9/21/14

Lecture 9

Communication Networks

Prep

- PS5 (congestion of butterfly, construct 16-bit debruijn sequence, number of shuffles for 52 and 54 card decks, bisection of grid, find paths with low congestion)

- Rec9 (Benes result)

- 2 card tricks

- transparency machine setup

­Take

- Cards (with special prep for deBruijn trick x2)

- Transparencies (including blanks and markers)

- Handout for networks

- Transparency machine??

- Prizes (6 Tosci’s??)

Handout: Communication networks

Reminder: PS 5 due Monday at 7:30pm

Today, we are going to talk about the use of graphs in communication networks. Graphs are a very good way to model communication networks because we can use the nodes of the graph to represent switches or routers and sources or destinations, and we can use the edges to represent wires or fiber or transmission lines through which data can be sent.

For some networks, like the internet, the corresponding graph is gigantic and chaotic-looking, with tens of millions of nodes and little apparent structure.

For other networks, like those used in parallel computers, specialized circuits, or certain kinds of switches, the underlying graph is highly structured and smaller—maybe containing thousands of nodes connected in a well-organized fashion instead of millions of nodes with haphazard connections.

In both cases, the kinds of things that we try to optimize are similar. The goal is usually to find short paths in the network through which the data can be routed and to minimize the amount of congestion that arises as lots of data moves from various sources through the network to its destination. By finding short paths and minimizing congestion, we can generally minimize the latency (or time that it takes for a packet of data to get through the network) and we can maximize the throughput, which is useful when it comes to streaming video through the network since with higher throughput, you can get a higher quality picture.

We will start today by taking a look at some of the most popular graphs used in well-structured switching networks and parallel computing. Then, if we have time, we will finish up by talking about the internet.

One of the simplest and most popular networks is the 2-dimensional array (AKA the 2-dim grid or crossbar). For example, the 4x4 crossbar is shown in the handout and on the screen.

Everyone have handout?

Put on transparency

In this case, there are four inputs or sources for data and four outputs or destinations for the data. And there are 16 2x2 switches arranged in a 4x4 grid. Each switch can take a packet or stream of data from above or the left and route it down or to the right.

Show on graph

So if we wanted to find a route from IN\_1 to OUT\_2, we would just go across from IN\_1 until we got the the 3rd column and then we would head down to OUT\_2.

Show on graph

This same approach works for any input and any output. So by using 16 2x2 switches arranged in a grid, we have in effect, constructed a 4x4 switch.

Q: Suppose that we wanted to build an NxN switch. How many 2x2 switches would we need using this approach?

A: N^2.

That is a lot as N gets large—for example, if N=100, then we would need 10,000 2x2 switches to build the crossbar. Lets write this down for comparison purposes later.

Make table on RHS Board and SAVE

Network # switches

2-d grid N^2

Q: How long is the path from IN\_1 to OUT\_2?

A: 6 edges.

Path length can be measured in terms of the number of nodes or the number of edges, but for this course, we’ll usually just count the number of edges.

Q: Is there any shorter path between IN\_1 and OUT\_2?

A: No. There are lots of other paths but all of them must go across 3 edges and down 3 edges. So the length of any path is 6.

Show alternate path on slide

The **shortest** path between a pair of nodes in a graph is said to be the distance between the nodes.

Def: The distance between nodes u and v in a graph is the length of the shortest path between u and v.

Questions?

In a communications network, we are often concerned about the distance between the input and output that are farthest apart and we call this quantity the **diameter** of the network.

Def: The diameter of a network is the distance between the input and output that are farthest apart.

Q: For example, what is the diameter of the 4x4 grid?

A: 8. Since IN\_0 and OUT\_3 are distance 8 apart and there is a shorter path for every other pair of inputs and outputs.

Show on slide

Q: What about for the general case? What is the diameter of the NxN grid?

A: 2N. Since you might have to go across N edges and then down N edges in the worst case.

Show on slide

Let’s write that down for comparison later.

Network # switches Diameter

2-d grid N^2 2N

The diameter is one way to measure the worst-case latency in a network. 2N is not necessarily terrible—a lot better than N^2, but we’ll see how to do better with other networks later.

Questions?

The last thing we will worry about in terms of the performance or usefulness of a switching network is whether or not it is going to have problems with congestion. The congestion of a set of paths is defined to be the number of paths that intersect at the most congested node. In particular:

Def: The congestion of a set of paths in a network is the max (over all nodes v) of the number of paths that go through v.

Go to slide

For example, suppose that we have the path from IN\_1 to OUT\_2 from before, and we add a path from IN\_2 to OUT\_0 and also from IN\_0 to OUT\_1.

Q: what is the congestion for this set of 3 paths?

A: 2.

Show on slide

And even if we added a 4th path from IN\_3 to OUT\_3, the congestion is still 2.

