AN ANALOG OF POLYAS THEOREM FOR A NEW TYPE OF RANDOM WALK

DAVID WHITE

1. Background on random walks

Let G be a finite connected graph, but you can think of G as $\mathbb{Z}^2$ for what follows. A **random walk** on G is a sequence of vertices $X_0, X_1, X_2, \ldots$ where each X_{n+1} is chosen uniformly at random from the neighbors of X_n (i.e. each option with probability $1/d(X_n)$ or $1/4$ in $\mathbb{Z}^2$). It is **recurrent** if it always returns to the starting vertex. This forces it to returns infinitely many times: the walk after the n-th visit is again a random walk, so an $(n+1)$-st visit is guaranteed. Equivalently: for each vertex v, $Pr(X_n = v \text{ for infinitely many } n) = 1$, since there is a constant probability of walking from the origin to any vertex and given infinitely many tries this will occur with probability 1. The walk is **transient** if it is not recurrent. Equivalently: with probability 1 every vertex is visited only finitely often. If it doesn't return then we say the walk has escaped to infinity. All walks on a connected graph are either recurrent or transient.

Polya studied this question for graphs of the form $\mathbb{Z}^d$. In his formulation, $\mathbb{Z}^2$ was a city and the edges were city blocks. The particle undergoing the random walk was a drunkard, and the walk was called the "drunkard's walk." Polya proved the following amazing theorem:

Theorem 1. *A simple random walk on $\mathbb{Z}^d$ is recurrent if $d = 1$ or $d = 2$ and is transient for all $d > 2$.*

Shizuo Kakutani described this result as follows: "A drunk man will always find his way home, but a drunk bird may not." I intend to use this the exact day that I hand in my thesis. **Polya's proof was by counting** the number of paths which return and dividing by the total number of paths. It's messy and gives little understanding. Also, it has no hope of generalizing to graphs G which are not lattices. I want to sketch a much nicer proof which uses electrical networks and which will generalize to G.

Place a **unit resistance on each edge, unit voltage at the origin, and ground at infinity**. Then v_x is the probability of return to origin before reaching infinity. Use Ohm's Law (I = V/R) and observation ($p_e^{(r)} = I^{(r)}/2d = 1/(2dR^{(r)})$) to show a walk is **recurrent iff the resistance from any point to infinity is infinite**. This resistance keeps the walker contained.

Let's see how it works to prove Polya's Theorem: Do $\mathbb{Z}$, then $\mathbb{Z}^d$ picture, then discussion of $\mathbb{Z}^2$ and $\mathbb{Z}^3$

This characterization works nicely for all locally finite graphs, but to characterize using other properties is difficult and has not been finished. Some work brings in flows, etc. Check out Doyle/Snell or my Mathoverflow post.

Date: April 20, 2012.

2. Basic Walk

A problem computer scientists are interested in is **graph exploration by a mobile entity using only constant memory**. Useful if it's software moving on a network, searching for pages on the internet, or for a robot exploring an unfamiliar terrain. My favorite example is **the Roomba**, which rolls around your house and vacuums the carpets. Early models were known to get stuck in corners or to end up running in circles. If the robot has unlimited memory then you could just give it a map of the room and it would never get trapped or confused. In practice, the **robot has limited memory**, and for AI programmers, the less used the better. For us, the room is $[n] \times [n]$.

