COMPUTATIONAL GEOMETRY
Assignment 1

Aman Choudhary
MTech Coursework, CSA 2020
Sr No: 17920

March 10, 2021

1 Problem 1

S1 : $CH(P)$ is the intersection of all convex objects containing P.
S2 : $CH(P)$ is the smallest convex object containing P.

We would prove the equivalence of **S1** and **S2** in two steps:

Part 1: $S1 \implies S2$
Let us assume that $S2$ is false, i.e. $CH(P)$ is not the smallest convex object that encapsulates P. Then, there would exist another convex object $C' \subset CH(P)$ which would be smaller than $CH(P)$ and contain P. However, the intersection of C' and $CH(P)$ would be $C' \neq CH(P)$. This implies that $S1$ is false. Hence, we have proved that $\neg\ S2 \implies \neg\ S1$. The proof of Part 1, thus follows from its contrapositive.

Part 2: $S2 \implies S1$
Let us assume that $S1$ is false, i.e. $CH(P)$ is not the intersection of all convex objects containing P. This means that there exists $I' \neq CH(P)$, which in fact is the intersection of all convex objects containing P. We note that $I' \subset CH(P)$, since I' is the result of intersection of all convex sets containing P, which also includes $CH(P)$. Now, I' must also be a convex set since it is the result of intersection of a collection of convex sets (prove of this result is stated as part of 2(a)). However, this leads us to the conclusion that I' is a smaller convex object containing P, which leads to falsification of $S2$. Hence, we have proved $\neg\ S1 \implies \neg\ S2$. The proof of Part 2, thus follows from its contrapositive.

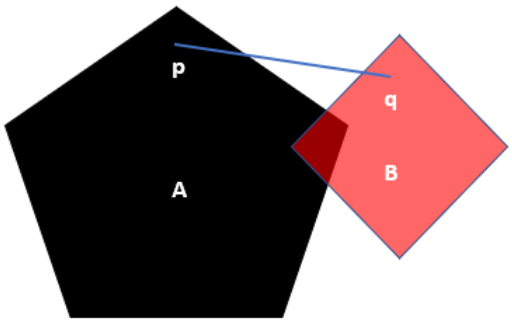
2 Problem 2

2.1 (a)

If $A \cap B = \phi$, then we have a trivially convex set. Otherwise, let us consider any two points p and q in $A \cap B$. These points are part of both A as well as B. Now, we can say that the line joining the two points, say PQ would lie completely in A, courtesy to the fact that A is a convex object. Using, the same argument, this line would also lie entirely inside B. Hence, we conclude that for any two arbitrary points p and q in $A \cap B$, the line segment PQ joining the two, lies in $A \cap B$. Therefore, $A \cap B$ is a convex object. We can extend this idea similarly to infinite number of convex objects.

2.2 (b)

$A \cup B$ is not a convex set, and this fact is exemplified by the following counter examples. A segment of the line joining p and q lies outside $A \cup B$.



3 Problem 3

3.1 (a)

The **clockwise turn** test for three points p, q and r checks whether the direction of the cross product of vectors PQ and QR is in the negative Z-axis, assuming that the operand vectors are in the XY-plane. This follows from the **right-hand rule** for cross-product.

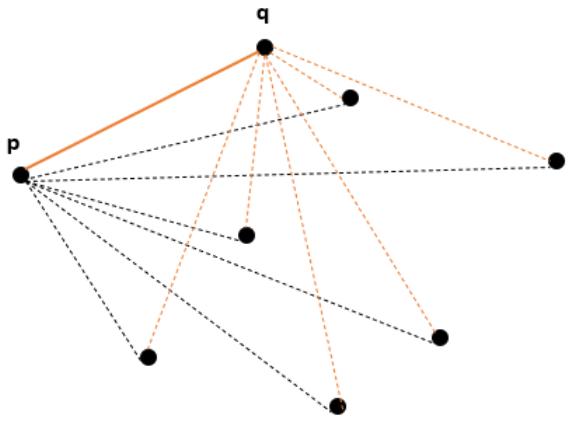
$PQ = (q_x - p_x)\,\hat{i} + (q_y - p_y)\,\hat{j}$
$QR = (r_x - q_x)\,\hat{i} + (r_y - q_y)\,\hat{j}$

$$PQ \times QR = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ (q_x - p_x) & (q_y - p_y) & 0 \\ (r_x - q_x) & (r_y - q_y) & 0 \end{vmatrix} = (q_x - p_x)(r_y - q_y) - (q_y - p_y)(r_x - q_x)\,\hat{k}$$