Show on slide

In fact, the really nice thing about the 2-d grid is that as long as we make the paths go across and down (with just one turn), then as long as there is at most one path starting at any input and at most one path ending at any output, then the congestion is always at most 2.

Show on slide

Q: anyone see why this is true?

Q: what paths can use a row edge?

A: only the path from the input on that row.

Q: what paths can use a column edge?

A: only the path with a destination in that column.

So each edge has at most one path, which means that each node can only have two paths going through it (one for row edge and one for column edge).

And, for the same reason, the same is true for the NxN grid.

In general, we define the **congestion of a routing problem** on a network to be the congestion of the best set of paths for the routing problem—i.e., the set of paths that minimizes the congestion.

Def: The congestion of a routing problem is the congestion of the best set of paths for that routing problem.

For example, we could pick stupid paths for routing on the grid that would have more than congestion 2, but that would not make sense so we just worry about the congestion of the best set of paths.

Show on slide

The nice thing about the grid is that the congestion is 2 for any routing problem where there is at most one path from any source and to any destination. That motivates the definition of the **congestion of a network** to be the congestion of the worst possible routing problem.

Def: the congestion of a network is the congestion of the worst case routing problem with at most one path from any source and to any destination.

Are there any questions about this definition? It’s a little tricky since you need to find the worst case routing problem, then the best set of paths for that problem, and then look at the most congested node for that set of paths. For example:

Ex: the congestion of the NxN grid is 2 (since **∀** routing problems, **∃** paths, s.t. **∀** nodes v, at most 2 paths go through v).

No matter what inputs need to send data to what outputs (as long as each input needs to reach just one output), you can find a set of paths where every node has at most 2 paths going through it.

Questions?

So let’s write that on the board for comparison later.

Network # switches Diameter Congestion

2-d grid N^2 2N 2

The fact that the 2-d grid has such low congestion is a key reason why it is such a popular architecture. And, of course, it is very simple, which always helps. But as N grows, the number of switches becomes a big issue and the diameter can start to be an issue in terms of latency.

If we just cared about the number of switches and the diameter, then it is easy to get a network that does a lot better. We could just use a complete binary tree. For example, you can see a CBT with 4 inputs and 4 outputs at the leaves in the handout and on the screen.

Show CBT and describe path from IN\_1 to OUT\_2.

Q: How long is the path from IN\_1 to OUT\_2?

A: 6. Same as for 4x4 grid.

But in general, the path lengths in the N-input CBT are a lot better than the path lengths in the N-input 2d grid.

Q: What is the diameter of the 4-input CBT?

A: 6.

Show it on slide

Q: what about for larger CBTs? What is the diameter of the N-input CBT?

A: 2logN + 2.

Show it on slide and explain base 2 log.

Put that on board.

Network # switches Diameter Congestion

CBT 2logN + 2

Q: How many switches does the N-input CBT have?

A: 1 + 2 + 4 + … + N = 2N+1

Show on slide

Put data on Board

Network # switches Diameter Congestion

CBT 2N + 1 2logN + 2

So this is looking great! Linear number of switches and logarithmic diameter. As N gets big, the CBT looks a lot better than the 2d grid.

Do comparison with N=1000

Q: does anyone see the problem with the CBT?

A: Congestion!!! Every path might need to go through the root! The root is a big bottleneck.

Show on slide

Q: so what is the congestion of the N-input CBT?

A: N.

Explain on slide.

Put data on board

Network # switches Diameter Congestion

CBT 2N + 1 2logN + 2 N

So CBT is good for number of switches and diameter and grid is good for congestion. What we would really like is the best of both in the same network. Then we would have the holy grail of communication networks.

There is a popular graph that gets close, called the butterfly. It’s used a lot in practice for switching as well as other applications, including signal processing and Fourier transforms. An example of the 8-input butterfly is shown in the handout and the slide.

Show on slide

Explain: row for every logN-bit binary number.

Explain: in ith level, you cross-connect rows that differ in the ith bit.

Q: How many levels?

A: (logN + 1). Base 2 log again.

Q: How many switches per level?

A: N.

So the total number of switches is N(logN + 1)

Put data on board

Network # switches Diameter Congestion

Butterfly N(logN + 1)

OK, not terrible—almost linear. Let’s check on the diameter. Suppose you want to go from IN\_1 to OUT\_3.

Show on slide: take path Up-Down-Down since going to row 011.

It turns out that you can get from any input to any output using the same approach.

Explain general path selection

Q: So what is the diameter of the N-input butterfly?

A: logN + 2.

Put data on board

Network # switches Diameter Congestion

Butterfly N(logn + 1) logN + 2

Even better than CBT! So far, so good. Now for congestion, the answer is a little tricky. It turns out that if you have a random routing problem, then with high probability, the congestion is low—around logN. (We will come back and prove this after we learn about probability later in the term.)

