Cycles in Sparse Random Graphs

Enzo Marinari and Valery Van Kerrebroeck
Dipartimento di Fisica, SMC-INFM-CNR and INFN, Sapienza Università di Roma, P. A.
Moro 2, 00185 Roma, Italy
E-mail: enzo.marinari@roma1.infn.it

Abstract. We study the cycles in sparse random graphs by treating them as a constraint satisfaction problem. We discuss how the model is defined, and the results that can be obtained in this way. In particular, we introduce and discuss an approximation to the self-avoiding walk-sum which emerges naturally from a slightly engineered version of the original problem.

1. Introduction

One way of characterizing (random) graphs is by analyzing their subgraphs. Rather than going about this problem in a direct way, it is generally more sensible to study the probability spaces, defined on appropriate ensembles of random graphs, which capture such particular subgraphs [1, 2]. Random graph theory is concerned both with the problem of deciding which subgraphs are interesting and reasonable to study, and with the problem of solving this question by applying probability theory.

In this paper, we concentrate on the presence of one type of subgraph which has since long been part of the discussion: cycles. A cycle is a non self-intersecting closed path along the edges of a graph. For regular graphs, probabilistic arguments have been very fruitful to find out for example what the distribution of cycles in these graphs look like [3, 4]. Similar results on Erdős-Rényi ensembles, where the connectivity of the vertices converges to a Poisson distribution, were until recently limited to specific regimes (near the percolation transition [5, 6, 7] or for very large connectivity [8, 9]). However, the introduction of methods traditionally used in statistical mechanics into the field of random graph theory has allowed to gather information regarding cycles for more generic random graph ensembles [10, 11, 12, 13, 14].

Here we use the statistical mechanics model that was introduced in [14] to allow the study of cycles in generic sparse random graphs, and discuss the results that can be obtained using this strategy. The same approach has been used to study other subgraphs and generic graph properties in [15, 16, 17].

In section 2 we introduce the necessary concepts and definitions. In section 3 we discuss the statistical mechanics model defining the problem and some generic tools which thereby become at our disposal. Section 4 is dedicated to a more in-depth discussion of the kind of information that can be obtained using this methodology. Finally, in section 5 we discuss the overall picture that emerges from this approach, and suggest some directions for possible future work.
2. Definitions

A graph $G$ is defined by a set $V$ of $N$ vertices and a set $E$ of $M$ edges. In this paper, we only consider undirected, simple graphs, i.e., neither self-loops nor multiple edges are allowed. An edge $\{i, j\} \equiv ij$ is uniquely defined by its ending vertices $i$ and $j$. If such an edge $\{i, j\}$ is part of the set $E$, the two vertices $i$ and $j$ are said to be adjacent. We refer to the set of neighboring vertices of vertex $i$ as $\partial i$. This information regarding which vertices are connected to each other in graph $G$ is summarized in the adjacency matrix $A = (a_{ij})_{i,j=1}^N$, where $a_{ij} = 1$ if $\{i, j\}$ is part of the set $E$ and zero otherwise. The connectivity $c_i$ of vertex $i$, which is defined as the number of edges the vertex $i$ belongs to, is then simply the sum of the elements on the $i$th column (or row) of the adjacency matrix.

We associate a weight $r_{ij}$ to each one of the edges $\{i, j\}$ of $E$ and define the degree $d_i$ of vertex $i$ as $d_i = \sum_{j \in \partial i} r_{ij}$, thereby making a clear distinction with its connectivity. In the particular case of unweighted graphs, all edge weights are assumed to be one and the degree and the connectivity of a vertex are equivalent. The partial correlation matrix contains the information about the edge weights,

$$ R = (r_{ij})_{i,j=1}^N. \tag{1} $$

The term “partial correlation matrix” derives from information theory, where it is common to analyze a graph by considering its information form [18]. This problem description is obtained by associating variables $x_i$ to each of the vertices which are distributed according to a joint Gaussian probability density $^1$,

$$ \text{Prob} [\mathbf{x}] \propto \exp \left\{ -\frac{1}{2} \mathbf{x}^T \mathbf{J} \mathbf{x} \right\}. \tag{2} $$

The key parameter in this graph model is the information matrix $\mathbf{J}$. Just like the adjacency matrix and the correlation matrix, $\mathbf{J}$ reflects the structure of the graph: only the diagonal elements and those off-diagonal elements corresponding to edges in $E$ are non-zero. Without loss of generality, one can assume $\mathbf{J}$ to be normalized such that $R = I - J$ [19].

