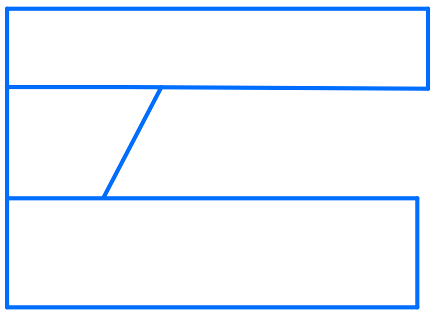
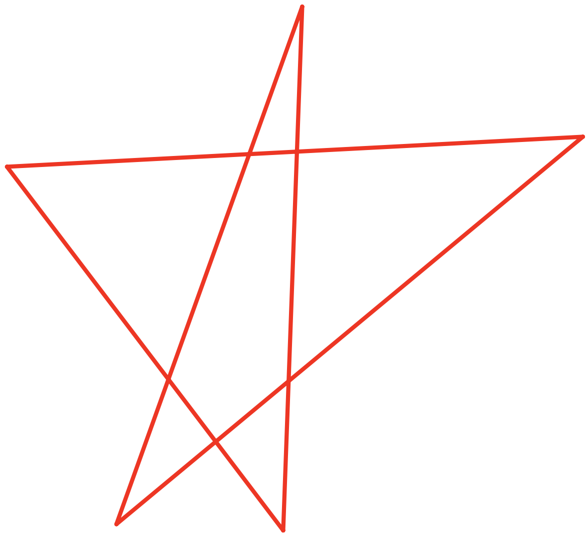
L3 - Map overlay

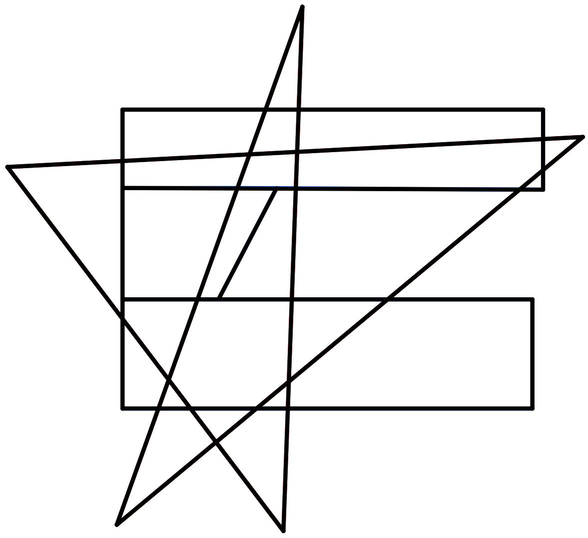
Given blue map



& red map



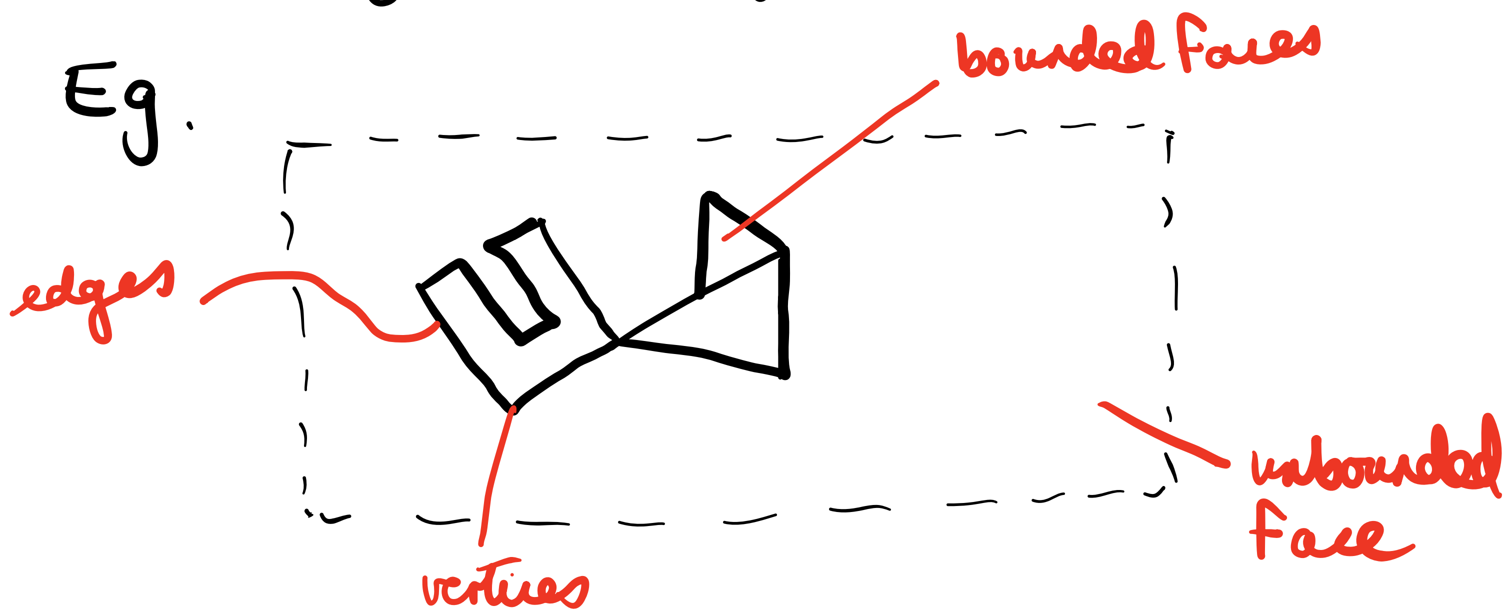
create overlay map



Maps

Maps are planar subdivisions - embedding of a graph into plane $\mathbb{R}^2$.

