The Multilevel Splitting Algorithm for Graph Coloring with Application to the Potts Model

Radislav Vaisman[a], Matthew Roughan[b], and Dirk P. Kroese[a]

[a]School of Mathematics and Physics, The University of Queensland Brisbane 4072, Australia
[b]School of Mathematical Sciences, The University of Adelaide Adelaide SA 5005, Australia

ARTICLE HISTORY
Compiled March 13, 2017

ABSTRACT
Calculating the partition function of the zero-temperature antiferromagnetic model is an important problem in statistical physics. However, an exact calculation is hard, since it is strongly connected to a fundamental combinatorial problem of counting proper vertex colorings in undirected graphs, for which an efficient algorithm is not known to exist. Thus, one has to rely on approximation techniques. In this paper we formulate the problem of the partition function approximation in terms of rare-event probability estimation and investigate the performance of a particle-based algorithm, called Multilevel Splitting, for handling this setting. The proposed method enjoys a provable probabilistic performance guarantee and our numerical study indicates that this algorithm is capable of delivering accurate results using a relatively modest amount of computational resources.

KEYWORDS
Partition Function; Graph Coloring; Multilevel Splitting; Rare Events

1. Introduction

Calculating the zero-temperature antiferromagnetic partition function is a fundamental problem in statistical physics, arising in the well-known general Potts model [1], which studies the behavior of ferromagnets and other phenomena of solid-state physics, and has been extensively explored in statistical physics [2–5], theoretical computer science and mathematics [6, 7], signal processing [8, 9], modelling of financial markets [10], biology [11], and social networks [12, 13].

To study the Potts model partition function [4], one should realize that it corresponds to a difficult combinatorial counting problem, namely calculating the Tutte polynomial [14]. In this paper we study the Potts model under the zero-temperature regime, for which the partition function is given by the Tutte polynomial related formula called the chromatic polynomial which counts the number of proper graph colorings as a function of the number of colors [15]. In Section 2, we show that knowing the chromatic polynomial, is equivalent to the calculation of the zero-temperature partition function.

The chromatic polynomial and the corresponding coloring problem belong to the well-known list of Karp's 21 NP-complete problems [16]. The proper graph coloring

CONTACT Radislav Vaisman. Email: r.vaisman@uq.edu.au

counting problem lies in the #P complexity class introduced by Valiant [17]. An exact polynomial time solution of a #P problem will imply P = NP [18], so the best we can reasonably hope for, is to find a good approximation technique. One of such techniques, called fully polynomial randomized approximation scheme (FPRAS), has been developed for approximate counting of some #P problems [19–21], but there are also many negative results [18, 22]. In particular, the inapproximability of the chromatic polynomial for a general case was established in [23] and [24].

There exists two basic approaches to tackle #P counting problems: Markov Chain Monte Carlo (MCMC) and Sequential Importance Sampling (SIS). These approximate counting algorithms exploit the fact that counting is equivalent to uniform sampling over a suitably restricted set [25]. The main idea when using the MCMC method, is to construct an ergodic Markov chain with limiting distribution which is equal to the desired uniform distribution. The MCMC methods plays a central role in counting approximations, [26–29], but SIS algorithms have their own merits [19, 30–33].

In view of the above discussion, we propose to apply an adaptive approach called Multilevel Splitting (MS) [27]. This powerful concept was first used by Kahn and Harris [34] for rare-event probability estimation. Their revolutionary idea can be summarized as follows. Given a state space, partition it in such a way that the problem becomes one of estimating conditional probabilities that are not rare, thus taking essentially an sequential approach, where a particle-based population (sample), is going through a mutation process to increase its performance and move to a rare regions of the state space.

The MS particle-based method is often applied to rare-event probability estimation, which makes it an appropriate candidate for estimating the number of proper graph colorings, as these are rare in the set of all colourings. Consequentially, it is natural to apply MS to estimating the zero-temperature Potts partition function. The paper contribution is described below.

(1) This paper develops the required adaptation of the MS algorithm for the coloring problem. In particular we define a performance function, that is shown to successfully handle the Potts model counting problem. The Gibbs sampler for this particular performance is also shown to be efficient from both practical (the ability to handle real physical grid models), and computational points of view. To the best of our knowledge, there is currently no other single-threaded method that is capable of handling graphs of few hundreds of vertices and edges within reasonable time.
(2) Similar to [35], we show that MS has provable probabilistic performance guarantees. Namely, a relatively cheap (from a computational point of view) lower bound can be obtained, thus introducing an additional merit.
(3) We develop a freely available research software package called ChromSplit to support our findings. Moreover, our method can be easily parallelized, and applied for many other problems.

In addition, we show that the proposed counting procedure, combined with simple binary search, can be easily adjusted to solve the important combinatorial optimization problem of finding a graph chromatic number. The MS method, allows us to perform an analysis of chromatic number statistics in small-world networks [36–39]. In particular, we present numerical evidence that two well-known classes of graph models, Gilbert-Erdős-Rényi (GER) [40, 41] and Watts-Strogatz (WS) [36], exhibit very similar scaling behavior of the chromatic number with respect to the average node degree, particularly

for sparse graphs. We show that in this case, and regardless of the random model parameters, the change in structure between the different graphs has no significant effect on the chromatic number!

Organization

The rest of the paper is organized as follows. In Section 2 we formulate the zero-temperature antiferromagnetic partition-function approximation problem and explain its connection to vertex coloring and the corresponding rare-event probability estimation problem. In Section 3 we give a brief introduction to the MS algorithm, show how it can be applied to graph coloring problems – both counting and optimization – and describe a set of probabilistic performance guarantees. We report our numerical findings in a detailed experimental study in Section 4, and present new results for the chromatic numbers of random graphs in Section 5. Finally, in Section 6 we summarize our findings and discuss possible directions for future research.

2. Problem Formulation

2.1. *Proper vertex coloring*

We start with the description of the (proper) vertex-coloring problem.

Definition 2.1 (Vertex coloring)**.** Given a finite undirected graph $G = (V, E)$ with vertex set V and edge set E, a proper vertex coloring of the graph's vertices with colors $\{1, \ldots, q\}$, $q \in \mathbb{N}$, (also called a q-coloring), is such that no two vertices that share an edge, have the same color.

In this paper we will be interested in the vertex-coloring counting problem. Namely, consider the number of different q-colorings of a given graph G as a function of q and denote this number by $\mathrm{chr}(G, q)$. It is well-known that the following is true [15].

(1) $\mathrm{chr}(G, q)$ is a polynomial of q.
(2) The degree of $\mathrm{chr}(G, q)$ is equal to the number of vertices in G.
(3) $\chi(G) = \min\{q : \mathrm{chr}(G, q) > 0\}$ is called the chromatic number and it stands for the smallest positive integer that is not a root of the chromatic polynomial. This number is equal to the minimal number of colors required to properly color a given graph.
(4) Using a deletion-contraction recursive algorithm, the chromatic polynomial can be computed in $\mathcal{O}\left(\varphi^{|V|+|E|}\right)$ time, where $\varphi = (1 + \sqrt{5})/2$.
(5) It is known now that there is no FPRAS for computing $\mathrm{chr}(G, q)$ for any $q > 2$, unless NP = RP holds [24], (where RP stands for randomized polynomial time).

There exist some graph topologies with known chromatic polynomial. For example, Figure 1 shows the polynomial of a small complete graph; but in general the polynomial is hard to calculate.

For completeness, we next give a brief review of the zero-temperature Potts model partition function and establish its correspondence to the proper vertex coloring problem. For an extensive overview, we refer to [42].

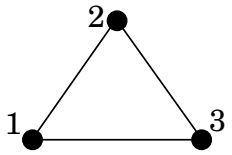


Figure 1. The complete graph $K_3 = (\{1,2,3\}, \{(1,2),(2,3),(3,1)\})$. By inspection, it is easy to verify that the chromatic polynomial of K_3 is equal to $\mathrm{chr}(K_3, q) = q(q-1)(q-2)$. For example, the number of proper colorings using three colors is equal to 6, *i.e.,* $\mathrm{chr}(K_3, 3) = 6$. In this case, the chromatic number of this graph is 3. That is, one will not be able to construct a proper coloring with one or two colors.

2.2. *Zero-Temperature Potts model*

In this paper we consider the Fortuin-Kasteleyn representation of the Potts Model [43]. In this setting, a finite undirected graph $G = (V, E)$ with the vertex set V and edge set E, represents physical particles and their interactions, respectively. In particular, the q-state Potts model is defined as follows. Let $q \in \mathbb{N}$ be a natural number and define $\boldsymbol{\sigma} = \{\sigma_v\}_{v \in V}$ to be a set of spins, where $\sigma_v \in \{1, \ldots, q\}$. Define $\{\mathcal{J}_e\}_{e \in E}$ to be a set of coupling constants such that for each $e = (u, v)$, $\mathcal{J}_e$ determines the interaction strength between u and v. The Hamiltonian (energy function) is defined by

$$\mathcal{H}(\boldsymbol{\sigma}) = -\sum_{e=(u,v)\in E} \mathcal{J}_e\, \delta(\sigma_u, \sigma_v),$$

where δ is the Kronecker delta, which equals one whenever $\sigma_u = \sigma_v$ and zero otherwise. The corresponding partition function is

$$\begin{aligned}
\mathcal{Z}_G(q) &= \sum_{\boldsymbol{\sigma}} \mathrm{e}^{-\frac{1}{\mathrm{kT}}\mathcal{H}(\boldsymbol{\sigma})} \\
&= \sum_{\boldsymbol{\sigma}} \mathrm{e}^{\varrho \sum_{e=(u,v)\in E} \mathcal{J}_e \delta(\sigma_u,\sigma_v)} = \sum_{\boldsymbol{\sigma}} \prod_{e=(u,v)\in E} \mathrm{e}^{\varrho \mathcal{J}_e \delta(\sigma_u,\sigma_v)} \\
&= \sum_{\boldsymbol{\sigma}} \prod_{e=(u,v)\in E} \left[1 + \left(\mathrm{e}^{\varrho \mathcal{J}_e} - 1\right) \delta(\sigma_u, \sigma_v)\right] \\
&= \sum_{\boldsymbol{\sigma}} \prod_{e=(u,v)\in E} \left(1 + \vartheta_e \delta(\sigma_u, \sigma_v)\right),
\end{aligned}$$

where k and T stand for Boltzmann's constant and the temperature, respectively, $\varrho = 1/\mathrm{kT}$, and $\vartheta_e = \mathrm{e}^{\varrho \mathcal{J}_e} - 1$.

Definition 2.2. In the Potts model a coupling $\{\mathcal{J}_e\}_{e \in E}$ is

- *ferromagnetic* if $\mathcal{J}_e \geq 0$ for all $e \in E$; and
- *antiferromagnetic* if $\mathcal{J}_e \leq 0$ for all $e \in E$.