A walk $W$ of length $l > 0$ in graph $G$ is a sequence $(i_1, i_2, \ldots, i_l)$, where each one of the vertices $i_k$ is adjacent to $i_{k+1}$ for all $k = 1, \ldots, l-1$. A cycle $C$ of length $l$ on graph $G$ is a closed, non self-intersecting walk in $G$. It is specified by a sequence $(i_1, i_2, \ldots, i_l, i_1)$, for which each couple of subsequent vertices are adjacent and all vertices $i_k$ for $k = 1, \ldots, l$ are distinct from each other. By definition, the length of a cycle cannot exceed the size of the graph $N$. A walk, on the other hand, could include multiple crossings of a single vertex or edge, hence its length can be infinite. The weight $w$ of a walk $W$, or cycle $C$, is the product of the weight of the edges composing the walk, or cycle, respectively,

$$ w[W] = \prod_{k=1}^{l-1} r_{i_k,i_{k+1}}, \quad w[C] = r_{i_1,i_l} \prod_{k=1}^{l-1} r_{i_k,i_{k+1}}. \tag{3} $$

In general, a simple subgraph $\mathcal{H}$ of graph $G$ is identified by the edges composing $\mathcal{H}$, which must be some subset of the original set $E$. We associate to each edge $\{i, j\}$ an edge variable $S_{ij}$ which takes on the value one if the edge is part of subgraph $\mathcal{H}$ and zero if it is not. In this way, any simple subgraph is completely and uniquely identified by the value of all the $M$ edge variables, i.e., by the configuration $\mathbf{S} = \{S_{ij}|\{i, j\} \in E\}$. The total length $l[\mathcal{H}]$ of the subgraph $\mathcal{H}$ demarcated by configuration $\mathbf{S}$ is the sum of edges composing $\mathcal{H}$,

$$ l[\mathcal{H}] = \sum_{\{i, j\} \in E} S_{ij}, \tag{4} $$

$^1$ The information form of a Gaussian graphical model generally also allows to associate a potential to each vertex, which here we assume to be zero for all vertices.
and its weight \( w[\mathcal{H}] \) is the product of the weight of these edges,

\[
w[\mathcal{H}] = \prod_{\{i,j\} \in E} r_{ij}^{S_{ij}}.
\]  

(5)

To ease the notation in the following we also introduce the vectorial vertex variables \( S_i = \{S_{ij}|j \in \partial i\} \) for each one of the vertices of the graph.

A possible subgraph which is uniquely identified by this configurational representation could be for example a cycle. A generic walk, with possible multiple crossings (of edges or vertices), on the other hand, is not uniquely represented in this format. The introduction of a Potts-like edge variable \( S_{ij} \in \{0, 1, 2, \ldots, q\} \) could account for \( q \) crossings in a single edge or vertex in order to preserve the sequential information of the walk. Analyzing the possible subgraphs arising in this enlarged phase space, now of size \((q+1)^M\), in the Bethe approximation, which we introduce below, is quite involved. In this paper, we limit the discussion to the configurational Ising-like framework and only deal with the characterization of simple subgraphs on simple graphs. On the other hand, an extension to undirected graphs is very straightforward, as it merely just doubles the number of edge variables we need to consider.

In the following we consider several types of sparse random graphs drawn from a fixed degree distribution ensemble [20, 21]. For example, we consider regular random graphs of connectivity \( c \), where all vertices have the same number of neighbours, i.e., with connectivity distribution \( q_c(k) = \delta_{k,c} \). We also consider graphs with a bimodal distribution

\[
q_{c_1,c_2}(k) = (1-\epsilon) \delta_{k,c_1} + \epsilon \delta_{k,c_2},
\]  

(6)

which interpolates between two different regular ensembles as \( \epsilon \) varies in \([0,1]\). The generation of these graphs with fixed degree distribution is based on the configuration model [22] and is described in more detail in [23].

