Journal of Physics: Conference Series

PAPER • OPEN ACCESS

Population annealing: Massively parallel simulations in statistical physics

To cite this article: Martin Weigel *et al* 2017 *J. Phys.: Conf. Ser.* **921** 012017

View the article online for updates and enhancements.

Related content

- Comparison of serial and parallel simulations of a corridor fire using FDS
L Valasek
- Statistical Physics and Thermodynamics
M C Jones
- Parallel simulation of dam-break flow by OpenMP-based SPH method
Zhao Luo, Qihe Wu and Lei Zhang

This content was downloaded from IP address 77.236.32.176 on 21/11/2017 at 13:12

IOP Conf. Series: Journal of Physics: Conf. Series **921** (2017) 012017 doi:10.1088/1742-6596/921/1/012017

Population annealing: Massively parallel simulations in statistical physics

Martin Weigel[1], Lev Yu. Barash[2,3], Michal Borovský[4], Wolfhard Janke[5], and Lev N. Shchur[2,3,6]

[1] Applied Mathematics Research Centre, Coventry University, Coventry, CV1 5FB, United Kingdom
[2] Landau Institute for Theoretical Physics, 142432 Chernogolovka, Russia
[3] Science Center in Chernogolovka, 142432 Chernogolovka, Russia
[4] P.J. Šafárik University, Park Angelinum 9, 040 01 Košice, Slovak Republic
[5] Institut für Theoretische Physik, Universität Leipzig, Postfach 100920, 04009 Leipzig, Germany
[6] National Research University Higher School of Economics, 101000 Moscow, Russia

E-mail: martin.weigel@complexity-coventry.org

Abstract.
The canonical technique for Monte Carlo simulations in statistical physics is importance sampling via a suitably constructed Markov chain. While such approaches are quite successful, they are not particularly well suited for parallelization as the chain dynamics is sequential, and if replicated chains are used to increase statistics each of them relaxes into equilibrium with an intrinsic time constant that cannot be reduced by parallel work. Population annealing is a sequential Monte Carlo method that simulates an ensemble of system replica under a cooling protocol. The population element makes it naturally well suited for massively parallel simulations, and bias can be systematically reduced by increasing the population size. We present an implementation of population annealing on graphics processing units and discuss its behavior for different systems undergoing continuous and first-order phase transitions.

1. Introduction

Population annealing (PA) was first suggested in 2001 by Iba [1] and later on discussed in more detail by Hukushima and Iba [2] as a method to tackle potentially difficult sampling problems, but with no particular view to a parallel implementation. It belongs to a family of population methods with examples in quantum Monte Carlo ("diffusion Monte Carlo" [3]), statistical inference ("particle filter" [4]) and statistical physics ("go with the winners" [5]), among others. PA considers an ensemble of configurations set up in equilibrium at high temperature. These are then propagated down to low temperatures by a combination of local updates and a resampling or population control step that clones or prunes configurations according to their relative statistical weight. PA is hence not a Markov chain method but belongs to the different realm of sequential Monte Carlo techniques [6] that are quite commonly used in statistics, but less well known in physics.

It is mostly due to limitations in power dissipation that the continuous increase in clock frequencies of commodity processors has come to an end more than 10 years ago. The computational power of a typical PC or cluster node has continued to increase, however, but

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd

now fueled by a multiplication of the cores available for computation, i.e., through an increase in parallelism [7]. As a consequence, high-performance computing clusters can now come with millions of cores, and an end to this development towards ever more parallel computational resources is not in sight. The search for algorithms that perform well in such environments is hence one of the main tasks of computational science today [8]. In Monte Carlo, Markov chain methods that are so successful in serial and moderately parallel setups are not a perfect fit for massively parallel computational resources: parallelization there is either through domain decomposition with the obvious limitations in strong scaling as the domains become too small, or through the simulation of independent or loosely coupled parallel walkers to increase statistics [9–11]. In the latter approach the fraction of time spent outside of the equilibration phase shrinks with the number of parallel threads, leading to asymptotically vanishing efficiency. In PA, on the other hand, the quality of approximation improves with the population size R, with systematic deviations (bias) asymptotically decaying proportional to $1/R$ [12, 13]. As a consequence, this approach and its variations hold great promise for becoming standard simulation techniques for the age of massively parallel computing.