The zero-temperature antiferromagnetic Potts model is defined in the limit $\varrho \to \infty$ to have $\vartheta_e = \mathrm{e}^{\varrho \mathcal{J}_e} - 1 = -1$ for all $e \in E$. In the zero-temperature regime we therefore have $\vartheta_{(u,v)} \delta(\sigma_u, \sigma_v) = -\mathbb{1}_{\{\sigma_u = \sigma_v\}}$. From the above equation, the expression

$$\mathcal{Z}_G(q) = \sum_{\boldsymbol{\sigma}} \prod_{e=(u,v)\in E} (1 + \vartheta_e \delta(\sigma_u, \sigma_v)) = \sum_{\boldsymbol{\sigma}} \prod_{(u,v)\in E} \mathbb{1}_{\{\sigma_u \neq \sigma_v\}},$$

is thus the partition function in question. Moreover, we now see that $\mathcal{Z}_G(q)$ is the number of proper graph colorings with q colors, since the inner product is equal to 1 only if the adjacent vertices have different spins (colors), and the outer summation passes through all possible graph colorings.

We proceed by reducing these graph counting problems to the problem of estimating the probability of randomly selecting a proper coloring.

2.3. *The probabilistic set-up*

For any $G = (V, E)$ there are $q^{|V|}$ colorings. Suppose that q is greater than or equal to G's chromatic number. Then, some of these colorings are proper. Consider now the uniform distribution on the set of all color assignments $\{1, \ldots, q\}^{|V|}$ and let $\mathbf{X} = \{X_1, \ldots, X_{|V|}\}$ be a uniform random assignment from $\{1, \ldots, q\}^{|V|}$. Let ℓ be the probability that a random assignment is a proper coloring. Then, $\mathcal{Z}_G(q) = q^{|V|}\ell$. That is, the estimation of $\mathcal{Z}_G(q)$ and ℓ is interchangeable.

Exact calculation of ℓ is as hard as calculating $\mathcal{Z}_G(q)$, but one could consider estimation of ℓ. The Crude Monte Carlo (CMC) procedure for the estimation of ℓ is straightforward and is summarized in Algorithm 1.

Algorithm 1 The CMC Algorithm for estimating ℓ

Input: A graph $G = (V, E)$, $q \in \mathbb{N}$, and a sample size $N \in \mathbb{N}$.
Output: Unbiased estimator of ℓ.

1: **for** $t = 1$ **to** N **do**
2: **Random coloring generation**: For each $v \in V$, assign a color from the $\{1, \ldots, q\}$ set, uniformly at random. Denote this coloring by $\mathbf{X}_t = \left\{X_1^{(t)}, \ldots, X_{|V|}^{(t)}\right\}$.
3: **Verify proper coloring**: If $\mathbf{X}_t$ is a proper coloring of G, that is, if

$$\prod_{(u,v)\in E} \mathbb{1}_{\left\{X_u^{(t)} \neq X_v^{(t)}\right\}} = 1,$$

set $Y_t = 1$; otherwise, set $Y_t = 0$.
4: **end for**
5: **return**:

$$\hat{\ell}_{\mathrm{CMC}} = \frac{1}{N}\sum_{t=1}^{N} Y_t. \tag{1}$$

2.4. *Rare-event Monte Carlo*

Unfortunately, Algorithm 1 will generally fail because of the rare-event setting. To see this, consider a relatively small complete graph K_{30} with 30 vertices. Suppose that we wish to color it with $q = 30$ colors. We can now calculate the exact ℓ value, since the first graph vertex can be colored with 30 colors, the second with 29 colors and so on. That is, there are 30! proper colorings. Having in mind that there are 30^{30} available colorings, we arrive at $\ell = 30!/30^{30} \approx 1.29 \times 10^{-12}$. This probability is so small that

during Algorithm 1 execution, the random variable $Y_t = 0$ almost certainly for all t, returning $\hat{\ell}_{\text{CMC}} = 0$, even for a large sample size like $N = 10^9$.

To better understand this rare-event phenomenon, consider the following general setting. Let $\mathcal{X}$ and $\mathcal{X}^* \subseteq \mathcal{X}$ be sets and suppose that the probability $\ell = |\mathcal{X}^*|/|\mathcal{X}|$ is to be estimated. In our context $\mathcal{X}$ and $\mathcal{X}^*$ stand for the set of all graph's colorings and the proper ones, respectively. Then, the CMC estimator $\hat{\ell}_{\text{CMC}}$ is given in (1). In particular, the $\{Y_t\}$ are Independent Identically Distributed (IID) Bernoulli Random Variables (RV). Hence, $\hat{\ell}_{\text{CMC}}$ is an unbiased estimator of ℓ, with known variance

$$\text{Var}\left(\hat{\ell}_{\text{CMC}}\right) = \frac{\ell(1-\ell)}{N}. \tag{2}$$

We next consider the accuracy and hence efficiency of the rare-event estimator. In this paper we will use two measures of accuracy: the relative experimental error (RER) [44], given by $\text{RER} = \left|\hat{\ell}_{\text{CMC}} - \ell\right| \cdot \ell^{-1}$, and the relative error (RE). The former measures error relative to the correct value, the latter considers the standard deviation relative to their mean. Both are useful: RER is preferred, but requires knowledge of the true value, which is not generally available (see [45] and [46] for details); on the other hand, the RE can be estimated and provide a confidence interval. The RE of $\hat{\ell}_{\text{CMC}}$ is

$$\text{RE}\left(\hat{\ell}_{\text{CMC}}\right) = \frac{\sqrt{\text{Var}\left(\hat{\ell}_{\text{CMC}}\right)}}{\mathbb{E}\left(\hat{\ell}_{\text{CMC}}\right)} \underbrace{=}_{(2)} \frac{\sqrt{\ell(1-\ell)/N}}{\ell}.$$

In the rare-event setting $\ell \ll 1$ so

$$\text{RE}\left(\hat{\ell}_{\text{CMC}}\right) \approx 1/\sqrt{N\ell}, \tag{3}$$

which imposes a serious challenge. To see this, consider the rare-event probability $\ell \approx 10^{-12}$, and suppose that we are interested in a modest 10% RE. It is easy to verify from (3), that the required number of experiments N is about 10^{14}.

The probabilities that concern us here are as small as 10^{-157}, so we have no hope of applying the naive CMC approach. We overcome this problem using the MS method for rare-event probability estimation and optimization, which we describe in the following section.

3. Multilevel Splitting

Here we adopt a quite general adaptive variance minimization technique called the MS algorithm [27]. The latter is essentially a particle-based method for rare-event probability estimation and optimization.

Careful examination of the CMC procedure reveals that the major problem is the rareness of samples from the set $\mathcal{X}^*$. The main idea of MS is to design an adaptive sequential sampling plan, with a view to decompose a "difficult" problem (sampling from $\mathcal{X}^*$), into a number of "easy" ones associated with a sequence of subsets in the sampling space $\mathcal{X}$. A general MS framework is summarized in Figure 2. The resulting MS algorithm, provides an unbiased estimator of ℓ [27], and is given in Algorithm 2.

Find a sequence of sets $\mathcal{X} = \mathcal{X}_0 \supseteq \mathcal{X}_1 \supseteq \cdots \supseteq \mathcal{X}_T = \mathcal{X}^*$. Assume that the subsets $\mathcal{X}_t$ can be written as level sets of some fitness function $S : \mathcal{X} \to \mathbb{R}$ for levels

$$\infty = \gamma_0 \geq \gamma_1 \geq \cdots \geq \cdots \geq \gamma_T = \gamma,$$

such that $\mathcal{X}_t = \{\mathbf{x} \in \mathcal{X} \ : \ S(\mathbf{x}) \leq \gamma_t\}$ for $t = 0, \ldots, T$.

Note that the quantity of interest ℓ is given by the telescopic product:

$$\ell = \frac{|\mathcal{X}^*|}{|\mathcal{X}|} = \prod_{t=1}^{T} \frac{|\mathcal{X}_t|}{|\mathcal{X}_{t-1}|} = \prod_{t=1}^{T} \mathbb{P}\left(S\left(\mathbf{X}\right) \leq \gamma_t \mid S\left(\mathbf{X}\right) \leq \gamma_{t-1}\right) = \mathbb{P}\left(S(\mathbf{X}) \leq \gamma_T\right).$$

For each $t = 1, \ldots, T$, develop an efficient estimator $\widehat{c}_t$ for the conditional probabilities

$$c_t = \mathbb{P}\left(S(\mathbf{X}) \leq \gamma_t \mid S(\mathbf{X}) \leq \gamma_{t-1}\right).$$

To avoid rare-event problems at the intermediate levels (γ_t), we assume that the sets $\mathcal{X}_t$, $t = 1, \ldots, T$, are specifically designed such that the $\{c_t\}$ are not rare-event probabilities.

Deliver

$$\widehat{\ell} = \prod_{t=1}^{T} \widehat{c}_t,$$

as an estimator of ℓ.

Figure 2. General Multilevel Splitting framework.

Let us now put the coloring problem into the MS framework. It is straightforward to define the $\mathcal{X}$ and the $\mathcal{X}^*$ sets as the set of all colorings and the set of proper colorings, respectively. Note that $\mathcal{X} \supseteq \mathcal{X}^*$ and that it is easy to sample uniformly from $\mathcal{X}$ by choosing one out of q available colors for each vertex with equal probability $1/q$. On the other hand, defining the performance function $S : \mathcal{X} \to \mathbb{R}$ and the corresponding sets $\mathcal{X}_0, \cdots, \mathcal{X}_T$ can be a delicate task. There are quite a few possibilities, but in this paper we propose that given a coloring $\mathbf{x} \in \mathcal{X}$, we define $S(\mathbf{x})$ to be *the number of adjacent vertices that share the same color*. Note that by definition (in Figure 2), it holds that $\mathcal{X} = \{\mathbf{x} \in \mathcal{X} \ : \ S(\mathbf{x}) \leq \infty\}$, and $\mathcal{X}^* = \{\mathbf{x} \in \mathcal{X} \ : \ S(\mathbf{x}) \leq 0\}$.

Remark 1 (Choosing alternative performance function)**.** The choice of this particular function is motivated by its efficient computational performance (see Proposition 3.1), and by our numerical finding in Section 4, which suggest that the proposed performance is sufficient for physical grid models. Namely, for this performance function, the conditional probabilities $\{c_t\}$ are not rare.

We can estimate c_t using N samples (particles) chosen uniformly at random from the $\mathcal{X}_{t-1}$ set. Namely, the estimate $\hat{c}_t$ is equal to the number of these samples that fall into $\mathcal{X}_t$ divided by N. We achieve this uniform sampling using MCMC, in particular, a Gibbs Sampler; see [27] and [32]. The resulting algorithm is described in Algorithm 2.

Algorithm 2 The MS Algorithm for estimating ℓ

Input: A graph $G = (V, E)$, a sequence of levels $\gamma_1, \ldots, \gamma_T$, a performance function $S : \mathcal{X} \to \mathbb{R}$, and a sample size $N \in \mathbb{N}$.

Output: Unbiased estimator of ℓ.

