Assignment 2 - Triangulation and Linear Programming

Duy Pham - 0980384
Mazen Aly - 0978251
Pattarawat Chormai - 0978675

November 20, 2015

1

- Given a simple polygon P a with n vertices.
- Perform triangulation on P.
- Construct the dual graph G of P.
- Find a root node R of G whose degree is 1.
- Perform $LabelNumberOfChildNodes(G, R)$.
- Select the node v_i whose label is n where $n = max(label_i, 0 < i < n \text{ and } label_i \leq \lfloor 2n/3 \rfloor)$
- Find the diagonal line corresponding to the edge between v_i and its parent.

Correctness

The first case is when we can pick the node with the exact label, that is $\lfloor 2n/3 \rfloor$. In this case, one polygon that the algorithm returns has exactly $\lfloor 2n/3 \rfloor + 2$ vertices. The other polygon has exactly $\lfloor n/3 \rfloor + 2$ vertices. Hence the algorithm is correct.

The second case is when we cannot find the exact label, so we have to find the closest node v^*, whose label is the maximum one that is less than $\lfloor 2n/3 \rfloor$. In this case, there is 2 branches starting from the parent of v^*. Hence, the label of the parent of v^* is the sum of the labels of its 2 children, and it is greater than $\lfloor 2n/3 \rfloor$. Therefore, if v^* is the maximum value between the 2 children, then the label of v^* is greater than $\lfloor n/3 \rfloor$. That is,

$$n/3 \leq label(v^*) < 2n/3$$

Algorithm 1 LabelNumberOfChildNodes

Require: a dual graph G and a node v_i

Label v_i as $Visited$
if $degree(v_i) > 1$ **then**
  $NumNodes = 0$
  for Each neighbor v_j of v_i **do**
    if v_j is not $Visited$ **then**
      $NumNodes = NumNodes + LabelNumberOfChildNodes(G, v_j)$
    end if
  end for
  $label_i = 1 + NumNodes$
  Return $label_i$
else
  $label_i = 1$
  Return $label_i$
end if

So if we cut by the edge between v^* and its parent, the neither of the 2 polygons has more than $\lfloor 2n/3 \rfloor + 2$ vertices. Then the algorithm is correct.

Running Time

- Performing triangulation on P takes $O(n \log n)$
- Constructing the dual graph G of P takes $O(n)$
- Finding a root node R of G whose degree is 1 takes $O(n)$
- Performing $LabelNumberOfChildNodes(G, R)$ takes $O(n)$ because we traverse each node only once.
- Selecting the appropriate node v_i take $O(n)$.
- Finding the diagonal line corresponding to the edge between v_i and its parent takes constant time.

Thus, the algorithm performs in $O(n \log n)$