$$

$$= -s_{ij}\cdot(e_i,\ e_i \times r_{\varepsilon(i)})\cdot \sum_{\beta\in M_j}\sum_{\alpha\in \bar{M}_{i(j)}} (c_{\alpha\beta}C_{\alpha\beta} + d_{\alpha\beta}D_{\alpha\beta})\cdot \begin{pmatrix} e_j \\ e_j \times r_{\varepsilon(j)} \end{pmatrix}$$

$$(1 \leqq i \leqq j \leqq n), \tag{1}$$

where e_i is a unit vector along the bond i pointing to the positive direction. The positive direction of bond is defined as the further direction from the amino terminus of the protein molecule. $r_{\varepsilon(i)}$ is a position vector of a certain point properly chosen on bond i. M_i is a set of units on the outside of bond i. The outside of bond i is defined as the side of positive direction of bond i. $\bar{M}_{i(j)}$ is a set of units on the "j-outside" of bond i which is depending on the mutual relationship of set M_i and M_j (Fig. 1):

$$\bar{M}_{i(j)} = \begin{cases} M_i^c & (M_i \supset M_j), \\ M_i & (M_i \cap M_j = \emptyset), \end{cases} \tag{2}$$

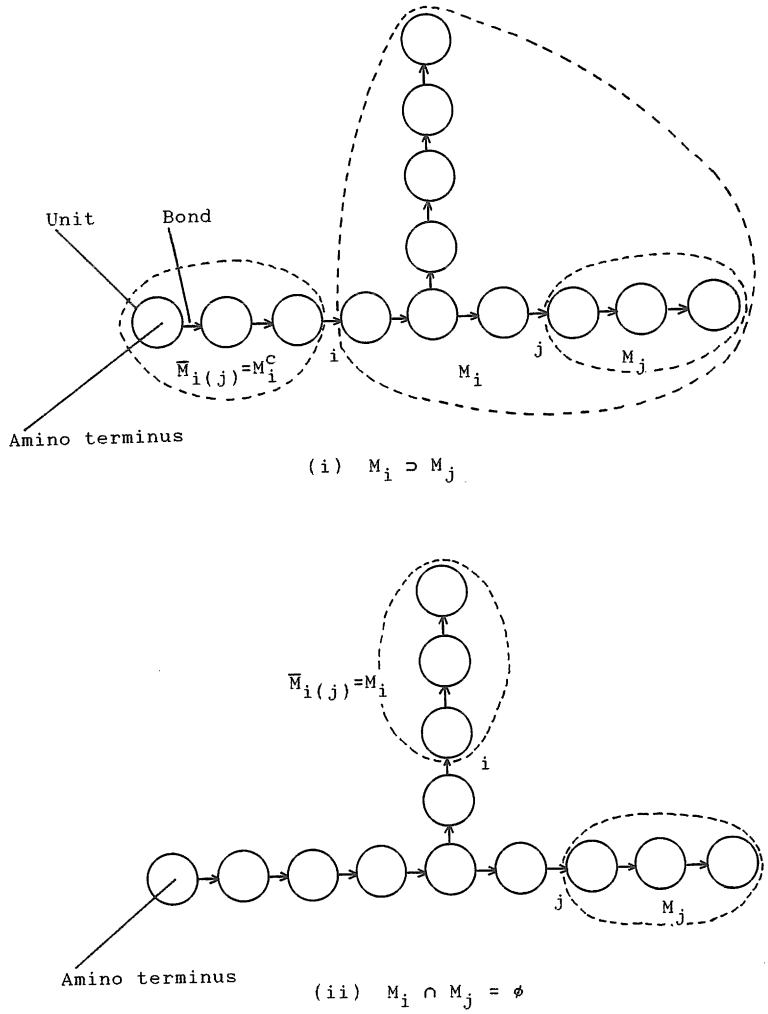

Fig. 1. Mutual relationship of set M_i and M_j

where M_i^c is the complement of set M_i and $\emptyset$ is the empty set. s_{ij} is a sign defined as:

$$s_{ij} = \begin{cases} -1 & (M_i \supset M_j), \\ +1 & (M_i \cap M_j = \emptyset). \end{cases} \tag{3}$$

$c_{\alpha\beta}$ and $d_{\alpha\beta}$ in Eq. (1) are scalars, $C_{\alpha\beta}$ and $D_{\alpha\beta}$ are 6×6 matrices with respect to atom pair (α, β). n is the number of bonds.