1: **Initialization**: Generate N independent particles $\mathcal{W}_0 = \{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$ uniformly from $\mathcal{X}$. Let $\mathcal{W}_1 \subseteq \mathcal{W}_0$ be the subset of elements $\mathbf{X}$ in $\mathcal{W}_0$ for which $S(\mathbf{X}) \leq \gamma_1$ holds, (that is, $\mathcal{W}_1$ is an elite population of particles), and let N_1 be the size of $\mathcal{W}_1$.
2: **for** $t = 1$ **to** $T - 1$ **do**
3: Draw $K_i \sim \mathsf{Bernoulli}(0.5)$, for $i = 1, \ldots, N_t$, such that $\sum_{i=1}^{N_t} K_i = N \bmod N_t$. The latter can be accomplished as follows. First, generate an $(N \bmod N_t)$-cardinality subset $I \subseteq \{1, \ldots, N_t\}$, $|I| = (N \bmod N_t)$, uniformly at random from the set of all $(N \bmod N_t)$-cardinality subsets. Then, for $i = 1, \ldots, N_t$, set $K_i = 1$ if $i \in I$, and $K_i = 0$ otherwise.
4: **for** $i = 1$ **to** N_t **do**
5: Set a splitting factor $S_{ti} = \left\lfloor \frac{N}{N_t} \right\rfloor + K_i$, where $\lfloor x \rfloor$ is the floor function.
6: Set $\mathbf{Y}_{i,0} = \mathbf{X}_i$.
7: **for** $j = 1$ **to** S_{ti} **do**
8: draw $\mathbf{Y}_{i,j} \sim \kappa_t(\mathbf{y} \mid \mathbf{Y}_{i,j-1})$, where $\kappa_t(\mathbf{y} \mid \mathbf{Y}_{i,j-1})$ is a Markov transition density whose stationary distribution is the uniform distribution on $\mathcal{X}_t$ (we describe this momentarily).
9: **end for**
10: **end for**
11: Set the population $\mathcal{V}_t = \{\mathbf{Y}_{1,1}, \ldots, \mathbf{Y}_{1,S_{t1}}, \cdots, \mathbf{Y}_{N_t,1}, \ldots, \mathbf{Y}_{N_t,S_{tN_t}}\}$. Note that $\mathcal{V}_t$ contains N elements.
12: Let $\mathcal{W}_{t+1} \subseteq \mathcal{V}_t$ be the subset of elements of $\mathcal{V}_t$ for which $S(\mathbf{X}) \leq \gamma_{t+1}$, and let N_{t+1} be the size of new population $\mathcal{W}_{t+1}$.
13: **Estimation**: Set $\hat{c}_t = N_{t+1}/N$.
14: **end for**
15: **return**: $\hat{\ell} = \prod_{t=1}^{T} \hat{c}_t$.

If at any stage in Algorithm 2 we get $N_t = 0$, then the algorithm stops and we return $\hat{\ell} = 0$, so it is important to choose the thresholds $\{\gamma_t\}$ such that this occurs rarely.

Efficiency of the MS Algorithm

Any successful application of the MS framework depends on the values of the conditional probabilities $\{c_t\}$ being not too small, since their estimation is performed via CMC (line 13 of Algorithm 2). These values are determined by the corresponding performance function. However, we cannot expect a single performance function to work well on all models, since the coloring problem is hard [24]. In this paper, for example, the MS algorithm failed to identify the correct chromatic number of 6 out of 31 benchmark instances in Section 4.3. Choosing a good performance function is not easy, and one should find a balance between flexibility and computational efficiency. For example, a costly but also very promising approach, which uses a continuous performance functions to the satisfiability problems [47], was introduced in [48].

Markov transition step

The Markov transition step of Algorithm 2 is crucial. We must be able to generate a Markov chain described by transition density κ_t, but this can be expensive in the sense of computational time. We use a Gibbs sampler for the coloring problem, described in Algorithm 3, and show that it is efficient in Proposition 3.1.

Algorithm 3 The Gibbs sampler for sampling uniformly from the population $\mathcal{X}_t$, given a starting point in that set.

Input: A graph $G = (V, E)$, the number of available colors q, an element $\mathbf{X} = \left(X_1, \ldots, X_{|V|}\right)$ with $X_i \in \{1, \ldots, q\}$ from the set $\mathcal{X}_t$, and the corresponding threshold value γ_t.

Output: $\tilde{\mathbf{X}}$ distributed approximately uniformly on the set $\mathcal{X}_t$.

1: **for** $i = 1$ **to** $|V|$ **do**
2: Set $A = \{X_i\}$.
3: **for** $c \in \{1, \ldots, q\} \setminus \{X_i\}$ **do**
4: **if** $S\left((\tilde{X}_1, \ldots, \tilde{X}_{i-1}, c, \ldots, X_{|V|})\right) \leq \gamma_t$ **then**
5: $A = A \cup \{c\}$.
6: **end if**
7: **end for**
8: Choose $\tilde{X}_i$ uniformly at random from A; that is, an element of A is chosen with probability $1/|A|$.
9: **end for**
10: **return** $\tilde{\mathbf{X}} = \left(\tilde{X}_1, \ldots, \tilde{X}_{|V|}\right)$.

Proposition 3.1 (Computational complexity of Algorithm 3)**.** *For the coloring problem, Algorithm 3 can be performed in* $\mathcal{O}(q\,|E|)$ *time.*

Proof. We first note that the conditional sampling steps of Algorithm 3, (lines 3–7), governs the running time of each cycle of the main *for* loop, and, one cycle is performed for each $v \in V$. To complete the proof, it remains to show that for each $v \in V$ this step can be performed in $\mathcal{O}(q\,d_v)$ time, where d_v stands for the degree of v, since it holds that:

$$\sum_{v \in V} \mathcal{O}(q\,d_v) = \mathcal{O}(q) \sum_{v \in V} d_v = \mathcal{O}(q) 2|E| = \mathcal{O}(q\,|E|)\,.$$

Let us consider the actual sampling of $\tilde{X}_i$ from the uniform conditional density

$$\mathsf{U}\left(x_i \mid \tilde{X}_1, \ldots, \tilde{X}_{i-1}, X_{i+1}, \ldots, X_{|V|}\right).$$

Given the assignment of colors $(\tilde{X}_1, \ldots, \tilde{X}_{i-1}, X_{i+1}, \ldots, , \ldots, X_{|V|})$, all we need to do, is to check which color can be assigned to $\tilde{X}_i$, such that the performance function satisfies

$$S(\tilde{X}_1, \ldots, \tilde{X}_{i-1}, \tilde{X}_i, X_{i+1}, \ldots, X_{|V|}) \leq \gamma_t,$$

and pick one of these colors uniformly at random. It turns out that this operation can

be performed in $\mathcal{O}(q\, d_{v_i})$ time. Consider the sampling step and suppose that

$$S\left(\tilde{X}_1, \ldots, \tilde{X}_{i-1}, X_i, X_{i+1}, \ldots, X_{|V|}\right) = \gamma_t' \leq \gamma_t$$

holds. Now, only the variable X_i is subject to change, so this change is localized to the vertex v_i and its immediate neighborhood of adjacent vertices. In particular, for any given color from $\{1, \ldots, q\}$, we can determine the new performance value by checking v_i's neighbors and counting the ones that are colored with the same color. Let $\mathrm{Ng}(v_i, c)$ be a set of vertices adjacent to v_i such that each $v \in \mathrm{Ng}(v_i, c)$ is colored with color c. It is not very hard to verify now that for any $c \in \{1, \ldots, q\}$ it holds that

$$S\left(\left(\tilde{X}_1, \ldots, \tilde{X}_i = c, \ldots, X_{|V|}\right)\right) = \gamma_t' - 2\left|\mathrm{Ng}(v_i, X_i)\right| + 2\left|\mathrm{Ng}(v_i, c)\right|. \tag{4}$$

Noting that for any $c \in \{1, \ldots, q\}$ the computation of $\mathrm{Ng}(v_i, c)$ can be done in $\mathcal{O}(d_{v_i})$ time, and having in mind that we need to perform it for every color, we arrive at $\mathcal{O}(q\, d_{v_i})$ time complexity and complete the proof. □

The efficient performance calculation approach discussed above, is demonstrated in Example 3.2.

Example 3.2 (Efficient calculation of $S(\cdot)$ in Algorithm 3)**.** Consider the simple graph in Figure 3, and suppose that we have two colors, white (w) and gray (g).

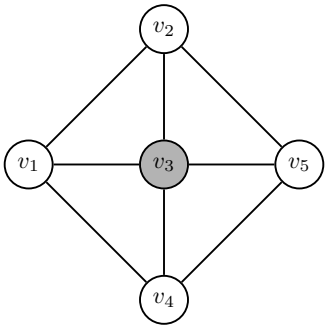


Figure 3. A simple graph with performance $\gamma = 8$.

Note that the v_3 vertex is g-colored and the rest of the vertices are w-colored, so it is not hard to calculate the overall performance $S(c(v_1), \ldots, c(v_5)) = S(w, w, g, w, w) = 8$, since each v_1, v_2, v_4 and v_5 have exactly two neighbors that are colored with the same white color. Suppose now, that we would like to change the v_1 vertex color to gray and calculate the new performance value γ', which is required in line 4: of Algorithm 3. Using (4), we arrive at

$$\gamma' = \gamma - 2\left|\mathrm{Ng}(v_1, w)\right| + 2\left|\mathrm{Ng}(v_1, g)\right| = 8 - 2 \cdot 2 + 2 \cdot 1 = 6,$$

since $\mathrm{Ng}(v_1, w) = 2$ and $\mathrm{Ng}(v_1, g) = 1$. Note that during the performance value update, we only considered v_1's neighbors; that is, we did not consider the vertex v_5.

We next proceed with a clarification for a few remaining technical issues regarding the MS Algorithm 2.

The MS algorithm parameters

Algorithm 2 enjoys a property of having a relatively small number of parameters: the sample (population) size N and the intermediate performance levels $\gamma_1, \ldots, \gamma_T$. Choosing N is relatively easy, as we show in Section 4. Namely, for the coloring problem, our numerical study indicates that setting $N = 10|E|$ is sufficient. Choosing the levels $\{\gamma_t\}$ is not as easy. We run the MS algorithm multiple times to obtain statistics of the results, so $\gamma_1, \ldots, \gamma_T$ should be fixed in advance, in order to ensure an estimator's unbiasedness [27]. However, we also wish to adapt the chosen thresholds to the particular problem.

A common method to resolve this problem is perform a single *pilot run* of Algorithm 2 using a so-called rarity parameter ρ. The rarity parameter specifies the percentage of the population $\mathcal{V}_t$ elements that will progress to $\mathcal{W}_t$, and works by taking the ρth order statistic of $S(\cdot)$ of the elements of $\mathcal{V}_t$ to define γ_t. This proceeds until $t = T$ such that $\gamma_T \geq \gamma$.

The pilot run establishes a set of threshold values adapted to the problem at hand, with approximately the same reduction in the size of the sets $\mathcal{X}_t$ at each step (until the last). We then use the pilot values of population's performance in subsequent runs. See [27] for further details.