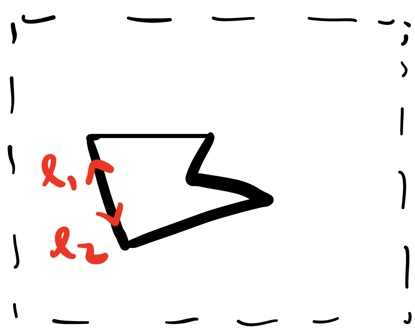
Eg.



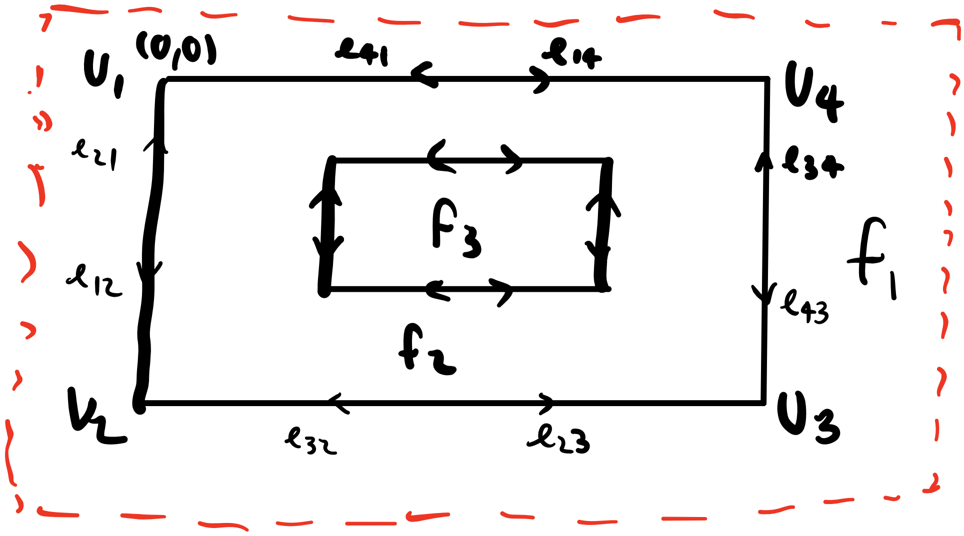
- Store planar subdivision in a DCEL: doubly connected edge list.

- In this approach, orientation of edges is important.

Eg.



DCELs



- 3 tables (vertices, edges, faces)

Table for vertices

Name of vertex	Co-ordinate	Edge originating @ vertex
V_1	(0,0)	e_{12}

DCELs

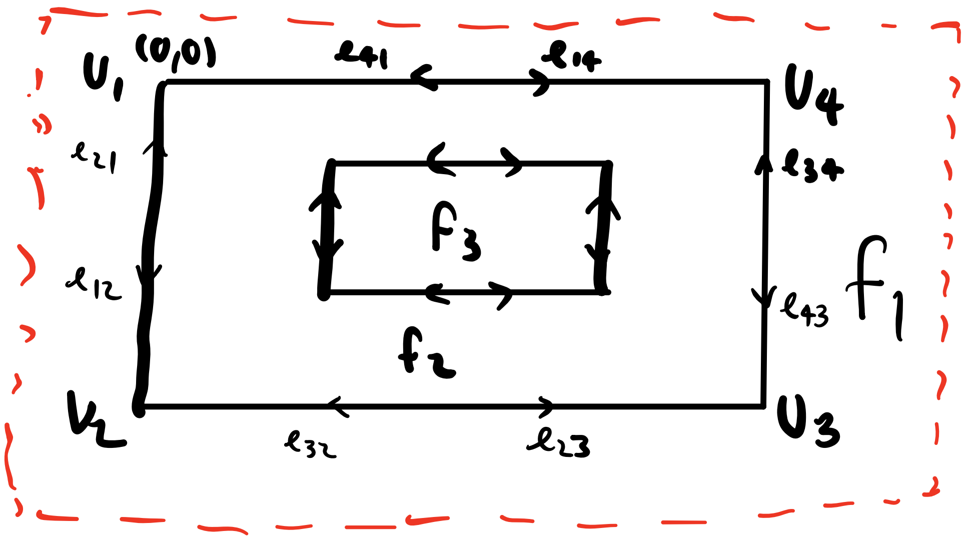


Table for edges

Name	Origin (vertex)	Twin	Next edge	Previous edge	Adjacent Face
e_{21}	v_2	e_{12}	e_{14}		f_1

- Adjacent Face : face to left of oriented edge

DCELs

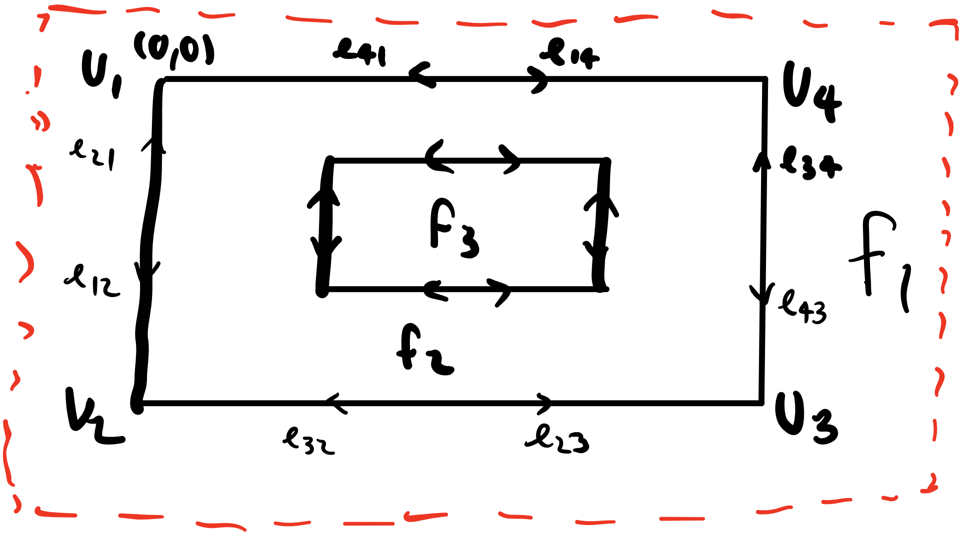


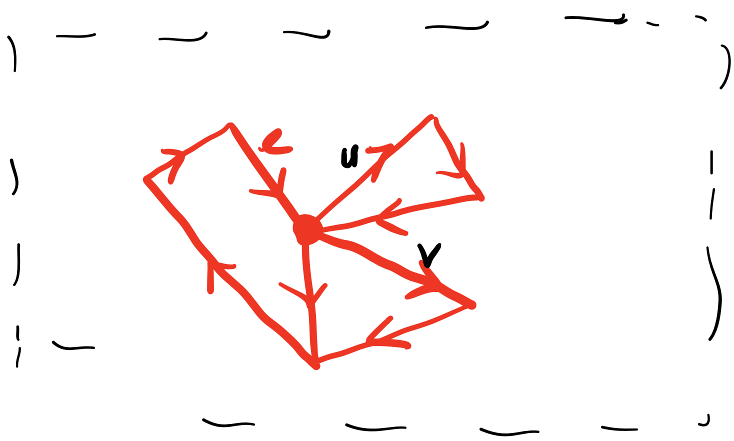
Table for edges

Name	Origin (vertex)	Twin	Next edge	Previous edge	Adjacent Face
e_{21}	v_2	e_{12}	e_{14}		f_1

- Next edge next(e) : ① Origin = endpoint of e
  ② same adjacent face as e

There is no edge between e & next(e) with these two properties.

Eg. in picture to right, u,v satisfy ①,② relative to e, but u = next(e).



DCELS

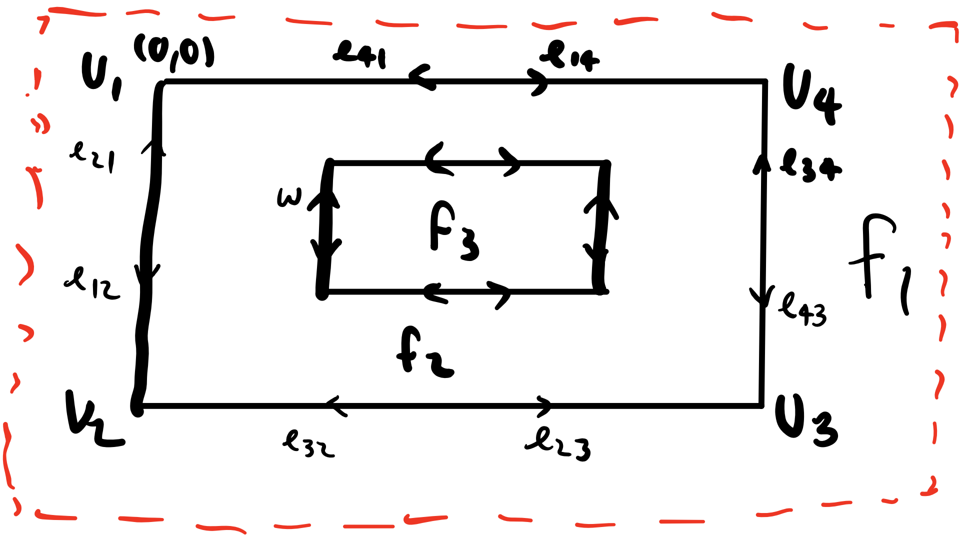


Table for faces

Name	1 edge on outer cycle	1 edge on each inner cycle
f_2	e_{12}	w
f_1	none	e_{21}

- Cycle : sequence $(e_1, \dots, e_n)$ of edges with $next(e_i) = e_{i+1}$ & $next(e_n) = e_1$ & no element repeated.

- Cycle of f if $f = adj(e_i)$ each e_i
- Outer cycle of f if e_i lies on outer boundary of f.
- Inner cycle of f otherwise.

Complexity of planar subdivision/DCEL is no. of vertices + no of edges + no of faces

Exercise : using DCEL, calculate all edges with vertex v in clockwise order.

