A dynamic grouping strategy for implementation of the particle filter on a massively parallel computer

Shin'ya Nakano
The Institute of Statistical Mathematics
Tachikawa, Tokyo, Japan.
shiny@ism.ac.jp

Tomoyuki Higuchi
The Institute of Statistical Mathematics
Tachikawa, Tokyo, Japan.
higuchi@ism.ac.jp

Abstract – *A practical way to implement the particle filter (PF) on a massively parallel computer is discussed. Although the PF is a useful tool for sequential Bayesian estimation, the PF tends to be computationally expensive in applying to high-dimensional problems because a enormous number of particles is required in order to appropriately approximate a PDF. One way to overcome this problem is to use large computing resources of a massively parallel computer. However, in implementing the PF on such a massively parallel computer, it is crucial to reduce the time cost for data transfer between different processing elements (PEs). In addition, in using a parallel computer with a multi-dimensional torus network topology, it is necessary to avoid data transfers between nodes distant from each other. The present study proposes a strategy in which the PEs in use are divided into small groups and the grouping is changed at each time step. The resampling is carried out within each group in parallel and data transfers between distant nodes never occur. Therefore, the time cost for data transfer would be greatly reduced and the efficiency is remarkably improved in comparison with the normal PF.*

Keywords: Particle filter, filtering, parallel computing

1 Introduction

The particle filter (PF) [4, 5, 6] is one of useful algorithms for sequential Bayesian estimation. The PF is based on an approximation of a probability density function (PDF) by a large number of particles which is samples drawn from the PDF. The relationship between a predictive PDF and the filtered PDF at the previous step is modeled by computing temporal evolution for each particle, and an approximation of the posterior PDF is obtained by resampling the particles. The PF is applicable to general problems including non-linear and non-Gaussian problems, and it is now widely used in various fields including signal processing and target tracking [3].

Recently, the application of the PF to high-dimensional problems is of growing interest. For example, the fusion between a numerical simulation model with millions of variables and a large amount of meteorological measurement data is an important issue in meteorology and the PF is considered one of potentially effective method for that purpose [10]. However, the PF tends to be computationally expensive in applying to high-dimensional systems [9]. In the case with high-dimensional systems, a enormous number of particles is required in order to appropriately approximate a PDF and computation for temporal evolution is required for each of the enormous particles.

While we proposed a modification of the PF algorithm to overcome the expensive computational cost in our previous work [8], the present study employs a more straightforward approach in which we use large computing resources of a massively parallel computer. As a matter of fact, the PF is basically suitable for parallel computing because the computation for each particle can be carried out independently. Recent distributed computing system allows us to run more than thousands of processes in parallel and thus computation for more than thousands of particles can be done at once.

However, if the PF with a large number of particles is naïvely implemented on parallel computing systems, many data transfers occur randomly between different processing elements (PEs) in the resampling step. Such 'inter-PE' data transfers are difficult to effectively parallelize because of their randomness. In the case that the PF is implemented on a distributed computing system with a multi-dimensional torus network topology, the randomness of the inter-PE data transfers is particularly unfavorable. For example, in a two-dimensional torus network consisting of a large number of nodes as illustrated in Figure 1, while a data transfer with a neighbor four node is fast, a data transfer with a distant node requires multiple hops which would impede the network communication speed and even interfere