Our numerical study showed that for this problem a value of $\rho = 20\%$ resulted in a good tradeoff between the number of outer loops T, (the main *for* loop of Algorithm 2), and the work required inside the loops.

Time complexity of the MS algorithm

The initialization of the MS Algorithm 2 can be performed in $\mathcal{O}\left(N|V|q\right)$ time because we generate N samples each with $|V|$ elements, and for each of these we evaluate $S(\cdot)$.

The MCMC step is performed $\mathcal{O}(T)$ times, with cost given in Proposition 3.1 to be $\mathcal{O}\left(N|E|q\right)$. Hence, the overall complexity is equal to

$$\mathcal{O}\big(N|V|q\big) + \mathcal{O}\big(TN|E|q\big) = \mathcal{O}\big(Nq\left(|V| + T|E|\right)\big). \tag{5}$$

Clearly, the complexity depends on the number of levels T. There exists versions of the Splitting Algorithm 2 where one can bound this value. For example, the splitting version in [48] satisfies

$$T = \left\lfloor \frac{\ln \ell}{\ln \rho} \right\rfloor. \tag{6}$$

However, this T value depends on the rare-event probability ℓ, so one should at least know ℓ's order of magnitude to get a meaningful complexity bound.

The MS algorithm error

We saw that Algorithm 2 outputs an unbiased estimator of the rare-event probability ℓ. A common practice is to repeat this algorithm for R independent replications, obtain

independent unbiased estimators $\hat{\ell}_1, \ldots, \hat{\ell}_R$ and report the average:

$$\hat{\ell} = R^{-1} \sum_{r=1}^{R} \hat{\ell}_r.$$

Under this setting, we can calculate the RER via

$$\text{RER} = \left| \hat{\ell} - \ell \right| \cdot \ell^{-1},$$

and estimate the relative error via

$$\text{RE} = \sqrt{\frac{\frac{1}{R-1} \sum_{r=1}^{R} \left(\hat{\ell}_r - \hat{\ell} \right)^2}{\hat{\ell}^2}}.$$

As mentioned earlier, the RER is rarely available, since our task is to calculate ℓ. However, this measure is useful for benchmarking the performance of the MS algorithm on instances for which we know the analytical value. See the first example in Section 4.

Finding a graph chromatic number via the MS algorithm

The adaptation of the MS Algorithm 2 to a method for determining a graph chromatic number is straightforward. Recall that the MS algorithm counts the number of q-colorings in a graph. Let $\Delta(G)$ be the maximal degree of the graph $G = (V, E)$, and recall that G can be colored with at least one and with at most $\Delta(G) + 1 \leq |V|$ colors via greedy coloring. With this in mind, we propose to apply a binary search procedure on $q = 1, \ldots, |V|$. The above idea is summarized in Algorithm 4.

Algorithm 4 Binary search algorithm for finding a graph's chromatic number

Input: A graph $G = (V, E)$, a sequence of thresholds $\gamma_1, \ldots, \gamma_T$, a performance function $S : \mathcal{X} \to \mathbb{R}$, and population size $N \in \mathbb{N}$.

Output: Chromatic number approximation of the $G = (V, E)$ graph.

Set $low = 0$ and $high = |V|$.
while $low < high$ **do**
  Set $mid = \lceil low + high \rceil / 2$.
  Let $\hat{\ell}_{mid}$ be the estimated number of proper mid-colorings of G which is obtained using the MS Algorithm 2.
  if $\hat{\ell}_{mid} > 0$ **then**
    $high = mid$.
  else
    $low = mid$.
  end if
end while
return mid as an approximation of G's chromatic number.

Algorithm 4 is relatively cheap in its running time. In particular, recalling that $q \leq \Delta(G) + 1 \leq |V|$, noting that Algorithm 4 will execute the main MS method (Algorithm 2), for $\mathcal{O}(\log(|V|))$ times, and combining this with Equation (5), yields the time complexity of $\mathcal{O}\big(\log(|V|) N |V| \left(|V| + T|E|\right)\big)$.

We complete this section by providing provable probabilistic performance lower bounds guarantees introduced in [35].

Probabilistic lower bounds

When running the MS algorithm independently for R replications, we get an unbiased estimators for the rare-event probability ℓ. Combining this with Markov's inequality [49], we can deliver lower bounds for ℓ using Theorem 3.3.

Theorem 3.3 (Probabilistic lower bounds)**.** *Given n samples $(Z_1, \ldots, Z_R)$, drawn independently from a proposal distribution Q such that $\mathbb{E}(Z_r) = \mu$ for $r = 1, \ldots, R$, and a constant $0 \leq \alpha < 1$, the following probabilistic lower bounds exist.*

(1) Minimum scheme bound (MSB):

$$\mathbb{P}\left(\min_{1\leq r\leq R}\left[\frac{Z_r}{\eta}\right] \leq \mu\right) \geq \alpha, \quad \textit{where } \eta = \left(\frac{1}{1-\alpha}\right)^{\frac{1}{R}}.$$

(2) Average scheme bound (ASB):

$$\mathbb{P}\left(\left[\frac{\frac{1}{R}\sum_{r=1}^{R} Z_r}{\eta}\right] \leq \mu\right) \geq \alpha, \quad \textit{where } \eta = \frac{1}{1-\alpha}.$$

(3) Maximum scheme bound (MASB):

$$\mathbb{P}\left(\max_{1\leq r\leq R}\left[\frac{Z_r}{\eta}\right] \leq \mu\right) \geq \alpha, \quad \textit{where } \eta = \frac{1}{1-\alpha^{\frac{1}{R}}}.$$

(4) Permutation scheme bound (PSB):

$$\mathbb{P}\left(\max_{1\leq r\leq R}\left[\left(\frac{1}{\eta}\prod_{j=1}^{r} Z_j\right)^{1/r}\right] \leq \mu\right) \geq \alpha,$$

where $\eta = 1/(1-\alpha)$.

(5) Order Statistics bound (OSB):

$$\mathbb{P}\left(\max_{1\leq r\leq R}\left[\left(\frac{1}{\eta}\prod_{j=1}^{r}\frac{O_{(R-j+1)}}{\binom{R}{r}}\right)^{1/r}\right] \leq \mu\right) \geq \alpha,$$

where $\eta = 1/(1-\alpha)$, and $\left(O_{(1)}, \ldots, O_{(R)}\right)$ is an order statistics over the sample set $(Z_1, \ldots, Z_R)$, such that for $1 \leq r_1 < r_2 \leq R$, it holds that $O_{(r_1)} \leq O_{(r_2)}$.

Proof. See [35]. □

These bounds introduce a bonus feature to a user of the MS algorithm, since they are available after the first few independent runs of Algorithm 2. In particular, the following can be achieved in a relatively small amount of computational effort.

- Our numerical study indicates that these bounds are tight, so we can estimate the order of magnitude of ℓ after only a few iterations of MS, and from this estimate how many runs will be needed in total to obtain a suitable accuracy of the final estimator.
- From practical point of view, when we explore different graph (atomic) structures, one might be interested in a rough estimation of their counting values. For example, when the model is big and thus reaching a predefined RE is expensive, or in order to perform a model comparison. For the latter, a computationally inexpensive (and tight!) lower bounds become very valuable, especially when many such comparisons should be made.

Remark 2 (Performance of Probabilistic Lower Bounds)**.** It was readily noted in [35], these bounds sometimes decrease when the number of samples grows, *i.e.,* the bound tightness decreases. Ideally, we would expect the bound quality to improve with more data, so this behavior is undesirable. However, on the positive side, the bounds are easy to obtain and they are accurate enough to be useful (our numerical study indicates that the bounds are generally within an order of magnitude of the true result or better after only a few runs).

4. Benchmarks

We benchmark the algorithm on three classes of example networks. In particular, we are dealing with the following graphs.

(1) The first class are the book graphs, for which the chromatic polynomial is known, and hence we can use these to precisely assess accuracy of the MS algorithm.
(2) The second class includes the two- and three-dimensional grids, which are used to model a physical atomic structure, and so provide a more realistic challenge on which to judge speed.
(3) The third class is a well-known graph coloring benchmark set[1]. Our MS benchmarking investigation is thus concluded by considering the performance of Algorithm 4 on these instances.

We implemented the MS algorithm in a C++ package called ChromSplit, which is freely available at `http://www.smp.uq.edu.au/people/RadislavVaisman/#software`, along with its source code and all examples. Timing measures were instrumented directly into the code. All the tests were executed on a Intel Core i7-3770 quad-core 3.4Ghz processor with 8GB of RAM, running 64 bit Windows 7. All tests were single-threaded, though parallelization would be easy to add.

MS parameters

We ran a number of preliminary benchmarks (not reported here) to determine reasonably robust parameter settings for N and ρ. The following parameters were used for all results described here.

- q is the number of colors (spins).
- For the MS pilot run, we take $\rho = 20\%$.

[1] `http://mat.gsia.cmu.edu/COLOR/instances.html`

- For the probabilistic lower bounds in Theorem 3.3, we take $\alpha = 0.95$, which is equivalent to choosing 95% confidence.
- We set the sample size to be $N = 10|E|$, where $|E|$ is the number of edges in a graph under consideration.

4.1. *Book graph*

We start by considering the n-book graph $\mathcal{B}_n$, which is defined as the graph Cartesian product $\mathcal{S}_{(n+1)} \times \mathcal{P}_2$, where $\mathcal{S}_{(n+1)}$ is a star graph and $\mathcal{P}_2$ is the path graph on two nodes [50]. An example, $\mathcal{B}_6$, is given in Figure 4.

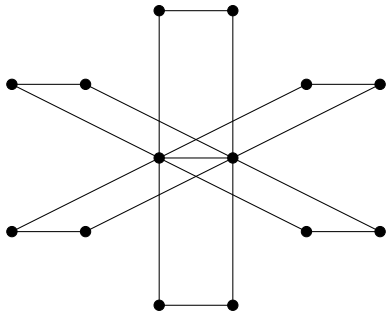


Figure 4. The 6-book graph, $\mathcal{B}_6$.

The $\mathcal{B}_n$ graph has $2n + 2$ vertices and $3n + 1$ edges. Importantly, the exact chromatic polynomial is known to be $q\,(q-1)\left(q^2 - 3q + 3\right)^n$ [50], thus we can measure our estimator's relative experimental error RER, and provide precise benchmarks for the accuracy of the MS algorithm.

We perform benchmarks on $\mathcal{B}_{100}$ for $q = 2, \ldots, 10$. As noted earlier, we choose $\rho = 20\%$ and $N = 3010$, since $\mathcal{B}_{100}$ has 301 edges.