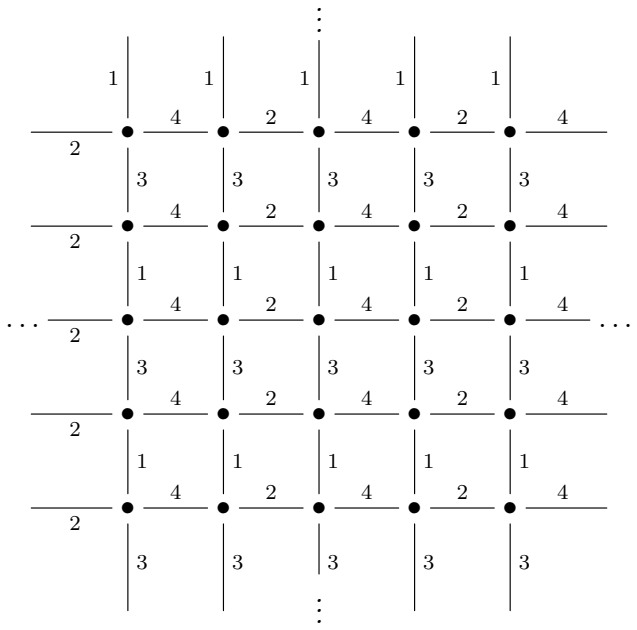
One option is to **give the robot no memory**. At a given vertex it picks a direction uniformly at random, i.e. with probability $1/4$ on the internal nodes, $1/3$ on the sides, and $1/2$ in the corners. The path of the robot is then a simple random walk. The problem is that we have no control over where it goes, how long it takes to cover all the ground, or how many times it re-vacuums the same spot. It turns out that this method will explore all n^2 vertices, but will take $O(|V| \log^2 |V|)$ time on average.

Leszek Gasieniec from Liverpool (Go-She-Nietz) considered another way to explore, where we **give the robot 2 bits of memory** and help it out by **labeling the edges** in the room. From each vertex, label the outgoing edges by $0, 1, 2, 3$ in such a way that each label is used once. Note that this means an edge can have two different labels; one from each direction. The robot only has to remember the label it just walked on and it can decide it's next label. When the robot enters a vertex by label $i \in \{0, 1, 2, 3\}$, it should exit following label $i + 1 \mod 4$. Practically, this means the robot can distinguish between the edges at any given vertex, i.e. has a local orientation. If you know that $G = \mathbb{Z}^2$ and you know where you start then it is not difficult to assign a labeling so that the robot explores a unique vertex at each step, i.e. vacuums the whole room in $O(|V|) = O(n^2)$ time. One way is the spiral out from the center. If you don't know what the room looks like (i.e. on a general G) then there is a way to assign a labeling to explore in $O(|V|)$ time but it requires the nodes to have a large memory and do a lot of work updating their labelings, so it might be infeasible from the point of view of applications.

Let's try to reduce the amount of memory we're requiring and make the labeling scheme easier. We'll just assign the labels randomly, all at once, and see what happens. This process is called the **Basic Walk**. Note that all the randomness is in the initial labeling. Once that's fixed and the starting vertex is chosen it's deterministic. Examples:

Note that with an arbitrary labeling a **robot can become trapped**, e.g. in a 4-cycle, and then you don't get to see all the nodes (i.e. you leave some dirty spots). We want to investigate the likelihood of this occurring. Here's an open question: what's the expected length of the largest cycle? Does it cover a constant fraction of the nodes? Gasieniec conjectured it would. This is an asymptotic question, so we need to let $n \to \infty$. Another pro of letting $n \to \infty$ is that we get a closer approximation to a true room, since you can think of this as making a finer and finer mesh. The analogous question to the open problem on $[n] \times [n]$ is exactly the recurrence question, except we must replace "recurrence" with "gets trapped." DRAW THE SQUIGGLY LINE AND CYCLE

Example 2. *A labeling in $\mathbb{Z}^2$ where any basic walk escapes to infinity:*



The reader can easily verify that every starting vertex and port number leads to an infinite staircase which moves in the directions specified by the first two steps and which gets further away from the starting location with every step. It is clear how this generalizes to $\mathbb{Z}^d$ with a staircase consisting of a sequence of moves, one in each of the d directions. This example generalizes to create an infinite family of examples where the basic walk escapes to infinity from any starting vertex and any initial label. We simply add in blocks of 4 columns which act like plateaus for the robot to move east or west for $4n$ steps between a given north-south step on the staircase.

3. Quasirandom and Self-Avoiding

Obviously, a big property of the Basic Walk is that it's all defined locally. There is another model you may have heard of which is similarly defined locally. It's the **Rotor Router model of Jim Propp**, and it's an example of a quasirandom analogue of a random walk on a directed graph, i.e. it's **deterministic** and designed to give the **same limiting behavior but with faster convergence**. Place a rotor at each v, like what balls come out of at the batting cages. Next, fix a rotor pattern $e_1, e_2, \ldots, e_{d(v)}$ running through all edges out of v and point the rotor via e_1. When v is first visited, the particle exits by e_1 and then the rotor rotates to e_2. When v is next visited, the particle exits by e_2 and we repeat this process. These rotor routers have been studied quite a bit in the past 10 years, and have found numerous applications, e.g. load balancing. It's supposed to be **"better than random"** because the central limit theorem behavior is achieved immediately,

e.g. on $[n]$ it will have exactly half the particles (every other particle) leave by the left as by the right, which the CLT predicts but only with an error depending on the number of trials.

