Algorithmics II (H) Exam April / May 2025 – Solutions

1. **(a)** **(i)** **onOppositeSides(p,q,-r-s-)=true** as p and q lie on the line through r and s.
onOppositeSides(r,s,-p-q-)=true as r and s lie on the line through p and q.
boundingBox(p-q,r-s)=true as the bounding boxes of the two line segments intersect.

(ii) **onOppositeSides(p,q,-r-s-)=false** as p and q do not lie on opposite sides of the line through r and s.
onOppositeSides(r,s,-p-q-)=true as r and s lie on opposite sides of the line through p and q.
boundingBox(p-q,r-s)=false as the bounding boxes of the two line segments do not intersect.

(b) First, find the distance d_R between a closest pair of red points in R in O(n log n) time using the closest pair algorithm applied to R. Repeat for the blue points to obtain d_B, the distance between a closest pair of blue points in B in O(n log n) time , and then return the minimum of d_R and d_B.

(c) Sort the points in S on x-coordinate, breaking ties on y-coordinate. Now consider every pair of points P and Q in S. Suppose firstly that $P.x=Q.x$, and without loss of generality suppose that $P.y>Q.y$. Let $r=P.y\text{-}Q.y$ and let $s=\sqrt{3}r/2$. Using binary search, determine either there is a point R with coordinates $(P.x+s,\ Q.y+r/2)$ or $(P.x\text{-}s,\ Q.y+r/2)$. A similar approach can be used if $P.y=Q.y$.

2. **(a)**

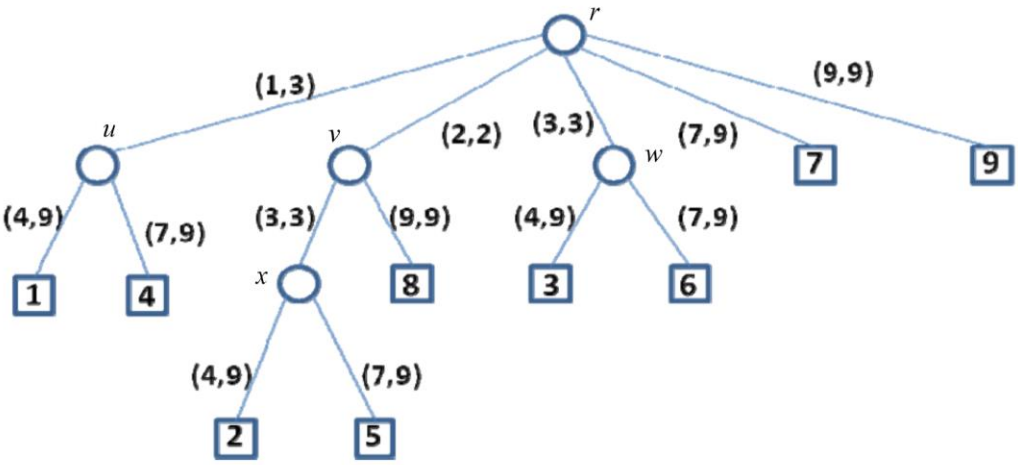
$$D_0=\begin{pmatrix} 0 & 2 & 4 & 5 \\ \infty & 0 & \infty & 2 \\ \infty & -3 & 0 & 3 \\ \infty & \infty & \infty & 0 \end{pmatrix}=D_1 \quad D_2=\begin{pmatrix} 0 & 2 & 4 & 4 \\ \infty & 0 & \infty & 2 \\ \infty & -3 & 0 & -1 \\ \infty & \infty & \infty & 0 \end{pmatrix} \quad D_3=\begin{pmatrix} 0 & 1 & 4 & 3 \\ \infty & 0 & \infty & 2 \\ \infty & -3 & 0 & -1 \\ \infty & \infty & \infty & 0 \end{pmatrix}=D_4$$

The length of a shortest path from v_i to v_j is given by $D_4(i,j)$.

(b) Create a weighted directed graph $G=(V,E)$ where $V=X$ and $E=R$, and where every edge has weight 1. Run the Floyd-Warshall algorithm on G to compute D^*. Then $R^*(x,y)$ if and only if $D^*(x,y)$ lies between 1 and n.

(c) $D_k(i,j) = \max\ \{\ D_{k\text{-}1}(i,j),\ \min\ \{\ D_{k\text{-}1}(i,k),\ D_{k\text{-}1}(k,j)\ \}\ \}.$

3. **(a)** The tree can be represented as follows: $r(1,3)u$, $r(2,2)v$, $r(3,3)w$, $r(7,9)7$, $r(9,9)9$, $u(4,9)1$, $u(7,9)4$, $v(3,3)x$, $v(9,9)8$, $w(4,9)3$, $w(7,9)6$, $x(4,9)2$, $x(7,9)5$. The tree is illustrated below (though this illustration was not required as part of the solution):



(b) Build the suffix tree T for X. Using a traversal, find a branch node of T with greatest string depth that has at least r descendant leaf nodes. The number of descendant leaf nodes of a given node can be computed using a bottom-up traversal of T.

4. **(a)** Let S be the set of vertices returned by A and let $v \in V$. Suppose that $v \notin S$. Then v was removed from X because some w adjacent to v was added to S. It follows that S is a dominating set of G.

(b) Let $c>1$ be a constant and let $k=\lfloor c \rfloor+1$ Construct a graph G with vertices $\{u, v_1, v_2, \ldots, v_k\}$ and edges $\{u, v_i\}$ $(1 \leq i \leq k)$. Algorithm A might choose vertex v_1 first, terminating with the dominating set $S=\{v_1, v_2, \ldots, v_k\}$ of size k. However $D=\{u\}$ is a minimum dominating set in G, of size 1. Hence $|S| = k > c = c.|D|$, so that A is not a c-approximation algorithm for MDS.

(c) Let $c>1$ be a constant and let $k=\lfloor c \rfloor+1$ Construct a graph G with the following vertices:

$$\{u, v_1, \ldots, v_k, w_1, \ldots, w_k, x\}$$

and the following edges:

$$\{u, v_i\}\ (1 \leq i \leq k),\ \{v_i, w_i\}\ (1 \leq i \leq k) \text{ and } \{w_i, x\}\ (1 \leq i \leq k).$$

[The following was not required as part of the solution: Algorithm B might choose edges $\{v_i, w_i\}$ $(1 \leq i \leq k)$, leading to a vertex cover of size $2k$. However $D=\{u, x\}$ is a minimum dominating set in G, of size 2. Hence $|S| = 2k > 2c = c.|D|$, so that B is not a c-approximation algorithm for MDS.]