TheoretiCS

Approximate counting for spin systems in sub-quadratic time

Received May 31, 2024
Revised Nov 4, 2024
Accepted Nov 17, 2024
Published Jan 13, 2025

Key words and phrases
Randomised algorithm, Approximate counting, Spin system, Sub-quadratic algorithm

Konrad Anand [a,b]
Weiming Feng [c]
Graham Freifeld [b]
Heng Guo [b]
Jiaheng Wang [d]

a School of Mathematical Sciences, Queen Mary University of London, United Kingdom

b School of Informatics, University of Edinburgh, United Kingdom

c Institute for Theoretical Studies, ETH Zürich, Zürich, Switzerland

d Faculty of Informatics and Data Science, University of Regensburg, Germany

ABSTRACT. We present two randomised approximate counting algorithms with running time $\widetilde{O}\left(\left(\frac{n}{\varepsilon}\right)^{2-c}\right)$, for some constant $c > 0$ and accuracy ε:

1. for the hard-core model with fugacity λ on graphs with maximum degree Δ when $\lambda = O(\Delta^{-1.5-c_1})$ where $c_1 = c/(2-2c)$;
2. for spin systems with strong spatial mixing (SSM) on planar graphs with quadratic growth, such as $\mathbb{Z}^2$.

For the hard-core model, Weitz's algorithm (STOC, 2006) achieves sub-quadratic running time when correlation decays faster than the neighbourhood growth, namely when $\lambda = o(\Delta^{-2})$. Our first algorithm does not require this property and extends the range where sub-quadratic algorithms exist.

Our second algorithm appears to be the first to achieve sub-quadratic running time up to the SSM threshold, albeit on a restricted family of graphs. It also extends to (not necessarily planar) graphs with polynomial growth, such as $\mathbb{Z}^d$, but with a running time of the form $\widetilde{O}\left(\left(\frac{n}{\varepsilon}\right)^2 / 2^{c(\log \frac{n}{\varepsilon})^{1/d}}\right)$ where d is the exponent of the polynomial growth and $c > 0$ is some constant.

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 947778). Weiming Feng acknowledges the support from Dr. Max Rössler, the Walter Haefner Foundation and the ETH Zürich Foundation. A preliminary version of this article appeared at ICALP 2024 [1].

Cite as Konrad Anand, Weiming Feng, Graham Freifeld, Heng Guo, Jiaheng Wang. Approximate counting for spin systems in sub-quadratic time. TheoretiCS, Volume 4 (2025), Article 3, 1-27.

https://theoretics.episciences.org
DOI 10.46298/theoretics.25.3

1. Introduction

The study of counting complexity was initiated by Valiant [34] with the introduction of the complexity class #**P**. An intriguing phenomenon emerging in counting complexity is that many #**P**-complete problems admit fully polynomial-time randomised approximation schemes (FPRAS), which output an ε-approximation in time polynomial in n and $1/\varepsilon$ with n being the input size. This is most commonly found for the so-called partition function of spin systems, as demonstrated by the pioneering work of Jerrum and Sinclair [24, 25]. Spin systems are physics models for nearest neighbour interactions, and the partition functions are the normalising factors for their Gibbs distributions. This quantity can express the count of combinatorial objects such as the number of matchings, independent sets, or colourings in a graph, and is much more expressible by allowing real parameters of the system.

In this paper we are most interested in the fine-grained aspects of the complexity of estimating partition functions. While for most spin systems, exact counting is #**P**-hard [7], there are parameter ranges where the model exhibits spatial mixing[1] that yields efficient approximation algorithms. Roughly speaking, spatial mixing gives us a way to bound the correlation or influence a partial configuration has on each vertex with respect to its distance to the vertex. Without spatial mixing, the partition function is usually **NP**-hard to approximate [32, 20, 19].

Efficient approximate counting was first enabled by the work of Jerrum, Valiant, and Vazirani [26] who gave self-reductions from approximate counting to sampling for a large class of problems. The sampling task is then most commonly solved via Markov chains. The efficiency of a Markov chain is measured by its mixing time (i.e. how long it takes to get close to the target distribution). For spin systems with spatial mixing, in many situations, the standard chain, namely the Glauber dynamics, mixes in $O(n \log n)$ time [11, 3, 9, 10].

Another later technique, simulated annealing, provides a more efficient counting to sampling reduction [33, 23, 28]. Together with the $O(n \log n)$ mixing time mentioned above, this leads to[2] $\widetilde{O}((n/\varepsilon)^2)$ approximate counting algorithms. These Markov chain Monte Carlo (MCMC) algorithms are the fastest for estimating partition functions in general, but $\Omega(n^2)$ appears to be a natural barrier to this approach. This is because generating a sample would take at least linear time (and there are $\Omega(n \log n)$ lower bounds for the mixing time of Markov chains [22] for many spin systems), and, restricted to the standard way of using the samples, the number of samples required for simulated annealing is at least $\Omega(n/\varepsilon^2)$ [28, Theorem 10].

On the other hand, when we relax the parameters, $\Omega(n^2)$ is no longer a barrier to algorithms. Let us take the hard-core gas model as an example. Here the Gibbs distribution μ is over the

1 There are multiple notions of spatial mixing with varying degrees of strength. In this paper we use strong spatial mixing, defined in Section 2, Definition 2.3.

2 The notation $\widetilde{O}(\cdot)$ hides logarithmic factors in n and sometimes other constants, such as ones depending on the maximum degree Δ of the input graph.

set $\mathcal{I}$ of independent sets of a graph G. For an independent set I, $\mu(I) := \lambda^{|I|}/Z(G)$, where λ is a parameter of the system (so-called fugacity), and $Z(G) := \sum_{I\in\mathcal{I}} \lambda^{|I|}$ is the partition function. For graphs with degree bound Δ, spatial mixing holds when $\lambda < \lambda_c(\Delta) := \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^{\Delta}} \approx \frac{e}{\Delta}$. The aforementioned MCMC results [11, 9, 10] imply FPRASes running in time $\widetilde{O}((n/\varepsilon)^2)$ as long as $\lambda < \lambda_c(\Delta)$. Yet much earlier, Weitz [35] gave the first fully polynomial-time approximation scheme (FPTAS, the deterministic counterpart to FPRAS) for the partition function of the hard-core model when $\lambda < \lambda_c(\Delta)$, which is not based on Markov chains. While Weitz's algorithm has a running time $n^{O(\log \Delta)}$ in general, it has an interesting feature that it gets faster as λ decreases. Roughly speaking, for $k > 0$ and $\lambda = O((1/\Delta)^{1+k})$, Weitz's FPTAS runs in time $O(n^{1+1/k}/\varepsilon^2)$. In particular, if $\lambda = o(\Delta^{-2})$, Weitz's algorithm passes the $\Omega(n^2)$ barrier, whereas the aforementioned MCMC method still takes $\Omega(n^2)$ time. This leads to an intriguing question:

When can we achieve sub-quadratic running time for approximate counting? (1)

In this paper we make some progress towards this question. For hard-core models, Weitz's algorithm uses the self-reduction [26] to reduce approximate counting to estimating marginal probabilities (probabilities of a partial configuration of the system). We provide a quadratic speedup for the marginal estimation step for λ well below $\lambda_c(\Delta)$, albeit with the introduction of randomness. The result is summarised as follows.

THEOREM 1.1. *Fix a constant $k > 0$. Let $\Delta \geq 2$ be an integer and $\lambda < \frac{1}{\Delta^k(\Delta-1)}$. For graphs with maximum degree Δ, there exists an FPRAS for the partition function of the hard-core model with parameter λ in time $\widetilde{O}\left(\left(\frac{n}{\varepsilon}\right)^{1+\frac{1}{2k}}\right)$, where n is the number of vertices and ε is the error margin.*

REMARK 1.2 (Decay rate vs. neighbourhood growth). For a constant ε, the running time of Theorem 1.1 is sub-quadratic if $\lambda = o(\Delta^{-1.5})$, and $\widetilde{O}(n^{1.5})$ if $\lambda = O(\Delta^{-2})$. In contrast, to achieve sub-quadratic running-time, Weitz's algorithm requires $\lambda = o(\Delta^{-2})$, which is also the threshold when correlation decays faster than the growth of the neighbourhood. This threshold has algorithmic significance in other contexts [18, 2], but Theorem 1.1 implies that it is not essential to achieve sub-quadratic approximate counting.

Figure 1 is a sketch comparing the running times of MCMC,[3] Weitz's algorithm, and Theorem 1.1.[4] For the limiting case of $k = 0$, our algorithm works when $\lambda < \frac{1}{\Delta-1}$ and still presents a quadratic speedup comparing to Weitz's algorithm. However in this case the running time is $\left(\frac{n}{\varepsilon}\right)^{O(\log \Delta)}$ and thus our speedup is hidden in the big-O notation and is less significant. The parameter constraint $\frac{1}{\Delta-1}$ is imposed by the running-time tail bound of a subroutine we used, namely the recursive marginal sampler of Anand and Jerrum [2].

3 The running time of MCMC usually also depends on the parameter λ, but changing λ does not change the exponent of n. The effect of λ is usually a small polynomial factor hidden in the $\widetilde{O}(\cdot)$ notation, and the sketch in Figure 1 ignores this effect.

4 Another notable FPTAS is via zeros of polynomials [4, 30]. It can achieve similar subquadratic running time when $\lambda = o(\Delta^{-2})$, but it is apparently no faster than Weitz's correlation decay algorithm.

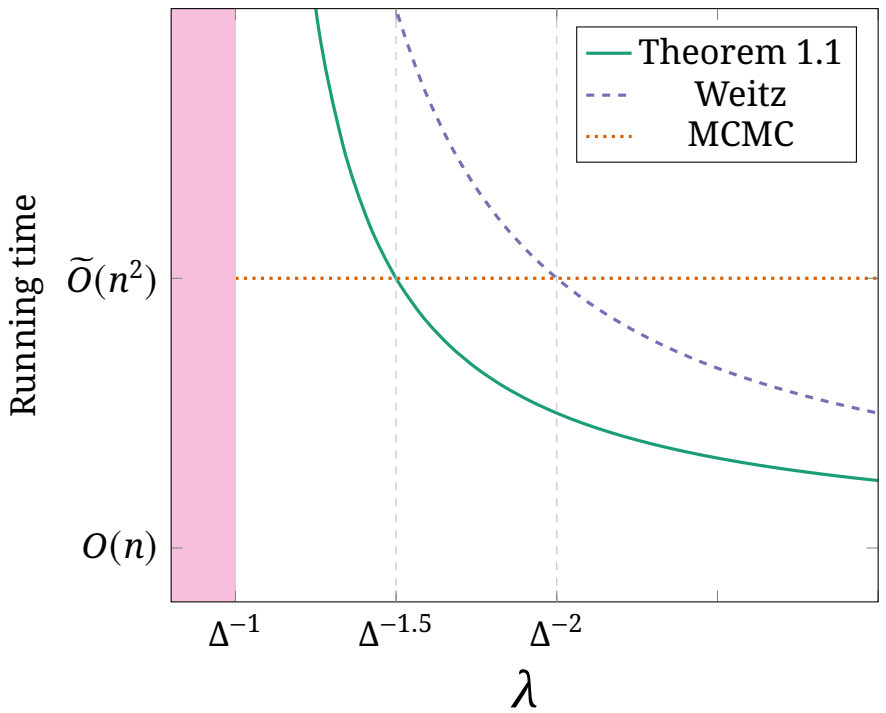


Figure 1. Running time comparison among MCMC, Weitz's algorithm, and Theorem 1.1

The key to our method is to find a new estimator of the marginal probability that simultaneously has low variance and can be evaluated very fast. Our technique combines Weitz's self-avoiding walk (SAW) tree construction and the marginal sampler of Anand and Jerrum [2]. The marginal of the root of the SAW tree preserves the desired marginal probability, and can be evaluated in time linear in the size of the tree via standard recursion. We use the unbiased marginal sampler to draw a random boundary condition at a suitable depth on the SAW tree, and compute the marginal of the root using recursion under this boundary condition. Both steps can be computed in time near-linear in the size of the sub-tree. In the self-reduction [26], it suffices to have $O(1/n)$ variance for each marginal estimation. Thus we only need absolute error $O(1/\sqrt{n})$ rather than the $O(1/n)$ error typically required by Weitz's algorithm. This larger error tolerance roughly halves the depth where we truncate the SAW tree comparing to Weitz's truncation. As a result, we obtain our quadratic improvement on the marginal estimation step over Weitz's algorithm. This method also extends to other anti-ferromagnetic 2-spin systems.

