Parallel Processing of Geological Surface Estimation using Scattered Field Data

Susumu NONOGAKI [1], Tatsuya NEMOTO[2] and Shinji MASUMOTO[2]

[1]Geological Survey of Japan, National Institute of Advanced Industrial Science and Technology,
Central 7, 1-1-1 Higashi, Tsukuba, Ibaraki 3058567, JAPAN
Email: s-nonogaki@aist.go.jp
[2]Department of Geosciences, Graduate School of Science, Osaka City University,
3-3-138 Sugimoto, Sumiyoshi-ku, Osaka 5588585, JAPAN
Email: tnemoto@sci.osaka-cu.ac.jp; masumoto@sci.osaka-cu.ac.jp

ABSTRACT

Scattered data obtained from geological field survey play an important role in understanding subsurface condition. However, raw geological data cannot be used directly in practical geological analyses. A form of geological surface shall be estimated using raw geological data. Most systems for geological surface estimation cannot process a large number of data due to time constraint. The purpose of this study is to develop a new system for geological surface estimation that has a higher processing speed and breaks through the time constraint. The purpose of this paper is to report on our work. For the purpose of our work, studies have been made on parallel processing of geological field data. We developed a multi-threading system for geological surface estimation using scattered elevation data and trend data. The developed system was evaluated through some calculations. As a result, an increase in the number of threads led to an increase in the processing speed. In conclusion, we realized a higher-speed processing of geological surface estimation than before. Based on the result of calculations, it is recommended that a multi-processing technique be introduced as well as a multi-threading technique.

1. INTRODUCTION

Geological information is quite useful in analyses of a subsurface condition. In particular, geological point data such as borehole data and chemical concentration data play an important role in understanding a subsurface condition. In general, the geological point data are scattered at random in the field. Thus, the point data cannot be used directly in the analyses. The point data must be converted to regular grid data.

In the field of geology, many studies have been conducted on algorithms for generating the regular grid data from the point data, what is called surface estimation algorithms, such as IDW (Inverse Distance Weighted interpolation), spline-fitting, and krigging. In addition, various systems have been developed based on those algorithms. However, most of the systems cannot process massive data and densely-distributed data. If the systems can process those data, it takes an immense amount of time. This becomes a big obstacle to high-resolution geological surface estimation and to rapid 3D geological modeling.

Recently, many kinds of multi-core processers have been developed in the field of computer science. The concern with the parallel processing techniques with multi-core processers has also been growing. For example, multi-processing and multi-threading. The parallel processing techniques enable us to enhance a processing speed of various numerical

analyses. However, there has been few studies that tried to apply parallel processing techniques into geological analyses. In this study, in order to break through the time constraint in geological surface estimation, we have developed a multi-threading system for geological surface estimation.

2. PARALLEL PROCESSING

2.1 Parallel computers

Parallel processing is executed in parallel computer. There are two main types of parallel computers. One is a computer that has a distributed memory architecture. In the distributed memory system, each processer in the system can only access its own local memory (Figure 1a). In order to access data on the memory connected to other processers, it is necessary to explicitly pass messages through interconnect network. The other is a computer that has a shared memory architecture. In the shared memory system, all of the processers can directly access all of the memory in the system through a logically direct connection (Figure 1b).

2.2 Multi-threading

Multi-threading is one of the parallel processing techniques. It can be implemented only in shared memory system (Figure 1b). The term "thread" in here means the smallest unit of processing that shares memory. Figure 2a shows a simple process flow diagram of multi-threading. Each vertical arrows represents a thread. Time is running downward along the arrows. In Figure 2a, a single thread executes a process at a start point. This single thread is referred to as master thread. Then, the master thread creates three slave threads, what is called "fork", and executes the process with the slave threads. Finally, the master thread combines results from each threads, what is called "join", and finishes the process. In general, processing speed of multi-threading is faster than that of single-threading.

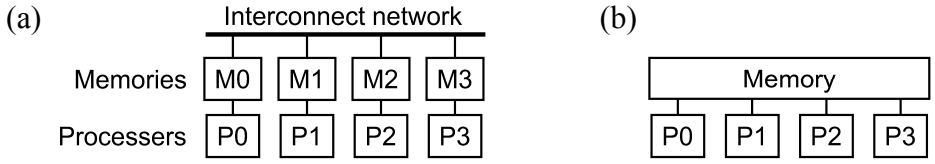

Figure 1. Difference between architectures of parallel computers. (a) distributed memory architecture, (b) shared memory architecture.

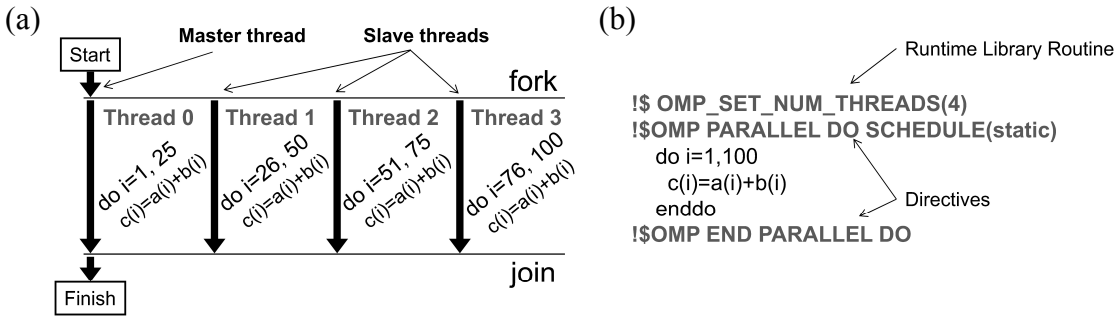

Figure 2. An example of multi-threading. (a) process flow diagram, (b) source code in FORTRAN with OpenMP.

2.3 OpenMP

OpenMP is an application programming interface (API) for multi-threading. It works in conjunction with either FORTRAN or C/C++. OpenMP consists of a set of "Compiler directive", "Runtime library routine", and "Environment variable". The directives describe the parallelism in the source code. They take the form of source code comments in FORTRAN in order to enhance application portability when porting to non-OpenMP environments (Chandra *et al*., 2001). Figure 2b shows an example of multi-threading program in FORTRAN with OpenMP. The program provides a process shown in Figure 2a.

3. METHODOLOGY

3.1 Algorithm for surface estimation

An algorithm proposed by Nonogaki *et al.* (2008) is adopted to estimate a geological surface. The algorithm have been coded in FORTRAN77. In here, we describe only important parts of the algorithm. As for the detail of the algorithm, see Nonogaki *et al.* (2008).

In this algorithm, two types of scattered field data are used as constraints to estimate a geological surface. One is elevation data that constrain the height of the surface. The other is strike-dip data that constrain the trend of the surface. Suppose that a geological surface can be expressed in $z = f(x, y)$. We use "Bi-cubic B-spline function" to express the surface $f(x, y)$. Let $\Omega = \Omega_x \times \Omega_y$ be a rectangular domain in x-y plane. When the domain Ω is divided into $M_x \times M_y$ sections, the surface in the domain Ω can be expressed in a quadratic form:

$$f(x,y) = \sum_{i=1}^{M_x+3} \sum_{j=1}^{M_y+3} c_{ij}\, N_i(x)\, N_j(y) \tag{1}$$

where $N_i(x)$ and $N_j(y)$ are normalized cubic B-spline bases with respect to x and y, respectively. c_{ij} are the constants for the products $N_i(x)N_j(y)$.

There may be many feasible surfaces that satisfy the elevation data and strike-dip data. We assume that an optimal geological surface must be the smoothest one among the feasible surfaces. With the exterior penalty function method, we defined an augmented objective function as follow:

$$Q(f) = \{m_1 J_1(f) + m_2 J_2(f)\} + \alpha \{R_H(f) + \gamma R_D(f)\} \tag{2}$$

where $J_1(f)$ and $J_2(f)$ are functional that evaluate the flatness and smoothness of the surface, respectively. $R_H(f)$ and $R_D(f)$ are functional that evaluate the goodness of fit of elevation data and strike-dip data, respectively. The optimal surface is the one that minimizes an augmented objective function.

