Geometria Computacional

Aula 7

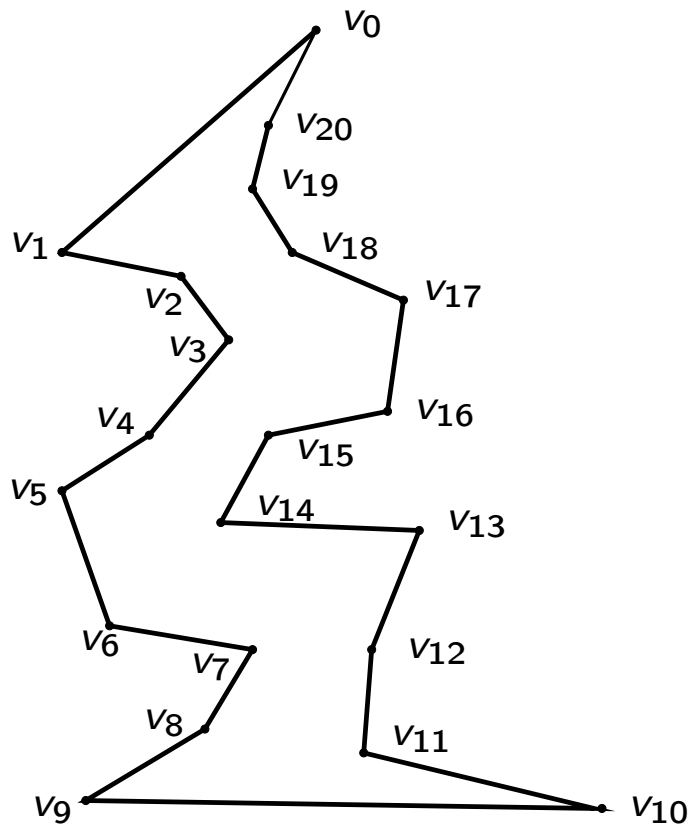
Triangulação de polígonos arbitrários

Sec 5 do texto do JAI 2009
https://www.ime.usp.br/~cris/jai2009/

Polígonos monótonos

Um polígono P é monótono em relação a uma reta L se $P \cap L'$ é conexo para toda reta L' perpendicular a L.

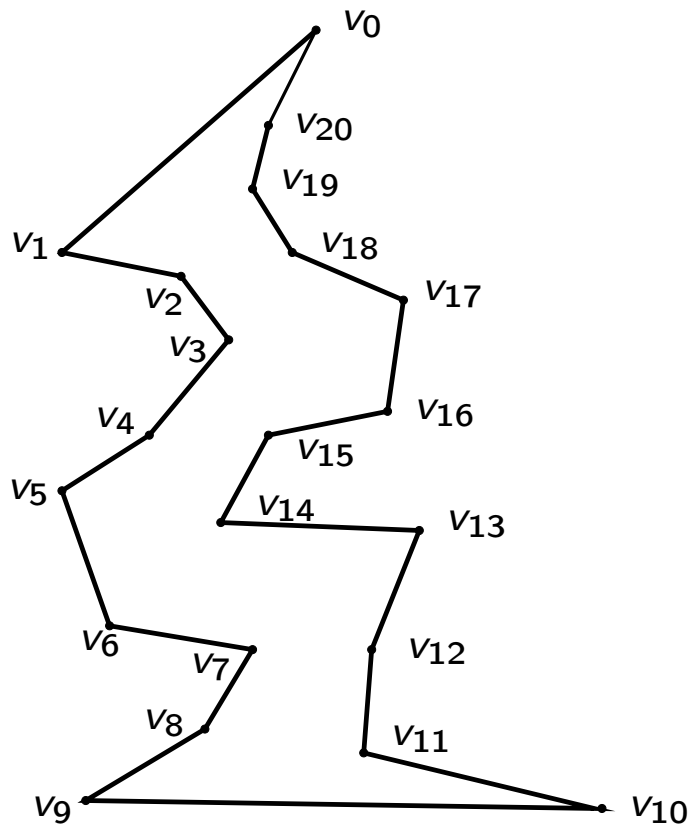
Se L é o eixo y, dizemos que P é y-monótono.



Polígonos monótonos

Um polígono P é monótono em relação a uma reta L se $P \cap L'$ é conexo para toda reta L' perpendicular a L.

Se L é o eixo y, dizemos que P é y-monótono.



Sabemos triangular P em tempo linear.

Triangulação em $O(n \lg n)$

P: polígono arbitrário com n vértices

Ideia do algoritmo:

Triangulação em $O(n \lg n)$

P: polígono arbitrário com n vértices

Ideia do algoritmo:

- particionar P por diagonais em polígonos monótonos
- triangular cada um deles em tempo linear

Triangulação em $O(n \lg n)$

P: polígono arbitrário com n vértices

Ideia do algoritmo:

- particionar P por diagonais em polígonos monótonos
- triangular cada um deles em tempo linear

Partição tem que consumir tempo $O(n \lg n)$!

Triangulação em $O(n \lg n)$

P: polígono arbitrário com n vértices

Ideia do algoritmo:

- particionar P por diagonais em polígonos monótonos
- triangular cada um deles em tempo linear

Partição tem que consumir tempo $O(n \lg n)$!

Como fazemos isso?

Triangulação em $O(n \lg n)$

P: polígono arbitrário com n vértices

Ideia do algoritmo:

- particionar P por diagonais em polígonos monótonos
- triangular cada um deles em tempo linear

Partição tem que consumir tempo $O(n \lg n)$!

Como fazemos isso?

Usando uma trapezoidação especial de P.

Trapezoidação

Trapézio: quadrilátero com duas arestas paralelas

Trapezoidação

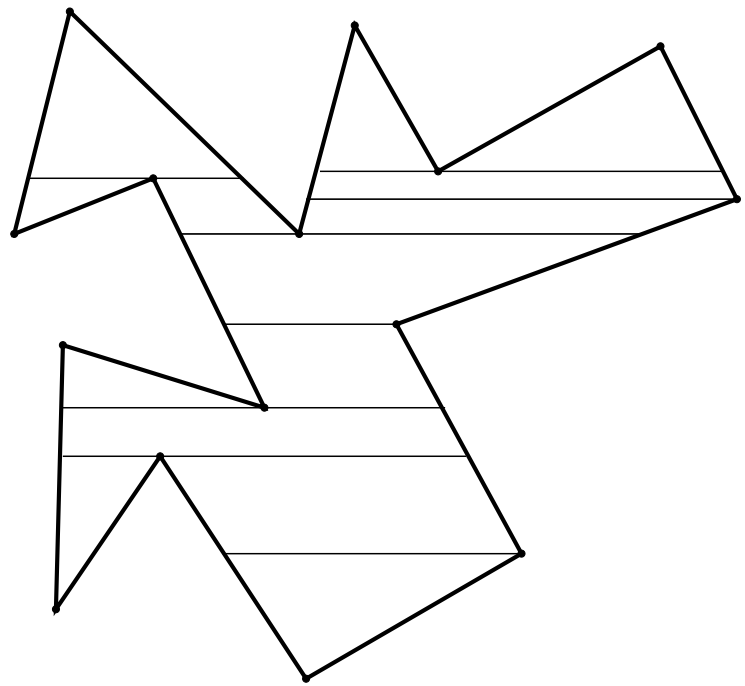
Trapézio: quadrilátero com duas arestas paralelas

Trapezoidação horizontal de um polígono P:
resultado de traçar segmentos horizontais maximais contidos em P, passando por cada vértice de P.

Trapezoidação

Trapézio: quadrilátero com duas arestas paralelas

Trapezoidação horizontal de um polígono P:
resultado de traçar segmentos horizontais maximais contidos em P, passando por cada vértice de P.



Trapezoidação

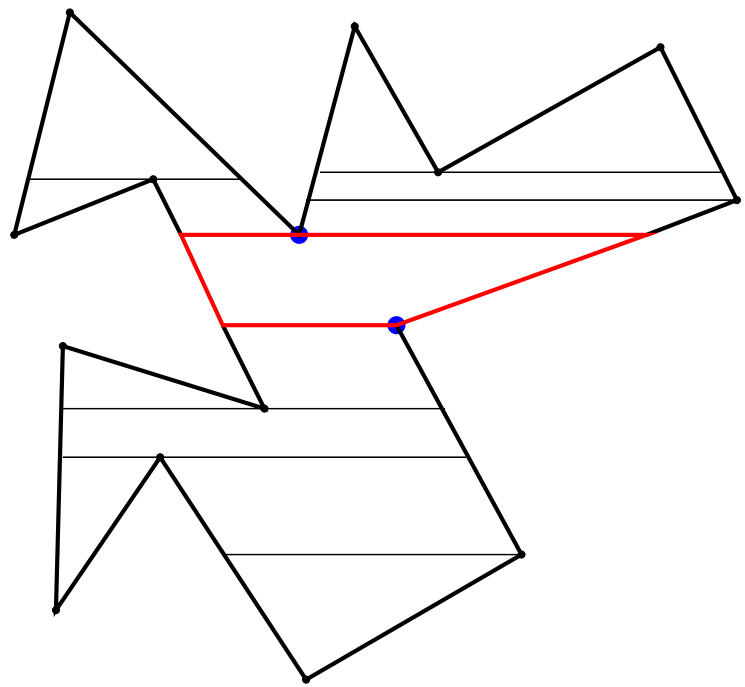
Hipótese simplificadora:

não há dois vértices com a mesma y-coordenada.