Our second contribution is about graphs with polynomial growth. In particular, for planar graphs with quadratic growth, we provide $\widetilde{O}((n/\varepsilon)^{2-c})$ algorithms for some constant $c > 0$. An informal statement is as follows. (The detailed statement is Theorem 4.2.)

THEOREM 1.3. *Let $\mathbb{G}$ be a family of planar graphs with quadratic growth. For a spin system exhibiting spatial mixing on $\mathbb{G}$, there exists an FPRAS for the partition function of $G \in \mathbb{G}$ with n vertices. The run-time is $\widetilde{O}((n/\varepsilon)^{2-c})$ for some constant $c > 0$, where ε is the error margin.*

We note that one of the most important graphs in statistical physics, the 2D integer grid $\mathbb{Z}^2$, indeed has quadratic growth. More generally, any planar graph with a bounded radius circle

packing has quadratic growth. Thus Theorem 1.3 covers many important families of planar graphs, including most lattices. (A *non*-example would be the Cayley tree.) Specialised to the hard-core model, Theorem 1.3 works up to the critical threshold, which is at least $\lambda_c(\Delta)$,[5] when the graph satisfies the condition in the theorem and has maximum degree Δ.

The key to Theorem 1.3 is once again a suitable estimator for marginal probabilities. We choose a distance ℓ boundary around a vertex v in G with a carefully chosen ℓ, and our estimator is the marginal under random boundary conditions. This boundary condition is yet again sampled using the algorithm of Anand and Jerrum [2]. Our main observation is that due to quadratic growth, the number of possible boundary conditions do not grow very fast. It turns out to be more efficient to create a look-up table by enumerating all boundary conditions first, and instead of computing the marginal for each sample, we simply find it in this table. Since planar graphs have linear local tree-width, the table can be created efficiently. This last step is inspired by the work of Yin and Zhang [36].

This method extends to any (not necessarily planar) graph families with polynomial growth. Without planarity, we use brute-force enumeration instead to create the table. This makes our gain on the running time smaller. Again an informal statement is as follows, with the full version in Theorem 4.11.

THEOREM 1.4. *Let $\mathbb{G}$ be a family of graphs with polynomial growth. For a spin system exhibiting spatial mixing on $\mathbb{G}$, there exists an FPRAS for the partition function of $G \in \mathbb{G}$ with n vertices. The run-time is $\widetilde{O}\left(\frac{n^2}{\varepsilon^2 2^{c(\log \frac{n}{\varepsilon})^{1/d}}}\right)$ where $c > 0$ is some constant, d is the exponent of the polynomial growth, and ε is the error margin.*

An example of such graphs would be the d-dimensional integer lattice $\mathbb{Z}^d$. Note that Theorem 1.3 is better than Theorem 1.4 for $d = 2$ but requires the extra assumption of planarity. The speedup factor $2^{c(\log n)^{1/d}}$ in Theorem 1.4 is slower than any polynomial in n but faster than any polynomial in $\log n$.

We note an interesting related work by Chu, Gao, Peng, Sachdeva, Sawlani, and Wang [12], who give an approximate counting algorithm with running time $\widetilde{O}(m^{1+o(1)} + n^{1.875+o(1)}/\varepsilon^{1.75})$ for spanning trees of graphs, where m is the number of edges and n is the number of vertices. Notice that the input size here is $O(m)$ and $m = \Omega(n)$. Thus their running time is also sub-quadratic. However, there are some key differences between this work and ours. Aside from not being a spin system, spanning trees can be counted exactly in polynomial time, thanks to Kirchhoff's matrix-tree theorem. This allows them to use various efficient exact counting subroutines, whereas the problems we consider are #**P**-hard in general and no such subroutine is likely to exist.

Another more recent related result is the sub-quadratic all-terminal unreliability estimation algorithm by Cen, He, Li, and Panigrahi [8], which runs in sub-quadratic time $m^{1+o(1)}\varepsilon^{-3} +$

5 For a given graph family, such as subgraphs of $\mathbb{Z}^2$, the critical threshold may be well above $\lambda_c(\Delta)$.

$\widetilde{O}(n^{1.5}\varepsilon^{-2})$. This problem, while #**P**-hard, is not a spin system either. Their method features a recursive Monte Carlo estimator that is very different from ours, and not applicable to spin systems.

A crucial ingredient of our algorithm is the recursive marginal sampler of Anand and Jerrum [2], which allows us to sample configurations on part of the graph while examining a low number of vertices in expectation. This type of local / marginal sampler enables partial access to a large random object (in this case, a random spin configuration on the whole graph) with substantially less information than traditional samplers. It has found applications in local computation algorithms [5], and in derandomising Markov chains [17]. Our results offer yet another application, namely to accelerate computation of the global partition function.

We hope that our results are just the first step towards answering Question (1). In particular, it is not clear whether an $O((n/\varepsilon)^{2-c})$ algorithm exists for the hard-core model when $\lambda = \Theta(1/\Delta)$ on graphs with maximum degree Δ, or if more efficient algorithms exist for graphs with polynomial or sub-exponential growth. We leave these questions as open problems.

2. Preliminaries

We are interested in spin systems which exhibit strong spatial mixing.

DEFINITION 2.1. A q state spin system (or q-spin system for short) is given by a graph $G = (V, E)$, a q-by-q interaction matrix A, and a field $b : [q] \to \mathbb{R}$. A configuration of G is an assignment of states to vertices, $\sigma : V \to [q]$. The weight of a configuration σ is determined by the assignments to the vertices and the interactions between them,

$$w(\sigma) := \prod_{(u,v)\in E} A_{\sigma(u),\sigma(v)} \prod_{v\in V} b_{\sigma(v)}.$$

The Gibbs distribution μ is one where the probability of each configuration is proportional to its weight, namely, $\mu(\sigma) := \frac{w(\sigma)}{Z(G)}$, where the partition function $Z(G) = \sum_\sigma w(\sigma)$ is a normalising factor.

In this paper, we consider the following permissive spin system, which says any locally feasible configuration can be extended to a globally feasible configuration.

DEFINITION 2.2. A q-spin system on $G = (V, E)$ is permissive if for any $\Lambda \subseteq V$, any $\sigma \in [q]^\Lambda$, if $b_{\sigma(v)} > 0$ for all $v \in \Lambda$ and $A_{\sigma(u),\sigma(v)} > 0$ for all $u, v \in \Lambda$ satisfying $(u, v) \in E$, then σ can be extended to a full configuration $\sigma' \in [q]^V$ such that $w(\sigma') > 0$.

Many natural spin systems are permissive. Examples include the hard-core model, the graph q-colouring with $q \geq \Delta + 1$, where Δ is the maximum degree of the graph, and all spin systems with soft constraints (e.g. the Ising model and the Potts model).

We call the problem of evaluating Z the counting problem for the q-spin system. The standard algorithmic aim here is a *fully-polynomial randomised approximation scheme* (FPRAS), where given the spin system and an accuracy $\varepsilon > 0$, the algorithm outputs $\widetilde{Z}$ such that $1 - \varepsilon \leq \frac{\widetilde{Z}}{Z} \leq 1 + \varepsilon$ with probability at least $3/4$, and runs in time polynomial in the size of the system and $1/\varepsilon$. To understand the requirement of an FPRAS, note that the probability $3/4$ can be boosted arbitrarily close to 1 via standard means. The accuracy can also be boosted by taking many disjoint copies of the system. In fact, any polynomial accuracy can be boosted to an arbitrarily small ε in polynomial-time.

Also note that if G is disconnected, then $Z(G) = \prod_i Z(G_i)$ where $G_i's$ are the connected components of G. Thus, we always consider connected graphs in the paper.

Similar to $\mu(\sigma)$ for the probability of a configuration, for an $S \subseteq V$ and a partial configuration σ_S on S, we use $\mu(\sigma_S)$ for the marginal probability of σ_S under μ. We denote the marginal distribution induced by μ on S by μ_S. When $S = \{v\}$, we also write μ_v. For the distribution conditioned on a partial configuration σ_S, we use μ^{σ_S} or $\mu_v^{\sigma_S}$.

Strong spatial mixing is a property of the spin system where a partial configuration of G does not significantly influence the assignment of a distant vertex.

DEFINITION 2.3 (SSM). A q-spin system is said to have strong spatial mixing with decay rate $f(\ell)$ for a graph $G = (V, E)$ if for any $v \in V, S \subset V$, and two configurations σ_S, τ_S,

$$d_{\mathrm{TV}}(\mu_v^{\sigma_S}, \mu_v^{\tau_S}) \leq f(\ell),$$

where d_{TV} denotes the total variation distance, $T \subseteq S$ is the subset where the configurations are different, and $\ell = \mathrm{dist}(v, T)$ is the minimum distance from v to a vertex in T.

We say a q-spin system has strong spatial mixing for a family of graphs $\mathbb{G}$ if there exists $f(\cdot)$ such that the system has strong spatial mixing with the same decay rate $f(\ell)$ for all $G \in \mathbb{G}$.

Strong spatial mixing is a very strong form of correlation decay. When $f(\ell) = \exp(-\Omega(\ell))$ we say we have strong spatial mixing with exponential decay.

2.1 Two-state spin systems

A spin system is symmetric if $A_{ij} = A_{ji}$ for all i, j. When $q = 2$ and the system is symmetric, we have states $\{0, 1\}$ and can normalise A and b so that the interaction between 0 and 1 and the contribution of 0 are 1, and $A = \begin{bmatrix} \beta & 1 \\ 1 & \gamma \end{bmatrix}$ and $b = (1, \lambda)$ for $\beta, \gamma \geq 0$, and $\lambda > 0$.

When $\beta = \gamma$ the system is an Ising model, and for $\beta = 1, \gamma = 0$ the system is a hard-core gas model. We call a system *anti-ferromagnetic* if disagreeing assignments of adjacent vertices are more heavily weighted, namely $\beta\gamma < 1$.

For a tree T rooted at v and a partial configuration σ_S we define the marginal ratio

$$R_T^{\sigma_S} := \frac{\mu_v^{\sigma_S}(1)}{\mu_v^{\sigma_S}(0)} = \frac{\mu_v^{\sigma_S}(1)}{1-\mu_v^{\sigma_S}(1)},$$

or $R_T^{\sigma_S} := \infty$ if $\mu_v^{\sigma_S}(1) = 1$. These ratios satisfy a well-known recurrence relation:

$$R_T^{\sigma_S} = \lambda \prod_{i=1}^{d} \frac{\gamma R_{T_i}^{\sigma_S} + 1}{R_{T_i}^{\sigma_S} + \beta}, \tag{2}$$

where T_i is the ith subtree of T. Similarly, for a graph G we can define $R_{G,v}^{\sigma_S} = \mu_v^{\sigma_S}(1)/(1-\mu_v^{\sigma_S}(1))$. While $R_{G,v}^{\sigma_S}$ does not admit a simple recursion, the self-avoiding walk (SAW) tree of G at v as constructed by Weitz [35] can be used to compute it.

THEOREM 2.4 (Theorem 3.1 of [35]). *For any $G = (V, E)$, a configuration σ_S on $S \subset V$, and any $v \in V$, there exists a tree $T_{\text{SAW}} = T_{\text{SAW}}(G, v)$ such that*

$$R_{G,v}^{\sigma_S} = R_{T_{\text{SAW}}}^{\sigma_S}.$$

The SAW tree is rooted at v. Each node corresponds to a self-avoiding walk starting from v. The length of the walk is the same as the distance between the node and the root v. When a walk is closed, the node is set to unoccupied or occupied according to if the penultimate vertex is before or after the starting vertex of the cycle in some pre-determined local ordering at the last vertex. For details, see [35].

The SAW tree can have depth up to n, so may be exponential in size. Marginals on the SAW tree are therefore difficult to compute, but using the recursion in Equation (2) we can approximate them by truncating the tree. This approximation is accurate when strong spatial mixing holds, and the time to compute the marginal is linear in the size of the truncated tree. To maintain a polynomial running time, Weitz [35] choose to truncate it at a suitable logarithmic depth.