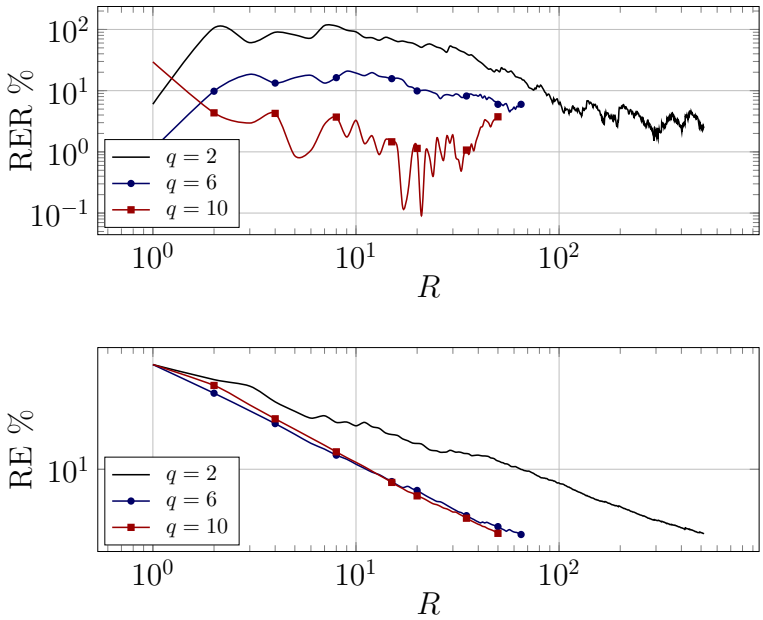


Figure 5. The RER and RE of the MS algorithm on $\mathcal{B}_{100}$ as a function of R – the number of independent MS replications.

Figure 5 shows the RER and the RE as a function of R, the number of independent MS replications, for $q = 2, 6$ and 10. The figure shows that the RER drops below 10% within a reasonable number of repetitions. One should remember that the probabilities being measured here are *very* small (as small as 10^{-60}), so a 10% RER is quite reasonable. One should also bear in mind, that the MS estimator unbiasedness implies the downwards trend of RER for $R \to \infty$.

The rightmost point on each curve in Figure 5 corresponds to the R value for which the estimated RE reaches 3%, and so we can see that the RE is a reasonable (if somewhat conservative) metric against which to estimate when the algorithm has converged sufficiently. In further tests, where we do not know the analytic form of the chromatic polynomial, we will use this criteria — the R value at which the estimated RE drops below 3% — to select the number of replications. Figure 5 also shows that

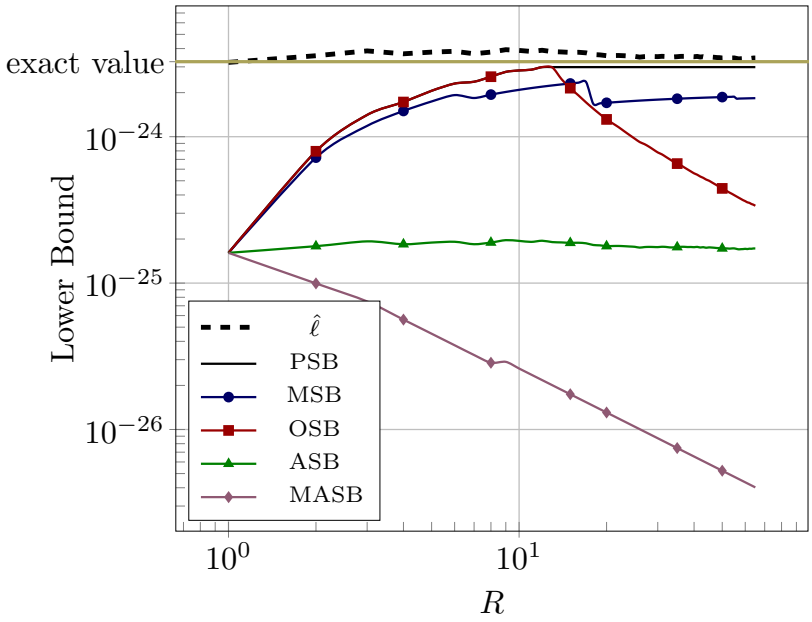


Figure 6. The estimator $\hat{\ell}$ and the corresponding lower bounds for the $\mathcal{B}_{100}$ graph for $q = 6$ as a function of replications number R. The estimator $\hat{\ell}$ is unbiased, but in this particular experiment, it first increased beyond the true value, so the PSB which is based on the maximum function, became the tightest bound.

the counting problem is harder (the MS method converges more slowly) for smaller values of q. This is not very surprising, since the corresponding decision problem of determining if the graph is q-colored, is also hard for smaller q values. This hardness follows from the fact that for smaller q values, one need to satisfy more coloring constraints.

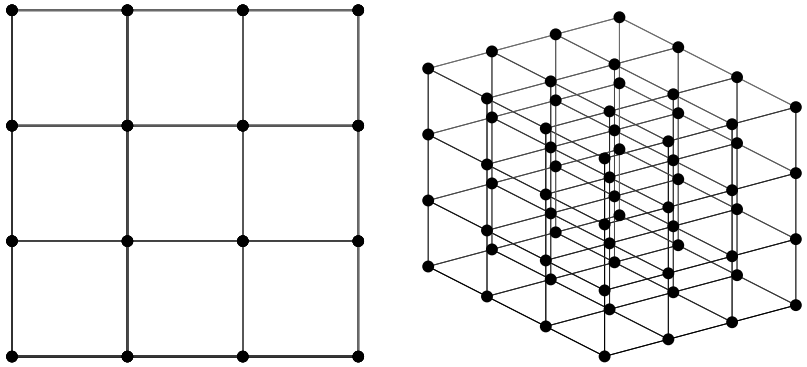
A similar set of results is shown in Figure 6, this time illustrating the lower-bounds as a function of the number of replications R. For clarity we show only $q = 6$ (in our experiments, we verified that the lower bounds behavior is similar for all $q = 2, \ldots, 10$). The figure shows that the best lower bound is the PSB. This lower bound becomes close to the exact value after only 10 replications of the MS algorithm, indicating its utility for providing rough estimates of ℓ early in the proceedings. For a six coloring ($q = 6$) of the $\mathcal{B}_{100}$ graph, our algorithm completes a single replication in about 5.3 seconds (see Table 1), so we can conclude that a tight lower bound can be obtained in less than a minute. Table 1 provides exact details of the data obtained for the $\mathcal{B}_{100}$ graph. The best lower bound is marked in bold.

Table 1. Performance summary of the MS algorithm on the $\mathcal{B}_{100}$ graph.

q	$\hat{\ell}$	RER	T	R	CPU (s)	PSB	MSB	OSB	ASB	MASB
2	3.19×10^{-61}	2.40%	103	517	3.80×10^{3}	$\mathbf{2.67 \times 10^{-61}}$	1.37×10^{-62}	3.76×10^{-63}	1.59×10^{-62}	1.58×10^{-64}
3	1.23×10^{-48}	5.09%	91	166	1.24×10^{3}	$\mathbf{7.57 \times 10^{-49}}$	3.73×10^{-49}	2.87×10^{-50}	6.14×10^{-50}	9.62×10^{-52}
4	1.00×10^{-36}	6.62%	67	124	7.93×10^{2}	$\mathbf{5.88 \times 10^{-37}}$	3.24×10^{-37}	3.80×10^{-38}	5.01×10^{-38}	8.35×10^{-40}
5	3.20×10^{-29}	0.54%	54	132	8.07×10^{2}	$\mathbf{2.48 \times 10^{-29}}$	1.54×10^{-29}	1.51×10^{-30}	1.60×10^{-30}	2.77×10^{-32}
6	3.45×10^{-24}	5.98%	44	65	3.41×10^{2}	$\mathbf{2.99 \times 10^{-24}}$	1.83×10^{-24}	3.37×10^{-25}	1.72×10^{-25}	4.02×10^{-27}
7	1.12×10^{-20}	0.50%	38	85	4.23×10^{2}	$\mathbf{8.48 \times 10^{-21}}$	5.49×10^{-21}	1.04×10^{-21}	5.58×10^{-22}	1.26×10^{-23}
8	4.33×10^{-18}	7.51%	34	56	2.52×10^{2}	$\mathbf{3.53 \times 10^{-18}}$	2.80×10^{-18}	6.56×10^{-19}	2.17×10^{-19}	6.07×10^{-21}
9	5.09×10^{-16}	4.52%	30	60	2.63×10^{2}	$\mathbf{4.28 \times 10^{-16}}$	2.85×10^{-16}	8.78×10^{-17}	2.55×10^{-17}	6.49×10^{-19}
10	2.00×10^{-14}	3.52%	27	51	2.13×10^{2}	$\mathbf{1.69 \times 10^{-14}}$	1.13×10^{-14}	4.57×10^{-15}	1.00×10^{-15}	2.58×10^{-17}

4.2. *Two- and three-dimensional grids*

The second class of models considered are the two- and three-dimensional grids, which are illustrated in Figure 7. The *order* of a grid graph refers to the length of a side, so Figure 7 shows the order-4 grid graphs.



(a) The 2D order-4 grid. (b) The 3D order-4 grid.

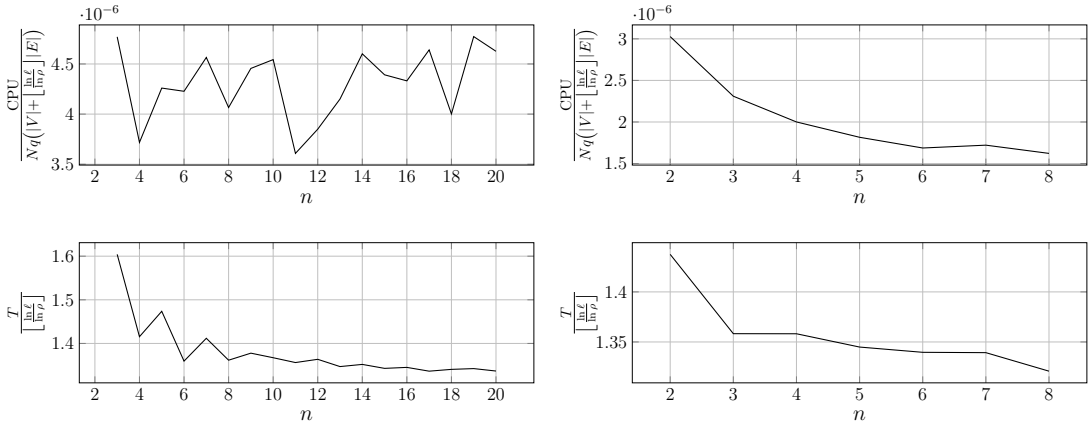
Figure 7. Example grid graphs.

We consider 18 2D graph instances of orders $n = 3, \ldots, 20$, and 6 3D graph instances of orders $n = 3, \ldots, 8$, each for $q = 2, 3, 4$ and 5. There are $|E| = 2n(n-1)$ and $3n^2(n-1)$ edges in 2D and 3D grids of order n, respectively. We choose $N = 10|E|$ as before and we choose the number of replications R such that the 3% RE requirement is met.

Figure 8 shows the average experimental CPU and the number of levels T as a function of the 2D and 3D grid order. The lower plots show the number of levels, T, derived from the pilot run, divided by the estimate (6), from which we note that the experimental value of T is at most 61% higher than the suggested rule of thumb in (6), and converges to being about 30% higher for larger networks.

Figure 8 also shows, in the upper plots, the CPU times in seconds, scaled by (5). We can see from these, that the scaled times are roughly constant with n (for all n for the 2D case, and for $n \geq 6$ for the 3D case). The unscaled CPU times are shown in Tables 2 and 3 for the 20×20 and $8 \times 8 \times 8$ grids, respectively. The tables also provide summaries of the T and R values for the different q values, along with the actual ℓ estimates. Note that the smallest of these is $\sim 10^{-157}$. Estimating probabilities this small naively is completely impractical.

Note also the small inversion in probabilities: *i.e.,* in the 3D order-8 grid, when we go from $q = 2$ to $q = 3$ the estimated value decreases, when naively we should expect



(a) 2D grid. (b) 3D grid.

Figure 8. The average CPU time (s) and the average number of levels T divided by analytical bounds (5) and (6). Note that the relative value of T is at most 61% higher than 1, and converges towards around 1.3, indicating that the (6) provides a reasonable estimate of the number of levels required. For the 2D graph, the scaled CPU time estimates appears consistent, and in the 3D case they appear to converge towards a consistent value, again suggesting the performance estimate is a good guide.

this number to increase as it should be harder to find proper colorings with a smaller number of colors (and indeed it is for all other cases). The reasons for this inversion must lie in the 3D structure, though the exact reason is unclear.

We next consider the probabilistic lower bounds. It is important to note that the calculation effort one should invest for the bounds calculation is relatively cheap as compared to the overall CPU time, so a reasonable approach is to compute all available bounds, and choose the maximum among them. We define the probabilistic maximal lower bound (PMLB) to be the maximum among PSB, MSB, OSB, ASB, and MASB.

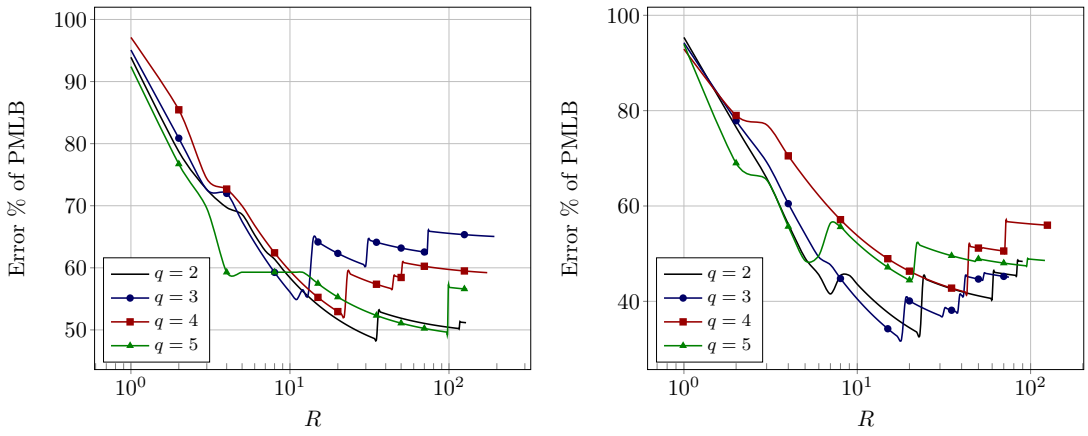
Table 2. Performance summary of the MS algorithm on the 20×20 2D grid.

q	$\hat{\ell}$	T	R	CPU	PSB	MSB	OSB	ASB	MASB
2	7.45×10^{-121}	248	128	1.07×10^{4}	1.58×10^{-121}	$\mathbf{3.64 \times 10^{-121}}$	3.90×10^{-123}	3.72×10^{-122}	6.54×10^{-124}
3	5.97×10^{-113}	216	193	1.62×10^{4}	1.14×10^{-113}	$\mathbf{2.09 \times 10^{-113}}$	2.45×10^{-115}	2.99×10^{-114}	4.08×10^{-116}
4	7.78×10^{-90}	165	174	1.31×10^{4}	2.00×10^{-90}	$\mathbf{3.17 \times 10^{-90}}$	6.90×10^{-92}	3.89×10^{-91}	6.16×10^{-93}
5	6.11×10^{-72}	130	130	8.70×10^{3}	2.49×10^{-72}	$\mathbf{2.65 \times 10^{-72}}$	9.35×10^{-74}	3.05×10^{-73}	5.12×10^{-75}

Table 3. Performance summary of the MS algorithm on the $8 \times 8 \times 8$ 3D grid.

q	$\hat{\ell}$	T	R	CPU	PSB	MSB	OSB	ASB	MASB
2	1.55×10^{-154}	280	90	1.93×10^{4}	3.65×10^{-155}	$\mathbf{8.03 \times 10^{-155}}$	1.04×10^{-156}	7.77×10^{-156}	1.76×10^{-157}
3	9.27×10^{-158}	309	75	2.07×10^{4}	2.05×10^{-158}	$\mathbf{5.09 \times 10^{-158}}$	6.52×10^{-160}	4.63×10^{-159}	1.06×10^{-160}
4	6.89×10^{-142}	270	125	3.46×10^{4}	1.45×10^{-142}	$\mathbf{3.03 \times 10^{-142}}$	3.83×10^{-144}	3.45×10^{-143}	6.43×10^{-145}
5	2.10×10^{-121}	226	121	3.23×10^{4}	6.52×10^{-122}	$\mathbf{1.08 \times 10^{-121}}$	1.13×10^{-123}	1.05×10^{-122}	1.85×10^{-124}

Figure 9 presents the estimation of the RE of PMLB, with respect to the final mean value obtained, as a function of R, for $q = 2, \ldots, 5$. Note the decrease in the PMLB accuracy as the number of samples grows. See Remark 2. However, as noted earlier, a good idea regarding the ℓ's order of magnitude is obtained after only few iterations. Despite that the reported error stands within 30%–60% interval, one should bear in mind that we are dealing with the counting problem, which is much harder than its optimization counterpart. In particular, from a strictly theoretical point of view, one



(a) 2D order-20 grid.

(b) 3D order-8 grid.

Figure 9. The estimated relative error of PMLB with respect to $\hat{\ell}$ as a function of R.

cannot hope to achieve a bounded error approximation, since it implies that NP=RP holds [24].

Tables 2 and 3 provide the detailed bounds for the 20×20 and $8 \times 8 \times 8$ grids, respectively, with the best lower bound is marked in bold. In general, for these graphs the MSB bound introduced the best performance in contrast to the book model, where PSB was the best.

Remark 3 (Exact Tutte and chromatic polynomial calculation)**.** The chromatic polynomial can be obtained using the Tutte polynomial [42, 50], for which an exact solver exists [51]. Table 4 summarizes the CPU times for the exact and the approximate (MS), approaches applied to 2D grids (note that the comparison is not quite apples for apples in that the exact Tutte solver provides the complete polynomial, not the result for a single q value, but we perform measurements here for $q = 2$ which is usually the slowest case for the MS solver).

Table 4. CPU times (seconds) of the exact Tutte solver and the MS algorithm on 2D grids. The exact solver outperforms MS on small graphs, but its poor scaling means MS is quickly superior, and moreover the exact solver is limited to at most 8×8 grids. 3D grid results are described above.

Instance	CPU (exact)	CPU (MS)
3×3	5.01×10^{-4}	1.23×10^{-1}
4×4	2.18×10^{-3}	5.34×10^{-1}
5×5	3.21×10^{-2}	2.96
6×6	0.253	8.05
7×7	2.46	19.3
8×8	36.9	34.8
9×9	—	74.5

For small models, the exact solver performed extremely well. Indeed it was faster than solving using the MS algorithm. However, it scaled badly. The MS algorithm is better by the time we reach order 8, and the exact solver crashed after 4.5 *hours* on

the order 9 grid, as compared to a 74 s run by the MS approach. It is inconceivable that we could find the exact solution for the order 20 grid considered above using the exact solver.

The results are even more extreme for the 3D grids, where the MS algorithm is already faster for the order-3 grid (0.11 as compared to 4.71 seconds), and the exact solver crashed after 4 hours for even the order 4 grid. Thus the MS algorithm makes possible computations that are completely impractical by exact means.

We have demonstrated the accuracy and efficiency of the MS method for calculating the partition function or ℓ. Algorithm 4 presents a further step: computation of the chromatic number for a graph, *i.e.,* the minimum number q, for which a proper coloring exists.

4.3. *Benchmarking optimization problem by Algorithm 4*

As usual, for each $G = (V, E)$ we set $N = 10|E|$ and $\rho = 0.2$. Table 5 summarizes the obtained results; the graphs are sorted by their number of vertices. Here, $\hat{\chi}(G)$ and # bin. stand for the chromatic number estimator delivered by Algorithm 4, and the corresponding number of binary search iterations, respectively.

Table 5. Performance summary of Algorithm 4 on the set of benchmark problems from `http://mat.gsia.cmu.edu/COLOR/instances.html`. The "?" sign indicates graphs with an unknown chromatic number. The gray rows stand for instances, where Algorithm 4 could not identify the correct chromatic number.

$G(\|V\|, \|E\|)$	$\chi(G)$	$\hat{\chi}(G)$	# bin.	CPU
myciel3(11,20)	4	4	5	0.010
myciel4(23,71)	5	5	6	0.250
queen5.5(25,160)	5	5	6	1.212
queen6.6(36,290)	7	7	6	4.825
myciel5(47,236)	6	6	6	3.708
queen7.7(49,476)	7	7	6	14.67
queen8.8(64, 728)	9	9	7	49.26
huck(74,301)	11	11	8	11.37
jean(80,254)	10	10	7	8.782
queen9.9(81, 2112)	10	10	7	238.1
david(87,406)	11	11	7	23.05
myciel6(95,755)	7	7	8	72.69
queen8.12(96,1368)	12	12	7	212.7
queen10.10(100,2940)	?	11	8	615.7
games120(120,638)	9	9	8	76.30
queen11.11(121,3960)	11	12	8	1251
miles250(128,387)	8	8	8	38.39
miles500(128,1170)	20	20	8	234.0
miles750(128,2113)	31	31	8	629.5
miles1000(128,3216)	42	42	8	1290
miles1500(128,5198)	73	73	8	2471
anna(138,493)	11	11	9	60.88
queen12.12(144,5192)	?	13	8	2349
queen13.13(169,6656)	13	15	9	5020
mulsol.i.3(184,3916)	31	33	9	2791
mulsol.i.4(185,3946)	31	33	8	2454
mulsol.i.5(186,3973)	31	33	8	2516
mulsol.i.2(188,3885)	31	33	8	2561
myciel7(191,2360)	8	8	9	1195
queen14.14(196,8372)	?	16	8	7419
mulsol.i.1(197,3925)	49	49	9	2744

For a majority of the benchmark networks, Algorithm 4 was successful in finding the $\chi(G)$. In addition, the MS method delivers an upper bound for networks with unknown $\chi(G)$. However, MS failed to provide a correct answer for two `queen` networks, and the four graphs from the `mulsol` family. We conjecture, that this is due to the chosen performance function, since the only cause for the MS method potential failure, is an existence of a rare-event conditional probability $\{c_t\}$. Consequentially, a good choice of performance functions, is an important direction of future research.

