A tree-decomposed transfer matrix for computing exact partition functions for arbitrary graphs

Andrea Bedini[1] Jesper L. Jacobsen[2]

[1]MASCOS, The University of Melbourne, Melbourne

[2]LPTENS, École Normale Supérieure, Paris

Monash University
23 March, 2011

Outline

1. Introduction
2. Potts model and vertex colourings
3. The transfer matrix
4. Example
5. Tree-decomposition
6. Application to the distribution of chromatic roots

Outline

1. Introduction
2. Potts model and vertex colourings
3. The transfer matrix
4. Example
5. Tree-decomposition
6. Application to the distribution of chromatic roots

Counting problems and Statistical Mechanics

- SM studies the properties emerging in very large systems
- The possible emerging behaviour are often due to the competing effects of energy and entropy
- energy is a physical problem

  (what interaction is this energy due to?)
- entropy is a combinatorial problem

  (how many possible configurations are there?)

Counting problems and Statistical Mechanics

- SM studies the properties emerging in very large systems
- The possible emerging behaviour are often due to the competing effects of energy and entropy

- energy is a physical problem

  (what interaction is this energy due to?)
- entropy is a combinatorial problem

  (how many possible configurations are there?)

Fundamental concepts

- In SM one has a “model” defined on some graph (regular or not).
- A model is made out of two elements:
  - A configuration space $\mathcal{C}$
  - A function associating to each configuration a discrete energy E.
- The statistical weight of a configuration is e^{-E} (Gibbs weight)
- Some configurations can be very rare but still dominat if their energy is small

All the properties of a system are deduced from the *partition function*:

$$Z = \sum_{\mathcal{C}} e^{-E}$$

which can be thought as an energy generating function

- Computing the partition function is usually out of reach both analytically and numerically (only small systems are tractable)
- A better algorithm can reach larger sizes and consequently shed more light on the phenomena relevant at infinite size
- The transfer-matrix is a simple but efficient method to compute exactly partition functions for finite systems

Outline

The Potts model as a *spin* model

Given a graph $G = (V, E)$, we consider the set of colorings:

$$\sigma : V \longrightarrow [1, Q] \qquad Q \in \mathbb{N} \text{ (colors)}$$

each of them are assigned an *energy*

$$\mathcal{H}(\sigma) = -K \sum_{(ij)\in E} \delta(\sigma_i, \sigma_j) \qquad K \in \mathbb{R} \text{ (coupling)}$$

so the partition function is given by

$$Z_G(Q, K) = \sum_{\sigma} e^{-\mathcal{H}(\sigma)} = \sum_{\sigma} \prod_{(ij)\in E} e^{K\,\delta(\sigma_i,\sigma_j)}$$

The Potts model as a *cluster* model

We defined the model in terms of *spins*, but the same model can be view as a geometrical model

Fortuin-Kasteleyin representation

Rewriting $e^{K\,\delta(\sigma_i,\sigma_j)} = 1 + v\,\delta(\sigma_i,\sigma_j)$, we have

$$Z_G(Q,v) \;=\; \sum_{A \subseteq E} v^{|A|}\, Q^{k(A)}$$

"Objects" carrying enery are no longer localised but extended

Tutte polynomial

$Z_G(Q, v)$ is equivalent to the Tutte polynomial

$$\begin{aligned} T_G(x,y) &= \sum_{A \subseteq E} (x-1)^{r(E)-r(A)} (y-1)^{|A|-r(A)} \\ &\propto Z_G((x-1)(y-1), y-1) \end{aligned}$$

where $r(A) = |V| - k(A)$ is the rank of subgraph A.

Counting proper colorings

In the limit $K \to \infty$ (or $v = -1$) non proper colorings get weight zero and proper Q-colourings contribute with weight one.

Chromatic polynomial

$$\chi_G(Q) = Z_G(Q, v = -1) = \sum_{A \subseteq E} (-1)^{|A|} \, Q^{k(A)}$$

Complexity classes for counting problems

#P is the class of enumeration problems in which structures being counter are recognisable in polynomial time.

$A \in$ #P-complete if $\forall B \in \#P$ then $B \leq_P A$

$A \in$ #P-hard if $\exists B \in$ #P-complete s.t. $B \leq_P A$

Jaeger et al, 1990

Computing $Z_G(Q, v)$ is #P-hard except few exceptional points in the (Q, v) plane.

In practice ...