Differences from Basic Walk: It doesn't matter where you enter v from, you will leave by the direction the rotor points. The scheduling policy is purely local and deterministic–there is no randomness at all. You can't trap in any way. Also, there is a theorem which says it behaves like the simple random walk enough that it'll have the same recurrence/transience behavior (in terms of number of times the origin is visited if you start a large number of particles at once) so it's recurrent in $\mathbb{Z}^2$ and transient in $\mathbb{Z}^d$ for $d > 2$.

Professor Johanna Franklin (University of Connecticut) gave a talk here in March on "Randomness and applied recursion theory" discussing how to formally state which properties a random sequence should not have and how randomness corresponds to computational strength. These quasirandom processes have some of those properties, but lack the others (e.g. statistical randomness). You'll probably hear more about them in the future.

Another big property of the Basic Walk is its ability to get trapped. The robot problem is not a random walk. It acts like one at vertices which have never been visited before, but appears to act like a **self-avoiding random walk** at vertices which have been visited before. This is because if the walk previously left a vertex by label 2 and if it comes in by a label other than 1, then it must avoid the edge previously traveled. As soon as the robot uses the same edge in the same direction it's in a cycle. Self avoiding random walks can also get trapped. What's known is that for $d \leq 2$ the walk is expected to get trapped and for $d \geq 5$ there's enough space and it's expected to escape. Based on this and the example above, the problem group conjectured that the robot would get trapped (this is the analog of recurrence) in $\mathbb{Z}^2$ but would escape in sufficiently high dimension.

4. New Results for exploration with memory

Polya didn't have the option for his walk to get trapped in a small space. This option exists for the robot and it's exactly what we want to study. Indeed, for the $d = 1$ case (line) we have a simple argument to see that the robot always gets trapped. Trap on line and grid...(Images: on line it's a labeling where the label from m to $m + 1$ is 1 and the label from $m + 1$ to m is 2. On grid it's 4-cycle and a trap configuration where center vertex is entered from below by label i and the path goes left on $i+1$, back to center on $i+2$, right by $i+3$, and back to center by i, so the trap bounces the robot back and forth forever among these three vertices.)

To avoid this trap on the line, the labels must alternate $1, 2, 1, 2, 1, 2, \ldots$. This occurs with probability $0 = \lim_{n\to\infty}(1/2)^n$. For $d = 2$ there are many ways for the robot to get trapped. Above are two. Because these traps are small and local (they don't depend on how far you've come on the random walk), they occur with constant, nonzero probability. We'll use these traps to prove the following theorem:

Theorem 3. *In $\mathbb{Z}^2$, the robot will get trapped with probability 1, i.e. the process is not transient.*

Shells Method. The trap with 3 vertices occurs with constant probability $c > 0$. Draw concentric squares S_n (shells) of side length $2n$ centered at the origin. Let E_n be the event that the walk reaches S_n and the first time it does so is not a trapping configuration. Note that this "first vertex hit" cannot be a corner because corners are not adjacent to interior vertices. Because the trap exists entirely on the shell, it is independent of the walk up to that point.

Let E be the event that the walk escapes to infinity. It's not hard to see $E = \bigcap E_n$ because to escape to infinity you must never trap and you must pass each S_n. These E_n are not independent,

but because $E_1 \subset E_2 \subset E_3 \ldots$, we can still write $P(E) = \prod P(E_n|E_{n-1})$. The probability of a path from S_{n-1} to S_n existing is ≤ 1. The probability that the first vertex on S_n hit is not a trap is $1 - c$. Thus, $P(E_n|E_{n-1}) \leq 1 * (1 - c)$ and so $P(E) \leq \prod 1 - c = 0$ because $c > 0$ implies $1 - c < 1$. □