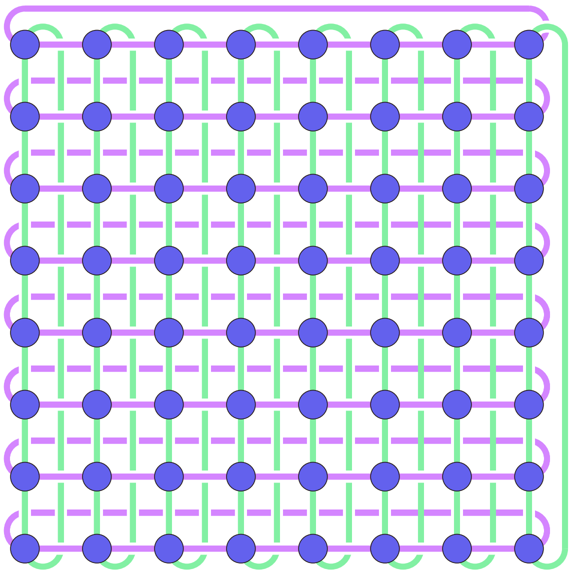

Figure 1: two-dimensional torus network

with data transfers between other nodes.

Although there have been some studies addressing the implementation of the PF on parallel computing systems [1, 2], they did not necessarily consider a massively parallel computer. The present study revisits the concept of grouping and re-grouping originally discussed by a past study by Bolić et al. [2] and extend it so as to be suitable for a massively parallel computer. The idea of this study is originally motivated by the feasibility study for the Next-Generation Supercomputer in Japan, which is currently under development. Since the network topology of this supercomputer is planned to be a virtually three-dimensional torus, our strategy is designed to be suitable especially for a distributed computer with a multi-dimensional torus network topology. However, because of its high parallelity in computation, it would be much more effective than the normal PF even on a distributed computer with another network topology. Indeed, the experiments for evaluating the performance of our strategy shown later are carried out on a distributed computer with a flat-tree network topology.

2 Particle filter

The PF approximates a PDF by an ensemble consisting of a large number of particles which are discrete samples drawn from the PDF. Suppose that a filtered distribution at time $T = t_{k-1}$, $p(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1})$, is approximated by particles $\{\boldsymbol{x}^{(1)}_{k-1|k-1}, \boldsymbol{x}^{(2)}_{k-1|k-1}, \cdots, \boldsymbol{x}^{(N)}_{k-1|k-1}\}$ as

$$p(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1}) \approx \frac{1}{N}\sum_{i=1}^{N}\delta\left(\boldsymbol{x}_{k-1} - \boldsymbol{x}^{(i)}_{k-1|k-1}\right) \qquad (1)$$

where δ is Dirac's delta function, N is the number of particles in the ensemble, and $p(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1})$ means $p(\boldsymbol{x}_{k-1}|\boldsymbol{y}_1, \cdots, \boldsymbol{y}_{k-1})$. We can then obtain an approximation of the predictive distribution of the state at the next observation time $T = t_k$ as

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}) \approx \frac{1}{N}\sum_{i=1}^{N}\delta\left(\boldsymbol{x}_k - \boldsymbol{x}^{(i)}_{k|k-1}\right) \qquad (2)$$

where $\boldsymbol{x}^{(i)}_{k|k-1}$ is given by a system model as

$$\boldsymbol{x}^{(i)}_{k|k-1} \sim p(\boldsymbol{x}_k|\boldsymbol{x}^{(i)}_{k-1|k-1}) \qquad (3)$$

for each i.

From the predictive distribution $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1})$ and observed data $\boldsymbol{y}_k$, we can approximate the filtered PDF $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$ as:

$$\begin{aligned} &p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) \\ &\quad= \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k)\,p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1})}{\int p(\boldsymbol{y}_k|\boldsymbol{x}_k)\,p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1})\,d\boldsymbol{x}_k} \\ &\quad\approx \frac{\sum_{i=1}^{N} p\left(\boldsymbol{y}_k|\boldsymbol{x}^{(i)}_{k|k-1}\right)\delta\left(\boldsymbol{x}_k - \boldsymbol{x}^{(i)}_{k|k-1}\right)}{\sum_{j=1}^{N} p\left(\boldsymbol{y}_k|\boldsymbol{x}^{(j)}_{k|k-1}\right)} \end{aligned} \qquad (4)$$

where $p(\boldsymbol{y}_k|\boldsymbol{x}^{(i)}_{k|k-1})$ is the likelihood of $\boldsymbol{x}^{(i)}_{k|k-1}$ given the data $\boldsymbol{y}_k$, which corresponds to fitness of a particle to the observation $\boldsymbol{y}_k$. If we define a weight $w^{(i)}_k$ as

$$w^{(i)}_k = \frac{p(\boldsymbol{y}_k|\boldsymbol{x}^{(i)}_{k|k-1})}{\sum_j p(\boldsymbol{y}_k|\boldsymbol{x}^{(j)}_{k|k-1})}, \qquad (5)$$

we can rewrite Eq. (4) as follows:

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) = \sum_{i=1}^{N} w^{(i)}_k\,\delta\left(\boldsymbol{x}_k - \boldsymbol{x}^{(i)}_{k|k-1}\right). \qquad (6)$$

In a usual implementation of the PF, a new ensemble $\{\boldsymbol{x}^{(1)}_{k|k}, \cdots, \boldsymbol{x}^{(N)}_{k|k}\}$ is obtained by resampling the predictive ensemble $\{\boldsymbol{x}^{(1)}_{k|k-1}, \cdots, \boldsymbol{x}^{(N)}_{k|k-1}\}$ with replacement such that each particle from the predictive ensemble are drawn with probability proportional to the weight $w^{(i)}_k$. While the new ensemble may contain multiple duplicates of some of particles in the predictive ensemble, some of particles in the predictive ensemble are abandoned. The number of duplicates $m^{(i)}_k$ satisfies

$$m^{(i)}_k \approx N w^{(i)}_k \qquad \left(\sum m^{(i)}_k = N;\; m^{(i)}_k \geq 0\right) \qquad (7)$$

for each particle $\boldsymbol{x}^{(i)}_{k|k-1}$ in the predictive ensemble. Using Eq. (7), Eq. (6) can be further approximated as

$$\begin{aligned} p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) &\approx \sum_{i=1}^{N}\frac{m^{(i)}_k}{N}\delta\left(\boldsymbol{x}_k - \boldsymbol{x}^{(i)}_{k|k-1}\right) \\ &= \frac{1}{N}\sum_{i=1}^{N}\delta\left(\boldsymbol{x}_k - \boldsymbol{x}^{(i)}_{k|k}\right). \end{aligned} \qquad (8)$$

As it turned out, the newly generated filtered ensemble $\{\boldsymbol{x}_{k|k}^{(i)}\}$ approximates the filtered PDF $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$. The procedure of obtaining $\{\boldsymbol{x}_{k|k}^{(i)}\}$ from $\{\boldsymbol{x}_{k|k-1}^{(i)}\}$ is the filtering step. Eq. (8) has the same form as Eq. (1). We can then repeat the procedure from Eq. (1) to Eq. (8) to incorporate a sequence of observed data into the system model over all the steps.

3 Grouping

The efficiency of parallel computing is often highly affected by inter-PE data transfers. It is thus important to reduce the occurrence of inter-PE data transfers in parallel computing. The ensemble-based algorithms such as the PF are basically suitable for parallel computing because the temporal evolution for each particle can be calculated separately in parallel. In the PF, data transfers between different PEs occurs in resampling the particles at the filtering step where some of particles with low likelihood are abandoned and replaced by copies of particles with high likelihood. Whenever the copy substituted for the abandoned particle is taken from a different PE, the inter-PE data transfer is required for routing the copy. Such inter-PE data transfers occur many times between randomly selected PEs. The resampling procedure is difficult to parallelize due to the randomness and it can spoil the computational efficiency of parallel computing systems. In order to avoid this problem, we consider restricting inter-PE data transfers within a small PE group. The particles is thus divided into subsets each of which is assigned to each PE group and resampling is carried out within each subset (i.e. within each PE group). Since the resampling for different subsets can be performed in parallel, the data transfers can also be done in parallel for the resampling for different subsets. This would remarkably improve the computational efficiency.

