Homework 6

Robbie McKinstry, Jack McQuown, Cyrus Ramavarapu

15 September 2016

Problem 5:

Problem 6:

Let $\mathcal{P}$ be an n sided convex polygon with vertex set $\mathcal{V}$. Pick a vertex $v \in \mathcal{V}$ and label it 1. In a clockwise manner starting from the vertex labeled 1, continue labeling the vertices $2, 3, \ldots, n$.

The minimum permimeter $n - 2$ triangulation of $\mathcal{P}$ can be define recursively as follows.

```
Function: N_2Trig
Globals: Polygon 𝒫
Input: Polygon P
Define: 𝒱 := Numbered vertices of P
if |𝒱| = 3 then
    return Perimeter(P)
else
    Define: P[ ]
    for i ← 1 to |𝒱| − 1 do
        /* [a,b] means the line from vertex a to vertex b  */
        Define: e := [i, i + 1]
        foreach v not i,i + 1 do
            /* Polygon(a,b) constructs polygon with edge a,b
               embedded on 𝒫                                */
            /* Polygon(x,y,z) constructs a polygon with
               vertices x,y,z                               */
            Define: P_a := Polygon(i, v)
            Define: P_b := Polygon(i + 1, v)
            Define: P_3 := Polygon(i, i + 1, v)
            P.append(N2Trig(P1) + N2Trig(P2) + N2Trig(P3) −
            dist(i, v) − dist(i + 1, v))
        end
    end
    return min(P)
end
```

Although this algorithm produces a minimum perimeter $n-2$ triangulation of polygon $\mathcal{P}$, it runs in exponential time due to the recursive calls placed within 2 nested loops.

The triangulation process can be improved significantly by moving to an array based as follows.

Function: Dyn_N_2Trig
Globals: $Polygon\ \mathcal{P}$

```
/* Initialization                                                  */
for i ← 1 to n do
    for j ← 1 to n do
        if i − j < 2 then
            A[i][j] = 0
        end
    end
    if i − j = 2 then
        A[i][j] = Perimeter(i, j)
    end
end
/* calculation                                                     */
Define: P[ ]
for i ← 1 to |V| − 1 do
    /* [a,b] means the line from vertex a to vertex b              */
    Define: e := [i, i + 1]
    for j ← |v| to 1 do
        Define: P_1 := Polygon(i, j)
        Define: P_2 := Polygon(i + 1, j)
        Define: P_3 := Polygon(i, i + 1, j)
        P.append(A(P1) + A(P2) + A(P3) − dist(i, j) − dist(i + 1, j))
    end
    A[Polygon(i, j + 1)] = min(P)
end
```

Problem 7: