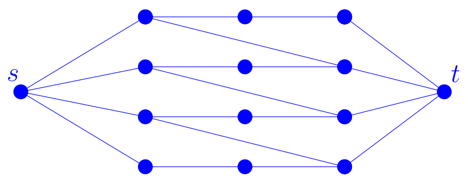
Cs445 — Homework #6
Due: 12/**9**/2015 midnight

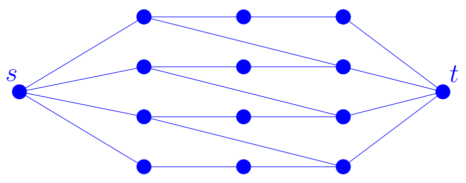
Instructions.

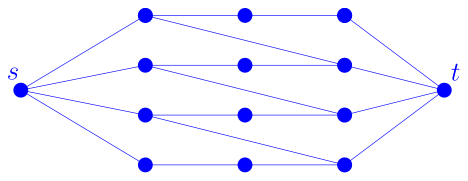
1. Solution could **not** be submitted by students in pairs.
2. You could submit a pdf of the homework, either printed or hand-written and scanned, as long as it is **easily** readable.
3. If your solution is not clearly written, it might not be graded.
4. Prove the correctness of your answer. A correct answer without a proof might not be accepted.
5. If you have discussed the solution with other students, mention their names clearly on the homework. These discussions are not forbidden and are actually **encouraged**. However, you must write your whole solution yourself.
6. All questions have same weight.
7. You could refer to a data structure studied in class, and just mention briefly their guaranties. For example "*It is known that a Red-Black tree could support the insert/delete/find operations on a set of n elements in time $O(\log n)$.*
8. If your answer uses one of the data structures or algorithms that were studied in class, you could refer to it without having to repeat details studied in class. If you answer requires only modifications of one of the algorithm, it is enough to mention the required modifications, and what's the effect (if any) on the running time and on other operations that the algorithm performs.
9. In general, a complete answer should contain the following parts:
   (a) High level description of the data structures (if needed). *E.g. We use a binary balanced search tree. Each node contains, a key and pointers to its children. We augment the tree so each node also contains a field...*
   (b) High level description of the algorithms
   (c) Proof of correctness (why your algorithm provides what is required).
   (d) A claim about the running time, and a proof showing this claim.

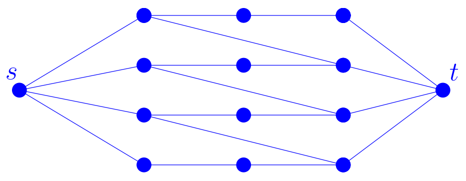
1. Run Edmonds-Karp Algorithm on the given network. Assume the capacity of all edges is 1, and directions of all edges are left to right. As the algorithm progresses, specify flow, residual capacity, and direction of edges in G_f. Directions of all edges are left to right.

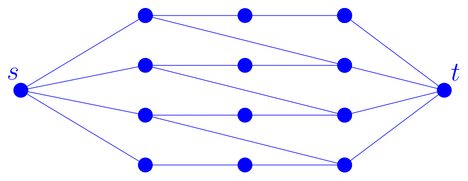
   For your convenience, the graph appears several times, so you could use these copies for your illustrations. Show the residual network after each stage, especially the residual capacities and directions of edges.

