HOW A ROBOT CAN EXPLORE AN INFINITE GRAPH

DAVID WHITE

1. Motivation

The Roomba 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. We want to model this problem mathematically, so we assume it's a square room and we consider a grid to **make it a discrete rather than continuous** problem. So the graph is $P_n \Box P_n$. We'll consider more general graphs at the end.

One option is to **give the robot no memory**. At a given vertex it picks a direction uniformly at random, i.e. each with probability $1/4$. The path of the robot is then a **simple random walk**, as investigated famously by Polya on infinite lattices $\mathbb{Z}^d$ in 1921. It turns out that this method will explore all n^2 vertices, but will take $O(n^4)$ 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 $1, 2, 3, 4$ 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 $i \in \{1, 2, 3, 4\}$, 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. 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 $O(n^2)$ time. One way is the spiral out from the center (if you get to pick starting vertex and label) or a zig-zag (if you don't, but it's on a torus).

If the room is more complicated, though, it might be difficult to assign a good labeling. Also, makers of the Roomba can't be expected to know the layout of every room in a customer's house. Thus, it is natural to consider a random labeling on the edges and ask how long it takes to explore the room. Note that with an arbitrary labeling a robot can become trapped, e.g. in a 4-cycle. It may be difficult to find values for a specific n, especially since it's not 100% clear what the inductive step would be. Once might also need to use differential equations and run into the percolation problem. So we'll let $n \to \infty$ (i.e. run the random process on $P_\infty \Box P_\infty$) and get an "asymptotic answer" instead. Note that this gives a closer and closer approximation to the continuous room via a finer mesh.

2. Background on random walks

On an infinite grid, this question of whether or not a simple random walk reaches each vertex is known as recurrence vs. transience. Formally, a **random walk** on a graph 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

Date: December 14, 2011.

X_n. It is **recurrent** if it always returns to the starting vertex, i.e. returns infinitely many times. 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 for each vertex v, $Pr(X_n = v \text{ for infinitely many } n) = 0$. 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." Polya's proof came down to writing the probability $p(n)$ of return after n steps, approximating it using Stirling's formula, and proving that $\sum_n P(n)$ either converged (transient) or diverged (recurrent). It's messy. A much nicer proof by electrical networks places a **unit resistance on each edge** and proves that a walk is **recurrent iff the resistance from any point to infinity is infinite**. This resistance keeps the walker contained. With this formulation, the proof is very simple and comes down to just Ohm's Law and Kirchkoff's Law from physics. This characterization works nicely for all locally finite graphs, but to characterize using other properties is difficult and has not been finished.

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. It should be noted that those avoid vertices while ours avoids edges, but this can be fixed by considering our edges as vertices and connecting two if they share an endpoint. You can label the new edges by the label on the second of the two old edges which make it up. Understanding self-avoiding random walks is one of the biggest open problems in probability theory (right up there with finding the critical point for percolation). What's known is that for $d \leq 2$ the walk is recurrent and for $d \geq 5$ it is transient. Based on this, 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.

3. 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 2. *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 3. *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. □

This means that the set of labelings where the robot escapes has measure zero. I have a drawing, but it's too complicated to explain in words. I'll draw it eventually and include it. The robot can start anywhere, with any starting label and will escape to infinity via a staircase. It's easy to generalize this by adding in plateaus where it goes straight for $4n$ steps between steps on the staircase, so this gives an infinite family of labelings where the robot always escapes.

4. Other locally finite graphs

At one point in the process it was suggested that the hexagonal lattice (think chicken-wire or honeycomb) might be easier to work with. It turns out this is not the case because it's harder to form traps there. Indeed, you can't form traps with just a vertex and its neighbors unless those traps include the vertex the walk came in from, i.e. the shell surfaces are too small. This led to a different kind of trap, called a spire, where you enter a vertex of degree D by label i and travel out along a path by labels $i, i+1, i+2, \ldots$ for D steps, then turn around and come back along the exact same path. This path is a trap, since it re-enters the initial vertex by label i again. This idea works easily on the hexagonal lattice because the path has length 3 and occurs with a constant nonzero probability.

