



دانشگاه یزد

Yazd Univ.

Computational
Geometry

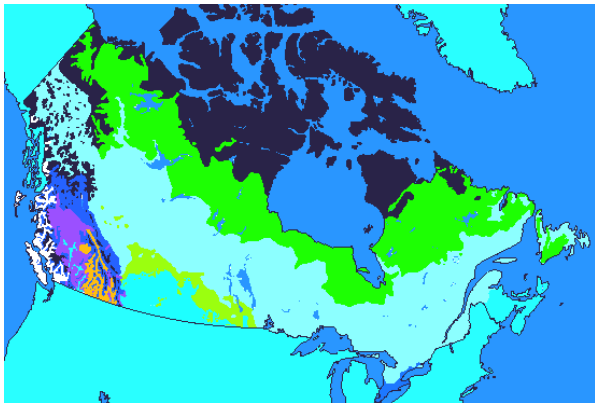
Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

1393-1

- We have solved the easiest case of the map overlay problem, where the two maps are networks represented as collections of line segments.
- In general, maps have a more complicated structure: they are subdivisions of the plane into labeled regions.



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

- Before we can give an algorithm for computing the overlay of two subdivisions, we must develop a suitable representation for a subdivision.
- Storing a subdivision as a collection of line segments is not such a good idea.
- Operations like reporting the boundary of a region would be rather complicated.
- Add topological information: which segments bound a given region, which regions are adjacent, and so on.

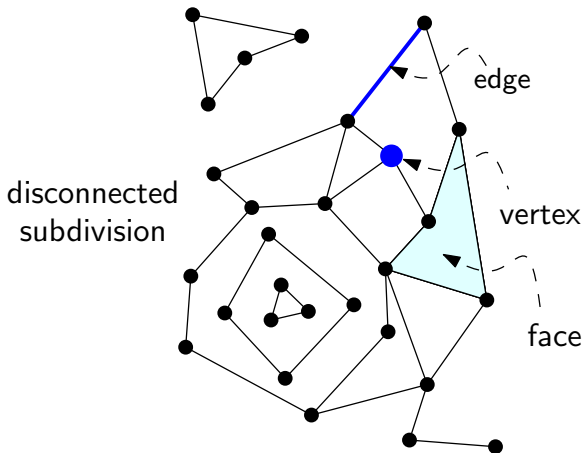


دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



Complexity of a subdivision

$\#faces + \#edges + \#vertices.$



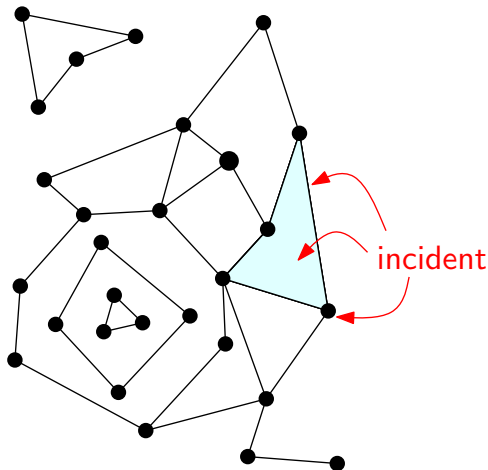
دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



Complexity of a subdivision

$\#faces + \#edges + \#vertices.$

What kind of queries?

- What is the face containing a given point? (TOO MUCH!)
- Walking around the boundary of a given face,
- Find the face from an adjacent one if we are given a common edge,
- Visit all the edges around a given vertex.

The representation that we shall discuss supports these operations. It is called the doubly-connected edge list (DCEL).



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



DCEL contains:

- a record for each edge,
- a record for each vertex,
- a record for each face,
- plus attribute information.



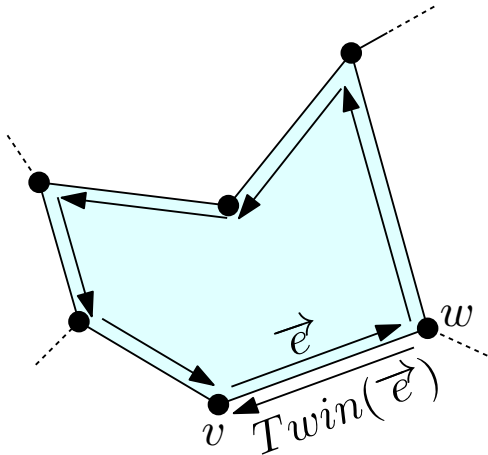
دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



To be able to traverse the boundary of a face, we need to keep a pointer to a half-edge of any boundary component and isolated points.