But there are rare worst-case routing problems that make the congestion be higher—as high as sqrt(N). We’ll look at those in homework.

Put data on board

Network # switches Diameter Congestion

Butterfly N(logn + 1) logN + 2 sqrt(N)

So the hunt for the holy grail continued. It turns out that the best network was discovered by a fellow named Vaclav Benes at AT&T Bell Labs in the 1960s. Benes made the amazing discovery that if you put 2 butterflies back-to-back, then the congestion drops from sqrt(N) to 1, which is perfect. It means that there is never any congestion.

Show benes slide and explain network.

It’s a little tricky to construct the optimal paths for any routing problem where each input needs to reach only one output. The construction proceeds by recursion and the proof uses induction. You will do it in recitation tomorrow. One key thing to observe is that if you slide off the first and last levels of the benes network, you get two benes networks of half the size and so the routing will proceed recursively.

Show on slide

Put data on board

Network # switches Diameter Congestion

Benes 2N(logN + 1) 2(logN + 2) 1

Questions?

While we are on the subject of well-structured interconnection networks, I want to mention a couple of other networks that are closely related to the butterfly and that tend to arise frequently in practice. One of the most well known is the hypercube.

Show how to get hypercube from butterfly on slides.

Show 3-dimensional hypercube on slide.

Show correspondence with butterfly.

Explain N nodes (rows) with degree logN.

Explain logN dimensions of edges.

The hypercube is often used as the communications network for parallel machines because it has low diameter (logN) and is good for routing paths--embeds good properties of butterfly and benes network.

Questions?

The other popular network that is similar to the butterfly is the deBruijn graph. The deBruijn graph is a little trickier to understand.

Show isomorphism of butterfly and then collapsing rows.

Explain nodes and edges.

Redraw again and label edges for last bit.

The deBruijn graph comes up in a variety of data processing applications, especially in computational biology. Amazingly enough, the structure of the deBruijn graph also serves as the basis for some pretty good card tricks.

Get volunteer and do shuffle trick.

Show card to class

Have volunteer agree I did not cheat and give him prize.

Q: How did I do it?

Explain two kinds of ways to do perfect shuffle. Show inshuffle and outshuffle with cards.

Explain that by 3 shuffles of one type or the other, I can move a card from any position to any other position.

Explain perfect shuffle is left-shift.

000--------A

001--------B

010--------C

011--------D

100--------E

101--------F

110--------G

111--------H

A E

B F

C G

D H

Outshuffle

000--------A A

001--------B E

010--------C B

011--------D F

100--------E C

101--------F G

110--------G D

111--------H H

Explain that outshuffle is left-cyclic shift. Explain this is multiplying by 2 (maybe subtracting N-1 if in bottom half)

Explain outshuffle is inshuffle and flipping last bit.

Outshuffle Inshuffle

000--------A A E

001--------B E A

010--------C B F

011--------D F B

100--------E C G

101--------F G C

110--------G D H

111--------H H D

Show solution to card trick as routing path in debruijn graph since edges are inshuffle and outshuffle movements of data.

Label edges as inshuffle or outshuffle based on whether last bit is flipped.

Show diameter of deBruijn graph is logN.

Questions?

Tell story of showing 8-year old on plain. Looked at me like I was senile. “it’s magic, of course.”

Explain percy diaconus. Perfect cuts and shuffle 52-card deck 8 times. Show 8 shuffles works for HW. One of 10 in world. Life history: circus, Harvard, first blackjack counter.

There is another card trick that percy invented based on the debruijn graph. To do this one, I need 4 volunteers.

Do trick. Cut deck twice. Red card =1.

0000 3 C

0001 A S

0011 3 S

0111 7 S

1111 2 D

1110 A D

1101 K H

1011 J H

0110 6 S

1100 Q H

1001 9 H

0010 2 S

0101 5 S

1010 10 H

0100 4 S

1000 8 H

Give certificates to volunteers

Explain deBruijn sequence in terms of red and black cards.

Show debruijn sequence from Ham path in graph.

Talk about debruijn sequence from eulerian path in graph—left for homework.

Questions?

OK, that’s all we will say about well-structured networks.

Well structured networks are always better to work with than unstructured networks since you can design for short paths and low congestion routing solutions. Just like it is easier to drive around in a well-planned city with wide rectilinear streets.

But sometimes the network just grows chaotically, like the internet and there is no apparent structure. This makes finding algorithms for good paths harder since they need to work for any network.

Explain BGP in Internet:

IP addresses announced and propogated—router keeps next best hop, converges to shortest paths.

Does not consider congestion!

Carriers trick it to save cost.

Israel story

L3 story

No security—steal IP addresses for SPAM and data theft.