The study of systems undergoing phase transitions and, in particular, those with complex free-energy landscapes has been one of the main application areas and one of the main driving forces of algorithm development within the realm of computer simulations in statistical physics [14]. The effect of critical slowing down observed in the vicinity of continuous phase transitions led to the development of cluster updates [15]. Phase coexistence and the suppression of the region of mixed states connecting the pure phases for first-order transitions became amenable to quantitative studies through the multicanonical method and related approaches [16, 17]. On the other hand, frustrated systems with many metastable states can be simulated (more) efficiently through the use of parallel tempering [18] and similar techniques. This latter case of systems with complex free-energy landscapes is where PA has been recently used with some success, namely for the simulation of spin glasses [13, 19, 20]. A more general understanding of the performance of PA for simulations of systems undergoing continuous and discontinuous phase transitions is lacking to date.

In the following we show how the highly parallel resources provided by graphics processing units (GPUs) can be used to achieve an efficient implementation of PA for the case of the Potts model. We then study the behavior of the algorithm for simulations of the square-lattice Potts model, distinguishing the regimes where it undergoes second-order and where it undergoes first-order phase transitions, respectively.

2. Population annealing

Population annealing as a weighted sequential algorithm was first introduced in Refs. [1, 2]. The variant which we discuss here follows the scheme proposed more recently by Machta [21]. The algorithm is defined by the following steps:

(i) Set up an equilibrium ensemble of $R_0 = R$ independent replicas of the system at inverse temperature β_0. Often one chooses $\beta_0 = 0$, where this can be easily achieved.

(ii) To create an approximately equilibrated population at $\beta_i > \beta_{i-1}$, resample configurations $j = 1, \ldots, R_{i-1}$ with their relative Boltzmann weight $\tau_i(E_j) = \exp[-(\beta_i - \beta_{i-1})E_j]/Q_i$, where

$$Q_i \equiv Q(\beta_{i-1}, \beta_i) = \frac{1}{R_{i-1}} \sum_{j=1}^{R_{i-1}} \exp[-(\beta_i - \beta_{i-1})E_j]. \tag{1}$$

(iii) Update each replica by θ sweeps of a Markov chain Monte Carlo (MCMC) algorithm at inverse temperature β_i.

(iv) Calculate estimates for observable quantities $\mathcal{O}$ as population averages $\sum_{j=1}^{R_i} \mathcal{O}_j/R_i$.

(v) If the the target temperature β_{f} has not been reached, goto step (ii).

If one starts with an equilibrium sample, for instance by using a set of random configurations at infinite temperature, one arrives at a valid algorithm even without the additional rounds of sampling in step (iii). As in each temperature step some of the properly normalized resampling factors will be smaller than unity, however, this would lead to an exponential decrease in the number of distinct replicas and hence a vanishing efficiency in representing the target distribution. A combination with MCMC updates is therefore crucial for the approach, which hence becomes a hybrid of sequential and Markov chain methods.

The resampling step is very similar to what happens in histogram reweighting [22], i.e., taking account of the non-unit weight of an equilibrium configuration at inverse temperature β for the ensemble at temperature β'. As these weights are immediately translated into a resampling of the population, in the present implementation individual configurations all carry the same weight [2]. The expected number of copies of each replica at β_{i-1} to be included in the population at β_i should be $\hat{\tau}_i(E_j) = (R/R_{i-1})\tau_i(E_j)$. Other than fixing the expectation values, there is some freedom of choice for the actual probability distribution of the number r_i^j of copies of replica j in the population at inverse temperature β_i. The two most common cases are a Poisson distribution [21],

$$r_i^j \sim \mathrm{Pois}[\hat{\tau}_i(E_j)], \tag{2}$$

or a fixed integer part plus a Bernoulli trial corresponding to the fractional contribution [23, 24],

$$r_i^j = \begin{cases} \lfloor \hat{\tau}_i(E_j) \rfloor & \text{if } \; r > \hat{\tau}_i(E_j) - \lfloor \hat{\tau}_i(E_j) \rfloor \\ \lfloor \hat{\tau}_i(E_j) \rfloor + 1 & \text{otherwise} \end{cases} . \tag{3}$$

Here, r is a random number uniformly distributed in $[0, 1)$ drawn for each replica of the population at β_{i-1}, and $\lfloor x \rfloor$ denotes the largest integer not exceeding x.