Trapezoidação

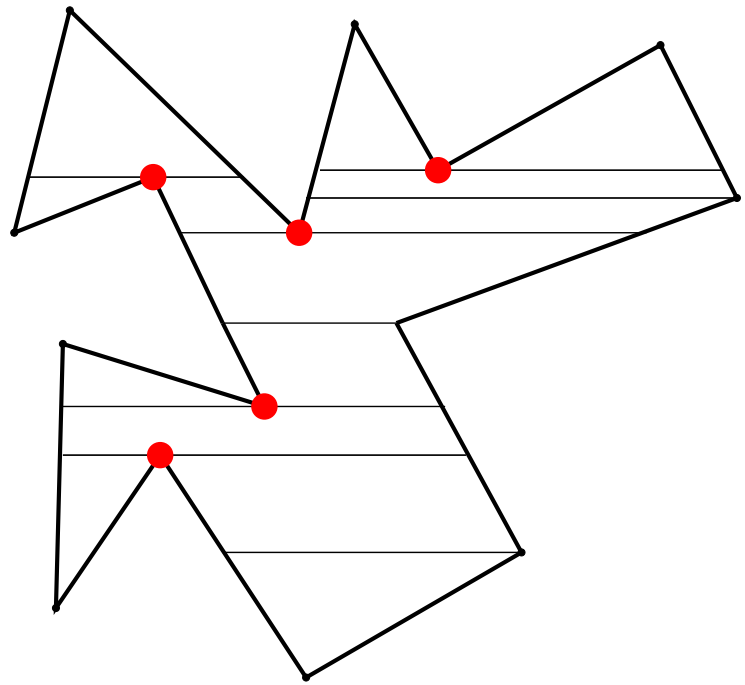
Hipótese simplificadora:
não há dois vértices com a mesma y-coordenada.

Afirmação: todo trapézio tem exatamente dois vértices de P em sua fronteira (vértices de suporte), um na aresta superior, outro na inferior.



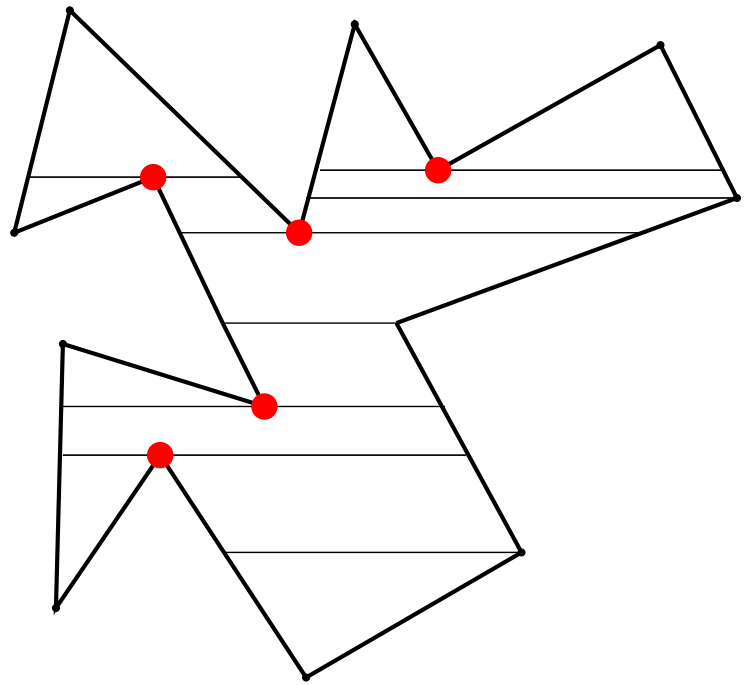
Pontas interiores

Ponta interior de P: vértice v reflexo cujos vizinhos em δP estão ambos acima ou ambos abaixo de v.



Pontas interiores

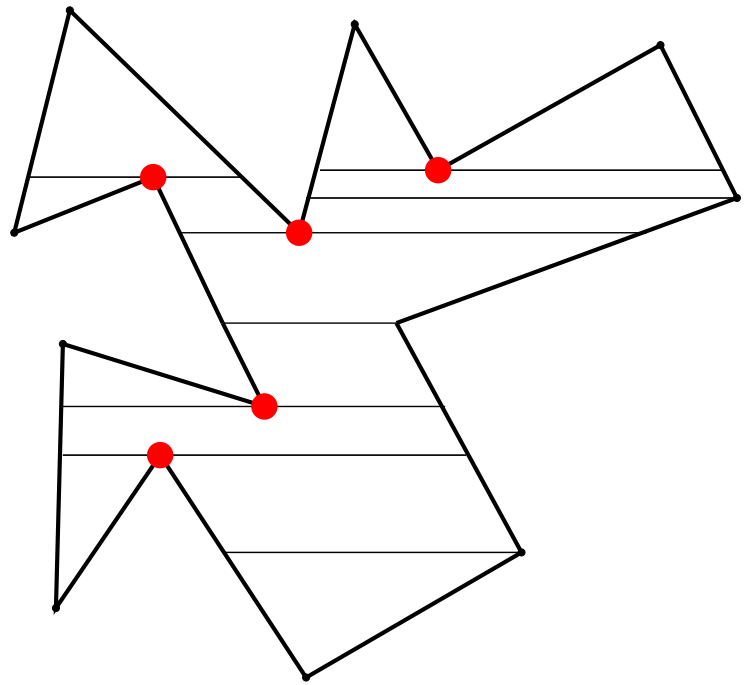
Ponta interior de P: vértice v reflexo cujos vizinhos em δP estão ambos acima ou ambos abaixo de v.



Lema: Se P não tem pontas interiores, então P é y-monótono.

Pontas interiores

Ponta interior de P: vértice v reflexo cujos vizinhos em δP estão ambos acima ou ambos abaixo de v.



Lema: Se P não tem pontas interiores, então P é y-monótono.

Ponta interior de P:
vértice de suporte no interior da aresta do seu trapézio.

Partição em polígonos monótonos

Lema: Se P não tem pontas interiores, então P é monótono.

Ideia: acabar com as pontas interiores!

Partição em polígonos monótonos

Lema: Se P não tem pontas interiores, então P é monótono.

Ideia: acabar com as pontas interiores!

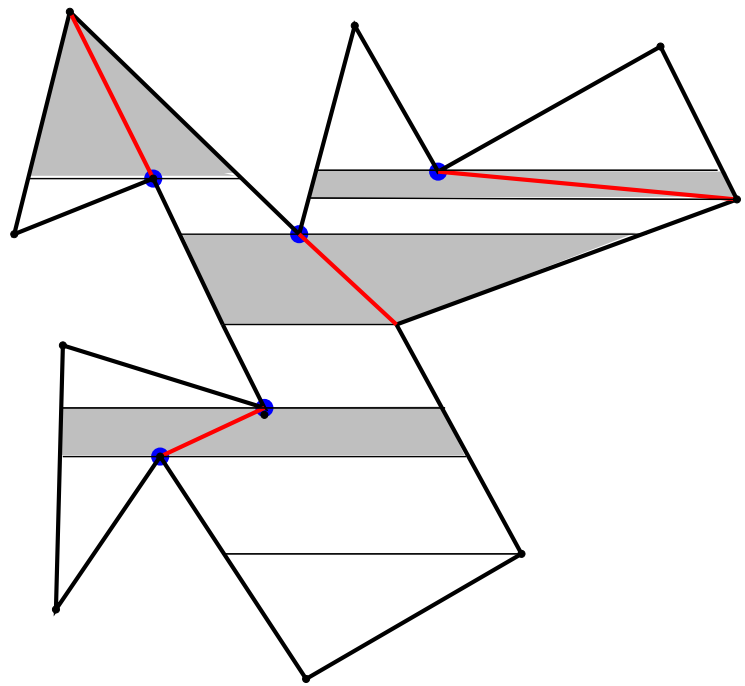
Como?

Partição em polígonos monótonos

Lema: Se P não tem pontas interiores, então P é monótono.

Ideia: acabar com as pontas interiores!

Como?



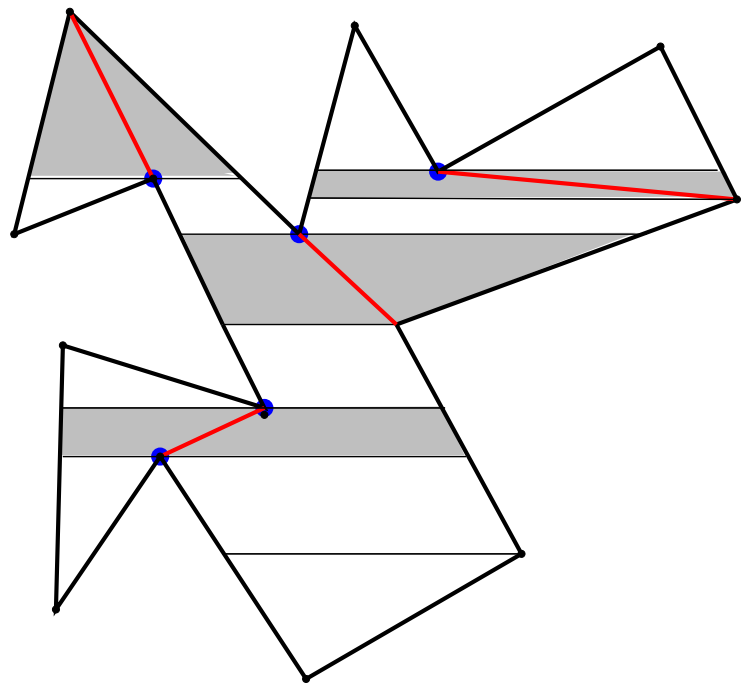
Uma diagonal a partir de cada ponta interior:

Partição em polígonos monótonos

Lema: Se P não tem pontas interiores, então P é monótono.

