ADAPTIVE PARTICLE ROUTING IN PARALLEL/DISTRIBUTED PARTICLE FILTERS

Xudong Zhang
Department of Computer Science
Graduate Center of The City University of New York
365 5th Avenue
New York, NY 10016
xzhang5@gradcenter.cuny.edu

Lixin Huang
Department of Electrical Engineering and Computer Science
Syracuse University
4-206 Center for Science and Technology
Syracuse, NY 13244
lhuang24@syr.edu

Evan Ferguson-Hull
Department of Mathematics
Bates College
2 Andrews Road
Lewiston, ME 04240
eferguso@bates.edu

Feng Gu
Department of Computer Science
College of Staten Island
2800 Victory Boulevard
Staten Island, NY 10314
Feng.Gu@csi.cuny.edu

ABSTRACT

Particle filters estimate the state of dynamic systems through Bayesian interference and stochastic sampling techniques. Parallel/distributed particle filters aim to improve the performance by deploying all particles on different processing units. However, the communication cost of transferring particles is high due to the centralized processing in resampling step. To reduce the communication cost without loss of accuracy, the hybrid particle routing policy is designed for the resampling step, which mainly executes particles resampling and exchanges locally and routes them globally every specific number of calculation steps. However, the global particle routing is more necessary when the convergence of particles is low. In this paper, we propose the adaptive particle routing algorithm, in which the local resampling and particle exchange are used, and the planned global particle routing is adopted only when the measured convergence is below the set threshold. The experimental results show the improved performance.

Keywords: particle filters, parallel/distributed computing, adaptive particle routing.

1 INTRODUCTION

Particle filters, also called sequential Monte Carlo (SMC) methods, provide a numerical approximation to the nonlinear filtering problem. Particle filters use Bayesian inference and stochastic sampling techniques to recursively estimate the states of dynamic systems from some given observations (Smith, Schmidt, and McGee 1962; Kailath, Sayed, and Hassibi 2000; Gu 2010; Gustafsson 2010; Helmke and Moore 2012) with little or without assumptions of the system model's properties. Therefore, particle filters have been used in many non-linear and/or non-Gaussian applications, such as positioning, navigation, visual tracking, and wildfire spread systems (Freeman 1987; Ikeda and Matsumoto 1987; Kocarev and Parlitz 1995; Gustafsson et al. 2002; van Leeuwen 2003). In the applications of particle filters, sequential

SpringSim-HPC 2017, April 23-26, Virginia Beach, VA, USA

importance sampling and resampling (SISR) is one of the widely used particle filtering algorithms. The SISR algorithm has two main stages, sampling and resampling. In the sampling stage, a set of particles representing the belief of the system is used to generate a new set of particles to represent the system model. These new particles represent the posterior belief according to the prior distribution. An observation measures the particles by calculating and normalizing the weights of all the particles. In the resampling stage, offspring particles are obtained according to the normalized weights. At each time step, sampling and resampling are executed and the resampled particles will be the input of the sampling of next time step. This procedure continues until the observation is unavailable.

One of the challenges to apply particle filters is the performance due to the used large number of particles, especially for large-scale dynamic systems. To improve the performance, parallel/distributed particle filters are introduced (Bolic, Djuric, and Hong 2005; Sheng et al. 2005; Bai et al. 2016). There are no communications between processing units in the sampling stage. Therefore, the main difference for these algorithms lies in how to route the resampled particles to other processing units in the resampling stage due to its centralized processing. Different particle routing policies define how the processing units with extra particles send particles to those with shortage of particles to achieve the load balance. Although efficient particle routing policies can achieve speedups to some extent, they still suffer from high communication costs. To further enhance the performance, decentralized resampling algorithms are designed (Bolic, Djuric, and Hong 2005), in which the global resampling is removed and only a small percentage of particles are exchanged between processing units after the local resampling on each processing unit. However, it may decrease the accuracy of state estimation due to lack of the global resampling.

To improve the performance without loss of accuracy, a hybrid particle routing policy is adopted (Bai et al. 2016). The hybrid routing policy is mainly based on the decentralized resampling and invokes the centralized resampling every a certain number of calculation steps. Therefore, it combines both the decentralized resampling and the centralized resampling to achieve better speedups and accuracy of the estimated states. However, in many scenarios, the particles are well converged, therefore, the centralized resampling (scheduled at every k time steps) may not be needed. To further improve the performance, we propose the adaptive particle routing policy, in which the decentralized resampling is adopted and the centralized resampling every a certain number of steps is invoked only when the convergence of particles is low. We measure the convergence of particles to decide if the centralized resampling is needed at those scheduled centralized resampling steps. It avoids unnecessary centralized particle routing steps and reduces their incurred extra communication costs.

