Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.829 Fall 2002 **Problem Set 1** September 10, 2002

This problem set has nine questions, each with several parts. Answer them as clearly and concisely as possible. You may discuss ideas with others in the class, but your solutions and presentation must be your own. Do not look at anyone else's solutions or copy them from anywhere. Turn in your solutions in on **Tuesday, September 24, 2002** in class.

1 Multiplexing

In this problem, we will compare statistical multiplexing to time-division multiplexing (TDM) to understand the differences between packet switching and circuit switching.

In our statistical multiplexing scheme, packets of all sessions are merged into a single queue and transmitted on a first-come first-served (FCFS) basis. Our TDM scheme is the same as the one described during the first lecture (see the L1 notes).

A switch is said to be *work conserving* if the only time it is idle is when there are no frames waiting for service.

1. Is our TDM scheme work conserving? What about our statistical multiplexing scheme?

   Answer:

   No, the TDM scheme is *not work conserving*. This is because TDM scheme schedules a slot for a session even if there is no data on that session (i.e. it wastes slots even though it could have scheduled transmission for some other session instead in that slot).

   The statistical multiplexing scheme is *work conserving*. This is because it never wastes a slot as long as there is data for some session.

2. Let's study the impact of statistical multiplexing on queuing delays. Suppose there are N concurrent sessions each with a Poisson traffic stream with rate λ frames/second. Also suppose that frame lengths are exponentially distributed, such that the average rate at which frames are serviced at the switch is μ frames per second ($\mu > N\lambda$). What is the average delay seen by a frame in TDM and in statistical multiplexing? What is the physical interpretation of your result?

   Answer:

   The average queuing delay for a FIFO queue with transmission rate Poisson with mean μ and arrival rate λ is given by[1]:

   $$T = \frac{1}{\mu - \lambda}$$

[1] See any book on queuing theory, or Bertsekas and Gallager, pg. 170.

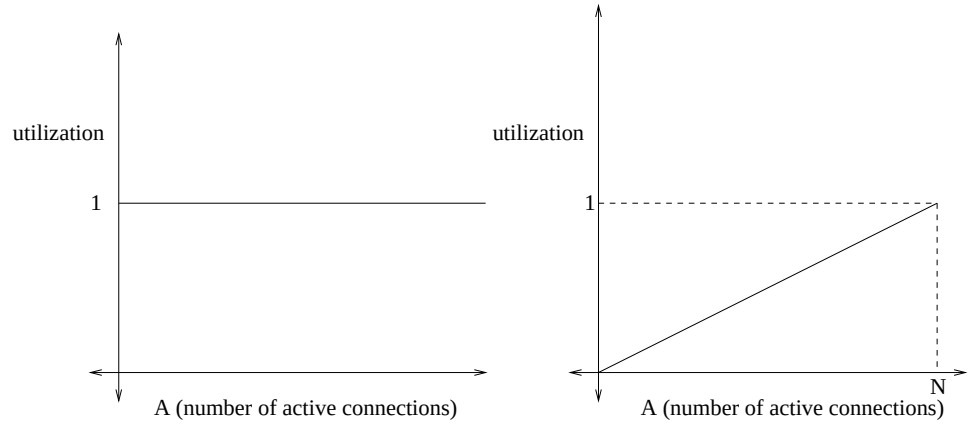


Figure 1: Utilization of output link for Statistical Multiplexing (left) and TDM (right)

In case of statistical multiplexing, all the input streams are merged into one queue and served from there. Thus, the total input rate to this stream is $N\lambda$. The service rate of this queue is μ. Hence, the average queuing delay per packet is given by:

$$T = \frac{1}{\mu - N\lambda}$$

On the other hand, with the TDM scheme, the output stream is divided into N equal portions (time slots), one per input stream. Thus, each portion behaves like an $M/M/1$ (formal notation for a queue with Poisson arrival and service processes) with arrival rate λ and average service rate μ/N. Thus, the average delay per packet is:

$$T = \frac{1}{\frac{\mu}{N} - \lambda}$$
$$T = \frac{N}{\mu - N\lambda}$$

Thus, you can see that the average delay in a TDM scheme is N times that in statistical multiplexing. This is because the TDM scheme is not work conserving.

3. Assume that all the sessions send frames at a simple constant bit rate and that there are A active sessions at a given point in time out of a possible N, sharing an output link. What is the utilization of the output link when the aggregate input rate for the A active sessions is μ frames/second. Sketch this as a function of A for both statistical multiplexing and TDM.

   Answer:

   The plots are shown in Figure 1.

4. Explain why TDM has smaller variation in the delay of a frame through a switch, compared to statistical multiplexing. (This delay variation is sometimes called the *jitter*.)

   Answer:

   TDM schedules each stream independently of other streams and assigns time slots for a stream at regular intervals. Hence, burstiness of data in one stream does not effect other stream. Thus, TDM has lower jitter than statistical multiplexing. This is the reason why

telephony world uses TDM and why equivalent Voice over IP is not easy to achieve over the Internet, without appropriate Quality of Service mechanisms, which we will study later in the course.