While the normal PF approximates a PDF by equally weighted particles, we here assume that a filtered PDF at time $T = t_{k-1}$ is approximated by weighted particles as follows:

$$p(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1}) \approx \frac{1}{N}\sum_{i=1}^{N} \zeta_{k-1}^{(i)} \delta\left(\boldsymbol{x}_{k-1} - \tilde{\boldsymbol{x}}_{k-1|k-1}^{(i)}\right). \tag{9}$$

From the ensemble $\{\tilde{\boldsymbol{x}}_{k-1|k-1}^{(i)}\}$, we obtain a new ensemble $\{\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\}$ where each particle of the ensemble is generated by a system model as

$$\tilde{\boldsymbol{x}}_{k|k-1}^{(i)} \sim p(\boldsymbol{x}_k|\tilde{\boldsymbol{x}}_{k-1|k-1}^{(i)}). \tag{10}$$

This ensemble provides an approximation of the predictive PDF as

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}) \approx \frac{1}{N}\sum_{i=1}^{N} \zeta_{k-1}^{(i)} \delta\left(\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\right). \tag{11}$$

Using the same ensemble $\{\tilde{\boldsymbol{x}}_{k-1|k-1}^{(i)}\}$, an approximation of the filtered PDF at time $T = t_{k-1}$ can be obtained as

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) \approx \frac{\sum_{i=1}^{N} \zeta_{k-1}^{(i)} p\left(\boldsymbol{y}_k|\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\right) \delta\left(\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\right)}{\sum_{i=1}^{N} \zeta_{k-1}^{(i)} p(\boldsymbol{y}_k|\tilde{\boldsymbol{x}}_{k|k-1}^{(i)})}. \tag{12}$$

If we define a new weight $\eta_k^{(i)}$ to satisfy

$$\eta_k^{(i)} = \frac{\zeta_{k-1}^{(i)} p(\boldsymbol{y}_k|\tilde{\boldsymbol{x}}_{k|k-1}^{(i)})}{\sum_{i=1}^{N} \zeta_{k-1}^{(i)} p(\boldsymbol{y}_k|\tilde{\boldsymbol{x}}_{k|k-1}^{(i)})} \tag{13}$$

we can rewrite Eq. (12) in the following form:

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) \approx \frac{1}{N}\sum_{i=1}^{N} \eta_k^{(i)} \delta\left(\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\right). \tag{14}$$

We divide the PEs into groups where the number of groups is denoted by Λ. The ensemble of N particles is accordingly divided into Λ subsets, each of which contains $\lambda(= N/\Lambda)$ particles, and each subset is assigned to a different PE group. For example, a predictive ensemble $\{\boldsymbol{x}_{k|k-1}^{(i)}\}_{i=1}^{N}$ is divided into the subsets $\{\boldsymbol{x}_{k|k-1}^{([\mu-1]\lambda+\nu)}\}_{\nu=1}^{\lambda}$ ($\mu = 1, \ldots, \Lambda$), and the particles in each of the subsets are assigned to the corresponding PE according to the value of μ. We then define weights for comparing particles within each group as $\omega_k^{(\mu,\nu)}$ and those for comparing among different groups as $\Omega_k^{(\mu)}$. We constrain $\omega_k^{(\mu,\nu)}$ and $\Omega_k^{(\mu)}$ to satisfy the following conditions:

$$\Omega_k^{(\mu)} \omega_k^{(\mu,\nu)} = w_k^{([\mu-1]\lambda+\nu)}, \tag{15a}$$

$$\sum_{\nu=1}^{\lambda} \omega_k^{(\mu,\nu)} = 1, \tag{15b}$$

$$\sum_{\mu=1}^{\Lambda} \Omega_k^{(\mu)} = 1. \tag{15c}$$

Eq. (14) can thus be rewritten as

$$\begin{aligned} p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) &\approx \sum_{i=1}^{N} \eta_k^{(i)} \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_{k|k-1}^{(i)}\right) \\ &= \sum_{\mu=1}^{\Lambda} \Omega_k^{(\mu)} \sum_{\nu=1}^{\lambda} \omega_k^{(\mu,\nu)} \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_{k|k-1}^{([\mu-1]\lambda+\nu)}\right). \end{aligned} \tag{16}$$