Theorem 4. *On the hexagonal lattice, the robot will be trapped with probability 1, i.e. the process is not transient.*

Spires Method. Let $S_n = \{v \mid d(v_0, v) = 4n\}$ and let $D_n = \{v \mid d(v_0, v) \leq 4n\}$. Let E_n be the event that you hit S_n and that the first vertex where that occurs (call it v_n) does not have a spire in the region between D_n and S_{n+1}. As long as a spire consists entirely of vertices which have not been visited before, it occurs with constant probability $c > 0$. Without this restriction on the vertices, the probability changes because some labels will already be set. Thus, we'll restrict spires we consider to those strictly between S_n and S_{n+1}. The probability of getting from S_n to S_{n+1} without re-entering D_n is ≤ 1. Thus, $P(E) = \prod P(E_n|E_{n-1}) \leq \prod (1 - c) = 0$. □

This method generalizes nicely to arbitrary locally finite graphs, as long as we have a fixed constant M of radius for our spheres around the starting vertex. The same proof proves:

Theorem 5. *On any locally finite graph G such that there exists an integer M with $d(v) < M$ for all v, the robot will be trapped with probability 1, i.e. the process is not transient.*

To even understand the theorem we must define a process on a non-regular graph. There are many ways to do this if you first label each edge going out from v by $1, 2, \ldots, d(v)$. The one we choose is: if the robot enters v along label i then it leaves v along the label $i + 1 \mod d(v)$. With this method one can define the spires as above and going $d(v)$ steps away and then $d(v)$ steps back on the same path will create a trap. A different way is: if the robot enters v along label i and $d(v) > i$ then it leaves along label $i + 1$. Otherwise it leaves along label 1. There are problems with this system. For one, a random walk will use label 1 much more than any other label because it's more likely to pick 1 than another label if moving from a higher degree vertex to a lower degree one. Secondly, if we're trying to construct a trap at v and we entered v by a label $i > 3$ then there is no guarantee that a spire will exist. It could be the case that all the neighbors of v have degree 2 and so paths exist but there is no way to enter v by label i after completing a spire. So we will use the first method and not this method.

Proof. Let S_n be the sphere of radius $n * M$. For a robot to escape it must reach each S_n in a vertex v_n which does not have a spire of length $d(v_n) < M$. As above, we require this spire to take place entirely between D_n and S_{n+1}. If v_n has no path to S_{n+1} which doesn't require going back into D_n, then allow the walk to continue. It will either get trapped between S_n and S_{n+1} or it will re-enter D_n. If it never leaves D_n again we're done, as the walk is trapped. If it reaches a different vertex on S_n then define that vertex to be v_n and repeat the argument. If the walk is to ever escape, there must be some $v_n \in S_n$ with a path to S_{n+1} that does not pass through D_n. Let c be the probability of a spire of length M, so the probability of a spire of length $d(v_n)$ is $> c$. Thus, $P(E_n|E_{n-1}) \leq 1 - P(\text{spire at } v_n) \leq 1 - c$. From here the proof proceeds as above, i.e. $P(E) \leq \prod (1 - c) = 0$. □

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. Here is a graph with unbounded degree such that the probability of a path to infinity is strictly greater than zero. This proves our hypothesis was truly necessary and it classifies locally finite graphs where the robot gets trapped: (Image is a tree where the root has degree 1, the next node has degree 4, the three nodes below that have degree 8, the $3*7$ nodes below that have degree 16, etc. All nodes at depth n have degree 2^n with only 1 edge pointing back at the root and $2^n - 1$ pointing down)

Start a walk from the root in the tree above. 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. In previous examples, any infinite path was guaranteed to contain traps, but this one will not because the degree is unbounded. Define events P_i to be "at step i the robot is distance i from the root." So $P_0 \subset P_1 \subset P_2 \subset \ldots$

and because each vertex has only one edge pointing back at the root, $Pr(P_0) = Pr(P_1) = 1$, $Pr(P_2|P_1) = 1 - 1/4$, $Pr(P_3|P_2) = 1 - 1/8, \ldots, Pr(P_n|P_{n-1}) = 1 - 1/2^n$. Thus:

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

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:

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

$$\geq -\sum_{2}^{\infty} \frac{1}{2^n} - \sum_{2}^{\infty} \frac{1}{2^{2n}} - \sum_{2}^{\infty} \frac{1}{2^{3n}} - \cdots = -\frac{1}{2} - \frac{1}{12} - \frac{1}{56} - \cdots \geq -\sum_{n=1}^{\infty} \frac{1}{2^n} = -1$$

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

5. Future Directions

- 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.

6. References

(1) G. Polya. Über eine Aufgabe betreffend die Irrfahrt im Strassennetz. Math. Ann, 84:149-160, 1921.

(2) Dobrev, S., Jansson, J., Sadakane, K., Sung, W.K.: Finding short right-hand-on-the-wall walks in graphs. In: Pelc, A., Raynal, M. (eds.) SIROCCO 2005. LNCS, vol. 3499, pp. 127139. Springer, Heidelberg (2005)

(3) Ga sieniec, L., Klasing, R., Martin, R., Navarra, A., Zhang, X.: Fast periodic graph exploration with constant memory. J. Computer and System Science 74(5), 808822 (2008)

(4) Cohen, R., Fraigniaud, P., Ilcinkas, D., Korman, A., Peleg, D.: Label-guided graph exploration by a finite automaton. ACM Transactions on Algorithms 4(4), 331344 (2008)

(5) more to come