The rest of the paper is organized as follows. Section 2 introduces the related work in particle filters and parallel/distributed particle filters. Section 3 presents the hybrid particle routing policy in parallel/distributed particle filters. Section 4 describes the proposed adaptive particle routing policy and its algorithm in parallel/distributed particle filters. Section 5 provides the experiments and achieved results. Section 6 concludes the paper and points out the future work.

2 RELATED WORK

The applications of particle filters can be found in a variety of domains, including epidemic predictions, geophysical systems, geosciences and remote sensing, transportation systems, and wildfire spread simulations. Dawson, Gailis, and Meehan (2015) consistently analyzed the probability that a disease happened in a population based on the medical records of the individual of the target popular using particle filters. The results showed the improvement of detection times for outbreaks in populations with electronic medical records available. Mattern, Dowd, and Fennel (2013) assimilated satellite observations of surface chlorophyll into a 3-D biological ocean model to improve its state estimation using particle filters. They tested the feasibility of biological state estimation with particle filters for realistic models. Yan, DeChant, and Moradkhani (2015) estimated soil moisture and soil hydraulic parameters using particle filters. The proposed approach corrected the soil moisture state and estimated the soil hydraulic parameters. Yan, Gu, and Hu (2013) applied particle filters to reconstruct the event like a traffic jam by the collected information of deployed cameras. They detected the slow moving vehicle in the road network to cause the traffic jam. Xue, Gu, and Hu (2012) assimilated temperature data from deployed fire sensors into a wildfire spread simulation model to estimate the fire fronts and the related experimental results verified the improved state estimation.

In many other applications, the parallel/distributed particle filters are adopted to address the performance issue. Ing and Coates (2005) implemented a distributed particle filters algorithm for object tracking in wireless sensor networks. The designed scheme significantly reduced the energy cost of communication. Hong et al. (2006) designed and implemented a flexible resampling mechanism for parallel particle filters in a CMOS process, and then analyzed its complexity and performance. Sutharsan et al. (2012) presented an optimization-based scheduling algorithm for parallel implementation of particle filters and evaluated the effectiveness of the proposed algorithm by the application of multi-target tracking. Hegyi et al. (2007) described two different parallel particle filter algorithms for the state estimation of freeway traffic network. Their accuracy, performance, and communication costs are analyzed and compared. Rosencrantz et al. (2002) developed a decentralized parallel particle filters algorithm to exchange information between nearby platforms in robotic systems. They illustrated the scaling capability to a large team of vehicles. Liu et al. (2009) used parallel particle filters algorithm in face tracking and it worked robustly for cluttered backgrounds and different illuminations. The multi-core parallel computing achieved a good linear speedup compared to its sequential implementation.

From the above applications, there are two main categories of resampling algorithms in parallel/distributed particle filters algorithms, including the centralized resampling algorithm and the decentralized resampling algorithm. Teuilere and Brun (2003) used a centralized approach to parallelize the resampling step and applied it to Doppler-hearing tracking of maneuvering sources, in which a central unit collected the weights from each processing unit, did the resampling, and returned replication factors to each processing unit. Bolic et al. (2005) proposed the decentralized resampling strategies and implemented four versions of parallel/distributed particle filters algorithms. They removed the centralized resampling and utilized the local weight information to decide the exchange of particles between processing units. The centralized resampling and the distributed resampling have their own disadvantages, either achieving low speedups or losing accuracy. Bai et al. (2016) systemically analyzed various centralized resampling and decentralized resampling routing policies and proposed a novel approach to combine both to achieve better speedups without loss of accuracy. This proposed hybrid particle routing policy was based on the decentralized resampling schema and invoked the centralized resampling every k time steps. It was examined by an application of large-scale spatial temporal system, wildfire spread simulation, and exhibited its effectiveness. However, in some cases, the particles distribution is "good" and the scheduled centralized resampling is not needed. Therefore, it should be called as needed for those time steps to reduce the communication cost. Based on this idea, we develop the adaptive particle routing policy and provide details in the following sections.

