# Aula 8

## Doubly-connected edge list - DCEL

Sec 2.2 do livro de de Berg e outros

# Representação de mapas retilineares

Como acoplar o algoritmo de triangulação de polígonos monótonos ao algoritmo de Lee e Preparata?

# Representação de mapas retilineares

Como acoplar o algoritmo de triangulação de polígonos monótonos ao algoritmo de Lee e Preparata?

Doubly-connected edge list – DCEL

Cada aresta aparece duas vezes nesta ED: uma vez em cada direção, uma cópia apontando para a outra (*twins* ou *half-edges*).

# Representação de mapas retilineares

Como acoplar o algoritmo de triangulação de polígonos monótonos ao algoritmo de Lee e Preparata?

Doubly-connected edge list – DCEL

Cada aresta aparece duas vezes nesta ED: uma vez em cada direção, uma cópia apontando para a outra (*twins* ou *half-edges*).

Cada meia-aresta está em uma face do mapa, e aponta para a próxima meia-aresta e para a anterior na face.

# Representação de mapas retilineares

Como acoplar o algoritmo de triangulação de polígonos monótonos ao algoritmo de Lee e Preparata?

Doubly-connected edge list – DCEL

Cada aresta aparece duas vezes nesta ED: uma vez em cada direção, uma cópia apontando para a outra (*twins* ou *half-edges*).

Cada meia-aresta está em uma face do mapa, e aponta para a próxima meia-aresta e para a anterior na face.

Pode se manter registro para cada face ou vértice do mapa.

Cada face/vértice tem apontador para uma das (meia-)arestas incidentes a ele.

# Doubly-connected edge list – DCEL

Cada aresta $e = uv$ tem duas entradas na ED: $(u, v)$ e $(v, u)$.

# Doubly-connected edge list – DCEL

Cada aresta $e = uv$ tem duas entradas na ED: $(u, v)$ e $(v, u)$.

Para cada meia-aresta $(u, v)$:
$\texttt{twin}(u, v) = (v, u)$
$\texttt{prox}(u, v)$: próxima meia-aresta na face
$\texttt{prev}(u, v)$: meia-aresta anterior na face

# Doubly-connected edge list – DCEL

Cada aresta $e = uv$ tem duas entradas na ED: $(u, v)$ e $(v, u)$.

Para cada meia-aresta $(u, v)$:
$\texttt{twin}(u, v) = (v, u)$
$\texttt{prox}(u, v)$: próxima meia-aresta na face
$\texttt{prev}(u, v)$: meia-aresta anterior na face

Para cada face $f$, apontador para uma meia-aresta da face.

# Doubly-connected edge list – DCEL

Cada aresta $e = uv$ tem duas entradas na ED: $(u, v)$ e $(v, u)$.

Para cada meia-aresta $(u, v)$:
$\texttt{twin}(u, v) = (v, u)$
$\texttt{prox}(u, v)$: próxima meia-aresta na face
$\texttt{prev}(u, v)$: meia-aresta anterior na face

Para cada face $f$, apontador para uma meia-aresta da face.

Revisite a primeira fase do algoritmo de Lee e Preparata (partição em monótonos) e a adapte para devolver essa representação da partição obtida em vez do conjunto de diagonais apenas.

# Doubly-connected edge list – DCEL

Cada aresta $e = uv$ tem duas entradas na ED: $(u, v)$ e $(v, u)$.

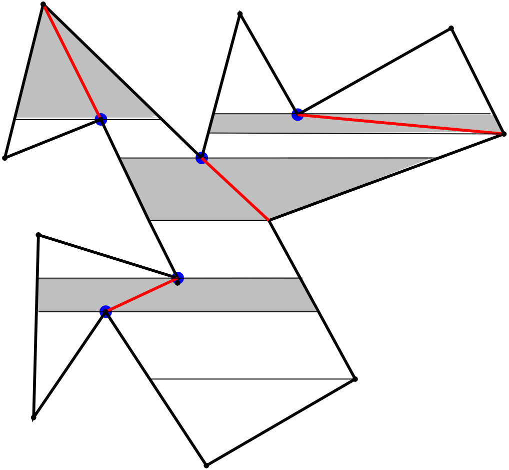
Para cada meia-aresta $(u, v)$:
$\texttt{twin}(u, v) = (v, u)$
$\texttt{prox}(u, v)$: próxima meia-aresta na face
$\texttt{prev}(u, v)$: meia-aresta anterior na face

Para cada face $f$, apontador para uma meia-aresta da face.