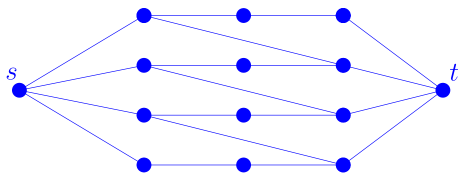


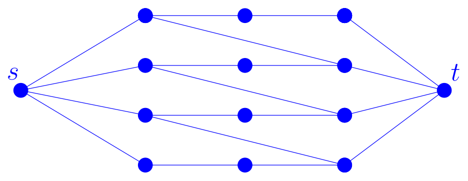


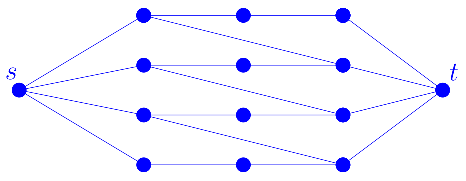


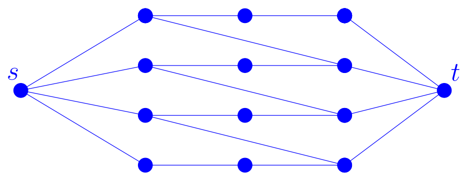


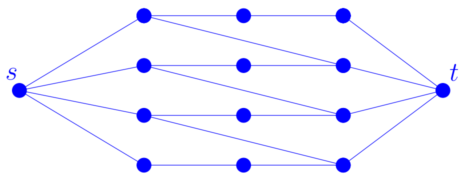


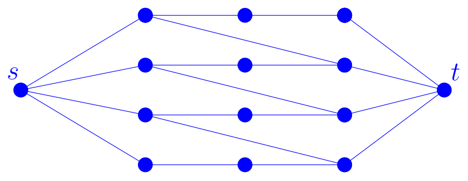


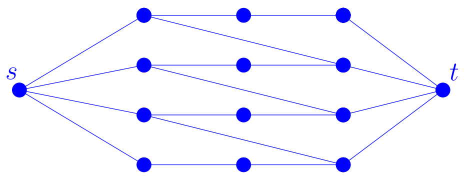












2. Let $G(V, E)$ be a directed graph, with arbitrary (positive or negative) weights given for its edges. With every edge $e \in E$ you are given its weight $w(e)$. Let $s \in V$. Assume that for every $v \in V$ the shortest path $s \to v$ contains at most 5 edges.

   Suggest an $O(m)$-time algorithm that computes $\delta(s, v)$ for every $v \in V$.

   You could assume that G contains no negative cycles. Prove the correctness of your algorithm, and the bound on its running time.

3. The output of Floyd-Warshall algorithm is a matrix D of the form $D[1..n, 1..n]$ such that $D[i, j]$, contains the length of the shortest path from v_i to v_j.

   Modify the algorithm such that after the algorithm is executed, then for every pair v_i, v_j, you can find the path $v_i \to v_j$ in time $O(k)$, where k is the number of edges along this path. Explain exactly how this path is computed.

4. Suggest a reasonable strategy for a navigator program, which preprocesses the road map of USA, and after the preprocessing, any query could be answered efficiently. A query consists of a source address and a target address, (s, t), and the answer to the query is the shortest path from s to t. Try to find solutions that could be fully stored on your cellphone.

   Note that the number n of residential addresses in USA is $133M$ (Google), hence storing a matrix of $\Theta(n^2)$ cells, that is computed in $\Theta(n^3)$ time is not desired.

5. You are given a set B of n points in the plane, and a set R of n points in the plane. Each point is given by its coordinates. Suggest an $O(n^3)$ time algorithm for determining if there is a one to one matching between them, where each red point is matched to a blue point whose distance is at most 1 unit away.

6. Given a network flow $G(V, E)$ with source s and sink t, explain how to find a cut (S, T) such that $c(S, T) \leq c(S', T')$ for any other cut (S', T'). The running time of the algorithm is $O(nm^2)$, where $n = |V|$ and $m = |E|$.

7. Run the LCS algorithm for the strings $X = “ABCA''$ and $Y = “AABBC''$. Show the table (array) used by the algorithm in its final stage.

8. Run the edit distance algorithm for the string $X = “ABBB''$ and $Y = “BBBC''$. Assume the cost of each operation is 1. Show the table (array) used by the algorithm, including the values it stores, in its final stage of the algorithm.

9. Assume each character c in the alphabet has a weight $W[c]$. Suggest an algorithm that, given two words $X[1..m], Y[1..n]$, finds the *Weighted Longest Common Sub-sequence*, or WLCS(X, Y), defined as a common sub-sequence so that the sum of the weights of its elements is as large as possible. The running time should be $O(mn)$.

   Example 1. If the weights of all characters are the same, then your algorithm should return the LCS(X, Y).

   Example 2. Assume $W[A] = 1, W[B] = 1, W[C] = 20$. Then
   WLCS($“ABABABC''$, $“ACABAB''$)=“AC”, and its weighted length is $W[A] + W[C] = 21$. Note that LCS($“ABABABC''$, $“ACABAB''$)=“$ABAB$”