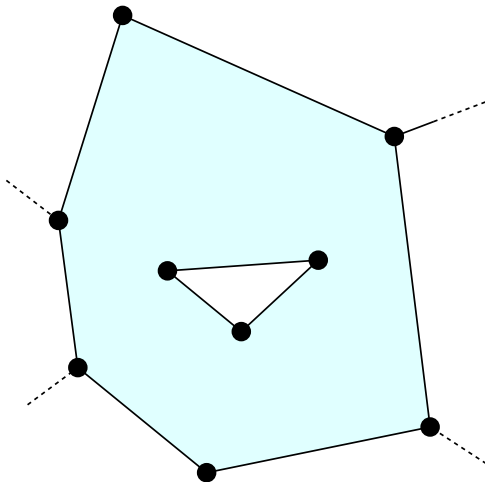
دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



To be able to traverse the boundary of a face, we need to keep a pointer to a **half-edge** of any boundary component and isolated points.



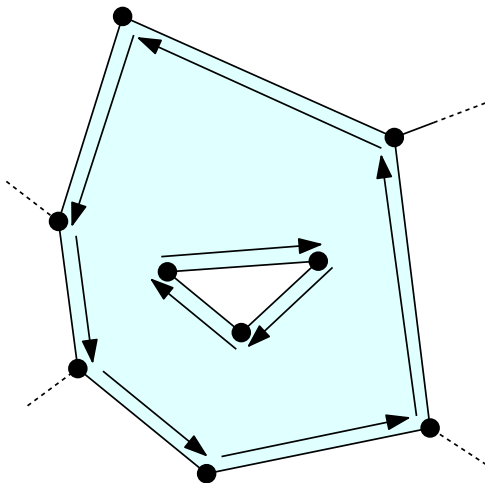
دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



To be able to traverse the boundary of a face, we need to keep a pointer to a **half-edge** of any boundary component and isolated points.



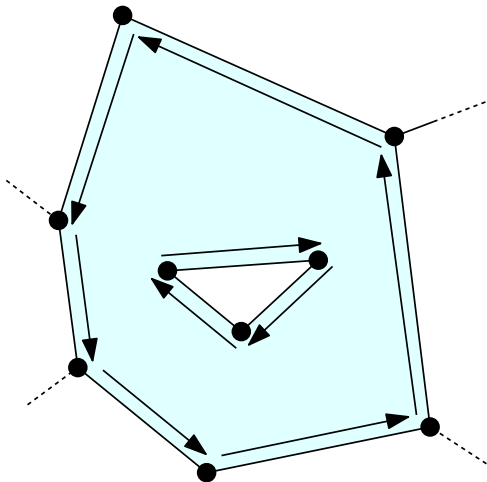
دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



To be able to traverse the boundary of a face, we need to keep a pointer to **a half-edge** of any boundary component and isolated points.

DCEL contains:

- a record for each vertex,
 - ① *Coordinates*(v): the coordinates of v ,
 - ② *IncidentEdge*(v): a pointer to an arbitrary half-edge that has v as its origin.
- a record for each face,
 - ① *OuterComponent*(f): to some half-edge on its outer boundary (nil if unbounded),
 - ② *InnerComponents*(f): a pointer to some half-edge on the boundary of the hole, for each hole.
- a record for each half-edge $\vec{e}$,
 - ① *Origin*($\vec{e}$): a pointer to its origin,
 - ② *Twin*($\vec{e}$) a pointer to its twin half-edge,
 - ③ *IncidentFace*($\vec{e}$): a pointer to the face that it bounds.
 - ④ *Next*($\vec{e}$) and *Prev*($\vec{e}$): a pointer to the next and previous edge on the boundary of *IncidentFace*($\vec{e}$).



دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

DCEL contains:

- a record for each vertex,
 - ① $Coordinates(v)$: the coordinates of v ,
 - ② $IncidentEdge(v)$: a pointer to an arbitrary half-edge that has v as its origin.
- a record for each face,
 - ① $OuterComponent(f)$: to some half-edge on its outer boundary (nil if unbounded),
 - ② $InnerComponents(f)$: a pointer to some half-edge on the boundary of the hole, for each hole.
- a record for each half-edge $\vec{e}$,
 - ① $Origin(\vec{e})$: a pointer to its origin,
 - ② $Twin(\vec{e})$: a pointer to its twin half-edge,
 - ③ $IncidentFace(\vec{e})$: a pointer to the face that it bounds.
 - ④ $Next(\vec{e})$ and $Prev(\vec{e})$: a pointer to the next and previous edge on the boundary of $IncidentFace(\vec{e})$.



دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



DCEL contains:

- a record for each vertex,
 - 1 $Coordinates(v)$: the coordinates of v ,
 - 2 $IncidentEdge(v)$: a pointer to an arbitrary half-edge that has v as its origin.
- a record for each face,
 - 1 $OuterComponent(f)$: to some half-edge on its outer boundary (nil if unbounded),
 - 2 $InnerComponents(f)$: a pointer to some half-edge on the boundary of the hole, for each hole.
- a record for each half-edge $\vec{e}$,
 - 1 $Origin(\vec{e})$: a pointer to its origin,
 - 2 $Twin(\vec{e})$: a pointer to its twin half-edge,
 - 3 $IncidentFace(\vec{e})$: a pointer to the face that it bounds.
 - 4 $Next(\vec{e})$ and $Prev(\vec{e})$: a pointer to the next and previous edge on the boundary of $IncidentFace(\vec{e})$.



DCEL contains:

- a record for each vertex,
 - ① $Coordinates(v)$: the coordinates of v ,
 - ② $IncidentEdge(v)$: a pointer to an arbitrary half-edge that has v as its origin.
- a record for each face,
 - ① $OuterComponent(f)$: to some half-edge on its outer boundary (nil if unbounded),
 - ② $InnerComponents(f)$: a pointer to some half-edge on the boundary of the hole, for each hole.
- a record for each half-edge $\vec{e}$,
 - ① $Origin(\vec{e})$: a pointer to its origin,
 - ② $Twin(\vec{e})$: a pointer to its twin half-edge,
 - ③ $IncidentFace(\vec{e})$: a pointer to the face that it bounds.
 - ④ $Next(\vec{e})$ and $Prev(\vec{e})$: a pointer to the next and previous edge on the boundary of $IncidentFace(\vec{e})$.

DCEL contains:

- a record for each vertex,
 - ① $Coordinates(v)$: the coordinates of v ,
 - ② $IncidentEdge(v)$: a pointer to an arbitrary half-edge that has v as its origin.
- a record for each face,
 - ① $OuterComponent(f)$: to some half-edge on its outer boundary (nil if unbounded),
 - ② $InnerComponents(f)$: a pointer to some half-edge on the boundary of the hole, for each hole.
- a record for each half-edge $\vec{e}$,
 - ① $Origin(\vec{e})$: a pointer to its origin,
 - ② $Twin(\vec{e})$: a pointer to its twin half-edge,
 - ③ $IncidentFace(\vec{e})$: a pointer to the face that it bounds.
 - ④ $Next(\vec{e})$ and $Prev(\vec{e})$: a pointer to the next and previous edge on the boundary of $IncidentFace(\vec{e})$.



دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

DCEL contains:

- a record for each vertex,
 - ① $Coordinates(v)$: the coordinates of v ,
 - ② $IncidentEdge(v)$: a pointer to an arbitrary half-edge that has v as its origin.
- a record for each face,
 - ① $OuterComponent(f)$: to some half-edge on its outer boundary (nil if unbounded),
 - ② $InnerComponents(f)$: a pointer to some half-edge on the boundary of the hole, for each hole.
- a record for each half-edge $\vec{e}$,
 - ① $Origin(\vec{e})$: a pointer to its origin,
 - ② $Twin(\vec{e})$: a pointer to its twin half-edge,
 - ③ $IncidentFace(\vec{e})$: a pointer to the face that it bounds.
 - ④ $Next(\vec{e})$ and $Prev(\vec{e})$: a pointer to the next and previous edge on the boundary of $IncidentFace(\vec{e})$.