Algorithm for map overlay

S_1 red map ---- D_1 DCEL

S_2 blue map ---- D_2 DCEL

$S = \text{Overlay}(S_1, S_2)$ - calculate DCEL D for overlay S.

Algorithm has 3 steps:

1. Put tables for vertices & edges of D_1 & D_2 into single table D (record colour of edges)
2. At this point D is incorrect: update using segment intersection algorithm,
3. Finally create table for faces. For each face $f \in D$, find blue face $f_1 \in D_1$ & red face $f_2 \in D$ in which f lies?

(Detailed version in Janku's Thesis - see E-learning)

Algorithm involves :

- event queue Q (balanced binary tree)
- balanced bin tree T of line segments (coloured)
- For each $p \in Q$, sets $L(p)$, $U(p)$ & $C(p)$ of coloured line segments on which p lies (like last week)

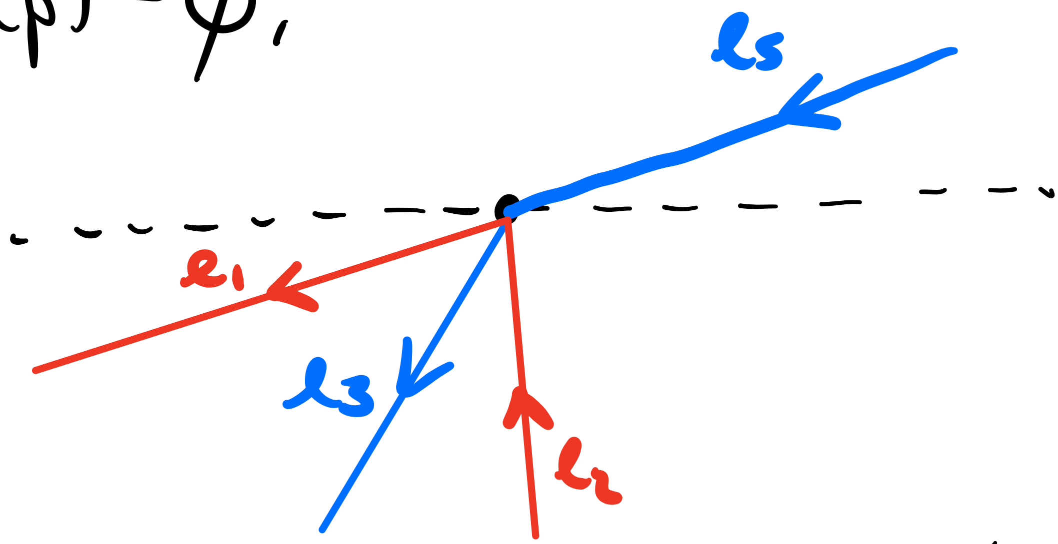
- Step ① of algorithm is straightforward. Here we describe steps ② & ③.

- For Step ②, we begin by adding all endpoints of segments to Q.

- Several cases to consider @ event point $p \in Q$.

• At $p \in Q$, update Q & T as in segment intersection alg but keeping track of colours of segs.

• If $C(p) = \phi$,



do not add new edges or vertices, only update next & previous.