We also consider Poissonian graphs of size \( N \) with mean connectivity \( c \), i.e., \( q_c^{ER}(k) = e^{-c}c^k/k! \). Such graphs are obtained by placing an edge between any of the \( N(N-1)/2 \) couple of vertices with a probability \( c/N \). In the following, we analyze the 2-core of these graphs by first applying a leaf removal algorithm [14]. The 2-core is defined as the largest subgraph for which each vertex has a minimal degree of two. From this definition it is clear that the longer cycles of a graph will be contained in its 2-core.

3. A Statistical Mechanics Approach

3.1. Cycles from a Constraint Satisfaction Problem

In [14] a probability law was introduced which selects those subgraphs defined on \( G \) that represent a set of vertex disjoint cycles of \( G \). We here extend that probability distribution to the following law which also accounts for the weight of the cycles,

\[
\text{Prob}[\mathcal{S}] = \frac{1}{Z(u)} \prod_{\{i,j\} \in E} (r_{ij}u)^{S_{ij}} \prod_{i \in V} f_i(\mathcal{S}_i),
\]  

(7)

where \( f_i(\mathcal{S}_i) = 1 \) if \( \sum_{j \in \partial i} S_{ij} \in \{0, 2\} \), \( f_i(\mathcal{S}_i) = 0 \) otherwise and \( Z(u) \) is a normalization constant which depends on the external parameter \( u \). The constraints \( f_i \), one to each vertex \( i \), ensure the only configurations having a non-zero probability are those where all vertices have either none or exactly two of the neighbouring edges occupied. Thus, the law (7) admits only those configurations which represent a set of vertex disjoint cycles and associates a probability to them which is proportional to their total weight \( w[\mathcal{S}] \).
Let us define the energy of a given allowed configuration as

\[
E_u(S) = - \sum_{\{i,j\} \in E} S_{ij} \log (ur_{ij}) = - \log \left( \prod_{\{i,j\} \in E} (ur_{ij})^{S_{ij}} \right).
\] (8)

In the case of a weighted graph, the energy of a given configuration is proportional to the logarithm of its weight. In the particular case of an unweighted graph, i.e. \(r_{ij} = 1\) for all \(\{i, j\} \in E\), the energy of a configuration is just its length. Combining (7) and (8), we obtain an optimization problem in the usual canonical form,

\[
\text{Prob}[S] \propto e^{-E_u(S)},
\] (9)

where an external parameter \(u\) determines the more probable configurations solely based on their energy. At this point we can revert to standard statistical mechanics techniques to answer questions related to for example the determination of a ground state or the most probable configuration.

By associating (edge) variable nodes to the edges and function nodes to the vertices of a given graph, we basically construct its corresponding factor graph (see figure 1). This factor graph representation clarifies the bipartite structure of a constraint satisfaction problem. In the case of (7), an edge variable \(S_{ij}\) depends on the two neighbouring function nodes \(f_i\) and \(f_j\), while the situation at a function node \(f_i(S_i)\) depends on its \(\partial i\) neighbouring edge variables \(S_{ij}\).

**Figure 1.** By associating a variable to each of the edges and introducing constraints on each of the vertices of the original simple graph \(G\) (on the left), we construct the corresponding bipartite factor graph (middle figure) composed of variable nodes (○) and function nodes (□). On the right, two oppositely directed messages flow between each set of adjacent vertices (or function nodes).

In order to restrict the subset of allowed configurations to those representing one single cycle, one could consider Potts-like variables as proposed in [14]. This is a possible direction we decided not to take in the previous section. Another possibility is to include a global weight which favours those configurations consisting of less vertex disjoint cycles. This breaks the factorizability of the problem, which is problematic in the Bethe approximation. Depending on which analyzing method is used, such a global factor could be included as we discuss below. In general, in section 4 we analyze the details of the differences between the two distributions in more detail.