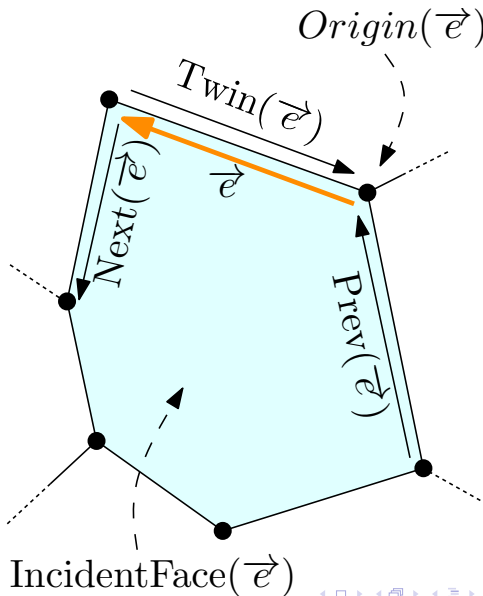


دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



DCEL: Example

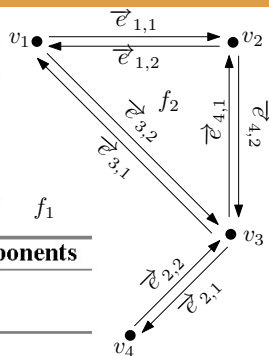


دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



Vertex	Coordinates	IncidentEdge
v_1	(0,4)	$\vec{e}_{1,1}$
v_2	(2,4)	$\vec{e}_{4,2}$
v_3	(2,2)	$\vec{e}_{2,1}$
v_4	(1,1)	$\vec{e}_{2,2}$

Face	OuterComponent	InnerComponents
f_1	nil	$\vec{e}_{1,1}$
f_2	$\vec{e}_{4,1}$	nil

Half-edge	Origin	Twin	IncidentFace	Next	Prev
$\vec{e}_{1,1}$	v_1	$\vec{e}_{1,2}$	f_1	$\vec{e}_{4,2}$	$\vec{e}_{3,1}$
$\vec{e}_{1,2}$	v_2	$\vec{e}_{1,1}$	f_2	$\vec{e}_{3,2}$	$\vec{e}_{4,1}$
$\vec{e}_{2,1}$	v_3	$\vec{e}_{2,2}$	f_1	$\vec{e}_{2,2}$	$\vec{e}_{4,2}$
$\vec{e}_{2,2}$	v_4	$\vec{e}_{2,1}$	f_1	$\vec{e}_{3,1}$	$\vec{e}_{2,1}$
$\vec{e}_{3,1}$	v_3	$\vec{e}_{3,2}$	f_1	$\vec{e}_{1,1}$	$\vec{e}_{2,2}$
$\vec{e}_{3,2}$	v_1	$\vec{e}_{3,1}$	f_2	$\vec{e}_{4,1}$	$\vec{e}_{1,2}$
$\vec{e}_{4,1}$	v_3	$\vec{e}_{4,2}$	f_2	$\vec{e}_{1,2}$	$\vec{e}_{3,2}$
$\vec{e}_{4,2}$	v_2	$\vec{e}_{4,1}$	f_1	$\vec{e}_{2,1}$	$\vec{e}_{1,1}$



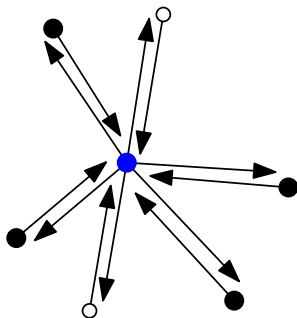
Time complexity of queries?

- Walking around the boundary of a given face,
- Find the face from an adjacent one if we are given a common edge,
- Visit all the edges around a given vertex.



Time complexity of queries?

- Walking around the boundary of a given face,
- Find the face from an adjacent one if we are given a common edge,
- Visit all the edges around a given vertex.



Computing the Overlay of Two Subdivisions

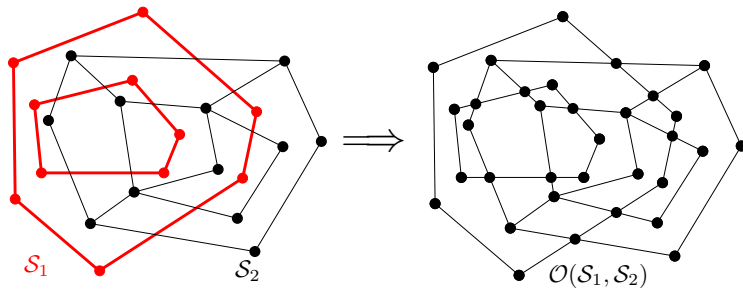


دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions



Main Idea in computing $\mathcal{O}(S_1, S_2)$

- Copy DCEL of S_1 and S_2 into new DCEL (Not a Valid DCEL).
- Make the new DCEL valid.

