Counting Independent Sets up to the Tree Threshold

Dror Weitz
Tel Aviv

AISP, Santa Fe
May 2007

What is this work about?

Novel exact tree representation for the marginal probability at a vertex in any binary spin system.

➡ The regular tree is the worst-case graph for an appropriate notion of spatial decay of correlations (Strong Spatial Mixing).

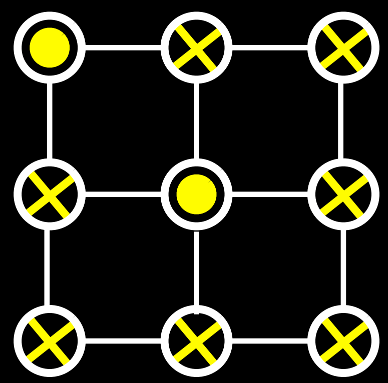
➡ New efficient algorithm for approximating marginals (and hence the partition function) in the regime where the regular tree exhibits SSM.

- Strong application: hard-core model (independent sets).

The Hard-Core Model (Independent Sets)

- Count/sample weighted independent sets of a graph G.
- Weights are determined by an activity parameter λ:

$$w(I) = \lambda^{|I|}$$



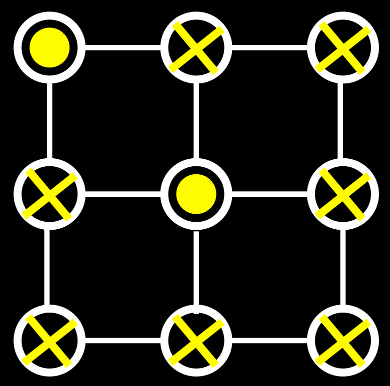
- Occupied vertex

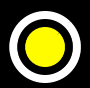
- Unoccupied vertex

The Hard-Core Model (Independent Sets)

- Count/sample weighted independent sets of a graph G.
- Weights are determined by an activity parameter λ:

$$w(I) = \lambda^{|I|}$$



 - Occupied vertex

 - Unoccupied vertex

- Model for lattice gas, communication networks, ...

The Hard-Core Model (Independent Sets)

- Count/sample weighted independent sets of a graph G.
- Weights are determined by an activity parameter λ:

$$w(I) = \lambda^{|I|}$$

- Model for lattice gas, communication networks, ...

Computational Problem

- Aim: $(1+\epsilon)$-approximation of the partition function -

$$Z \equiv Z_G^\lambda = \sum_I \lambda^{|I|}$$

Equivalently: approximately sample independent sets where $\Pr(I) = \lambda^{|I|}/Z$.

Computational Problem

- Aim: $(1+\epsilon)$-approximation of the partition function -

$$Z \equiv Z_G^\lambda = \sum_I \lambda^{|I|}$$

Equivalently: approximately sample independent sets where $\Pr(I) = \lambda^{|I|}/Z$.

- Intuitively, the problem becomes harder as λ grows.
(Sampling with large λ will output a maximum ind. set.)

Known bounds

- NP-hard to approximate Z within a polynomial factor for: max degree Δ and $\lambda \geq c/\Delta$, where c is a (large enough) constant. [Luby-Vigoda]

Known bounds

- NP-hard to approximate Z within a polynomial factor for: max degree Δ and $\lambda \geq c/\Delta$, where c is a (large enough) constant. [Luby-Vigoda]

- FPRAS exists for (based on the Glauber dynamics) -

  easy: $\lambda \leq \frac{1}{\Delta - 1}$ (Dobrushin's uniqueness condition)

  best: $\lambda \leq \frac{2}{\Delta - 2}$ [Dyer-Greenhill, Vigoda]

Known bounds

- NP-hard to approximate Z within a polynomial factor for: max degree Δ and $\lambda \geq c/\Delta$, where c is a (large enough) constant. [Luby-Vigoda]

- FPRAS exists for (based on the Glauber dynamics) -

  easy: $\lambda \leq \frac{1}{\Delta - 1}$ (Dobrushin's uniqueness condition)

  best: $\lambda \leq \frac{2}{\Delta - 2}$ [Dyer-Greenhill, Vigoda]

- Finding out exact constants is important -

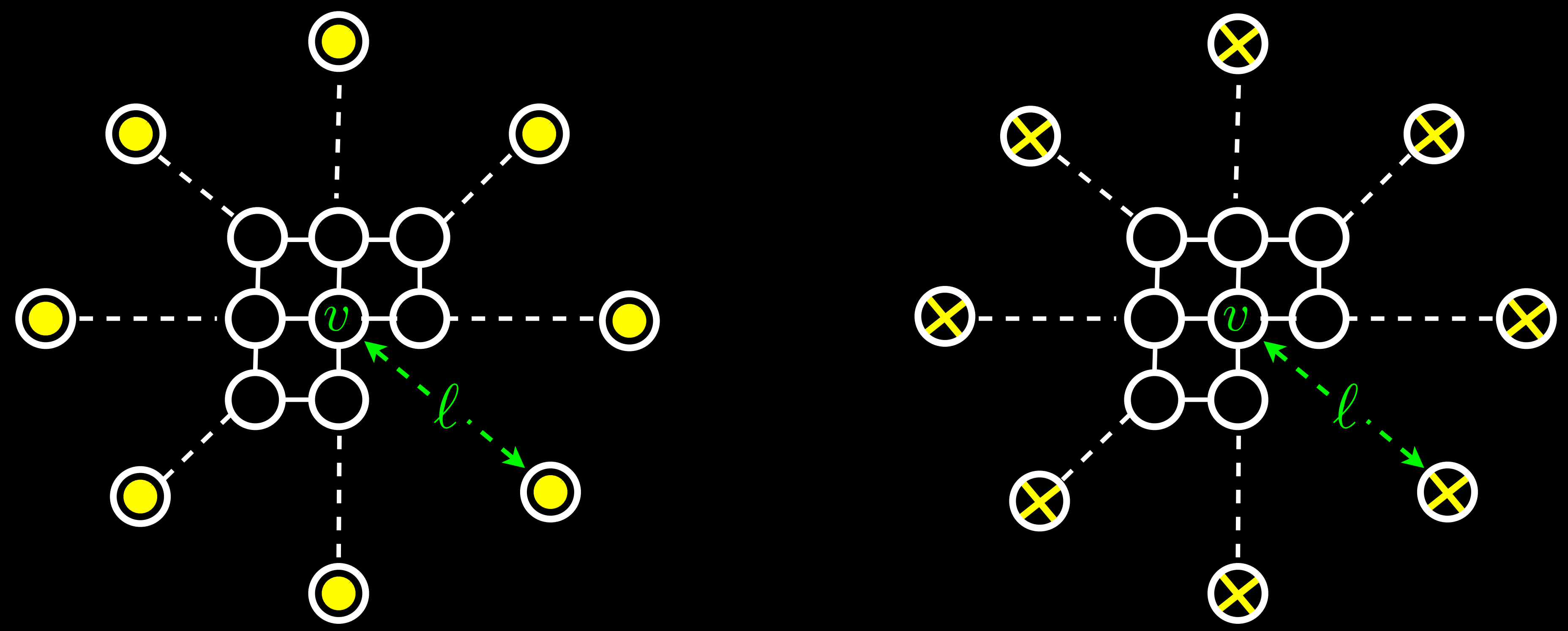
  most interesting graphs are low dimensional lattices.

Combinatorial Problem

For what values of λ is the 'Gibbs' measure unique?

uniqueness of Gibbs measure:

$$|\Pr(v \text{ is occupied} \,|\, \sigma_\ell) - \Pr(v \text{ is occupied} \,|\, \tau_\ell)| \underset{\ell\to\infty}{\longrightarrow} 0$$



Uniqueness for General Graphs

For what values of λ is there a decaying rate $\delta(\ell) \underset{\ell \to \infty}{\longrightarrow} 0$

such that for every graph G of maximum degree Δ

and every $v \in G$,

$$|\Pr(v \text{ is occupied} \,|\, \sigma_\ell) - \Pr(v \text{ is occupied} \,|\, \tau_\ell)| \;\le\; \delta(\ell)$$

?

Known Bounds

- Gibbs measure is unique on all graphs of maximum degree Δ for $\lambda < \frac{2}{\Delta-2}$. [Vigoda]

Same bound as the algorithmic one; uses essentially the same argument. (Part of a general correspondence between computational complexity and decay of correlations in the Gibbs distribution.)

Known Bounds

- Gibbs measure is unique on all graphs of maximum degree Δ for $\lambda < \frac{2}{\Delta-2}$. [Vigoda]

- On the Δ-regular tree, Gibbs measure is unique if and only if $\lambda \le \lambda_c = \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^{\Delta}} \quad \left(\ge \frac{e}{\Delta-2}\right)$.

Algorithmic implications: although it is easy to count independent sets of the tree for arbitrary λ, arguments that imply uniqueness are bound to fail above λ_c .

Known Bounds

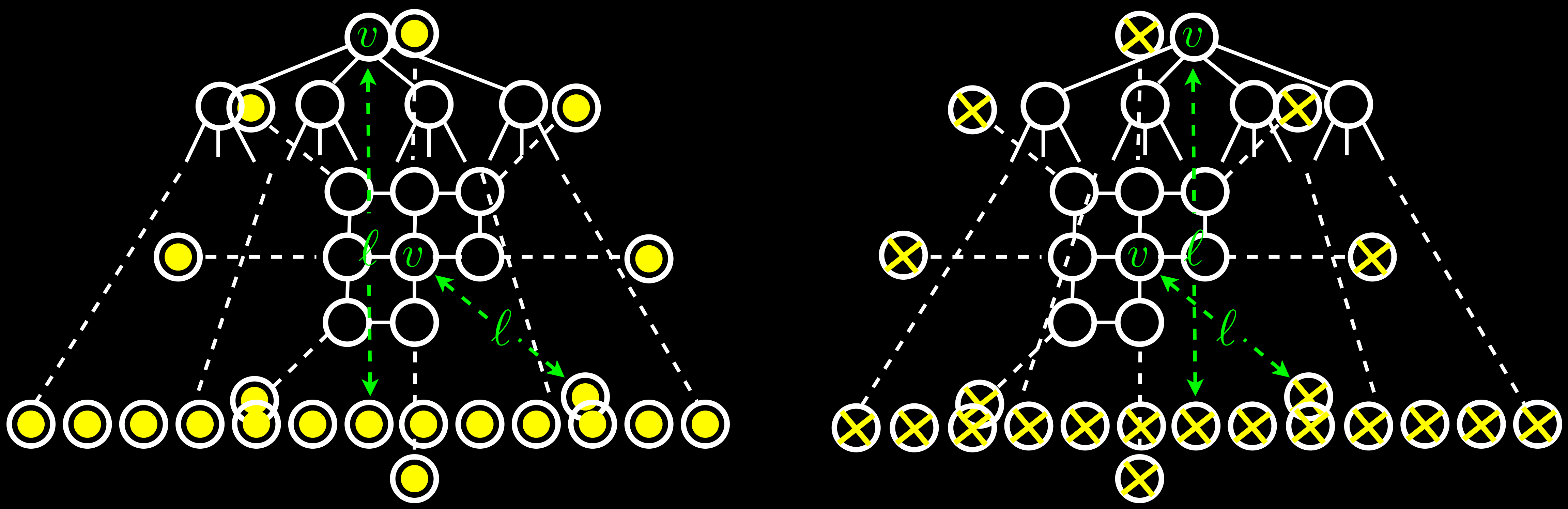
- Gibbs measure is unique on all graphs of maximum degree Δ for $\lambda < \frac{2}{\Delta-2}$. [Vigoda]

- On the Δ-regular tree, Gibbs measure is unique if and only if $\lambda \leq \lambda_c = \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^{\Delta}} \quad \left(\geq \frac{e}{\Delta-2}\right)$.

- Conjecture [Sokal]: the tree is the worst case - uniqueness on all graphs for $\lambda \leq \lambda_c$.

Main Result

Theorem: Fix Δ and λ. For a general graph G of maximum degree Δ, consider the influence of placing conditions at any given distance. This influence is maximized by taking G to be the regular tree.



Main Result

Theorem: Fix Δ and λ. For a general graph G of maximum degree Δ, consider the influence of placing conditions at any given distance. This influence is maximized by taking G to be the regular tree.

Corollary: The Gibbs measure is unique for all graphs of maximum degree Δ and $\lambda \leq \lambda_c = \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^{\Delta}}$.

Algorithmic Implications

<u>**New algorithm**</u>: fix Δ and $\lambda < \lambda_c$; deterministic $(1+\epsilon)$-approximation for any graph of max degree Δ in time $\mathrm{poly}(n, 1/\epsilon)$.
(degree of poly depends on Δ and λ.)

<u>**Corollary:**</u> For all graphs of *'sub-exponential growth'* and $\lambda < \lambda_c$ the Glauber dynamics is rapidly mixing.
$(\Rightarrow \mathbf{FPRAS})$

Interesting Specific Cases

- Uniformly weighted independent sets ($\lambda = 1$):
  - New: efficient approximation scheme for $\Delta \leq 5$.
  - Previous bound is $\Delta \leq 4$.
  - Believed to be hard for $\Delta \geq 6$.
  - First <u>deterministic</u> approx scheme for #P-complete problem.

Interesting Specific Cases

- Uniformly weighted independent sets $(\lambda = 1)$:
  - New: efficient approximation scheme for $\Delta \leq 5$.
  - Previous bound is $\Delta \leq 4$.
  - Believed to be hard for $\Delta \geq 6$.
  - First deterministic approx scheme for #P-complete problem.

- The sqaure lattice $\mathbb{Z}^2$:
  - Believed to have a critical activity at ~ 3.79.
  - Previously best known lower bound: 1.25 $\{1.45\}$ (site-perc.)
  - New bound: 1.6875.

Proof of Main Theorem

Theorem: Fix Δ and λ. For a general graph G of maximum degree Δ, consider the influence of placing conditions at any given distance. This influence is maximized by taking G to be the regular tree.

Part 1: prove the theorem when G is a general (irregular) tree.

In other words: on the regular tree SSM holds all the way up to the uniqueness threshold.

Tree Representation for General Graphs

Theorem: For every graph G and vertex $v \in G$ there exists a tree $T(G,v)$ of the same maximum degree such that

$$\Pr_G(v \text{ is occupied}) = \Pr_{T(G,v)}(\text{root is occupied}).$$

Tree Representation for General Graphs

Theorem: For every graph G and vertex $v \in G$ there exists a tree $T(G,v)$ of the same maximum degree such that

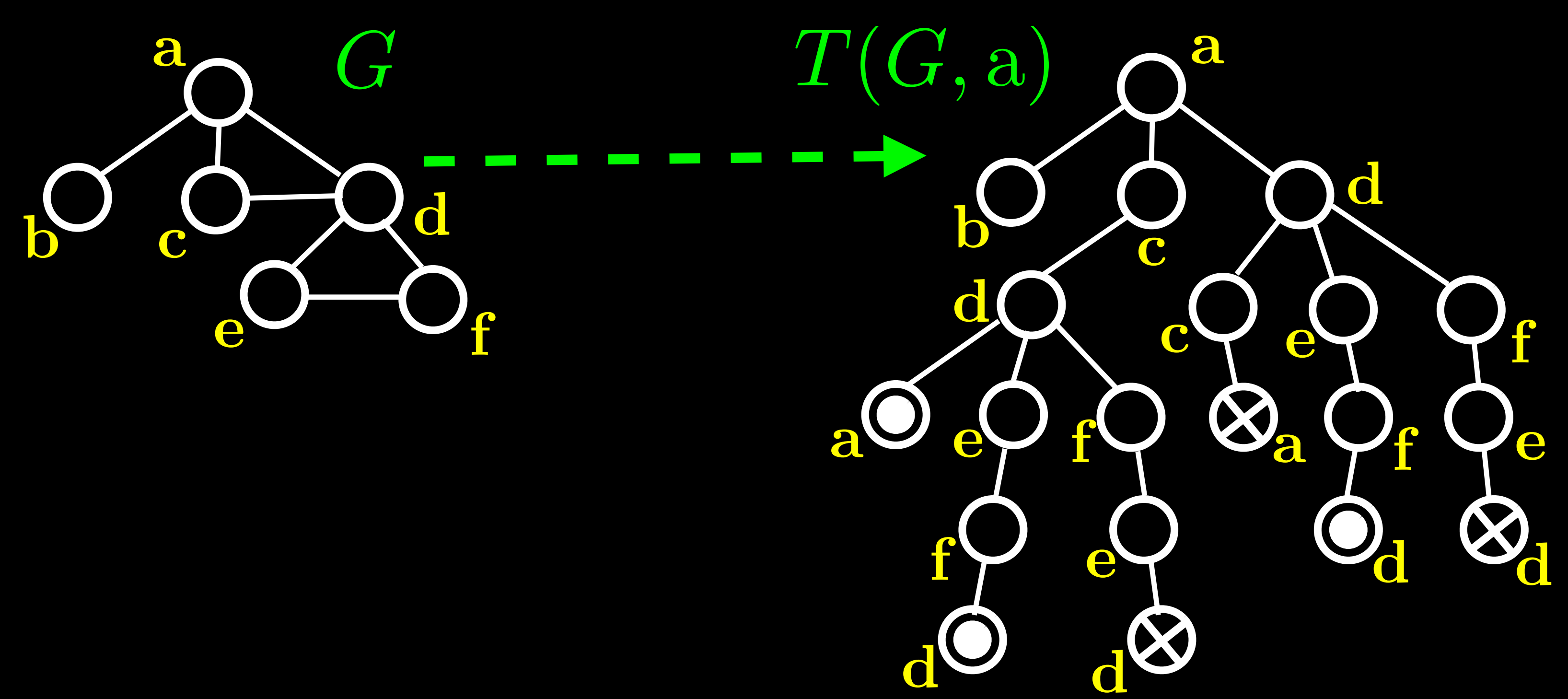
$$\Pr_G(v \text{ is occupied} \mid \sigma_\ell) \;=\; \Pr_{T(G,v)}(\text{root is occupied} \mid \widehat{\sigma_\ell}).$$

Furthermore, the correspondence (with the same tree) continues to hold when placing a condition on G (and a corresponding condition on $T(G,v)$).

Construction of $T(G, v)$

Similar to the tree of self-avoiding walks originating at v:

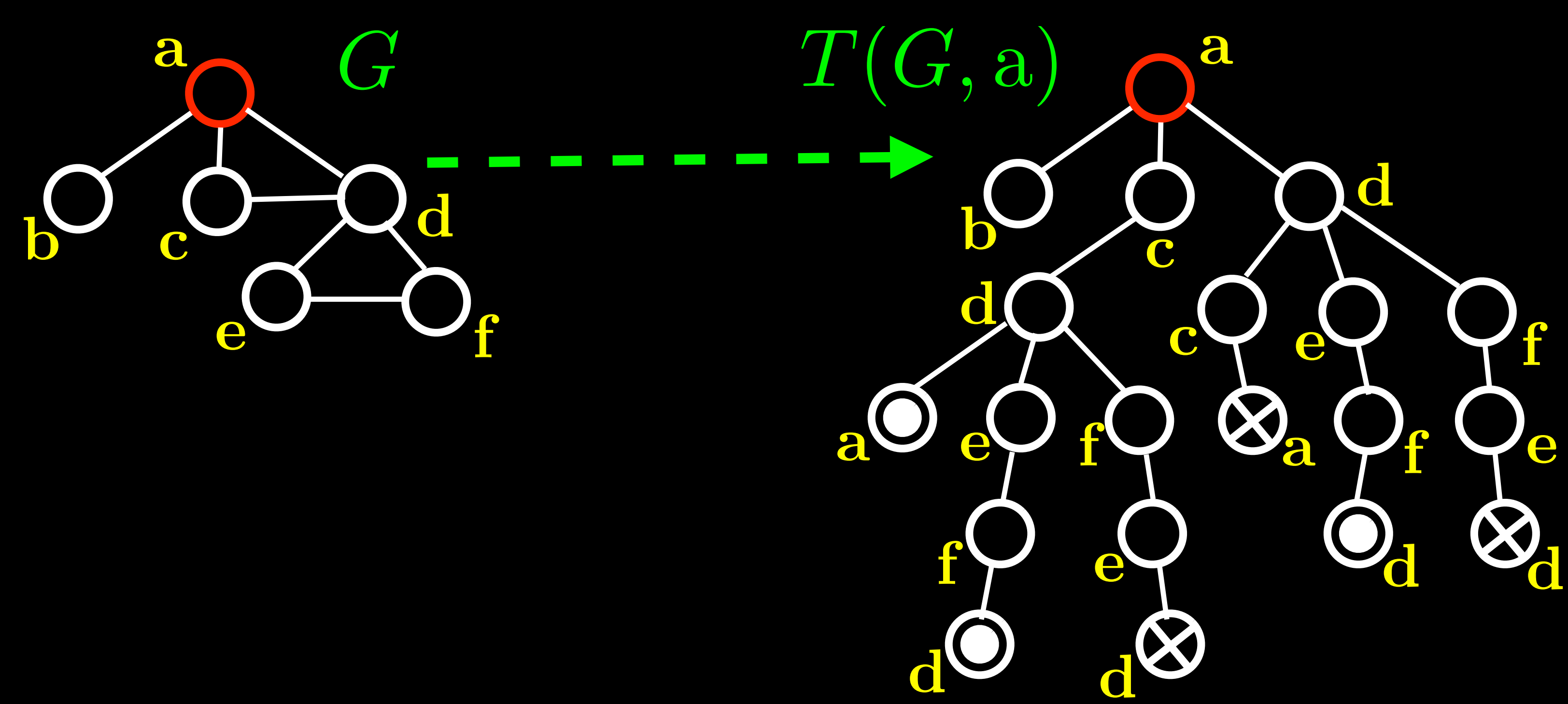
- order the neighbors of each vertex;
- construct the tree of paths originating at v;
- vertices that close cycles are fixed to be occupied or unoccupied (determined by the above ordering).