3 ADAPTIVE PARTICLE ROUTING ALGORITHM

3.1 Parallel/Distributed Particle Filters

There are three main steps in the general particle filters algorithm (SISR algorithm), including sampling, weight computation, and resampling step. Since resampling needs the global information of all particles, it is the main obstacle to parallelize particle filters algorithms. In general, two primary categories of resampling in parallel/distributed particle filters are developed, including centralized resampling and decentralized resampling. In the centralized resampling, there are two types of nodes, the central unit and the processing unit. Sampling and weight computation are independently executed on each processing unit due to no data dependency, and resampling is conducted on the central unit because of the demand of global information. The central unit collects the weights of all particles from all the processing units, performs particle resampling, and transfers the particles between the central unit and the processing units according to different particle routing policies. Figure 1 shows the procedure of the centralized resampling. In the figure, fours processing units (PU1, PU2, PU3, and PU4) send their weights to the central unit (CU), and CU serves as the hub for four processing units to exchange particles after resampling. Different centralized resampling routing policies and their corresponding analysis could be found in (Bai et al. 2016). Decentralized resampling removes the central unit to reduce the communication cost. Sampling, weight computation, and resampling are executed on each processing unit separately. To make "good" particles propagate to other processing units, a specific percentage of particles on each processing unit are sent to its neighboring processing unit at each time step. Figure 2 displays the decentralized resampling schema. In the figure, four processing units (PU1, PU2, PU3, and PU4) perform independent particle filters steps and forward some number of particles to their neighboring processing units in the clockwise order. More decentralized particle routing policies were discussed in (Bolic et al. 2005).

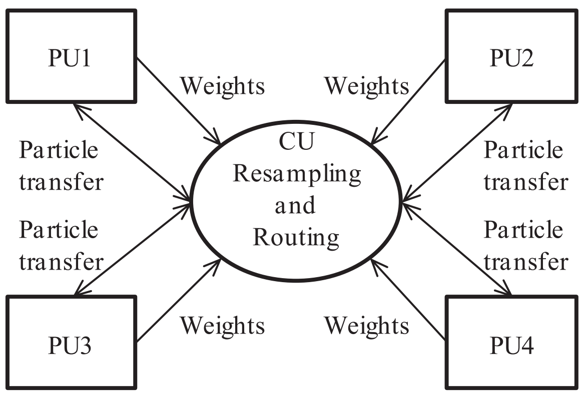

Figure 1: Centralized resampling.

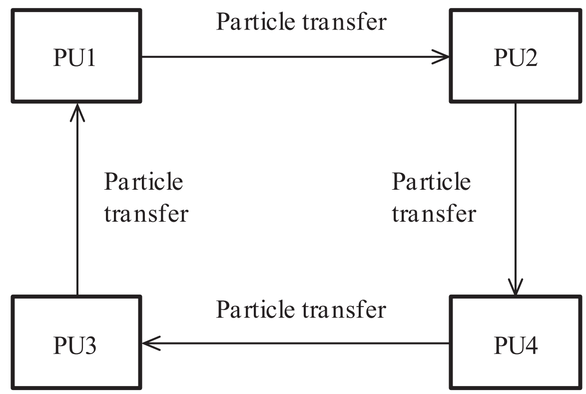

Figure 2: Decentralized resampling.

The centralized resampling schema precisely implements the particle filters algorithm, but suffers from the scalability due to the central unit. The distributed resampling schema improves the scalability, but may need a large number of iterations for fully resampling because of its local nature and limited particle exchanges between processing units. A hybrid routing policy (Bai et al. 2016) was proposed, in which the decentralized resampling was mainly adopted to achieve a large degree parallelism. Processing units performed local resampling and exchanged particles between neighboring processing units. To overcome the limitation of local particle exchanges in the decentralized resampling, the centralized resampling was occasionally invoked to utilize the full knowledge of weights of all particles. It helped quickly and efficiently route "good" particles to all the processing units. This hybrid particle routing policy has been applied in large-scale spatial temporal systems, such as wildfire spread simulation. Through the simulation results, the hybrid particle routing policy greatly improved the performance of the data assimilation of wildfire spread simulation without loss of the state estimation accuracy. More details can be referred to the work in (Bai et al. 2016).

3.2 Adaptive Particle Routing Algorithm

In the adaptive particle routing, both of the decentralized resampling and the centralized resampling are used to achieve both of performance and accuracy of particle filters algorithms. The centralized resampling is invoked every a certain number of steps in the hybrid particle routing policy. However, in many those steps, the particles have good convergence. Therefore, the centralized resampling is not necessary. To more efficiently utilize the centralized resampling, we need to evaluate the convergence of particles to decide its necessity. Towards this objective, we propose the adaptive particle routing algorithm to adaptively invoke the scheduled centralized resampling every k steps when needed. To measure the convergence, we adopt the effective sample size $\widehat{N}_{eff}$, as defined in Equation (1).

$$\widehat{N}_{eff} = \frac{1}{\sum_{i=1}^{N}(\tilde{q}_t{}^{(i)})^2}, \tag{1}$$

where $\tilde{q}_t{}^{(i)}$ is the normalized weight of particle i at time step t, and N is the number of particles. A threshold is used and the centralized resampling scheduled at every specific number of times steps is invoked if the effective sample size of particles is smaller than the predefined threshold. Table 1 lists the adaptive particle routing algorithm.

