# 11 The occupancy method

## 11.1 Introducing the technique

Let's think more about how to approach independent sets in graphs – in physics, this is related to the hard-core model. Imagine a physical system with solid balls that cannot overlap (for example, in a system of atomic particles). This corresponds to having "independent sets," since we don't want adjacent vertices to be occupied.

Consider picking a random independent set $I \subset V(G)$, so that every set $I$ is chosen with probability proportional to $\lambda^{|I|}$. Here, $\lambda$ is known as the **fugacity**. Normalizing this probability, the denominator is something that will come up frequently:

**Definition 11.1**

Define the **independence polynomial** or **partition function**

$$P_G(\lambda) = \sum_{I \text{ independent set}} \lambda^{|I|}.$$

Note that the probability of a set $I$ being picked is

$$P(I) = \frac{\lambda^{|I|}}{P_G(\lambda)}.$$

In particular, if $\lambda = 1$, then $P_G(1) = i(G)$ (the number of independent sets) and we're choosing our independent set uniformly at random. We actually have a result bounding the value of our partition function across all graphs $G$:

**Theorem 11.2** (Kahn, Zhao, Galvin-Tetali)

If $G$ is an $n$-vertex $d$-regular graph, and $\lambda \ge 0$ is our parameter,

$$P_G(\lambda)^{1/n} \le P_{K_{d,d}}(\lambda)^{1/2d}.$$

Previously, we proved this theorem for $\lambda = 1$. A proof for general $\lambda$ was recently found (credited to Davies-Jenssen-Perkins-Roberts) that is very probabilistic! Here, we have a change of perspective: instead of trying to count everything directly, we find an "observable" by sampling at random.

**Definition 11.3**

Let the **occupancy fraction** be the expected fraction of $V(G)$ contained in the random independent set $I$ that we've chosen:

$$\overline{\alpha_G}(\lambda) = \frac{1}{|V(G)|}\mathbb{E}[|I|].$$

We can explicitly write this out and then rewrite it as a derivative:

$$= \frac{1}{nP_G(\lambda)}\sum_I |I|\lambda^{|I|} = \frac{\lambda}{n}\frac{d}{d\lambda}\log P_G(\lambda).$$

This is related to the cumulant, which is the logarithm of the moment generating function.

**Theorem 11.4**

Let $G$ be a $d$-regular graph, and let $\lambda \geq 0$ be our fugacity parameter. Then

$$\overline{\alpha_G}(\lambda) \leq \overline{\alpha_{K_{d,d}}}(\lambda).$$

So sampling an independent set, the expected number of vertices is maximized at $K_{d,d}$. In particular, note that

$$\frac{\log P_\lambda(G)}{|V(G)|} = \int_0^\lambda \frac{\overline{\alpha_G}(\tau)}{\tau} d\tau, \quad \frac{\log P_\lambda(K_{d,d})}{2d} = \int_0^\lambda \frac{\overline{\alpha_{K_{d,d}}}(\tau)}{\tau} d\tau,$$

and now we can bound over the integral to get the following, which is what we're trying to show in Theorem 11.2 above about the partition function:

**Corollary 11.5** (implies Theorem 11.2)

$$\frac{\log P_G(\lambda)}{|V(G)|} \leq \frac{\log P_{K_{d,d}(\lambda)}}{2d}.$$

Let's first prove a special case of our theorem:

*Proof of Theorem 11.4 for G triangle-free.* Pick a random independent set $I$ (under our hard-core model) on $G$ with fugacity $\lambda$. Call a vertex $v$ **occupied** if $v \in I$ and **uncovered** if $N(v) \cap I = \varnothing$; that is, $v$ has no occupied neighbors (though $v$ can still be occupied).

If $v$ is occupied, it must be uncovered (because a vertex and its neighbor can't both be in an independent set), and we also have the following facts:

- If $v$ is uncovered, $v$ can be occupied or not: the probability

$$\Pr[v \text{ occupied} | v \text{ uncovered}] = \frac{\lambda}{1+\lambda},$$

  because $v$ acts in isolation.
- Meanwhile, what about

$$\Pr[v \text{ uncovered} | v \text{ has exactly } j \text{ uncovered neighbors}]?$$