Ideia: acabar com as pontas interiores!

Como?



Uma diagonal a partir de cada ponta interior:

diagonal entre a ponta e o outro vértice de suporte.

Algoritmo de Lee e Preparata

Entrada: polígono P com n vértices
Saída: triangulação de P

Algoritmo de Lee e Preparata

Entrada: polígono P com n vértices
Saída: triangulação de P

Técnica: linha de varredura

Eventos: vértices de P, ordenados por y-coordenada

Algoritmo de Lee e Preparata

Entrada: polígono P com n vértices
Saída: triangulação de P

Técnica: linha de varredura

Eventos: vértices de P, ordenados por y-coordenada

ED para a linha de varredura ℓ: ABBB ou treap ou skip list

O que é guardado na ED da linha de varredura?

Algoritmo de Lee e Preparata

Entrada: polígono P com n vértices
Saída: triangulação de P

Técnica: linha de varredura

Eventos: vértices de P, ordenados por y-coordenada

ED para a linha de varredura ℓ: ABBB ou treap ou skip list

O que é guardado na ED da linha de varredura?

Trapézios que cruzam ℓ, dados por triplas (e, u, f), onde

- e e f são as arestas de P que contêm respectivamente o lado esquerdo e direito do trapézio
- u é o vértice de suporte superior do trapézio

Algoritmo de Lee e Preparata

Entrada: polígono P com n vértices
Saída: triangulação de P

Técnica: linha de varredura

Eventos: vértices de P, ordenados por y-coordenada

ED para a linha de varredura ℓ: ABBB ou treap ou skip list

O que é guardado na ED da linha de varredura?

Trapézios que cruzam ℓ, dados por triplas (e, u, f), onde

- e e f são as arestas de P que contêm respectivamente o lado esquerdo e direito do trapézio
- u é o vértice de suporte superior do trapézio

(u: candidato a extremo de uma diagonal particionadora)

Algoritmo de Lee e Preparata

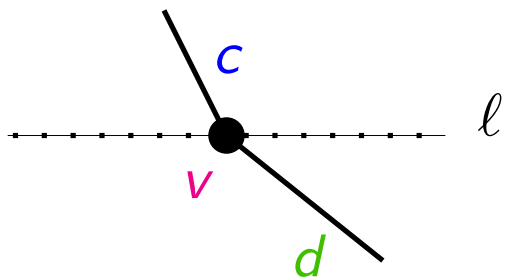
Em cada iteração, um evento (vértice) v é processado.
Linha de varredura ℓ sobre v.

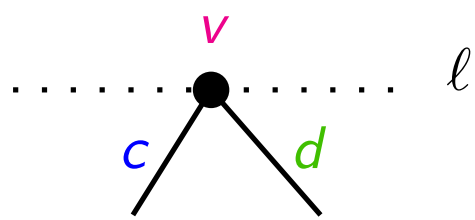
Algoritmo de Lee e Preparata

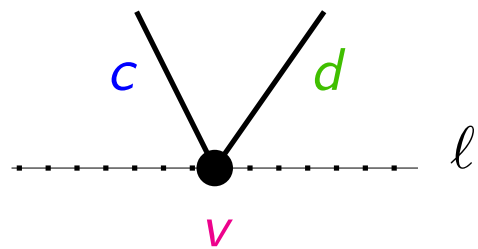
Em cada iteração, um evento (vértice) v é processado.
Linha de varredura ℓ sobre v.

c e d: arestas do polígono incidentes a v

Três casos a considerar:





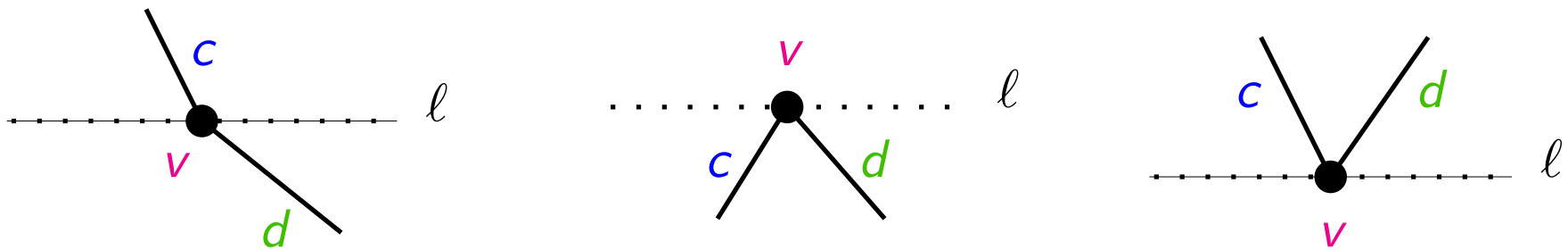


Algoritmo de Lee e Preparata

Em cada iteração, um evento (vértice) v é processado.
Linha de varredura ℓ sobre v.

c e d: arestas do polígono incidentes a v

Três casos a considerar:



Caso 1. Aresta c está acima de ℓ e d abaixo

Caso 2. Arestas c e d estão abaixo de ℓ

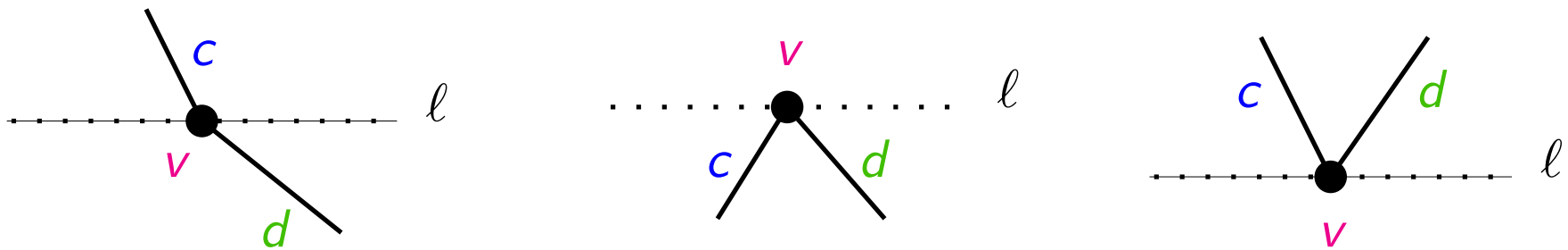
Caso 3. Arestas c e d estão acima de ℓ

Algoritmo de Lee e Preparata

Em cada iteração, um evento (vértice) v é processado.
Linha de varredura ℓ sobre v.

c e d: arestas do polígono incidentes a v

Três casos a considerar:



Caso 1. Aresta c está acima de ℓ e d abaixo

Caso 2. Arestas c e d estão abaixo de ℓ

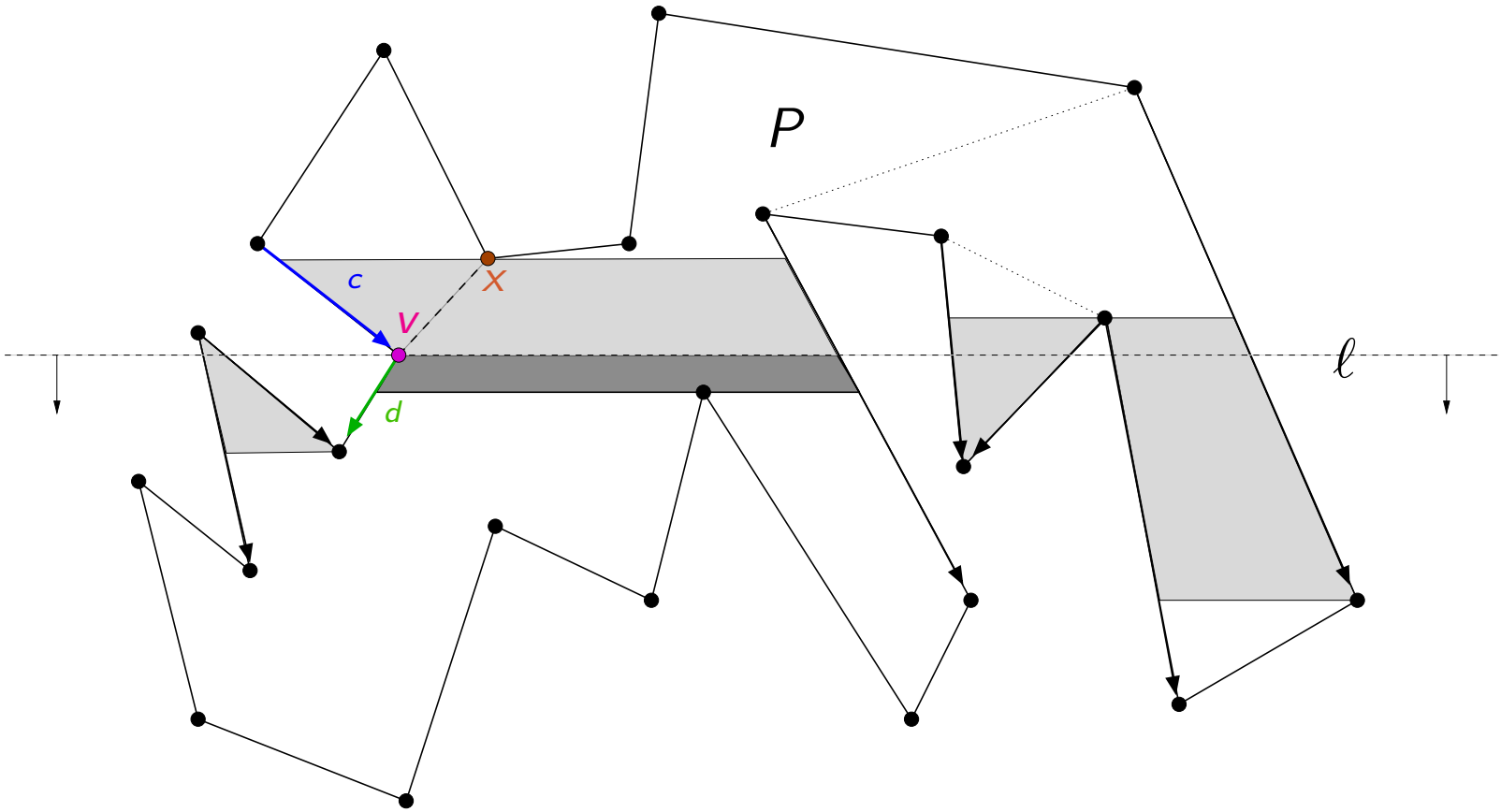
Caso 3. Arestas c e d estão acima de ℓ

No que segue, a ED da linha de varredura é uma ABBB T.