If we resample with replacement within a subset $\{\tilde{\boldsymbol{x}}_{k|k-1}^{([\mu-1]\lambda+\nu)}\}_{\nu=1}^{\lambda}$ such that each particle from the subset are drawn with probability proportional to the weight $\omega_k^{(\mu,\nu)}$, we can obtain a new subset $\{\tilde{\boldsymbol{x}}_{k|k}^{([\mu-1]\lambda+\nu)}\}_{\nu=1}^{\lambda}$ which contains $m_k^{(\mu,\nu)}$ copies of

$\tilde{\boldsymbol{x}}_{k|k-1}^{([\mu-1]\lambda+\nu)}$, where $m_k^{(\mu,\nu)}$ is an integer satisfying

$$m_k^{(\mu,\nu)} \approx \lambda\omega_k^{(\mu,\nu)}, \quad m_k^{(\mu,\nu)} \geq 0, \tag{17}$$

for all μ and ν and

$$\sum_{\nu=1}^{\lambda} m_k^{(\mu,\nu)} = \lambda \tag{18}$$

for all μ as similar to Eq. (7). The new subset also provides an approximation of the filtered PDF $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$ as

$$\begin{aligned} p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) &\approx \sum_{\mu=1}^{\Lambda} \Omega_k^{(\mu)} \sum_{\nu=1}^{\lambda} \frac{m_k^{(\mu,\nu)}}{\lambda} \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_{k|k-1}^{([\mu-1]\lambda+\nu)}\right) \\ &= \sum_{\mu=1}^{\Lambda} \frac{\Omega_k^{(\mu)}}{\lambda} \sum_{\nu=1}^{\lambda} \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_{k|k}^{([\mu-1]\lambda+\nu)}\right). \end{aligned} \tag{19}$$

Since each of the new subsets is obtained using the same algorithm as the PF, each new subset also offers an approximation of the filtered PDF. It should be noted that the approximations by different subsets must have different sampling errors. The integration of all the subsets under consideration with the weights $\Omega_k^{(\mu)}$ would reduce these sampling errors, and reinforce the estimation. Therefore, the whole ensemble consisting of all the subsets would provides a much better approximation than each subset. Defining a weight $\zeta_k^{(i)}$ as

$$\zeta_k^{([\mu-1]\lambda+\nu)} = \frac{\Omega_k^{(\mu)}}{\lambda}, \tag{20}$$

Eq. (19) can be rewritten as

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) \approx \frac{1}{N} \sum_{i=1}^{N} \zeta_k^{(i)} \delta\left(\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_{k|k}^{(i)}\right). \tag{21}$$

Since Eq. (21) has the same form as Eq. (9), we can repeat the procedure from Eq. (9) to Eq. (21) to incorporate the measurement into the system model for all the steps.

4 Switching of the grouping

In the above argument, the grouping of the PEs can be determined arbitrarily. The grouping can even be changed at each time step; that is, PEs which previously embraced different groups can be put together to form a new group [2]. If the grouping is static and no information is exchanged between different groups, the procedure in each group is equivalent to the normal PF with a small subset of the ensemble where the number of particles is λ. If the ensemble size is small, the diversity of the ensemble would be rapidly lost after repeated resampling. Thus, the diversity of each small subset would be rapidly lost, which may spoil estimates by the whole ensemble. On the other hand, if the grouping is changed at each time step, particles assigned to each node are exchanged with various other nodes and the diversity of the ensemble would be maintained.