- The previously best known algorithm is due to Haggard, Pearce and Royle (2008)
- It uses an optimized deletion/contraction recursion

$$Z_G(Q, v) = Z_{G\setminus e}(Q, v) + v\, Z_{G/e}(Q, v)$$

where $G \setminus e$ is the graph obtained by *deleting* the edge e
and G/e is the graph obtained by *contracting* e.

- It runs in exponential time and takes $\sim$ 10s to deal with a planar graph of 40 vertices

Outline

Basic ideas

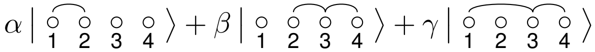
$$Z_G(Q,v) = \sum_{A \subseteq E} v^{|A|} \, Q^{k(A)}$$

- The sum is constructed iteratively by the action of linear operators.
- These operators act on “states”, properly weighted super-imposition of partially built configurations.
- When all possible configurations of a part of G have been elaborated, we forget their state and re-sum all the information into the weights.

Definitions for the Potts model

$$Z_G(Q, v) = \sum_{A \subseteq E} v^{|A|} Q^{k(A)}$$

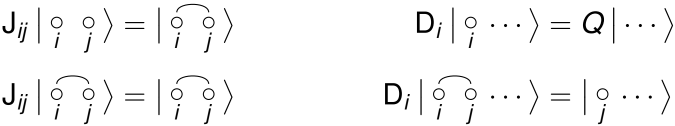
To keep track of k the state will be linear combinations of vertex partitions (non-crossing if G is planar)



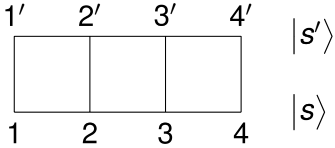
The number of partitions is the Catalan number $C_N = \frac{1}{N+1}\binom{2N}{N}$

$\sim \frac{4^N}{N^{3/2}\pi}$ if planar and the Bell number B_N otherwise.

We will act on these states with the operators:



If G has a layer structure then $|s'\rangle = \mathsf{T}|s\rangle$ where



$$\mathsf{T} = \prod_i \mathsf{D}_i(1 + v\,\mathsf{J}_{i,i'})(1 + v\,\mathsf{J}_{i,i+1})$$

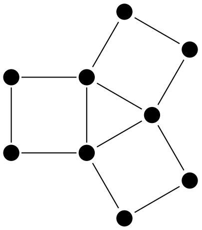
The same procedure can be implemented for general graphs.

- We fix the order in which *process* vertices
- To *process a vertex i* we first process all its incident edges and then we delete it with D_i.
- To *process an edge* (ij) we act with $(1 + v\,J_{ij})$
- New vertices are inserted into partitions as needed

$$D_i \prod_{j \sim i} (1 + v\,J_{ij})$$

Outline

Example



$$\left|s'\right\rangle = D_1 \, (1 + vJ_{12}) \, (1 + vJ_{13}) \left| \underset{1}{\circ} \ \underset{2}{\circ} \ \underset{3}{\circ} \right\rangle$$

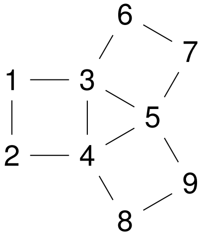
$$= (Q + 2v) \left| \underset{2}{\circ} \ \underset{3}{\circ} \right\rangle + v^2 \left| \overset{\frown}{\underset{2}{\circ} \ \underset{3}{\circ}} \right\rangle$$

$$\left|s''\right\rangle = D_2 \, (1 + vJ_{24}) \left| s' \ \underset{4}{\circ} \right\rangle$$

$$= (\ldots) \left| \underset{3}{\circ} \ \underset{4}{\circ} \right\rangle + (\ldots) \left| \overset{\frown}{\underset{3}{\circ} \ \underset{4}{\circ}} \right\rangle$$

$$\left|s'''\right\rangle = \ldots$$

Example



$$\left|s'\right\rangle = D_1\,(1+vJ_{12})\,(1+vJ_{13})\left|\underset{1}{\circ}\;\underset{2}{\circ}\;\underset{3}{\circ}\right\rangle$$

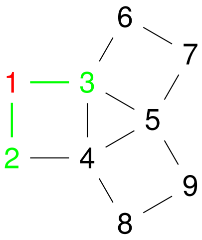
$$= (Q+2v)\left|\underset{2}{\circ}\;\underset{3}{\circ}\right\rangle + v^2\left|\overset{\frown}{\underset{2}{\circ}\;\underset{3}{\circ}}\right\rangle$$

$$\left|s''\right\rangle = D_2\,(1+vJ_{24})\left|s'\;\underset{4}{\circ}\right\rangle$$

$$= (\ldots)\left|\underset{3}{\circ}\;\underset{4}{\circ}\right\rangle + (\ldots)\left|\overset{\frown}{\underset{3}{\circ}\;\underset{4}{\circ}}\right\rangle$$

$$\left|s'''\right\rangle = \ldots$$

Example



$$|s'\rangle = D_1\,(1+vJ_{12})\,(1+vJ_{13})\,\big|\underset{1}{\circ}\;\underset{2}{\circ}\;\underset{3}{\circ}\big\rangle$$