3. Fast SSM regime for 2-spin systems

In this section we give a quadratic speedup of Weitz's Algorithm to estimate the marginal of a single vertex in 2-spin systems, albeit being randomised instead of deterministic. We use the hard-core model as our running example to illustrate the main ideas. The main result of the section is Theorem 1.1.

Let the hard-core model be described by $A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$ and $b = (1, \lambda)$. The support of the Gibbs distribution is the set of independent sets of G. Let vertices assigned 0 not be in the independent set (unoccupied) and vertices assigned 1 be in the independent set (occupied). Our algorithm uses self-reduction [26] as follows. Since unoccupied vertices contribute 1 to the

weight of a configuration, we can consider the all 0 configuration σ_0 where

$$\frac{1}{Z(G)} = \mu(\sigma_0) = \mu_{v_1}(0)\mu^{v_1\leftarrow 0}_{V\setminus\{v_1\}}(\mathbf{0}) = \mu_{v_1}(0)\frac{1}{Z(G\setminus\{v_1\})} = \mu_{G_1,v_1}(0)\mu_{G_2,v_2}(0)\cdots\mu_{G_n,v_n}(0), \quad (3)$$

and where $G_i = G\setminus\{v_1,\ldots,v_{i-1}\}$ for all $i\in[n]$. This reduces the problem of computing $Z(G)$ to computing μ_{v_1} and recursively $Z(G\setminus\{v_1\})$. As the G_i's are subgraphs of G, they have the same degree bound and still exhibit SSM. The crux of our algorithm is to design a random variable that estimates μ_v in time $\widetilde{O}(n^{1/(2k)})$.

Another ingredient we need is the lazy single-site sampler by Anand and Jerrum [2], which allows us to rapidly sample a partial configuration vertex by vertex. The original setting of [2] requires sub-exponential neighbourhood growth in order to work up to the strong spatial mixing threshold, but in our parameter regime no sub-exponential growth is required. Moreover, only the expected running time is studied in [2], while we need a tail bound. A similar analysis is done in [17, Appendix B]. For completeness, we provide a proof specialised to our setting in Appendix A.1.

LEMMA 3.1. *Let $\Delta\geq 2$ be an integer and $\lambda<\frac{1}{\Delta-1}$. Let $G=(V,E)$ be a graph with maximum degree Δ. There exists an algorithm that, for any $v\in V$, draws a sample from μ_v and halts in time $O(\log\frac{1}{\varepsilon})$ with probability at least $1-\varepsilon$.*

Our algorithm then combines the lazy sampler of Lemma 3.1 with the SAW tree of [35]. We expand the SAW tree, and then use Lemma 3.1 to sample a truncated boundary, from which we use the recursion in (2) to get our estimate. The depth of the truncation controls the variance of this estimator. In our algorithm, we only need to bound the variance from above by $1/n$. In contrast, Weitz's algorithm requires the error of the marginal incurred by the truncation to be bounded from above by $O(1/n)$. As the variance of our estimator decays twice as fast as the marginal errors, our truncation depth is roughly half of that in Weitz's algorithm. Consequently, we achieved a quadratic speedup for estimating each term in (3).

LEMMA 3.2. *Let $\mathbb{G}_\Delta$ be the family of graphs with maximum degree Δ. Suppose the hard-core model has strong spatial mixing with decay rate $C\Delta^{-k\ell}$ for some constant $C>0$ for $\mathbb{G}$. Then there exists an algorithm that, for any $G=(V,E)\in\mathbb{G}$ and $v\in V$, generates a random sample $\widetilde{p}_v$ and halts in time $O(n^{1/(2k)}(\log\frac{n}{\delta})^2)$ with probability at least $1-\frac{\delta}{8}$. Furthermore, $\mathbb{E}[\widetilde{p}_v]=\mu_v(0)$ and $\mathrm{Var}\left(\widetilde{p}_v\right)\leq 1/n$.*

PROOF. Let T_{SAW} be the self-avoiding walk tree for G rooted at v as defined in Theorem 2.4, and let $S=\{u\in V|d_{T_{\mathrm{SAW}}}(v,u)=\ell\}$ where ℓ is a parameter we will fix later. We have

$$\mu_{T_{\mathrm{SAW}},v}(0) = \sum_{\sigma\in\{0,1\}^S}\mu_{T_{\mathrm{SAW}}}(\sigma)\mu^{\sigma}_{T_{\mathrm{SAW}},v}(0) = \mathbb{E}_{\sigma\sim\mu_{T_{\mathrm{SAW}},S}}[\mu^{\sigma}_{T_{\mathrm{SAW}},v}(0)].$$

We use Lemma 3.1 to sample σ. Fix an arbitrary order of $S=\{s_1,s_2,\ldots,s_{|S|}\}$. We sample first the marginal of s_1 with $\varepsilon:=\frac{\delta}{8|S|}$. Then, conditioned on the result on s_1, we sample s_2 with the

same ε, and so on and so forth. Note that whatever the result on s_1 is, it always reduces to a hard-core instance of a smaller graph. Thus, the condition of Lemma 3.1 is always satisfied until all of S are sampled. This gives a boundary condition σ_S in T_{SAW}.

As the full SAW tree may be exponential in size, a little care is required to implement the outline above. We first expand T_{SAW} up to level ℓ, denoted $T_{\mathrm{SAW},\ell}$. The algorithm in Lemma 3.1 (Algorithm 1 in Appendix A.1) is essentially an exploration process. When we apply it to sample the boundary condition σ_S, we expand the SAW tree below $T_{\mathrm{SAW},\ell}$ on the fly, only creating vertices that are explored by the algorithm. Note that the construction of the SAW tree imposes a boundary condition whenever a vertex in G is encountered again in a self-avoiding walk. We implement this pinning by remembering a list of all ancestors of a given node in the SAW tree and checking the next vertex to explore against this list. Since Lemma 3.1 halts in $O(\log \frac{1}{\varepsilon})$ time with probability at least $1-\varepsilon$, this extra check incurs a multiplicative slowdown factor $O(\ell + \log \frac{1}{\varepsilon}) = O(\ell + \log \frac{|S|}{\delta})$ with probability at least $1-\varepsilon$.

Given σ_S, we can compute $\mu_v^{\sigma_S}(0) = \widetilde{p}_v$ with the standard dynamic programming approach. By a union bound, the total running time of sampling the boundary is $O(|S| \log \frac{|S|}{\delta}(\ell + \log \frac{|S|}{\delta}))$ with probability at least $1 - \frac{\delta}{8}$, and the dynamic programming step uses time $O(\left|T_{\mathrm{SAW},\ell}\right|)$.

We choose $\ell := \left\lceil \frac{\log(n)/2 - \log C}{k \log(\Delta)} \right\rceil$ so that $C\Delta^{-k\ell} \le n^{-0.5}$ and $\Delta^{\ell} \le C' n^{1/(2k)}$ for some constant $C' > 0$. Note that $|S| \le (\Delta-1)^{\ell}$ and $|T_{\mathrm{SAW},\ell}| \le \Delta^{\ell}$. Then the total runtime to draw a sample is $O(n^{1/(2k)}(\log \frac{n}{\delta})^2)$ with probability at least $1 - \frac{\delta}{8}$.

Finally, we analyse the variance. Strong spatial mixing implies that $\left|\mu_v^{\sigma_S}(0) - \mu_v(0)\right| \le C\Delta^{-k\ell}$ for any σ_S, so

$$\mathrm{Var}\left(\widetilde{p}_v\right) = \mathrm{Var}_{\sigma_S}\left(\mu_v^{\sigma_S}(0)\right) = \mathbb{E}_{\sigma_S \sim \mu_{T_{\mathrm{SAW}},S}}\left[\left|\mu_v^{\sigma_S}(0) - \mu_v(0)\right|^2\right] \le \left(C\Delta^{-k\ell}\right)^2 \le n^{-1}, \tag{4}$$

which is what we desire. ■

LEMMA 3.3. *For a graph G with maximum degree Δ, if $\lambda \le \frac{1}{\Delta^k(\Delta-1)}$ for some constant $k > 0$, the hard-core model on G exhibits strong spatial mixing with decay rate $C\Delta^{-k\ell}$.*

PROOF. It is well-known that if $\lambda < \lambda_c(\Delta) = \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^{\Delta}} \approx \frac{e}{\Delta}$, strong spatial mixing holds with exponential decay Cr^{ℓ} for some constant C and $r < 1$ [35]. Moreover, the decay rate r can be controlled by a quantity related to the recursion (2) [31]. For example, by [21, Lemma 7.20], r is bounded by $r \le |f'(\widehat{x})|$, where $f(x) := \frac{\lambda}{(1+x)^{\Delta-1}}$ is the symmetric version of the recursion in (2) and $\widehat{x}$ is the unique positive fixed point of f. (Note that when the degree of G is at most Δ, all vertices but the root in T_{SAW} have branching number $\Delta - 1$.) Then we have

$$|f'(x)| = \left|-\frac{f(x)(\Delta-1)}{1+x}\right| < (\Delta-1)f(x).$$

As $\widehat{x} > 0$ and $\widehat{x}$ is a fixed point,

$$\widehat{x} < \widehat{x}(1+\widehat{x})^{\Delta-1} = \lambda.$$

Thus, as $\lambda \le \frac{1}{\Delta^k(\Delta-1)}$,

$$r \le |f'(\widehat{x})| < (\Delta-1)f(\widehat{x}) = (\Delta-1)\widehat{x} < \frac{1}{\Delta^k}.$$ ■

Now we are ready to prove Theorem 1.1.

PROOF OF THEOREM 1.1. We first give an algorithm that runs in time $O\left(\frac{n^{1+1/(2k)}}{\varepsilon^2}(\log\frac{n}{\varepsilon})^2\right)$, and then improve the dependency on ε by a simple trick at the end of the proof. It would be easier to first introduce an algorithm that runs within the desired bound with high probability. To have a fixed running time upper bound, we then truncate the algorithm.

Set $N := \lceil 8e^{(1+\lambda)^2}/\varepsilon_0^2 \rceil$ where $\varepsilon_0 = \varepsilon/2$. Let $X := \prod_{i=1}^{n} \widetilde{p}_{G_i,v_i}$ where $G_1 = G$ and $G_i = G_{i-1} \setminus \{v_{i-1}\}$. By Lemma 3.3, we can use Lemma 3.2 to draw N samples of X and take its average, where we set $\delta = \frac{1}{nN}$ in Lemma 3.2. Each $\widetilde{p}_{G_i,v_i}$ can be computed in time $O(n^{1/(2k)}(\log\frac{n}{\delta})^2)$ with probability at least $1-\frac{\delta}{8}$, so computing one sample of X takes time $O(n^{1+1/(2k)}(\log\frac{n}{\delta})^2)$ time with probability at least $1-\frac{n\delta}{8}$ by a union bound. By a union bound again, the overall running time of taking the average is $O(Nn^{1+1/(2k)}(\log\frac{n}{\delta})^2) = O\left(\frac{n^{1+1/(2k)}}{\varepsilon^2}(\log\frac{n}{\varepsilon})^2\right)$ with probability at least $1-\frac{\delta}{8}\cdot nN = \frac{7}{8}$.

Since $\{\widetilde{p}_{G_i,v_i}\}$ are mutually independent, by Lemma 3.2,

$$\mathbb{E}[X] = \mathbb{E}\left[\prod_{i=1}^{n}\widetilde{p}_{G_i,v_i}\right] = \prod_{i=1}^{n}\mu_{G_i,v_i}(0) = \frac{1}{Z(G)}.$$

We bound Var (X) as follows

$$\begin{aligned}
\frac{\text{Var}\,(X)}{(\mathbb{E}[X])^2} &= \frac{\mathbb{E}[X^2]}{(\mathbb{E}[X])^2} - 1 = \frac{\prod_{i=1}^{n}\mathbb{E}[\widetilde{p}^2_{G_i,v_i}]}{\prod_{i=1}^{n}\mathbb{E}[\widetilde{p}_{G_i,v_i}]^2} - 1 \\
&= \prod_{i=1}^{n}\left(1 + \frac{\text{Var}\left(\widetilde{p}_{G_i,v_i}\right)}{(\mathbb{E}[\widetilde{p}_{G_i,v_i}])^2}\right) - 1 \\
&\le \left(1+\frac{c}{n}\right)^n - 1 && \text{(by Lemma 3.2)} \\
&< e^c,
\end{aligned}$$

where $c = \max_i(1/\mu_{G_i,v_i}(0)^2)$. Note that as $\mu_{G_i,v_i}(0) \ge \frac{1}{1+\lambda}$, $c \le (1+\lambda)^2$ and is a constant.