Caso 1

Caso 1. Aresta c está acima de ℓ e d abaixo

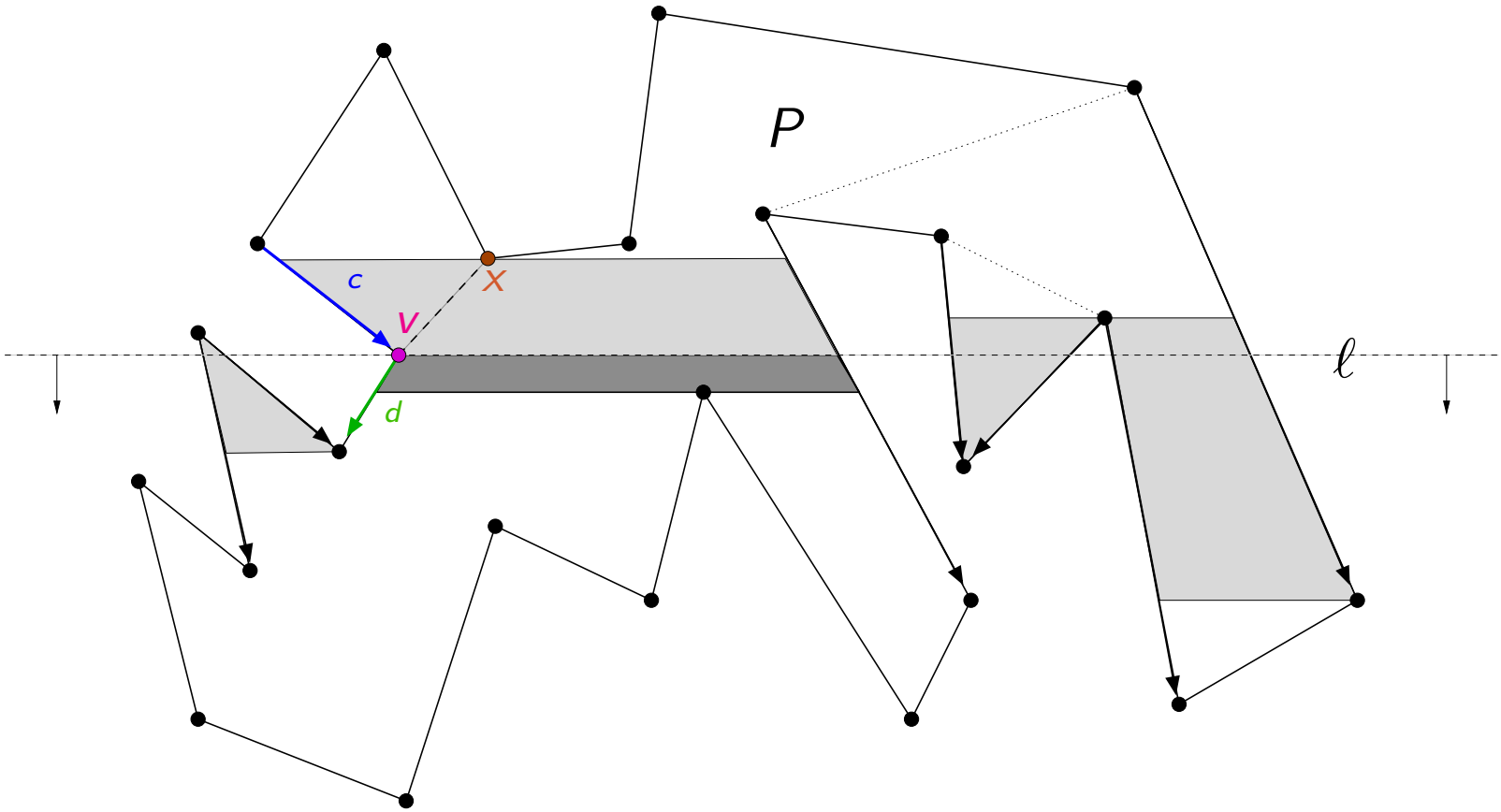
Remova o trapézio (c, x, e) ou (e, x, c)
e insira o trapézio (d, v, e) ou (e, v, d) em T.



Caso 1

Caso 1. Aresta c está acima de ℓ e d abaixo

Remova o trapézio (c, x, e) ou (e, x, c)
e insira o trapézio (d, v, e) ou (e, v, d) em T.

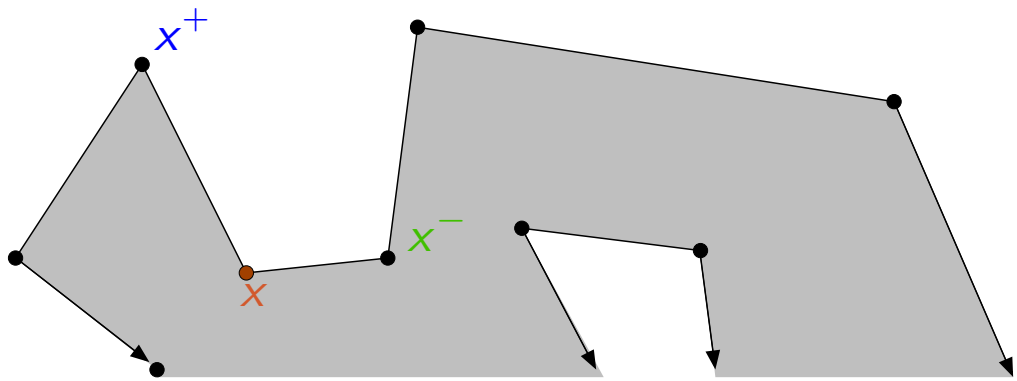


Se x for ponta para baixo, acrescente a diagonal (x, v).

Teste de ponta para baixo

PontaParaBaixo(x, Y, n)

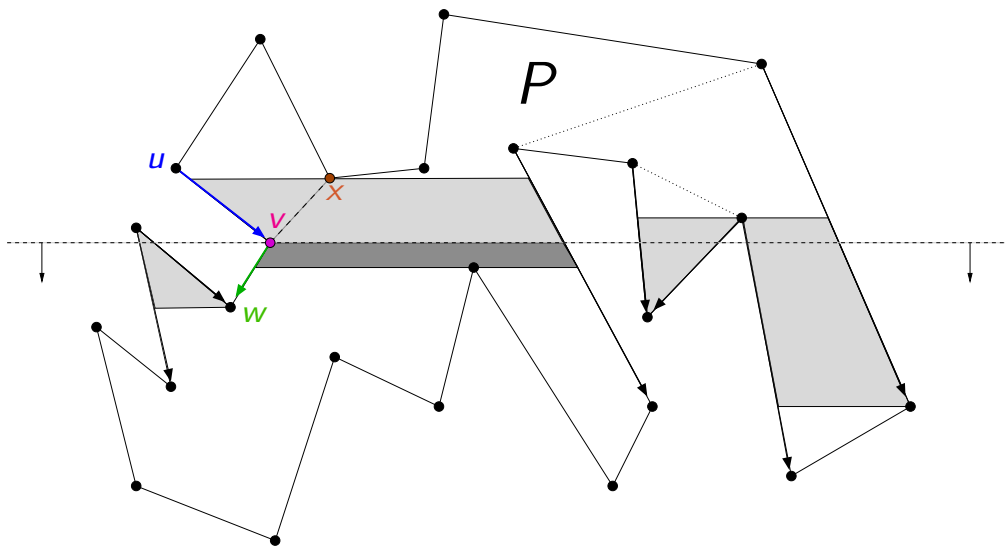
1 $x^- \leftarrow x - 1 \quad x^+ \leftarrow x + 1$
2 se $x^- = 0$ então $x^- \leftarrow n$
3 se $x^+ = n + 1$ então $x^+ \leftarrow 1$
▷ x^- e x^+ são o predecessor e o sucessor de x em δP
4 se $Y[x^-] > Y[x]$ e $Y[x^+] > Y[x]$
▷ x é ponta interior para baixo?
5 então devolva verdade
6 senão devolva falso