Computing the Overlay of Two Subdivisions



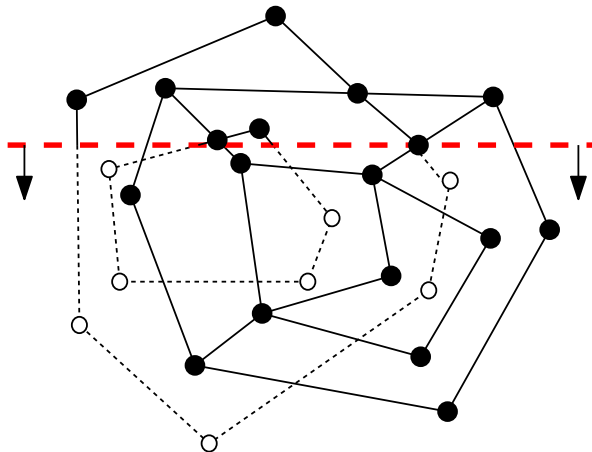
دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

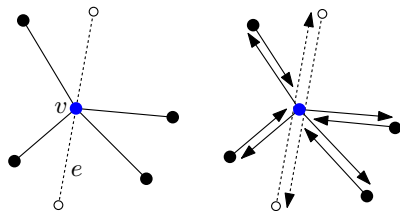
Computing the
Overlay of Two
Subdivisions



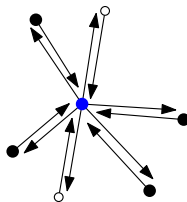
Computing the Overlay of Two Subdivisions

Updating half-edges

the geometric situation and
the two doubly-connected
edge lists before handling the
intersection



the doubly-connected edge
list after handling the inter-
section



دانشگاه یزد

Yazd Univ.

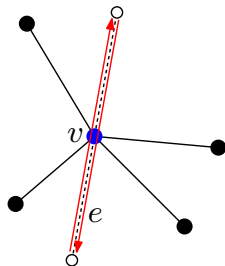
Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating half-edges



- Make two new edge with origin v .
- Set $Twin$ of new edges.
- Set $Next()$ of the two new half-edges.
- Set $Prev()$ of the half-edges to which these pointers point.



دانشگاه یزد

Yazd Univ.

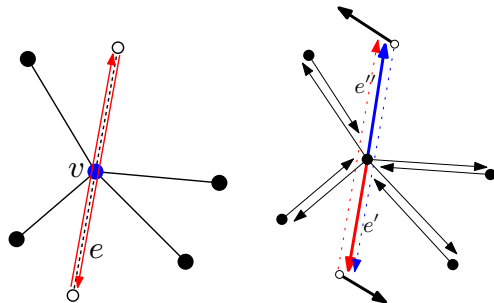
Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating half-edges



- Make two new edge with origin v .
- Set $Twin$ of new edges.
- Set $Next()$ of the two new half-edges.
- Set $Prev()$ of the half-edges to which these pointers point.



دانشگاه یزد
Yazd Univ.

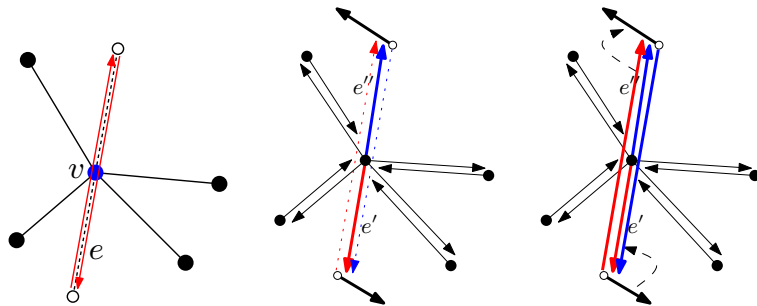
Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating half-edges



- Make two new edge with origin v .
- Set $Twin$ of new edges.
- Set $Next()$ of the two new half-edges.
- Set $Prev()$ of the half-edges to which these pointers point.



دانشگاه یزد
Yazd Univ.

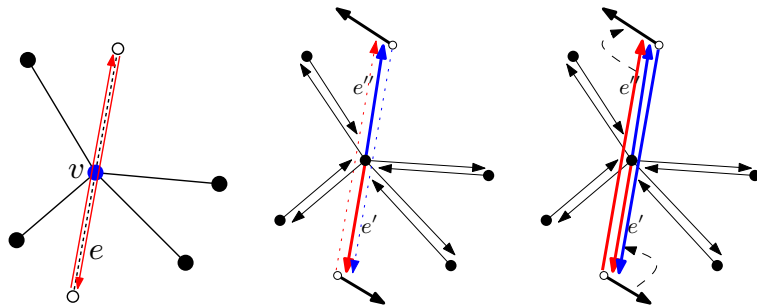
Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating half-edges



- Make two new edge with origin v .
- Set $Twin$ of new edges.
- Set $Next()$ of the two new half-edges.
- Set $Prev()$ of the half-edges to which these pointers point.



دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating half-edges



دانشگاه یزد

Yazd Univ.

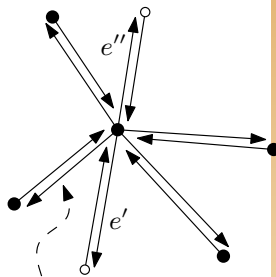
Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Fix the situation around v

- The half-edge for e' that has v as its destination must be linked to the first half-edge, seen clockwise from e' , with v as its origin.
- The half-edge for e' with v as its origin must be linked to the first counterclockwise half-edge with v as its destination.
- The same for e'' .
- Time complexity: $\mathcal{O}(m)$ (m : degree of v).



first clockwise half-
edge from e' with v as
its origin

Computing the Overlay of Two Subdivisions

Updating half-edges



دانشگاه یزد

Yazd Univ.

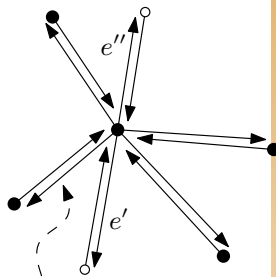
Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Fix the situation around v

- The half-edge for e' that has v as its destination must be linked to the first half-edge, seen clockwise from e' , with v as its origin.
- The half-edge for e' with v as its origin must be linked to the first counterclockwise half-edge with v as its destination.
- The same for e'' .
- Time complexity: $\mathcal{O}(m)$ (m : degree of v).



first clockwise half-
edge from e' with v as
its origin

Computing the Overlay of Two Subdivisions

Updating half-edges



دانشگاه یزد

Yazd Univ.

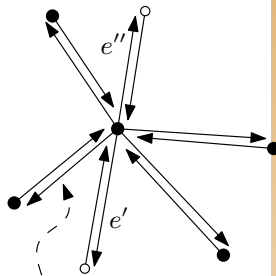
Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Fix the situation around v

- The half-edge for e' that has v as its destination must be linked to the first half-edge, seen clockwise from e' , with v as its origin.
- The half-edge for e' with v as its origin must be linked to the first counterclockwise half-edge with v as its destination.
- The same for e'' .
- Time complexity: $\mathcal{O}(m)$ (m : degree of v).



first clockwise half-
edge from e' with v as
its origin

Computing the Overlay of Two Subdivisions

Updating half-edges



دانشگاه یزد

Yazd Univ.

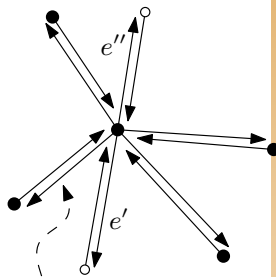
Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Fix the situation around v

- The half-edge for e' that has v as its destination must be linked to the first half-edge, seen clockwise from e' , with v as its origin.
- The half-edge for e' with v as its origin must be linked to the first counterclockwise half-edge with v as its destination.
- The same for e'' .
- Time complexity: $\mathcal{O}(m)$ (m : degree of v).



first clockwise half-
edge from e' with v as
its origin

Computing the Overlay of Two Subdivisions

Updating faces

Updating Faces:

- Create a face record for each $f \in \mathcal{O}(\mathcal{S}_1, \mathcal{S}_2)$.
- Set $OuterComponent(f)$ and $InnerComponent(f)$.



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating faces

- $\# \text{ faces} = \# \text{ outer boundaries} + 1$ (unbounded face).
- From half-edges we can construct the boundaries.
- To determine whether the boundary is outer boundary or boundary of a hole:



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating faces

- $\# \text{ faces} = \# \text{ outer boundaries} + 1$ (unbounded face).
- From half-edges we can construct the boundaries.
- To determine whether the boundary is outer boundary or boundary of a hole:



دانشگاه یزد

Yazd Univ.

Computational
Geometry

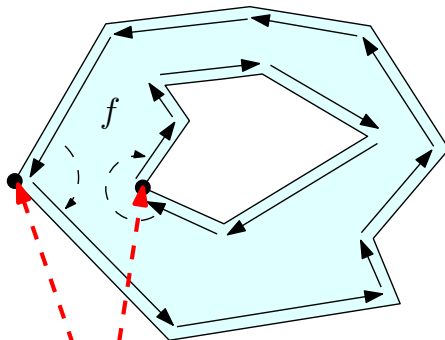
Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating faces

- $\# \text{ faces} = \# \text{ outer boundaries} + 1$ (unbounded face).
- From half-edges we can construct the boundaries.
- To determine whether the boundary is outer boundary or boundary of a hole:



Leftmost vertex of cycle



دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating faces



دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Which boundary cycles bound the same face?