The temperature protocol is not formally restricted in the algorithm as outlined above. The simplest choice corresponds to a constant inverse temperature step $\Delta\beta$, $\beta_i = \beta_{i-1} + \Delta\beta$. More advanced schemes are possible, however, such as an adaptive choice of temperature steps guaranteeing sufficient overlap of neighboring energy histograms [25]. A particular feature of the PA algorithm is that it provides a natural estimator of the free energy through a combination of the resampling factors Q_k [21],

$$-\beta_i F(\beta_i) = \ln Z_{\beta_0} + \sum_{k=1}^{i} \ln Q_k. \tag{4}$$

Here Z_{β_0} denotes the partition function at the initial temperature point β_0 which is required in order to get absolute free energies. For $\beta_0 = 0$ it is easily found as it corresponds to the case of the non-interacting system.

3. Implementation on GPU

As a sufficiently versatile example for the study of phase transitions of first and second order, we considered the Potts model on the square lattice with Hamiltonian [26]

$$\mathcal{H} = -J \sum_{\langle i,j \rangle} \delta_{s_i, s_j}, \tag{5}$$

where $s_i \in \{1, \ldots, q\}$ and $J > 0$ corresponds to a ferromagnetic coupling (we choose units such that $J = 1$). As indicated in the sum in Eq. (5) we restrict interactions to nearest neighbors only, and we assume periodic boundary conditions. The model has a phase transition at the

Table 1. Optimal performance of the CPU and GPU codes for PA for the $q = 2$ model in units of the time t_{SF} per spin flip. As the best performance is found for large θ, we quote here data for $\theta = 500$ and use $R = 50\,000$. The speedups for the single-spin coding (SSC) and multi-spin coding (MSC) GPU programs are relative to the CPU results. GPU performance data are for the Tesla K80 device. The CPU code was benchmarked on an Intel Xeon E5-2683 v4 CPU running at 2.1 GHz.

	CPU	GPU			
		SSC		MSC	
L	t_{SF} [ns]	t_{SF} [ns]	speedup	t_{SF} [ns]	speedup
16	23.1	0.094	246	0.0096	2406
32	22.9	0.092	249	0.0095	2410
64	22.6	0.092	246	0.0098	2306
128	22.6	0.097	233	0.0098	2306
256	22.5	0.098	230	0.0099	2273

self-dual point $\beta_t = \ln(1 + \sqrt{q})$ [27, 28] which is of second order for $q \leq 4$ and of first order for $q > 4$ [29, 30]. The Ising model is equivalent to the special case $q = 2$ as becomes clear on making the identification $\sigma_i = 2s_i - 3$ between Ising spins σ_i and the Potts spins s_i.

The PA algorithm outlined above consists of the three repeating steps (ii) population resampling, (iii) MCMC updates, and (iv) measurement cycle. In typical applications, one mostly chooses $\theta \gtrsim 10$, and as a consequence the MCMC updates typically take more than 80% of the time. Hence their efficient implementation is the most significant element [31]. We chose the Metropolis algorithm to update the spins [14]. This part of the simulation is trivially parallel due to the need to update all members of the population, and it is hence ideally suited for the massively parallel hardware provided by GPUs. It is important to take into account that GPUs have properties of vector machines as well as features of parallel computers [32]. Certain groups of threads are scheduled for execution on the available multiprocessors simultaneously and operate in lockstep. Memory transactions for such thread groups are served collectively, and it is hence crucial to lay out data in memory in a way that ensures that logically neighboring threads access consecutive locations in GPU memory. It is possible to run the algorithm with a setup with only one thread per replica which is particularly simple and easily generalized to different systems. To achieve best performance, however, we used a checkerboard decomposition [33–35] of the lattice and hence several threads to update parts of each replica in parallel, for details see Ref. [31]. To achieve good memory *coalescence* of accesses for this setup, it is crucial to store spin configurations in a *spin-parallel* way, i.e., such that neighboring spins in memory correspond to neighboring lattice sites. Additionally, we store the two sub-lattices separately which improves the performance further. The total amount of data transferred over the GPU bus is kept at a minimum by using 8-bit integers for the spin variables and employing the stateless, counter-based random-number generator (RNG) Philox [36, 37]. To achieve decent performance on current devices it is vital to use a number of threads that is several times larger than the available number of actual compute units, as this allows the scheduler to pause thread groups waiting for data and activate other groups that have already completed memory accesses instead. The resulting benefit of *latency hiding* is fully taken advantage of in the present setup as typical population sizes R are at least 10^4 while the number of cores of present cards is of the order of 10^3.