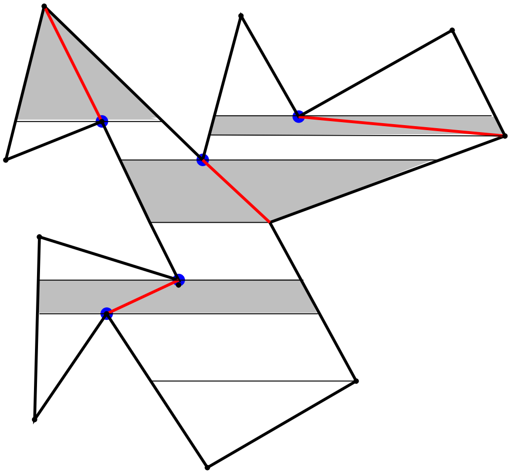
Revisite a primeira fase do algoritmo de Lee e Preparata (partição em monótonos) e a adapte para devolver essa representação da partição obtida em vez do conjunto de diagonais apenas.

Omita a face externa da representação, já que ela não é usada na segunda fase.

# Partição em polígonos monótonos



# Partição em polígonos monótonos



Qual é o grau máximo de um vértice no mapa retilinear final?

# Algoritmo de Lee e Preparata

DivideEmMonótono-LP$(X, Y, n)$

para $k \leftarrow 1$ até $n$ faça<br>
    $E[k] \leftarrow k$<br>
MergeSort$(Y, X, 1, n, E)$    $\triangleright$ ordenação indireta decrescente de $Y$<br>
Crie$(T)$    $t \leftarrow 0$<br>
para $k \leftarrow 1$ até $n$ faça<br>
    $v \leftarrow E[k]$<br>
    $v^- \leftarrow v - 1$    $v^+ \leftarrow v + 1$<br>
    se $v^- = 0$ então $v^- \leftarrow n$<br>
    se $v^+ = n + 1$ então $v^+ \leftarrow 1$<br>
    se $Y[v^-] < Y[v] < Y[v^+]$ ou $Y[v^+] < Y[v] < Y[v^-]$<br>
        então TrataCaso1$(T, v^-, v, v^+, Y, n, D, t)$<br>
        senão se $Y[v^-] < Y[v]$<br>
                então TrataCaso2$(T, v^-, v, v^+, D, t)$<br>
                senão TrataCaso3$(T, v, Y, n, D, t)$<br>
devolva $(D, t)$

# Algoritmo de Lee e Preparata

DivideEmMonótono-LP$(X, Y, n)$

para $k \leftarrow 1$ até $n$ faça
    $E[k] \leftarrow k$
MergeSort$(Y, X, 1, n, E)$ $\triangleright$ ordenação indireta decrescente de $Y$
Crie$(T)$ $\cancel{t \leftarrow 0}$ $\triangleright$ crie DCEL para $P$
para $k \leftarrow 1$ até $n$ faça
    $v \leftarrow E[k]$
    $v^- \leftarrow v - 1$ $\quad v^+ \leftarrow v + 1$
    se $v^- = 0$ então $v^- \leftarrow n$
    se $v^+ = n + 1$ então $v^+ \leftarrow 1$
    se $Y[v^-] < Y[v] < Y[v^+]$ ou $Y[v^+] < Y[v] < Y[v^-]$
        então TrataCaso1$(T, v^-, v, v^+, Y, n, D, t)$
        senão se $Y[v^-] < Y[v]$
                então TrataCaso2$(T, v^-, v, v^+, D, t)$
                senão TrataCaso3$(T, v, Y, n, D, t)$
devolva $(D, t)$

# Caso 1

TrataCaso1($T, u, v, w, Y, n, D, t$)

se $Y[u] < Y[w]$ então $u \leftrightarrow w$

$((i,j), x, (k,l)) \leftarrow$ Remove($T, v$)

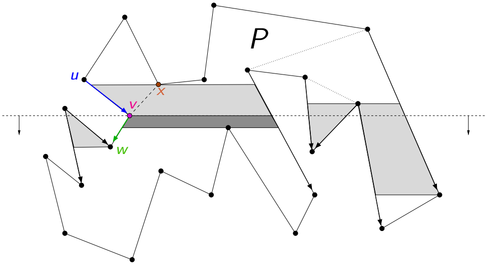
se $v = j$ $\triangleright$ o trapézio está à direita de $v$?

    então Insere($T, (v,w), v, (k,l)$)

    senão Insere($T, (i,j), v, (v,w)$)

se PontaParaBaixo($x, Y, n$)

    então $t \leftarrow t + 1$ $D[t] \leftarrow (x, v)$



# Caso 1

TrataCaso1($T, u, v, w, Y, n, D, t$)

se $Y[u] < Y[w]$ então $u \leftrightarrow w$

$((i, j), x, (k, l)) \leftarrow$ Remove($T, v$)

se $v = j$ $\quad\triangleright$ o trapézio está à direita de $v$?

    então Insere($T, (v, w), v, (k, l)$)

    senão Insere($T, (i, j), v, (v, w)$)

se PontaParaBaixo($x, Y, n$)

    então $t \leftarrow t + 1$ $\quad \cancel{D[t] \leftarrow (x, v)}$ $\quad\triangleright$ adiciona $xv$ à DCEL