Performance of Replica-Exchange Wang-Landau Sampling for the 2D Ising Model: A Brief Survey

Yiwei Zhao\textsuperscript{a}, Siu Wun Cheung\textsuperscript{a}, Ying Wai Li\textsuperscript{b,\,*}, Markus Eisenbach\textsuperscript{b}

\textsuperscript{a}Department of Mathematics, The Chinese University of Hong Kong, Shu Tin, Hong Kong S.A.R.
\textsuperscript{b}National Center for Computational Sciences, Oak Ridge National Laboratory, Oak Ridge, TN 37831, U.S.A.

Abstract

We report a brief performance study of the replica-exchange Wang-Landau algorithm, a recently proposed parallel realization of Wang-Landau sampling, using the 2D Ising model as a test case. The simulation time is found to scale inversely with the square root of the number of subwindows (and thus number of processors) used to span the global parameter space. We also investigate the “time profiles” for random walkers in different subwindows to complete iterations, which will aid the development of an adaptive load-balancing scheme.

© 2014 Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/3.0/).

Peer-review under responsibility of The Organizing Committee of CSP 2014 conference

Keywords: Wang-Landau sampling; Monte Carlo; replica exchange; performance; Ising model

1. Introduction

The Wang-Landau (WL) algorithm [Wang et al. (2001a,b), Landau et al. (2004)] is an effective Monte Carlo sampling method to obtain the density of states (DOS) for a physical system. Depending on the system in question, the DOS can span a wide energy domain either discretely or continuously. Hence it can become difficult for the original, serial WL scheme to reach convergence. Furthermore, the simulation speed becomes significantly slower as the complexity of the system increases and the energy range becomes larger. The replica-exchange Wang-Landau (REWL) sampling method [Vogel et al. (2013)] is recently proposed to achieve parallelization of the algorithm, in which the energy range is split into several subwindows and the simulation task is distributed over multiple processors.

2. Simulation Method

We follow the REWL setup introduced in Ref. [Vogel et al. (2013)]. In particular, we choose to divide the entire energy range into equally sized, overlapping subwindows in our simulations. Figure 1a is an example of an energy domain divided into three equal-sized subwindows, numbered as 0, 1, 2, with an overlap ratio of 80%. We allow

* Corresponding author.
E-mail address: yingwai.li@mailaps.org
overlap between neighboring processors with a large overlap ratio so as to ensure a reasonable acceptance probability for replica exchange. Given the size of the energy range \( R \), the total number of subwindows \( N \) with an overlapping ratio \( \mu \), we are able to determine the size for a single subwindow \( L \) by

\[
L = \frac{R}{N - N\mu + \mu}.
\] (1)

We investigated a two-dimensional Ising model. At the beginning of the simulation, a random configuration is generated for each of the processors. We repeatedly generate random initial configurations until the energy falls within the subwindow’s energy range. We then proceed with generating a Markov chain of configurations: picking one spin at a time and decide the acceptance of the move according to

\[
P_{\text{flip}} = P(S_1 \rightarrow S_2) = \min \left\{ 1, \frac{g(E(S_1))}{g(E(S_2))} \right\},
\] (2)

where \( g \) is the instantaneous DOS estimation; the function \( E(S) \) yields the energy of configuration \( S \); \( S_1 \) and \( S_2 \) are the current and proposed configurations, respectively. If the proposed move is accepted, we multiply \( g(E(S_2)) \) by the current modification factor \( \lambda \), giving \( g* = g(E(S_2)) = \lambda g(E(S_2)) \). Otherwise, the configuration stays at \( S_1 \) and \( g(E(S_1)) \) is updated, as in the standard WL procedure.

During the simulation, replica exchange moves between neighboring energy subwindows are proposed after a predetermined number of Monte Carlo (MC) sweeps. The replica exchange can occur only if both configurations fall within the neighboring subwindow’s energy range. It is accepted with a probability similar to the Wang-Landau algorithm, with the difference that the acceptance ratio depends on both \( g_1 \) and \( g_2 \) to satisfy detailed balance:

\[
P_{\text{exch}} = P((S_1, S_2) \rightarrow (S_2, S_1)) = \min \left\{ 1, \frac{g_1(E(S_1))g_2(E(S_2))}{g_1(E(S_2))g_2(E(S_1))} \right\},
\] (3)

here, \( S_1 \) and \( S_2 \) are the replica exchange candidates from the two neighboring subwindows having the densities of states \( g_1 \) and \( g_2 \) respectively. After a certain number of replica exchanges, all processors perform a flatness test for their individual histograms. A schematic procedure is illustrated in Figure 1b. Those processors passing the test can proceed to the next iteration with a reduced modification factor, \( \lambda \rightarrow \sqrt[2]{\lambda} \). The simulation ends when ALL processors reach the specified target modification factor.