  The covered neighbors can't be occupied, so we don't need to worry about them. We know that $G$ is triangle-free, so $N(v)$ is an independent set, and if we condition on everything farther away from $v$ than $N(v)$, the $j$ uncovered neighbors are independent of each other. Thus, this is just

$$= \frac{1}{(1+\lambda)^j}.$$

So now let's find $\overline{\alpha_G}(\lambda)$ in two different ways. For one, this is just

$$= \frac{1}{n} \sum_{v \in V} \Pr(v \text{ occupied}) = \frac{1}{n} \cdot \frac{\lambda}{1+\lambda} \sum_{v \in V} \Pr(v \text{ uncovered}),$$

since all occupied vertices are uncovered. But by the second fact above, we also know that this is equal to

$$= \frac{1}{n} \cdot \frac{\lambda}{1+\lambda} \sum_{v \in V} \sum_{j=0}^{d} \Pr(v \text{ has } j \text{ uncovered neighbors})(1+\lambda)^{-j}$$

by Bayes' rule.

Now here's another way to compute this quantity:

$$\overline{\alpha_G}(\lambda) = \frac{1}{nd}\sum_{v}\sum_{u\in N(v)} \Pr(u \text{ occupied}),$$

using the fact that $G$ is $d$-regular, so each vertex is equally likely to come up. This can then be written as

$$= \frac{1}{nd}\frac{\lambda}{1+\lambda}\sum_{v}\sum_{u\in N(v)} \Pr(u \text{ uncovered}).$$

This can be thought of as running a two-part experiment: pick $I$ from the hard-core model, and then pick $v$ to be a uniform random vertex of $G$. Let $Y$ be the random variable equal to the number of uncovered neighbors of $v$ with respect to $I$.

On one hand, the occupancy fraction $\overline{\alpha_G}(\lambda)$ is

$$\overline{\alpha}_G(\lambda) = \frac{\lambda}{1+\lambda}\mathbb{E}[(1+\lambda)^{-Y}]$$

from the first calculation. But by the second calculation, we also have

$$\overline{\alpha_G}(\lambda) = \frac{1}{d}\frac{\lambda}{1+\lambda}\mathbb{E}[Y].$$

Setting these equal,

$$\mathbb{E}[(1+\lambda)^{-Y}] = \frac{1}{d}\mathbb{E}[Y].$$

where we should remember that $Y$ is a random variable supported on $\{0, 1, \cdots, d\}$, since our graph is $d$-regular.

But now instead of considering $Y$ coming from this specific process, let's imagine $Y$ is any probability distribution supported on $\{0, 1, \cdots, d\}$ that satisfies the condition we've just derived. Our goal is to show that

$$\frac{1}{d}\frac{\lambda}{1+\lambda}\mathbb{E}[Y] \le \overline{\alpha_{K_{d,d}}}(\lambda).$$

Note that this can be simplified as a linear program: let $x_k = \Pr(Y = k)$, and then we want to maximize

$$\mathbb{E}[Y] = \sum_{k=0}^{d} kx_k$$

under the linear constraints

$$x_k > 0, \sum_{k=0}^{k} x_k = 1, \sum_{k=0}^{d} x_k\left((1+\lambda)^{-k} - \frac{k}{d}\right) = 0.$$

It turns out that the maximium for this linear program occurs for the value that arises from $G = K_{d,d}$. This is because we have convexity of $Y \to (1+\lambda)^{-Y}$: maximizing $(1+\lambda)^{-Y}$ when conditioned on the total expectation happens when we concentrate everything at $\{0, d\}$. In addition, for every $\lambda$, there is a unique random variable $Y$ supported on $\{0, d\}$ satisfying the constraints we want. The one that results from $K_{d,d}$ satisfies all such constraints: thus, it must be the maximizer. □

But remember that we've been doing this for $G$ triangle-free. How are things different when we look at $G$ in general?

*Proof in general.* Again, we do a two-part experiment: pick $I$ to be a random independent set from the hard-core model, and choose $v$ to be a uniform random vertex of $G$. Let $H$ be the graph induced by uncovered neighbors of $v$ (beforehand, we only cared about the number of such neighbors for calculations).