This is not so surprising, since it's the same for Polya and for self-avoiding random walks. Much more surprising is that the method of proof generalizes to show:

Theorem 4. *For all d, in $\mathbb{Z}^d$, the robot will get trapped with probability 1, i.e. the process is not transient.*

Proof. For $d = 3$ each vertex is degree 6 and a trap can be found using only 3 neighbors (i.e. a trap can be found on a cubical shell around the origin). The trap occurs with constant probability $c' > 0$ and so the probability of escape is $\leq (1 - c') * (1 - c') * \cdots = 0$. The same idea works for $d > 3$ as you can always find a trap on the surface of the **hypercube** and this means independence of the previous steps is no problem. This is because a vertex on S_n will have $2d - 2$ neighbors on the shell and only needs d to make a trap.

Also worth noting: this proves that the probability of escape from the origin is zero. By symmetric, the probability of escape from any fixed vertex is zero. $Pr(\exists$ vertex which the walk can escape to infinity from$) = 0$ because it's a countable union of events, each of probability zero. □

5. Other locally finite graphs

To generalize this proof to other graphs you'd need them to have shells, and this comes down to expander properties. A graph which fails to have shells is the hexagonal lattice (think chicken-wire or honeycomb). We found a proof there for trapping and it generalizes to show that any regular graph has trapping basic walk. The trick is to use spires of length d where d is the degree. What about graphs which are not regular? Well, a third proof works in that case:

Theorem 5. *On any locally finite graph G with all vertex degrees bounded by a constant D, the basic walk cycles with probability 1.*

Star Method. The pigeonhole principle guarantees us that there are infinitely many vertices v with a neighbor w of degree $d(w) \geq d(v)$. This is because every time a vertex v only has neighbors of smaller degree, all those neighbors have a neighbor (v) with larger degree. If the basic walk is to have any chance of escaping to infinity, then it must be the case that infinitely often the robot moves from a vertex v to a vertex w such that $d(v) \geq d(w)$. Here we are using the hypothesis of bounded degree, which means the robot cannot move from a vertex of degree 1 to one of degree 2, then one of degree 3, etc. Such a chain would eventually hit a vertex of degree D and then need to move to one of degree $\leq D$. We label these steps of the robot (from larger degree to smaller degree) by $w_1 \to v_1, w_2 \to v_2, \ldots$, where we do not assume $w_i \neq w_j$ for $i \neq j$ but we do assume $v_i \neq v_j$ by simply removing the pairs (w_i, v_i) where v_i has appeared in the list before. Clearly this will not change the fact that there are infinitely many such pairs.

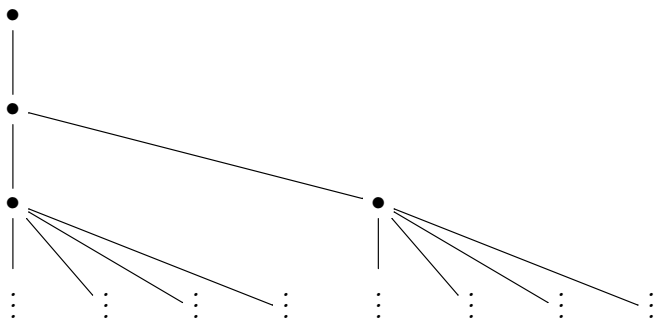
Whenever the basic walk takes a step $w_i \to v_i$ from the list above, we consider the event E_i that the configuration C_{v_i} is achieved. Let N denote the set of neighbors of v_i which are not w_i. Because $d(v_i) < D$, $|N| < D$. Furthermore, $d(x) < D$ for all $x \in N$, so

$$P(E_i) = \frac{1}{d(v)} * \prod_{x \in N(v)} \frac{1}{d(x)} > c = \frac{1}{D} * \left(\frac{1}{D}\right)^D > 0$$

In order for the basic walk to escape to infinity, this event must be avoided for infinitely many v_i. So the probability p that the basic walk escapes to infinity satisfies $p \leq (1-c) * (1-c) * \cdots = 0$, proving that the basic walk cycles with probability 1. □

The final question I will consider is whether or not this hypothesis of bounded degree is necessary. It's clear that finite degree is necessary to even define the process.