Caso 1

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

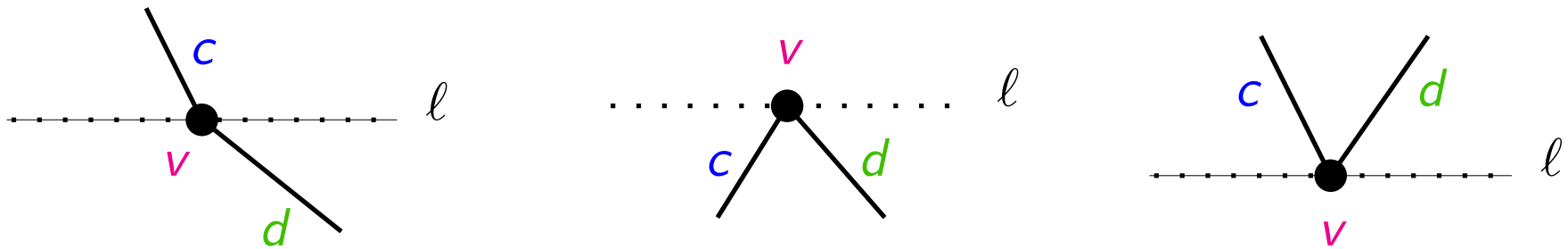
1 se $Y[u] < Y[w]$ então $u \leftrightarrow w$
2 $((i,j), x, (k,l)) \leftarrow$ Remove(T, v)
3 se $v = j$ ▷ o trapézio está à direita de v?
4 então Insere$(T, (v,w), v, (k,l))$
5 senão Insere$(T, (i,j), v, (v,w))$
6 se PontaParaBaixo(x, Y, n)
7 então $t \leftarrow t + 1$ $D[t] \leftarrow (x, v)$



Algoritmo de Lee e Preparata

T: ED da linha de varredura ℓ

Três casos a considerar:



Caso 1. Aresta c está acima de ℓ e d abaixo

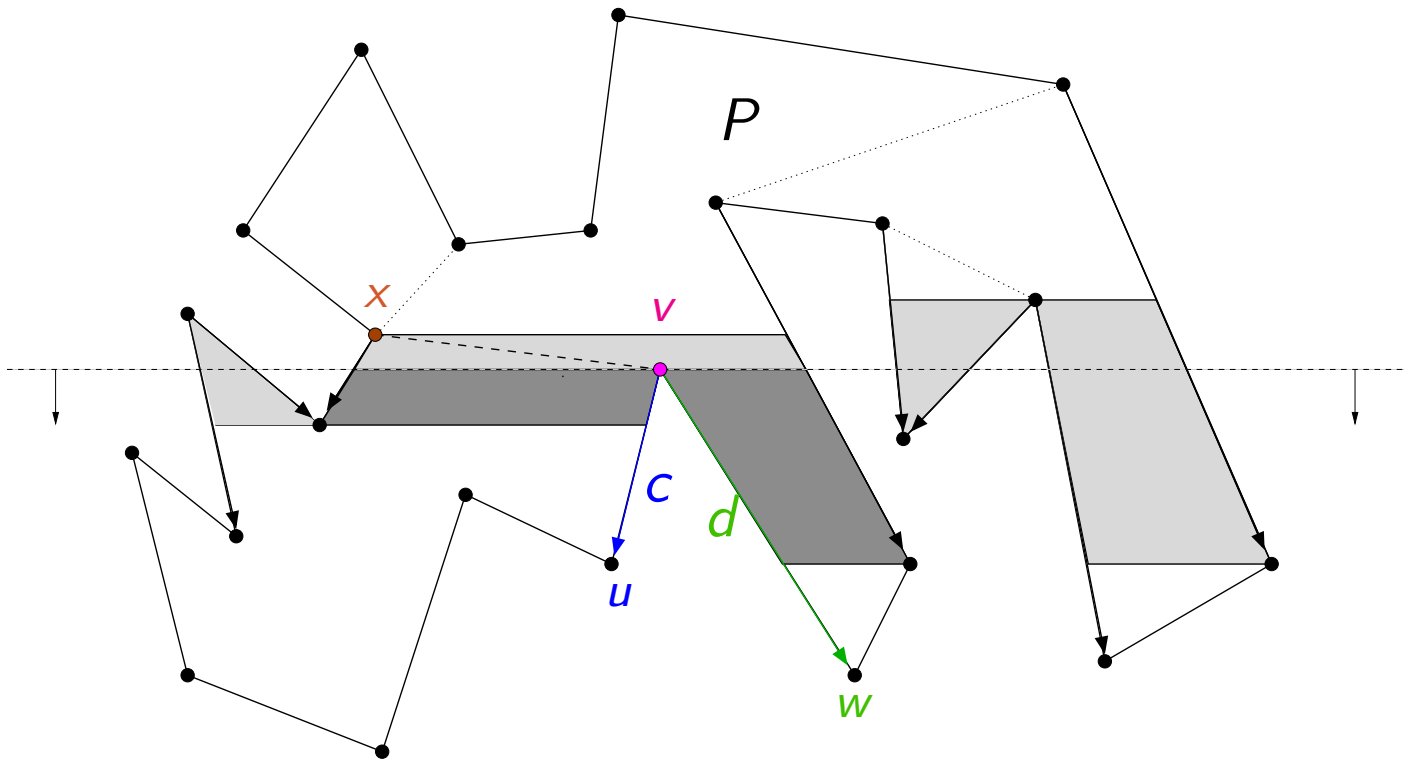
Caso 2. Arestas c e d estão abaixo de ℓ

Caso 3. Arestas c e d estão acima de ℓ

Caso 2

Caso 2. Arestas c e d estão abaixo de ℓ

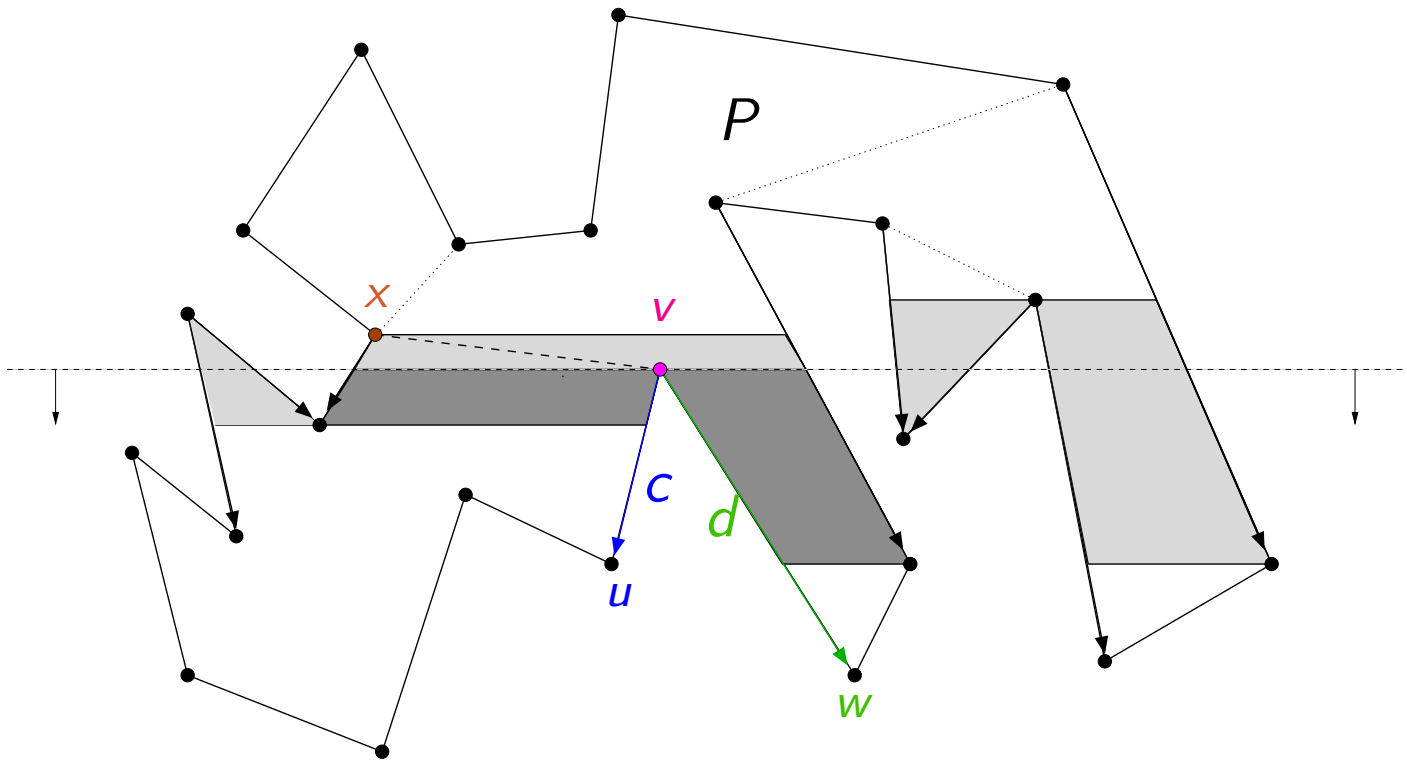
Se há trapézio com v em T,
então substitua-o por dois trapézios:
um com lado direito em c e outro com lado esquerdo em d.



Caso 2

Caso 2. Arestas c e d estão abaixo de ℓ

Se há trapézio com v em T,
então substitua-o por dois trapézios:
um com lado direito em c e outro com lado esquerdo em d.