So now we repeat the calculation: the first calculation becomes

$$\overline{\alpha_G}(\lambda) = \frac{\lambda}{1+\lambda}\mathbb{E}[P_H(\lambda)^{-1}]$$

and the second becomes

$$\frac{1}{d}\mathbb{E}[\text{number of occupied neighbors of } v] = \frac{\lambda}{d}\mathbb{E}\left[\frac{P_H'(\lambda)}{P_H(\lambda)}\right].$$

So now instead of enumerating over all possible distributions on $Y$, we can set up a similar linear program: our main constraint is that the two ways of finding $\overline{\alpha_G}(\lambda)$ are equal. Then we just need to show that our guess for the optimal solution is correct, and this can be done by showing that all of the dual constraints are satisfied. □

## 11.2 An alternative approach to the above problem

Here's another way to set up a linear program! Again, let's consider $\overline{\alpha_G}(\lambda)$ in two different ways (we'll drop $G$ and $\lambda$ throughout for sake of notation). Then

$$\overline{\alpha} = \Pr(v \text{ occupied}) = \frac{\lambda}{1+\lambda}\Pr(v \text{ uncovered}).$$

Letting $X$ be the number of occupied neighbors of $v$, we have a random variable that depends both on our independent set $I$ and our vertex $v$. Note that $v$ is uncovered is the same as saying that $X = 0$.

On the other hand, if we pick $v$ and then pick a uniform neighbor $u$, then $u$ is also a uniform random vertex (because $G$ is $d$-regular). Now

$$\overline{\alpha} = \frac{1}{d}\sum_{u \in N(v)} \Pr(u \text{ occupied}) = \frac{1}{d}\cdot\mathbb{E}[X]$$

by linearity of expectation. We know that $X$ takes on one of the values $\{0, 1, \cdots, d\}$: denote $p_k$ to be the probability $\Pr(X = k)$. Since we counted $\overline{\alpha}$ in two ways, we can set the values equal:

$$\frac{\lambda}{1+\lambda}\Pr(X=0) = \frac{1}{d}\mathbb{E}[X],$$

and plugging in values yields

$$\frac{\lambda}{1+\lambda}p_0 = \frac{1}{d}(p_1 + 2p_2 + \cdots + dp_d).$$

$X$ is some variable with various constraints, such as this one, and we can relax the problem by throwing in more constraints to form a linear program. But somehow this isn't capturing the whole system, since this doesn't give a strong enough answer.

So what else can we do? We can consider the probabilities $p_k$ and $p_{k-1}$ and try to come up with relations between them. If exactly $k$ neighbors of $v$ are occupied, we can produce another independent set $I'$ with $k-1$ neighbors by removing one of them from $I$. There are $k$ ways to remove a neighbor, and there are at most $d-k+1$ ways to go back to $I$: putting this together with the $\lambda$ factor from larger independent sets, we have

$$\frac{\lambda p_{k-1}}{k} \ge \frac{p_k}{d-k+1}$$

for all $2 \le k \le d$. Throw these into our linear program as well: we now want to maximize $\frac{\lambda}{1+\lambda}p_0$ given $p_0, \cdots, p_k \ge 0, p_0 + \cdots + p_k = 1,$, plus the constraints we've found above. This will give us some upper bound to $\overline{\alpha_G}(\lambda)$ for a $d$-regular graph $G$ – it may not be optimal, since we've only considered some of the constraints.

But it turns out this is indeed enough:

**Lemma 11.6**

If $(p_0, \cdots, p_d)$ is a maximizer of our linear program, then every inequality of the form $\frac{\lambda p_{k-1}}{k} \geq \frac{p_k}{d-k+1}$ is an equality.

*Proof.* Otherwise, if there is some $k$ with a strict inequality

$$\frac{\lambda p_{k-1}}{k} > \frac{p_k}{d-k+1},$$

and we can perturb our $p$s a bit: increase $p_0$ by $\varepsilon$, decrease $p_{k-1}$ by $\left(\frac{d\lambda}{1+\lambda} + k\right)\varepsilon$, and increase $p_k$ by $\left(\frac{d\lambda}{1+\lambda} + (k-1)\right)\varepsilon$ and we have a new maximizer (contradiction). □

So now we have a full-rank system of linear equalities, so there exists a unique solution. Indeed $G = K_{d,d}$ satisfies all of the equalities, and we're done.

The lesson here is that picking a uniform random $v$ and considering it locally gives linear constraints. By only looking at those constraints, we can get some bound on the occupancy fraction, and this is usually enough to solve the problem.

## 11.3 Further bounds with the occupancy method

Let's try to lower bound the occupancy fraction instead of upper bounding it. This was initially developed by Shearer: "triangle-free graphs have large independent sets."

