Lecture 6 - Half-plane intersection

Consider a set $H = \{h_1, \ldots, h_n\}$ of half-planes

$h_i : a_i x + b_i y \leq c_i$

Goal: compute intersection $C = \bigcap_{h_i \in H} h_i$

Approach: recursive (divide & conquer)

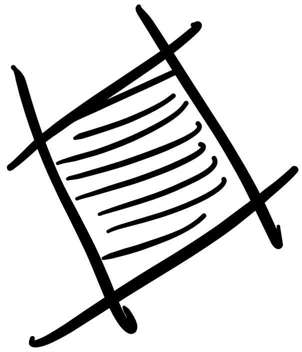
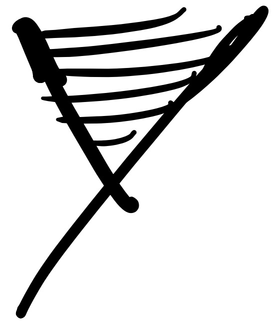
divide H into two sets H_1, H_2
of roughly same size;

Compute $C_1 = \bigcap_{h_i \in H_1} h_i$, $C_2 = \bigcap_{h_i \in H_2} h_i$

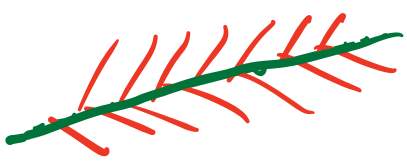
& then calc $C = C_1 \cap C_2$.

intersection of
<u>convex sets</u>

- Half-plane  is convex set.
- Intersection of convex sets is convex

Eg.  or 

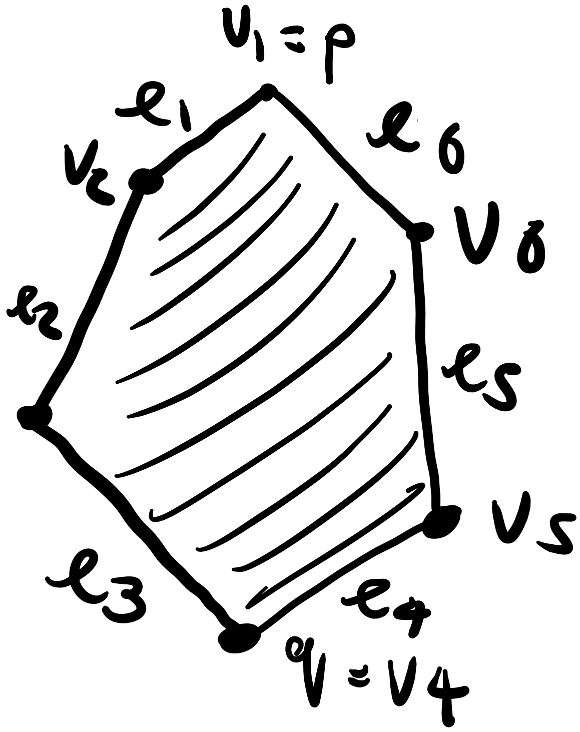
- What shapes of convex sets can arise as intersection of finitely many half-planes?
- How can we represent them computationally?
- Will consider those cases which are not subsets of a line



- Use lex ordering (top to bottom, left to right)

1) C has max p & min. q in lex order:

The boundary of C splits into left path $L(C)$ & right path $R(C)$ from top to bottom:

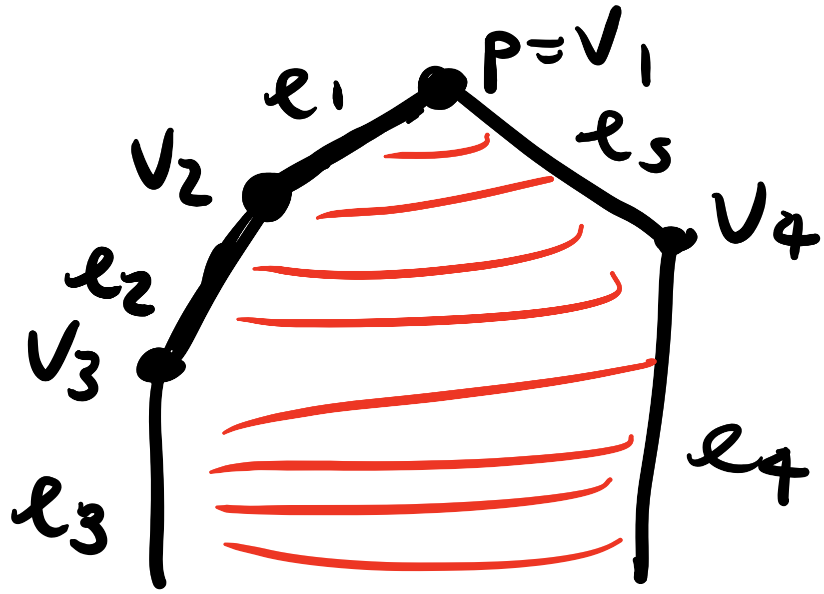


$LC = (v_1, e_1, v_2, e_2, v_3, e_3, v_4)$

$RC = (v_1, e_6, v_6, e_5, v_5, e_4, v_4)$.

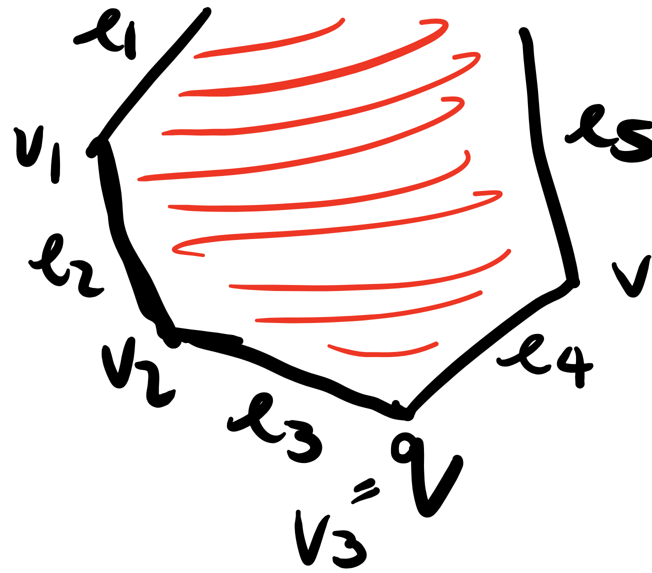
- C is determined by LC & RC.
- In each case (1 above & 2, 3, 4) we represent C by two sequences LC, RC consisting of vertices, edges, half-edges & unbounded edges.

2) C has max, no min



LC, RC are sequences beginning with a vertex & ending with a half-edge.