3.2 Target part for parallel processing

There are five steps to perform a surface estimation:(i) specify a set of field data, (ii) define a calculation range, (iii) define a set of parameters such as M_x, M_y, m_1, m_2, α and γ, (iv) calculate/determine an optimal geological surface, and (v) generate a DEM (Digital

Elevation Model) for calculated surface. Among the five steps, computational load in the step (iv) is much larger than that in other steps. Thus, we targeted the step (iv) for parallel processing. In particular, we focused on following three parts in the step (iv):
Target part 1: processing for deriving the simultaneous equation (2),
Target part 2: processing for solving the simultaneous equation (2),
Target part 3: processing for evaluating the smoothness of the surface and the goodness of fit.

4. RESULTS

4.1 Evaluation method of effect of multi-threading

Computational load in target parts depends on the number of small cells inside the calculation range (the number of sections inside domain Ω: M_x and M_y). Thus, we evaluate an effect of multi-threading by varying the parameters M_x and M_y as well as the number of threads. For numerical evaluation, we calculate a "speedup" given by following formula:

$$\text{speedup} = \frac{T_1}{T_p} \tag{3}$$

where p is the number of threads, T_1 is the execution time with single thread, and, T_p is the execution time with p threads. The speedup x means that a processing speed with multi-threading is x times faster than the one with single thread. The ideal maximum value of speedup is p.

4.2 Surface estimation using a small number of data

Surface estimation is performed using elevation data modified from TABLE 5.11 in Davis (1986) (Figure 3). The number of data is 52. Table 1 shows a list of calculated speedups. The result shows that the speedups depend not on the parameters M_x and M_y but depend on the number of threads. The same number of threads gives almost the same speedup. The larger the number of threads is, the greater the speedup becomes. The speedup in $M_x = M_y = 25$ is smaller than the ones in other cases. This is because a proportion of fork-join process to the whole process in $M_x = M_y = 25$ is relatively high in comparison with the ones in other cases. When the number of thread is 4, the speedup becomes about 2.

4.3 Surface estimation using a large number of data

As an example of surface estimation using a large number of data, DEM generation is performed based on STRIPE method (Noumi, 2003) (Figure 4). The number of data is 14,691,658. Table 2 shows a list of calculated speedups. The result shows that the speedups depend both on the parameters M_x and M_y and on the number of threads. When M_x and M_y are constant, the larger the number of threads is, the greater the speedup becomes. When the number of threads is constant, the larger M_x and M_y are, the smaller the speedup becomes, except when the number of threads is 1. When the number of thread is 4, the speedup becomes more than 2.5.

The speedups of three target parts for multi-threading do not depend on M_x and M_y. They are almost constant: 3.9-4.0 (part1), 2.2-2.4 (part2), and, 3.9-4.0 (part 3). However, the proportions of three target parts to the whole process drastically changes with a change in the

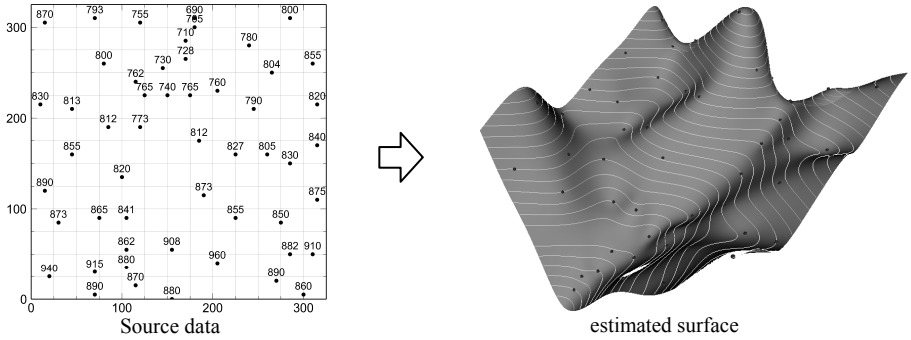

Figure 3. An example of surface estimation using a small number of data.