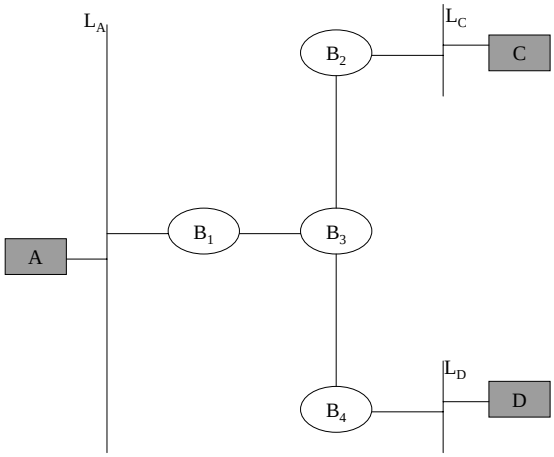


Figure 2: Bridge topology.

Table 1: Learning bridges

Transmission	B1	B2	B3	B4
A sends to C	A on L_1	A on $B_2 - B_3$ LAN	A on $B_1 - B_3$	A on $B_2 - B_4$
C sends to A	C on $B_1 - B_3$ LAN	C on L_2	C on $B_2 - B_3$	
D sends to C		D on $B_2 - B_3$ LAN	D on $B_4 - B_3$ LAN	D on L_3

2 Learning (about) bridges

Consider the bridge topology shown in Figure 2. Assuming that all the forwarding tables are initially empty, write out the forwarding tables at each of the four bridges $B1$ through $B4$ at the conclusion of the following transmissions:

1. A sends to C.
2. C sends to A.
3. D sends to C.

In the forwarding table at each node, identify the port by the unique LAN segment ($L_A, L_C,$ *or* L_D) reachable using that port, unless there isn't one, in which case use the identifier of the neighboring bridge to identify the port.

Answer: The answer is given in table 1.

3 Link packet traversals

Suppose source S sends a packet to destination D in a packet-switched network. Suppose the network topology and state in the switches do not change. Clearly explain why each of these statements below is true or false.

1. If datagram routing is used, correct forwarding can occur even if the packet traverses the same network link (and switch pair) in opposite directions.

Answer:

In datagram networks, the state that a packet carries helps the routers decide how the packet should be routed is never changed by intermediate routers. Hence, when a packet traverses the same link in opposite directions, the router at the edge of the link receiving the packet twice, will route it back on the same path it did in the first place. Hence, the packet ends up looping (and TTL field eventually kills it). Many students asked how a packet could traverse in opposite directions in a datagram network. This is often due to a routing fault.

2. If virtual circuit switching is used, correct forwarding can occur even if the packet traverses the same link (and switch pair) in opposite directions.

Answer:

In a virtual circuit based network, each router uses the tag (and maybe other information) to route a packet. These tags get modified by the routers. As a result, when a router receives the same packet again, it may have a different tag and hence, maybe routed on a different path now.

4 TCP retransmission timers

1. TCP computes an average round-trip time (RTT) for the connection using an exponential weighted moving average (EWMA) estimator:

$$y(n) \leftarrow \alpha r(n) + (1-\alpha)y(n-1)$$

where $r(n)$ is the n^{th} RTT sample and $y(n)$ is the average estimate updated after the arrival of the n^{th} sample. Suppose that at time 0, the initial estimate, $y(0)$ is equal to the true value, r_0. Suppose that immediately after this time, the RTT for the connection increases to a value R and remains at that value for the remainder of the connection. You may assume that $R >> r_0$.

Suppose that the TCP retransmission timeout value at step n, $RTO(n)$, is set to $\beta y(n)$. Calculate the number of RTT samples before we can be sure that there will be no spurious retransmissions. Old TCP implementations used to have $\beta = 2$ and $\alpha = 1/8$. How many samples does this correspond to before spurious retransmissions are avoided, for this problem? (Today's TCPs use the mean linear deviation rather than $\beta y(n)$ as the RTO formula.)

Answer:

The important point that we wanted to get across from this question is that due to smoothing of RTT estimation, if there is a sudden jump in actual RTT, the RTT estimate may take quite a while to reflect the new RTT. This can result in spurious retransmissions. Basing rto values on mean deviation helps reduce this convergence time and avoid spurious transmissions (see Jacobson's paper).

After the actual RTT becomes R, we will avoid spurious retransmissions when the RTT estimate becomes R/β. Suppose this happens at the n^{th} sample. Then, $y(n) = R/\beta$. We need to solve for n.