F_{ij} in Eq. (1) can be simply written by:

$$F_{ij} = -s_{ij} b_i^t \cdot \sum_{\beta \in M_j} \sum_{\alpha \in \bar{M}_{i(j)}} L_{\alpha\beta} \cdot b_j \qquad (1 \leqq i \leqq j \leqq n), \tag{4}$$

where b_i and b_j are 6-dimensional column vectors, $L_{\alpha\beta}$ is a 6×6 matrix.

§3. Parallel algorithm for second derivative matrix

We introduce here a set of new variables concerned with the nearest outside units:

$$T^{kl} = \sum_{\beta \in V_l} \sum_{\alpha \in \bar{V}_{k(l)}} L_{\alpha\beta}, \tag{5}$$

$$G_{ij}^{kl} = -s_{ij} b_i^t T^{kl} b_j. \tag{6}$$

In Eq. (5) V_l is the nearest outside unit of bond l, and $\bar{V}_{k(l)}$ is the nearest "l-outside" unit of bond k (Fig. 2):

$$\bar{V}_{k(l)} \subseteqq \begin{cases} M_k^c & (M_k \supset M_l), \\ M_k & (M_k \cap M_l = \emptyset), \end{cases} \tag{7}$$

$$V_l \subseteqq M_l. \tag{8}$$

Bond i and bond j in Eq. (6) should satisfy the following relationship with bond k and bond l:

$$\bar{M}_{i(j)} \supseteqq \begin{cases} M_k^c & (M_k \supset M_l), \\ M_k & (M_k \cap M_l = \emptyset), \end{cases} \tag{9}$$

$$M_j \supseteqq M_l. \tag{10}$$

Note that T^{kl} in Eq. (5) is a 6×6 matrix and G_{ij}^{kl} in Eq. (6) is a scalar.

Now F_{ij} in Eq. (4) can be rewritten as:

$$F_{ij} = \sum_{l \in M_j} \sum_{k \in \bar{M}_{i(j)}} G_{ij}^{kl} \tag{11}$$

$$= \sum_{l=j}^{n_j} \sum_{m=1}^{l} G_{ij}^{k(l,m)l} \qquad (1 \leqq i \leqq j \leqq n). \tag{12}$$

The expression of Eq. (12) is obtained by introducing an index m and a function k depending on l and m:

$$k = k(l, m), \tag{13}$$

which is given explicitly in §5, and n_j is defined by

$$n_j = \max_{l \in M_j} l. \tag{14}$$

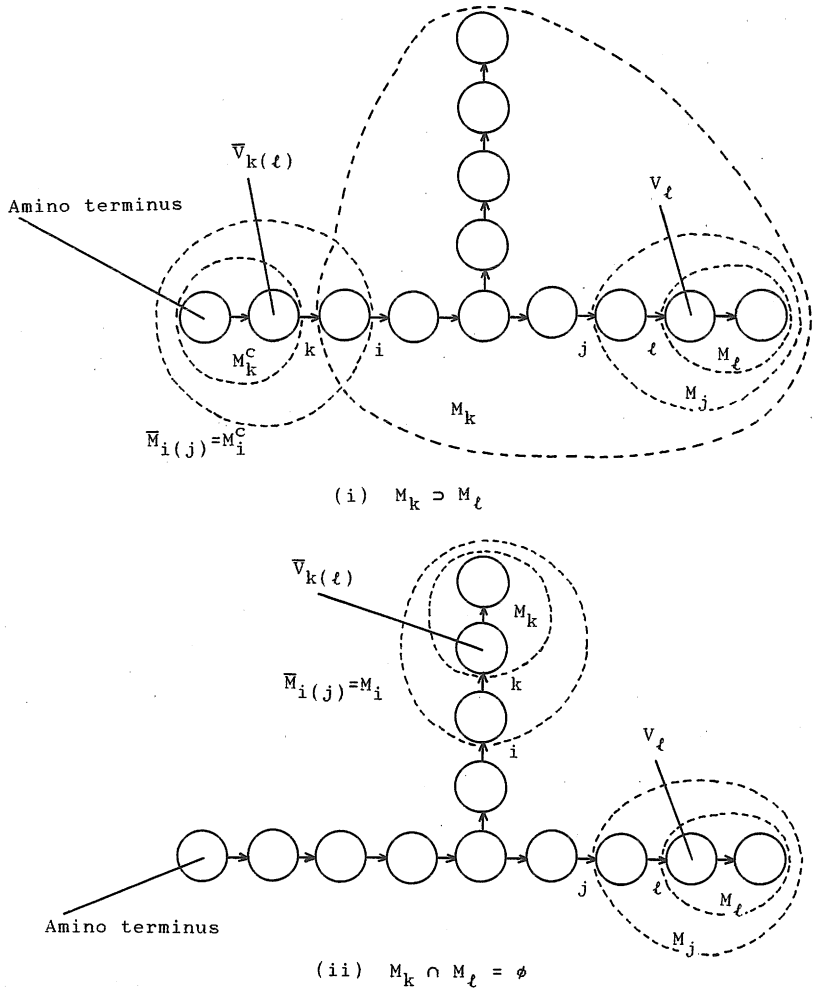

Fig. 2. Nearest outside units $\overline{V}_{k(l)}$ and V_l

Now Eq. (12) can be also expressed by using the recurrent equations:

for inner loop, $F_{ij}^l := F_{ij}^l + G_{ij}^{k(l,m)l}$

$$(F_{ij}^l = 0 \text{ if } m = 1;\ m = 1,\ldots,l), \tag{15}$$

for outer loop, $F_{ij}^l := F_{ij}^l + F_{ij}^{l-1}$

$$(F_{ij}^{j-1} = 0;\ l = j,\ldots,n_j), \tag{16}$$

where := means the substitution of data from right hand side to left hand side. In Eq. (16), $F_{ij}^{n_j}$ is equal to the resulting sum, F_{ij}, in Eq. (12).

If the indices l and m are referred to the processing unit number (PU No.) and step No. respectively, the following parallel algorithm can be introduced:

(i) Processing units are linked as 1-dimensional chains: PU_l $(l = 1,\ldots,n)$. Data of bonds and units for the calculation of $T^{ll}(l = 1,\ldots,n)$ are initially loaded in each PU_l.

(ii) Step m: Calculate $T^{k(l,m)l}$ and $G_{ij}^{k(l,m)l}(1 \leqq i \leqq j \leqq l)$ at each PU_l $(l = m, m+1,\ldots,n)$.

(iii) Transfer the data of bond k and unit $\bar{V}_{k(l)}$ in PU_l into $PU_{l+1}(l = 1,\ldots,n-1)$.

(iv) Repeat (ii) and (iii) for $m = 1,\ldots,l$. When step No. m reaches to l at PU_l $(l = j,\ldots,n_j)$, F_{ij}^l in each $PU_l(l = j,\ldots,n_j)$ should be summed recurrently by Eq. (16). At the step $m = n_j$, $F_{ij}^{n_j}$ in PU_{n_j} is the resulting sum F_{ij}.

The parallel algorithm proposed above calculates all elements of the second derivative matrix, F_{ij} $(1 \leqq i \leqq j \leqq n)$, in n steps of data transfer between neighboring processing units.

§4. Introduction of order of branch and related sets

To emphasize the merit of our parallel algorithm which can be applied to polymer chains with side chains of any complexity, we assume that polypeptide has a series of branches which are discriminated as "order of branches":

$\lambda = 0$: 0-th order branch (back bone),

$\lambda = 1$: 1-st order branch (side chain),

...

$\lambda = v$: v-th order branch (highest order branch).