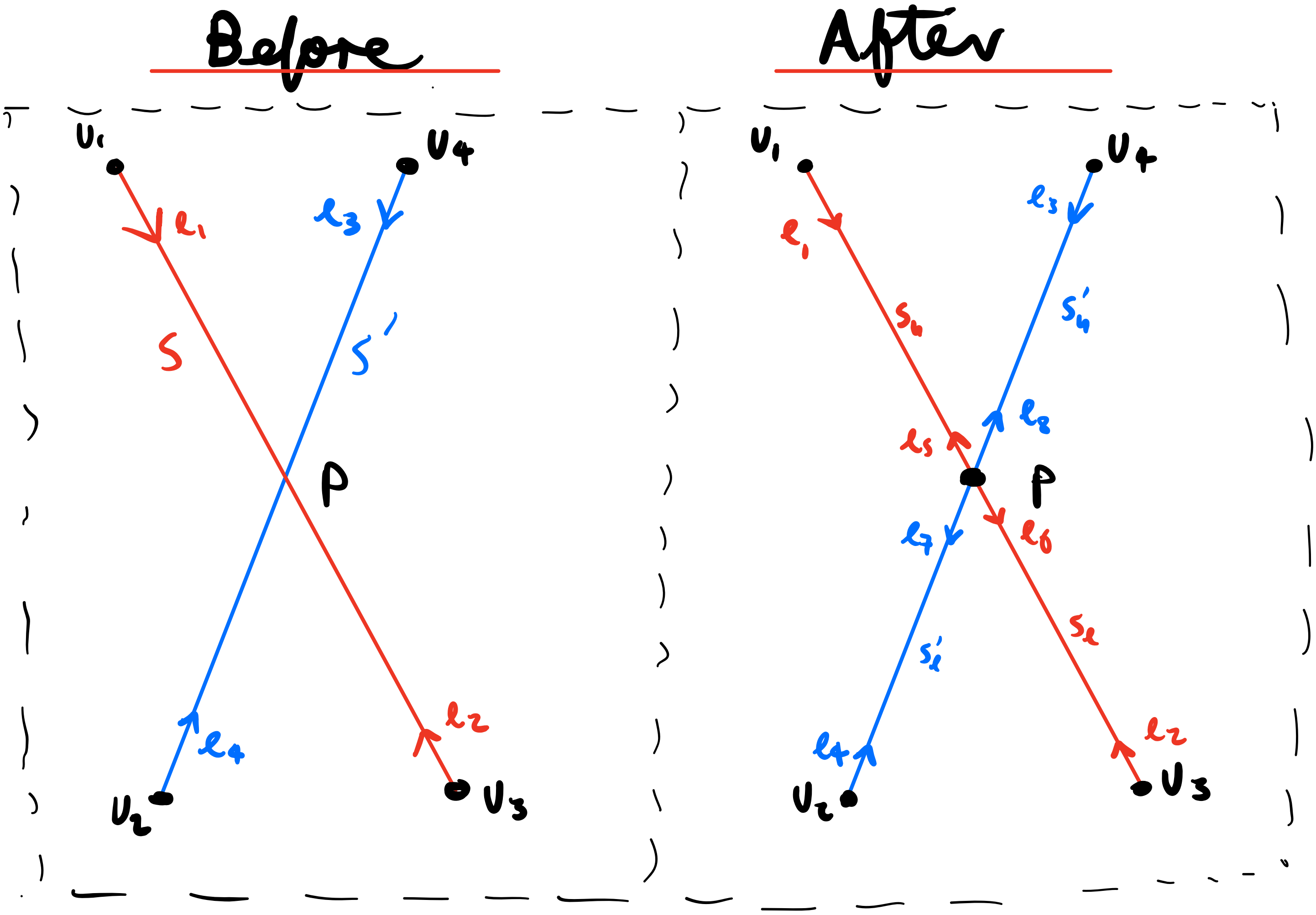
Original table

$next(e_2) = e_1$

New table

$next(e_2) = e_3$

• If $C(p) \neq \emptyset$,



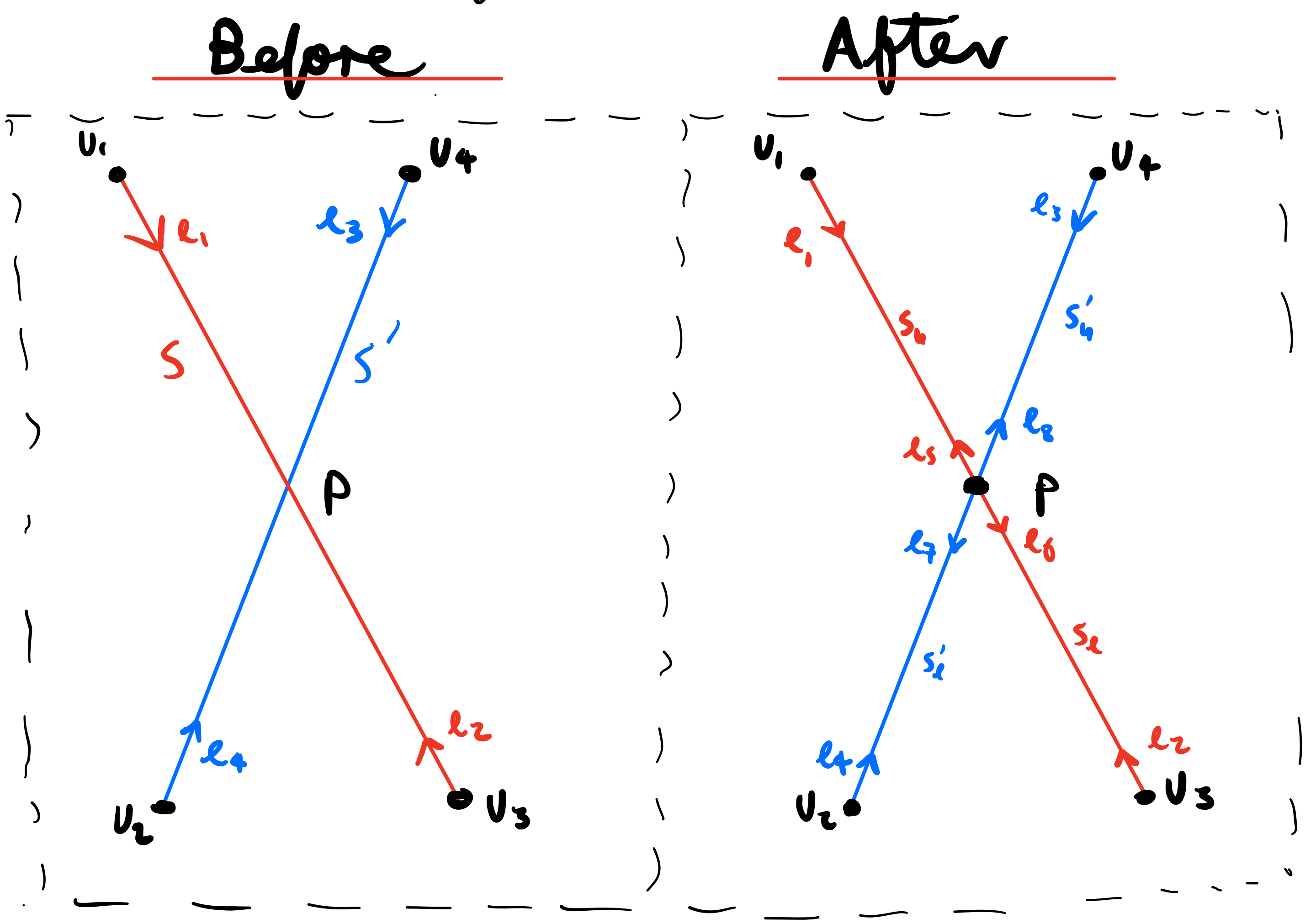
• split segments in $C(p)$ into 2 parts – upper & lower. Add lower parts to T.