Construction of $T(G, v)$

Similar to the tree of self-avoiding walks originating at v:

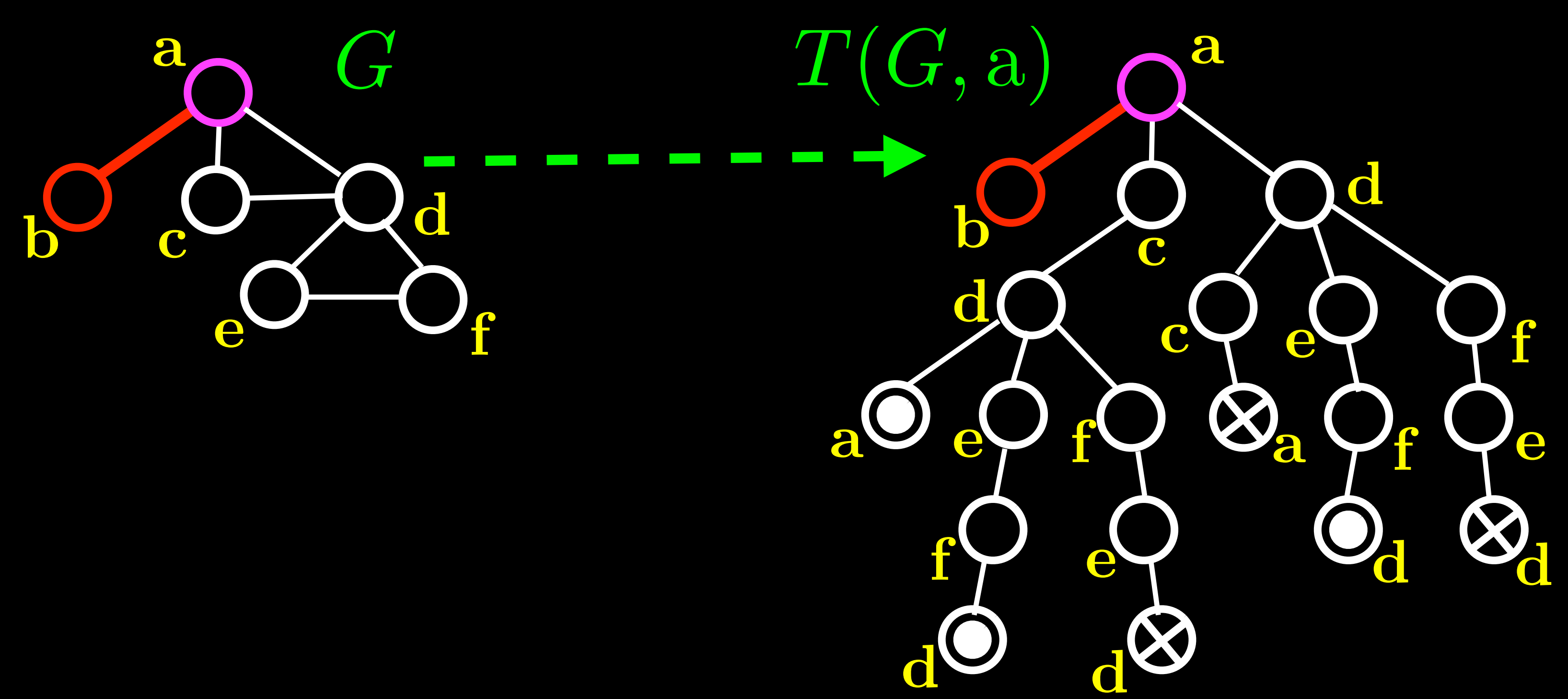
- order the neighbors of each vertex;
- construct the tree of paths originating at v;
- vertices that close cycles are fixed to be occupied or unoccupied (determined by the above ordering).



Construction of $T(G, v)$

Similar to the tree of self-avoiding walks originating at v:

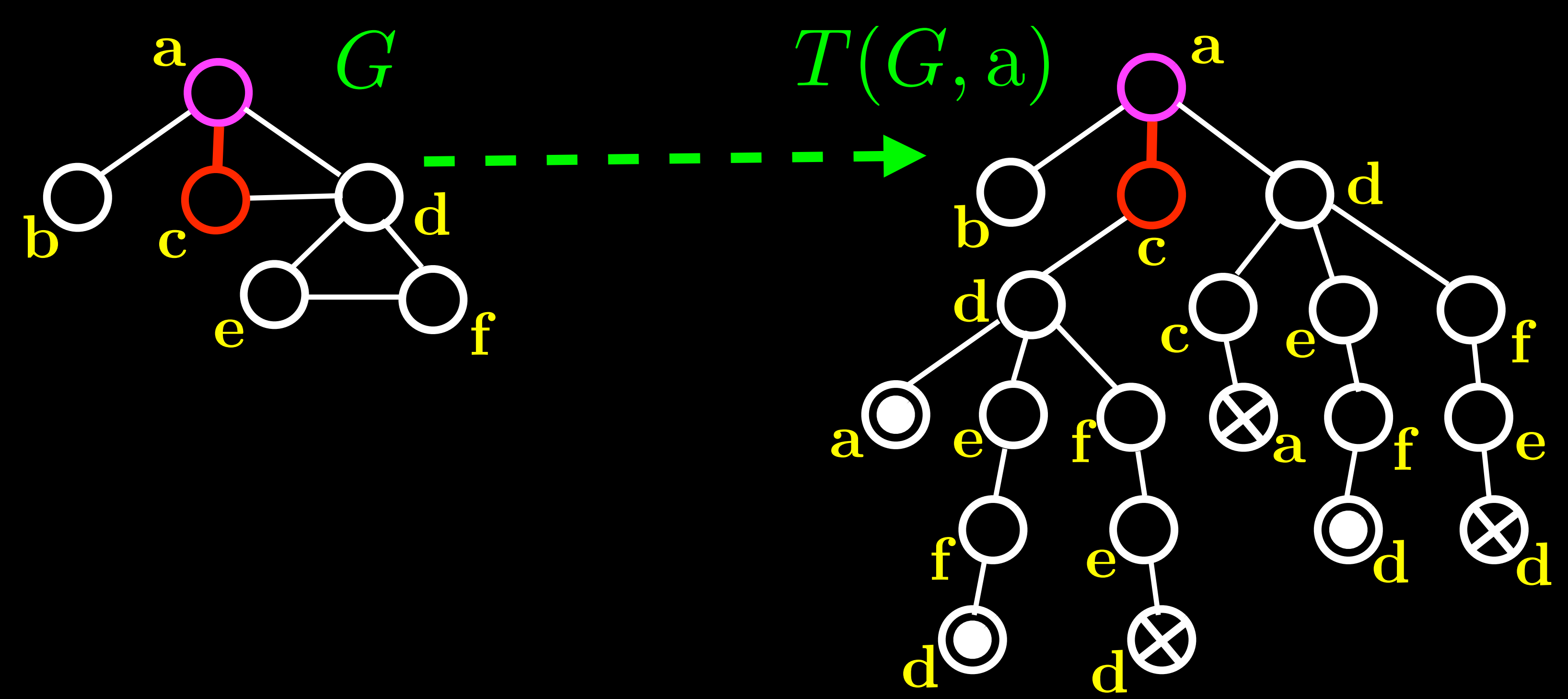
- order the neighbors of each vertex;
- construct the tree of paths originating at v;
- vertices that close cycles are fixed to be occupied or unoccupied (determined by the above ordering).



Construction of $T(G, v)$

Similar to the tree of self-avoiding walks originating at v:

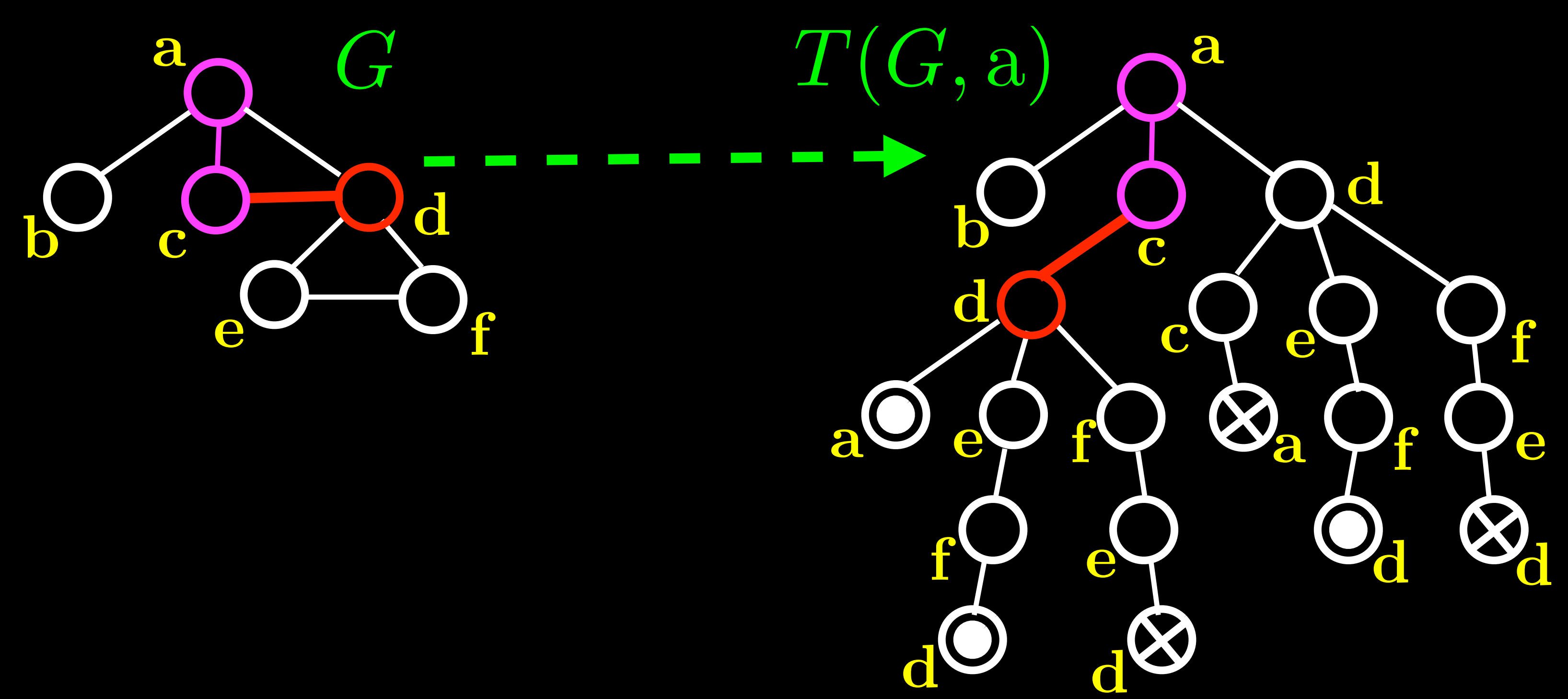
- order the neighbors of each vertex;
- construct the tree of paths originating at v;
- vertices that close cycles are fixed to be occupied or unoccupied (determined by the above ordering).



Construction of $T(G,v)$

Similar to the tree of self-avoiding walks originating at v:

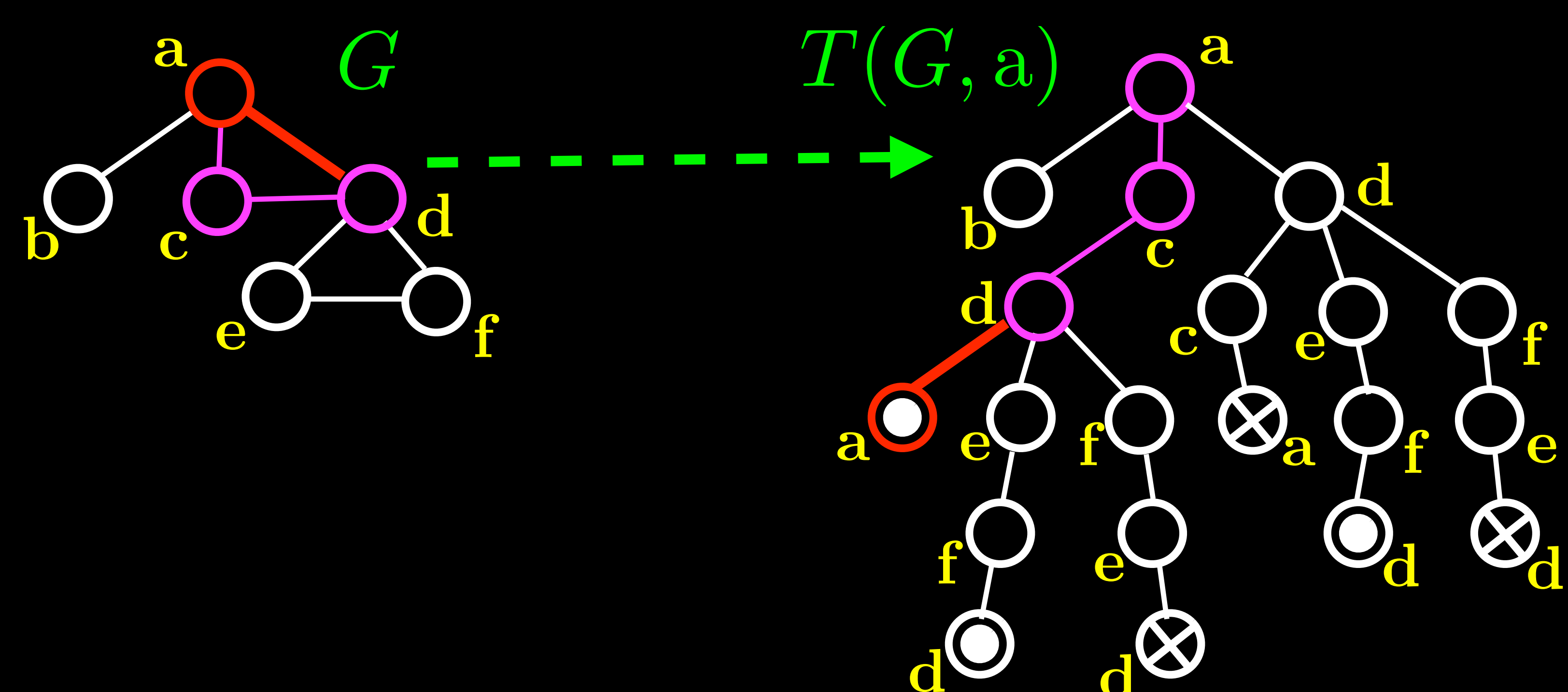
- order the neighbors of each vertex;
- construct the tree of paths originating at v;
- vertices that close cycles are fixed to be occupied or unoccupied (determined by the above ordering).



Construction of $T(G, v)$

Similar to the tree of self-avoiding walks originating at v:

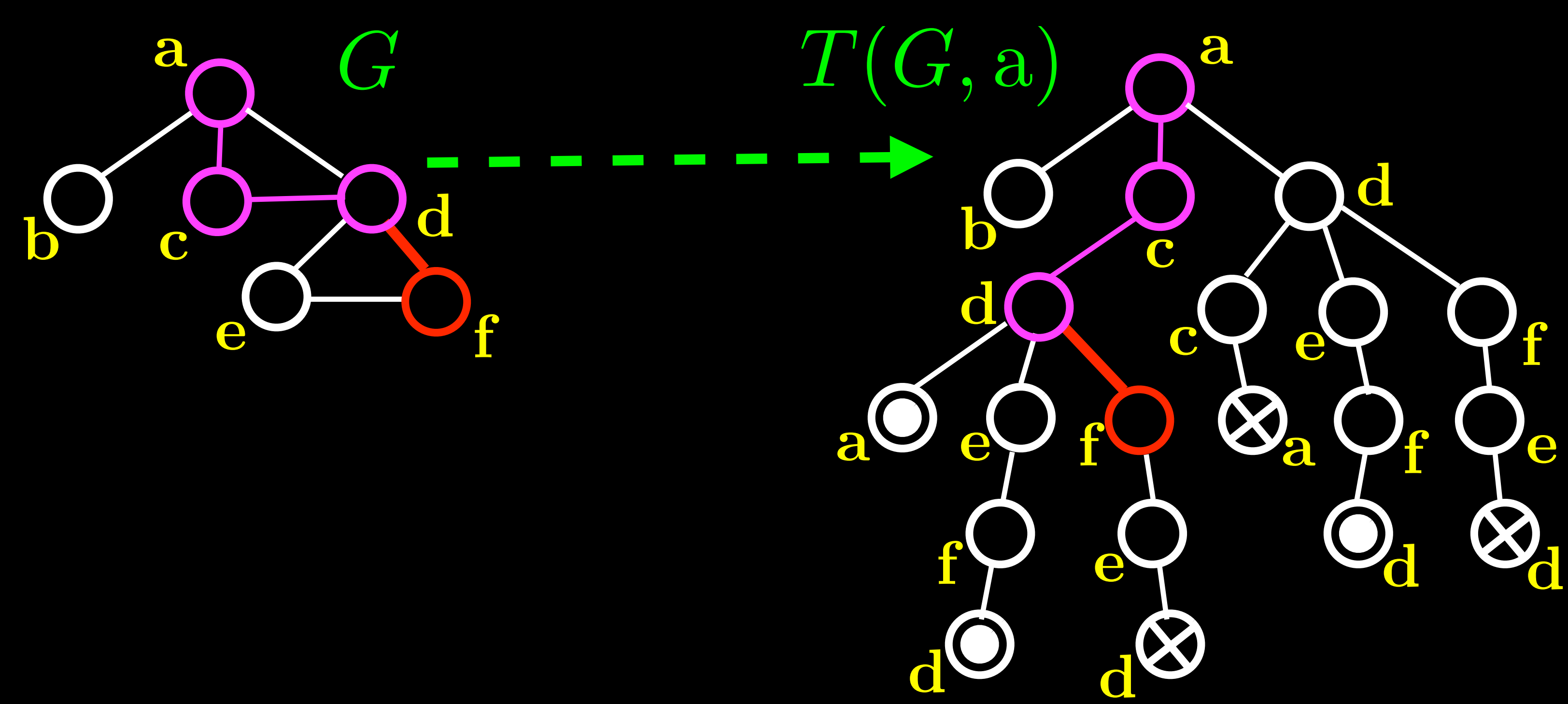
- order the neighbors of each vertex;
- construct the tree of paths originating at v;
- vertices that close cycles are fixed to be occupied or unoccupied (determined by the above ordering).