An implementation of the resampling process on GPU requires several steps. The normalization constants Q_i of Eq. (1) are calculated using a parallel or "butterfly" reduction

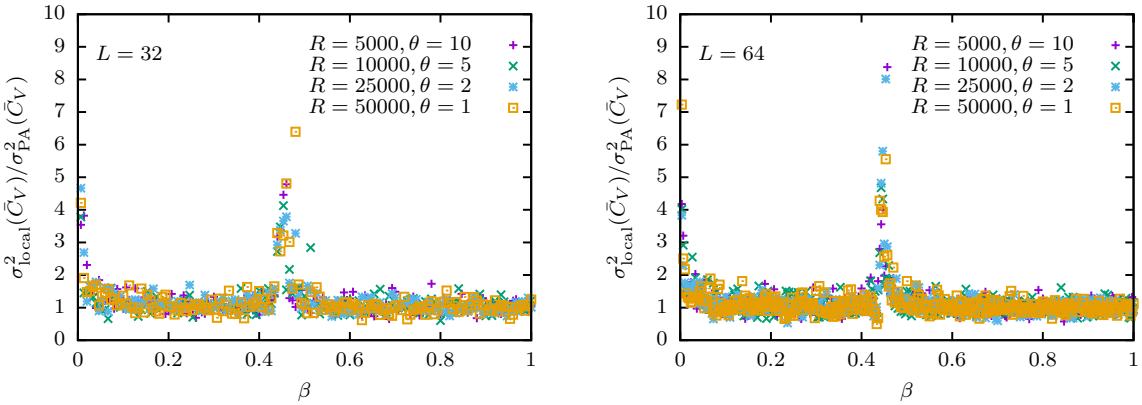

Figure 1. Ratio of variances of the specific heat of the 2D Ising model as estimated from simple Metropolis simulations of R samples and θ sweeps and of PA simulations with the same parameters. The inverse temperature steps were $\Delta\beta = 0.0067$ for $L = 32$ and $\Delta\beta = 0.0033$ for $L = 64$. The protocols for both types of simulations are identical apart from the fact that for the Metropolis runs the resampling step was turned off.

method [31]. The resampling factors $\hat{\tau}_i(E_j)$ are then computed in a separate kernel according to the prescription in Eq. (3). The resampling itself is done in parallel on GPU too and leads to a memory requirement that is twice that of the actual population size. (Note that the population size R_i weakly fluctuates between temperature steps.) Finally, measurements are again performed in a *replica-parallel* way for elementary quantities such as the energy and magnetization and their respective moments, using parallel reductions of the values for different population members to accumulate the population averages.

The resulting GPU code shows excellent performance as is illustrated by the peak performance data for the Ising case $q = 2$ collected for a Tesla K80 GPU and summarized in Table 1. The speedup of more than 200 times against a serial CPU code is almost independent of system size. For this specific case of only two states, an additional compression of spin states into the machine words is possible, leading to a method usually called *multi-spin coding* (MSC) [38] that yields additional performance increases against the standard *single-spin coded* (SSC) program. In the context of PA, this is most efficiently done by storing spins from the same lattice sites but different replicas in one word (sometimes called asynchronous multi-spin coding [35]). A performance bottle-neck in this setup is the generation of random numbers for the Metropolis updates. We use a linear-congruential generator seeded by the high-quality Philox RNG and achieve an about 10-fold speedup for MSC with 32 spins per word against the SSC implementation (see details in Ref. [31]). The corresponding spin-update times and speedups compared to the serial CPU code are summarized in Table 1.

4. Behavior at continuous transitions

The question of how well PA performs compared to other approaches, and in particular MCMC algorithms, can be answered quantitatively in terms of the dependence of the systematic errors (bias) and the statistical errors on the simulation parameters as well as on properties of the system such as the system size. We have discussed elsewhere the dependence of bias and variances on R, θ and $\Delta\beta$ for the example of the Ising model corresponding to the Hamiltonian (5) for $q = 2$ [25]. Here we focus on the computational efficiency in the vicinity of the critical point.