Table 1: Adaptive Particle Routing Algorithm.

Processing unit side:

for all the parallel processing units at time step *t*

1. Run the sampling step.
2. Calculate the importance weight of each particle.
3. Send all weights to the central unit.
4. Receive information from the central unit. If the centralized resampling needs to be performed, go to step 5, otherwise go to step 9.
5. Receive routing information from the central unit.
6. If having surplus of particles, send the selected particles (based on the received routing information from the central unit) to the central unit.
7. If having shortage of particles, receive particles from the central unit.
8. End.
9. Normalize, perform resampling locally, and send partial particles to the neighboring processing units in the clockwise order .
10. End.

Central unit side:

At every *k* time step

1. Predefine the threshold *TD*.
2. Receive the weight of each particle from all processing units.
3. Calculate the normalized importance weights of all particles and the effective particle size $\widehat{N}_{eff}$

$$\widehat{N}_{eff} = \frac{1}{\sum_{i=1}^{N}\left(\tilde{q}_t{}^{(i)}\right)^2}$$

4. If $\widehat{N}_{eff} < TD$, go to step 5 (activate the centralized resampling), otherwise skip the following and inform all the processing units whether global resampling is needed.
5. Exert the centralized resampling and compute routing information.
6. Send the routing information to processing units.
7. Receive particles from processing units that have surplus of particles.
8. Send particles according to the routing information to the processing units that have shortage of particles.
9. End.

4 EXPERIMENTS AND RESULTS

In order to evaluate the performance of the adaptive particle routing algorithm, we implemented the sequential particle filters algorithm, the parallel particle filters algorithm with the hybrid particle routing policy and the parallel particle filters algorithm with the adaptive particle routing policy. The SISR algorithm was applied to the following system with the system equation in Equation (2) and the measurement equation in Equation (3). In the equations, x_{t+1} and x_t are the system state at time step *t*+1 and time step *t* respectively; y_t is the measurement variable at time step *t*; ω_t and e_t are the system noise and measurement noise at time step *t*. In the above system, the associated configurations are: $x_0 \sim N(0,5)$, $\omega_t \sim N(0,10)$, and $e_t \sim N(0,1)$. This system has been analyzed in many particle filters publications (Gordon et al. 1993; Kitagawa 1996; Doucet 1998; Arulampalam et al. 2002). We will also use this system to evaluate our proposed adaptive particle routing policy in parallel particle filters algorithms. We compare the accuracy among the sequential implementation, the parallel implementation with the hybrid particle routing policy, and the parallel implementation with the adaptive particle routing policy, and also compare the performance of these two parallel implementations. We run the all the experiments with 15,000 time steps using 10,000 particles. We present the results below.

$$x_{t+1} = \frac{x_t}{2} + \frac{25x_t}{1+x_t^2} + 8\,cos(1.2t) + \omega_t \tag{2}$$

$$y_t = \frac{x_t^2}{20} + e_t \tag{3}$$

Figure 3, Figure 4, and Figure 5 show the plots for the sequential particle filters implementation, the parallel particle filters implementation with the hybrid particle routing policy, and the parallel particle filters implementation with the adaptive particle routing policy respectively. In the figures, the horizontal axis and the vertical axis refer to the time step and the state respectively, and the blue line and the red line represent the true states and the estimated states respectively. From the figures we know that the estimated states are close to the true states by applying the observations into the system model for all the three implementations. To compare the accuracy, we calculate the time-averaged root mean square error (RMSE) as defined in Equation (4) for the three cases, where *R* is the calculated time-averaged RMSE, $\hat{x}_t$ is the estimated state at time step *t*, x_t is the true state at time step *t*, and *T* is the total number of time steps. The calculated time averaged RMSEs for the sequential particle filters implementation, the parallel particle filters implementation with the hybrid particle routing policy, and the parallel particle filters implementation with the adaptive particle routing policy are 0.103,00, 0.091,98, and 0.092,00 respectively. They are small and very close, which further indicates all of the three implementations are able to estimate the system states with high accuracy and their estimated accuracies are similar.