5. The chromatic number of random graphs

We test the binary search Algorithm 4 by applying it (using parameters as set above) to calculate chromatic numbers of two standard random-graph models: the Gilbert-Erdős-Rényi (GER) [40, 41] and Watts-Strogatz (WS) [36] random graphs. The GER random-graph is the classic "random" network for which there are known bounds on the chromatic number [52].

The WS model is included because this has the so-called *small-world* property: for certain parameter values the average distance between nodes is small, but clustering coefficient is still high. Moreover, the GER is approximately embedded within the class of WS models (for sparse graphs), as are the k-regular graphs, and so the WS model shows a continuum of models interpolating between the two extremes of regularity and randomness. As far as we are aware, despite the importance of small-world models, there are no existing results for chromatic numbers on WS random graphs.

Note that the average node degree in $G(n,p)$ network, (where $G(n,p)$ stands for GER model with n nodes and an edge existence probability p), is equal to $(n-1)p$. We used bounds on the chromatic number of any GER random graph, $\chi\big(G(n,p)\big)$, given in (7), [52], namely:

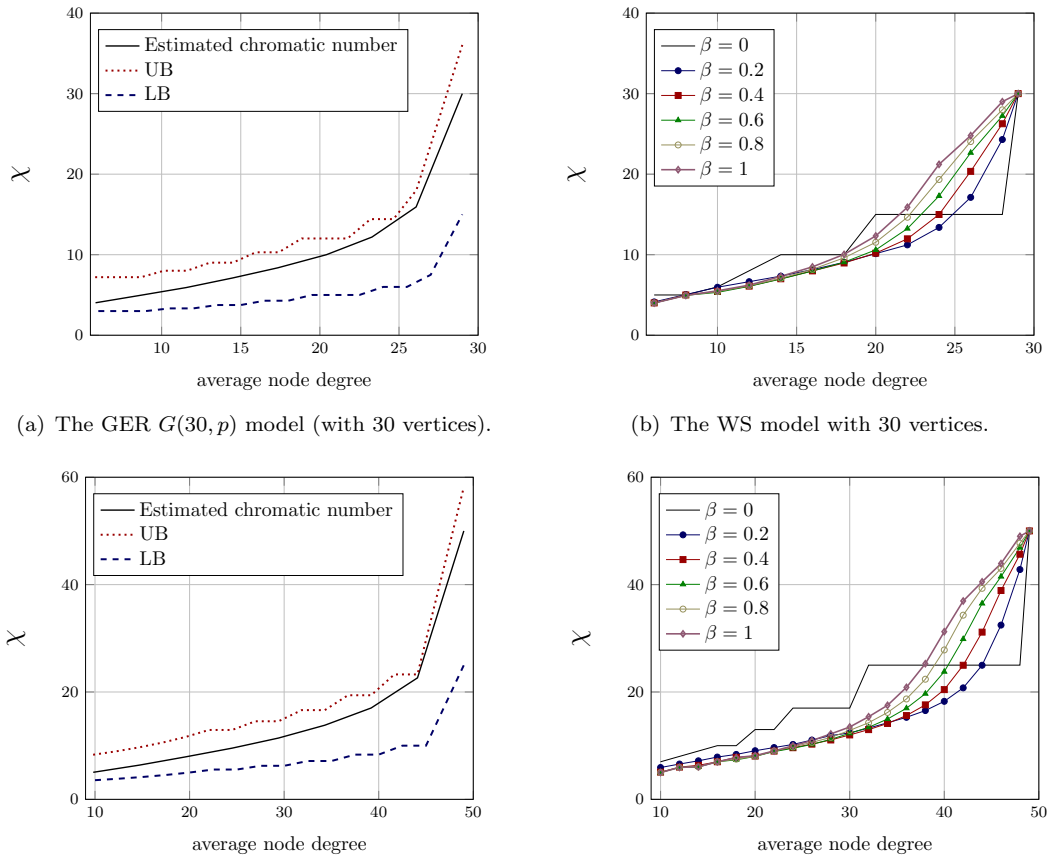
$$\frac{n}{s_0} \leq \chi\big(G(n,p)\big) \leq \frac{n}{s_0}\left(1+\frac{3\log\log(n)}{\log(n)}\right), \tag{7}$$

where $s_0 = \lceil r(n)+1 \rceil$, $b = 1/(1-p)$, and

$$r(n) = 2\log_b(n) - 2\log_b\log_b(n) + 2\log_b\left(\frac{\mathrm{e}}{2}\right).$$

The WS model used here starts by placing nodes regularly on a circle, and each node connects to its k nearest neighbors, where k is a model parameter that takes its values in $\{1,\ldots,n-1\}$. Then edges are *rewired* (one end is reconnected to a new node chosen uniformly from the alternatives) at random with probability $\beta \in [0,1]$. The average node degree in this network remains k, as the total number of edges remains constant.

It is noteworthy that the WS model is a k-regular graph for $\beta = 0$, and approximately equivalent to the GER random graph for $\beta = 1$ (and for sparse graphs), thus the WS model interpolates between these two extremes. And it does so in an interesting manner in that for moderately small values of β the graph has the small-world property. We calculated the chromatic numbers for these random graphs for $n = 10, 20, 30, 40$ and $n = 50$ with different values of p, k and β. For every parameter set, we calculated the chromatic number of 100 randomly generated instances, and we report the average values.



(a) The GER $G(30, p)$ model (with 30 vertices).

(b) The WS model with 30 vertices.

(c) The GER $G(50, p)$ model (with 50 vertices).

(d) The WS model with 50 vertices.

Figure 10. Chromatic number as a function of average node degree. The leftmost plot is for the $G(n, p)$ GER model and the rightmost plot is for the WS model.

Figure 10 summarizes the results for $n = 30$ and $n = 50$ (the $n = 10, 20$ and 40 results are omitted as they are very similar). The graphs show average chromatic number with respect to the average node degree, so that we can easily compare the two differently parameterized graphs on equivalent axes. The bounds in (7) are shown in Figure 10 as LB and UB.

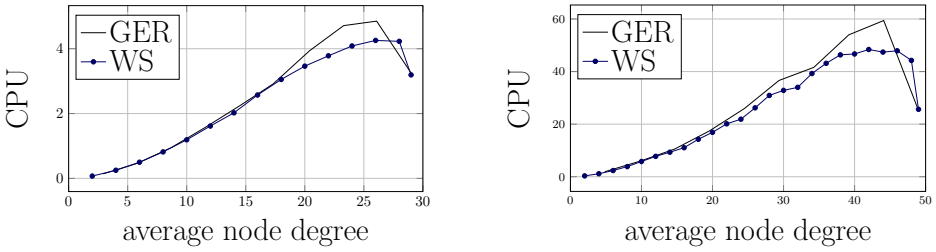
The GER results are quite expected, but noteworthy in that the chromatic number increases only slowly with average node degree, until the graph is quite highly connected, where the chromatic numbers shoot up (it is known that the chromatic number is bounded below by the size of the smallest clique, and hence the number must eventually converge to the size of the network as $p \to 1$). Also, the scaled shape of the curve is almost independent of n.

More interesting are the WS results. The $\beta = 0$ results are not stochastic, and hence follow a somewhat regular pattern. However, as soon as $\beta > 0$, we see that for graphs with low to medium values node degrees (less then or equal to 15 and 30 for WS graph with 30 and 50 vertices, respectively), the value of β has an insignificant effect. That is, all of the graphs from the almost k-regular, to the GER, have the same average chromatic number!

For highly connected graphs the value of β starts to have some effect on the chromatic numbers, with χ increasing as β increases.

It is interesting to note that the lower and upper bounds of the GER model hold for the WS graphs as well for graphs of moderate degree, *i.e.,* in the range where most applications of WS lie.

In both cases, the size of the network $n = |V|$ has little effect on the relative shape of the curves.



(a) The average CPU time as a function of average node degree of GER and WS graphs with 30 vertices.

(b) The average CPU time as a function of average node degree of GER and WS graphs with 50 vertices.

Figure 11. Average computation times of Algorithm 4 for calculating χ as a function of the average node degree.

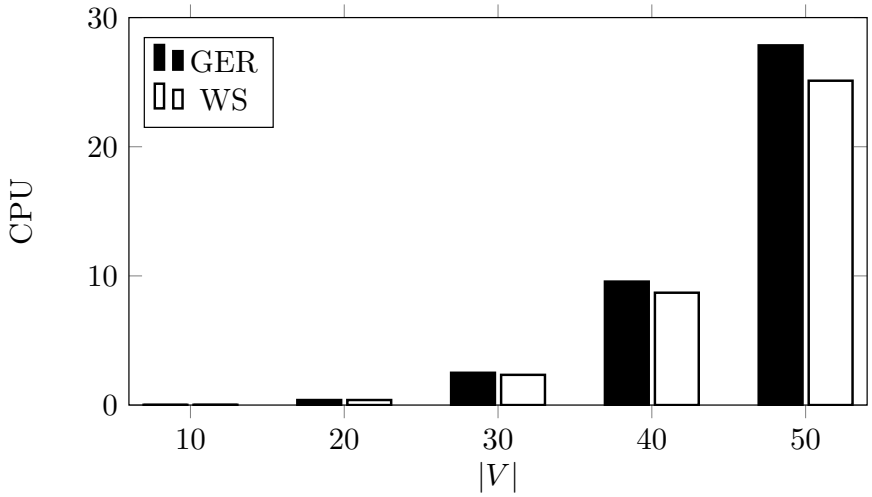


Figure 12. The average computation times of the Algorithm 4 for calculating χ as a function of $|V|$ — the number of graph's vertices.

Average computation times are shown in Figure 11 and Figure 12. In Figure 11, we report the CPU as a function of average node degree for graph sizes $|V| = 30$ and $|V| = 50$ in sub-figures (a) and (b), respectively (the $n = 10, 20$ and 40 results are omitted as they are very similar). Figure 12 summarizes the average computation times as a function of the graph's number of vertices $|V| \in \{10, 20, 30, 40, 50\}$. Both GER and WS models introduce a similar behavior in the sense of the required CPU time. Both numerical findings indicate that the computation time scales linearly with the number of edges in the graph.

6. Conclusion

In this paper, we investigated two hard counting problems, the graph coloring and the Potts model zero-temperature partition function approximation. By introducing an equivalent rare-event estimation problem, we were able to apply the adaptive MS approach. We showed that the MS algorithm provides a provable probabilistic performance lower bound guarantee which is easy to calculate on-line. Our numerical results indicate that the proposed method is successful in handling these hard rare-event estimation problems and the proposed probabilistic lower bounds seem to be close to the final estimator value.

In addition, we considered the chromatic number properties in random ER and WS networks. Our findings imply that these networks with low to medium average vertex degrees, share a similar behavior in the sense of their typical chromatic number. Based on the obtained results, we conjecture that this behavior is probably due to the fact that clustering coefficients in small-world networks do not change quickly in response to rewiring, whereas the network diameter does. So clustering appears to be much more important for determining chromatic numbers than properties such as the network diameter.