Figure 1 shows the ratio of estimated variances (squared error bars) for the specific heat of the 2D Ising model from two different algorithms: a Metropolis update and the PA method. Note

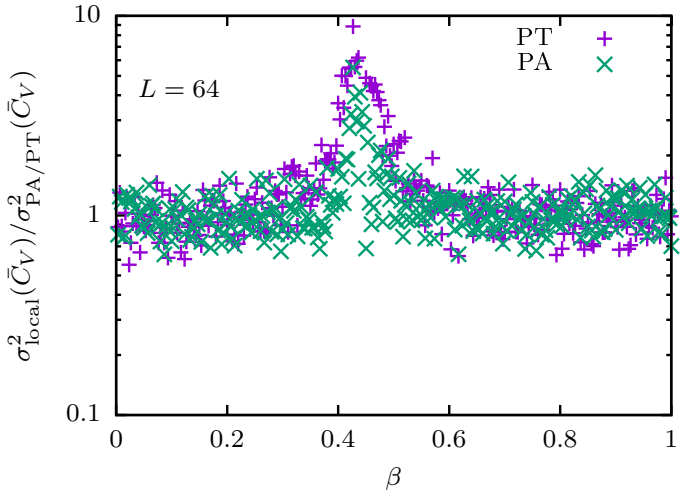

Figure 2. Ratio of variances of the specific heat for PA and parallel tempering (PT) simulations with the same number of MC steps as a function of inverse temperature β. In contrast to the data in Fig. 1, spin flips were performed according to a heatbath update rule here [14]. For PA we chose $R = 50\,000$ and $\theta = 1$ and used the same number of updates for the PT run.

that the Ising model at inverse temperature β corresponds to the $q = 2$ Potts model at 2β. Given that statistical errors in both methods decay with the inverse square root of the computational effort [14, 25] and that the run times (on CPU) of both setups were identical apart from a small overhead for the resampling steps, this ratio indicates an algorithmic speedup from using PA over a standard spin-flin only simulation, i.e., it shows how much longer the Metropolis simulation would need to run to produce error bars of the same size as those of the PA approach. To make the comparison as fair as possible, the type of Metropolis simulation used here is very similar to the setup in PA, i.e., a population of R replicas is simulated in parallel starting from $\beta_0 = 0$ and decreasing the temperature in steps of $\Delta\beta$, performing θ Metropolis sweeps (and measurements) at each temperature. The Metropolis setup is hence identical to the PA algorithm, but only with the resampling step turned off. As Fig. 1 reveals, the resampling is irrelevant sufficiently far away from the critical point, resulting in a unit variance ratio. In this regime, the MCMC steps are sufficient to decorrelate the configurations at each temperature step and hence resampling cannot decrease statistical errors further. In contrast, in the vicinity of the phase transition at $\beta_c \approx 0.44068$ critical slowing down leads to an imperfect decorrelation of configurations through the MCMC moves, and in this case the resampling results in an additional relaxation of configurations. This yields reduced statistical errors for the PA runs there and consequently a speedup factor that is larger than one. The two panels in Fig. 1 for system sizes $L = 32$ and $L = 64$ show such a speedup, which appears to increase with system size.

Population annealing is similar in spirit to parallel tempering [12] and one might hence wonder how they compare to each other for simulating a system with a continuous transition. Figure 2 shows a comparison of the statistical speedup factors defined through the ratio of variances against the spin-flip only run for the PA simulation (corresponding to the data from the right panel of Fig. 1 for $R = 50\,000$) and a parallel tempering (PT) simulation using the same temperature sequence and the same number of spin updates. It is clear that both PT and PA show comparable speedup factors for this problem, an observation that is in line with experiences previously reported for spin-glass systems [13, 24]. At the same time, however, it is clear that PA is much better suited for highly parallel computational environments than PT.