Let $\widetilde{X}$ be the average of N samples of X. Then Var $\left(\widetilde{X}\right) = \frac{\text{Var}(X)}{N} \le \frac{e^c}{N\cdot Z(G)^2}$. By Chebyshev's inequality,

$$\Pr\left[\left|\widetilde{X} - \frac{1}{Z(G)}\right| \ge \frac{\varepsilon_0}{Z(G)}\right] \le \frac{\text{Var}\left(\widetilde{X}\right)}{\frac{\varepsilon_0^2}{Z(G)^2}} \le \frac{e^c}{N\cdot Z(G)^2}\cdot\frac{Z(G)^2}{\varepsilon_0^2} \le \frac{1}{8}.$$

Thus, with probability at least $7/8$, we have that $\frac{1-\varepsilon_0}{Z(G)} \le \widetilde{X} \le \frac{1+\varepsilon_0}{Z(G)}$. Finally, we output $\widetilde{Z} = 1/\widetilde{X}$. To make sure that the algorithm runs within the time bound $O\left(\frac{n^{1+1/(2k)}}{\varepsilon^2}(\log\frac{n}{\varepsilon})^2\right)$, we truncate the algorithm if it runs overtime and output an arbitrary value in that case. This truncated version

can be coupled with the untruncated algorithm with probability at least $7/8$, and its output $\widetilde{Z}$ satisfies $1-\varepsilon \le \frac{\widetilde{Z}}{Z(G)} \le 1+\varepsilon$ with probability at least $7/8 - 1/8 = 3/4$.

To improve the dependency on $1/\varepsilon$, construct G' as $t = \lceil 2/\varepsilon \rceil$ copies of G. Without loss of generality we may assume that $\varepsilon < 1$ so that $t > 2$ Denote $Z' = Z(G')$ and $Z = Z(G)$. Then $Z' = Z^m$. We set $\varepsilon_1 = 1 - 1/e$ and apply the algorithm above on G' to get an estimate $\widetilde{Z}$ to Z', which satisfies

$$e^{-1} \le \frac{\widetilde{Z}}{Z'} \le 2 - 1/e < e.$$

Thus,

$$e^{-1/t} \le \frac{\widetilde{Z}^{1/t}}{Z} \le e^{1/t}.$$

As $t = \lceil 2/\varepsilon \rceil > 2$, $e^{1/t} < 1 + 2/t \le 1 + \varepsilon$ and $e^{-1/t} > 1 - \frac{1}{t} > 1 - \varepsilon$. Thus, $\widetilde{Z}^{1/t}$ is an estimate to Z within the desired accuracy. As for the running time of this algorithm, the size of G' is $tn = O\left(\frac{n}{\varepsilon}\right)$ and $\varepsilon_1 = \Omega(1)$. Thus, the overall running time is $O\left(\left(\frac{n}{\varepsilon}\right)^{1+\frac{1}{2k}} (\log \frac{n}{\varepsilon})^2\right) = \widetilde{O}\left(\left(\frac{n}{\varepsilon}\right)^{1+\frac{1}{2k}}\right)$. ■

Note that, Weitz's algorithm is faster if the correlation decay is faster, but in that case so is our algorithm. In Appendix B, Lemma B.1 shows that the correlation decay cannot be much faster than the standard analysis in the parameter regimes of Theorem 1.1, and our speed-up, comparing to Weitz's algorithm, is always at least $\widetilde{O}(n^{1/2k - o(1/k^2)})$.

We also remark that Theorem 1.1 generalises to antiferromagnetic 2-spin systems. This is because all the key ingredients, namely correlation decay, Weitz's SAW tree, and the marginal sampler of Anand and Jerrum all generalise, except that the Anand-Jerrum algorithm would require the neighbourhood growth rate smaller than the decay rate (see Theorem A.3). This is also the parameter regime where Weitz's algorithm is faster than $O(n^2)$. Thus, our speedup is still in the sub-quadratic regime. The self-reduction in (3) also generalises (as we will see in (5) in the next section). One needs to redo the calculations in Lemma 3.3 to get a precise statement, which we will omit here.

4. Speed-up on planar graphs

In this section we mainly consider (not necessarily two-state) spin systems on planar graphs. We show that for any planar graph with quadratic neighbourhood growth, when SSM holds with exponential decay, approximate counting can be done in sub-quadratic time. For example, this includes all subgraphs of the 2D lattice $\mathbb{Z}^2$. The circle-packing theorem asserts that any planar graph is the tangent graph of some circle packing. All planar graphs with bounded-radius circle packings have quadratic neighbourhood growth. Thus this is a substantial family of planar graphs. Moreover, in Section 4.2 we extend the result to (not necessarily planar) graphs with

polynomial growth, but the speed up factor there is sub-polynomial yet faster than $(\log n)^k$ for any k.

DEFINITION 4.1. A graph family $\mathbb{G}$ has *quadratic growth*, if there is a constant C_0 such that for any $G = (V, E) \in \mathbb{G}$, $v \in V$, and any integer $\ell > 0$, $|B_v(\ell)| \leq C_0\ell^2$.

Subgraphs of the 2D lattice $\mathbb{Z}^2$ satisfies Definition 4.1 with $C_0 = 5$. Note that by taking $\ell = 1$, Definition 4.1 implies that the maximum degree is no larger than C_0.

THEOREM 4.2. *Let $\mathbb{G}$ be a family of planar graphs with quadratic growth (assume the rate is $C_0\ell^2$). Let $\mathbf{A}$ and $\mathbf{b}$ specify a q-state spin system, which exhibits SSM with decay rate $Cr^{-\ell}$ on $\mathbb{G}$. Then there is a constant $c > 0$ such that there exists an FPRAS for the partition function of $G \in \mathbb{G}$ with n vertices with run-time $\widetilde{O}\left(\left(\frac{n}{\varepsilon}\right)^{2-c}\right)$. The constant c depends on C_0, q, and r.*

Theorem 4.2 is the detailed version of Theorem 1.3.

Essentially the idea is still to find an estimator for the marginal of an arbitrary vertex that can be evaluated very quickly. Let us first consider a $\sqrt{n}$-by-$\sqrt{n}$ grid. For any vertex v, we consider the sphere $S_v(\ell)$ of radius $\ell = O(\log n)$ centered at v, and a random configuration τ on $S_v(\ell)$. Let $B_v(\ell)$ be the ball of radius ℓ centered at v. Since any planar graph has linear local tree-width [13, 16], $B_v(\ell)$ has tree-width $O(\ell)$. Thus, given a configuration τ on S, the law of μ_v^τ can be computed in time $2^{O(\ell)}\text{poly}(\ell)$ for a fixed τ (see, e.g. [36][6]). This step can be very efficient with a carefully chosen ℓ.

For a general bounded degree planar graph, $|S_v(\ell)|$ can be a polynomial in n, which makes the number of possible τ's exponential in n. However, for a $\sqrt{n}$-by-$\sqrt{n}$ grid, $|S_v(\ell)| \leq 4\ell = O(\log n)$, and the number of possible τ's is much smaller and is a small polynomial in n. Thus, it would be more efficient to first create a table to list all possibilities of τ, and then, instead of computing μ_v^τ each time, simply look up the answer from this table. We can do the same for any subgraph of $\mathbb{Z}^2$ by choosing a boundary based on distance in the original grid.

For a general $G \in \mathbb{G}$, we no longer have a linear bound on the size of the boundary. See Appendix C for a subgraph of $\mathbb{Z}^2$ where the distance ℓ boundary has size $\Omega(\ell^2)$. However, since $\mathbb{G}$ has quadratic growth, we know that $B_v(\ell) \leq C_0\ell^2$ for some constant $C_0 > 0$. It implies that

$$\sum_{i=\ell/2}^{\ell} |S_v(\ell)| \leq |B_v(\ell)| \leq C_0\ell^2.$$

Thus, there must exist an $\ell' \in [\ell/2, \ell]$ such that $|S_v(\ell')| \leq 2C_0\ell$. We will find this ℓ' and use $S_v(\ell')$ instead.

6 The algorithm in [36] uses the separator decomposition. Another possibility is to first find a constant approximation of the tree decomposition first [27], and then apply Courcelle's theorem.

Once again, we use a self-reduction similar to (3). For q-spin systems, given a feasible configuration σ, we have the decomposition,

$$\frac{w(\sigma)}{Z_G} = \mu(\sigma) = \mu_{v_1}(\sigma_{v_1})\mu_{v_2}^{\sigma_{v_1}}(\sigma_{v_2})\mu_{v_3}^{\sigma_{v_1},\sigma_{v_2}}(\sigma_{v_3})\dots\mu_{v_n}^{\sigma_{v_1},\dots,\sigma_{v_{n-1}}}(\sigma_{v_n}). \tag{5}$$

When computing our table, we will have to condition on the already pinned vertices.

LEMMA 4.3. *Let* **A**, **b**, q *and* G *be as in Theorem 4.2. For* $v \in V$, *a partial configuration* σ, *and an integer* ℓ, *we can find an* ℓ' *such that* $\ell' \in [\ell/2, \ell]$, *and then construct a table of* $\mu_v^{\sigma,\tau}$, *indexed by every boundary configuration* τ *on unpinned vertices of* $S_v(\ell')$. *The total run-time is* $2^{C_1\ell}$, *where* C_1 *is a constant depending on* C_0 *and* q.

PROOF. As discussed earlier, due to the quadratic growth of G, there must exist an ℓ such that $\ell' \in [\ell/2, \ell]$ and $|S_v(\ell')| \le 2C_0\ell$. To find this ℓ, we do a breadth-first-search to check $S_v(i)$ from $i = \ell/2$ to ℓ. The running time is at most $O(B_v(\ell)) = O(\ell^2)$.

Once ℓ' is found, $\left|S_v^\sigma(\ell')\right| \le |S_v(\ell')| \le 2C_0\ell$, and there are at most $q^{2C_0\ell}$ configurations τ in our table. As G is a planar graph, the tree-width of the ball $\mathrm{tw}(B_v(\ell')) = O(\ell') = O(\ell)$. Thus, using for example the algorithm of [36], each entry of the table can be computed in time $2^{O(\ell)}\,\mathrm{poly}\,(\ell)$. The total amount of time required is $O(\ell^2) + q^{2C_0\ell}2^{O(\ell)}\,\mathrm{poly}\,(\ell) \le 2^{C_1\ell}$, for some sufficiently large constant C_1. ∎

While we may construct this table very quickly, it is not clear how to compute or estimate the marginals of the boundary condition τ's rapidly. Instead, we sample a random one using the marginal sampler [2] that terminates in almost linear time with high probability. See Theorem A.3.

LEMMA 4.4. *Let* **A**, **b**, q *and* $G \in \mathbb{G}$ *be as in Theorem 4.2. Let* σ *be a partial configuration. For any* $v \in V$ *not pinned under* σ *and any* $k \in [q]$, *there exists an algorithm that generates a random variable* $\widetilde{Z}$ *such that* $\mathbb{E}[\widetilde{Z}] = \mu_v^\sigma(k)$ *and* $\mathrm{Var}\left(\widetilde{Z}\right) \le 1/n$. *Moreover, its running time is* $\widetilde{O}(n^{1-c})$ *with high probability where* c *depends on* C_0, q, *and* r.

PROOF. Let ℓ be a constant that we will choose later. Let $\ell' \in [\ell/2, \ell]$ be as in Lemma 4.3, and let τ be a boundary condition on the unpinned vertices of $S_v(\ell')$ under σ. Let $Z_v(\tau) = \mu_v^{\sigma,\tau}(k)$ so that $\mathbb{E}_\tau[Z_v(\tau)] = \mu_v^\sigma(k)$. Then, let

$$\widetilde{Z} := \frac{1}{m}\sum_{j=1}^{m} Z_v(\tau_j)$$

be the empirical mean over m random samples τ_j, where we will choose m later.