Now,

$$y(n) \quad = \quad R\alpha + R\alpha(1-\alpha) + ... + R\alpha(1-\alpha)^{n-1} + r_0(1-\alpha)^n$$

$$
\begin{aligned}
&= r_0(1-\alpha)^n + \alpha R[1 + (1-\alpha) + .. + (1-\alpha)^{n-1}] \\
&= r_0(1-\alpha)^n + \alpha R\frac{1-(1-\alpha)^n}{1-(1-\alpha)} \\
&= r_0(1-\alpha)^n + R[1-(1-\alpha)^n] \\
&= r_0(1-\alpha)^n + R - R(1-\alpha)^n
\end{aligned}
$$

Putting $y(n) = R/\beta$ and solving for n, we get $n = log_(1-\alpha)(R/(R-r_0))(1-1/\beta)$

Putting $\alpha = 1/8, \beta = 2$ and neglecting r_0, we get

$n = log_{7/8}(1/2) = 5.19$.

Hence, after 6 samples, there can be no spurious retransmissions.

2. Suppose that, instead of the EWMA estimator, TCP computed the average RTT by averaging over a fixed amount of past history. I.e.,

$$y(n) \leftarrow \frac{\sum_{i=n-k}^{n-1} r(i)}{k}; k > 1.$$

Now suppose that that the previous k samples are all equal to r_0, the true value, and that the RTT for the connection increases to a value $R(>> r_0)$ and remains at that value for the remainder of the connection. Using the same RTO as in part 1, calculate the number of RTT samples before we can be sure that there will be no spurious retransmissions.

Answer: Assume at time 0, $r(0) = r_0$ and the previous k samples are all equal to r_0. We know $r(n) = R$, for $n > 0$. When $0 < n \leq k$, $k - n + 1$ samples have the old value r_0 and $n - 1$ samples have the new value R. When $n > k$, all k samples are of the new value R. Thus,

$$y(n) = \frac{\sum_{n-k}^{n-1} r(i)}{k} = \begin{cases} \frac{(k-n+1)r_0+(n-1)R}{k} & 0 < n \leq k \\ R & n > k \end{cases}$$

After the estimation $\beta y(n) > R$, we will avoid spurious retransmissions. Since $\beta > 1$, n must be less than k. Given $R >> r$, solve

$$\frac{\beta((k-n+1)r_0 + (n-1)R)}{k} > R$$

We get $n > k/\beta + 1$.

3. In your opinion, which estimator is a better one for TCP? Why?

XXXX this answered a slightly different question, need to check it over

Answer: As shown above, for the history-based estimator to get the same convergence speed as that of EWMA, k should be no greater than 10. Now suppose there is a spike $r(1) = 100r_0$ in the RTT sequence, where $r(n) = r_0$, for $n \neq 1$. For the EWMA estimator,

$$
\begin{aligned}
y(n) &= r_0 - \alpha(1-\alpha)^{(n-1)}r_0 + \alpha(1-\alpha)^{(n-1)}r(1) \\
&\approx \alpha(1-\alpha)^{(n-1)}r(1)
\end{aligned}
$$

We have $y(10) \approx 1/8 \cdot (1-1/8)^9 \cdot 100r_0 = 3.76r_0$. If all packets in the window after the 10th packet are lost, it will take $7.52r_0$ time to trigger retransmissions. For the history-based

estimator, the 10th estimation $y(10) = (100r_0 + 9r_0)/10 = 10.9r_0$. Thus it takes longer to trigger TCP retransmissions.

Unlike the history-based estimator, the EWMA estimator weights the recent samples more than the older ones. It gives good performance both in catching up a sudden change in RTT values and in filtering out spikes.

Another advantage is implementation-wise. The history based estimator keeps the values of k samples, while the EWMA only keeps the most recent RTT sample, yet it gives good enough performance.

5 TCP checksums

TCP has an end-to-end checksum that covers part of the IP header, in addition to the TCP header and data. When the receiver receives a data segment whose checksum doesn't match, it can do one of two things:

1. Discard the segment and send an ACK to the data sender with the cumulative ACK field set to the next in-sequence byte it expects to receive, or
2. Discard the segment and do nothing else.

Is one action preferable to the other (or are they both equivalent)? Why? (You might look up the TCP and headers in any standard networking textbook; note that the header formats in Cerf & Kahn's paper aren't used any more.)

Answer: (ii) is preferable. The IP header contains important address information such as source/destination addresses and source/destination ports. In case the address information is corrupted, the receiver doesn't know for sure whether this packet is destinated for it nor whether this packet is from the correct sender. As TCP is a reliable transport protocol, sending an ACK using possible incorrect header information may do more harm than good.