- Construct a graph $\mathcal{G}$.
 - Every boundary cycle is a node in $\mathcal{G}$.
 - One node for the imaginary outer boundary of the unbounded face.
 - Add an arc between two cycles if and only if one of the cycles is the boundary of a hole and the other cycle has a half-edge immediately to the left of the leftmost vertex of that hole cycle.
 - If there is no half-edge to the left of the leftmost vertex of a cycle, then the node representing the cycle is linked to the node of the unbounded face.

Computing the Overlay of Two Subdivisions

Updating faces



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Which boundary cycles bound the same face?

- Construct a graph $\mathcal{G}$.
- Every boundary cycle is a node in $\mathcal{G}$.
- One node for the imaginary outer boundary of the unbounded face.
- Add an arc between two cycles if and only if one of the cycles is the boundary of a hole and the other cycle has a half-edge immediately to the left of the leftmost vertex of that hole cycle.
- If there is no half-edge to the left of the leftmost vertex of a cycle, then the node representing the cycle is linked to the node of the unbounded face.

Computing the Overlay of Two Subdivisions

Updating faces



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Which boundary cycles bound the same face?

- Construct a graph $\mathcal{G}$.
- Every boundary cycle is a node in $\mathcal{G}$.
- One node for the imaginary outer boundary of the unbounded face.
- Add an arc between two cycles if and only if one of the cycles is the boundary of a hole and the other cycle has a half-edge immediately to the left of the leftmost vertex of that hole cycle.
- If there is no half-edge to the left of the leftmost vertex of a cycle, then the node representing the cycle is linked to the node of the unbounded face.

Computing the Overlay of Two Subdivisions

Updating faces



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Which boundary cycles bound the same face?

- Construct a graph $\mathcal{G}$.
- Every boundary cycle is a node in $\mathcal{G}$.
- One node for the imaginary outer boundary of the unbounded face.
- Add an arc between two cycles if and only if one of the cycles is the boundary of a hole and the other cycle has a half-edge immediately to the left of the leftmost vertex of that hole cycle.
- If there is no half-edge to the left of the leftmost vertex of a cycle, then the node representing the cycle is linked to the node of the unbounded face.

Computing the Overlay of Two Subdivisions

Updating faces



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

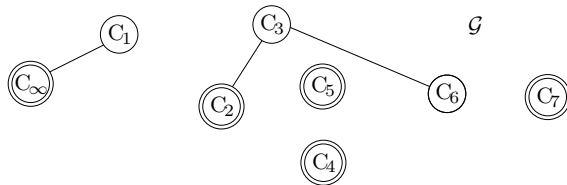
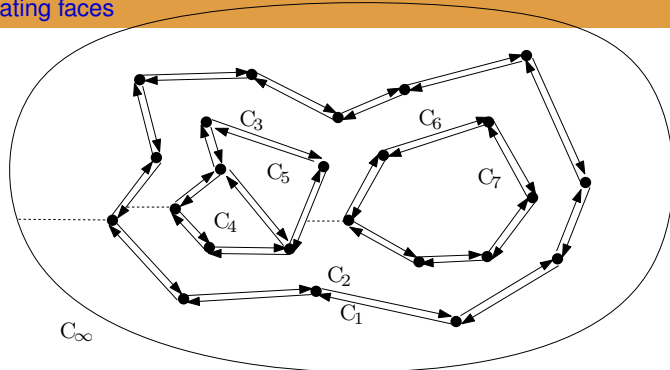
Computing the
Overlay of Two
Subdivisions

Which boundary cycles bound the same face?

- Construct a graph $\mathcal{G}$.
- Every boundary cycle is a node in $\mathcal{G}$.
- One node for the imaginary outer boundary of the unbounded face.
- Add an arc between two cycles if and only if one of the cycles is the boundary of a hole and the other cycle has a half-edge immediately to the left of the leftmost vertex of that hole cycle.
- If there is no half-edge to the left of the leftmost vertex of a cycle, then the node representing the cycle is linked to the node of the unbounded face.

Computing the Overlay of Two Subdivisions

Updating faces



دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Updating faces

Lemma 2.5

Each connected component of the graph $\mathcal{G}$ corresponds exactly to the set of cycles incident to one face.