$$R = \frac{1}{T}\sqrt{\sum_{t=1}^{T}(\hat{x}_t - x_t)^2} \tag{4}$$

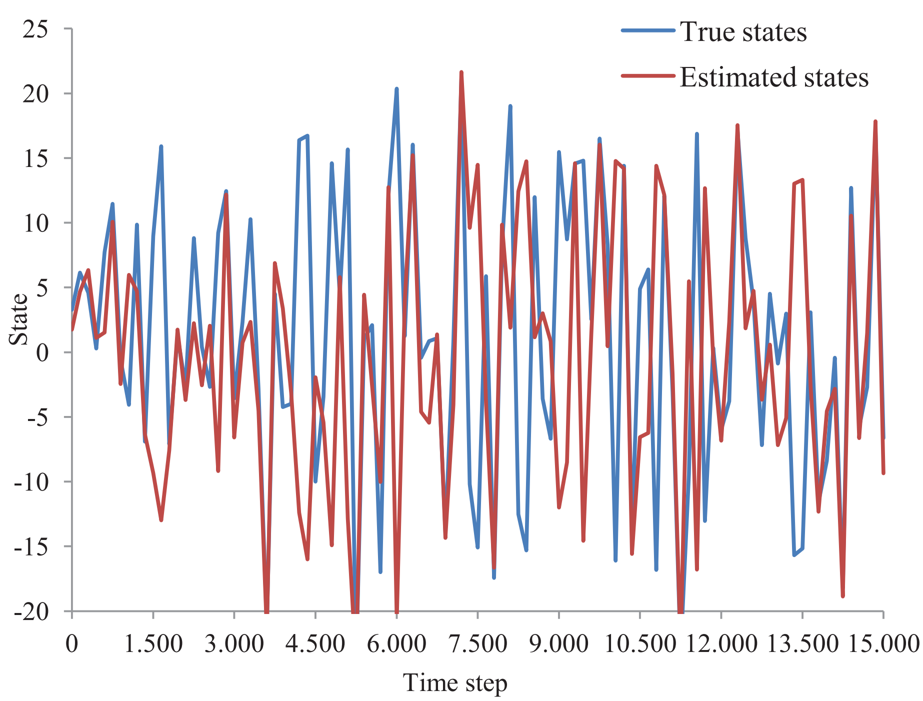

Figure 3: Sequential particle filters.

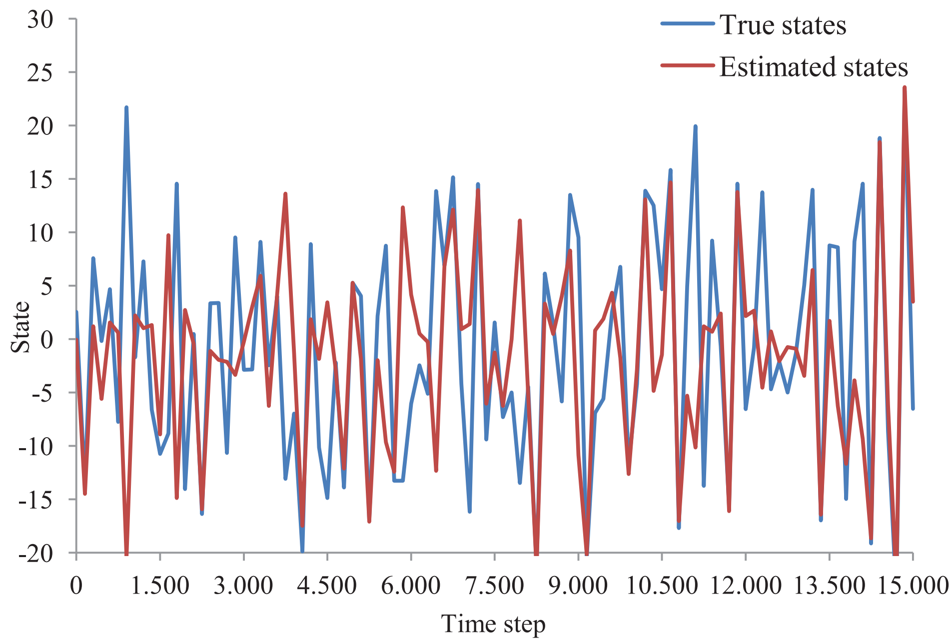

Figure 4: Parallel/distributed particle filters using hybrid particle routing policy.