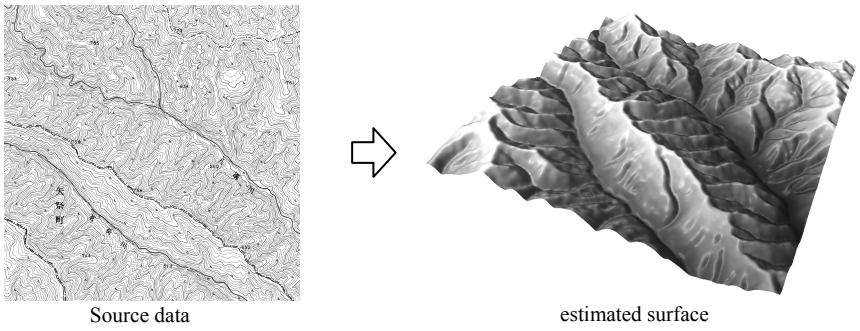

Figure 4. An example of surface estimation using a large number of data.

Table 1. Speedup derived from surface estimation using a small number of data.

$M_x \times M_y$	Number of threads			
	1	**2**	**3**	**4**
25×25	1.00	1.15	1.44	1.64
50×50	1.00	1.33	1.81	2.20
100×100	1.00	1.33	1.75	2.19
200×200	1.00	1.31	1.79	2.30

Table 2. Speedup derived from surface estimation using a large number of data.

$M_x \times M_y$	Number of threads			
	1	**2**	**3**	**4**
50×50	1.00	1.89	2.80	3.74
100×100	1.00	1.75	2.50	3.33
200×200	1.00	1.44	1.96	2.58
300×300	1.00	1.34	1.91	2.50

Table 3. Proportions (%) of three target parts to the whole process.

$M_x \times M_y$	Part 1	Part 2	Part 3	Other
50×50	75.68	2.52	21.71	0.09
100×100	48.82	31.92	18.97	0.29
200×200	6.96	90.07	2.81	0.16
300×300	1.30	98.09	0.54	0.07

parameters M_x and M_y. Table 3 shows the proportion of three target parts and the other parts to the whole process in the surface estimation using a large number of data. When the parameters M_x and M_y are small ($M_x = M_y = 50$ or $M_x = M_y = 100$), the target part 1 and part 3 have higher proportions than the target part 2. When the parameters M_x and M_y are great ($M_x = M_y = 200$ or $M_x = M_y = 300$), the target part 2 has much higher proportion than the target part 1 and part 3. Consequently, it caused a decrease in the speedups associated with an increase in M_x and M_y.

5. CONCLUSIONS

We developed a multi-processing system for geological surface estimation using scattered field data. This system adopts the multi-threading technique to enhance the processing speed. In order to evaluate the effect of multi-threading, we performed surface estimations using a small and larger number of data. The numerical results show that we realized a higher-speed processing of geological surface estimation than before. This will break through a time constraint in the surface estimation and greatly contribute to the high-resolution geological surface estimations using massive data and densely-distributed data. A further direction of this study will be to introduce a multi-processing technique into the current system.

6. ACKNOWLEDGEMENT

This study was supported by KAKENHI (22500094; Grant-in-Aid for Scientific Research by Japan Society for Promotion of Science).

7. REFERENCES

Chandra, R., Dagum L., Kohr, D., Maydan, D., McDonald, J. and Menon, R., 2001. *Parallel Programming in OpenMP*. Morgan Kaufmann.

Davis, J. C., 1986. *Statistics and Data Analysis in Geology* (2nd Ed.). John Wiley & Sons, Inc. New York.

Nonogaki, S., Masumoto, S. and Shiono, K., 2008. Optimal Determination of Geologic Boundary Surface using Cubic B-Spline. *Geoinformatics* 19, 61-77.

Noumi, Y.(2003) Generation of DEM Using Inter-Contour Height Information on Topographic Map. *Journal of Geosciences, Osaka City University*, 46, 217-230.