Let S^λ $(\lambda = 0,\ldots,v)$ be the set of bonds on the λ-th order branch. λ-th order branch $(\lambda = 0,\ldots,v-1)$ is divided into two parts by the $(\lambda+1)$-th order branch. The sub-branch which is nearer to the amino terminus is distinguished by subscript A, and another sub-branch is distinguished by subscript B. Since the v-th order branch is not divided, the subscripts A and B of v-th order branch are regarded as equivalence. Therefore, the set of bonds on the λ-th order branch can be expressed as:

$$S^\lambda = \begin{cases} \{S_A^\lambda, S_B^\lambda\} & (0 \leqq \lambda \leqq v-1), \\ \{S_A^\lambda\} = \{S_B^\lambda\} & (\lambda = v). \end{cases} \tag{17}$$

Now we introduce the numbering rule of bonds. Bond 1 should be the bond attached to the amino terminus. Bonds $2, 3,\ldots,n$ are sequentially given for the bonds which are listed in the following set of bond sets (Fig. 3):

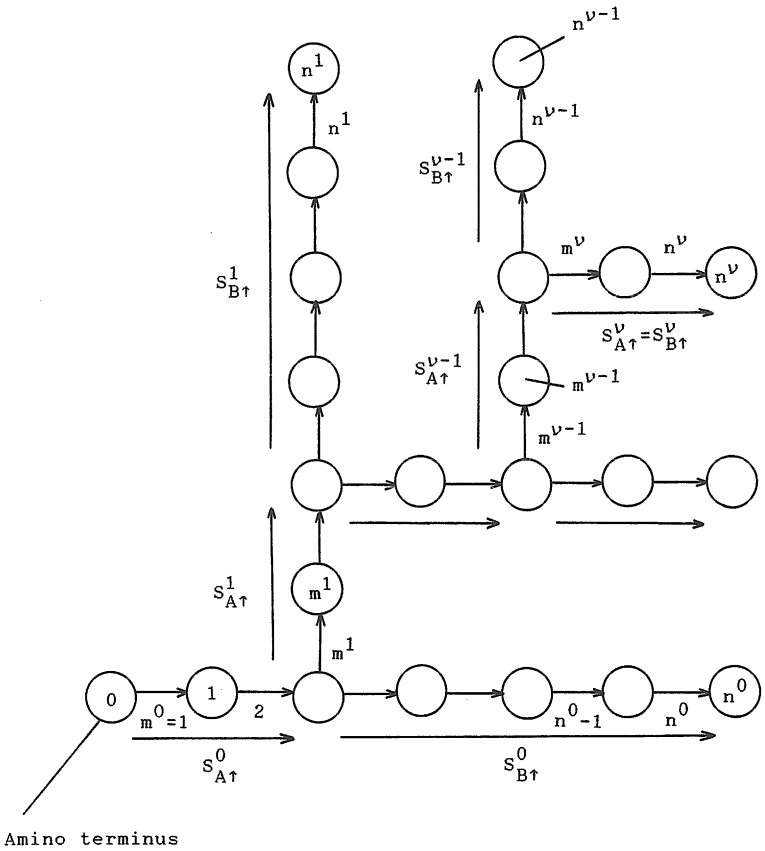

Fig. 3. Numbering rule of bonds

$$\{S^0_{A\uparrow}, S^1_{A\uparrow},\ldots,S^{\nu-1}_{A\uparrow}, S^{\nu}_{A\uparrow}, S^{\nu-1}_{B\uparrow},\ldots,S^1_{B\uparrow}, S^0_{B\uparrow}\}, \tag{18}$$

where $S^{\lambda}_{A\uparrow}(\lambda = 0,\ldots,\nu)$ and $S^{\lambda}_{B\uparrow}(\lambda = 0,\ldots,\nu-1)$ are the ordered sets of bond sets S^{λ}_{A}, S^{λ}_{B} respectively in the sequence of nearness to the amino terminus.

We also define the minimum bond No. and maximum bond No. on the λ-th order branch:

$$m^{\lambda} = \min_{l \in S^{\lambda}} l, \qquad n^{\lambda} = \max_{l \in S^{\lambda}} l. \tag{19}$$

Unit numbers are defined such that unit 0 is the amino terminus, and unit l ($l = 1,\ldots,n$) is the further unit, from the amino terminus, which is attached to bond l.

§5. Initial setting and transfer of data

We present here the theory of initial setting and transfer of data on processing units for our new parallel algorithm.