Construction of $T(G,v)$

Similar to the tree of self-avoiding walks originating at v:

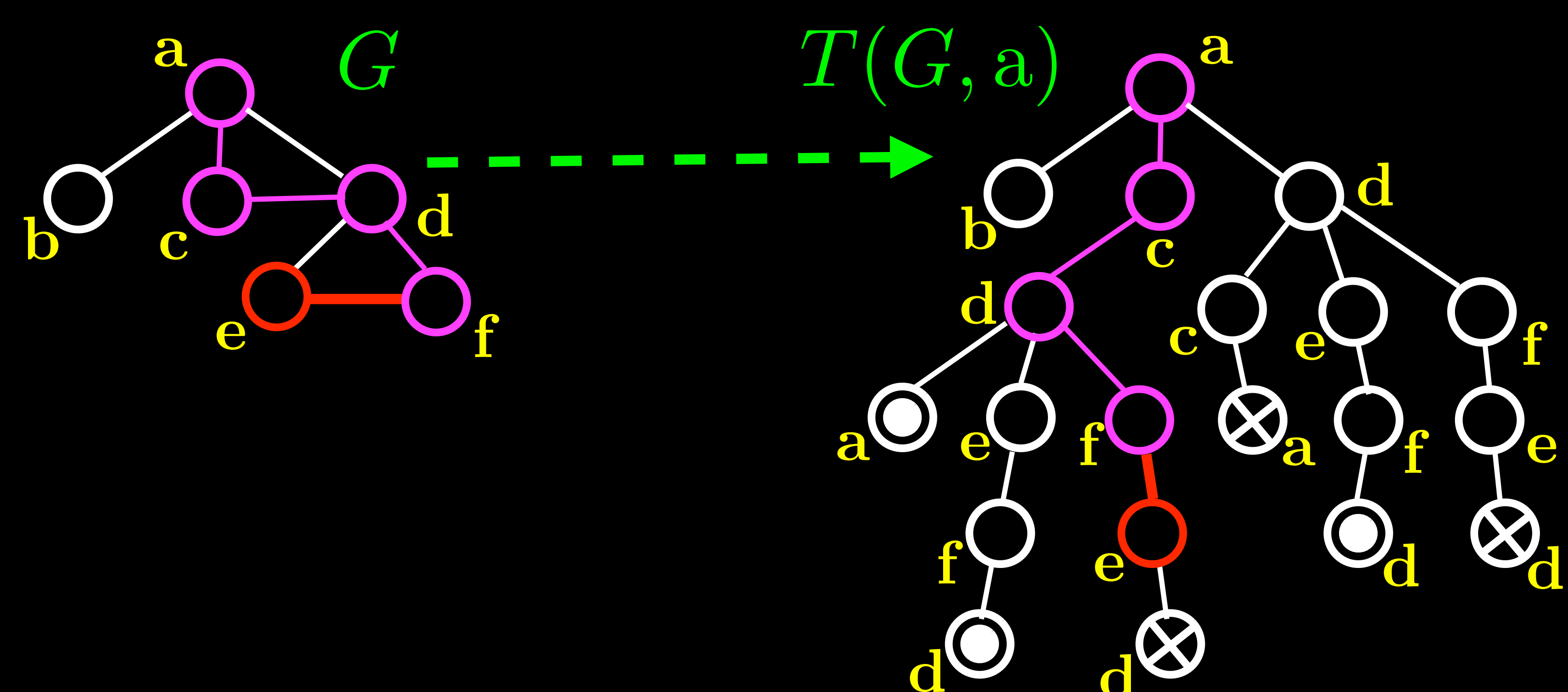
- order the neighbors of each vertex;
- construct the tree of paths originating at v;
- vertices that close cycles are fixed to be occupied or unoccupied (determined by the above ordering).



Construction of $T(G, v)$

Similar to the tree of self-avoiding walks originating at v:

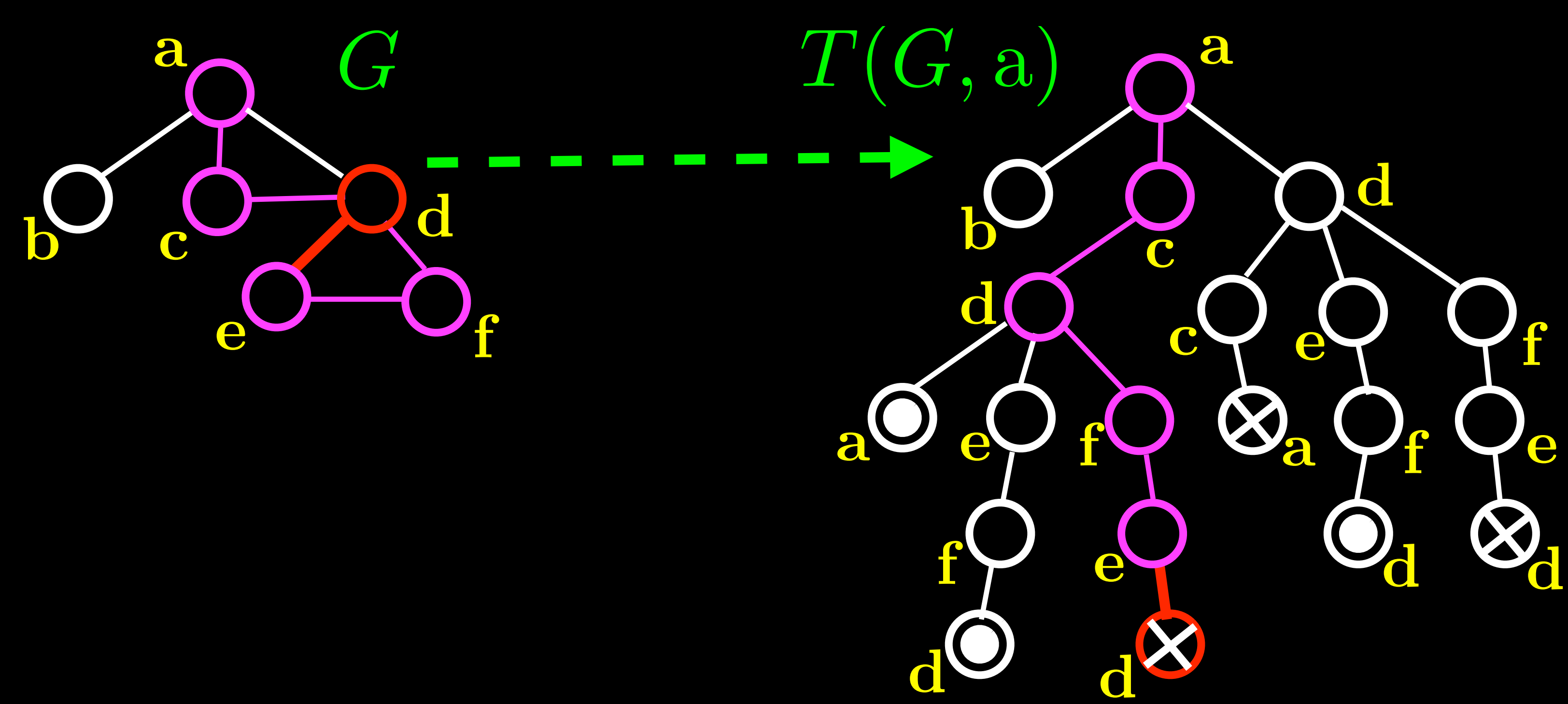
- order the neighbors of each vertex;
- construct the tree of paths originating at v;
- vertices that close cycles are fixed to be occupied or unoccupied (determined by the above ordering).



Construction of $T(G, v)$

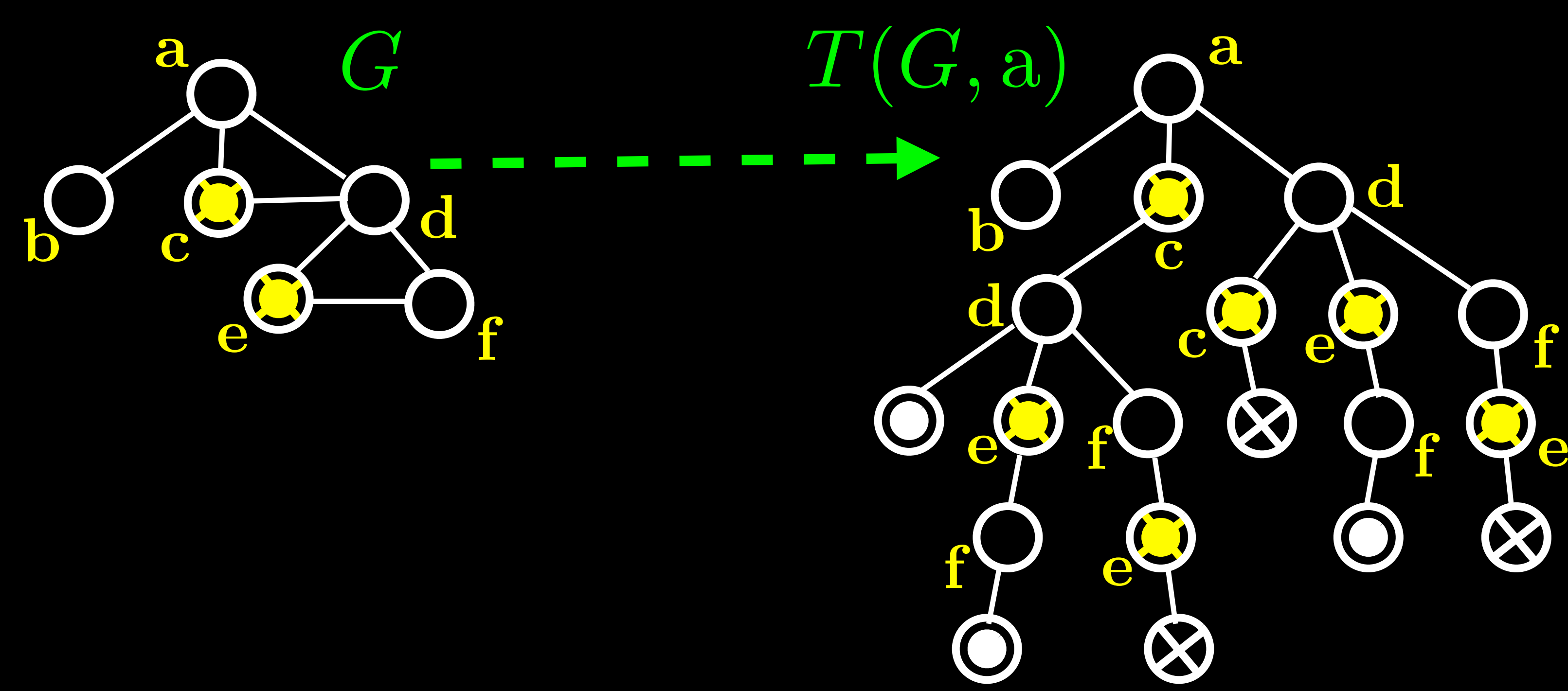
Similar to the tree of self-avoiding walks originating at v:

- order the neighbors of each vertex;
- construct the tree of paths originating at v;
- vertices that close cycles are fixed to be occupied or unoccupied (determined by the above ordering).