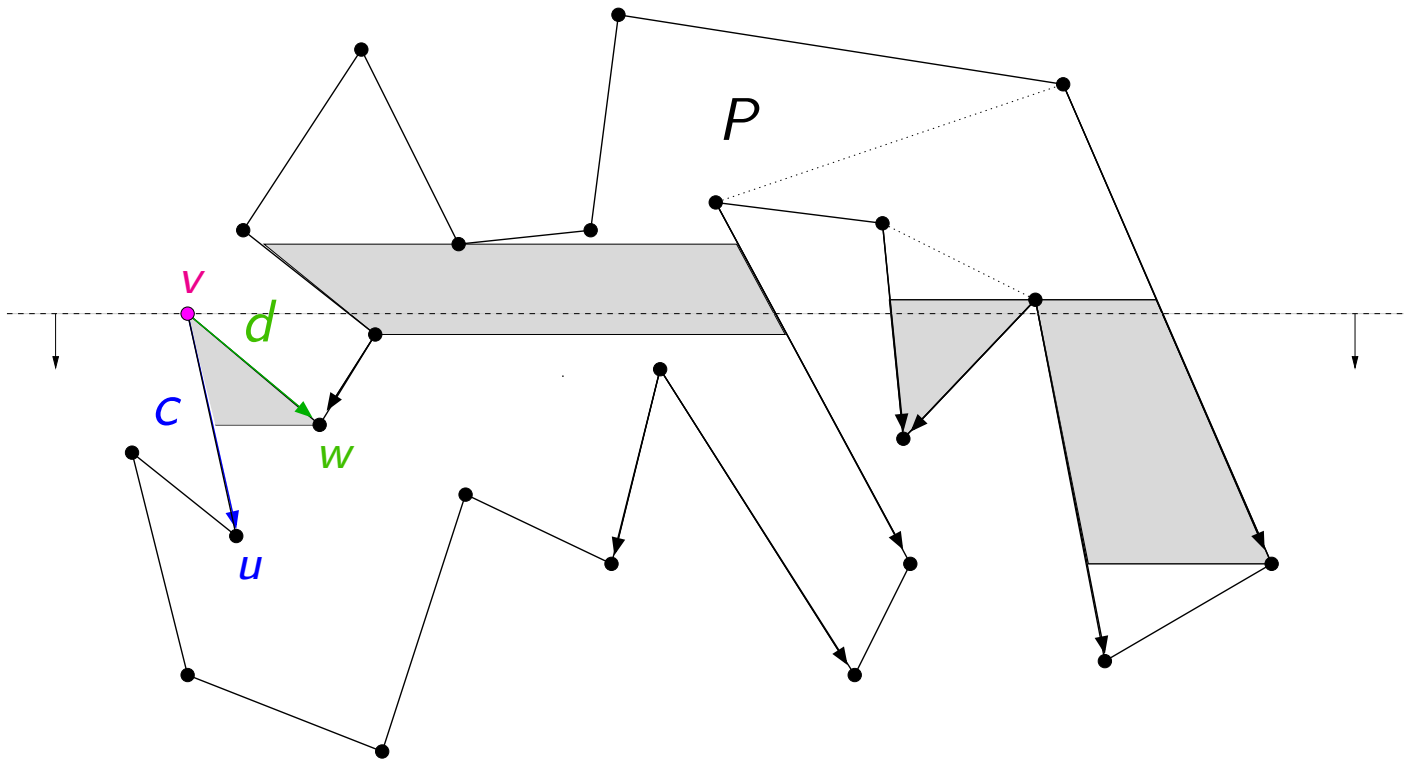


Acrescente a diagonal (x, v).

Caso 2

Caso 2. Arestas c e d estão abaixo de ℓ

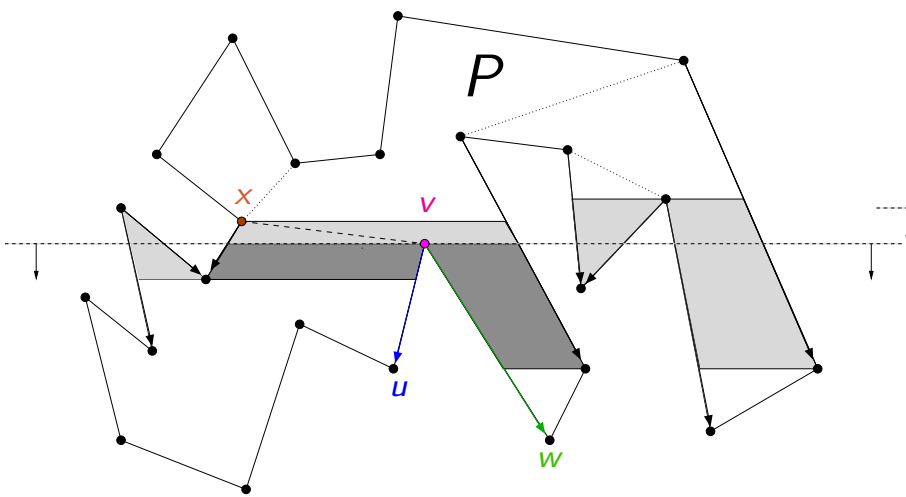
Se há trapézio com v em T,
então substitua-o por dois trapézios:
um com lado direito em c e outro com lado esquerdo em d.
Se não há trapézio com v em T, então insira (c, v, d) em T.

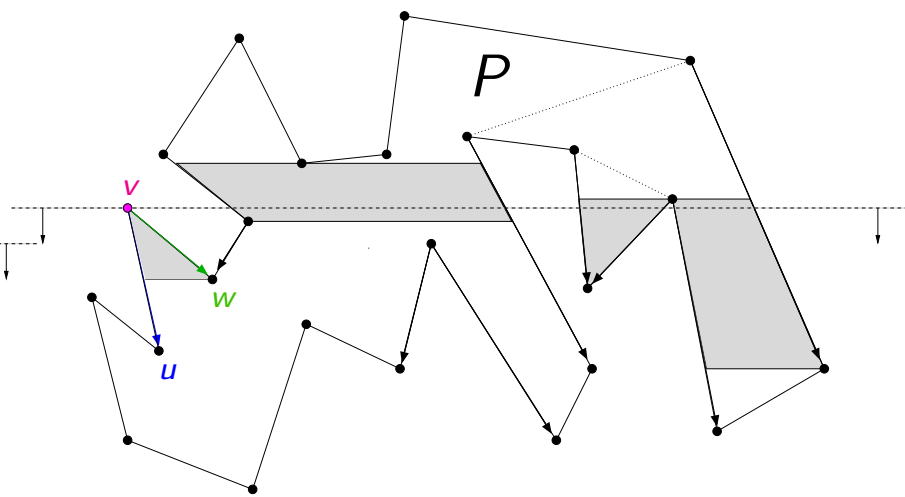


Caso 2

TrataCaso2 (T, u, v, w, D, t)

1 se Esquerda$^+(u, v, w)$ então $u \leftrightarrow w$
2 $trap \leftarrow$ Remove(T, v)
3 se $trap =$ NIL
4 então Insere$(T, (v, u), v, (v, w))$
5 senão $((i, j), x, (k, l)) \leftarrow trap$
6 Insere$(T, (i, j), v, (v, u))$
7 Insere$(T, (v, w), v, (k, l))$
8 $t \leftarrow t + 1$ $D[t] \leftarrow (x, v)$ ▷ v é ponta interior para cima

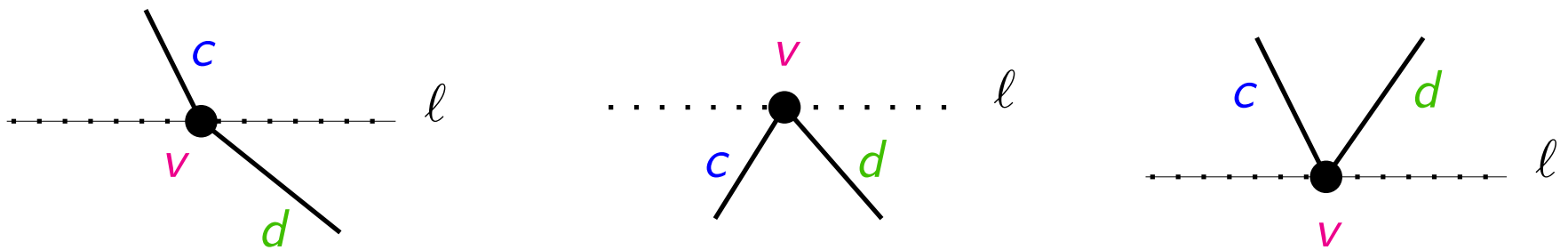




Algoritmo de Lee e Preparata

T: ED da linha de varredura ℓ

Três casos a considerar:



Caso 1. Aresta c está acima de ℓ e d abaixo

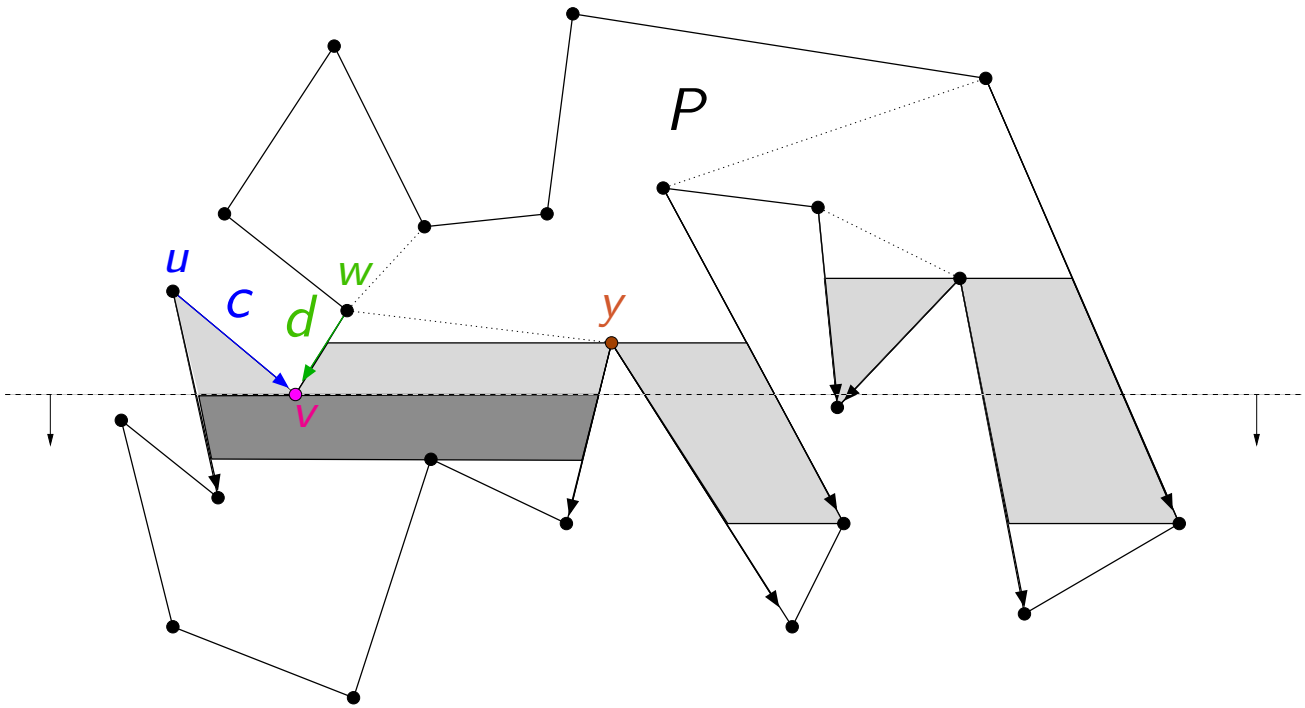
Caso 2. Arestas c e d estão abaixo de ℓ

Caso 3. Arestas c e d estão acima de ℓ

Caso 3

Caso 3. Arestas c e d estão acima de ℓ

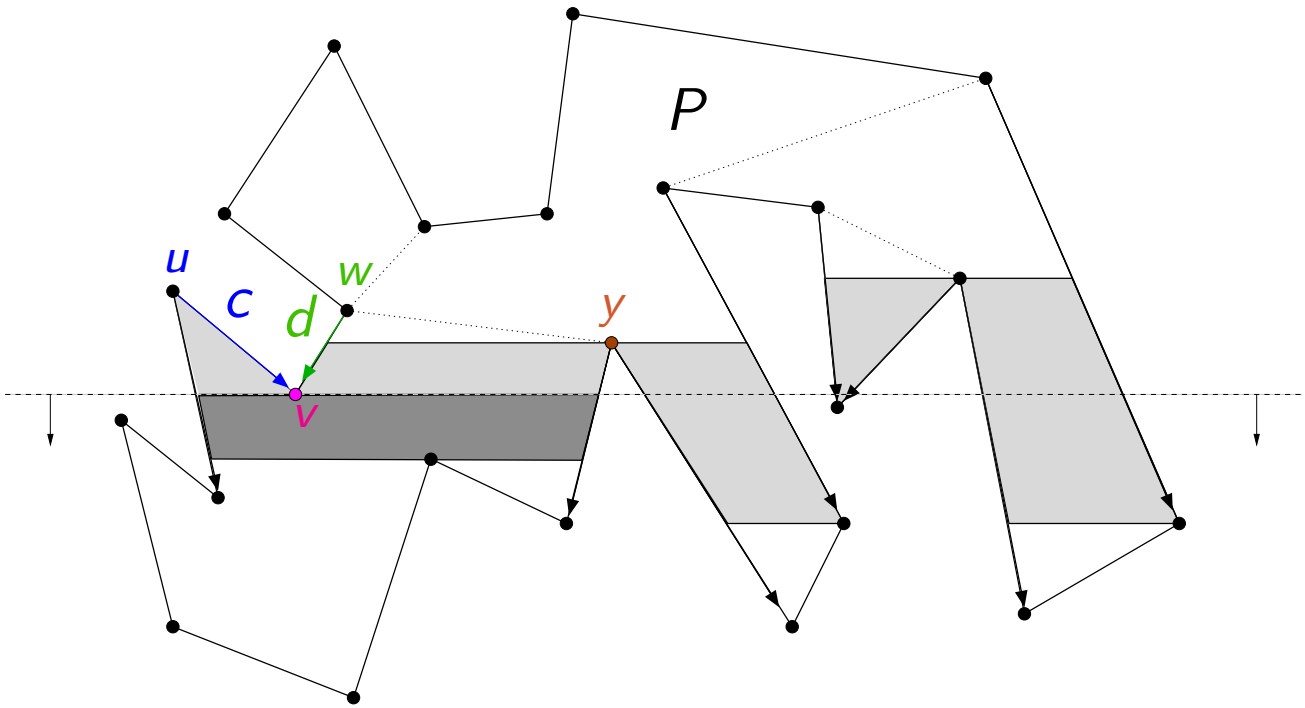
Pode haver um ou dois trapézios contento v em T.
Se houver dois, remova (e,x,c) e (d,y,f) de T e insira (e,v,f).



Caso 3

Caso 3. Arestas c e d estão acima de ℓ

Pode haver um ou dois trapézios contento v em T.
Se houver dois, remova (e,x,c) e (d,y,f) de T e insira (e,v,f).

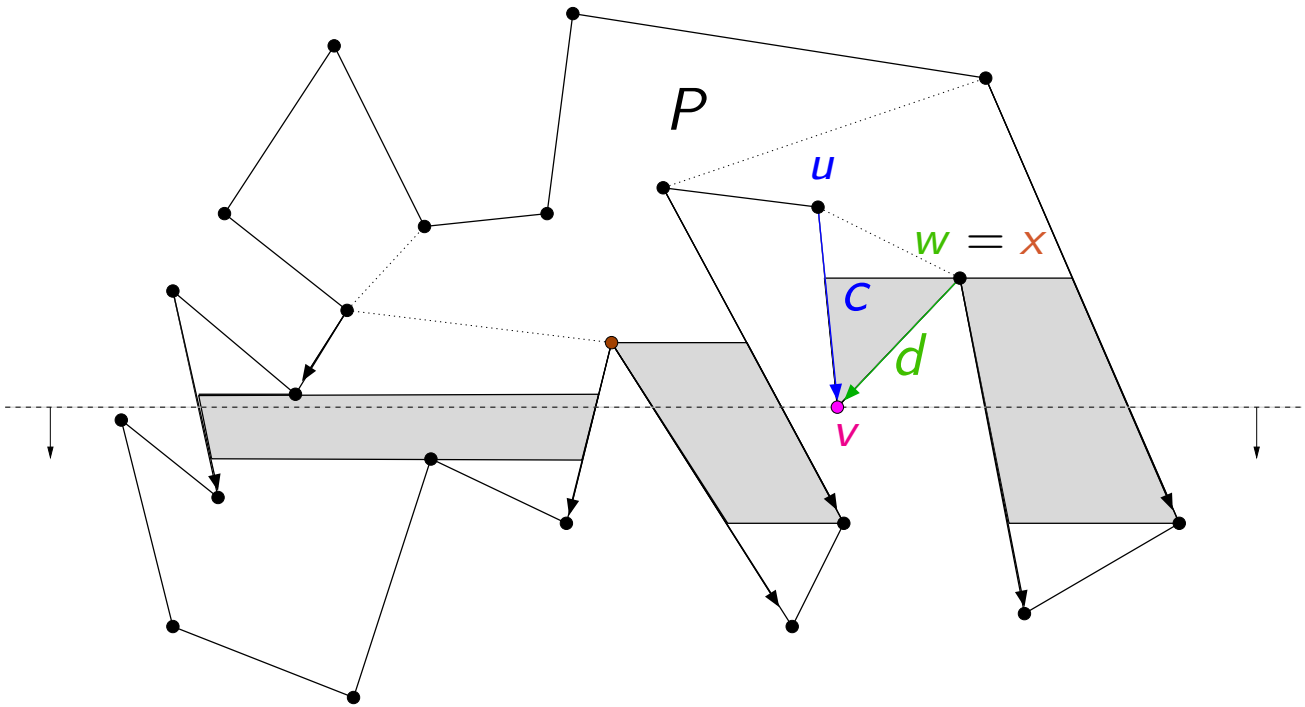


Se x e/ou y forem pontas para baixo,
então acrescente as diagonais (x, v) e/ou (y, v).