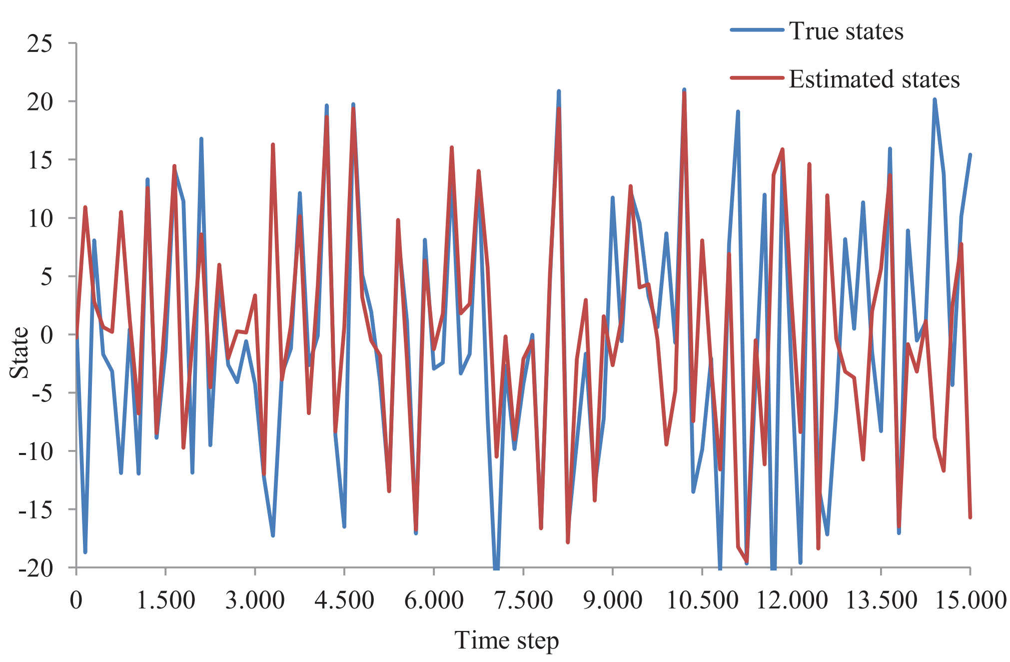

Figure 5: Parallel/distributed particle filters using adaptive particle routing policy.

We also compare the performance of the two parallel particle filters implementations using the hybrid particle routing policy and the adaptive particle routing policy. Firstly, we calculate the number of transferred particles for both of the algorithms during the execution. Figure 6 display the numbers of transferred particles for the parallel particle filters implementations with both of the particle routing policies. In Figure 6, the horizontal axis and the vertical axis represent the time step and the number of transferred particles (in thousand) respectively, and the green line and blue lines represent the numbers of transferred particles for the parallel implementations with the hybrid particle routing policy and the adaptive particle routing policy respectively. It indicates that the number of transferred particles for the parallel particle filters with the adaptive particle routing policy is smaller than that for the parallel particle filters with the hybrid particle routing policy, because the former policy avoids the unnecessary global resampling. Therefore, the communication cost of the former is less than that of the latter. The time consumptions for the parallel particle filters with the adaptive particle routing policy and the parallel particle filters with the hybrid particle routing policy are 56.9 seconds and 68.1 seconds respectively, which is consistent with the results of the number of transferred particles in Figure 6.

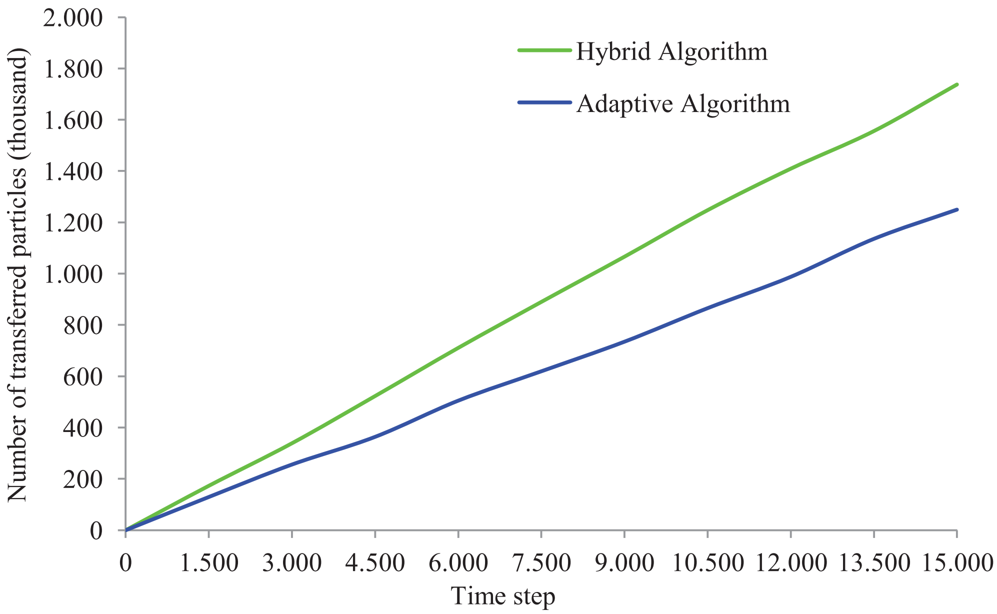

Figure 6: Number of transferred particles for parallel particle filters with the hybrid particle routing policy and the adaptive particle routing policy.