Since the spin system exhibits SSM with decay rate $Cr^{-\ell}$, similar to (4), it follows that $\mathrm{Var}\,(Z_v(\tau)) \le C^2r^{-2\ell'} \le C^2r^{-\ell}$. Then

$$\mathrm{Var}\left(\widetilde{Z}\right) = \mathrm{Var}\left(\frac{1}{m}\sum_{j=1}^{m} Z_v(\tau_j)\right) \le \frac{C^2}{mr^{\ell'}}.$$

Thus, we set $m = \lceil nC^2 r^{-\ell'} \rceil$ samples so that $\mathrm{Var}\left(\widetilde{Z}\right) \le 1/n$.

For the running time, we first construct the table as in Lemma 4.3. Then we take m samples of τ, each of which can be generated in time almost linear in $|S_v(\ell')|$ with high probability using Theorem A.3. As $|S_v(\ell')| \le 2C_0\ell$, the runtime in total is at most $O(2^{C_1\ell} + n\ell r^{-\ell'} \log n)$ with high probability. We choose $\ell = \frac{1-c}{C_1} \log n$ for $c = \frac{\log r}{\log r + 2C_1} \in [0,1]$, so that $\ell' \ge \ell/2 = \frac{1-c}{2C_1} \log n$ and the total runtime is $O(n^{1-c} + n^{1-(1-c)\log r/(2C_1)} \log^2 n) = \widetilde{O}(n^{1-c})$. ■

Now we are ready to prove Theorem 4.2.

PROOF OF THEOREM 4.2. Using the same trick at the end of the proof of Theorem 1.1, it suffices to give an algorithm that runs in time $\widetilde{O}(n^{2-c}/\varepsilon^2)$.

We are going to use (5) to do a self-reduction. First we construct the target configuration σ adaptively. Given σ on $v_1, \dots, v_{i-1}$, we want to choose σ_{v_i} to be $k \in [q]$ with the largest marginal. In other words, $\sigma_{v_i} = \mathrm{argmax}_{k\in[q]} \mu_{v_i}^{\sigma_i}(k)$ for each i, where σ_i is what has been constructed so far, namely $\sigma_{v_1}, \dots, \sigma_{v_{i-1}}$. Of course, this step cannot be done exactly. Instead, we may fix a constant $t = t(C, r, q)$ such that $Cr^{-t} \le \frac{1}{2q}$, fix an arbitrary boundary configuration τ on $S_{v_i}^{\sigma_i}(t)$ and then pick $k \in [q]$ that maximises $\mu_{v_i}^{\sigma_i,\tau}(k)$. SSM guarantees that $\mu_{v_i}^{\sigma_i}(\sigma_{v_i}) \ge 1/2q$, where $\sigma_{v_i} = k$. This step takes constant time as t is a constant.

The rest of the proof is very similar to that of Theorem 1.1. Set $N := \lceil 10e^{4q^2}/\varepsilon_0^2 \rceil$ where $\varepsilon_0 = \varepsilon/2$. We compute $X = \prod_{i=1}^n \widetilde{Z}_i$ where each $\widetilde{Z}_i$ is from Lemma 4.4 plugging in v_i and σ_i. Due to the decomposition (5) we have

$$\mathbb{E}[X] = \mathbb{E}\left[\prod_{i=1}^n \widetilde{Z}_i\right] = \prod_{i=1}^n \mathbb{E}\left[\widetilde{Z}_i\right] = \prod_{i=1}^n \mu_{v_i}^{\sigma_i}(\sigma_{v_i}) = \mu(\sigma).$$

We also compute $w(\sigma)$ which can be done in $O(n)$ on a planar graph with quadratic growth. By Lemma 4.4, the time to generate one X is $O(n^{2-c}\,\mathrm{polylog}\,(n))$ with high probability. We bound $\mathrm{Var}\,(X)$ as follows

$$\begin{aligned}\frac{\mathrm{Var}\,(X)}{(\mathbb{E}[X])^2} = \frac{\mathbb{E}[X^2]}{(\mathbb{E}[X])^2} - 1 = \frac{\prod_{i=1}^n \mathbb{E}[\widetilde{Z}_i^2]}{\prod_{i=1}^n \mathbb{E}[\widetilde{Z}_i]^2} - 1 &= \prod_{i=1}^n \left(1 + \frac{\mathrm{Var}\left(\widetilde{Z}_i\right)}{(\mathbb{E}[\widetilde{Z}_i])^2}\right) - 1 \\ &\le \left(1 + \frac{4q^2}{n}\right)^n - 1 \le e^{4q^2},\end{aligned}$$

where we use $\mu_{v_i}^{\sigma_i}(\sigma_{v_i}) \ge 1/2q$ for any $i \in [n]$. Let $\widetilde{X}$ be the average of N samples of X. Then $\mathrm{Var}\left(\widetilde{X}\right) = \frac{\mathrm{Var}(X)}{N} \le \frac{e^{4q^2}}{N \cdot Z(G)^2}$. By Chebyshev's inequality,

$$\Pr\left[\left|\widetilde{X} - \frac{1}{Z(G)}\right| \ge \frac{\varepsilon_0}{Z(G)}\right] \le \frac{\mathrm{Var}\left(\widetilde{X}\right)}{\frac{\varepsilon_0^2}{Z(G)^2}} \le \frac{e^{4q^2}}{N \cdot Z(G)^2} \cdot \frac{Z(G)^2}{\varepsilon_0^2} \le \frac{1}{10}.$$

Thus, with probability at least $9/10$, we have that $\frac{1-\varepsilon_0}{Z(G)} \le \widetilde{X} \le \frac{1+\varepsilon_0}{Z(G)}$. Finally, we output $\widetilde{Z} = w(\sigma)/\widetilde{X}$. Since Definition 4.1 implies a constant degree bound, the graph is sparse and $w(\sigma)$ can be computed in $O(n)$ time. To make sure that the algorithm runs within the time bound $O\left(\frac{n^{2-c}}{\varepsilon^2}\right)$, we truncate the algorithm if it runs overtime and output an arbitrary value in that case. This truncated version can be coupled with the untruncated algorithm with probability at least $7/8$, and its output $\widetilde{Z}$ satisfies $1-\varepsilon \le \frac{\widetilde{Z}}{Z(G)} \le 1+\varepsilon$ with probability at least $3/4$. ■

4.1 Bounded-radius circle packing

Here we show that Theorem 4.2 applies to any planar graph with bounded-radius circle packings. We begin with the definition of a circle packing.

DEFINITION 4.5. A *circle packing* is a collection C of interior-disjoint circles over the 2-dimensional plane. A *tangency graph* of a circle packing is a graph having a vertex for each circle, and an edge between two vertices if and only if the two corresponding circles are tangent.

The Koebe-Andreev-Thurston *circle packing theorem* states the following.

THEOREM 4.6. *For every connected locally finite simple planar graph $\mathbb{G}$, there exists a circle packing whose tangency graph is (isomorphic to) $\mathbb{G}$.*

We are concerned with the radius of the circles used in the packing, especially the ratio between the smallest and largest ones.

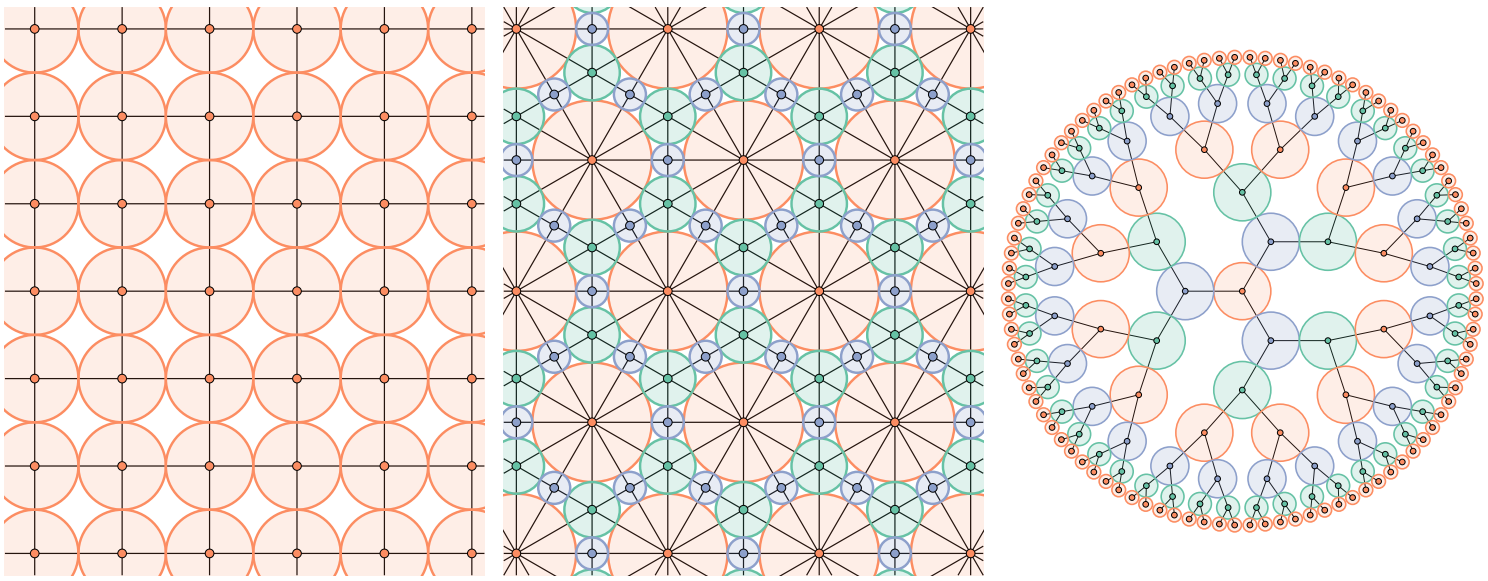
DEFINITION 4.7. A locally finite simple planar graph $\mathbb{G}$ is said to have an *R-bounded-radius circle packing* (R-BRCP) for some constant $R > 0$, if there exists a circle packing C whose tangency graph is (isomorphic to) $\mathbb{G}$ such that

$$\frac{\inf\limits_{\odot\in C} r_\odot}{\sup\limits_{\odot\in C} r_\odot} \ge R$$

where $r_\odot$ denotes the radius of a circle $\odot$ in the packing.

Three examples are given in Figure 2. The $\mathbb{Z}^2$ grid can be naturally packed by unit disks, leading to $R = 1$. Such a graph is called a "penny graph". The 3, 6-kisrhombille tiling is a tiling of the 2-dimensional plane by $\pi/6$-$\pi/3$-$\pi/2$ triangles. This lattice can be packed by circles of radii $1, 2\sqrt{3}-3, 2-\sqrt{3}$, so $R = 2-\sqrt{3}$. The degree-3 Bethe lattice, also known as the infinite 3-regular tree, can be drawn as a planar graph on the 2-dimensional plane. However, the neighbourhood growth is so fast that $R = 0$.

Fix the underlying graph $\mathbb{G}$ and its R-BRCP C. Without loss of generality, we assume the diameter of the largest circle in C is 1. Thus, the radius of an arbitrary circle in C is between $R/2$ and $1/2$. Let G be a finite subgraph of $\mathbb{G}$. Here we need to distinguish the graph



(a) (b) (c)

Figure 2. Circle packings of some lattices. (a): $\mathbb{Z}^2$ grid, $R = 1$. (b): Kisrhombille tiling, $R = 2 - \sqrt{3}$. (c): degree-3 Bethe lattice, $R = 0$.

distance in G and the geometric distance (the Euclidean distance $\|\cdot\|_2$ between the center of their corresponding disks on the 2-dimensional plane). For two vertices u and v, we use $\mathrm{dist}_G(u, v)$ to denote their graph distance, and use $\|u - v\|_2$ to denote their geometric distance. Note that $\mathrm{dist}_{\mathbb{G}}(u, v) \geq \|u - v\|_2$ and $\mathrm{dist}_G(u, v) \geq \mathrm{dist}_{\mathbb{G}}(u, v)$.