Caso 3

Caso 3. Arestas c e d estão acima de ℓ

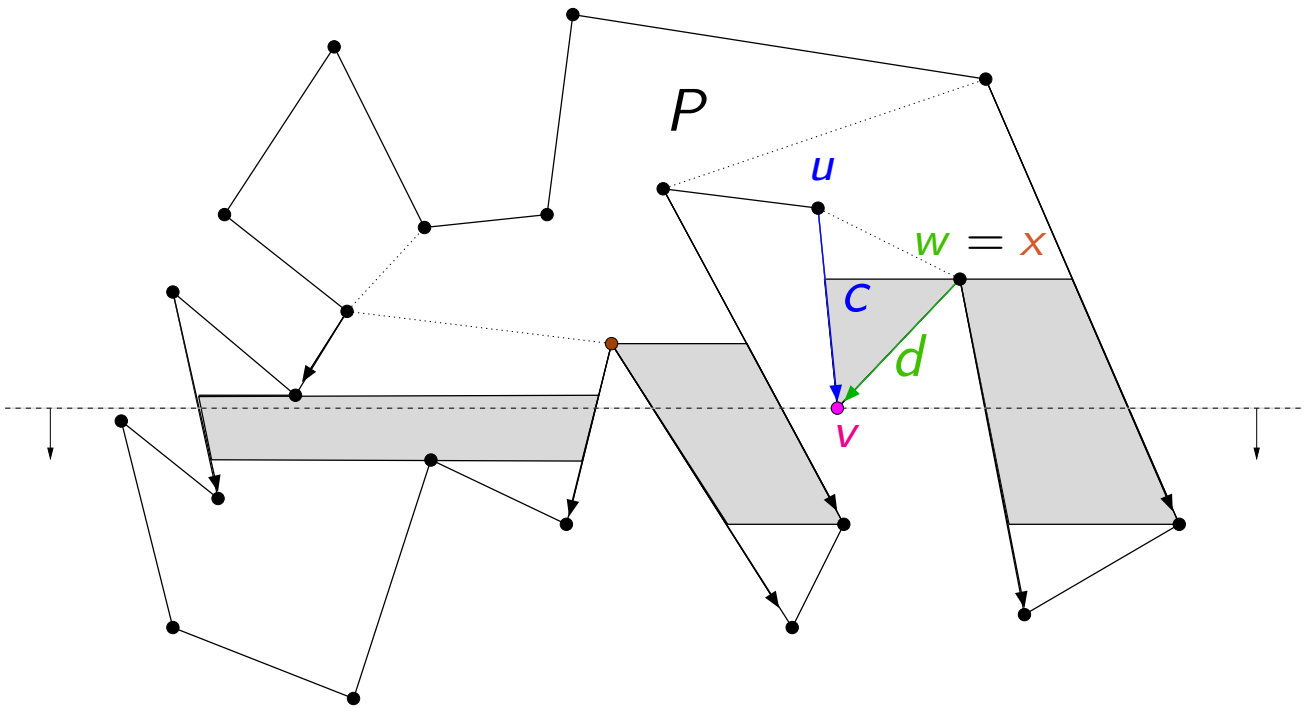
Pode haver um ou dois trapézios contento v em T.
Se houver dois, remova (e,x,c) e (d,y,f) de T e insira (e,v,f).
Se houver um, (c,x,e), remova-o.



Caso 3

Caso 3. Arestas c e d estão acima de ℓ

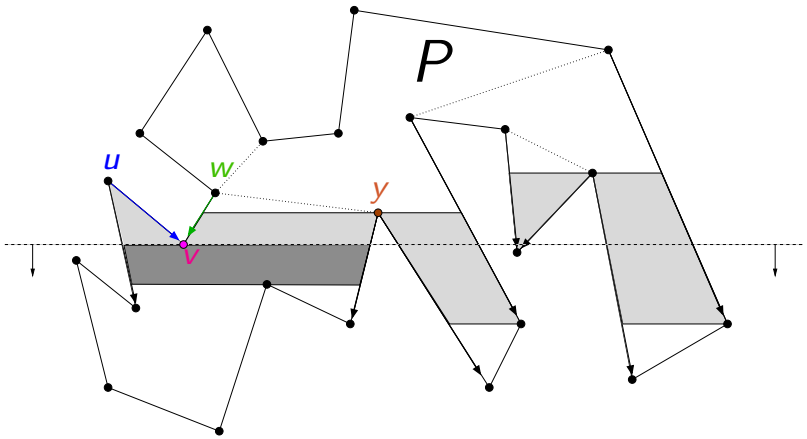
Pode haver um ou dois trapézios contento v em T.
Se houver dois, remova (e,x,c) e (d,y,f) de T e insira (e,v,f).
Se houver um, (c,x,e), remova-o.

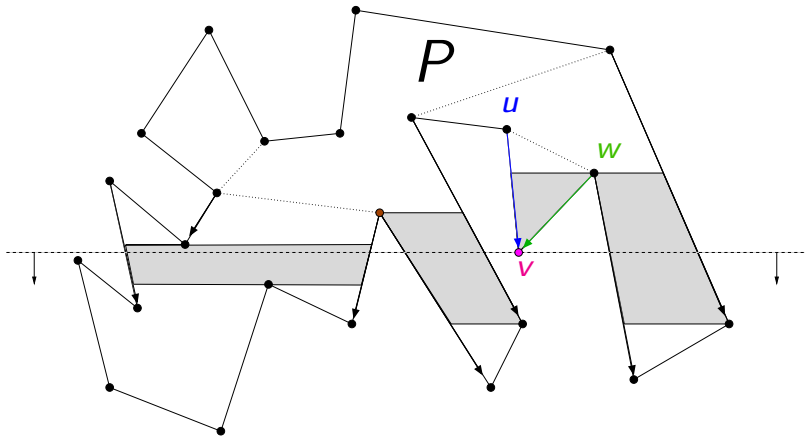


Se x for ponta para baixo, acrescente a diagonal (x, v).

Caso 3

```
TrataCaso3 (T, v, Y, n, D, t)
 1  ((i,j), x, (k,l)) ← Remove(T, v)
 2  se PontaParaBaixo(x, Y, n)
 3      então t ← t + 1    D[t] ← (x, v)
 4  se j ≠ v ou l ≠ v              ▷ há um outro trapézio em T?
 5      então ((i',j'), y, (k',l')) ← Remove(T, v)
 6            se PontaParaBaixo(y, Y, n)
 7                então t ← t + 1    D[t] ← (y, v)
 8            se l = v
 9                então Insere(i, j, v, k', l')
10                senão Insere(i', j', v, k, l)
```





Algoritmo de Lee e Preparata

DivideEmMonótono-LP(X, Y, n)

1 para $k \leftarrow 1$ até n faça
2 $\quad E[k] \leftarrow k$
3 MergeSort$(Y, X, 1, n, E)$ $\quad \triangleright$ ordenação indireta decrescente de Y
4 Crie(T) $\quad t \leftarrow 0$
5 para $k \leftarrow 1$ até n faça
6 $\quad v \leftarrow E[k]$
7 $\quad v^- \leftarrow v - 1 \quad v^+ \leftarrow v + 1$
8 $\quad$ se $v^- = 0$ então $v^- \leftarrow n$
9 $\quad$ se $v^+ = n + 1$ então $v^+ \leftarrow 1$
10 $\quad$ se $Y[v^-] < Y[v] < Y[v^+]$ ou $Y[v^+] < Y[v] < Y[v^-]$
11 $\quad\quad$ então TrataCaso1 $(T, v^-, v, v^+, Y, n, D, t)$
12 $\quad\quad$ senão se $Y[v^-] < Y[v]$
13 $\quad\quad\quad\quad$ então TrataCaso2 (T, v^-, v, v^+, D, t)
14 $\quad\quad\quad\quad$ senão TrataCaso3 (T, v, Y, n, D, t)
15 devolva (D, t)

Consumo de tempo

O tratamento de cada caso consome tempo $\mathrm{O}(\lg n)$.

Assim, o algoritmo DivideEmMonótono-LP
consome tempo $\mathrm{O}(\lg n)$ por iteração.

São n iterações, logo o consumo de tempo total é $\mathrm{O}(n \lg n)$.

Consumo de tempo

O tratamento de cada caso consome tempo $\mathrm{O}(\lg n)$.

Assim, o algoritmo DivideEmMonótono-LP
consome tempo $\mathrm{O}(\lg n)$ por iteração.

São n iterações, logo o consumo de tempo total é $\mathrm{O}(n \lg n)$.

Como se livrar da hipótese simplificadora?

Hipótese simplificadora:
não há dois vértices com a mesma y-coordenada.

Consumo de tempo

O tratamento de cada caso consome tempo $\mathrm{O}(\lg n)$.

Assim, o algoritmo DivideEmMonótono-LP
consome tempo $\mathrm{O}(\lg n)$ por iteração.

São n iterações, logo o consumo de tempo total é $\mathrm{O}(n \lg n)$.

Como se livrar da hipótese simplificadora?

Hipótese simplificadora:
não há dois vértices com a mesma y-coordenada.

Como acoplar isso com o algoritmo linear
para triangular polígonos monótonos?

Como se livrar da hipótese simplificadora?

Quando há vértices com a mesma y-coordenada,
é suficiente tratá-los da esquerda para a direita.

Como se livrar da hipótese simplificadora?

Quando há vértices com a mesma y-coordenada,
é suficiente tratá-los da esquerda para a direita.

Ou seja, considere acima um ponto com y-coordenada menor
ou com mesma y-coordenada e x-coordenada menor.

Considere abaixo um ponto com y-coordenada maior
ou com mesma y-coordenada e x-coordenada maior.

Como se livrar da hipótese simplificadora?

Quando há vértices com a mesma y-coordenada,
é suficiente tratá-los da esquerda para a direita.

Ou seja, considere acima um ponto com y-coordenada menor
ou com mesma y-coordenada e x-coordenada menor.

Considere abaixo um ponto com y-coordenada maior
ou com mesma y-coordenada e x-coordenada maior.

Exercício: Simule o algoritmo com essa
modificação no polígono árvore de Natal.

Como se livrar da hipótese simplificadora?

Quando há vértices com a mesma y-coordenada,
é suficiente tratá-los da esquerda para a direita.

Ou seja, considere acima um ponto com y-coordenada menor
ou com mesma y-coordenada e x-coordenada menor.

Considere abaixo um ponto com y-coordenada maior
ou com mesma y-coordenada e x-coordenada maior.

Exercício: Simule o algoritmo com essa
modificação no polígono árvore de Natal.

No algoritmo de triangulação de monótonos,
considere esta mesma ordem nos vértices.