3) C has no max, has min



LC, RC begin with half-edges & end in vertices

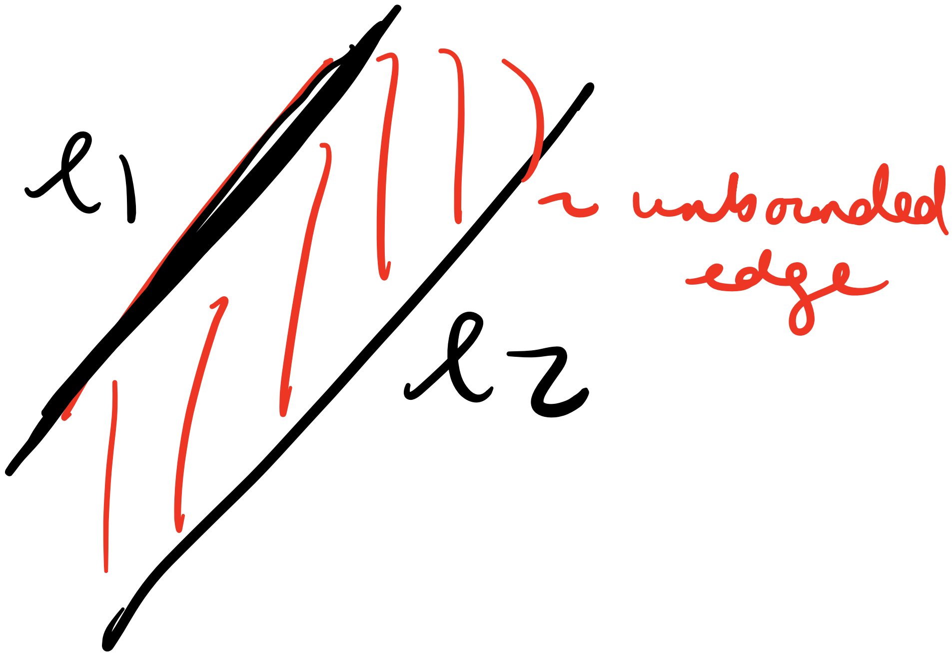
$$LC = (e_1, v_1, e_2, v_2, e_3, v_3)$$
$$RC = (e_5, v_4, e_4, v_3)$$

half-edges.

4) C has no min or max.

$LC = (\ell_1)$

$RC = (\ell_2)$



or

$LC = (\ell_1, v_1, \ell_2, v_3, \ell_3)$

$RC = \emptyset$

$LC = \emptyset$

$RC = (\ell_1, v_1, \ell_2, v_2, \ell_3)$

Algorithm : Half-plane Intersection (H)

- Input $H = \{h_1, \ldots, h_n\}$ set of half planes
- Output : Intersection C of H, described using sequences LC & RC of vertices, edges, half-edges & unbounded edges describing left & right path.

- If $n = 1$, determine LC & RC.

- Else, put
$$H_1 = \{h_1, \ldots, h_{\lfloor n/2 \rfloor}\} \ \& \ H_2 = H / H_1$$

- Set C_1 = Half-plane Intersection (H_1)
  C_2 = Half-plane Intersection (H_2)

C = IntersectionofTwo (C_1, C_2)