CLOCKWISE(p, q, r)
$(q_x - p_x)(r_y - q_y) - (q_y - p_y)(r_x - q_x) < 0$

3.2 (b)

- Let us consider the below figure, with a set of points P. We denote the point with the left-most x-coordinate by p. We would like to find q, the point next to p, which is a part of $CH(P)$.
- An important point to note is that, for any point r, other than p and q, the test $Clockwise(p, q, r)$ is satisfied. We can find q by its unique property that it acts as the clockwise turn point for every such r.
- Also, for every such r, there exists at least one such point, i, such that $Clockwise(p, i, r)$ holds true. However, the same is not true about q.



JARVIS_MARCH(P)

1. $l =$ **FIND_LEFT(P)**
2. $p = l,\ q = NULL$
3. while $q\ != \ l$
4. $\quad q = p + 1$
5. $\quad q =$ **FIND_NEXT (p, q)**
6. $\quad p.next = q,\ q.prev = p$
7. $\quad p = q$

FIND_LEFT (P)

1. $l = 1$
2. for $i = 2$ to n
3. $\quad$ if $i.x < l.x$
4. $\quad\quad l = i$
5. return l

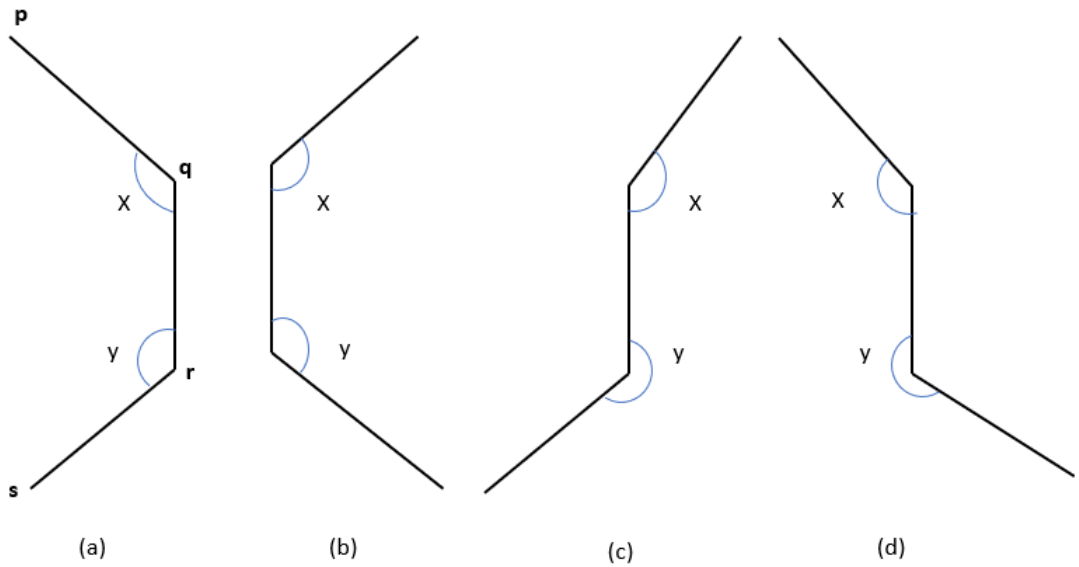
FIND_NEXT (p, q)

1. for $i = 1$ to n
2. if $(i\ != p$ AND $i\ != q)$
3. $\quad$ if $(CLOCKWISE(p, i, q) == TRUE)$
4. $\quad\quad q = i$
5. return q

4 Problem 4

Claim: We argue that a polygon is convex if every possible sequence of three consecutive vertices of the polygon, has identical kind of turn i.e. either all clockwise or all are anti-clockwise.

Proof: Let us consider a sequence of three vertices p, q, r on the polygon Q. The next sequence would be q, r, s. We have two turns for these points, one at q and the next at r. In total, we have a total of 4 possibilities for these two turns as shown in figure. As long as we keep encountering the same kind of turn, then only non-reflex angles are formed, see figure (a) and (b). Whereas, in figure (c) and (d), we observe that as soon as we see a different turn than expected, reflex angles get created, which destroy the convexity of the polygon. Using this idea, for every pair of consecutive turns, we conclude that, for a polygon to be convex, each of its turn at the vertices must have identical nature i.e. either all clockwise or all anti-clockwise.