Let $\bar{S}^{\lambda}$ be the set of ordered bond sets (Fig. 4):

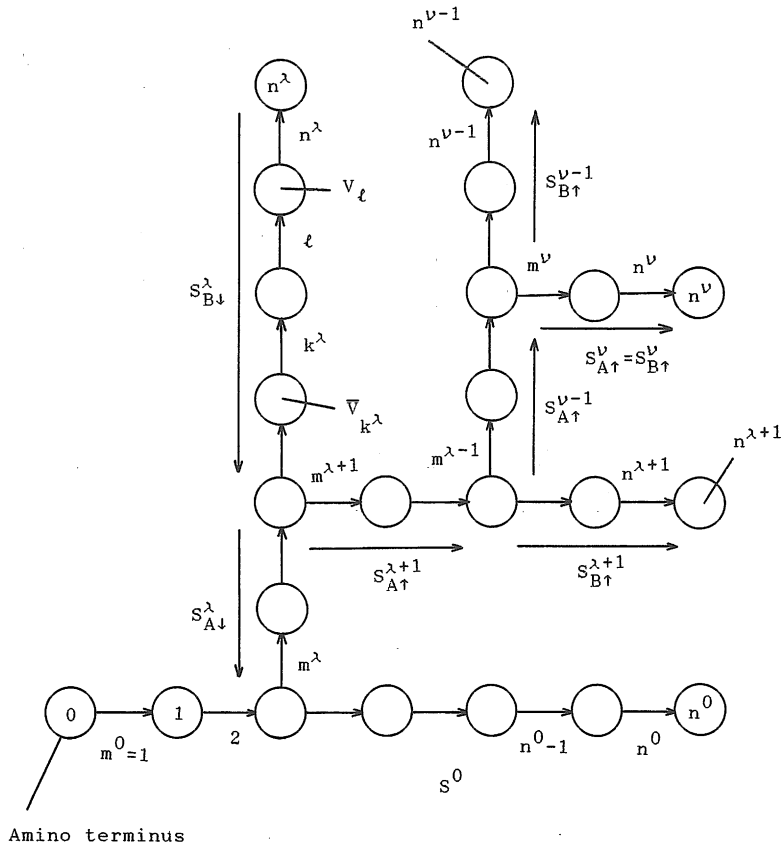

Fig. 4. Set of ordered bond sets $\bar{S}^{\lambda}$

$$\bar{S}^{\lambda}=\begin{cases}\{S^{\lambda}_{B\downarrow}, S^{\lambda+1}_{A\uparrow},\ldots,S^{\nu-1}_{A\uparrow}, S^{\nu}_{A\uparrow}, S^{\nu-1}_{B\uparrow},\ldots,S^{\lambda+1}_{B\uparrow}, S^{\lambda}_{A\downarrow}\} & (0\leqq\lambda\leqq\nu-1),\\ \{S^{\lambda}_{B\downarrow}\}=\{S^{\lambda}_{A\downarrow}\} & (\lambda=\nu).\end{cases} \tag{20}$$

In this equation, the subscript $\uparrow$ means the increasing order in bond number sequence, and the subscript $\downarrow$ means the decreasing order.

Let $k^{\lambda}(l, m)$ be the bond of λ-th order branch stored in the processing unit l at step m. $k^{\lambda}(l, m)$ is defined by

$$k^{\lambda}(l, m) = \begin{cases} \bar{S}^{\lambda}_{n^{\lambda}-l+1} & (m = 1), \\ k^{\lambda}(l-1, m-1) & (2 \leqq m \leqq n); \end{cases}$$

$$(0 \leqq \lambda \leqq v,\ l \in S^{\xi}\ (\lambda \leqq \xi \leqq v)), \tag{21}$$

where $\bar{S}^{\lambda}_{n^{\lambda}-l+1}$ is the $(n^{\lambda}-l+1)$-th element of bond number set $\bar{S}^{\lambda}$, and n^{λ} is the maximum bond number on the λ-th order branch defined in Eq. (19). If $\bar{S}^{\lambda}_{n^{\lambda}-l+1} = \emptyset$, then we set that $k^{\lambda}(l, m) = 0$. From Eqs. (19), (20) and (21), the following equations can be derived:

$$k^{\lambda}(m^{\lambda}, 1) = m^{\lambda}, \qquad k^{\lambda}(n^{\lambda}, 1) = n^{\lambda}. \tag{22}$$

We also define the nearest outside unit of bond $k^{\lambda}(l, m)$ as $\bar{V}_{k^{\lambda}(l,m)}$ which is further unit from bond n^{λ}.