6 AS interconnections

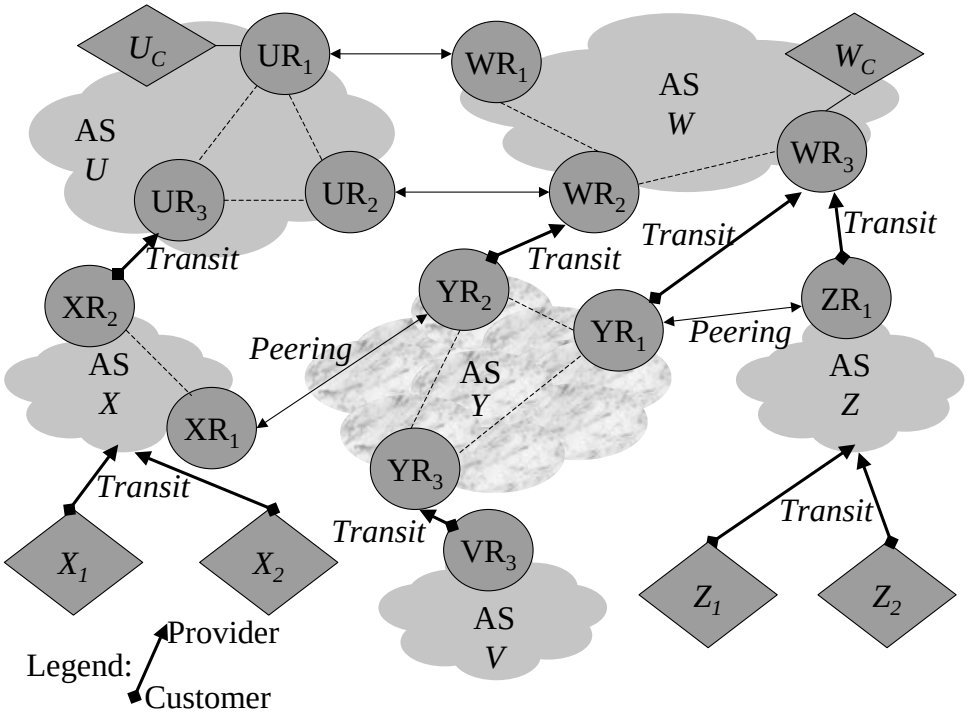
There are six AS's shown in the picture above: U, W, X, Y, Z, V. The diamond-shaped boxes are customers of the AS's (ISP's) they're connected to. Some relationships are marked: the *Transit* ones involve the higher AS (in the picture) providing Internet service to the lower one(s) for money. The X-Y and Y-Z interconnections are standard peering relationships. The circles with names like WR_1 stand for BGP routers; WR_1 refers to a BGP router in AS W. Within each AS, the dotted lines show IBGP interconnections.

1. Which AS does not have correct IBGP interconnections?

   Answer: AS W is missing a connection from WR_1 to WR_3.

2. Suppose an AS has k routers. What is the minimum number of IBGP sessions required for correct configuration, assuming no use of route reflectors or confederations?

   Answer: $k(k-1)/2$



3. Consider the peering relationship between Y and Z. Which of these statements are true?

   (a) Z will hear routes to V announced by Y, *and* may also hear routes to V announced by W.

   (b) Z *must* use a route to V announced to it by Y, since that's a route announced from a peering relationship.

   (c) Y will usually not announce routes to Z_1 and Z_2 to W.

   (d) Y will usually not announce routes to Z_1 and Z_2 to V.

   Answer: a and **c** are true. **b** is false; Z *should* use the route to V through Y, but is not obligated to.

4. U wants to ensure that packets sent to U_C from W are sent to it via UR_1 and not UR_2.

   (a) Clearly explain how it might try to do this. (A small picture explaining routing messages will help.)

   (b) Can it always ensure that the desired behavior happens? Why (not)?

   Answer: AS U can set different MED values on the advertisements sent from UR_1 and UR_2. AS U cannot ensure that the desired bahavior happens because AS W can ignore the MED values.

5. W would like to ensure that packets sent to W_C from X reach it via AS's X and U, rather than via AS Y. How can this be done? (Describe the BGP routing messages involved.)

Answer: W can pad the ASPATH on it's advertisement of W_C sent to AS Y. However, if AS X chooses AS Y based on a local-pref, W cannot prevent this without issuing a withdrawl of the route to W_C to Y. Normally, however, none of this will be necessary since Y will choose not to advertise the prefix to W_C to X.

6. W would like to ensure that packets sent to W_C from X_1 reach it via AS's X and U, and packets sent to W_C from X_2 reach it via AS Y. Can this be done with BGP? If so, how? (Describe the BGP routing messages involved.)

   Answer: No, AS W cannot do this using BGP.

7 Understanding BGP using table dumps

For this question, you will need to download the Routeviews routing table from `http://nms.lcs.mit.edu/6.829/ps/ps1/route-views.bgp.20020903.gz`

This file contains a Cisco BGP-4 routing table snapshot, taken at Oregon Route Views (`http://www.routeviews.org/`) on September 3, 2002. If you are curious about what other snapshots look like, you can find daily snapshots at `ftp://ftp.routeviews.org/pub/routeviews/bgpdata/`.

1. To start with, find the routing table entry for the MIT network.

   (a) What is the IP address of the best next hop from this router to MIT? How does this router know how to reach that next hop IP address?

   (b) How many AS's must a packet traverse between the time it leaves the router and the time that it arrives at MIT?

   (c) Use `traceroute` today to trace the route from MIT to the router that took the snapshot. Is the current route from MIT to the router the same as the reverse route in the trace data?