Example 6. *T is the tree where every vertex in level n has 2^n children. As usual, the root is in level 0.*



Proof. Start a basic walk from the root in the tree above, using the fact that the initial step is determined because the root has degree 1. Define an event P to be "for each i, the robot is distance i away from the root at step i" i.e. "all steps are away from the root." Certainly, $\Pr(\text{escape}) \geq \Pr(P)$ since P is a way for the robot to escape. Define events P_i to be "at step i the robot is distance i from the root." So $P_0 \subset P_1 \subset P_2 \subset \ldots$, $P = \bigcap P_i$, and $\Pr(P) = \prod \Pr(P_n|P_{n-1})$. Because each vertex has only one arc pointing back at the root, $\Pr(P_0) = Pr(P_1) = 1$, $\Pr(P_2|P_1) = 1 - 1/3 \geq 1/2$, $\Pr(P_3|P_2) = 1 - 1/5 \geq 1/4$, $\ldots$, $\Pr(P_n|P_{n-1}) = 1 - 1/(2^{n-1}+1) \geq 1/2^n$. Thus:

$$\Pr(P) \geq \prod_{n=2}^{\infty} 1 - \frac{1}{2^n} \text{ and taking logs we get } \ln(\Pr(P)) \geq \sum_{2}^{\infty} \ln\left(1 - \frac{1}{2^n}\right)$$

To prove $\Pr(P) > 0$ we must prove this sum is greater than $-\infty$. We'll use the Taylor Series expansion of $\ln(1-x)$ around 0:

$$\ln(1-x) = -x - \frac{x^2}{2} - \frac{x^3}{3} - \ldots \text{ for } -1 \leq x < 1$$

For x of the form $1/2^n$ and $n \geq 1$ this equality holds. Thus:

$$\ln(\Pr(P)) = -\sum_{2}^{\infty} \frac{1}{2^n} - \frac{1}{2}\sum_{2}^{\infty} \frac{1}{2^{2n}} - \frac{1}{3}\sum_{2}^{\infty} \frac{1}{2^{3n}} - \cdots - \frac{1}{k}\sum_{2}^{\infty} \frac{1}{2^{kn}} - \ldots$$

$$\geq -\sum_{2}^{\infty} \frac{1}{2^n} - \sum_{2}^{\infty} \frac{1}{2^{2n}} - \sum_{2}^{\infty} \frac{1}{2^{3n}} - \cdots = -\frac{1/2^2}{1-1/2} - \frac{1/2^4}{1-1/2^2} - \frac{1/2^6}{1-1/2^3} - \cdots \geq -\sum_{n=1}^{\infty} \frac{1}{2^n} = -1$$

To show $2^{-2k}/(1-2^{-k}) \leq 2^{-k}$ as needed in the last inequality, note that $1 - 2^{-k} \geq 1/2 \geq 2^{-k}$ so that $2^{-2k}/(1-2^{-k}) \leq 2^{-2k}/(2^{-k}) = 2^{-k}$.

Undoing the log, we see that P(escape)$\geq \Pr(P) \geq e^{\ln(\Pr(P))} \geq e^{-1} > 0$. This proves there is a positive chance that the robot escapes, i.e. the process is transitive. □

Theorem 7. *As $n \to \infty$, a basic walk on K_n is expected to visit at least $(1 - 1/e) * n$ nodes*

Conjecture 8. *The expected number of arcs traversed by a basic walk on K_n is $1.8 * n$ as $n \to \infty$.*

6. Future Directions

- In $\mathbb{Z}^2$, what is the expected length of time it takes the robot to hit a cycle? (i.e. the expected number of steps taken). What is the expected number of vertices visited? The number of edges used per vertex visited?
- Return to the case of the finite grid and find the expected number of vertices visited on a walk. If it's less than n^2 (seems extremely likely) then this process is not good for vacuuming the room.
- As $n \to \infty$ can we prove that for all $k < n$ there must be a path of length k with probability 1? Note that there is a labeling where every vertex is in a 4-cycle.
- What's the expected size of a cycle, i.e. a trap more complicated than those considered above? What if there are big holes in the graph? Maybe this forces very long cycles to go around those holes.