main thing

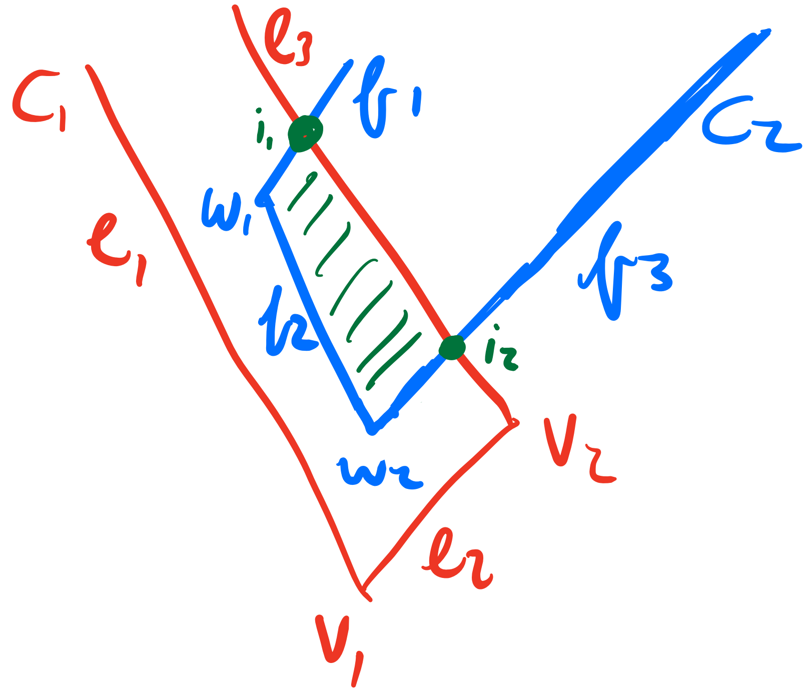
Illustration

$$
\begin{tikzcd}[column sep=small]
 & & & H = \{h_1, h_2, h_3, h_4\} \arrow[dll, dash] \arrow[drr, dash] & & & \\
 & H_1 = \{h_1, h_2\} \arrow[dl, dash] \arrow[dr, dash] & & & & H_2 = \{h_3, h_4\} \arrow[dl, dash] \arrow[dr, dash] & \\
H_{11} = \{h_1\} \arrow[d, equal] & & H_{12} = \{h_2\} \arrow[d, equal] & & H_{21} = \{h_3\} \arrow[d, equal] & & H_{22} = \{h_4\} \arrow[d, equal] \\
c_{11} \arrow[dr, dash] & & c_{12} \arrow[dl, dash] & & c_{21} \arrow[dr, dash] & & c_{22} \arrow[dl, dash] \\
 & c_1 = c_{11} \cap c_{12} \arrow[drr, dash] & & & & c_2 = c_{21} \cap c_{22} \arrow[dll, dash] & \\
 & & & c = c_1 \cap c_2. & & &
\end{tikzcd}
$$

Algorithm : Intersection Of Two (C_1, C_2)

Idea

- Sweepline going from top bottom.



- Calculate intersections as we move down page, & decide which vertices & edges on left/right path of C_1, C_2 lie on left right path of intersection.

$$L(C_1 \cap C_2) = \{i_1, f_1, w_1, f_2, w_2\}$$
$$R(C_1 \cap C_2) = \{i_2, f_3, w_2\}.$$

Algorithm : IntersectionOfTwo (C_1, C_2)

Input

C_1, C_2 : intersection of sets of halfplanes, described using lists of left & right boundaries LC_1, RC_1, LC_2 & RC_2.

Output

$C_1 \cap C_2$ descr. using lists $L(C_1 \cap C_2)$ & $R(C_1 \cap C_2)$

- Use sweep-line method:

  Q of events - vertices + intersections

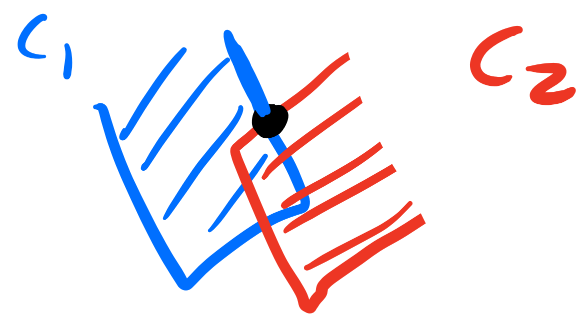
  T - ord. seq. of edges int. sweep-line

  (at most 4 so no need for tree)

Algorithm : Intersection Of Two (C_1, C_2)

① Add upper endpoints to Q, unbounded upper edges to T.

② If neither C_1, C_2 has max, calc. int. of unbounded edges & add to Q,

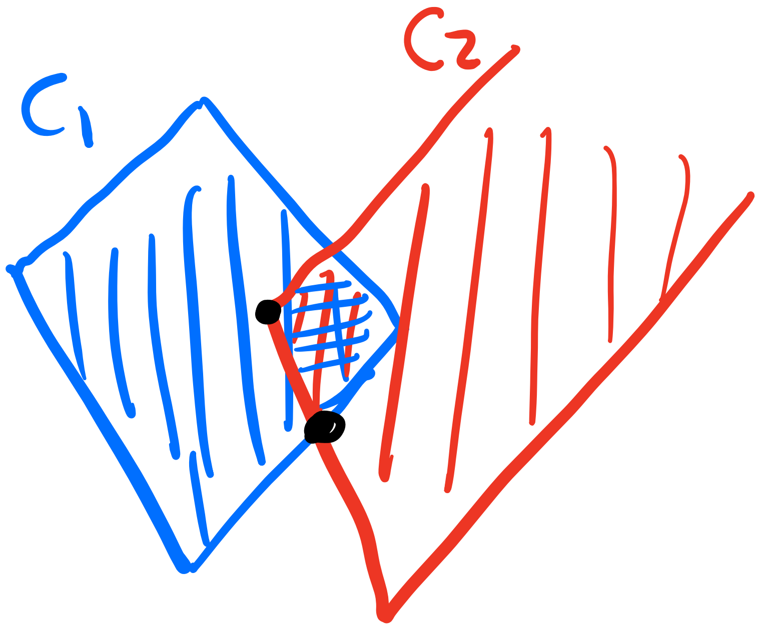


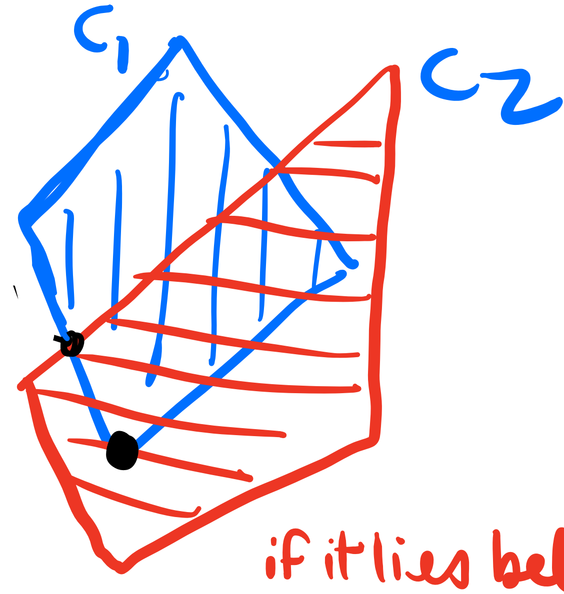
③ At event point v,

a) decide if $v \in L(C_1 \cap C_2) / R(C_1 \cap C_2)$ as below & add to appropriate list.

How to decide ?

Vertices of $L(C_1 \cap C_2)$

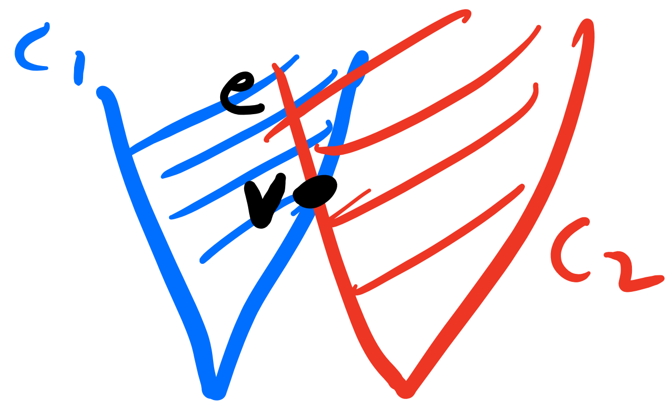




if it lies bet. l & r edges of C_1 in T!

- vertices of $L(C_2)$ inside C_1.
- vertices of $L(C_1)$ inside C_2.
- intersection points of $L(C_1)$ & $L(C_2)$,
- intersection points of left path of one & right path of the other (these will be max & min of $C_1 \cap C_2$)
- Sim. $R(C_1 \cap C_2)$

b)



Add e to $L(C_1 \cap C_2)$, so $\{e, v\}$.

