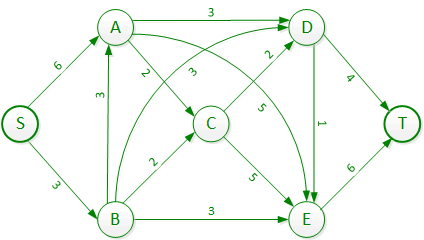
Homework B (15 pts)

1. [10 pts] Consider the following graph of a network with given flows:



Part A [4 pts]: Use the Edmonds-Karp algorithm to find the maximum flow from S to T in the graph below. On each line given in the table, list the path and the total augmented flow along that path. At the end, list the total flow through the network. When choosing among two different paths that are equally valid choices for the next augmentation in the Edmonds-Karp algorithm, choose the path that augments the flow by the greatest amount. (If you do this correctly, there is only one right answer). You may not need all four rows. List the total flow through the network.

|  |  |
| --- | --- |
| **Path** | **Augmented Flow** |
| SAET | 5 |
| SBDT | 3 |
| SADT | 1 |
|  |  |
| **TOTAL FLOW** | 9 |

Part B [1 pt]: In the table below, list the flow along each edge at the maximum network flow.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Edge | S🡪A | S🡪B | A🡪C | A🡪D | A🡪E | B🡪A | B🡪C | B🡪D | B🡪E | C🡪D | C🡪E | D🡪E | D🡪T | E🡪T |
| Flow | 6 | 3 | 0 | 1 | 5 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 4 | 5 |

Part C [2 pts]: In the table below, list the residual capacity of each edge in each direction in the final residual graph.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Edge | S🡪A | S🡪B | A🡪C | A🡪D | A->E | B🡪A | B🡪C | B🡪D | B🡪E | C🡪D | C🡪E | D🡪E | D🡪T | E🡪T |
| Flow | 0 | 0 | 2 | 2 | 0 | 3 | 2 | 0 | 0 | 3 | 5 | 1 | 0 | 1 |
| Edge | A🡪S | B🡪S | C🡪A | D🡪A | E🡪A | A🡪B | C🡪B | D🡪B | E🡪B | D🡪C | E🡪C | E🡪D | T🡪D | T🡪E |
| Flow | 6 | 3 | 0 | 1 | 5 | 0 | 0 | 3 | 5 | 0 | 0 | 0 | 4 | 5 |

Part D [1 pt] : In the final residual graph, which other nodes are reachable from source node S?

None

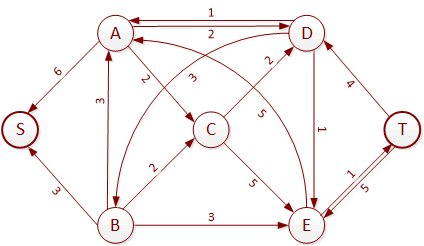
Part E [1 pt]: In the final residual graph, from which other nodes can you reach sink node T?

A,B,C,D,E

Part F [1 pt]: Which edges in the original graph cross the maximum cut formed by the residual graph at the end of the Edmonds-Karp algorithm?

S🡪A and S🡪B

*FYI, here is the residual graph.*



1. [5 pts] Consider the following problem. Using no more space than the next page, describe how you would solve it (describe an algorithm you would invent to do so, which may or may not involve using one or more of the algorithms we’ve used this year as subroutines).

Suppose that you have the problem of serving a large number of users with live streaming content from a small number of servers, in an environment in which the flow to a given consumer will come from exactly one server. At any given time you have some ongoing streams, and you know which consumers have requested new streams.

You want to assign paths through the network so that if possible, you can serve all requests. If this is not possible, you want to serve as many requests as possible.

Note a few things:

* Each user may request different content than each other user. So each stream is independent of each other stream.
* Any server can serve any steam at any requested rate.
* Once a server is selected to give content to a user, that user will continue to get content from that server – they can’t “switch servers in midstream”.
* Different requests may have a different bit rates, taken from a small selection of possibilities (some people may request higher definition than others, for instance).

Explain what algorithm(s) you would use or develop to solve this problem, and how you would do them.

*It would be nice if this were typed up for legibility.*

***Solution:***

First, the graph would have edge capacities that equal the maximum bandwidth of that network link. Then I would create a supersource with links (edges) of effectively infinite capacity to each original source. I would not create a supersink, though, since the choice of sink (user) for each request is forced.

When the first request came in, I would use the Edmonds-Karp algorithm to find the shortest path (in terms of number of links) from the supersource to the destination that has sufficient capacity to serve the request. I would assign it, and reduce the remaining capacity on the links used accordingly. Then I would do the same for each subsequent request. When a user terminated, I would release the capacity along the links.