Construction of $T(G, v)$

Condition on G $\longrightarrow$ Condition on $T(G, v)$



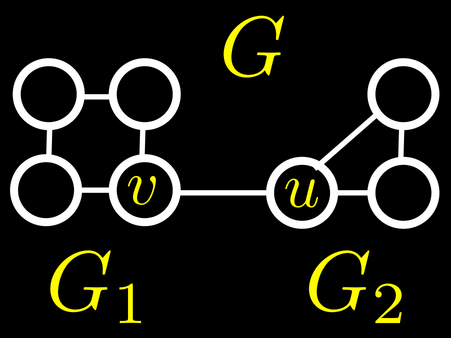
Calculating Pr(occupation)

- Notation: $R^{\sigma}_{G,v} = \dfrac{\Pr_G(v \text{ is occupied} \,|\, \sigma)}{\Pr_G(v \text{ is unoccupied} \,|\, \sigma)}$.

Calculating Pr(occupation)

- Notation: $R^{\sigma}_{G,v} \;=\; \dfrac{\Pr_G(v \text{ is occupied} \mid \sigma)}{\Pr_G(v \text{ is unoccupied} \mid \sigma)}\,.$
- Basic: when connection two separate graphs -

$$R_{G,v} \;=\; R_{G_1,v} \cdot \frac{1}{1 + R_{G_2,u}}$$

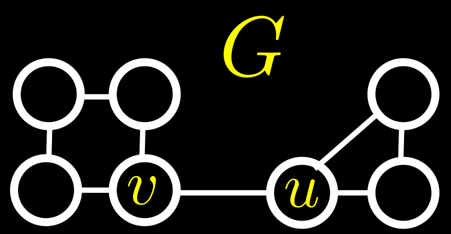


Calculating Pr(occupation)

- Notation: $R^{\sigma}_{G,v} = \dfrac{\Pr_G(v \text{ is occupied} \,|\, \sigma)}{\Pr_G(v \text{ is unoccupied} \,|\, \sigma)}.$

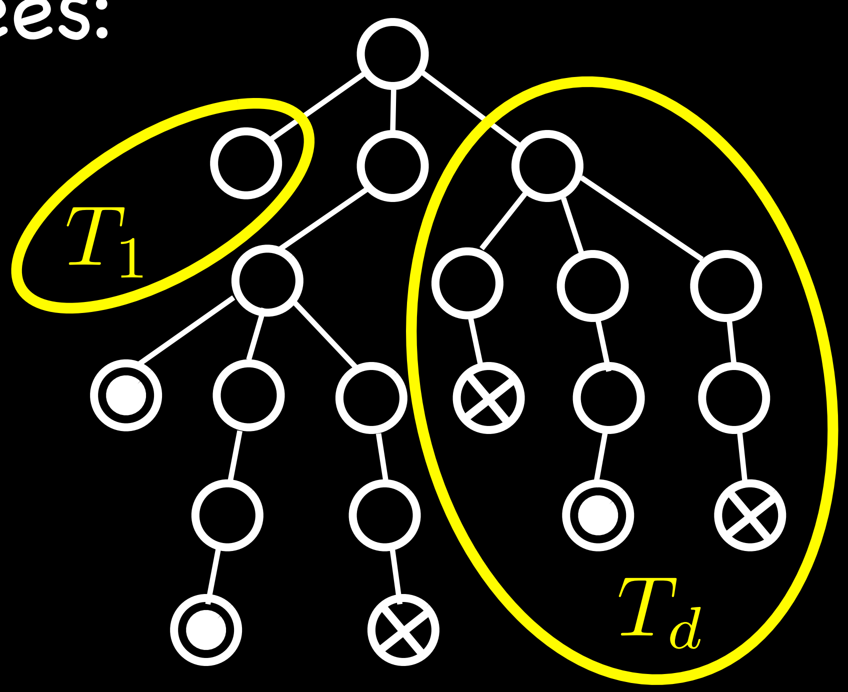
- Basic: when connection two separate graphs -

$$R_{G,v} = R_{G_1,v} \cdot \frac{1}{1 + R_{G_2,u}}$$



- Standard recursive procedure for trees:

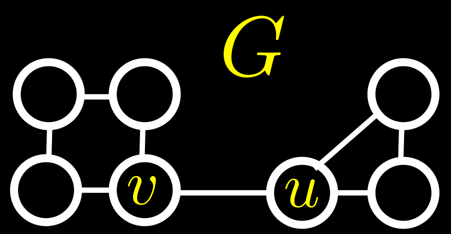
$$R_T = \lambda \prod_{i=1}^{d} \left(\frac{1}{1 + R_{T_i}} \right)$$



Calculating Pr(occupation)

- Notation: $R^{\sigma}_{G,v} \;=\; \dfrac{\Pr_G(v \text{ is occupied} \mid \sigma)}{\Pr_G(v \text{ is unoccupied} \mid \sigma)}.$
- Basic: when connection two separate graphs -

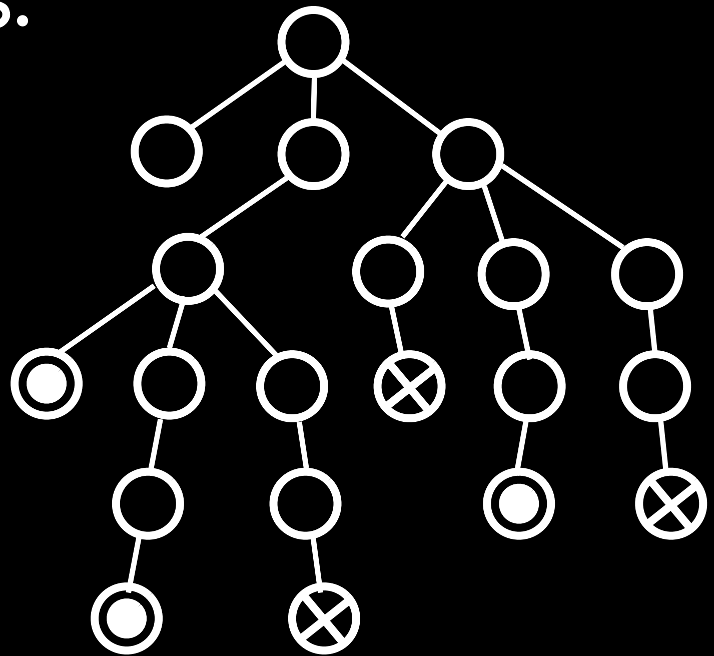
$$R_{G,v} \;=\; R_{G_1,v} \cdot \frac{1}{1 + R_{G_2,u}}$$



- Standard recursive procedure for trees:

$$R_T \;=\; \lambda \prod_{i=1}^{d} \left(\frac{1}{1 + R_{T_i}} \right)$$

Stopping rules -

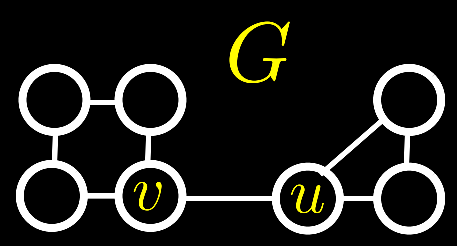


Calculating Pr(occupation)

- Notation: $R^{\sigma}_{G,v} \;=\; \dfrac{\Pr_G(v \text{ is occupied} \,|\, \sigma)}{\Pr_G(v \text{ is unoccupied} \,|\, \sigma)}\,.$

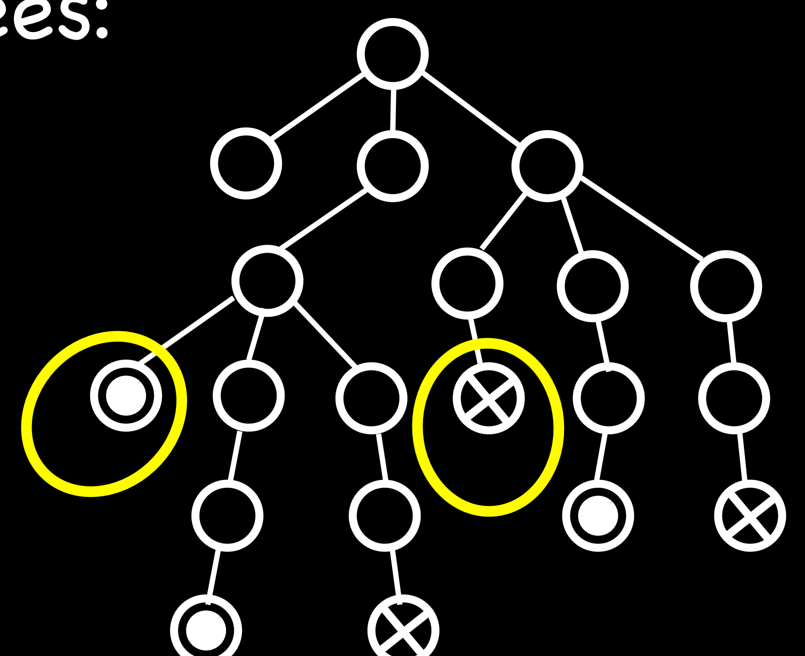
- Basic: when connection two separate graphs –

$$R_{G,v} \;=\; R_{G_1,v} \cdot \frac{1}{1+R_{G_2,u}}$$



- Standard recursive procedure for trees:

$$R_T \;=\; \lambda \prod_{i=1}^{d} \left(\frac{1}{1+R_{T_i}} \right)$$



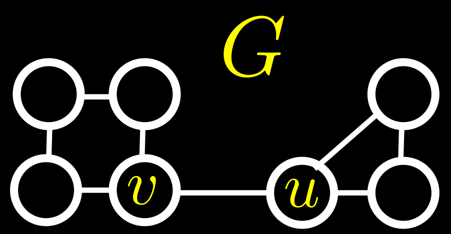
Stopping rules –

- fixed vertices: $R = \infty$ or 0;

Calculating Pr(occupation)