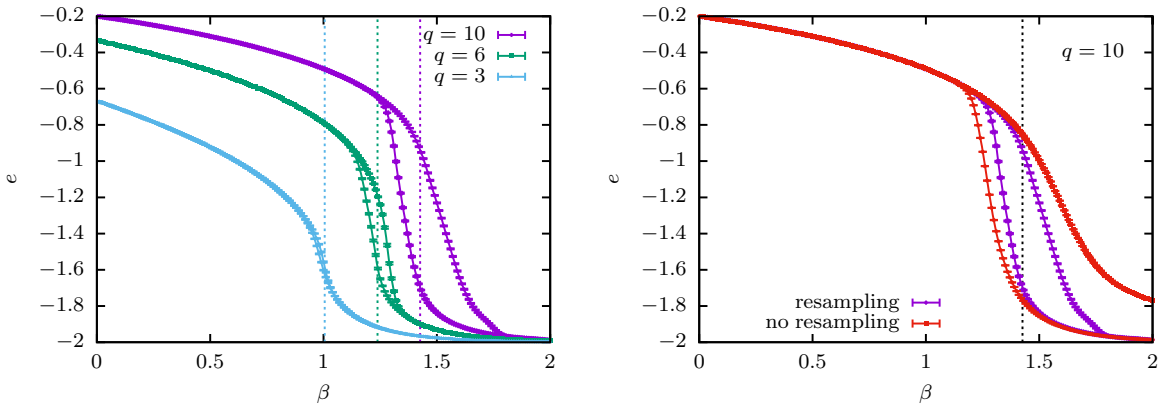

Figure 3. Hysteresis effect in PA runs for the q-states Potts model. Left: Energy as measured in cooling runs (right curve of each color) and heating runs (left curve of each color) for $q = 3$, $q = 6$ and $q = 10$ for $L = 32$ and $R = 10\,000$ with $\theta = 10$ and $\Delta\beta = 0.01$. The vertical lines indicate the asymptotic transition points at $\beta_t = \ln(1 + \sqrt{q})$. Right: The data points for $q = 10$ compared to the data for equivalent PA runs with the resampling step turned off.

5. Behavior at first-order transitions

At first-order points it is not the critical slowing down seen for continuous transitions that impedes the efficient simulation but rather the phenomenon of phase coexistence and the associated metastability [39]. As a consequence, the system remains in its original phase as the transition point is crossed and decays to the now stable opposite phase on a time scale that depends on the cooling or heating rate. This is the well-known effect of hysteresis. An established way of preventing this effect to occur is the simulation in generalized ensembles that artificially enhance the suppressed states connecting both phases, for example through the multicanonical method [16, 40]. It is interesting to see how the ensemble of configurations in PA behaves in this respect [41].

To clearly reveal any potential hysteresis effect, we added to the standard PA protocol that cools the population from its initial temperature point at β_0 down to β_f a heating run, starting from β_f and ending at β_0. The algorithm described in Sec. 2 works in the same way for such a temperature schedule, but now one needs to prepare the population in equilibrium at the inverse temperature β_f instead of at $\beta_0 = 0$. This is achieved by preparing the population as a uniformly random sample of the q-fold degenerate ground state of the Hamiltonian (5), corresponding to an equilibrium sample at $T = 0$, i.e., at $\beta_f = \infty$. In practice, it is also an almost perfect approximation of the equilibrium distribution at the finite $\beta_f = 3$ used here.

The left panel of Fig. 3 shows the internal energies $e = \langle \mathcal{H} \rangle / N$ estimated from PA runs with a cooling and runs with a heating schedule, respectively, for various values of q. It is clearly visible that hysteresis occurs for the first-order cases $q = 6$ and $q = 10$ while it is absent for the second-order model $q = 3$. This hysteresis effect is found to be more pronounced for large (inverse) temperature steps $\Delta\beta$. Thus, PA does not remove the hysteresis observed in a purely local Metropolis simulation. Nevertheless, it is able to reduce this effect as is apparent from a comparison of the energy curves for the regular PA simulations to PA runs with the resampling step turned off, corresponding to pure Metropolis simulations. This is shown in the plot in the right panel of Fig. 3.

Such hysteresis is also observed in other quantities, for example in estimates of the free energy deduced from the expression of Eq. (4). This is illustrated in Fig. 4 for the same values of q as shown in Fig. 3 and the additional case of the Ising model. Recall that in order to get absolute