   (d) On September 3, 2002 at 4 pm EDT, the AS path to `route-views2.oregon-ix.net` from MIT was `10578 11537 4600 3701`. Why is this path not simply the reverse of the path from MIT to Routeviews? Why does this traceroute (which was run at the same time), not match the AS path?[2]

```
running /usr/local/bin/traceroute -A  198.32.162.102...
 1  anacreon (18.31.0.1) [AS3]  1 ms  1 ms  1 ms
 2  radole (18.24.10.3) [AS3]  6 ms  2 ms  1 ms
 3  B24-RTR-1-LCS-LINK.MIT.EDU (18.201.1.1) [AS3]  2 ms  2 ms  1 ms
 4  EXTERNAL-RTR-2-BACKBONE.MIT.EDU (18.168.0.27) [AS3]  185 ms  19 ms  2 ms
 5  192.5.89.89 (192.5.89.89) [AS1742]  1 ms  2 ms  3 ms
 6  ABILENE-GIGAPOPNE.NOX.ORG (192.5.89.102) [AS1742]  6 ms  7 ms  7 ms
 7  clev-nycm.abilene.ucaid.edu (198.32.8.29) [(null)]  20 ms  20 ms  24 ms
 8  ipls-clev.abilene.ucaid.edu (198.32.8.25) [(null)]  25 ms  25 ms  27 ms
 9  kscy-ipls.abilene.ucaid.edu (198.32.8.5) [(null)]  34 ms  36 ms  34 ms
10  dnvr-kscy.abilene.ucaid.edu (198.32.8.13) [(null)]  47 ms  45 ms  44 ms
11  pos-6-3.core0.eug.oregon-gigapop.net (198.32.163.13) [AS4600]  80 ms  78 ms  80 ms
12  nero.eug.oregon-gigapop.net (198.32.163.151) [AS4600]  77 ms  77 ms  78 ms
13  198.32.162.102 (198.32.162.102) [AS3582]  79 ms  79 ms  78 ms
```

[2] You can try this for yourself at `http://bgp.lcs.mit.edu/diag.html`.

(e) From the routing table file, what is the AS number for MIT?

(f) How many routes are there to get from this router to MIT?

(g) From the routing table, what is the best route to MIT? Why was this route selected as the best route?[3]

(h) What AS's do all of the different routes to MIT have in common? Which occurs most frequently? What is the likely relationship between the dominating AS and MIT?

(i) What IP network does the above AS correspond to? Again, all the information you need to answer this question is contained in the routing table. You can use `nslookup` to some host on this network to find out which company this is.

2. Several of the IP prefixes in the table are formatted as $w.x.y.z/m$. The mask field, m, specifies the length of the network mask to use when matching input destination addresses to entries in the table.

   (a) Write down the bit-wise operation to determine whether a destination address, A_i, matches a prefix A/m in the routing table. A_i and A are 32 bits each.

   (b) Find the first "Class C" CIDR address in the table (address prefix $\geq$ 192.0.0.0). How many class C networks does this address correspond to? What is the maximum number of routing table entries that this single CIDR address saves? Why is it that we can only infer the maximum, and not the actual, number of addresses that this CIDR address saves?

   (c) In the table, there are examples of groups of prefixes that have the same advertised AS path, but show up as separate entries in the routing table.[4]

      (i) Provide an example of non-contiguous prefixes (and the corresponding AS path) for which this is true. Why might non-contiguous prefixes have the same AS path?

      (ii) Provide an example of contiguous prefixes (and the corresponding AS path) for which this is true. This practice is often called *deaggregation*. Why might this be done?

3. Ben Bitdiddle is interested in studying the characteristics of the Internet using routing table snapshots. The Oregon Exchange has agreed to give Ben Bitdiddle some partial routing table snapshots from 1995 to the current day, including some snapshots from before the upgrade to BGP-4. They will give him snapshots containing the following:

   (a) Only the destination addresses.

   (b) Only the lines marked `*>`.

   (c) Only the paths, with best next-hops marked.

   Ben doubts that these partial snapshots could tell him anything interesting, but you disagree. What information about the evolution of the Internet could you infer from each type of partial snapshot?

Answer (Thanks to Mike Walfish):

[3] If you're interested, see the L4 notes or `http://www.cisco.com/warp/public/459/25.shtml` for an overview of the BGP decision process. Note that the process is slightly vendor-specific.

[4] For both parts of this problem, it's sufficient to find the existence of one AS path that is advertised more than once. It is *not* necessary to find two prefixes for which *all* advertised paths are the same.

1. (a) 4.0.4.90. This IP address corresponds to a Genuity router that is participating in a BGP session with the Oregon router (as is clear from the routeviews home page). This means that the Oregon router has conducted a TCP session with the Genuity router (since BGP runs over TCP) over some physical port. The Oregon router, when it wants to send to the next hop IP address 4.0.4.90, simply sends out this same physical port. The information about which physical port to use is stored in the forwarding table (and was presumably populated when the router first came up or when the TCP connection with 4.0.4.90 was first initiated).