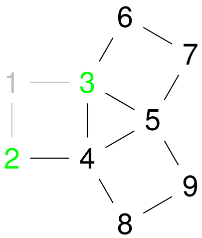
$$= (Q+2v)\,\big|\underset{2}{\circ}\;\underset{3}{\circ}\big\rangle + v^2\,\big|\overset{\frown}{\underset{2}{\circ}\;\underset{3}{\circ}}\big\rangle$$

$$|s''\rangle = D_2\,(1+vJ_{24})\,\big|s'\;\underset{4}{\circ}\big\rangle$$

$$= (\dots)\,\big|\underset{3}{\circ}\;\underset{4}{\circ}\big\rangle + (\dots)\,\big|\overset{\frown}{\underset{3}{\circ}\;\underset{4}{\circ}}\big\rangle$$

$$|s'''\rangle = \dots$$

Example



$$|s'\rangle = D_1\,(1+vJ_{12})\,(1+vJ_{13})\,\big|\underset{1}{\circ}\ \underset{2}{\circ}\ \underset{3}{\circ}\big\rangle$$

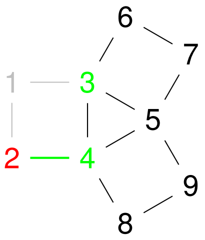
$$= (Q+2v)\,\big|\underset{2}{\circ}\ \underset{3}{\circ}\big\rangle + v^2\,\big|\overset{\frown}{\underset{2}{\circ}\ \underset{3}{\circ}}\big\rangle$$

$$|s''\rangle = D_2\,(1+vJ_{24})\,\big|s'\ \underset{4}{\circ}\big\rangle$$

$$= (\ldots)\,\big|\underset{3}{\circ}\ \underset{4}{\circ}\big\rangle + (\ldots)\,\big|\overset{\frown}{\underset{3}{\circ}\ \underset{4}{\circ}}\big\rangle$$

$$|s'''\rangle = \ldots$$

Example



$$\left|s'\right\rangle = D_1\,(1 + vJ_{12})\,(1 + vJ_{13})\left|\underset{1}{\circ}\;\underset{2}{\circ}\;\underset{3}{\circ}\right\rangle$$

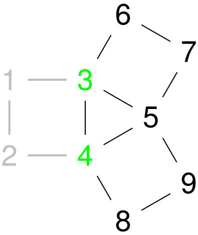
$$= (Q + 2v)\left|\underset{2}{\circ}\;\underset{3}{\circ}\right\rangle + v^2\left|\overset{\frown}{\underset{2}{\circ}\;\underset{3}{\circ}}\right\rangle$$

$$\left|s''\right\rangle = D_2\,(1 + vJ_{24})\left|s'\;\underset{4}{\circ}\right\rangle$$

$$= (\ldots)\left|\underset{3}{\circ}\;\underset{4}{\circ}\right\rangle + (\ldots)\left|\overset{\frown}{\underset{3}{\circ}\;\underset{4}{\circ}}\right\rangle$$

$$\left|s'''\right\rangle = \ldots$$

Example



$$\left|s'\right\rangle = D_1\,(1+vJ_{12})\,(1+vJ_{13})\left|\underset{1}{\circ}\;\underset{2}{\circ}\;\underset{3}{\circ}\right\rangle$$

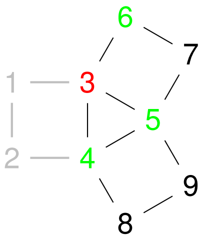
$$= (Q+2v)\left|\underset{2}{\circ}\;\underset{3}{\circ}\right\rangle + v^2\left|\overset{\frown}{\underset{2}{\circ}\;\underset{3}{\circ}}\right\rangle$$

$$\left|s''\right\rangle = D_2\,(1+vJ_{24})\left|s'\;\underset{4}{\circ}\right\rangle$$

$$= (\dots)\left|\underset{3}{\circ}\;\underset{4}{\circ}\right\rangle + (\dots)\left|\overset{\frown}{\underset{3}{\circ}\;\underset{4}{\circ}}\right\rangle$$

$$\left|s'''\right\rangle = \dots$$

Example



$$\left|s'\right\rangle = D_1\,(1 + vJ_{12})\,(1 + vJ_{13})\left|\underset{1}{\circ}\ \underset{2}{\circ}\ \underset{3}{\circ}\right\rangle$$

$$= (Q + 2v)\left|\underset{2}{\circ}\ \underset{3}{\circ}\right\rangle + v^2\left|\overset{\frown}{\underset{2}{\circ}\ \underset{3}{\circ}}\right\rangle$$