There are many ways to group the PEs. In choosing the ways of grouping, one important point is to consider the network topology of the computing system. In this study, we assume a two-dimensional torus network topology. We then consider a strategy to use two patterns of grouping which is suitable to the network topology as illustrated in Figure 2. In this strategy, the two grouping patterns are alternately switched. In both of the grouping patterns (Figures 2(a) and 2(b)), each group consists of four nodes forming a square. It should be noted that a uppermost node is adjacent to the lowermost node aligned along the same vertical line and that a leftmost node is adjacent to the rightmost node aligned along the same horizontal line in this two-dimensional torus network. Hence two nodes within the same group on one side in Figure 2 (b) is embraced in the same group as two of the nodes on the other side. Each of the four nodes in a group is adjacent to two of the nodes in the same group and can reach the other node at two hops; that is, data transfers within each group takes two hops at most.

By dynamically switching between the two grouping pattern in Figures (2) (a) and (2) (b), each node is put together with three nodes each of which was embraces in a different group. Particles assigned to each node are thus exchanged with various other nodes. If we consider an 8×8 two-dimensional torus, the information of a particle with high likelihood may propagate to all the nodes only by resampling 4 time steps. At each time step, inter-PE data transfers occur only within each group, and the data transfers in each group can be done in parallel. The time for data transfers would thus be remarkably reduced in comparison with the normal 'global' PF even on a distributed computer with a flat-tree network topology. In the next section, the efficiency and accuracy of the proposed strategy are evaluated by experiments using a nonlinear model with 40 variables on a distributed computing system.

5 Experiments and discussion

In order to demonstrate the effectiveness of the proposed strategy, we compared between the naïve global PF and the proposed alternate switching strategy in terms of both computational efficiency and accuracy. The comparison was carried out by applying both methods to a simple state estimation problem with the Lorenz 96 model [7]. The Lorenz 96 model is a one-dimensional non-linear model described by the following equations:

$$\frac{dx_j}{dt} = (x_{j+1} - x_{j-2})x_{j-1} - x_j + F \tag{22}$$

for $j = 1, \ldots, J$. Here, $x_{-1} = x_{J-1}$, $x_0 = x_J$, and $x_{J+1} = x_1$. In this study, the dimension J was set to be 40. The forcing term F was set to be 8. One time step was set to be 0.005.

In order to generate data for the experiments, we ran this model from the initial condition as

$$x_j = 8.0 \qquad \text{(for } j \neq 20\text{)} \qquad (23a)$$

$$x_j = 8.008 \qquad \text{(for } j = 20\text{)}. \qquad (23b)$$

In order to allow fluctuations in the system to develop sufficiently, we iterated the model through 2000 time steps and we define the time after the 2000 time step as $T = 0$. The artificial data were generated every 10 time steps with errors having standard deviations of 1.5 over 10000 time steps from $T = 0$. It was assumed that we can observe x_j if j is an even number ($j = 2, \ldots, 40$); that is, half of the state variables were observed.

In the experiments of the state estimation using the artificial data, the system noise was given by a Gaussian distribution with zero mean and a diagonal covariance as diag$(0.25, \ldots, 0.25)$. The population at the initial time step ($T = 0$) was generated from a Gaussian distribution with mean 2.0 and variance 2.0 for each component. The likelihood $p(\boldsymbol{y}_k|\boldsymbol{x}_{k|k-1})$ was calculated as follows:

$$p(\boldsymbol{y}_k|\boldsymbol{x}_{k|k-1}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{||\boldsymbol{y}_k - H\boldsymbol{x}_{k|k-1}||^2}{2\sigma^2}\right] \qquad (24)$$

where $\boldsymbol{y}_k$ is the observation vector $(y_{1,k}, \ldots, y_{20,k})$ and σ was set to be 3. The operator H extracts the observable components from the state vector $\boldsymbol{x}_{k|k-1}$. Since we assumed that we can observe x_j for an even number of j, $H\boldsymbol{x}_{k|k-1} = (x_{2,k|k-1} x_{4,k|k-1} \cdots x_{40,k|k-1})^T$.