   (b) Only 1, not including MIT (namely, it must traverse AS 1)

   (c) No, it is not.

   (d) Because MIT would rather send traffic cheaply over Internet2/Abilene, when it is allowed (which is when the traffic is between two educational institutions). MIT has a routing entry that directs traffic intended for U of Oregon (and other universities, presumably) to its Internet2 link. The traceroute likely does not match the AS path because the IP address-to-AS number resolution, in the traceroute, is accomplished via queries to a database that map IP addresses to the database owner's concept of AS. The AS numbers in the AS path are reported by the actual AS, and the database that maps IP addresses to ASes may no longer be aware of which ASes have which IP addresses (IP addresses can be resold). Moreover, there are cases in which the same logical AS uses different AS numbers. In this example, the queries to the database induced by the "-A" option can only return one AS number, so it might be returning one different from what the AS reports as its own. In any case, note that the AS Path in the MIT table is logically the same as the one traceroute indicates — the first hops are Harvard (10578 aka 1742), followed by Abilene/Internet2 links, followed by AS 4600, followed by two different AS numbers that refer to slightly different pieces of the Oregon organization.

   (e) The AS number for MIT appears to be 3

   (f) 23

   (g) The best route is the one with AS PATH=1 3. We cannot be totally sure why this route was selected as the best route. There are no LOCAL_PREFs set for this route, by inspection. So BGP should select the route with smallest ASPATH length, but there are two such routes: (1 3) and (293 3). None of the other BGP attributes (e.g., ORIGIN, MED, eBGP/iBGP, IGP path) appears to be relevant (since these two routes have the same state for each attribute or the attribute isn't relevant, as with MED, for comparing two routes that start at different ASes). One possibility is that the ROUTER ID criterion worked in (1 3)'s favor. Another is that, by inspection, (1 3) represents the path through the AS (1) with much more connectivity and reachability. The router may have an automated way of telling that the (1 3) route is more "credible", but we cannot be sure.

   (h) Aside from MIT's AS (3), all of the routes to MIT, except for (293 3), have in common only AS 1. AS 1 is likely MIT's service provider.

   (i) AS 1 corresponds to Genuity's IP network.

2. (a) `(A == (A_i & (-1 << (32 - m)))`

   (b) 192.0.32.0/20. This address corresponds to 16 class C networks (since class C networks are /24, and this is /20, representing a block of 16 class C addresses). The maximum number of routing table entries saved is 15 (16 networks minus 1 for the entry itself). We cannot infer the actual number of addresses that this CIDR address saves, just from the

address, for two reasons: 1) we do not know which networks are out there; it may be that there are other networks in the range given by the prefix but that a given router doesn't know about them (because, for example, they might not be advertised to the router in question). In that case, the number of routing table entries saved would be smaller than 15. 2) Furthermore, there could be other networks that fall into this 192.0.32.0/20 block but that have longer prefix matchings (so their routing occurs differently). An example would be the prefix 192.0.33.0/24. In this case, routing table entries are not saved

(c) i. See lines 125 and 409582 in the route views table. The prefixes are 6.1.0.0/16 and 128.37.0.0, respectively. The AS paths are equal and are (1 7170 1455). Non-contiguous prefixes might have the same AS path if the prefixes belong to the same logical entity. The prefixes could be discontinuous for several reasons. Two that come to mind are: 1) the entity bought two sets of addresses at two different times and 2) there was a merger/acquisition, so previously logically distinct entities with different IPs are now under the same roof, advertising the same AS path.

ii. See lines 672756 and 672779 in the route views table. The prefixes are: 171.162.208.0/20 and 171.162.240.0/20, with the AS path = (1 701 10794). Deaggregation might occur for the following reason: it is quite possible that two contiguous prefixes correspond to different logical organizations with the same provider (e.g., two different corporations). It is also possible that these corporations want different backup systems, so one fails over to, for example, AS 10 and the other fails over to AS 11, if the main AS that they share isn't working. In this case, the corporations will need separate routing entries so that when the failure occurs a packet can be matched differently for the two different organizations. Or it may be that the two contiguous blocks themselves represent different providers (who are buying service from bigger providers). In that case, it could very well be the case that each of the smaller providers runs its own BGP session and separately contributes its reachability information to the route-views router.

3. (a) With this information, we get a complete list of destination networks and masks. This time series data could tell us several things:

- How much of IP space has actually been allocated and is in use; if there are blocks of the 32 bit address space that "come online" (because, for example, they simply weren't represented at a previous point but later, a routing entry appears for them), we can infer growth of the Internet.
- This information can also tell us something about deaggregation: if an address block was previously represented by one routing entry and is now split into several entries with longer prefixes, then we can infer that addresses have been resold, or new companies or providers have come online, etc. We could, for example, estimate the growth in the number of autonomous systems.