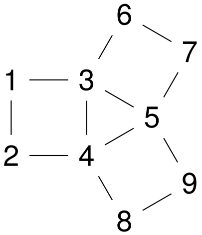
$$\left|s''\right\rangle = D_2\,(1 + vJ_{24})\left|s'\ \underset{4}{\circ}\right\rangle$$

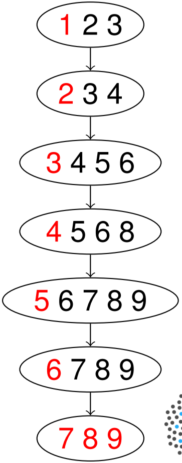
$$= (\ldots)\left|\underset{3}{\circ}\ \underset{4}{\circ}\right\rangle + (\ldots)\left|\overset{\frown}{\underset{3}{\circ}\ \underset{4}{\circ}}\right\rangle$$

$$\left|s'''\right\rangle = \ldots$$

Outline

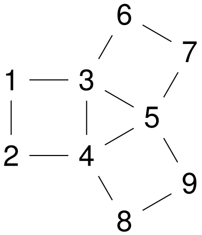
Time decomposition

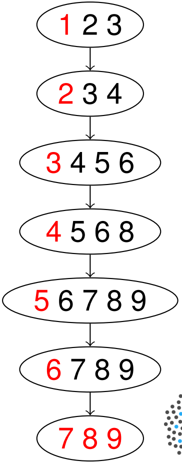




- The ordering defines a “time decomposition” in slices we call *bags*
- Time and memory requirements scale exponentially with the maximum bag size k.
- It happens to be a particular case of a more general construction

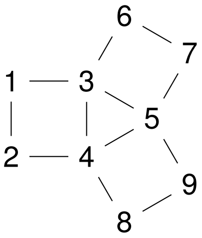
Time decomposition

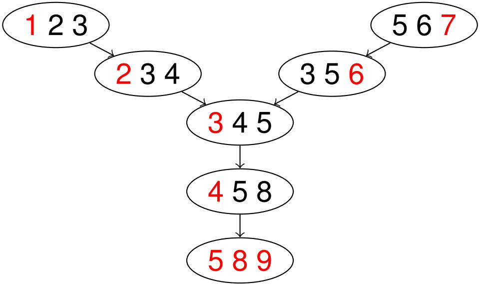




- The ordering defines a "time decomposition" in slices we call *bags*
- Time and memory requirements scale exponentially with the maximum bag size k.
- It happens to be a particular case of a more general construction

Tree decomposition

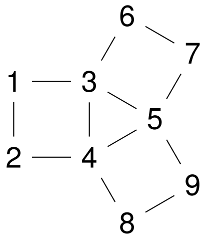


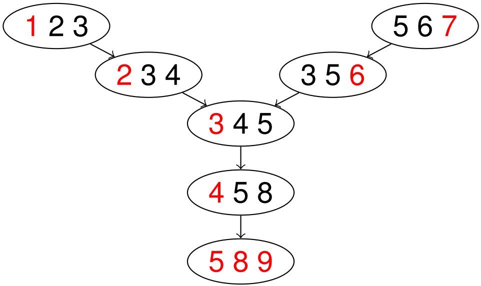


It is a collections of *bags* organised in a tree.

- $\forall i \in V$, there exists a bag containing i
- $\forall (ij) \in E$, there exists a bag containing both i and j
- $\forall i \in V$, the set of bags containing i is connected in the tree
- The *treewidth* k is the maximum bag size.