**Theorem 11.7**

Fix a parameter $\lambda \geq 0$, and let $G$ be a triangle-free graph with maximum degree $d$. Then

$$\overline{\alpha_G}(\lambda) \geq (1 + o_{d\to\infty}(1))\frac{\log d}{d}.$$

In comparison, remember that every max-degree-$d$ graph has an independent set of size at least $\frac{n}{d+1}$ (take a vertex, remove it and its neighbors). That means that we're gaining a factor of $\log d$ here by having $G$ be triangle-free.

*Proof.* Pick $I$ according to the hard-core model again, and let $v$ be a uniform vertex of $V(G)$. Let $Y$ be the number of uncovered neighbors of $v$, so we now also care about the neighbors of neighbors of $v$.

Just like last time, there's two different ways to write the occupancy fraction. Because we have a triangle-free graph, the neighbors of $G$ behave independently of each other, so the number of uncovered neighbors satisfies

$$\overline{\alpha_G}(\lambda) = \frac{\lambda}{1+\lambda}\mathbb{E}[(1+\lambda)^{-Y}]$$

where the expected value term is the probability that none of the neighbors of $v$ are in $I$. By convexity, we can bound this:

$$\geq \boxed{\frac{\lambda}{1+\lambda}(1+\lambda)^{-\mathbb{E}[Y]}}.$$

Now since $G$ has maximum degree $d$, which is similar to being $d$-regular, we also know that

$$\overline{\alpha_G}(\lambda) = \mathbb{E}[\Pr(v \text{ occupied})] \geq \frac{1}{d}\mathbb{E}\left[\sum_{u\in N(v)} \Pr(u \text{ occupied})\right] = \boxed{\frac{1}{d}\frac{\lambda}{1+\lambda}\mathbb{E}[Y]}$$

by linearity of expectation. The occupancy ratio is then at least the maximum of the two estimates:

$$\overline{\alpha} \ge \frac{\lambda}{1+\lambda} \max \left\{ (1+\lambda)^{-\mathbb{E}[Y]}, \frac{1}{d}\mathbb{E}[Y] \right\}.$$

Note that one of these expressions decreases with $\mathbb{E}[Y]$ and the other increases. That means that there is some absolute constant we can find here:

$$\ge \frac{\lambda}{1+\lambda} \min_{y>0} \max \left\{ (1+\lambda)^{-y}, \frac{1}{d}y \right\},$$

and after some optimization, this indeed yields the result we want. □

What consequences does this have? First of all, there are other situations where similar techniques and theorems apply: in fact, we saw one earlier when we were talking about dense sphere-packings in high dimensions. The **hard-core model** models non-overlapping spheres, and we can set up problems in similar ways where we draw a sphere packing according to some distribution to find the expected fraction of space taken up. Doing the calculations, we found a sphere-packing density of $n2^{-n}$ in $\mathbb{R}^n$. This is close to the best we know for almost all $n$, and it's definitely better than the $2^{-n}$ that we got with a greedy packing. (Notice that we have the same characteristic log term as in our graph theory problem.)

Similarly, we can pack spherical caps on a sphere, which is like saying that we want points on a sphere that are pairwise separated by some angle. The most prominent case of this is called the **kissing problem**, which asks about the maximum number of unit balls that are nonoverlapping but all touch a central unit ball. This problem is interesting even in 3 dimensions!

## 11.4 A useful corollary: Ramsey numbers

We can translate the occupancy number statement directly into a graph theoretic statement:

**Corollary 11.8**

Every triangle-free graph on $n$ vertices with max degree at most $d$ contains an independent set of size at least $(1 + o_{d\to\infty}(1))\frac{\log d}{d}n$.

This actually gives us a bound for the Ramsey numbers:

**Corollary 11.9**

We have

$$R(3, k) \le (1 + o(1))\frac{k^2}{\log k}.$$

In other words, there exists an $n \sim \frac{k^2}{\log k}$ such that every graph on $n$ vertices has either a triangle or a large independent set.

*Proof.* If our graph is triangle-free, every neighborhood is an independent set. So if any vertex has degree $k$, we automatically have an independent set of the desired size. Otherwise, by the corollary, there exists an independent set of size at least $(1 + o(1))\frac{\log k}{k} \cdot n$, and choosing $n = (1 + o(1))\frac{k^2}{\log k}$ provides us with the independent set of desired size. □

This is essentially the best upper bound we have, and we also know a pretty close lower bound:

$$R(3,k) \geq \left(\frac{1}{4} + o(1)\right)\frac{k^2}{\log k}.$$

To construct a graph with this many vertices, remember that we found lower bounds to $R(k,k)$ by taking a random graph. Use a similar philosophy here: construct our graphs randomly, but use a **triangle-free process**. Start with an empty graph on $n$ vertices, and keep adding uniform random edges subject to the constraint "don't make triangles."