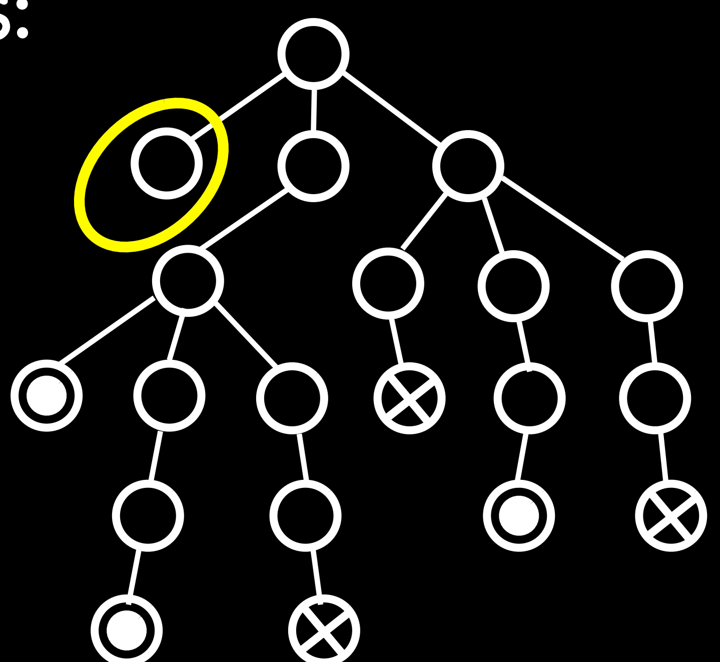
- Notation: $R^{\sigma}_{G,v} = \dfrac{\Pr_G(v \text{ is occupied} \mid \sigma)}{\Pr_G(v \text{ is unoccupied} \mid \sigma)}$.
- Basic: when connection two separate graphs -

$$R_{G,v} = R_{G_1,v} \cdot \frac{1}{1 + R_{G_2,u}}$$



- Standard recursive procedure for trees:

$$R_T = \lambda \prod_{i=1}^{d} \left(\frac{1}{1 + R_{T_i}} \right)$$

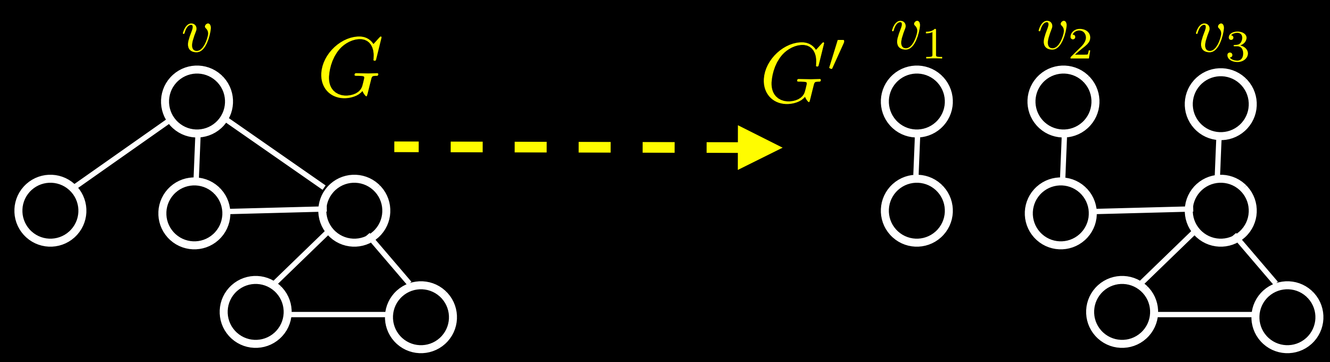


Stopping rules -

- fixed vertices: $R = \infty$ or 0;
- (unfixed) leaves: $R = \lambda$.

Calculating $R_{G,v}$

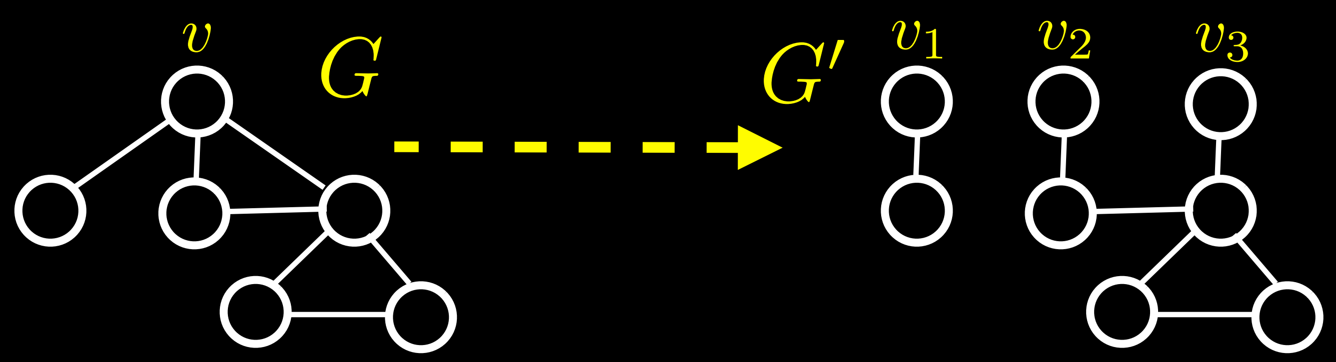
- Split v into $\deg(v)$ degree-one vertices:



associate the activity $\lambda^{1/d}$ with each v_i.

Calculating $R_{G,v}$

- Split v into $\deg(v)$ degree-one vertices:

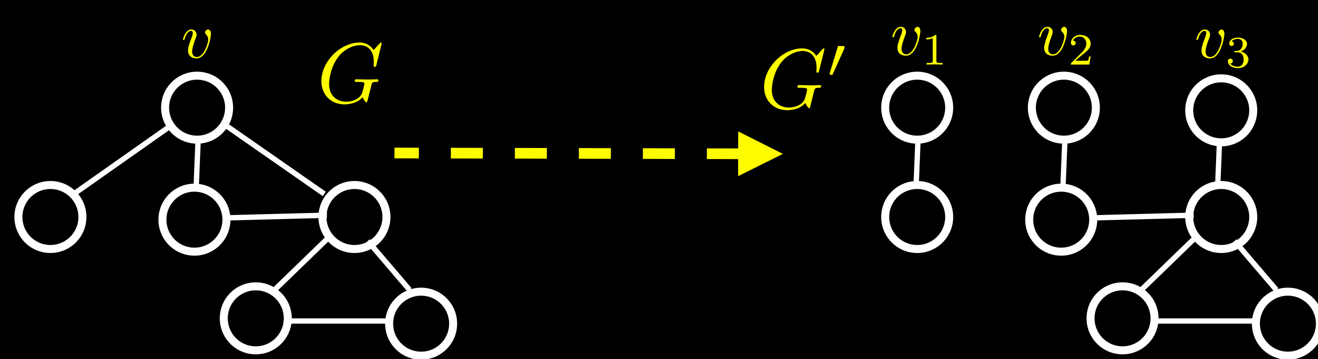


associate the activity $\lambda^{1/d}$ with each v_i.

- Observation:

$$R_{G,v} = \frac{\Pr_G(v \text{ is occupied})}{\Pr_G(v \text{ is unoccupied})} = \frac{\Pr_{G'}(\text{all } v_i \text{ are occupied})}{\Pr_{G'}(\text{all } v_i \text{ are unoccupied})}.$$

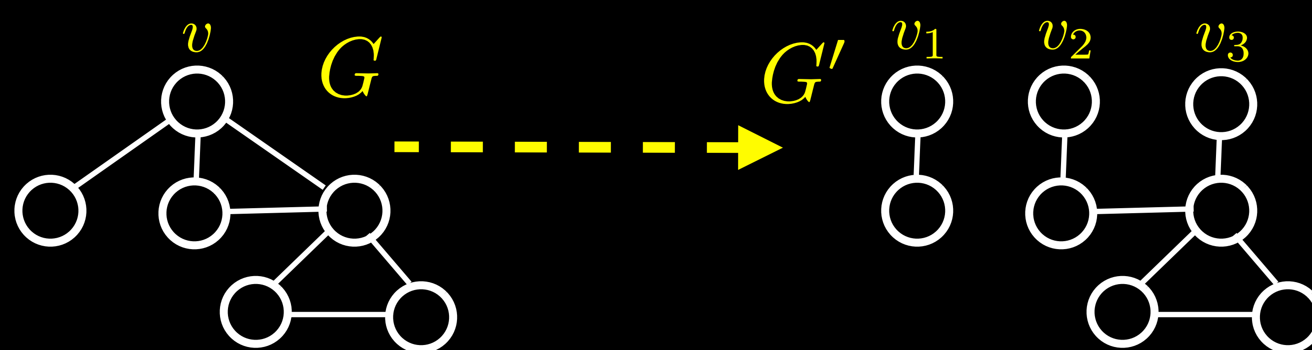
Telescopic Product



$$\frac{\Pr_{G'}(\text{all } v_i \text{ are occupied})}{\Pr_{G'}(\text{all } v_i \text{ are unoccupied})} = \prod_{i=1}^{d} \frac{\Pr(\otimes \cdots \otimes \odot \odot \cdots \odot)}{\Pr(\otimes \cdots \otimes \otimes \odot \cdots \odot)}$$

v_i

Conditional Probabilities



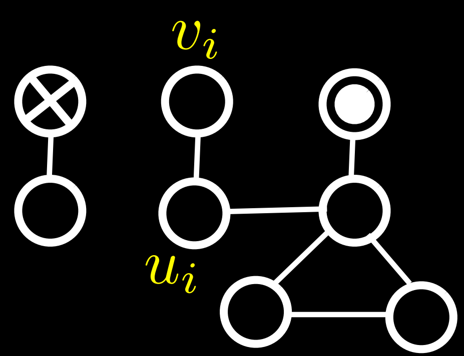
$$\frac{\Pr_{G'}(\text{all } v_i \text{ are occupied})}{\Pr_{G'}(\text{all } v_i \text{ are unoccupied})} = \prod_{i=1}^{d} \frac{\Pr(\otimes \cdots \otimes \odot \odot \cdots \odot)}{\Pr(\otimes \cdots \otimes \otimes \odot \cdots \odot)}$$

v_i

$$= \prod_{i=1}^{d} R_{G',v_i}^{\tau_i}.$$

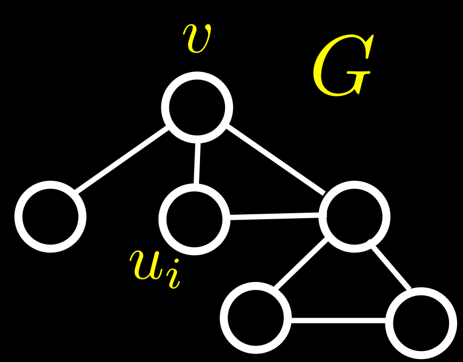
τ_i

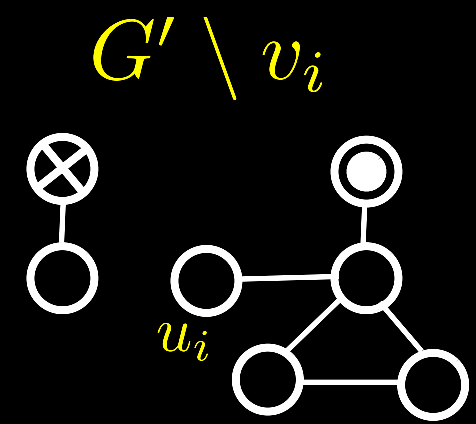
It's all about the Neighbors



$$R^{\tau_i}_{G',v_i} = \frac{\lambda^{1/d}}{1 + R^{\tau_i}_{(G'\setminus v_i),u_i}}$$