As for the future work, it will be important to identify specific graph topologies for which a rigorous performance guarantees (in the sense of the RE) could be obtained. From the practical point of view, a choice of a performance function is an essential direction to explore. In addition, it will be interesting to handle a general Potts model, which implies an approximation of the Tutte polynomial. Moreover, a comprehensive theoretical analysis of the small-world phenomena is of a clear interest. Finally, since the MS algorithm is easily parallelized, it will be of merit to develop a software package that runs on multiple CPU/GPU, in order to allow researchers dealing with statistical physics, to calculate the zero-temperature Potts model partition function for larger graphs.

Acknowledgements

We are thoroughly grateful to the anonymous reviewers for their valuable and constructive remarks and suggestions. This work was supported by the Australian Research Council Centre of Excellence for Mathematical & Statistical Frontiers, under grant number CE140100049. We would also like to thank Robert Salomone, for fruitful discussions at the early stage of this work.

References

[1] R.B. Potts and C. Domb, *Some generalized order-disorder transformations*, Proceedings of the Cambridge Philosophical Society 48 (1952), p. 106.

[2] M.I.D. Berganza, V. Loreto, and A. Petri, *Phase ordering and symmetries of the potts model*, Philosophical Magazine 87 (2007), pp. 779–786.

[3] F.Y. Wu, *The Potts model*, Rev. Mod. Phys. 54 (1982), pp. 235–268.

[4] R.J. Baxter, *Exactly solved models in statistical mechanics*, Academic Press, London, 1982.

[5] D. Hu, P. Ronhovde, and Z. Nussinov, *Phase transitions in random potts systems and the community detection problem: spin-glass type and dynamic perspectives*, Philosophical Magazine 92 (2012), pp. 406–445.

[6] M. Jerrum and A. Sinclair, *Polynomial-Time Approximation Algorithms for the Ising Model*, SIAM Journal on Computing 22 (1993), pp. 1087–1116.

[7] N. Alon, A.M. Frieze, and D. Welsh, *Polynomial Time Randomized Approximation Schemes for Tutte-Gröthendieck Invariants: The Dense Case*, Random Struct. Algorithms 6 (1995), pp. 459–478.

[8] F. Friedrich, A. Kempe, V. Liebscher, and G. Winkler, *Complexity Penalized M-Estimation: Fast Computation*, Journal of Computational and Graphical Statistics 17 (2008), pp. pp. 201–224.

[9] Y. Boykov, O. Veksler, and R. Zabih, *Fast Approximate Energy Minimization via Graph Cuts*, IEEE Trans. Pattern Anal. Mach. Intell. 23 (2001), pp. 1222–1239.

[10] J. Voit, *The Statistical Mechanics of Financial Markets*, Applications of Mathematics, Springer, New York, 2001.

[11] F. Graner and J.A. Glazier, *Simulation of biological cell sorting using a two-dimensional extended Potts model*, Phys. Rev. Lett. 69 (1992), pp. 2013–2016.

[12] C.I. Chou and S.P. Li, *Spin systems and Political Districting Problem*, Journal of Magnetism and Magnetic Materials 310 (2007), pp. 2889–2891, proceedings of the 17th International Conference on Magnetism.

[13] M. Bordewich, C.S. Greenhill, and V. Patel, *Mixing of the Glauber dynamics for the ferromagnetic Potts model*, CoRR abs/1305.0776 (2013).

[14] A.D. Sokal, *The multivariate Tutte polynomial (alias Potts model) for graphs and matroids*, in *Surveys in Combinatorics 2005*, B.S. Webb, ed., Cambridge University Press, 2005, pp. 173–226.

[15] G.D. Birkhoff and D.C. Lewis, *Chromatic Polynomials*, Trans. Amer. Math. Soc. 60 (1946), pp. 355–451.

[16] R. Karp, *Reducibility among Combinatorial Problems*, in *Complexity of Computer Computations*, R. Miller and J. Thatcher, eds., Plenum Press, 1972, pp. 85–103.

[17] L.G. Valiant, *The Complexity of Enumeration and Reliability Problems*, SIAM Journal on Computing 8 (1979), pp. 410–421.

[18] S.P. Vadhan, *The Complexity of Counting in Sparse, Regular, and Planar Graphs*, SIAM Journal on Computing 31 (1997), pp. 398–427.

[19] R.M. Karp and M. Luby, *Monte-Carlo algorithms for enumeration and reliability problems*, in *Proceedings of the 24th Annual Symposium on Foundations of Computer Science*. 1983, pp. 56–64.

[20] M. Dyer, *Approximate counting by dynamic programming*, in *Proceedings of the 35th ACM Symposium on Theory of Computing*. 2003, pp. 693–699.

[21] M. Jerrum, A. Sinclair, and E. Vigoda, *A Polynomial-Time Approximation Algorithm for the Permanent of a Matrix with Non-Negative Entries*, Journal of the ACM (2004), pp. 671–697.

[22] M. Dyer, A. Frieze, and M. Jerrum, *On counting independent sets in sparse graphs*, in *In 40th Annual Symposium on Foundations of Computer Science*. 1999, pp. 210–217.

[23] F. Jaeger, D.L. Vertigan, and D. Welsh, *On the computational complexity of the Jones and Tutte polynomials*, Mathematical Proceedings of the Cambridge Philosophical Society 108 (1990), pp. 35–53.

[24] L.A. Goldberg and M. Jerrum, *Inapproximability of the Tutte polynomial*, Information and Computation 206 (2008), pp. 908–929.

[25] M. Jerrum, L.G. Valiant, and V.V. Vazirani, *Random Generation of Combinatorial Structures from a Uniform Distribution*, Theor. Comput. Sci. 43 (1986), pp. 169–188.

[26] R.Y. Rubinstein, *The Gibbs Cloner for Combinatorial Optimization, Counting and Sampling*, Methodology and Computing in Applied Probability 11 (2009), pp. 491–549.

[27] Z.I. Botev and D.P. Kroese, *Efficient Monte Carlo simulation via the Generalized Splitting method*, Statistics and Computing 22 (2012), pp. 1–16.

[28] M. Jerrum and A. Sinclair, *The Markov Chain Monte Carlo Method: An Approach To Approximate Counting And Integration*, in *Approximation Algorithms for NP-hard Problems*, D. Hochbaum, ed. PWS Publishing, 1996, pp. 482–520.

[29] R.Y. Rubinstein, A. Dolgin, and R. Vaisman, *The Splitting Method for Decision Making*, Communications in Statistics - Simulation and Computation 41 (2012), pp. 905–921.
[30] J.K. Blitzstein and P. Diaconis, *A Sequential Importance Sampling Algorithm for Generating Random Graphs with Prescribed Degrees*, Internet Mathematics 6 (2011), pp. 489–522.
[31] Y. Chen, P. Diaconis, S.P. Holmes, and J.S. Liu, *Sequential Monte Carlo Methods for Statistical Analysis of Tables*, Journal of the American Statistical Association 100 (2005), pp. 109–120.
[32] R.Y. Rubinstein, A. Ridder, and R. Vaisman, *Fast Sequential Monte Carlo Methods for Counting and Optimization*, Wiley Series in Probability and Statistics, John Wiley & Sons, New York, 2013.
[33] J. Blanchet and D. Rudoy, *Rare Event Simulation and Counting Problems*, in *Rare Event Simulation Using Monte Carlo Methods* [45].
[34] H. Kahn and T.E. Harris, *Estimation of Particle Transmission by Random Sampling*, National Bureau of Standards Applied Mathematics Series 12 (1951), pp. 27–30.
[35] V. Gogate and R. Dechter, *Sampling-based Lower Bounds for Counting Queries*, Intelligenza Artificiale 5 (2011), pp. 171–188.
[36] D.J. Watts and S.H. Strogatz, *Collective dynamics of'small-world'networks.*, Nature 393 (1998), pp. 409–10.
[37] M. Barthélémy and L.A.N. Amaral, *Small-world networks: Evidence for a crossover picture*, Phys. Rev. Lett. 82 (1999), pp. 3180–3183.
[38] M. Humphries, K. Gurney, and T. Prescott, *The brainstem reticular formation is a small-world, not scale-free, network*, Proceedings of the Royal Society of London B: Biological Sciences 273 (2006), pp. 503–511.
[39] Q.K. Telesford, K.E. Joyce, S. Hayasaka, J.H. Burdette, and P.J. Laurienti, *The ubiquity of small-world networks.*, Brain Connectivity 1 (2011), pp. 367–375.
[40] P. Erdös and A. Rényi, *On random graphs, I*, Publicationes Mathematicae (Debrecen) 6 (1959), pp. 290–297.
[41] E. Gilbert, *Random graphs*, Annals of Mathematical Statistics 30 (1959), pp. 1441–1144.
[42] D.J.A. Welsh and C. Merino, *The potts model and the tutte polynomial*, Journal of Mathematical Physics 41 (2000), pp. 1127–1152.
[43] J. Salas and A.D. Sokal, *Transfer Matrices and Partition-Function Zeros for Antiferromagnetic Potts Models. I. General Theory and Square-Lattice Chromatic Polynomial*, Journal of Statistical Physics 104 (2001), pp. 609–699.
[44] P.T.D. Boer, D.P. Kroese, S. Mannor, and R.Y. Rubinstein, *A tutorial on the cross-entropy method*, Annals of Operations Research 134 (2002), pp. 19–67.
[45] G. Rubino and B. Tuffin, *Rare Event Simulation Using Monte Carlo Methods*, John Wiley & Sons, United Kingdom, 2009.
[46] R.Y. Rubinstein and D.P. Kroese, *Simulation and the Monte Carlo Method*, 2nd ed., Wiley Series in Probability and Statistics, John Wiley & Sons, New York, 2008.
[47] O. Strichman, *Efficient Decision Procedures for Validation: Translation Validation, Decision Procedures for Equality Logic, and SAT Tuning for Bounded Model Checking*, LAP Lambert Acad. Publ., 2009.
[48] F. Cérou, A. Guyader, R.Y. Rubinstein, and R. Vaisman, *On the Use of Smoothing to Improve the Performance of the Splitting Method*, Stochastic Models 27 (2011), pp. 629–650.
[49] E.M. Stein and R. Shakarchi, *Real analysis : measure theory, integration, and Hilbert spaces*, Princeton lectures in analysis, Princeton University press, Princeton (N.J.), Oxford, 2005.
[50] J.A. Gallian, *A Dynamic Survey of Graph Labeling*, Electronic J. Combinatorics 16 (2008), pp. 1–58.
[51] G. Haggard, D.J. Pearce, and G. Royle, *Computing Tutte Polynomials*, ACM Trans. Math. Softw. 37 (2010), pp. 24:1–24:17.
[52] A. Coja-Oghlan, K. Panagiotou, and A. Steger, *On the chromatic number of random graphs*, Journal of Combinatorial Theory, Series B 98 (2008), pp. 980 – 993.