For any vertex v and u in the ℓ-ball $B_v(\ell)$ in G, $\|u - v\|_2 \leq \mathrm{dist}_G(u, v) \leq \ell$. The disk $\odot_u$ corresponding to u must be contained completely in the circle centered at u with radius $\ell + 1/2$. By considering the area they cover,

$$|B_v(\ell)| \leq \frac{\pi(\ell + 1/2)^2}{\pi(R/2)^2} = O(\ell^2/R^2).$$

Thus, any family of subgraphs of $\mathbb{G}$ has quadratic growth, where the growth constant depends on R. Together with Theorem 4.2, we have the following corollary.

COROLLARY 4.8. *Let $\mathbb{G}$ be a locally finite simple planar graph, together with an R-BRCP where $R > 0$ is a constant. Let $\mathcal{G}$ be a family of subgraphs of $\mathbb{G}$, and $\mathbf{A}, \mathbf{b}$ specify a q-spin system that exhibits SSM with exponential decay on $\mathcal{G}$. Then there exists an FPRAS that takes a graph $G \in \mathcal{G}$ as an input and estimates the partition function of the spin system on G in time $\widetilde{O}\left(\left(\frac{n}{\varepsilon}\right)^{2-c}\right)$. Here, $n = |V(G)|$, and $c > 0$ is a constant depending on q, decay rate of SSM, and R.*

REMARK 4.9. The algorithm does not need to know the circle packing, as long as an R-BRCP exists.

On a separate note, although a good approximation of the circle packing of a finite planar graph can be found in near linear time [14], its output does not optimise the radius ratio. It is not clear how to generate a circle packing with a constant approximation of the optimal radius

ratio. In the extreme, it is **NP**-hard to decide if a given graph G (without geometric positions) is a penny graph, namely admitting a circle packing using unit circles [15], even if G is restricted to be a tree [6].

4.2 Polynomial-growth graphs

Our method goes beyond planar graphs with quadratic growth rate. For any graph with a polynomial growth rate, we have a speed-up that is faster than any polylog factors.

DEFINITION 4.10. A graph family $\mathbb{G}$ has *polynomial growth*, if there are constants C and d such that for any $G = (V, E) \in \mathbb{G}$, $v \in V$, and any integer $\ell > 0$, $|B_v(\ell)| \le C_0 \ell^d$.

Examples of graphs with polynomial growth include finite subgraphs of the d-dimensional integer lattice $\mathbb{Z}^d$. Again, by taking $\ell = 1$, Definition 4.10 implies that the maximum degree is no larger than C_0.

THEOREM 4.11. *Let $\mathbb{G}$ be a family of graphs with polynomial growth (assume the rate is $C_0\ell^d$). Let $\mathbf{A}$ and $\mathbf{b}$ specify a q-state spin system, which exhibits SSM with decay rate $Cr^{-\ell}$ on $\mathbb{G}$. Then there is a constant $c > 0$ such that there exists an FPRAS for the partition function of $G \in \mathbb{G}$ with n vertices with run-time $\widetilde{O}\left(\frac{n^2}{\varepsilon^2 2^{c(\log \frac{n}{\varepsilon})^{1/d}}}\right)$. The constant c depends on C_0, q, and r.*

Theorem 4.11 is the detailed version of Theorem 1.4.

In comparison to Theorem 4.2, the proof of Theorem 4.11 needs only a few small tweaks. Let ℓ be a parameter we will choose later, and our estimator is still set by using a random boundary condition on $S_v(\ell)$ to estimate the marginal at v. Note that we no longer need to find ℓ' for a smaller boundary. The main difference is in Lemma 4.3, where we no longer have linear local tree-width. Instead, we have to create the table by brute-force enumeration. There are $q^{C_0\ell^d}$ possible boundary conditions, and the overall time cost for creating the table is $O(q^{2C_0\ell^d})$.

We use the same estimator as in Lemma 4.4. To reduce the variance of our estimator to $1/n$, we need $nC^2r^{-2\ell}$ samples, each of which can be looked up quickly using the table. Let $\ell = \frac{0.99(\log n)^{1/d}}{2C_0 \log q}$. The overall time cost is

$$\widetilde{O}\left(\frac{n}{\varepsilon^2}\left(q^{2C\ell^d} + nr^{-2\ell}\right)\right) = \widetilde{O}\left(\frac{n}{\varepsilon^2}\left(n^{0.99} + \frac{n}{2^{c(\log n)^{1/d}}}\right)\right) = \widetilde{O}\left(\frac{n^2}{\varepsilon^2 2^{c(\log n)^{1/d}}}\right),$$

where $c = \frac{0.99 \log r}{C_0 \log q}$. To finish the proof of Theorem 4.11, we employ the trick at the end of the proof of Theorem 1.1 once again. Note that the factor $2^{c(\log \frac{n}{\varepsilon})^{1/d}}$ grows faster than $\left(\log \frac{n}{\varepsilon}\right)^k$ for any constant $k > 0$.

Acknowledgement

We would like to thank Chunyang Wang for pointing out how to shave a factor of e from Lemma A.2.

References

[1] **Konrad Anand**, **Weiming Feng**, **Graham Freifeld**, **Heng Guo**, **and Jiaheng Wang**. Approximate counting for spin systems in sub-quadratic time. *51st International Colloquium on Automata, Languages, and Programming*, volume 297 of *LIPIcs*, 11:1–11:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. DOI (1)

[2] **Konrad Anand and Mark Jerrum**. Perfect sampling in infinite spin systems via strong spatial mixing. *SIAM J. Comput.* 51(4):1280–1295, 2022. DOI (3–6, 9, 14, 20, 21)

[3] **Nima Anari**, **Vishesh Jain**, **Frederic Koehler**, **Huy Tuan Pham**, **and Thuy-Duong Vuong**. Entropic independence: optimal mixing of down-up random walks. *54th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 1418–1430. ACM, 2022. DOI (2)

[4] **Alexander I. Barvinok**. Combinatorics and Complexity of Partition Functions, volume 30 of *Algorithms and combinatorics*. Springer, 2016. DOI (3)

[5] **Amartya Shankha Biswas**, **Ronitt Rubinfeld**, **and Anak Yodpinyanee**. Local access to huge random objects through partial sampling. *11th Innovations in Theoretical Computer Science Conference, ITCS*, volume 151 of *LIPIcs*, 27:1–27:65. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. DOI (6)

[6] **Clinton Bowen**, **Stephane Durocher**, **Maarten Löffler**, **Anika Rounds**, **André Schulz**, **and Csaba D. Tóth**. Realization of simply connected polygonal linkages and recognition of unit disk contact trees. *GD*, volume 9411 of *Lecture Notes in Computer Science*, pages 447–459. Springer, 2015. DOI (18)

[7] **Jin-Yi Cai and Xi Chen**. Complexity of counting CSP with complex weights. *J. ACM*, 64(3):19:1–19:39, 2017. DOI (2)

[8] **Ruoxu Cen**, **William He**, **Jason Li**, **and Debmalya Panigrahi**. Beyond the quadratic time barrier for network unreliability. *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024*, pages 1542–1567. SIAM, 2024. DOI (5)

[9] **Xiaoyu Chen**, **Weiming Feng**, **Yitong Yin**, **and Xinyuan Zhang**. Optimal mixing for two-state anti-ferromagnetic spin systems. *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 588–599. IEEE, 2022. DOI (2, 3)

[10] **Yuansi Chen and Ronen Eldan**. Localization schemes: A framework for proving mixing bounds for markov chains (extended abstract). *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 110–122. IEEE, 2022. DOI (2, 3)

[11] **Zongchen Chen**, **Kuikui Liu**, **and Eric Vigoda**. Optimal mixing of Glauber dynamics: Entropy factorization via high-dimensional expansion. *53rd Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 1537–1550. ACM, 2021. DOI (2, 3)

[12] **Timothy Chu**, **Yu Gao**, **Richard Peng**, **Sushant Sachdeva**, **Saurabh Sawlani**, **and Junxing Wang**. Graph sparsification, spectral sketches, and faster resistance computation via short cycle decompositions. *SIAM J. Comput.* 52(6):S18–85, 2023. DOI (5)

[13] **Erik D. Demaine and Mohammad Taghi Hajiaghayi**. Equivalence of local treewidth and linear local treewidth and its algorithmic applications. *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 840–849. SIAM, 2004. (13)

[14] **Sally Dong**, **Yin Tat Lee**, **and Kent Quanrud**. Computing circle packing representations of planar graphs. *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 2860–2875. SIAM, 2020. DOI (17)

[15] **Peter Eades and Sue Whitesides**. The logic engine and the realization problem for nearest neighbor graphs. *Theor. Comput. Sci.* 169(1):23–37, 1996. DOI (18)

[16] **David Eppstein**. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27(3):275–291, 2000. DOI (13)

[17] **Weiming Feng**, **Heng Guo**, **Chunyang Wang**, **Jiaheng Wang**, **and Yitong Yin**. Towards derandomising Markov chain Monte Carlo. *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 1963–1990. IEEE, 2023. DOI (6, 9, 21, 22)

[18] **Weiming Feng**, **Heng Guo**, **and Yitong Yin**. Perfect sampling from spatial mixing. *Random Struct. Algorithms*, 61(4):678–709, 2022. DOI (3)

[19] **Andreas Galanis**, **Daniel Štefankovič**, **and Eric Vigoda**. Inapproximability for antiferromagnetic spin systems in the tree nonuniqueness region. *J. ACM*, 62(6):50:1–50:60, 2015. DOI (2)

[20] **Andreas Galanis, Daniel Štefankovič, and Eric Vigoda**. Inapproximability of the partition function for the antiferromagnetic Ising and hard-core models. *Combin. Probab. Comput.* 25(4):500–559, 2016. DOI (2)

[21] **Heng Guo**. Complexity Classification of Exact and Approximate Counting Problems. PhD thesis, University of Wisconsin - Madison, 2015. (10)

[22] **Thomas P. Hayes and Alistair Sinclair**. A general lower bound for mixing of single-site dynamics on graphs. *Ann. Appl. Probab.* 17(3):931–952, 2007. DOI (2)

[23] **Mark Huber**. Approximation algorithms for the normalizing constant of Gibbs distributions. *Ann. Appl. Probab.* 25(2):974–985, 2015. DOI (2)

[24] **Mark Jerrum and Alistair Sinclair**. Approximating the permanent. *SIAM J. Comput.* 18(6):1149–1178, 1989. DOI (2)

[25] **Mark Jerrum and Alistair Sinclair**. Polynomial-time approximation algorithms for the Ising model. *SIAM J. Comput.* 22(5):1087–1116, 1993. DOI (2)

[26] **Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani**. Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.* 43:169–188, 1986. DOI (2–4, 8)

[27] **Frank Kammer and Torsten Tholey**. Approximate tree decompositions of planar graphs in linear time. *Theor. Comput. Sci.* 645:60–90, 2016. DOI (13)

[28] **Vladimir Kolmogorov**. A faster approximation algorithm for the Gibbs partition function. *Conference On Learning Theory, COLT*, volume 75 of *Proceedings of Machine Learning Research*, pages 228–249. PMLR, 2018. (2)

[29] **Liang Li, Pinyan Lu, and Yitong Yin**. Correlation decay up to uniqueness in spin systems. *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 67–84. SIAM, 2013. Full version from arXiv at `abs/1111.7064`. DOI (24)

[30] **Viresh Patel and Guus Regts**. Deterministic polynomial-time approximation algorithms for partition functions and graph polynomials. *SIAM J. Comput.* 46(6):1893–1919, 2017. DOI (3)

[31] **Alistair Sinclair, Piyush Srivastava, and Marc Thurley**. Approximation algorithms for two-state anti-ferromagnetic spin systems on bounded degree graphs. *J. Stat. Phys.* 155(4):666–686, 2014. (10)

[32] **Allan Sly and Nike Sun**. Counting in two-spin models on d-regular graphs. *Ann. Probab.* 42(6):2383–2416, 2014. DOI (2)

[33] **Daniel Štefankovič, Santosh S. Vempala, and Eric Vigoda**. Adaptive simulated annealing: A near-optimal connection between sampling and counting. *J. ACM*, 56(3):18:1–18:36, 2009. DOI (2)

[34] **Leslie G. Valiant**. The complexity of computing the permanent. *Theor. Comput. Sci.* 8:189–201, 1979. DOI (2)

[35] **Dror Weitz**. Counting independent sets up to the tree threshold. *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, STOC*, pages 140–149. ACM, 2006. DOI (3, 8–10, 24)

[36] **Yitong Yin and Chihao Zhang**. Approximate counting via correlation decay on planar graphs. *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 47–66. SIAM, 2013. DOI (5, 13, 14)

A. Lazy marginal samplers

A.1 Specialised to hard-core models

Lemma 3.1 is proved in this subsection. The single-site Anand-Jerrum algorithm adapts to the hard-core model as in Algorithm 1.