5 CONCLUSIONS AND FUTURE WORK

Parallel/distributed particle filters are able to improve the performance by deploying the particles on multiple processing units. However, the high communication costs among multiple processing units for particles transfer in the resampling step decrease the entire performance. Although the decentralized particle routing policy can address this issue, the accuracy may be affected due to the local resampling on processing units and limited particle exchanges between processing units. The hybrid particle routing policy is based on the decentralized resampling schema and occasionally invokes the centralized resampling to achieve the speedups with the similar accuracy. The adaptive particle routing policy is able to avoid the unnecessary centralized resampling steps by measuring the convergence of particles in order to further improve the performance. The designed experiments show the parallel particle filters with the adaptive particle routing policy achieves better speedups without loss of accuracy. This will have an important impact on performance improvement for parallel particle filters applications, especially those large-scale dynamic systems due to their high dimensions and large system states. Our future work will focus on the following directions. Firstly, we will systematically analyze the theoretical communication and computation cost for different particle routing policies and measure their performances using the defined metrics. Secondly, we will apply the proposed adaptive particle routing algorithm in the large-scale spatial temporal systems to achieve better performance.

ACKNOWLEDGMENTS

This work was supported by the U.S. National Science Foundation under Grant 1356977 and 1359266.

REFERENCES

Arulampalam, M. S., S. Maskell, N. Gordon and T. Clapp. 2002. "A Tutorial on Particle Filters for Online Nonlinear/non-Gaussian Bayesian Tracking". *IEEE Transactions on Signal Processing* vol. 50, pp. 174-188.

Bai, F., F. Gu, X. Hu, and S. Guo. 2016. "Particle Routing in Distributed Particle Filters for Large-Scale Spatial Temporal Systems". *IEEE Transactions on Parallel and Distributed Systems* vol. 27, pp. 481-493.

Bolic, M., P. M. Djuric, and S. Hong. 2005. "Resampling Algorithms and Architectures for Distributed Particle Filters". *IEEE Transactions on Signal Processing* vol. 53, pp. 2442-2450.

Dawson, P., R. Gailis, and A. Meehan. 2015. "Detecting Disease Outbreaks Using a Combined Bayesian Network and Particle Filter Approach". *Journal of Theoretical Biology* vol. 370, pp. 171-183.

Doucet, A. 1998. "On Sequential Simulation-based Methods for Bayesian Filtering". Technical Report CUED/F-INFENG/TR.310, Signal Processing Group, Department of Engineering, University of Cambridge, UK

Freeman, W. J. 1987. "Simulation of Chaotic EEG Patterns with a Dynamic Model of the Olfactory System". *Biological Cybernetics* vol. 56, pp. 139-150.

Gordon, N. J., D. J. Salmond, and A. F. Smith. 1993. "Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation". In *IEE Proceedings F-Radar and Signal Processing* vol. 140, pp. 107-113.

Gu, F. 2010. *Dynamic Data Driven Application System for Wildfire Spread Simulation*. Ph.D. thesis, Department of Computer Science, Georgia State University, Atlanta, Georgia. Available via http://scholarworks.gsu.edu/cs_diss/57. Accessed Dec. 14, 2010.

Gustafsson, F. 2010. "Particle Filter Theory and Practice with Positioning Applications". *IEEE Aerospace and Electronic Systems Magazine* vol. 25, pp. 53-82.

Gustafsson, F., F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. J. Nordlund, 2002. "Particle Filters for Positioning, Navigation, and Tracking". *IEEE Transactions on Signal Processing* vol. 50, pp. 425-437.

Hegyi, A., L. Mihaylova, R. Boel, and Z. Lendek. 2007. "Parallelized Particle Filtering for Freeway Traffic State Tracking". In *Control Conference (ECC), 2007 European* pp. 2442-2449.

Helmke, U., and J. B. Moore. 2012. *Optimization and Dynamical Systems*. Springer Science & Business Media.

Hong, S., S. S. Chin, P. M. Djurić, and M. Bolić. 2006. "Design and Implementation of Flexible Resampling Mechanism for High-speed Parallel Particle Filters". *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology* vol. 44, pp. 47-62.