3.2. *The Bethe Approximation and Belief Propagation*

One approximate way to deal with constraint satisfaction problems such as (7), is by taking only first order correlations into account, which is known as the Bethe approximation. In numerical analysis this Bethe approximation is known to be equivalent to the standard Belief Propagation (BP) algorithm [24].
Belief Propagation is an iterative distributed message passing algorithm for solving inference problems [25]. In particular, BP produces the so-called beliefs, which are approximations to the marginal probabilities. We associate two messages $y_{i \rightarrow j}$ and $y_{j \rightarrow i}$ to each edge, one in each direction (see figure 1), which initially take on random values. Each iteration step consists in the computation of these $2M$ messages as a function of the values of their surrounding neighbouring messages at the previous iteration. In the case of (7), the update rules take on the following form,

$$y_{i \rightarrow j}[S_{ij}] = \frac{u \sum_{k \in \partial j \setminus i} r_{kj} y_{k \rightarrow j}}{1 + \frac{1}{2} u^2 \sum_{k_1, k_2 \in \partial j \setminus i} r_{jk_1} y_{k_1 \rightarrow j} r_{jk_2} y_{k_2 \rightarrow j}}.$$  \hspace{1cm} (10)

From these messages we then calculate the various marginal probability distributions, and from these, the observables we are interested in. In particular, in the following we are interested in the marginals $p_{ij}$ and $p_i$ expressing the probability that a given edge, or vertex respectively, is part of a cycle $C$ represented by configuration $S$.

$$p_{ij} = \frac{u r_{ij} y_{j \rightarrow i} y_{i \rightarrow j}}{1 + u r_{ij} y_{j \rightarrow i} y_{i \rightarrow j}} , \hspace{1cm} (11)$$

$$p_i = \frac{1}{1 + \frac{1}{2} u^2 \sum_{k_1, k_2 \in \partial i} r_{ik_1} y_{k_1 \rightarrow i} r_{ik_2} y_{k_2 \rightarrow i}}.$$ \hspace{1cm} (12)

On tree-like graphs, this iterative process converges and leads to the exact problem solution. On more general graphs containing loops, Belief Propagation is not guaranteed to converge, and if it does it only leads to approximate results, exactly because they do not take higher order correlations into account [18, 24]. For the (sparse) graphs we consider, BP does converge for graphs of a reasonable system size, i.e., $N \gtrsim O(10^2)$. We present in figure 2 the number of iterations required for convergence, which clearly grows slowly with the graph size $N$ for all graphs investigated.

![Figure 2](image.png)  

**Figure 2.** Average number of iterations as a function of the system size $N$ (when $u = 1$), for different types of graphs: $q_4(\cdot)$, $q_5(\times)$, $q_{3,4}^{0.5}(\ast)$, $q_{3,5}^{0.5}(\Box)$, $q_3^{ER}(\blacksquare)$, $q_4^{ER}(\circ)$ and $q_5^{ER}(\bullet)$.  

5
An analytical approach to characterize random graph ensembles consists in analyzing the observables under different assumptions on the distributions governing the messages (10).

### 3.3. The Monte Carlo Markov Chain Method

Another approximate technique to sample from the configuration space according to distribution (7) is by means of the Monte Carlo Markov chain method [26, 23]. The idea is to construct a Markov chain $S, S', \ldots$ which admits the desired distribution as its stationary distribution. This process must be ergodic, i.e., it must be able to reach any configuration which has a non-zero probability, for example by demanding detailed balance. The real challenge, however, is to optimize the mixing time, i.e., the required length of the Markov chain to obtain a sampling which is sufficiently close to the desired distribution. In combinatorial optimization problems simulated annealing can be adopted to find (close to) optimal solutions in a relatively fast way. At the same time, we need to minimize the computational difficulty to go from one configuration $S$ to the next $S'$.

Building up an ergodic Markov chain in the restricted phase space determined by distribution (7) would be very time-consuming due to the latter requirement. Therefore, when deploying the Monte Carlo Markov chain method, it is better to consider a relaxed version of (7), similar to the one proposed in [23],

$$
\text{Prob}[S] = \lim_{\eta \to 0} \frac{1}{Z(u)} \prod_{(i,j) \in E} (r_{ij} u)^{S_{ij}} \prod_{i \in V} \tilde{f}_i(S_i) \eta^{|S|},
$$

(13)

where the parameter $\eta \in [0, 1]$, $|S|$ is the number of vertex disjoint cycles constituting configuration $S$, and the vertex functions $\tilde{f}_i$ are slightly altered such that they also admit configurations in which vertices have just one neighbouring edge occupied, i.e.,

$$
\tilde{f}_i = \begin{cases} 
1 & \text{if } \sum_{j \in \partial_i} S_{ij} \in \{0, 1, 2\} \\
0 & \text{otherwise.}
\end{cases}
$$

By allowing for configurations containing such non-closed walks, two consecutive configurations in the corresponding Markov chain can differ simply in one edge variable which clearly minimizes the computation time. However, this enlargement of the phase space results in a larger mixing time. We make up for this slower mixing partially by introducing the global weight $\eta^{|S|}$ which, in the limit $\eta \to 0$, concentrates on those configurations which consist of just one unique walk or cycle.