The correctness of the algorithm is summarised by the following theorem, adapted to our setting.

THEOREM A.1 ([2, Theorem 5.3]). *Suppose G is a graph with maximum degree bounded by Δ, and $\lambda < \lambda_c(\Delta)$. If the untruncated algorithm* $\mathsf{HardcoreSampler}_{+\infty}(G, \lambda, (\Sigma, \sigma), v)$ *terminates with probability* 1*, then it generates a spin of v according to the correct marginal distribution upon termination, provided that the partial configuration (Σ, σ) is feasible.*

We remark that the correctness does not rely on the graph's neighbourhood growth being sub-exponential. However, the algorithm given here is a special case of that in [2], where

Input: a Δ-degree graph G, fugacity λ, a set of vertices $\Sigma \subseteq V$ with a configuration $\sigma \in \Omega_\Sigma$, and vertex to sample $v \notin \Sigma$
Output: the partial configuration passed in with a spin at v: $(\Sigma, \sigma) \oplus (v, i)$ for some $i \in \{0, 1\}$.

Decrease the global timer $T \leftarrow T - 1$
if there exists $u \in \Sigma \cap N(v)$ such that $\sigma(u) = 1$ **then**
    return $((\Sigma, \sigma) \oplus (v, 0))$
Sample random $X \in \{\perp, 0\}$ with $\Pr[X = \perp] = \lambda/(1+\lambda)$ and $\Pr[X = 0] = 1/(1+\lambda)$
if $X = \perp$ **then**
    $(\Sigma', \sigma') \leftarrow (\Sigma, \sigma)$
    $Y \leftarrow 1$
    forall $u \in N(v) \backslash \Sigma$ **do**
        $(\Sigma', \sigma') \leftarrow \mathsf{HardcoreSampler}(G, \lambda, (\Sigma', \sigma'), u)$
        if $\sigma'(u) = 1$ **then** $Y \leftarrow 0$
    return $((\Sigma, \sigma) \oplus (v, Y))$
else
    return $((\Sigma, \sigma) \oplus (v, 0))$

Algorithm 1. $\mathsf{HardcoreSampler}(G, \lambda, (\Sigma, \sigma), v)$

they look at an ℓ-distance neighbourhood. Fixing $\ell = 1$ as we do here results in the regime of fugacity λ being worse than the critical λ_c, as we will see very soon. The saving grace of [2] is that other ℓ's might be chosen in order to get to the critical regime, but this is at the cost of limiting the neighbourhood growth. Our main algorithm does not work up to the critical λ_c, so only the 1-hop neighbourhood is considered.

In [2], the expected running time is studied and turns out to be a constant depending on the parameters of the model. However, we further need an exponential tail bound of the algorithm. This is done by the same idea of [17, Section B.3], though we do not truncate this algorithm as is done there. As soon as an exponential tail bound of running time is established, the algorithm then terminates with probability 1 and hence is correct.

We treat the algorithm as a branching process. Each time the algorithm recurses into its neighbourhood, it creates at most $\Delta - 1$ new copies of the routine $\mathsf{HardcoreSampler}$. Such branching happens with probability $p := \lambda/(1+\lambda)$. This leads us to study the following Markovian process that stochastically dominates the actual branching process. Let $(X_t)_{t \in \mathbb{Z}_{\geq 0}}$ be

a discrete Markov chain where $X_t \in \mathbb{Z}_{\geq 0}$ with initial state $X_0 = 1$. This chain has an absorbing barrier at 0, and for any other $X_t > 0$, the transition probability is given by

$$X_{t+1} \leftarrow \begin{cases} X_t + \Delta - 1 & \text{with probability } p; \\ X_t - 1 & \text{with probability } 1 - p. \end{cases} \tag{6}$$

In the general case, the tail bound of this process is proved in [17, Lemma B.12], and this requires $\lambda \leq \frac{1}{2e\Delta - 1}$ when specialised to the hard-core model. Here we provide a stronger analysis to remove the constant.

LEMMA A.2. *Suppose $\lambda < \frac{1}{\Delta - 1}$. For any $0 < \varepsilon < 1$, let $T = \frac{2\Delta^2}{\left(\frac{\lambda}{1+\lambda}\Delta - 1\right)^2} \log \frac{1}{\varepsilon}$. Then with probability at most ε, the process (X_t) defined by (6) does not terminate in T rounds.*

PROOF. Given $\{X_t\}_{t \in \mathbb{Z}_{\geq 0}}$, define an auxiliary process $\{Y_t\}_{t \in \mathbb{Z}_{\geq 0}}$ in the following way. Let $Y_0 = 1$, and the transition probability is given by

$$Y_{t+1} \leftarrow \begin{cases} Y_t + \frac{1}{1+\lambda}\Delta & \text{with probability } \frac{\lambda}{1+\lambda}; \\ Y_t - \frac{\lambda}{1+\lambda}\Delta & \text{with probability } \frac{1}{1+\lambda}. \end{cases} \tag{7}$$

Then couple X_t with Y_t perfectly that, if X_t increases then so does Y_t, and vice versa, till X_t reaches the absorbing barrier. After this point, Y_t just performs the above transition independently.

Clearly, $\{Y_t\}$ is a martingale, and if $X_t > 0$ is not absorbed then $Y_t = X_t + \left(\frac{\lambda}{1+\lambda}\Delta - 1\right) t$. Also note that the regime on λ ensures $\frac{\lambda}{1+\lambda}\Delta - 1 > 0$. This allows us to bound the probability of $\{X_t\}$ not terminating after T rounds by applying Azuma–Hoeffding inequality:

$$\Pr[X_T > 0] = \Pr[X_T \geq X_0] = \Pr\left[Y_T - Y_0 \geq T \cdot \left(\frac{\lambda}{1+\lambda}\Delta - 1\right)\right] \leq \exp\left\{-\frac{T^2\left(\frac{\lambda}{1+\lambda}\Delta - 1\right)^2}{2\Delta^2 T}\right\} = \varepsilon.$$

■

Lemma 3.1 then follows by exactly the same argument as in [17, Proof of Lemma B.10], by noticing that the branching process (X_t) stochastically dominates the number of 'active' instances of HardcoreSampler, and using Lemma A.2.

A.2 Generic sampler

If we want to cover the whole strong spatial mixing regime but only work on amenable graphs, then we can invoke the original Anand-Jerrum algorithm, allowing us to do recursion at farther vertices rather than one-hop neighbours. For completeness, we include the algorithm here (Algorithm 2 and Algorithm 3). Its running time tail bound is shown in [17, Lemma B.10].

THEOREM A.3 ([17, Lemma B.10]). *Suppose a q-spin system $\mathcal{S} = (G, [q], \mathbf{b}, \mathbf{A})$ exhibits strong spatial mixing with decay rate $f(\ell)$, and there is a function $s(\ell)$ such that the neighbourhood*

Input: a spin system $\mathcal{S} = (G, [q], \mathbf{b}, \mathbf{A})$, a set of vertices $\Sigma \subseteq V$ with a configuration $\sigma \in \Omega_\Sigma$, a vertex to sample $v \notin \Sigma$, and a distance $r \in \mathbb{N}$

Output: the partial configuration passed in with a spin at v: $(\Sigma, \sigma) \oplus (v, i)$ for some $i \in [q]$.

for $i \in [q]$ **do**
    $p_v^i \leftarrow \min_{\tau \in \Omega_{S_r \setminus \Sigma}} \mu^{\sigma \oplus \tau}(i)$
$p_v^0 \leftarrow 1 - \sum_{i \in [q]} p_v^i$
Sample a random value $X \in \{0, 1, \ldots, q\}$ with $\Pr[X = i] = p_v^i$ for each $0 \le i \le q$
if $X = 0$ **then**
    $(\rho_1, \rho_2, \ldots, \rho_q) \leftarrow \text{BoundarySplit}(\mathcal{S}, (\Sigma, \sigma), v, r, (p_v^0, p_v^1, p_v^2, \ldots, p_v^q))$
    Sample a random value $Y \in [q]$ with $\Pr[Y = i] = \rho_i$ for each $1 \le i \le q$
    return $((\Sigma, \sigma) \oplus (v, Y))$
else
    return $((\Sigma, \sigma) \oplus (v, X))$

Algorithm 2. LazySampler$(\mathcal{S}, (\Sigma, \sigma), v, r)$

Input: a spin system $\mathcal{S} = (G, [q], \mathbf{b}, \mathbf{A})$, a set of vertices $\Sigma \subseteq V$ with a configuration $\sigma \in \Omega_\Sigma$, a vertex to sample $v \notin \Sigma$, a distance $r \in \mathbb{N}$, and a probability distribution $(p_v^0, p_v^1, p_v^2, \ldots, p_v^q)$

Output: a probability distribution $(\rho_1, \rho_2, \ldots, \rho_q)$

Let $S_r(v) \leftarrow \{u \mid \text{dist}_G(u, v) = r\}$ Give $S_r(v) \setminus \Sigma$ an arbitrary ordering $S_r(v) \setminus \Sigma = \{w_1, w_2, \ldots, w_m\}$
$(\Sigma', \sigma') \leftarrow (\Sigma, \sigma)$
for $1 \le j \le m$ **do**
    $(\Sigma', \sigma') \leftarrow \text{LazySampler}(\mathcal{S}, (\Sigma', \sigma'), w_j, r)$
for $i \in [q]$ **do**
    $\rho_i \leftarrow (\mu_v^{\sigma'}(i) - p_v^i)/p_v^0$
return $(\rho_1, \rho_2, \ldots, \rho_q)$

Algorithm 3. BoundarySplit$(\mathcal{S}, (\Sigma, \sigma), v, r, (p_v^0, p_v^1, p_v^2, \ldots, p_v^q))$

growth of G satisfies $|\{u \mid \mathrm{dist}_G(u,v) = \ell\}| \le s(\ell)$ for all v. If there is some $r \in \mathbb{Z}_{\ge 1}$ such that $2eq(1+s(r))f(r) \le 1$, then for any feasible boundary configuration (Σ, σ), the algorithm LazySampler$(\mathcal{S}, (\Sigma,\sigma), v, r)$ *generates a sample of v subject to the correct marginal distribution, and halts in time $O(s(r)\log\frac{1}{\varepsilon})$ with probability at least $1-\varepsilon$.*

B. A lower bound for Weitz's algorithm

In this section, we prove a lower bound for the running time of the standard implementation of Weitz's algorithm. Consider the hard-core model on $G = (V, E)$ with parameter λ. Suppose we want to estimate the partition function Z within a *constant* approximation error. Let $V = \{v_1, \ldots, v_n\}$ and $G_i = G \setminus \{v_1, \ldots, v_{i-1}\}$. Weitz's algorithm solves this task by estimating each $\mu_{G_i, v_i}(0)$ within an approximation error $O(\frac{1}{n})$. It first constructs the SAW tree of G_i rooted at v_i, then truncates the tree at level ℓ and applies dynamic programming on the truncated tree to estimate $\mu_{G_i,v_i}(0)$. The standard implementation of Weitz's algorithm [35, 29] ensures that for any tree with maximum degree Δ, any two configurations σ, τ at level ℓ, $d_{\mathrm{TV}}(\mu_v^\sigma, \mu_v^\tau) = O(\frac{1}{n})$. Standard analysis bounds the total running time from above by $T_{\mathrm{Weitz}} = \Theta(n\Delta^\ell)$.

By the same correlation decay analysis as in Lemma 3.3, when the algorithm in Theorem 1.1 has running time $\widetilde{O}(n^{1+1/2k})$, we need to choose ℓ so that $T_{\mathrm{Weitz}} = O(n^{1+1/k})$. This analysis only gives an upper bound on the correlation decay rate. If the decay rate is faster, then Weitz's algorithm is faster, and so is the algorithm in Theorem 1.1. The speedup will depend on how much faster the decay rate becomes. Nevertheless, the next lemma shows that the analysis in Lemma 3.3 is almost sharp in the worst case. The speedup in Theorem 1.1 is at least $\widetilde{\Omega}\left(n^{\frac{1}{2k} - O(\frac{1}{k^2\log\Delta})}\right)$.