Ikeda, K., and K. Matsumoto. 1987. "High-dimensional Chaotic Behavior in Systems with Time-delayed Feedback". *Physica D: Nonlinear Phenomena* vol. 29, pp. 223-235.

Ing, G., and M. J. Coates. 2005. "Parallel Particle Filters for Tracking in Wireless Sensor Networks". In *IEEE 6th Workshop on Signal Processing Advances in Wireless Communications,* pp. 935-939.

Kailath, T., A. H. Sayed, and B. Hassibi. 2000. *Linear Estimation*. Upper Saddle River, New Jersey, Prentice Hall, Inc.

Kitagawa, G. 1996. "Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models". *Journal of Computational and Graphical Statistics* vol. 5, pp. 1-25.

Kocarev, L., and U. Parlitz. 1995. "General Approach for Chaotic Synchronization with Applications to Communication". *Physical Review Letters* vol. 74, p. 5028-5031.

Liu, K. Y., S. Q. Li, L. Tang, L. Wang, and W. Liu. 2009. "Fast Face Tracking Using Parallel Particle Filter Algorithm". In *2009 IEEE International Conference on Multimedia and Expo*, pp. 1302-1305.

Mattern, J. P., M. Dowd, and K. Fennel. 2013. "Particle Filter‐Based Data Assimilation for a Three‐Dimensional Biological Ocean Model and Satellite Observations". *Journal of Geophysical Research: Oceans* vol. 118, pp. 2746-2760.

Rosencrantz, M., G. Gordon, and S. Thrun. 2002. August. "Decentralized Sensor Fusion with Distributed Particle Filters". In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence,* pp. 493-500. Morgan Kaufmann Publishers Inc..

Sheng, X., Y. H. Hu, and P. Ramanathan. 2005. "Distributed Particle Filter with GMM Approximation for Multiple Targets Localization and Tracking in Wireless Sensor Network". In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*. IEEE Press.

Smith, G. L., S. F. Schmidt, and L. A. McGee. 1962. *Application of Statistical Filter Theory to the Optimal Estimation of Position and Velocity on Board a Circumlunar Vehicle*. National Aeronautics and Space Administration.

Sutharsan, S., T. Kirubarajan, T. Lang, and M. McDonald. 2012. "An Optimization-based Parallel Particle Filter for Multitarget Tracking". *IEEE Transactions on Aerospace and Electronic Systems* vol. 48, pp. 1601-1618.

Teulière, V., and O. Brun. 2003. "Parallelisation of the Particle Filtering Technique and Application to Doppler-bearing Tracking of Maneuvering Sources". *Parallel Computing* vol. 29, pp. 1069-1090.

van Leeuwen, P. J. 2003. "A Variance-minimizing Filter for Large-scale Applications". *Monthly Weather Review* vol. 131, pp. 2071-2084.

Xue, H., F. Gu, and X. Hu. 2012. "Data Assimilation Using Sequential Monte Carlo Methods in Wildfire Spread Simulation". *ACM Transactions on Modeling and Computer Simulation (TOMACS)* vol. 22, p.23-26.

Yan, H., C. M. DeChant, and H. Moradkhani. 2015. "Improving Soil Moisture Profile Prediction with the Particle Filter-Markov Chain Monte Carlo Method". *IEEE Transactions on Geoscience and Remote Sensing* vol. 53, pp. 6134-6147.

Yan, X., F. Gu, X. Hu, and C. Engstrom. 2013. "Dynamic Data Driven Event Reconstruction for Traffic Simulation Using Sequential Monte Carlo Methods". In *Proceedings of the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World*, pp. 2042-2053. IEEE Press.

AUTHOR BIOGRAPHIES

XUDONG ZHANG is a PhD student in the Department of Computer Science at the Graduate Center of The City University of New York. His research interests include modeling and simulation, and high performance computing. His email address is xzhang5@gradcenter.cuny.edu.

LIXIN HUANG is an undergraduate student majoring in computer engineering at Syracuse University. Her research interests include high performance computing, social computing and digital humanities. Her email address is lhuang24@syr.edu.

EVAN FERGUSON-HULL is an undergraduate student majoring in mathematics at Bates College. His email is eferguso@bates.edu.

FENG GU is an Assistant Professor in the Department of Computer Science at College of Staten Island, The City University of New York. He holds a Ph.D. in Computer Science from Georgia State University. His research interests include modeling and simulation, high performance computing, and bioinformatics. His email address is Feng.Gu@csi.cuny.edu.