- If v is first element of $L(C_1 \cap C_2)$ or $R(C_1 \cap C_2)$ find edges with v as lower endpoint & decide which belong to $L(C_1 \cap C_2) / R(C_1 \cap C_2)$:
  - if e appears before v in $L(C_1)$, then $e \in L(C_1 \cap C_2) \Leftrightarrow e \in C_2$ above v

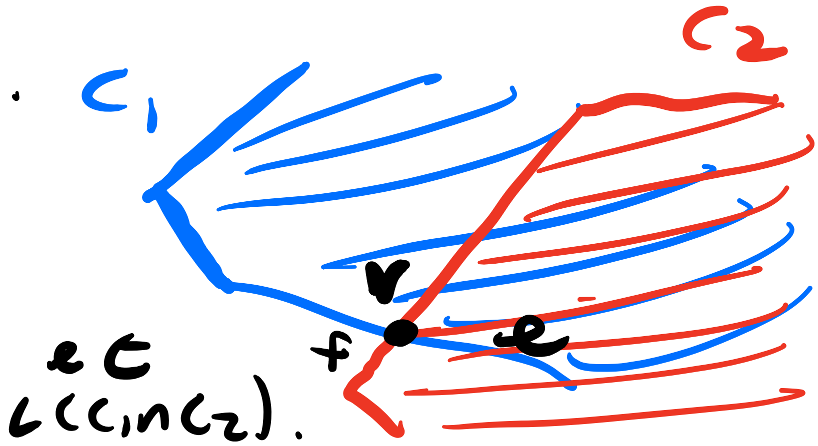
    etc

- Add to appropriate path.

c) Update T.

d) Look at edges going downwards from v - decide which lies in $C_1 \cap C_2$ & on which path $L(C_1 \cap C_2)$ or $R(C_1 \cap C_2)$.

Add to path.

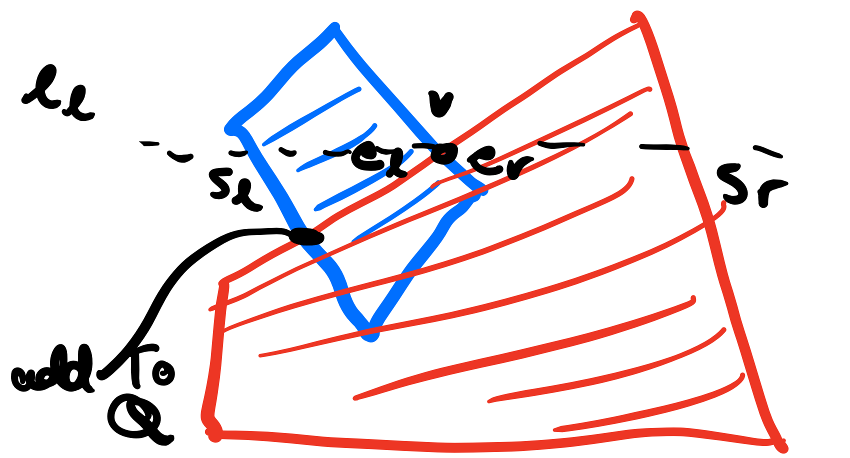


Add their lower endpoints to Q.

e) Let e_l, e_r be leftmost & rightmost edges going down from v

Find left edge s_l to e_l from other set.

Find right edge s_r to e_r from other set.



Calculate $s_l \cap e_l$

& $s_r \cap e_r$ & add them to Q.

f) If v is last member of $L(C_1 \cap C_2)$ &/or $R(C_1 \cap C_2)$

(ie no edge coming out below in int)

then we have computed intersection

→ empty the Q.

else

delete v from Q.

Warning / Note on complexity

- ELearning adds all endpoints to Q but then insertions to Q will cost $O(n_1+n_2)$, whereas we want complexity $O(n_1+n_2)$ overall.
- With above approach, Q contains at most

- uppermost verts of path (≤ 4)
- endpoints of segs int sweepline (≤ 8)
- int. points of segs int sw. (≤ 4)

so at most 16 events in Q at any time. → time to handle event is constant.