Finally we can obtain $k(l, m)$ in Eq. (13):

$$k(l, m) = k^{\mu}(l, m) \qquad (1 \leqq m \leqq l \leqq n), \tag{23}$$

where

$$\mu = \max_{\lambda \in K} \lambda, \tag{24}$$

$$K = \{\lambda |\ 0 \leqq \lambda \leqq v,\ k^{\lambda}(l, m) \neq 0\}. \tag{25}$$

§ 6. Conclusions

We have presented here a parallel algorithm of calculations for second derivative matrix for proteins with any branched side chains. The important problems to be solved in our approach were how to define the "nearest outside" units and related quantities, and how to store and transfer the unit and bond data on processing units connected as 1-dimensional chains whereas highly branched side chains exist in actural proteins.

In this paper, a new concept of nearest outside units $\bar{V}_{k(l)}$ and V_l is introduced. We also introduce T^{kl} and G^{kl}_{ij} which are partial contributions to the second derivative F_{ij}.

To make our mathematical expressions correspond with algorithm on parallel computer which has 1-dimensional processing unit (PU) chains, indices l and m are referred to PU No. and step No. respectively. By using new quantities and indices proposed above, our parallel algorithm can calculate all elements of the second derivative matrix, F_{ij} $(1 \leqq i \leqq j \leqq n)$, in n steps of data transfer between neighboring processing units.

To deal with highly branched side chains, an idea of "order of branch" is also introduced. A method of initial setting and transfer of bond data, $k^{\lambda}(l, m)$ for λ-th

order branch and $k(l, m)$ for actual computation, is presented in simple mathematical expressions which are easily coded in the computer program.

We would like to thank Dr. Masahiko Kishi, Mitsui E&S Co. Ltd., for his encouragement to this work.

References

[1] H. Abe, W. Braun, T. Noguti and N. Go, Rapid calculation of first and second derivatives of conformational energy with respect to dihedral angles for proteins, general recurrent equations, Computers & Chemistry, **8** (1984), 239–247.

[2] H. Hara, K. Kanehiro and N. Go, A parallel algorithm for second derivative matrix of conformational energy for proteins, Proc. of Int. Symp. on Computer Analysis for Life Science held at Hayashibara Biochemical Lab., Okayama (July 9–12, 1985).

[3] H. Hara, K. Kanehiro and N. Go, A parallel computation of second derivatives of conformational energy for polypeptide chains on the parallel computer MiPAX, Proc. of Int. Conf. on Computational Mechanics held at Tokyo (May 25–29, 1986), 79–84.

[4] H. Hara, M. Kishi, A. Maeda and Y. Kodera, Applications of parallel computer to computational mechanics, Proc. of JSST Conf. on Recent Advances in Simulation of Complex Systems held at Tokyo (July 15–17, 1986), 416–421.

[5] H. Hara, Y. Kodera and K. Kanehiro, Flow simulations by parallel computer MiPAX, Int. J. Supercomputer Applications, **2** (1988), 73–80.

[6] T. Hoshino et al., Highly parallel processor array PAX for wide scientific applications, Proc. of Int. Conf. on Parallel Processing held at Ohio State University (August 23–26, 1983), 95–105.

[7] M. Kishi, T. Kashizaki and K. Fujiwara, The MiPAX parallel computer for computational mechanics, Proc. of Int. Conf. on Computational Mechanics held at Toktyo (May 25–29, 1986), 67–72.

[8] T. Noguti and N. Go, A method of rapid calculation of a second derivative matrix of conformational energy for large molecules, J. Phys. Soc. of Japan, **52** (1983), 3685–3690.

[9] H. Wako and N. Go, Algorithm for rapid calculation of Hessian of conformational energy function of proteins by supercomputer, J. Computational Chemistry, **8** (1987), 625–635.