Tree decomposition





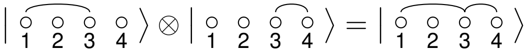
- Tree decomposition can have smaller bags therefore an *exponentially* smaller state space (C_k)
- Finding an optimal tree decompositions is NP-hard
- Heuristic algorithms give reasonably good decompositions in linear time

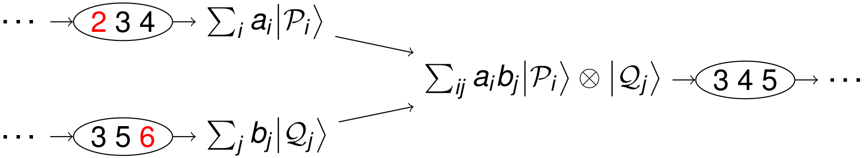
The fusion procedure

When a bag has several children, we need to “fuse” different timelines.
Given two partitions $|\mathcal{P}_1\rangle$ and $|\mathcal{P}_2\rangle$, we define

$$|\mathcal{P}_1\rangle \otimes |\mathcal{P}_2\rangle = |\mathcal{P}_1 \vee \mathcal{P}_2\rangle$$

Exemple:





This is a quadratic operation requiring time $\sim O(C_k^2)$

Complexity

- The planar separator theorem gives an upper bound on treewidth k of a planar graph:

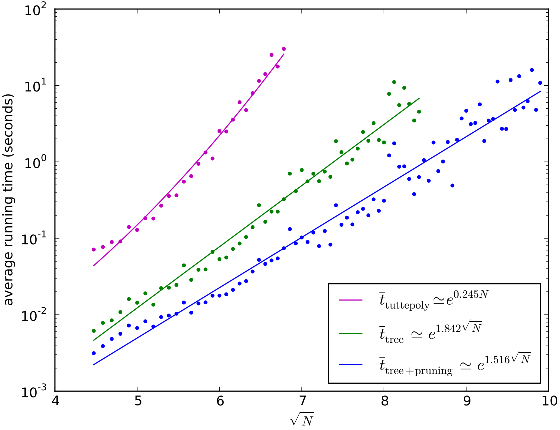
$$k < \alpha\sqrt{N} \qquad (\alpha < 3.182)$$

- The algorithm requires time $O(C_k^2) \simeq 16^k$
- This implies a sub-exponential upper bound for the running time

$$\mathrm{t} < 16^{3.182\sqrt{N}} = e^{8.222\sqrt{N}}$$

- It's the natural generalization of the traditional TM whose requirements scale as C_L, where the side $L \simeq \sqrt{N}$

Performances

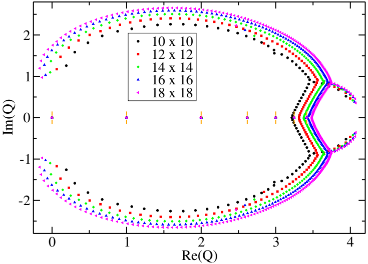


on an uniform sample of planar graphs

Outline

Chromatic roots – regular lattice

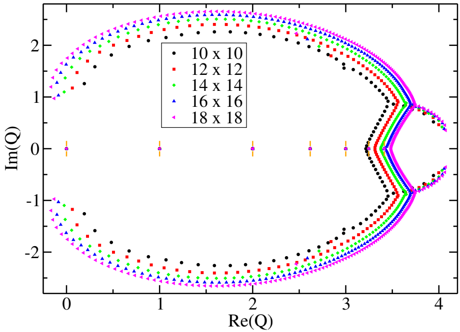
- Regular lattices have chromatic roots close to the Beraha numbers $B_k = 4\cos^2(\pi/k)$ up to a lattice specific limit

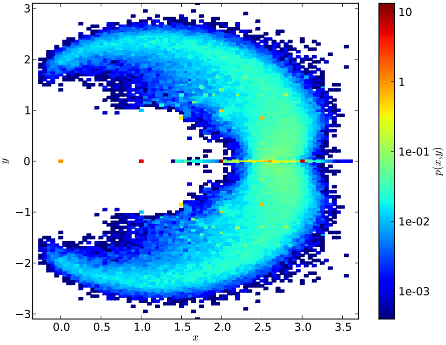


- We also know that chromatic roots are dense in $\mathbb{C}$
- Little is known about the roots of the *typical* planar graph

Chromatic roots – random planar

- We sampled 2500 planar graphs with $N = 100$ and for each of them we computed the chromatic polynomial and its roots





Outlook

In progress:

- Adapt the same algorithm to different graph models (hamiltonian walks, longest-path, vertex covering, maximum-biconnected subgraph, etc)
- Better understanding of the scaling of the treewidth and its heuristic approximations (hint: $\langle k \rangle$ scales as $N^{0.3} < N^{1/2}$)
- Look at other families of planar graphs (2-, 3-connected)

Further reading:

AB, J.L. Jacobsen, *J. Phys. A: Math. Theor.* **43**, 385001, 2010