We also employ multiple walkers with the same energy range, which is proven to reduce the error of the estimate of the DOS by \( \sqrt{m} \) [Vogel et al. (2013)], where \( m \) is the number of walkers using the same energy subwindow. When performing replica exchanges, the pairing of exchange partners follows a cyclic pattern, i.e., walkers using the same subwindow take turns to perform replica exchanges with a particular walker in the neighboring subwindow, to allow for a thorough “mixing” of replicas.

In the original REWL scheme in Ref. [Vogel et al. (2013)], all walkers in the same subwindows have to attain flat histograms and average out the estimators of DOS before proceeding to the next iteration. We relax this requirement and allow different random walkers to proceed independently, and terminate the simulation when all of them get to the final modification factor. We believe that this minimizes the possibility of introducing latent correlations for DOS pieces in the same subwindow, and reduces synchronizations during the course of the simulation. However, these will have to be studied more in depth before drawing any conclusions.

### 3. Results and Discussions

The performance and speedup factors of REWL using different number of subwindows is investigated. The input parameters for the experiments using the setup shown in Figure 2a are specified as follows:
(a) An example of dividing the energy domain. The global energy range is divided into 3 subwindows with 80% overlapping region with the adjacent windows.

(b) Procedure of REWL for one processor. Replica exchanges are proposed every a number of MC sweeps, flatness of histogram is tested every a number of replica exchanges. The modification factor is reduced and the simulation proceeds to the next iteration if the flatness test is passed.

Fig. 1: (color online) Basic set up of the replica-exchange Wang-Landau algorithm.

| Parameter specification | Value |
|-------------------------|-------|
| Model size              | 8 × 8 |
| ln(λ_{initial})         | 1     |
| ln(λ_{final})           | $1 \times 10^{-8}$ |
| Histogram flatness criterion | 0.8 |
| Number of MC sweeps between replica exchanges | $1 \times 10^3$ |
| Number of replica exchanges between flatness tests | 100 |
| Number of subwindows    | 1, 3, 6, 12 |
| Subwindow overlap ratio | 0.75 |

In our simulations, 10 independent runs are performed for each of the four cases with 1, 3, 6, 12 subwindows. In the single walker case, one walker is assigned to each subwindow; while in the multiple walkers case, the number of walkers are adjusted to make up a total of 24 processors in a run. For example, in the case of 3 subwindows with multiple walkers, 10 independent experiments are conducted using 3 subwindows with 8 walkers each. In fixing the number of processors used in each simulation, we compare the efficiency of different experimental settings and try to find the optimal set of parameters (i.e., the one which strikes a balance between converging time and number of processors employed).

We first compare the effect of using a single walker and multiple walkers in an energy subwindow. Figure 2a shows the dependence of speedup on the number of subwindows for both the single walker and multiple walkers cases. The number of MC sweeps taken by the slowest walker to terminate the simulation is used as a measure of simulation time. For both cases, simulation time decreases with the increase in the number of subwindows, following a power law relationship. We further quantify this by performing a linear fit to the logarithmic plot for the single walker case assuming $t = CN^D$ (and thus $\ln t = D \ln N + \ln C$), with $N$ representing the number of subwindows and $t$ denoting the number of MC sweeps of the slowest walker to finish the simulation. It is found from least-squared fit that $C = (1.615 \pm 0.047) \times 10^7$ and $D = -0.507 \pm 0.031$. This suggests that using $N$ subwindows for a simulation will
generally lead to a factor of $\sqrt{N}$ of reduction in computational time. The difference in terminating time for the single and multiple walkers cases is not significant, except when the number of subwindows is relatively small.