• add new vertex p to D.

• Edges associated to these segments are split in 2 as in example above: add new rows for these edges & update origin, next, prev., adj. face

(see E-Learning for more detail)

• If $C(p) \neq \emptyset$,



Name of vertex	Co-ordinate	Edge originating @ vertex
p	intersection co-ord	e_5

Name	Origin (vertex)	Twin	Next edge	Previous edge	Adjacent Face
e_1	v_1	e_5	e_8	leave orig.	
e_5	p	e_1	same as for e_2	e_3	

- For Step 3, need to calculate Faces.

Idea

- Each bounded Face has unique outer cycle
  Outer Face has no outer cycle
- So can define Faces of D to be
  OuterCycles(D) $\cup \{ C_\infty \}$

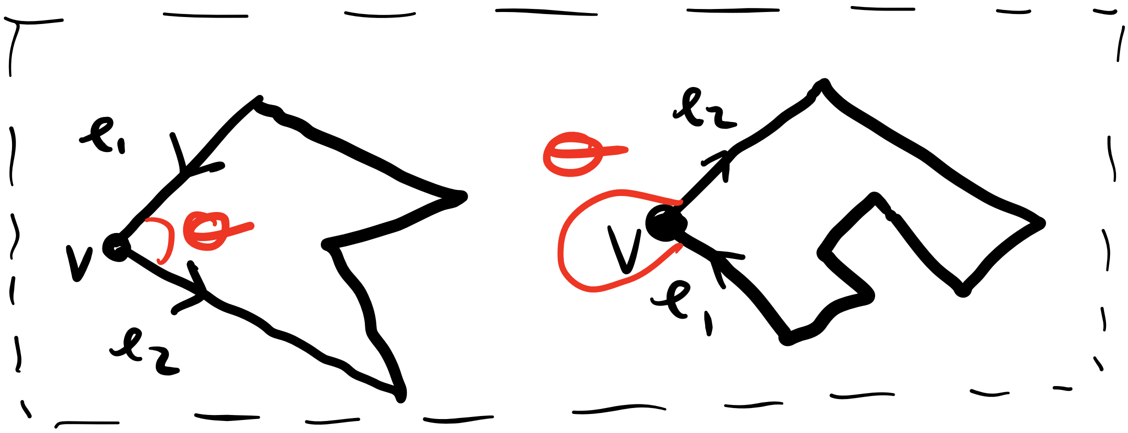
  imaginary outer cycle for unbounded region.

- Now we can calculate all of the cycles of D using the vertices & next pointers, which are already correct,

  but which of these cycles are outer & which are inner?

Outer or inner?

- At a cycle c, choose leftmost vertex v.



- Let e_1, e_2 be edges going into & out of v.
- Now measure angle θ between e_1, e_2 measured over adjacent region.

- $\theta < 180°$ ~ outer cycle
- $\theta > 180°$ ~ inner cycle

$$\theta < 180° \iff \det\begin{pmatrix} e_{1x} & e_{1y} \\ e_{2x} & e_{2y} \end{pmatrix} > 0$$

- Hence can determine outer cycles & Fill Faces in D (outer cycles $\cup$ C_∞) but need to complete tables

Name	Origin (vertex)	Twin	Next edge	Previous edge	Adjacent Face
✓	✓	✓	✓	✓	Some outer cycle?

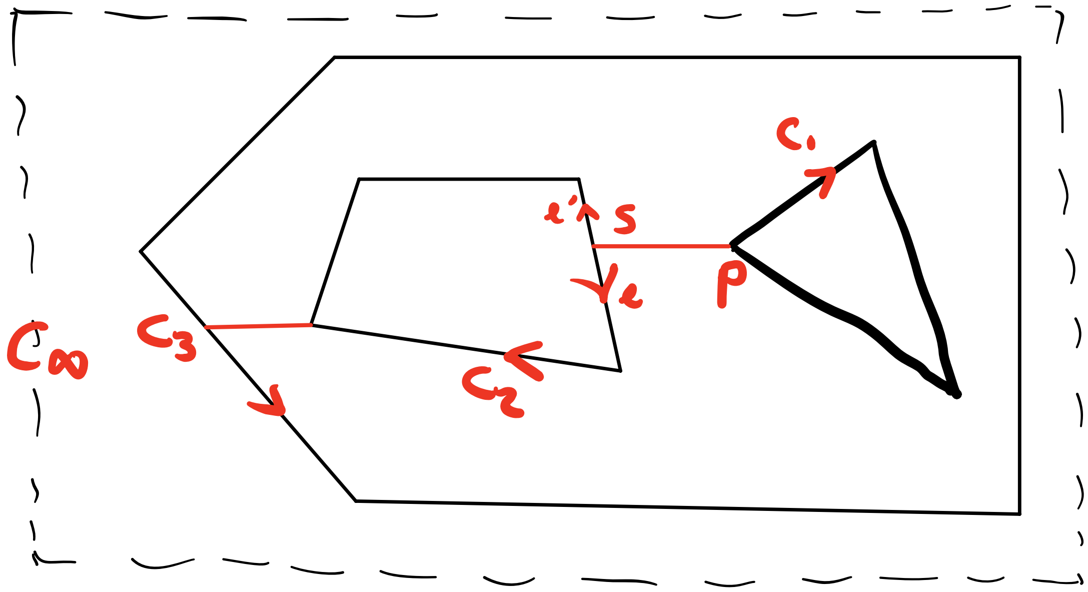
Table for faces

& Faces	1 edge on outer cycle	1 edge on each inner cycle
$(e_1, \ldots, e_n)$	?	?
C_∞	none	?

