Networks Algorithm and Analysis

Lecture 18

April 10, 2012

In the last lecture we stated Kesten's Theorem for Percolation on grid, which is:

IF $p > \frac{1}{2} \Rightarrow Pr[\exists \text{ infinite component}] = 1$.

To prove this theorem we introduced two lemmas.

Lemma 1. Let $p \geq \frac{1}{2}$ and R be an $m \times n$ rectangle. Then

$$I_p(e, H(R)) \leq 2.Pr_{\frac{1}{2}}[\text{origin is on a path of length } min(\frac{m}{2}, \frac{n}{2})].$$

Recall that $I_p(e, H(R))$ is the influence of edge e on event $H(R)$, assuming percolation is with probability p.

Lemma 2. Let $p > \frac{1}{2}$ be fixed and let R_n be a $3n \times n$ rectangle. Then $h_p(3n, n) \to 1$ as $n \to \infty$.

What Lemma 1 says is that as the rectangle gets bigger the influence of any particular edge on existence of a horizontal path gets smaller. We use this and invoke two other lemmas, due to Margulis-Russo and Friedgut-Kalai.

Margulis-Russo: For an edge e_i that is being included with probability p_i, and for any event E.

$$\frac{\partial}{\partial p_i} Pr[E] = I_{p_i}(e_i, E)$$

Friedgut-Kalai Inequality: Suppose $1 > Pr[E] \geq 0$ and $I(X_i, E) \leq \delta$, then there is a function f such that,

$$\sum_i I(X_i, E) \geq f(\delta) \quad and \quad \delta \to 0, \quad f(\delta) \to \infty.$$

The Friedgut-Kalai inequality states that if influence of each random variable on an event E is small (smaller that δ) then as the influence of each variable (δ) gets smaller, the sum of influences of all variables goes to infinity.

What Lemma 2 states is that as the rectangle gets bigger (goes to infinity) the probability of existence of a horizontal path, get bigger (goes to 1).

Now first we show that by having lemma 2 we can prove Kesten's theorem. we will do this in this manner :

a) Lemma 2 $\Rightarrow \quad h_p(2^{k+1}n, 2^k n) \geq 1 - \frac{\epsilon}{2^k}$.

b) $h_p(2^{k+1}n, 2^k n) \geq 1 - \frac{\epsilon}{2^k} \Rightarrow Pr($ Infinite open cluster exists $) \geq 0$.

Proof of b) Let us define the event E_k, the existence of a open horizontal crossing in a rectangle of $2^{k+1}n \times 2^k n$.

If event E_k happens for all$k$, then we can say that we have an infinite open component.

$$Pr[E_k^c] < \frac{\epsilon}{2^k}.$$

$$Pr[\cup_k E_k^c] \leq \sum_{k=0}^{\infty} \frac{\epsilon}{2^k},$$

$$\leq 2\epsilon < 1.$$

Proof of a) By lemma 2, we have that

$$\forall \epsilon > 0, \quad \exists n_0 \quad s.t. \quad \forall n > n_0 h_p(3n, n) \geq 1 - \epsilon.$$

We want to calculate $h_p(4n, n)$ in terms of $h_p(2n, n)$.
We can compose $4n \times n$ rectangle by putting 4, $n \times n$ squares next to each other. Now we can express the probability of existence of an open horizontal crossing in rectangle of $4n \times n$ in terms of existence of open horizontal crossing in smaller $2n \times n$ rectangles, we look at

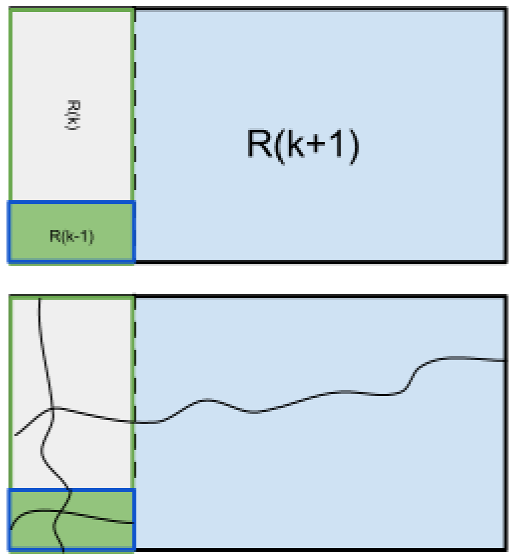

Figure 1: Infinite Open Component.

each two neighboring square as a $2n \times n$ rectangles. If we have a open horizontal crossing in each of these three smaller rectangles and we also have two open vertical crossing in each of the middle square, these paths together form an open horizontal crossing in the bigger $4n \times n$ rectangle. Therefore we can write:

$$h_p(4n, n) \geq h_p(2n, n)^3 h_p(2n, n)^2,$$

$$\geq h_p(2n, n)^5,$$

Now we want to calculate $h_p(4n, 2n)$ using $h_p(4n, n)$.

$$h_p(4n, n) \geq 1 - (1 - h_p(4n, n))^2,$$

$$= 1 - (5\epsilon)^2 = 1 - 25\epsilon^2 \geq 1 - \frac{\epsilon}{2}.$$

Proof Sketch of lemma 2

We look at the influence of a particular edge. we find an upper bound on the influence of an edge, which would suggest that $h_p(3n, n) \to 1$ as $n \to \infty$.