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

How can we Construct $\mathcal{G}$?



دانشگاه یزد
Yazd Univ.

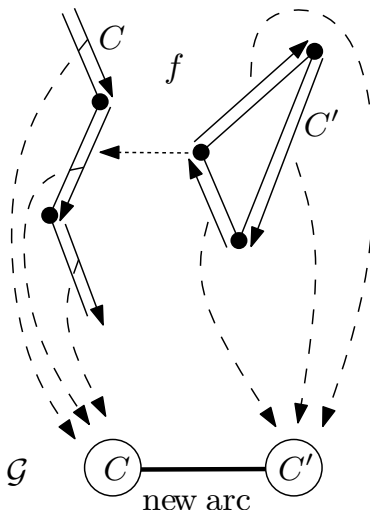
Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Constructing $\mathcal{G}$

- Similar to line segment intersection algorithm.
- Find the edge immediately to the left of point.
- We need a pointer from each edge to its boundary in $\mathcal{G}$.

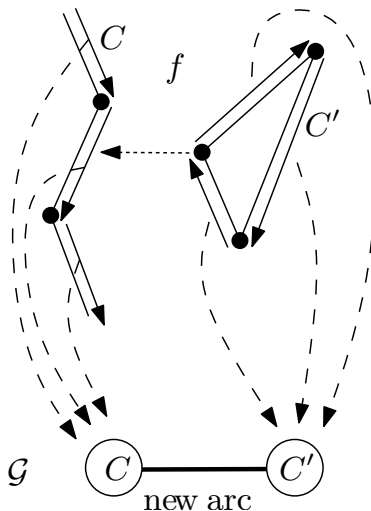


Computing the Overlay of Two Subdivisions

How can we Construct $\mathcal{G}$?

Constructing $\mathcal{G}$

- Similar to line segment intersection algorithm.
- Find the edge immediately to the left of point.
- We need a pointer from each edge to its boundary in $\mathcal{G}$.



دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

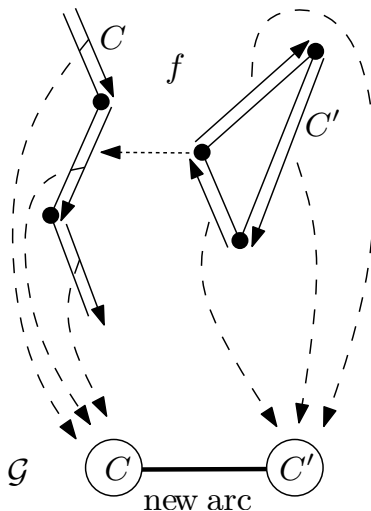
Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

How can we Construct $\mathcal{G}$?

Constructing $\mathcal{G}$

- Similar to line segment intersection algorithm.
- Find the edge immediately to the left of point.
- We need a pointer from each edge to its boundary in $\mathcal{G}$.



دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Computing the Overlay of Two Subdivisions

Theorem 2.6

Let $\mathcal{S}_1$ be a planar subdivision of complexity n_1 , let $\mathcal{S}_2$ be a subdivision of complexity n_2 , and let $n := n_1 + n_2$. The overlay of $\mathcal{S}_1$ and $\mathcal{S}_2$ can be constructed in $\mathcal{O}(n \log n + k \log n)$ time, where k is the complexity of the overlay.



دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions

Application: Boolean Operations

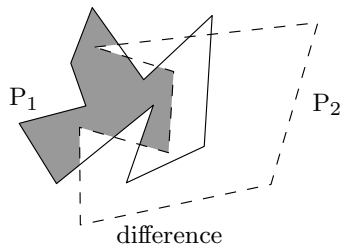
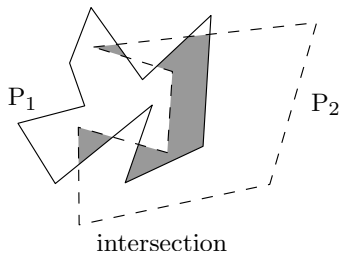
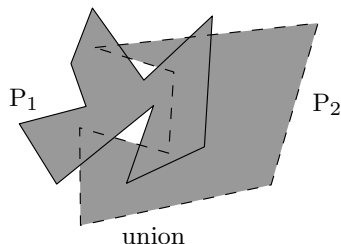


دانشگاه یزد
Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions





دانشگاه یزد

Yazd Univ.

Computational
Geometry

Doubly Connected
Edge List (DCEL)

Computing the
Overlay of Two
Subdivisions