![Graph showing dependence of simulation time on number of subwindows](chart1)

(a) Dependence of simulation time $t$ (measured by the number of MC sweeps per walker) on the number of subwindows for both the single walker and multiple walkers cases. Log scales are used for both axes; the linear fit for the single walker case in the plot indicates that $t \propto N^{-1/2}$ approximately.

(b) Time profiles (number of iterations completed as function of MC time) for typical random walkers from different subwindows. Notice that the difference in the number of MC sweeps needed to finish an iteration in different subwindows starts to be significant roughly after the 20th iteration, which suggests the need for a load-balancing operation at that point.

Fig. 2: (color online) Effects of subwindow settings on simulation time of REWL.

Next, we briefly explore the number of MC steps required to finish an iteration in different subwindows during a simulation, with simulation parameters set as below:

| Parameter specification                  | Value |
|-----------------------------------------|-------|
| Model size                              | $16 \times 16$ |
| $\ln(\lambda_{\text{initial}})$       | 1     |
| $\ln(\lambda_{\text{final}})$         | $1 \times 10^{-8}$ |
| Histogram flatness test threshold       | 0.8   |
| Number of MC sweeps between replica exchanges | $1 \times 10^3$ |
| Number of replica exchanges between flatness tests | 100 |
| Number of subwindows                    | 3     |
| Number of walkers per subwindow         | 8     |
| Subwindow overlap ratio                 | 0.75  |

Figure 2b shows the number of MC steps required to finish a certain number of iterations for the three subwindows employed: subwindows #0, #1 and #2 are responsible for the lowest, middle and highest energy regimes, respectively. All walkers in all subwindows have approximately the same pace for the first 20 iterations, after which the starts to deviate. The processor in subwindow #1 proceeds in a moderate speed, while walkers in the other two subwindows significantly slow down and lag behind. This is believed to be due to the larger difference in the order of magnitudes in the DOS needed to be covered by the “wing windows”. One possible solution, as suggested in Ref. [Vogel et al. (2013)], is to adopt a runtime-balanced subwindow scheme where subwindow sizes are determined in such a way that the workload for the walkers to finish are roughly the same. It is equivalent to bringing all the time profiles for different walkers in Figure 2b to coincide as much as possible. While, in principle, the adjustment of subwindow ranges can be repeated adaptively during the course of the simulation, it is probably more efficient to do so at points where the paces of the random walkers start to deviate, e.g., after $2 \times 10^7$ or $3 \times 10^7$ MC sweeps have been carried out in this example. This provides a measure for the determination of the need for load-balancing; whereas different ways of subwindow size adjustment are under investigation.
4. Conclusion

The performance of replica-exchange Wang-Landau (REWL) algorithm on the study of a 2D Ising model is investigated; some preliminary findings are presented in this article. We have particularly focused on the effects of the number of equal-sized subwindows on the simulation time required for convergence (as measured by MC sweeps), which is found to obey a power law relation. We looked further into the breakdown of the number of MC sweeps taken for walkers in different subwindows to complete each WL iteration (presented here), as an intermediate step to work towards a systematic scheme for runtime-balancing the workload of different walkers (work in progress).

Acknowledgements

Y. Zhao and S. W. Cheung were partly supported by the Joint Institute for Computational Sciences through the CSURE REU program under NSF Grant number 1262937. Y. W. Li and M. E. were sponsored by the Office of Advanced Scientific Computing Research; U.S. Department of Energy. Part of the work was performed at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725. Computing resource was supported by the National Science Foundation under Grant numbers 0711134, 0933959, 1041709, and 1041710 and the University of Tennessee through the use of Kraken at the National Institute for Computational Sciences.

References

Landau, D. P., Tsai, S.-H., Exler, M., 2004. A new approach to Monte Carlo simulations in statistical physics: Wang-Landau sampling. American Journal of Physics 72, 1294.

Vogel, T., Li, Y. W., Wüst, T., Landau, D. P., 2013. A general parallel framework for Wang-Landau sampling. Physical Review Letters 110, 210603.

Wang, F., Landau, D. P., 2001a. Efficient, multiple-range random walk algorithm to calculate the density of states. Physical Review Letters 86, 2050.

Wang, F., Landau, D. P., 2001b. Determining the density of states for classical statistical models: A random walk algorithm to produce a flat histogram. Physical Review E 64, 056101.