IS_CONVEX(Q)

1. $p = Q[1],\ q = Q[2],\ r = Q[3]$
2. if $(CLOCKWISE(p, q, r) == TRUE)$
3. $\quad dir = 1$
4. else
5. $\quad dir = -1$
6. for $i\ =\ 2$ to n
7. $\quad p = Q[i]$
8. $\quad q = Q[(i + 1) mod\ n]$
9. $\quad r = Q[(i + 2) mod\ n]$
10. $\quad$ if $((dir = 1$ AND $ANTICLOCKWISE(p, q, r) == TRUE)$
    OR $(dir = -1$ AND $CLOCKWISE(p, q, r) == TRUE))$
11. $\quad\quad$ return $FALSE$
12. return $TRUE$

Since, the evaluation of turn for clockwise or anti-clockwise nature takes constant time, and we do it for each of the n vertices, the overall complexity of the algorithm is $O(n)$.

5 Problem 6

We can use **Jarvis March** to compute the convex layers of a point set P. After every iteration the algorithm returns the convex hull of the set of points. After computing the hull, we remove the points and apply Jarvis March on it. We repeat this till the set becomes empty. Let us assume that it takes us k iterations to do so.

In 1^{st} iteration, we have n points with h_1 points in the output set.
In 2^{nd} iteration, we have $n - h_1$ points with h_2 points in the output set.
In 3^{rd} iteration, we have $n - h_1 - h_2$ points with h_3 points in the output set.
In k^{th} and last iteration we have $n - h_1 - h_2 - ... - h_{k-1}$ points with h_k points int the output set.

Each of the points in P gets removed in exactly one of the layers. So, $h_1 + h_2 + ... + h_k = n$. Also, we know that for an input set of n points and output set of h points, Jarvis March takes $O(nh)$ time to compute the convex hull.

Total Time Complexity = $nh_1 + (n - h_1)h_2 + (n - h_1 - h_2)h_3 + ... + (n - h_1 - h_2 - ... h_{k-1})h_k$
$= (nh_1 + nh_2 + nh_3 + ... + nh_k) - h_1h_2 - h_1h_3 - h_2h_3 - ... - h_1h_k - h_2h_k - ... - h_{k-1}h_k$
$= n(h_1 + h_2 + ... + h_k) - (h_1h_2 + h_1h_3 + h_2h_3 + ... + h_1h_k + h_2h_k + ... + h_{k-1}h_k)$
$= n^2 - (h_1h_2 + h_1h_3 + h_2h_3 + ... + h_1h_k + h_2h_k + ... + h_{k-1}h_k)$
$\geq n^2 = O(n^2)$

6 Problem 7

We know that the time complexty of Chan's complexity is $O(n \log h)$.

Case 1: When we guess $h = 3^{2^i}$

We make a series of guesses and follow trial and error approach until our guess just exceeds h. Thus, time spent by the algorithm is the sum of individual attempts:

$n \log 3^2 + n \log 3^4 + ... + n \log 3^{2^i} + ... + n \log 3^{2^k}$

Let k be the maximum number of iterations needed. Then, it must be true that, $3^{2^k} \geq h$. This implies that $k \geq \log_2 \log_3 h$. Also, k will not exceed that quantity. Hence, $k = \theta(\log \log h)$

Time Complexity $= \sum_{i=1}^{k} n \log 3^{2^i} = n \log 3 \sum_{i=1}^{k} 2^i = n \log 3 \,.\, 2(2^k - 1) = n \log 3 \,.\, 2(\log_3 h - 1) = O(n \log h)$

Case 2: When we guess $h = 3^i$

We make a series of guesses and follow trial and error approach until our guess just exceeds h. Thus, time spent by the algorithm is the sum of individual attempts:

$n \log 3^1 + n \log 3^2 + ... + n \log 3^i + ... + n \log 3^k$

Let k be the maximum number of iterations needed. Then, it must be true that, $3^k \geq h$. This implies that $k \geq \log_3 h$. Also, k will not exceed that quantity. Hence, $k = \theta(\log h)$

Time Complexity $= \sum_{i=1}^{k} n \log 3^i = n \log 3 \sum_{i=1}^{k} i = n \log 3 \,.\, \frac{k\,(k+1)}{2} = n \log 3 \,.\, \frac{\log_3 h\,(\log_3 h + 1)}{2} = O(n(\log h)^2)$

Thus, guessing in steps of 3^i, would slow down our algorithm.