**Remark.** *In contrast, we don't even know the order of magnitude for $R(4,k)$ yet.*

## 11.5 Back to independent sets

We found an upper bound on $i(G)^{1/V(G)}$ earlier on: can we find a way to minimize this quantity? Which graphs have the minimum number of independent sets?

**Remark.** *Two different students guessed "$K_{d+1}$" and "random."*

It turns out that as stated, among all $d$-regular graphs $G$, the minimizer is $K_{d+1}$. However, if we restrict ourselves to bipartite $d$-regular graphs $G$, the answer becomes "random" or "a $d$-regular infinite tree."

One way to think of this is to consider a "2-lift" $G'$: take two copies of $G$, and replace edges with their "crossed" versions. Then we have $i(G') \leq i(G)^2$, and now by repeatedly lifting to destroy small cycles in the graph, we find that the number of independent sets, normalized, approaches some constant:

$$i(G_n)^{1/v(G_n)} \to c_d$$

if the girth of $G_n$ goes to infinity. In other words, we have a "tree-like" graph!

Let's modify the question so that neither of these answers is allowed:

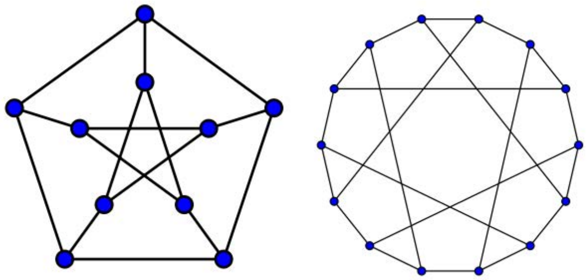
**Problem 11.10**

Let $G$ be a 3-regular graph. How do we maximize $i(G)^{1/v(G)}$ if we're not allowed to have 4-cycles? Similarly, how do we minimize this quantity if our graph is triangle-free?

**Theorem 11.11** (Perernau-Perkins, 2018)

Among 3-regular graphs $G$ without cycles of length 4, $i(G)^{1/v(G)}$ is minimized by the Peterson graph and maximized by the Heawood graph.

Here are the Peterson and Heawood graphs, respectively:



These are **Moore graphs** - they are essentially the smallest graphs that are $d$-regular with a specific girth condition. The idea here is that we can throw girth conditions into our linear program, because the additional constraints are local.

## 11.6 Proper colorings in graphs

Let's look at one more example: let $C_q(G)$ be the number of proper $q$-colorings of our graph $G$. Recall that we want to maximize $(C_q(G))^{1/V(G)}$ across $d$-regular graphs $G$: this can be done using the **Potts model** in statistical physics. Basically, sample a coloring $c : V \to [q]$< not necessarily proper, where a coloring occurs with probability proportional to $\beta^{m(c)}$, where $m(c)$ is the number of monochromatic edges. (Note that the parameter $\beta$ acts sort of like the parameter $\lambda$, and in the end, we can set $\beta = 0$. Notably, $\beta = 1$ gives a uniform coloring.)

So now we have a partition function for the Potts model

$$Z_{G,q}(\beta) = \sum_{c:V \to [q]} \beta^{m(c)};$$

note now that the log derivative

$$U_{G,q}(\beta) = \frac{\beta}{e(G)} \frac{d}{d\beta} \log Z = \frac{\mathbb{E}[m(c)]}{e(G)}$$

is the expected fraction of monochromatic edges. In physics, this is known as the "internal energy."

**Theorem 11.12**

For all 3-regular graphs $G$, $q \ge 2$, and $0 \le \beta \le 1$,

$$U_{g,q}(\beta) \ge U_{K_{3,3},q}(\beta).$$

Integrating this (the inequality is flipped because we integrate from 1 to $\beta \le 1$), we get

$$C_q(G)^{1/V(G)} \le c_q(K_{3,3})^{1/6}.$$

Proving this requires a similar kind of trick of constructing the two-part experiment and finding linear constraints. However, this has lots of variables - one for each possible configuration. We don't actually know how to do this by hand, but we plug this into a computer, and indeed $K_{3,3}$ is the maximizer! Unfortunately, we don't know how to get this to work for $d$-regular in general: the computation time is far too large.

18.218 Probabilistic Method in Combinatorics
Spring 2019