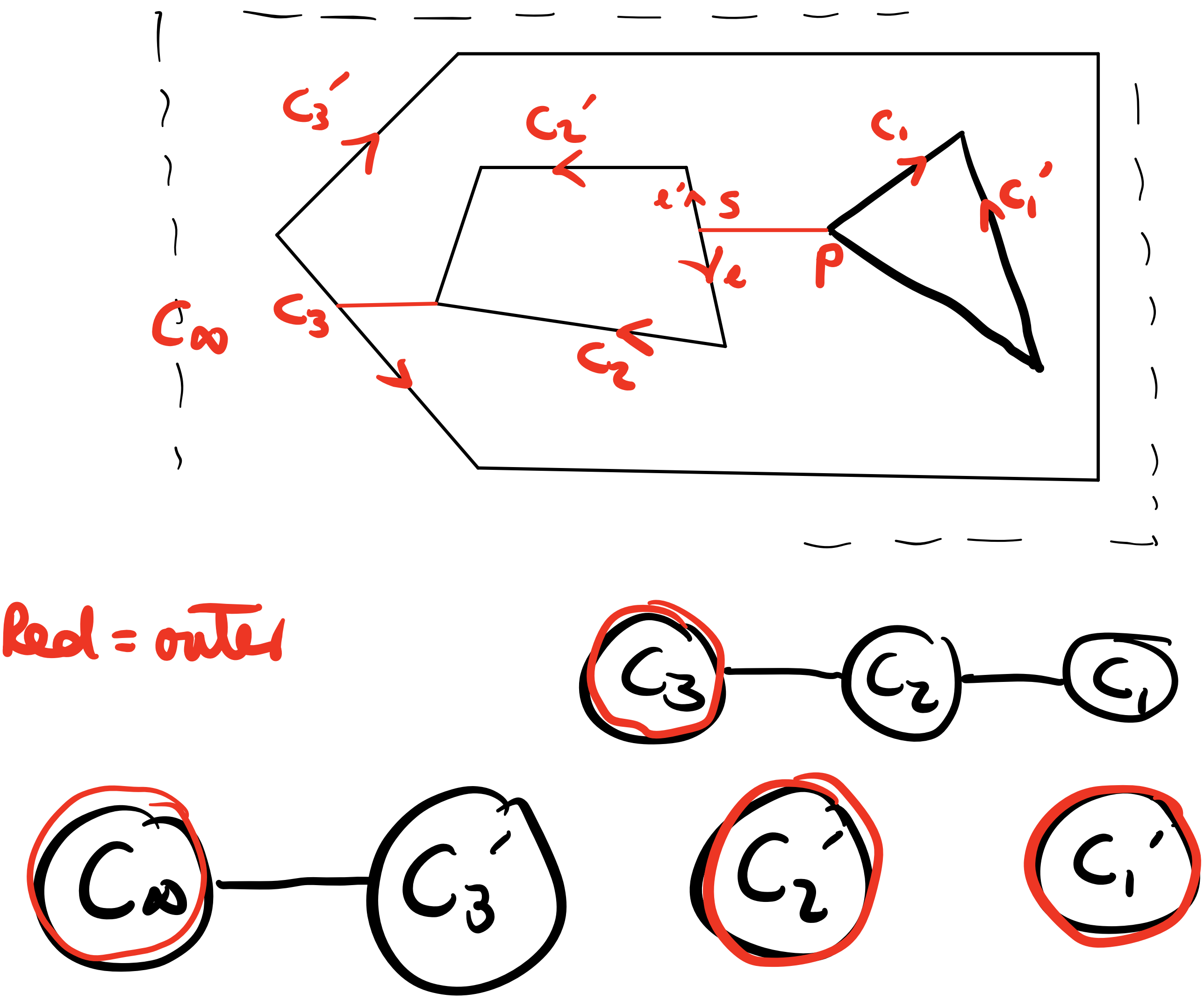
How to do it?

Make a graph G

- Vertices = cycles
- Take inner cycle c_1 with leftmost vertex p.
- Find closest segment s to its left, if it exists.



- It determines 2 edges e, e', one with p to its left, which we call c_2.
- Draw edge c_2 —— c_1
- If c_2 outer, stop; else repeat.
- If no edge to left, connect to c_∞.



- Each connected comp has exactly one face ~ outer cycle & specifies all its inner & outer cycles.