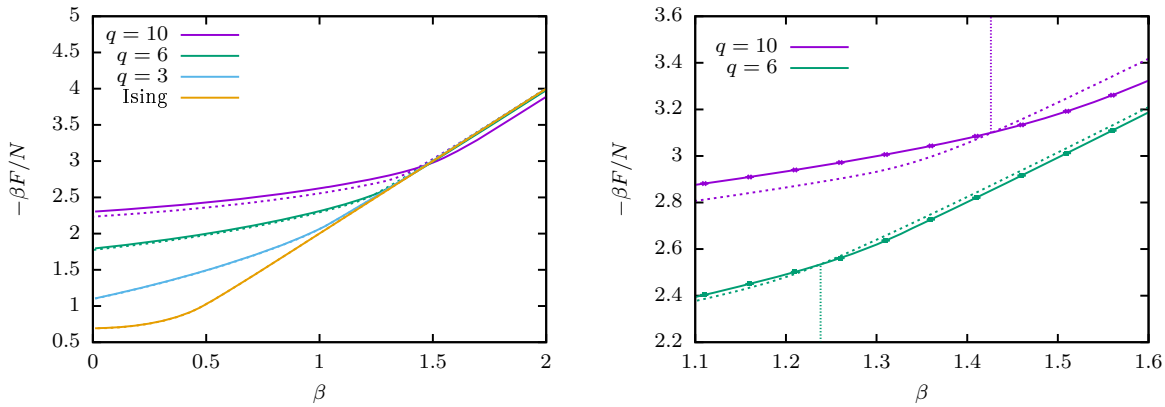

Figure 4. Left: Free energy of the Ising model and the $q = 3$, $q = 6$ and $q = 10$ Potts models as estimated from PA runs with cooling (solid lines) and with heating (dashed lines) schedules. The system size studied was $L = 32$ and the PA parameters were chosen to be $R = 10\,000$, $\theta = 10$ and $\Delta\beta = 0.01$. Right: Detail of the crossing of the metastable free energies for $q = 6$ and $q = 10$. Symbols with error bars are only shown for every fifth actual data point. The vertical dotted lines indicate the locations of the (exact) asymptotic transition points.

free energies, the expression (4) requires additional input in the form of the reference value Z_{β_0}. For the standard cooling schedule the relevant reference point is $-\beta F(\beta)/N = \ln q$ as $\beta \to 0$. For the heating runs, on the other hand, it is easy to see that $-\beta F(\beta)/N = (\ln q)/N - \beta e_0$ as $\beta \to \infty$, where $e_0 = E_0/N = -2$ is the ground-state energy per site [41]. The resulting free-energy estimates from cooling and from heating runs are perfectly consistent with each other for the second-order cases $q = 2$ (Ising) and $q = 3$, cf. the left panel of Fig. 4. For the first-order models $q = 6$ and $q = 10$, on the other hand, the low-temperature and high-temperature estimates differ, as expected [40], and both curves intersect close to the transition point, thus providing a method for determining from the simulations the location of the phase transition [40, 41].

6. Summary

We have studied population annealing as a promising new technique for the simulation of systems undergoing phase transitions. Being a hybrid of sequential and Markov chain Monte Carlo methods, it has the advantage of a theoretically perfect strong scaling behavior. While on increasing the number of parallel workers pure Markov chain methods eventually spend most time in the equilibration phase, bias as well as statistical errors in population annealing are systematically reduced with increasing numbers of parallel threads. Taking advantage of these excellent scaling properties we presented an efficient GPU implementation of population annealing for the Potts model. For the Ising case $q = 2$ we find a more than 200-fold speedup against a serial CPU code even for small system sizes. This can be further increased by using multi-spin coding techniques which yield an additional factor of 10 for the Ising model. Population annealing is found to offer no algorithmic advantage over the underlying Markov chain technique in terms of a possible reduction of statistical errors from the same computational effort for the Ising model off criticality. This is where the regular spin flips are sufficient to decorrelate configurations. In the vicinity of the critical point, on the other hand, an algorithmic speedup is achieved that potentially grows with system size. For systems with first-order phase transitions, population annealing is able to reduce, but not to eliminate the effect of hysteresis caused by metastability close to the transition. The free-energy estimate naturally deduced

from population annealing runs can be used to estimate the location of the transition point. A combination of population annealing with generalized-ensemble methods that might be able to more completely remove the effects of metastability would be an interesting topic for future research.

Acknowledgments
The work of L.B., M.B. and L.S. is supported by grant No. 14-21-00158 of the Russian Science Foundation. M.B. was also supported by the Scientific Grant Agency of the Ministry of Education of the Slovak Republic (Grant No. 1/0331/15). The authors acknowledge support from the European Commission through the IRSES network DIONICOS under Contract No. PIRSES-GA-2013-612707.

