College of Computer Science
Northeastern University
Wireless Networks CS G250

Lecture 13
October 27 2008
Lecturer: R. Sundaram

Capacity of Wireless Networks

1 (Piyush Gupta & P.R. Kumar, 1998)

In past lectures, we have seen how to obtain the capacity of a link between a sender and a receiver in both discrete and continuous cases. We have also seen several techniques to make newtorks out of links in different ways and environments. It seems natural to ask the question of what is the capacity of a wireless network.

Such is the subject of the paper by Piyush Gupta and P.R. Kumar. They present a fundamental result, even though the assumptions of which have been under criticism, the capacity of a wireless network is not only limited, but decreases by a function of the number of nodes, regardless of how the nodes use each other to relay communication.

Theorem 1 *When n identical randomly located nodes form a wireless network, the throughput $\lambda(n)$ obainable by each nodes is $\Theta(polylog(n)/\sqrt{n})$*

The proof for the upper bound of the capacity presented here is the same as in Peraki and Servetto's paper: *On multiflows in random unit-disk graphs and the capacity of some wireless networks.*

Proof: For $O(\text{polylog}(n)/\sqrt{n})$:

Consider a 1 by 1 square, in which n points have been set at random. We assume that if 2 nodes are within a distance r_n, a link between them exists, and therefore they can communicate with a capacity C bits per second[1]. Every node s chooses a random node t with which to communicate, creating a pairing of the n nodes. We wish to find out the capacity each $s \rightarrow t$ conversation has.

Taking a 1/2 cut of the square, we expect to have $n/2$ nodes on each side of the cut, and therefore, n/4 conversations cross said cut. The number of links that cross the cut is an upper bound on the capacity of a flow of bits from s to t.

[1]This assumption has been cirticized.

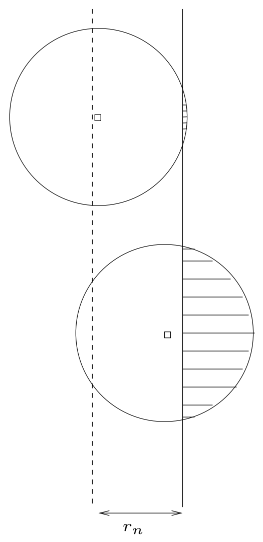

Figure 1: Areas crossing the half-cut

For a communication between s and t to exist, a path from s to t must exist. For the entire network to be connected, it can be shown that the minimum value for $r_n = \sqrt{\frac{\log n}{n}}$

The amount of capacity for each conversation is given by

$$\frac{\text{\# of links crossing the boundary}}{\text{\# of conversations}} \times C$$

To obtain the number of links crossing the boundary, consider the area r_n units left of the cut. Any node x falling inside this area who has another node y within r_n of it on the other side of the boundary, has a link to y. So, the expected number of links is the expected area across the boundary times $\binom{n}{2}$.

As shown in Figure 1 the area across the boundary starts at 0 when the node is r_n away from the boundary and grows as a function of r_n^2 as it approaches the boundary. Integrating, we obtain that the area is cr_n^3, and thus the expected number of links is:

$$E[\text{\# of links crossing the boundary}] = cn^2 \times c' r_n^3$$

The capacity is then

$$c\frac{n^2 r_n^3}{n} = cn\frac{\log n}{n}\sqrt{\frac{\log n}{n}} = \frac{(\log n)^{\frac{3}{2}}}{\sqrt{n}}$$

■

2 Capacity of Wired Networks

A wired network is simply a directed weighted graph $G(V, E)$ such as the one in Figure 2. The capacity of a conversation from s to t is an instance of a max-flow, min-cut problem. The central observation is that the capacity of the network is upper bounded by the sum of the capacities that cross any cut that has s on one side and t in the other.

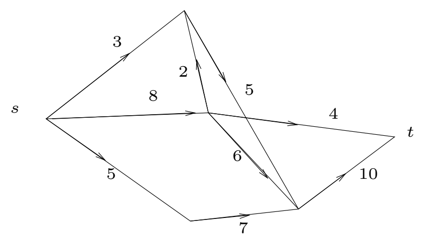

Figure 2: Wired network. The edges' weight is the link capacity

If any cut of the graph is at least an upper bound for the capacity of the network, then it would follow in our intuition that the one cut that minimizes the capacity is the cut that comes closest to the capacity of the network. In fact, such a cut achieves the capacity of the network as we shall see.

The idea is to start with some flow, and check if there is a path from s to t in which the capacity of every link in the path is still greater than the flow in such link. Then, we increase the flow as much as is possible in such path and then we update our flow information accordingly (intuitively, we would remove the path, but we will see this is not the right way to go). If we do this until we run out of paths to the sink, then we will have found a min-cut.

(continued in next lecture)