The experiments were conducted on a distributed computing system of 360 nodes connected by Infiniband. In each node, two quad-core Intel Xeon processors are installed; that is, 2880 cores are available in total. We regarded each core as one PE and each PE is assigned to one node of the two-dimensional torus shown in Figure 1. We carried out the experiments using each of the global PF and the alternate switching method for three cases with different numbers of nodes: 1) 4×4 nodes, 2) 8×8 nodes, and 3) 16×16 nodes. For all the three cases, 4096 particles are assigned to each PE (node); that is, 65536, 262144, and 1048576 particles are used in total for the case 1, 2, and 3, respectively.

Tables 1 and 2 respectively shows the accuracy (RMS deviation from the truth) and elapsed time for the three cases with the global PF and the PF with alternate switching strategy, respectively. As increasing the number of particles, the RMS deviation decreased for both the methods. This means that the accuracy was highly improved with a larger number of particles. Comparing between the two methods, the accuracy obtained by the two methods was comparable. However, elapsed time showed a remarkable advantage of the alternate switching method especially in the case that the number of particles (PEs) is large. When the global PF was employed using 16×16 nodes, the estimation for 10000 time steps took more than 17 minutes. On the other hand, the proposed method completed the estimation within 2 minutes and it achieved comparable accuracy with the global PF.

Table 1: Result using the global PF.

PEs	Particles	RMS dev.	Elapsed time
4×4	65536	0.85	01m12s
8×8	262144	0.77	03m55s
16×16	1048576	0.73	17m18s

Table 2: Result using the alternate switching method.

PEs	Particles	RMS dev.	Elapsed time
4×4	65536	0.84	00m40s
8×8	262144	0.77	00m54s
16×16	1048576	0.73	01m45s

The high computational speed of the proposed method in comparison with the global PF would be mostly due to the parallelization of inter-PE data transfers. Although the proposed method is designed to be suitable for a parallel computer with multi-dimensional torus network, the experiments in this study were carried out on a computer with a flat-tree network topology. Under the flat-free network topology, inter-PE data transfers of different PE groups would interfere with each other and this would be the reason why the elapsed time increased as the number of the nodes increased even in the case with the proposed alternate switching method. If a distributed computer with a multi-dimensional torus network topology is adopted and PE groups are defined like Figure 2, inter-PE data transfers within a PE group do not interfere with those of other PE groups. The elapsed time is thus theoretically independent from the number of the nodes. Therefore we can expect that the proposed method would be remarkably effective for high-dimensional sequential Bayesian estimation.

6 Summary

The present study proposes a dynamic grouping strategy which achieves high efficiency on a massively parallel computing system. In the proposed strategy, the resampling at each time step is performed within each PE group. Data transfers are thus restricted within the PE group. Since the resampling for different PE groups can be done in parallel, the time cost would be remarkably reduced. The grouping is not static but the two grouping patterns are alternately switched. This switching allows us to maintain the diversity of the ensemble because particles assigned to each node are exchanged with various other nodes by changing the grouping. The effectiveness of the proposed strategy was experimentally examined and it has been confirmed that this strategy is notably effective especially for the cases with a large number of nodes in which the normal global PF is prohibitive.

References

[1] A. S. Bashi, V. P. Jilkov, X. R. Li, and H. Chen. Distributed implementations of particle filters. In *Proc. of the Sixth Int. Conf. of Information Fusion*, page 1164, 2003.

[2] M. Bolić, P. M. Djurić, and S. Hong. Resampling algorithms and architectures for distributed particle filters. *IEEE Trans. Signal Processing*, 53:2442, 2005.

[3] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo methods in practice.* Springer-Verlag, New York, 2001.

[4] N. J. Gordon, D. J. Salmond, and A. F. M Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings F*, 140:107, 1993.