LEMMA B.1. *Let the real number $k > 0$ and the integer $\Delta \ge 2$ be two constants satisfying $\Delta^k \ge 4$. Let $\lambda = \frac{2}{(\Delta-1)\Delta^k}$. Let T be an infinite Δ-regular tree with root v. For any $\ell \ge 2$, let σ_0 and σ_1 be all-0 and all-1 configurations at level ℓ of T respectively. The Gibbs distribution μ of the hard-core model on T with parameter λ satisfies*

$$d_{\mathrm{TV}}(\mu_v^{\sigma_0}, \mu_v^{\sigma_1}) \ge \frac{1}{2}\left(\frac{1}{\Delta^k}\right)^\ell.$$

Let the parameters k, Δ, and λ be as in Lemma B.1. Consider a family of hard-core instances where the graphs are indeed Δ-regular trees. In Weitz's algorithm, in order to ensure an $O(\frac{1}{n})$ truncation error, Lemma B.1 implies that ℓ must satisfy $\frac{1}{2}\left(\frac{1}{\Delta^k}\right)^\ell = O(\frac{1}{n})$, namely,

$$\Delta^\ell = \Omega(n^{\frac{1}{k}}).$$

This makes the overall running time $T_{\mathrm{Weitz}} = \Omega(n^{1+\frac{1}{k}})$. In comparison, for these parameters, the algorithm in Theorem 1.1 has a running time upper bound $\widetilde{O}\left(n^{1+\frac{1}{2k}+O(\frac{1}{k^2\log\Delta})}\right)$, which is faster by a factor of roughly $\widetilde{\Omega}(n^{1/2k})$.

PROOF OF LEMMA B.1. Let w be an arbitrary vertex at level $0 \le t \le \ell$. Let π denote the Gibbs distribution on the subtree rooted T_w at w. Recall that σ_0, σ_1 are pinnings on $T(\ell)$, where $T(\ell)$ is level ℓ of T. Let $p_t^0(c) = \pi_w^{\sigma_0}(c)$ and $p_t^1(c) = \pi_w^{\sigma_1}(c)$ for $c \in \{0,1\}$, where we use σ_0 and σ_1 to denote all-0 and all-1 pinnings on $T_w \cap T(\ell)$. By symmetry, $p_t^0(\cdot)$ and $p_t^1(\cdot)$ depend only on t but not on w. In particular, $p_0^0 = \mu_v^{\sigma_0}$ and $p_0^1 = \mu_v^{\sigma_1}$ for the root v. For any $0 \le t \le \ell$, define

$$R_t^0 := \frac{p_t^0(1)}{p_t^0(0)}, \quad R_t^1 := \frac{p_t^1(1)}{p_t^1(0)}.$$

We next prove the following result holds for all $1 \le t \le \ell - 1$:

$$\left|R_t^0 - R_t^1\right| \ge \frac{1}{2}\left(\frac{1}{\Delta^k}\right)^{\ell-t-1}. \tag{8}$$

We need the following bound to prove (8). By considering the worst pinning on the neighbourhood, we have the following bound on both ratios R_s^0 and R_s^1

$$\forall 0 \le s \le \ell - 1, \quad R_s^0, R_s^1 \le \lambda = \frac{2}{(\Delta-1)\Delta^k}. \tag{9}$$

We prove (8) by induction on t from $\ell - 1$ to 1. The base case is $t = \ell - 1$. Note that $\Delta^k \ge 4$. A straightforward calculation shows that

$$\left|R_{\ell-1}^0 - R_{\ell-1}^1\right| = |1 - \lambda| = 1 - \frac{2}{(\Delta-1)\Delta^k} \ge \frac{1}{2}.$$

For the induction step, fix $1 \le t \le \ell - 2$. The recursion function in $(\Delta-1)$-ary tree is

$$f(x) = \lambda\left(\frac{1}{1+x}\right)^{\Delta-1}.$$

Note that $R_t^0 = f(R_{t+1}^0)$ and $R_t^1 = f(R_{t+1}^1)$. By the mean value theorem, there exists θ such that $\min(R_{t+1}^0, R_{t+1}^1) < \theta < \max(R_{t+1}^0, R_{t+1}^1)$ and

$$\left|R_t^0 - R_t^1\right| = |f'(\theta)| \cdot \left|R_{t+1}^0 - R_{t+1}^1\right|.$$

By (9) and the fact $\Delta^k \ge 4$, we have

$$\begin{aligned}|f'(\theta)| &= \lambda(\Delta-1)\left(\frac{1}{1+\theta}\right)^{\Delta} \ge \lambda(\Delta-1)\left(\frac{1}{1+\lambda}\right)^{\Delta} \ge \lambda(\Delta-1)\exp(-\lambda\Delta) \\ &= \frac{2}{\Delta^k}\exp\left(-\frac{2\Delta}{(\Delta-1)\Delta^k}\right) \ge \frac{2}{\Delta^k}\exp\left(-\frac{4}{\Delta^k}\right) \ge \frac{1}{\Delta^k}.\end{aligned}$$

By the induction hypothesis that $\left|R_{t+1}^0 - R_{t+1}^1\right| \ge \frac{1}{2}(\frac{1}{\Delta^k})^{\ell-t-2}$, we can prove (8) for t. This finishes the induction step for $1 \le t \le \ell - 3$.

Finally, we use (8) to bound $|R_0^0 - R_0^1|$. The proof is similar to the proof in the induction step. The only difference is that the recursion for root v becomes $g(x) = \lambda\left(\frac{1}{1+x}\right)^{\Delta}$. By a similar

calculation, there exists $\min(R_1^0, R_1^1) < \theta < \max(R_1^0, R_1^1)$ such that

$$\begin{aligned}|R_0^0 - R_0^1| = |g'(\theta)| \cdot \left|R_1^0 - R_1^1\right| &\geq \Delta\lambda\left(\frac{1}{1+\theta}\right)^{\Delta+1} \cdot \frac{1}{2}\left(\frac{1}{\Delta^k}\right)^{\ell-2} \\ &= \frac{\Delta}{(\Delta-1)(1+\theta)} \cdot \lambda(\Delta-1)\left(\frac{1}{1+\theta}\right)^{\Delta} \cdot \frac{1}{2}\left(\frac{1}{\Delta^k}\right)^{\ell-2} \\ &\geq \frac{\Delta}{(\Delta-1)(1+\lambda)} \cdot \frac{1}{2}\left(\frac{1}{\Delta^k}\right)^{\ell-1} \geq \frac{1}{3}\left(\frac{1}{\Delta^k}\right)^{\ell-1}.\end{aligned}$$

By the definitions of R_0^0 and R_0^1 and the fact $\Delta^k \geq 4$, we have

$$\begin{aligned}d_{\mathrm{TV}}\left(\mu_v^{\sigma_0}, \mu_v^{\sigma_1}\right) = \left|\mu_v^{\sigma_0}(1) - \mu_v^{\sigma_1}(1)\right| &= \mu_v^{\sigma_0}(0)\mu_v^{\sigma_1}(0)|R_0^0 - R_0^1| \\ &\geq \left(\frac{1}{1+\lambda}\right)^2 \cdot \frac{1}{3}\left(\frac{1}{\Delta^k}\right)^{\ell-1} \geq \frac{4}{27}\left(\frac{1}{\Delta^k}\right)^{\ell-1} \geq \frac{1}{2}\left(\frac{1}{\Delta^k}\right)^{\ell}.\end{aligned}$$

■

C. Grid graph with quadratic-sized boundary

Although the distance-n boundary of the $\mathbb{Z}^2$ lattice contains only $4n$ vertices, there are subgraphs that blow this number up to $\Omega(n^2)$. Below is one such graph constructed recursively. For large enough even n, the graph $G(n)$ is given by Figure 3.

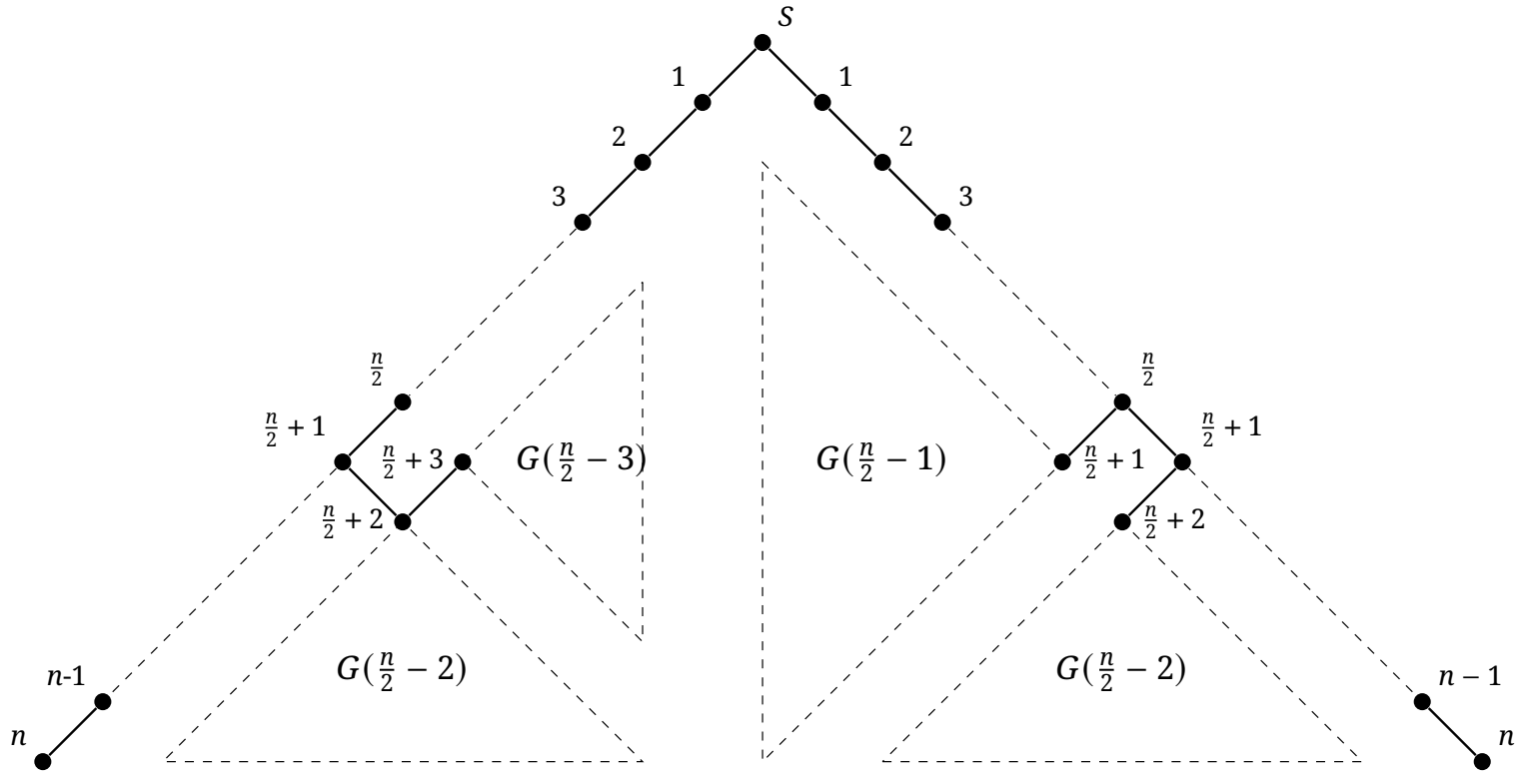


Figure 3. The graph $G(n)$ when n is even. The number next to a vertex indicates its distance from the starting vertex S.

The odd-n case can be constructed similarly. Let $f(n)$ be the number of distance-n vertices from the vertex S in the graph $G(n)$. Then $f(n) = 4f(\frac{n}{2} - \Theta(1)) + \Theta(1)$. By the Master Theorem, $f(n) = \Theta(n^2)$. Also note that such a construction can be made on an induced subgraph of $\mathbb{Z}^2$, by splitting each edge here into two edges joined by a vertex.