Then

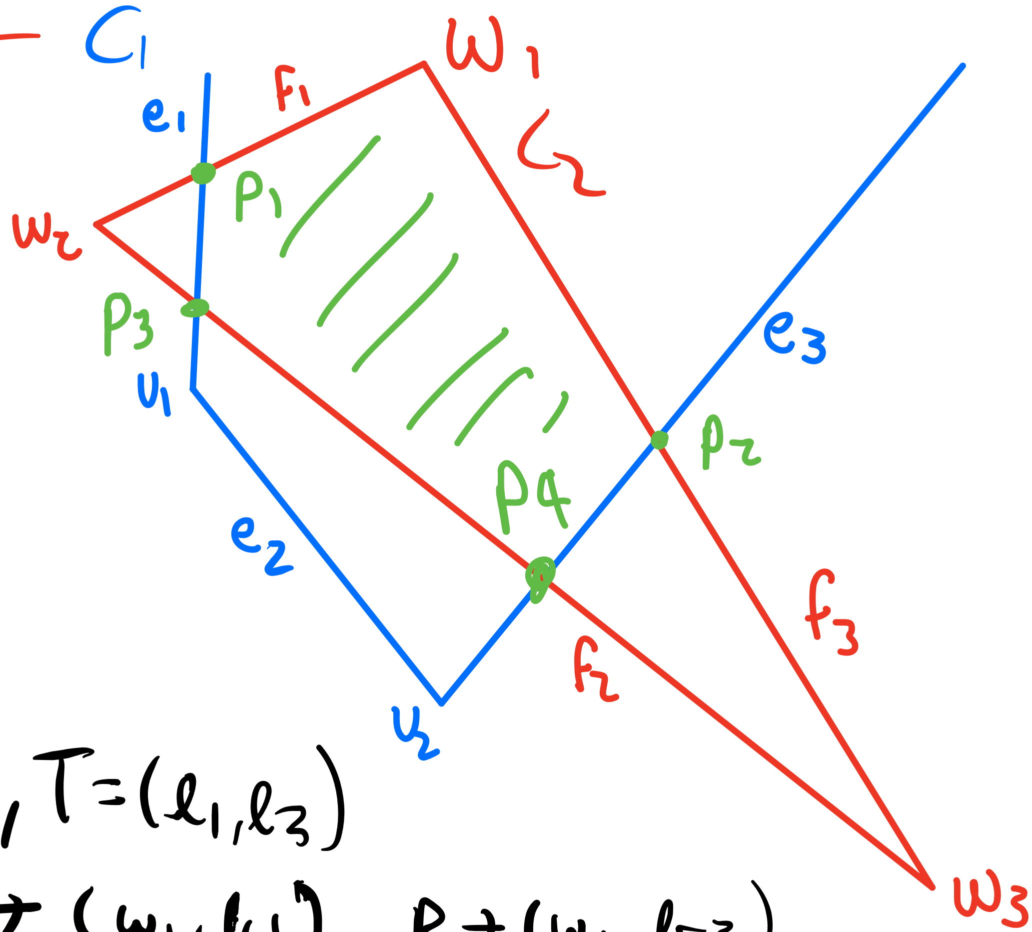
$$\mathrm{IntersectionOfTwo}(\underbrace{C_1}_{\substack{n_1 \\ \text{vertices}}}, \underbrace{C_2}_{\substack{n_2 \\ \text{vertices}}})$$

has complexity $O(n_1 + n_2)$

$T(n)$ - complexity of half-plane intersection alg.

$$\begin{aligned} T(n) &= 2T(n/2) + \mathrm{Time}(\mathrm{Int.OfTwo}(n/2, n/2)) \\ &= 2T(n/2) + O(n) \\ &\Rightarrow T(n) = O(n \log n). \end{aligned}$$