Identification of faces

- For each face $f \in D$, find faces $f_1 \in D_1$, $f_2 \in D_2$ in which f lies

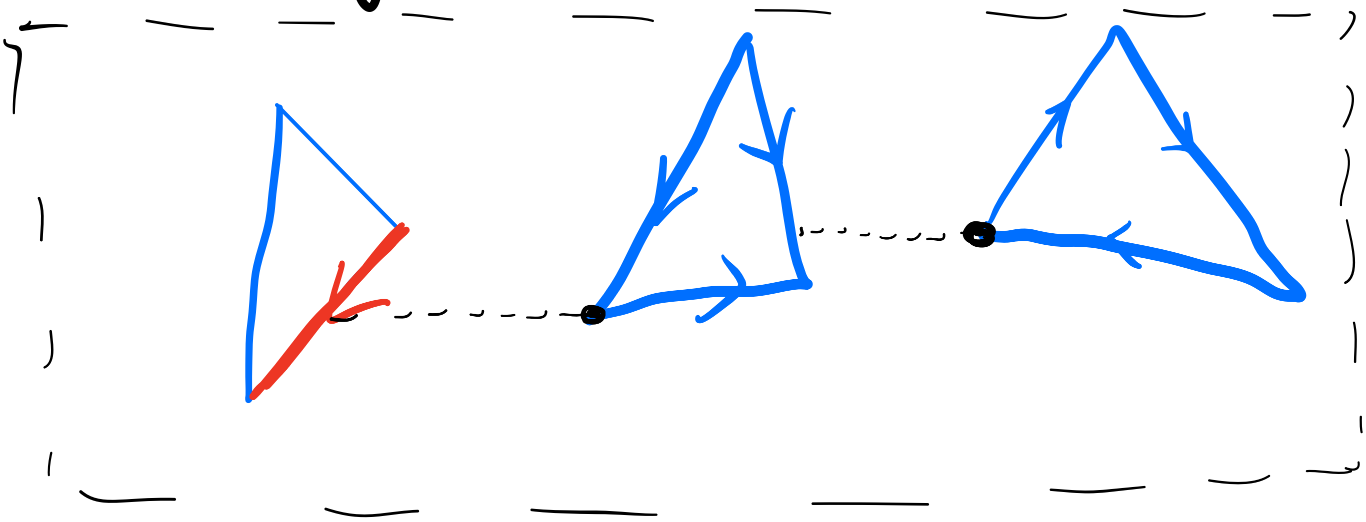
Two cases

① Cycle for f contains blue & red edges - take their adjacent faces in D_1 & D_2.

② - - - - - - - - only blue +
determines blue face in D_1.

What about red face?

- If f is unbounded, take unbounded face in D_2.
- Else f is bounded.



- Search to left, much as before, until we find cycle with red edge + take adj face in D_2.
- Else take unbounded face in D_2.
- See E-Learning for more detail.

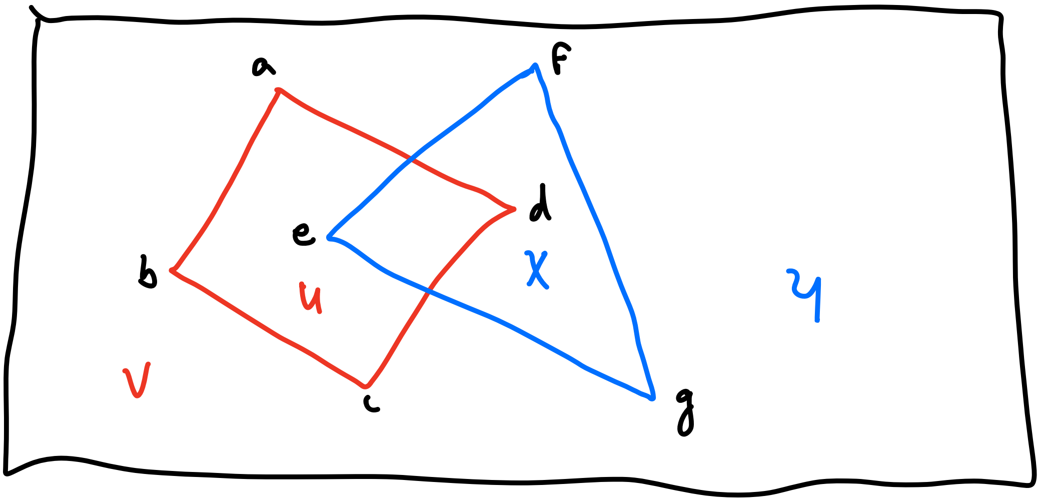
Complexity

- S_1 complexity n_1
- S_2 — — — — n_2
- $n = n_1 + n_2$

The alg for building DCEL for overlay has complexity

$$O(\underbrace{(n+k)}_{\text{no of intersections found}}\log n)$$

Exercise in class



Vertex	Co	Edge ov
a		ab
b		bc
c		cd
d		da
e		ef
f		fg
g		ge

Edge	Or	T	Ne	Pv	AF
ab	a	ba	bc		u
ba	b	ab	ad		v
bc	b	cb	cd		u
cb	c	bc	ba		v
cd	c	dc	da		u
dc	d	cd	cb		v
da	d	ad	ab		u
ad	a	da	dc		v
ef	e	fe	fg		y
fe	f	ef	eg		X
fg	f	gf	ge		y
gf	g	fg	fe		X
ge	g	eg	ef		y
eg	e	ge	gf		X

$Q = (f, a, d, e, b, c, g)$