### 4. Applications

The statistical mechanics description introduced in the previous section allows us to analyze different aspects of cycles in graphs. In this section we give an overview of how it has been used in previous papers [13, 14, 23] and we introduce a new interpretation and interesting application of cycles as self-avoiding self-returning walks.

#### 4.1. Enumerating Circuits

The general problem of counting the cycles of a given graph requires only the knowledge of the edges, and not of their weights. Therefore, this problem can be addressed by considering the unweighted version of $\text{Prob}[S]$, where the energy of a given configuration $S$ is proportional to its length. The probability (9) of a (vertex disjoint set of) closed circuit(s) with total length $l$ is then proportional to $u^l$. Thus, the normalization constant $Z(u)$ actually plays the role of a generating function for the number $\mathcal{N}_l$ of (vertex disjoint sets of) cycles with length $l$,

$$
Z(u) = \sum_l u^l \mathcal{N}_l.
$$

(14)
In [14] it is then argued that, at a leading exponential order, the average of the number \(N_l\) of vertex disjoint sets of circuits with length \(l\) equals the average number of single cycles with length \(l\). Thus, in order to get a first approximation to the cycle distribution it suffices to analyze the generating function \(Z(u)\).

In the thermodynamic limit, the sum (14) can be evaluated by means of the saddle point method and as fluctuations around the mean vanish for \(N \to \infty\), the microcanonical entropy \(\sigma(l/N) = \log N_l/N\) can be obtained directly from this canonical problem formulation.

In [13] a BP algorithm is proposed which allows for a numerical approximation of the cycle distribution of large sparse graphs. For regular graphs an explicit solution can be derived in terms of the connectivity. In case of general random graphs with an arbitrary connectivity distribution, one can revert to a population dynamics algorithm [27]. An explicit analytic discussion is restricted to the limit of short and long circuits [14].

### 4.2. Finding Long Circuits

Even if the distribution of circuits can be determined, the problem of locating cycles of a desired length is not necessarily easy, as exhaustive search is only achievable in a reasonable time frame for small circuits, or in general on small graphs. In [23] the focus lies on finding long cycles, possibly Hamiltonian circuits, i.e., of length \(N\), in sparse random graphs.

A first approach to tackle this problem of locating long cycles consists in a decimation procedure. We start by iterating the BP equations (10) until they converge to a fixed point, after which we compute the marginal probabilities (11) that a given edge lies on a circuit. By fixing the edge variable of the edge with the most pronounced marginal probability to its most probable value, we obtain a new problem similar to the original one, but now with \(M - 1\) degrees of freedom. We again revert to BP to compute the marginals of this new problem, and continue this strategy until all edge variables are fixed to either zero or one.

If the marginals are exact, this isolates unions of vertex disjoint cycle with a weight (or length in case of an unweighted graph) determined by the typical energy at the preset value of parameter \(u\). If we choose \(u \to \infty\), we can isolate the vertex disjoint sets of circuits with the largest weight (or length). In general, these sets of vertex disjoint cycles are usually composed of one large circuit and a few smaller ones. By repeating the decimation procedure with different initial random seeds, one hopes to eventually come across a unique Hamiltonian cycle. Another possibility is to apply some local rewiring procedure which tries to unify the several vertex disjoint cycles into a single circuit.

In practice, when addressing the problem of finding circuits on a loopy graph, the expressions (11) are merely approximations to the actual marginals. By introducing some specific heuristics which tend to fix edge variables of neighbouring edges to one, the decimation strategy is still effective in a lot of cases [23].

A second approach to locate (long) cycles in graphs is based on a direct sampling of (13). The mixing time of the corresponding Monte Carlo algorithm proposed in section 3.3 is actually quite low, i.e., equilibrium sets in rather fast. However, in the particular case of Hamiltonian circuits the corresponding configurations need to be recovered as a fluctuation around the equilibrium value determined by the parameters \(u\) and \(\eta\). This requires a careful tuning of the parameter values for each different type of graph, which does turn out to be rather time consuming.