[5] G. Kitagawa. Monte Carlo filtering and smoothing method for non-Gaussian nonlinear state space model. *Inst. Statist. Math. Res. Memo.*, 462, 1993.

[6] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *J. Comp. Graph. Statist.*, 5:1, 1996.

[7] E. N. Lorenz and K. A. Emanuel. Optimal sites for supplementary weather observations: Simulations with a small model. *J. Atmos. Sci.*, 55:399, 1998.

[8] S. Nakano, G. Ueno, and T. Higuchi. Merging particle filter for sequential data assimilation. *Nonlin. Process. Geophys.*, 14:395–408, 2007.

[9] C. Snyder, T. Bengtsson, P. Bickel, and J. Anderson. Obstacles to high-dimensional particle filtering. *Mon. Wea. Rev.*, 136:4629, 2008.

[10] P. J. van Leeuwen. Particle filtering in geophysical systems. *Mon. Wea. Rev.*, 137:4089, 2009.

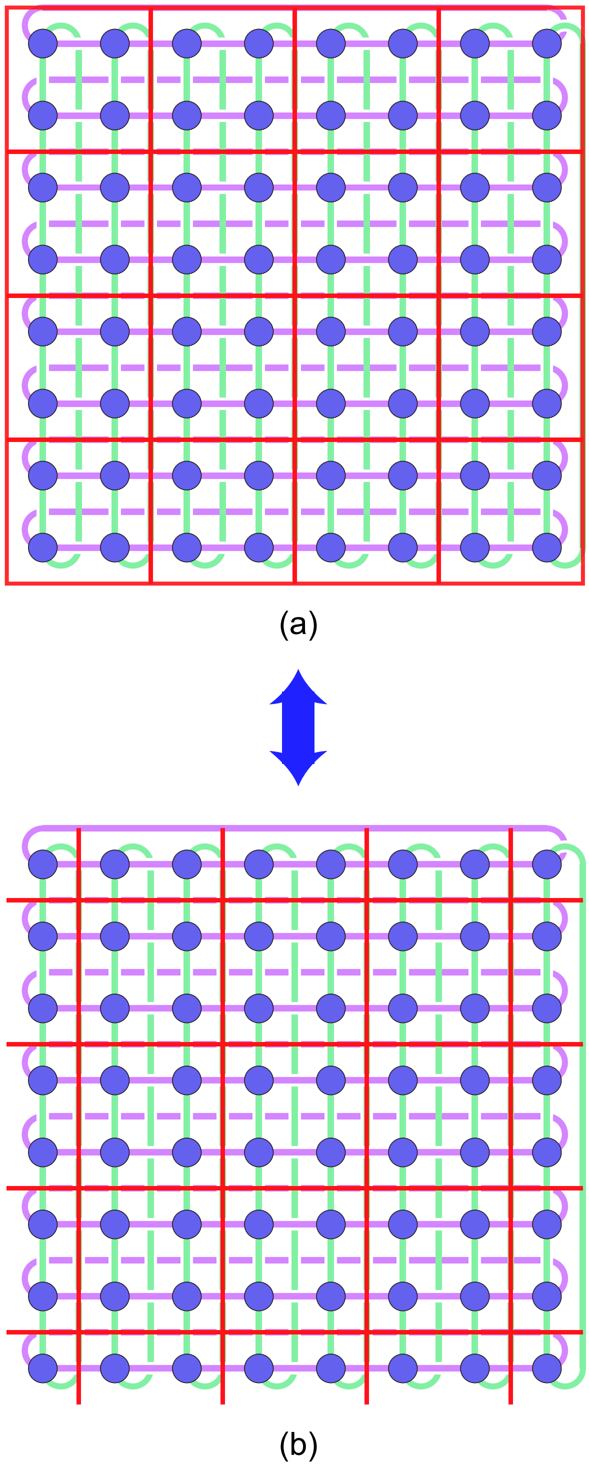

Figure 2: Alternate switching strategy proposed in this study