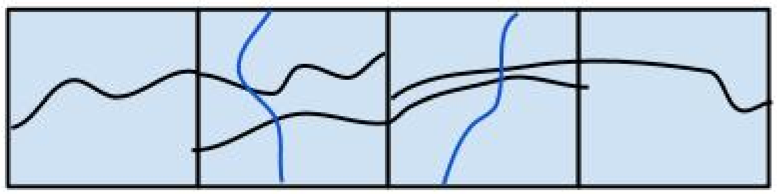

Figure 2: Composing a larger horizontal path with smaller paths.

By Harris's theorem we have,

$$Pr[\text{ Origin in an open path of length n}] \to 0 \text{ as } n \to \infty.$$

Let R be a $3n \times n$ rectangle, by lemma 1, we have

$$I_p(e, H(R)) < \delta \text{ where } \delta \to 0 \text{ as } n \to \infty.$$

By Friedgut-Kalai:

$$\Sigma_{e \in R} I_p(e, H(R)) = f(\delta), \text{as } \delta \to 0, f(\delta) \to \infty.$$

By Margulis-Russo:

$$\frac{d}{dp} Pr[X(R)] = \sum_{e \in n} I_p(e, H(R)) \to \infty \text{ as } \delta \to 0.$$

This is true as $n \to \infty$.

Multi-commodity Flow

We'll first start with the definition of the problem, and then describe some paradigm that have been used to solve this problem, and give some examples. Multi-commodity Flow is a generalization of network flow. In network flow problem, which is an example of the hardest problems that we can solve in polynomial time, you are given a capacitated directed graph, a node is specified as *source* s and another is specified as *sink* t, and the goal is to maximize the flow in the graph without violating the capacity constrains.

We can also call this a Single-commodity flow problem. In multi-commodity flow, we are again give a capacitated directed graph, the difference is that we have several commodities (s_i, t_i), and we want to maximize the total flow, without violating the capacity constrains and we also want to make sure that the flow that starts at s_i will end up at t_i for all pairs of source and sink in the graph.

Multi-commodity	Single-commodity Network Flow
Capacitated directed graph G.	Capacitated directed graph G.
$C : E \to \mathbb{R}$, k pairs of source and sink (s_i, t_i).	$C : E \to \mathbb{R}$, source s, sink t.

We introduce the flow variables $\forall 1 \leq i \leq k$, $f_i : E \to \mathbb{R}$. For every fixed i $f_i(e)$ is the flow that goes through edge e that started from source s_i. Using these variables we can write the LP problem that describes Multi-commodity flow:

We want to maximize the total flow from all sources, therefore,

$$Max \sum_i \Big(\sum_{e \text{ out of } s_i} f_i(e) \Big).$$

We have the condition that for every pair (s_i, t_i) the flow that reaches a vertex from s_i, (meaning sum of all $f_i(e)$ that go into that vertex)must be equal to the amount of flow that leaves that vertex (the sum of all $f_i(e)$ that leave the vertex), for all vertices except sources and the sinks.

$$\forall i \sum_{e \text{ into } u} f_i(e) = \sum_{e \text{ out of } u} f_i(e) \quad u \neq s_i, t_i.$$

The last set of constrains are the capacity constrains, we need to make sure that sum of flows that go through an edge, taken over commodities, do not exceed the capacity of that edge for all edges in the graph,

$$\forall e \in E \quad \sum_i f_i(e) \leq c(e).$$

It is important to notice the effect of having r pairs of commodities (s_i, t_i) instead of having r sources and r sinks. If we do not specify them as pairs, and add the constrain that the flow starting from s_i has to end at t_i, it would be the same as having only one source and one sink, because we can add a node s' and connect it to all sources with directed edges from s' to each source, and add a node t' and connect all sinks to it with a directed edge. Now if we solve the Max flow problem for this graph it would be the same as finding the total max Flow for the original graph since we do not care about how much flow goes from a specific source to a specific sink.

Facts known about Network Flow:

1. We can solve it using in polynomial time, using Augmenting paths algorithm, solving the LP etc.
2. The Min-Cut Max-Flow theorem. The maximum flow that can be sent from source to sink in a graph is equal to the size of the minimum cut of the graph.
3. If all capacities are integers then there exists an optimal flow that is an integer.

MCF:

1. We can solve it using in polynomial time, by solving the LP.
2. The Min-Cut Max-Flow theorem does not hold. Obviously $Max - flow \leq Min - cut$ always hold but we can not show $Max - flow \geq Min - cut$. In fact Max-flow can be much less than the Min-cut in many cases.
3. Solving the problem become hard if we insist on not breaking flow in fractions. Note that if we solve the LP, the answers are rational numbers, therefore they might not always be integers. One question is that if we can replace that flow somehow to have integer flows. The answer is that, we can not, and this problem is NP-hard if we insist on all integer flows.

Different Versions of MCF

Unsplittable Multi-commodity flow For each commodity the flow is on a single path.

Integral splittable flow Splitting the flow is allowed, but all flows need to be integers.

We can use an LP solver to solve MCF, but us may be too expensive for large problems. Next lecture we will a give combinatorial algorithm, called Multiplicative-weights algorithm for MCF.