### 4.3. Self-Avoiding Walk-Sums

The graph model described by the Gaussian distribution (2) is completely defined by its covariance matrix,

\[
P = \int \text{Prob}[\mathbf{x}] \; \mathbf{x} \mathbf{x}^T \; d\mathbf{x} = J^{-1},
\]

(15)
where the inverse of the information matrix $J^{-1}$ can be obtained by Gaussian Belief Propagation. In [18] this covariance matrix $P$ is interpreted as a so-called “walk-sum”, which is deeply connected to Gaussian Belief Propagation.

The Walk-Sum $\Sigma_W$ of a graph is defined as the matrix of which the $(i, j)$th element is given by the sum of the weights of all walks $W_{i \rightarrow j}$ starting in vertex $i$ and ending in $j$,

$$
(\Sigma_W)_{i,j} = \sum_{\{W_{i \rightarrow j}\}} w[W_{i \rightarrow j}].
$$

(16)

The $(i, j)$th element of the matrix product $R^l$ of the correlation matrix is the sum of the weights of all walks of length $l$ starting in $i$ and ending in $j$. Hence, the walk-sum is simply given by

$$
\Sigma_W = \sum_{l=0}^{\infty} R^l.
$$

(17)

In [18] the covariance matrix is found to be identical to the walk-sum, i.e., $P = \Sigma_W$. In particular, the variance $P_{i,i}$ of vertex $i$ is a sum over all self-returning walks in vertex $i$, i.e., walks starting and ending in $i$,

$$
P_{i,i} = (\Sigma_W)_{i,i} = \sum_{l=0}^{\infty} (R^l)_{i,i}.
$$

(18)

In [18] a one-to-one correspondence is established between the finite-length walks in the original graph $G$ that end at vertex $i$ and the walks in the computation tree of the corresponding Gaussian BP with root $i$. This graphical picture explains the approximate nature of loopy Belief Propagation, as it only captures a fraction of the total self-returning walk-sum. Namely, loopy BP only accounts for the backtracking self-returning walks, i.e., the walks that are decomposable in a walk and its mirror image (its backtrack).

We propose here a way to compute the Self-Avoiding Walk-Sum, which we define as the sum of the weights of all self-avoiding self-returning walks of a given vertex. As cycles are defined as self-avoiding self-returning walks, the self-avoiding walk-sum of vertex $i$ is simply,

$$
(\Sigma_C)_{i} = \sum_{\{C \in \mathcal{C}\}} w[C].
$$

(19)

This particular Walk-Sum captures only a particular subset of the self-returning walks that are missing in Gaussian loopy BP. Even if it is not the exact complement of what the loopy BP variance estimation is lacking, in future it could be extended to compute the complete Walk-Sum (18).

The advantage of considering only the self-avoiding walks contributing to the walk-sum is due to the fact that it is a natural by-product of the fixed point of the BP equations (10). The marginals (12) express the probability that a given vertex lies on a circuit. Thus, they are proportional to the self-avoiding walk-sum of a given vertex,

$$
(\Sigma_C)_{i} \propto p_i.
$$

(20)

The self-avoiding walk-sum for a given vertex is recovered if the total sum of weighted cycles can be obtained.

As mentioned in section 3.2, the marginals obtained with BP are themselves an approximation to the actual marginal probabilities. In particular, they over-count $\Sigma_C$ since (7) selects unions
of cycles and not just single cycles. In figure 3 we report the ratio between the total number of valid configurations defined by these two distributions. This can only be done in an exact way through exhaustive search and is therefore limited to small graphs [28]. Figure 3 suggests the sizes of both distributions are of the same order for these small graphs, though the difference between the two grows with the size of the graph. For larger graphs, the combinatorial argument of [14] suggests this ratio remains essentially of order one. Moreover, if the correlation matrix $R$ of a weighted graph is rescaled such that all of its elements $r_{ij}$ lie in the interval $[0,1]$, the approximation due to this overcounting should be smaller, as unions of cycles composed of two or more cycles will weigh less than one of its single cycles itself.