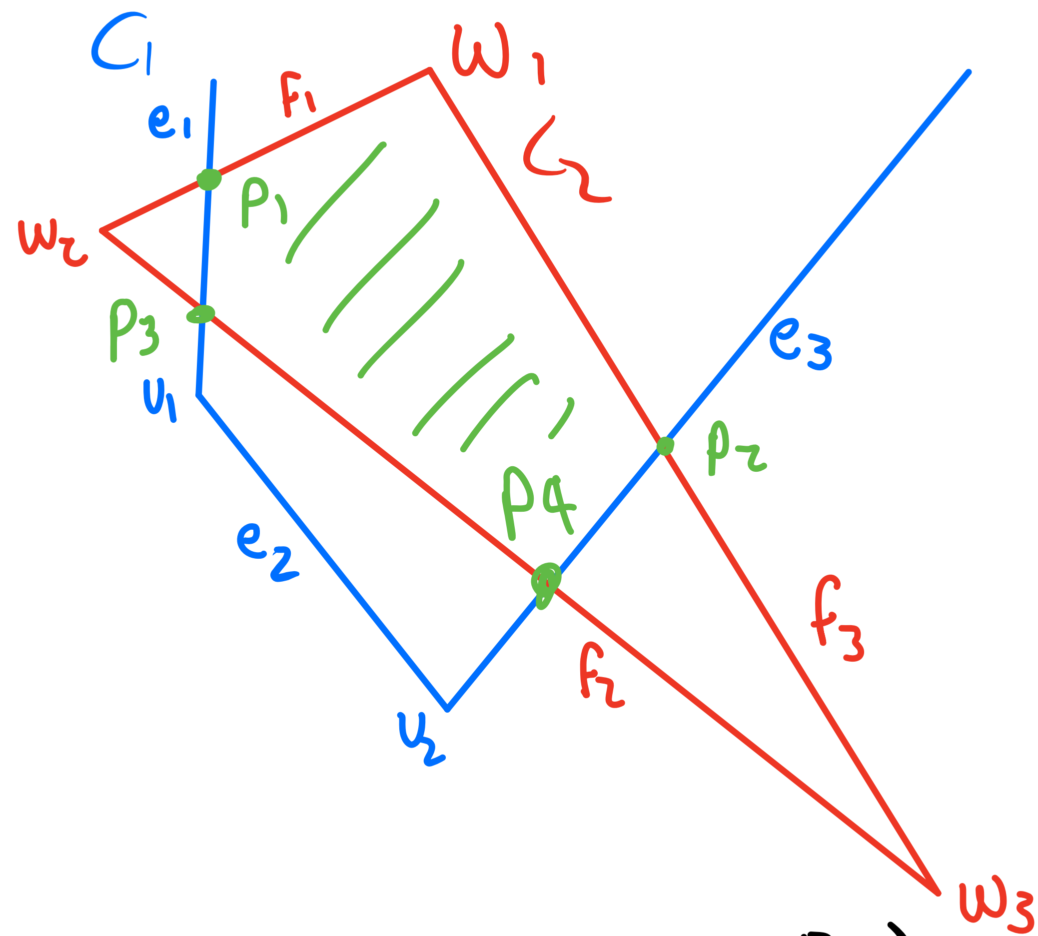
Example



$Q = (w_1)$, $T = (e_1, e_3)$

- At w_1, $\alpha \to (w_1, f_1)$, $R \to (w_1, f_3)$
  $Q \to (w_2, w_3)$
  $T \to (e_1, f_1, f_3, e_3)$ omit tree in future
  Find int $p_1, p_2 \to Q \to (w_1, p_1, p_2, w_3)$
- At p_1, $\alpha \to (w_1, f_1, p_1, e_1)$, $R \to R$
  $Q \to (w_2, v_1, v_3)$
- At w_2, α, R same. Find p_3.
  $Q \to (p_3, v_1, p_2, w_3)$

Example



- At p_3 $\alpha \to (w_1, f_1, p_1, e_1, p_3, F_2)$
  $Q \to (v_1, p_2, v_3)$
- @ v_1, $Q \to (p_2, v_2, w_3)$
- @ p_2, $R \to (w_1, f_3, p_2, e_3)$,
  calc $p_4 \to$
  $Q \to (p_4, v_2, w_3)$
- @ p_4 $\alpha \to (w_1, f_1, p_1, e_1, p_3, F_2, p_4)$
  $R \to (w_1, f_3, p_2, e_3, p_4]$