(b) This time series data could tell us several things:

- We could get some of the same information as in (a), because the sheer quantity of lines we have been given tells us something about the size of the Internet and the number of distinct networks (since there is one of these lines per routing entry).
- This is a list of all of the "best" routes, that is, the routes that the Oregon router actually would have used to forward traffic. Since the Oregon router does not set the LOCAL_PREF attribute, this data tells us the minimum length paths (after ASPATH padding). Since the lines we have been given are associated to next hop

IP addresses, and since we can map IP addresses to AS, we can determine which ASes have grown in connectivity and which are likely to carry traffic and over which part of their IP space.

(c) So under this assumption, we are given no information about IP addresses, but we can still determine a number of things:

- First, note that this is actually more information than we use in question 8, to calculate degrees and provider-customer, sibling-sibling relationships. So we can do everything we did/will do in question 8
- We can also do much of what we did in parts (a) and (b) in this question (since we again can tell how many best paths there are, which tells us something about the number of routing entries, which tells us something about the growth of the Internet)

8 Inferring AS Relationships

As you know, the Internet is composed of about 14,000 distinct origin AS's that exchange routes to establish global connectivity, and that business relationships determine which routes are exchanged between each pair of AS's.

Recall that one network will re-advertise its customer routes to its peers and providers, but will not re-advertise routes heard from a peer to other peers or providers. With the knowledge of these rules and a view of a default-free routing table (or multiple tables), one can deduce relationships between AS pairs based on links that exist in the AS graph.

In *On Inferring Autonomous System Relationships in the Internet*[5], Lixin Gao observes that, because of these constraints, AS paths must adhere to one of the following patterns:

1. a series of customer-provider links (an *uphill path*)
2. a series of provider customer links (a *downhill path*)
3. an uphill path followed by a downhill path
4. an uphill path followed by a peering link
5. an peering edge followed by a downhill path
6. an uphill path followed by a peering link, followed by a downhill path

This is called the "valley free" property of AS paths. The hard question, of course, is: where is the "top of the hill"? Gao suggests using the AS in the path that contains the larges degree: that is, the AS that connects to the most other AS's.

We have provided a Routeviews routing table for you at `http://nms.lcs.mit.edu/6.829/ps1/route-views.bgp.20020903.gz`. (Note that the file is 8MB.) Your task is to produce a good guess about relationship between each AS pair in the table.

[5] You can find a copy of this paper at `http://www-unix.ecs.umass.edu/~lgao/ton.ps`. While you don't need to read the paper to solve this problem, you may find it helpful and interesting.

1. Produce CDF of AS degree (i.e., plot the fraction of AS's that have an degree of $< n$, for all $n > 0$). Also include a table of the "top 10" AS's for degree and the value of their degrees. Do not count a link from an AS to itself as an edge. Also, consider *all* AS paths that are given in the table (about 2.4 million paths), not just the best path for each prefix.

   Answer:

AS	*Degree*
701	2598
1239	1679
7018	1373
3561	796
209	792
1	595
702	587
3549	568
2914	512
3356	453

   Answers may have differed slightly depending on whether one counted degrees for AS sets.

2. For each of the following AS paths, list the transit relationships inferred for each pair, based on that path. *This is a two-step process.*

   First, for each AS path, note the transit relationships. For example, for the path $ABCD$, if C were the AS with the highest degree, you would write "Transit relationships: $A \rightarrow B$, $B \rightarrow C$, $D \rightarrow C$". This will give you a list of AS pairs that have transit relationships.

   Once you have scanned all AS paths, you may find that you have a commutative transit relationship: i.e., A transits B *and* B transits A. This is called a sibling relationship. For all pairs in the following paths, note which AS transits for the other, or if the two pairs have a sibling relationship.

   (a) 3130 2914 701

   (b) 8121 19151 3356 18566

   (c) 16150 8434 3549

   (d) 6539 701 7018

   (e) 7911 209 19092 3908 10947

   Answer:

   (a) 3130→2914→701

   (b) 8121→19151↔3356←18566

   (c) 16150→8434→3549

   (d) 6539→701←7018

   (e) 7911→209←19092↔3908←10947

   Common errors included missing the sibling relationships, which was an indicator that perhaps people may have failed to perform two passes through the table.

3. Finding the "top of the hill" by using the AS with the highest degree sometimes produces the wrong answer. Another way to do this is to view the AS paths from one vantage point as a directed graph, and using a reverse pruning algorithm to the AS graph in order to assign ranks to each AS.

   First, leaf nodes of the AS graph are assigned the lowest rank. Then, these nodes and their incident edges are removed from the graph. The nodes that are leaves in this new graph are assigned the next highest rank. The process repeats until the graph is strongly-connected (i.e., there are no leaves); each node in the strongly-connected component of the graph receives the highest rank. AS relationships are inferred by comparing rankings from AS graphs as visible from *multiple vantage points*[6].

   (a) What are advantages of using this type of ranking scheme over a power-law based scheme? What are the disadvantages?