![Figure 3. Ratio between the total number of unions of cycles and the total number of cycles as a function of the size $N$ of the graph, for the same types of graphs as in figure 2.](image)

5. Discussion and Perspectives

In this note we have extended the statistical mechanics model proposed in [14], such that it allows to study possibly weighted cycles in sparse random graphs. By defining an appropriate probability law on an Ising-like configuration space, statistical mechanics methods become available to address typical problems, such as obtaining the cycle distribution or finding a particular cycle.

The Monte Carlo sampling method allows for a precise problem definition, which however results in a larger mixing time. Working in the Bethe approximation opens the way to a possible analytic characterization of the problem by means of the cavity method [14]. On the algorithmic side, it leads to Belief Propagation, a distributed message passing algorithm with a running time which is polynomial in the system size. On loopy graphs it is not guaranteed to converge, however, and it generally leads to only approximate results.

We have introduced a novel approach to compute self-avoiding walk-sums. In future, it would be interesting to expand the model further in order to compute the actual walk-sum defined in [18]. On the one hand this requires we allow for edge variables of the Potts-type. However, special care needs to be taken to ensure the one-to-one mapping between the corresponding configuration space emerging from this problem definition and the set of (possibly infinite) walks on the graph we are considering, especially since we consider sets of self-returning walks, and not just single cycles.

The general walk-sum also includes non self-avoiding walks. This can be done by extending the function nodes to allow for three, four, ... neighbouring occupied edges. This comes down to allowing for three-point, four-point, ... correlations in the problem definition. A similar generalization would extend the Walk-Sum beyond Gaussian graphical models, which so far has not been possible following the strategy of [18], since there the Walk-Sum arises from the power series of the partial correlation matrix.
References
[1] Bollobás B 2001 Random graphs (Cambridge: University Press)
[2] Janson S, Łuczak T and Rucinski A 2000 Random graphs (New York: John Wiley and sons)
[3] Wormald N C 1999 Surveys in Combinatorics vol 276, ed Lamb J D and Preece D A (London: Mathematical Society Lecture Note Series) p 239
[4] Garmo H 1999 Random Struct. Alg. 15 43
[5] Janson S 1995 Comb. Probab. Comput. 4 369
[6] Łuczak T 1991 Random. Struct. Alg. 2 421
[7] Flajolet P, Knuth D E and Pittel B 1989 Discrete Math. 75 167
[8] Frieze A 1986 Discrete Math. 59 243
[9] Pósa L 1976 Discrete Math. 14 359
[10] Bianconi G, Caldarelli G and Capocci A 2005 Phys. Rev. E 71 066116
[11] Rozenfeld H D, Kirk J E, Boltt E M and Ben-Avraham D 2005 J. Phys. A 38 4589
[12] Bianconi G and Marsili M 2005 J. Stat. Mech. P06005
[13] Marinari E, Monasson R and Semerjian G 2006 Europhys. Lett. 73 8
[14] Marinari E and Semerjian G 2006 J. Stat. Mech. P06019
[15] Engel A, Monasson R and Hartmann A K 2004 J. Stat. Phys. 117 387
[16] Zdeborová L and Mézard M 2006 J. Stat. Mech. P05003
[17] Petti M and Weigt M 2006 Europhys. Lett. 75 8
[18] Malioutov D M, Johnson J K and Willsky A S 2006 Journal of Machine Learning Research 7 2031
[19] Lauritzen S 1996 Graphical Models (Oxford: University Press)
[20] Molloy M and Reed B 1998 Combin. Probab. Comput. 7 295
[21] Newman M E J, Strogatz S H and Watts D J 2001 Phys. Rev. E 64 026118
[22] Bollobás B 1980 Europ. J. Combinatorics 1 311
[23] Marinari E, Semerjian G and Van Kerrebroeck V 2007 Phys. Rev. E 75 066708
[24] Yedidia J S, Freeman W T and Weiss Y 2001 Advances in Neural Information Processing Systems 13 689
[25] Yedidia J S, Freeman W T and Weiss Y 2005 IEEE Transactions on Information Theory 51 7
[26] Jerrum M and Sinclair A 1996 Approximation Algorithms for NP-hard Problems ed Hochbaum D S (Boston: PWS Publishing) p 482
[27] Mézard M and Parisi G 2001 Eur. Phys. J. B 20 217
[28] Johnson D B 1975 SIAM J. Comput. 4 77