Recursive Procedure for Calculating $R_{G,v}$





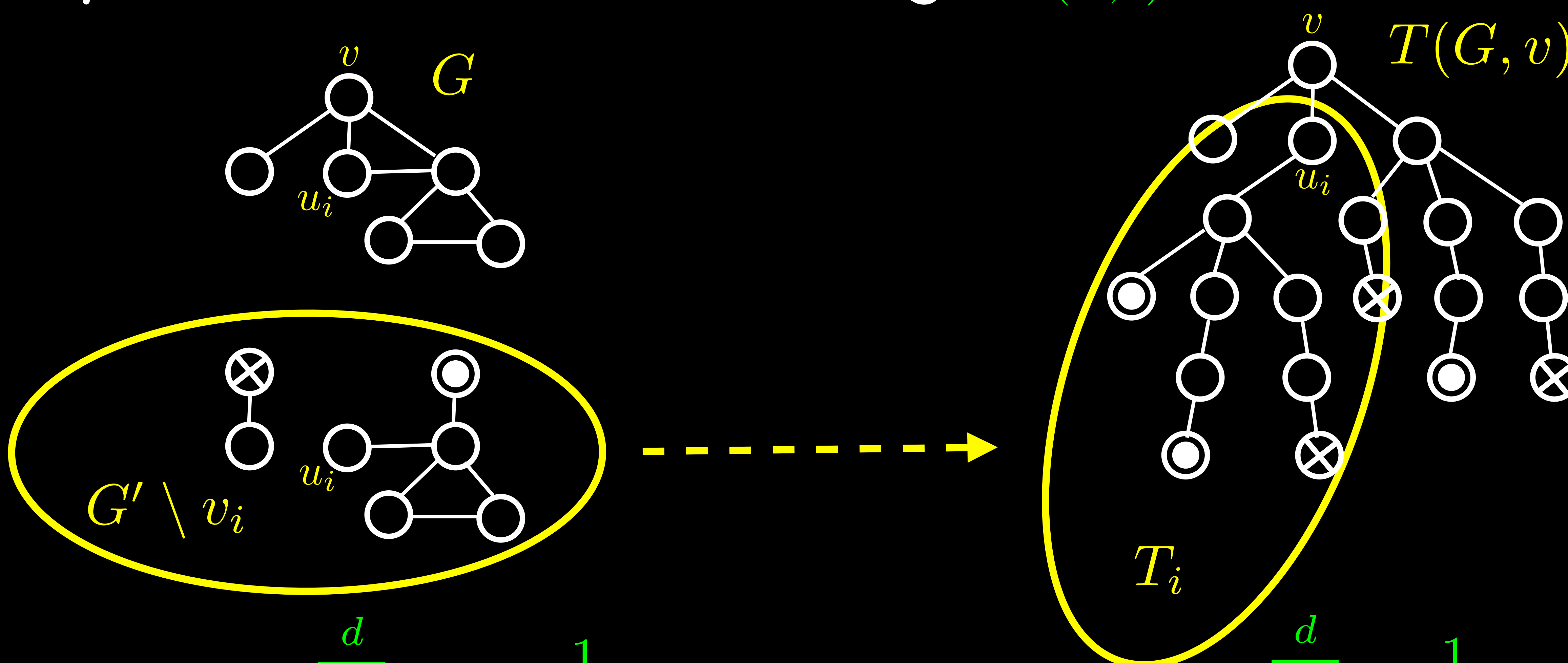
$$R^{\tau_i}_{G',v_i} = \frac{\lambda^{1/d}}{1 + R^{\tau_i}_{(G' \setminus v_i),u_i}}$$

$$\Downarrow$$

$$R_{G,v} = \lambda \prod_{i=1}^{d} \frac{1}{1 + R^{\tau_i}_{(G' \setminus v_i),u_i}}$$

$$R_{G,v} = R_{T(G,v)}$$

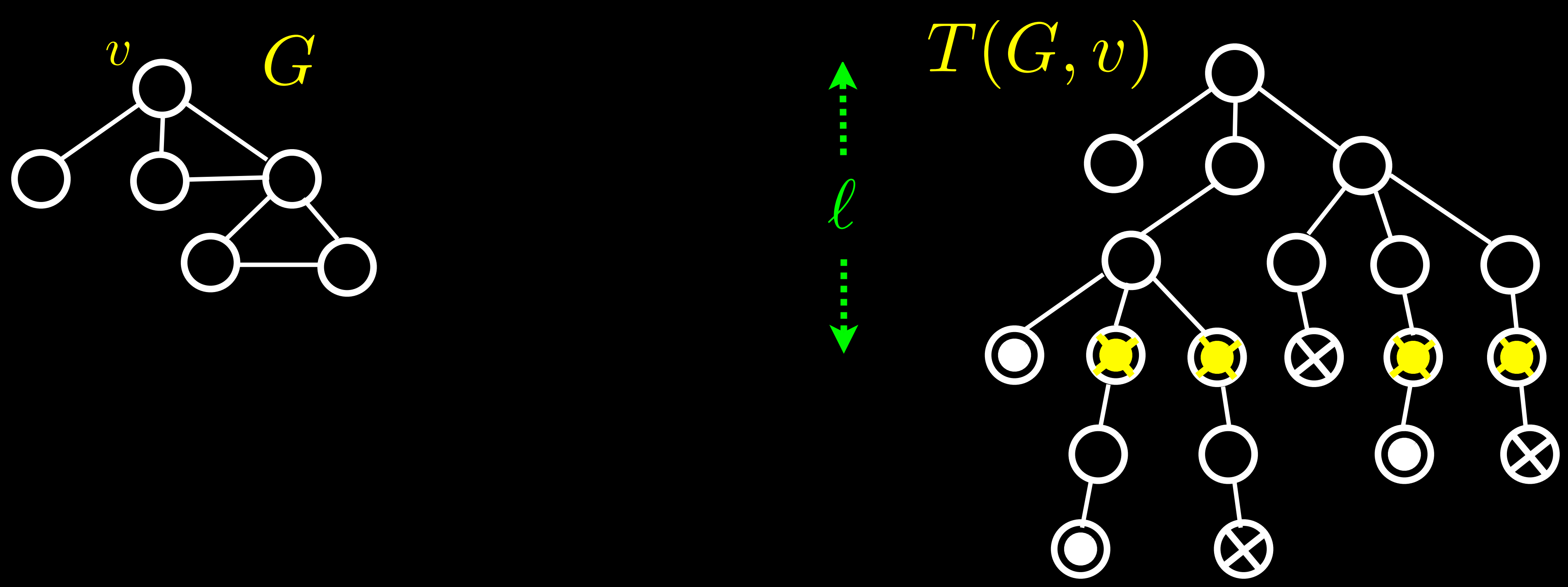
The procedure for calculating $R_{G,v}$ makes exactly the same calculations as the tree procedure for calculating $R_{T(G,v)}$.



$$R_{G,v} = \lambda \prod_{i=1}^{d} \frac{1}{1 + R^{\tau_i}_{(G' \setminus v_i),u_i}} \qquad R_T = \lambda \prod_{i=1}^{d} \frac{1}{1 + R_{T_i}}$$

Approximation Algorithm

- Run the previous recursive procedure, but if the stack of the recursion is ℓ levels deep return trivial lower and upper bounds.



Approximation Algorithm

- Run the previous recursive procedure, but if the stack of the recursion is ℓ levels deep return trivial lower and upper bounds.

- Running time is $O((\Delta - 1)^{\ell})$.

- For $\lambda < \lambda_c$ the difference between the resulting lower and upper bounds is $\leq \exp(-\ell)$.

  $\Rightarrow$ $(1+\epsilon)$-approximation for $\Pr(v \text{ is occupied})$ in time $\text{poly}(1/\epsilon)$.

Summary

- New Tree representation for general graphs.
- Proves that the tree is the "worst-case".
- New tree-like algorithm for approximately counting independent sets (works up to the tree threshold).
- Improved bounds for specific interesting settings:
  - Uniformly weighted independent sets with $\Delta \leq 5$.
  - The square lattice $\mathbb{Z}^2$.

Open Problems

1. Tree representation is valid for any binary spin system (i.e., Ising models). Is there a tree representation for models with more than two spins (e.g., proper colorings) ?
   - [Gamarnik-Katz, Nair-Tetali]: Tree-like algorithms (branching depends on spins as well, no direct comparison with model on the tree, require stronger and unnatural forms of decay of correlations).
   - Negative result [Sly]: tree is not worst case for uniqueness.

Open Problems

2. Improve the hardness threshold for approximately counting independent sets.

   – [Mossel-W-Wormald]: Conjecture that λ_c is the threshold for the computational probelm. Provide evedince that approximation is hard above λ_c.

3. More efficient variants of the algorithm (iterative?)

4. Solve other problems using the tree representation:
   – Spin glass Ising on $\mathbb{Z}^d$.
   – SSM down to T_c for Ising on $\mathbb{Z}^d$ for d>2.

Thanks

- Elchanan Mossel
- Alistair Sinclair & Fabio Martinelli