   Answer: A high-level of connectivity does not imply that the AS is a top-level AS. Consider the company Internap, which is a customer that tries to get its customers better performance by buying many transit providers themselves. Similarly, a small network may peer with a lot of other small networks, but this connectivity does not imply anything about position in the AS hierarchy.

   One disadvantage is that the scheme will incorrectly assign a high rank to AS's that are on a long path to the strongly-connected component. Multiple vantage points can mitigate this effect, but, in general, this scheme requires much more data, and the algorithm is more complicated.

   (b) Why does this scheme require multiple vantage points to be effective?

   Answer: The AS graph as seen from one vantage point will not contain every edge, due to policy. Looking at the graph from one vantage point introduces considerable bias that arises due to policy.

9 Traffic Flow Patterns

People often want to know how much traffic they are sending to each neighboring AS. Network operators use traffic volumes to detect congestion, determine if they are violating peering agreements, or sending too much traffic on an expensive transit link. Typically, network operators use Netflow[7] to calculate these volumes. In the absence of Netflow, packet monitoring and routing information can provide a crude approximation of traffic patterns.

In this problem, you will answer the question: "How much traffic from the MIT Lab for Computer Science flows toward Internet2?". This requires two pieces of information: how much traffic is destined for each IP address, and which routes correspond to which prefixes. Note that the latter requires doing a longest prefix match for each IP address.

To answer this question, you will need the routing table as seen from MIT to determine which prefixes have routes via Internet2 and which have routes via Genuity. The routing table was collected at LCS on September 9, 2002 via an IBGP session with MIT's border router; the machine has no

[6] You can find the paper that describes this algorithm in detail at `http://www.ieee-infocom.org/2002/papers/594.pdf`.

[7] `http://www.cisco.com/warp/public/732/netflow/`

other BGP sessions. This routing table is available at: `http://nms.lcs.mit.edu/6.829/ps/ps1/mit.bgp.20020909.gz`.

Additionally, you will need to know how many bytes were destined for each IP address. These byte counts, produced from a trace on December 6, 2000 are available at: `http://nms.lcs.mit.edu/6.829/ps/ps1/20001206.byte.summary.gz`.

1. Why is there only one route per prefix in this table?

   Solution: The router is running an iBGP session with MIT's border router, which only re-advertises best routes. Since the router only has this one BGP session, it will only hear one best route per prefix.

2. Produce the longest-prefix match for each of the following IP addresses from the routing table we provided. (*Hint:* You don't have to implement the most efficient longest-prefix match; a simple sorting-based scheme should be sufficient for this problem.[8]

   (a) 150.65.236.70
   (b) 24.218.254.226
   (c) 20.138.0.10
   (d) 47.249.128.12

   Solution:

   (a) 150.65.0.0/16
   (b) None
   (c) 20.138.0.0/22
   (d) 47.249.128.0/18

   Some people missed the fact that there was no match for the second IP address. Note that 24.217.0.0/16 does not contain this IP address!

3. How much traffic leaves MIT from LCS via Internet2? Via Genuity?

   Solution:

   45.1% Internet2, 42.3% Genuity, 12.5% Unknown. Many people failed to get the right answer for this problem. The correct way to do this problem is to look at the *next-hop AS* in the AS path to determine which AS routes which prefix. Internet2 routes exit via AS 10578; Genuity routes exit via AS 1. Some people tried to use the next-hop IP address, and seemed to incorrectly assume that an Internet2 next hop was actually a Genuity IP address. It's much safer to use next-hop AS to solve this problem.

4. List at least two potential sources of inaccuracy that may result from using this method to measure traffic volumes.

   Solution:

   - The table snapshot was taken two years after the traffic counts.
   - Table snapshots don't reflect the fact that routes may change over the course of the time that the traffic volumes were counted.

   Several people said that the scheme would count retransmissions. This is true, but the question asks about total traffic, not about "useful data". When operators meter traffic for whatever reason, they're typically not concerned with retransmissions anyhow.

[8] However, you are welcome to use existing longest-prefix match implementations. For example, Perl has a convenient Patricia module for IPv4 route lookups that you may consider using if you do this problem by writing a Perl program.

5. Now suppose MIT wanted to send less traffic outbound via Internet2 (typically this would be a bad idea since the Genuity link is more expensive, etc., but assume for the sake of the problem that this is reasonable). What would be a good way for MIT's network operator to adjust the outbound traffic volumes? What if MIT wanted to adjust inbound traffic volumes?

   Solution:

   - Outbound: Adjust localpref values to prefer more routes via Genuity. Adjustment of localpref affects the choice of best route, and thus where traffic will flow for those prefixes.
   - Inbound: Use AS path prepending to discourage an upstream provider from using a particular route. This could be done in conjunction with deaggregation.

   There were many wrong answers for how to control inbound traffic. Several people suggested not advertising routes via Internet2 or Genuity, which is not a great idea, since this eliminates the route, even as a backup route, and could adversely affect connectivity.

   Others suggested adjusting MED values, which doesn't make any sense, since MIT has only one connection to each of its upstream providers (and this attribute is not comparable across multiple providers anyhow).