References

[1] Iba Y 2001 *Trans. Jpn. Soc. Artif. Intell.* **16** 279–286
[2] Hukushima K and Iba Y 2003 *AIP Conf. Proc.* **690** 200–206
[3] Ceperley D M and Kalos M H 1986 Quantum many-body problems *Monte Carlo Methods in Statistical Physics* ed Binder K (Berlin: Springer) pp 145–194
[4] Kitagawa G 1996 *J. Comp. Graph. Stat.* **5** 1–25
[5] Grassberger P 2002 *Comput. Phys. Commun.* **147** 64–70
[6] Doucet A, de Freitas N and Gordon N (eds) 2001 *Sequential Monte Carlo Methods in Practice* (New York: Springer)
[7] Asanovic K, Bodik R, Catanzaro B C, Gebis J J, Husbands P, Keutzer K, Patterson D A, Plishker W L, Shalf J, Williams S W and Yelick K A 2006 The landscape of parallel computing research: A view from Berkeley *Tech. Rep. University of California, Berkeley, EECS Department, UCB/EECS-2006-183*
[8] Owens J D, Houston M, Luebke D, Green S, Stone J E and Phillips J C 2008 *Proceedings of the IEEE* **96** 879–899
[9] Zierenberg J, Marenz M and Janke W 2013 *Comput. Phys. Commun.* **184** 1155–1160
[10] Vogel T, Li Y W, Wüst T and Landau D P 2013 *Phys. Rev. Lett.* **110** 210603
[11] Gross J, Zierenberg J, Weigel M and Janke W 2017 Parallel multicanonical simulations on GPUs *Preprint* in preparation
[12] Machta J and Ellis R S 2011 *J. Stat. Phys.* **144** 541–553
[13] Wang W, Machta J and Katzgraber H G 2015 *Phys. Rev.* E **92** 063307
[14] Landau D P and Binder K 2015 *A Guide to Monte Carlo Simulations in Statistical Physics* 4th ed (Cambridge: Cambridge University Press)
[15] Swendsen R H and Wang J S 1987 *Phys. Rev. Lett.* **58** 86–88
[16] Berg B A and Neuhaus T 1992 *Phys. Rev. Lett.* **68** 9–12
[17] Wang F and Landau D P 2001 *Phys. Rev. Lett.* **86** 2050–2053
[18] Hukushima K and Nemoto K 1996 *J. Phys. Soc. Jpn.* **65** 1604–1608
[19] Wang W, Machta J and Katzgraber H G 2014 *Phys. Rev.* B **90** 184412
[20] Wang W, Machta J and Katzgraber H G 2015 *Phys. Rev.* B **92** 094410
[21] Machta J 2010 *Phys. Rev.* E **82** 026704
[22] Ferrenberg A M and Swendsen R H 1988 *Phys. Rev. Lett.* **61** 2635–2638
[23] Garel T and Orland H 1990 *J. Phys.* A **23** L621–L626
[24] Wang W, Machta J and Katzgraber H G 2015 *Phys. Rev.* E **92** 013303

[25] Weigel M, Barash L Y, Borovský M, Shchur L N and Janke W 2017 Population annealing for the Ising model *Preprint* in preparation

[26] Wu F Y 1982 *Rev. Mod. Phys.* **54** 235–268

[27] Baxter R J 1973 *J. Phys.* C **6** L445–L448

[28] Beffara V and Duminil-Copin H 2012 *Probab. Theory Rel.* **153** 511–542

[29] Duminil-Copin H, Sidoravicius V and Tassion V 2017 *Commun. Math. Phys.* **349** 47–107

[30] Duminil-Copin H, Gagnebin M, Harel M, Manolescu I and Tassion V 2016 Discontinuity of the phase transition for the planar random-cluster and Potts models with $q > 4$ *Preprint* arXiv:1611.09877

[31] Barash L Y, Weigel M, Borovský M, Janke W and Shchur L N 2017 GPU accelerated population annealing algorithm *Preprint* arXiv:1703.03676 *Comput. Phys. Commun.* in print (http://dx.doi.org/10.1016/j.cpc.2017.06.020)

[32] Kirk D B and Hwu W W 2010 *Programming Massively Parallel Processors* (Amsterdam: Elsevier)

[33] Weigel M 2012 *J. Comp. Phys.* **231** 3064–3082

[34] Weigel M 2011 *Comput. Phys. Commun.* **182** 1833–1836

[35] Lulli M, Bernaschi M and Parisi G 2015 *Comput. Phys. Commun.* **196** 290–303

[36] Salmon J K, Moraes M A, Dror R O and Shaw D E 2011 Parallel random numbers: As easy as 1, 2, 3 *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis* SC '11 (New York: ACM)

[37] Manssen M, Weigel M and Hartmann A K 2012 *Eur. Phys. J. Special Topics* **210** 53–71

[38] Zorn R, Herrmann H J and Rebbi C 1981 *Comput. Phys. Commun.* **23** 337–342

[39] Binder K 1987 *Rep. Prog. Phys.* **50** 783–859

[40] Janke W 2003 First-order phase transitions *Computer Simulations of Surfaces and Interfaces* (*NATO Science Series, II. Mathematics, Physics and Chemistry* vol 114) ed Dünweg B, Landau D P and Milchev A I (Dordrecht: Kluwer Academic Publishers) pp 111–135

[41] Barash L Y, Weigel M, Shchur L N and Janke W 2017 *Eur. Phys. J. Special Topics* **226** 595–604