Recorrido de una Subdivisión Quasi-Plana sin usar Bits de Marcado

Canek Peláez Valdés

29 de Junio de 2005

Recorrido de gráficas

Problema:

- Dada una gráfica G y un vértice inicial, recorrer la gráfica pasando por todos los vértices (o aristas).

Recorrido de gráficas

Problema:

- Dada una gráfica G y un vértice inicial, recorrer la gráfica pasando por todos los vértices (o aristas).

Si tenemos a toda la gráfica en memoria, existen varias soluciones clásicas (BFS, DFS, etc.), algoritmos que "marcan" los vértices para poder distinguir a los visitados de los no visitados.

Recorrido de gráficas

Problema:

- Dada una gráfica G y un vértice inicial, recorrer la gráfica pasando por todos los vértices (o aristas).

Si tenemos a toda la gráfica en memoria, existen varias soluciones clásicas (BFS, DFS, etc.), algoritmos que "marcan" los vértices para poder distinguir a los visitados de los no visitados.
¿Qué pasa si no queremos marcar los vértices?

Motivación

Redes inalámbricas ad hoc:

Motivación

Redes inalámbricas ad hoc:

- cada dispositivo conoce su ubicación (coordenadas (x, y)) y la de sus vecinos,

Motivación

Redes inalámbricas ad hoc:

- cada dispositivo conoce su ubicación (coordenadas (x, y)) y la de sus vecinos,
- dispositivos con poca memoria y poco poder,

Motivación

Redes inalámbricas ad hoc:

- cada dispositivo conoce su ubicación (coordenadas (x, y)) y la de sus vecinos,
- dispositivos con poca memoria y poco poder,
- dispositivos en la red pueden desaparecer y aparecer de forma altamente dinámica.

Motivación

Redes inalámbricas ad hoc:

- cada dispositivo conoce su ubicación (coordenadas (x, y)) y la de sus vecinos,
- dispositivos con poca memoria y poco poder,
- dispositivos en la red pueden desaparecer y aparecer de forma altamente dinámica.

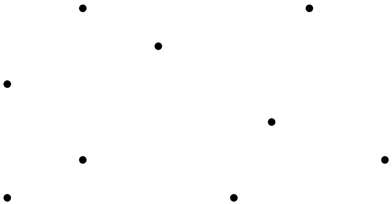
¿Cómo hacemos *broadcasting* en este tipo de escenarios?

Subvisiones Planas

Una *subdivisión plana* es una partición del plano $\mathbf{E}^2$

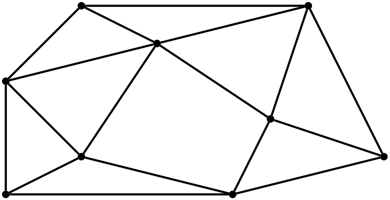
Subvisiones Planas

Una *subdivisión plana* es una partición del plano $\mathbb{E}^2$ en un conjunto V de vértices



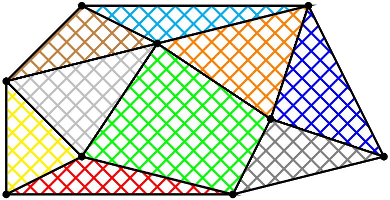
Subvisiones Planas

Una *subdivisión plana* es una partición del plano $\mathbf{E}^2$ en un conjunto V de vértices, un conjunto E de aristas (segmentos de rectas)



Subvisiones Planas

Una *subdivisión plana* es una partición del plano $\mathbf{E}^2$ en un conjunto V de vértices, un conjunto E de aristas (segmentos de rectas), y un conjunto F de caras (polígonos).



Recorridos en Subdivisiones Planas

Existen varios algoritmos para hacer recorridos en subdivisiones planes: se consiguió primero para subdivisiones triangulares [1], después para politopos convexos [2], y finalmente para subdivisiones planas arbitrarias (complejidad $O(n \log n)$, donde $|V| = n$) [3, 4].

Recorridos en Subdivisiones Planas

Existen varios algoritmos para hacer recorridos en subdivisiones planes: se consiguió primero para subdivisiones triangulares [1], después para politopos convexos [2], y finalmente para subdivisiones planas arbitrarias (complejidad $O(n \log n)$, donde $|V| = n$) [3, 4].
Todos estos algoritmos utilizan propiedades geométricas de las subdivisiones planas.

Recorridos en Subdivisiones Planas

Existen varios algoritmos para hacer recorridos en subdivisiones planes: se consiguió primero para subdivisiones triangulares [1], después para politopos convexos [2], y finalmente para subdivisiones planas arbitrarias (complejidad $O(n \log n)$, donde $|V| = n$) [3, 4].
Todos estos algoritmos utilizan propiedades geométricas de las subdivisiones planas.
Se pueden generalizar varias técnicas de estos algoritmos para una gran familia de gráficas que no representan subdivisiones planas de $\mathbf{E}^2$.

Ejemplo

Veamos el ejemplo de un problema cuya gráfica se quiere recorrer, pero que no es completamente plana.

Ejemplo

Veamos el ejemplo de un problema cuya gráfica se quiere recorrer, pero que no es completamente plana.

¡Búsqueda del tesoro!

Ejemplo

Veamos el ejemplo de un problema cuya gráfica se quiere recorrer, pero que no es completamente plana.

¡Búsqueda del tesoro! 

Se deja a un pirata con un GPS en algún punto de la ciudad de México, y se le dice que hay un tesoro en *alguna* calle de la ciudad. El pirata no tiene de otra más que recorrer *todas* las calles, y no puede dejar marcas.

Ejemplo

Veamos el ejemplo de un problema cuya gráfica se quiere recorrer, pero que no es completamente plana.

¡Búsqueda del tesoro! 

Se deja a un pirata con un GPS en algún punto de la ciudad de México, y se le dice que hay un tesoro en *alguna* calle de la ciudad. El pirata no tiene de otra más que recorrer *todas* las calles, y no puede dejar marcas.
Lamentablemente (para el pirata), la ciudad de México tiene cosas como esta:

Ejemplo

Veamos el ejemplo de un problema cuya gráfica se quiere recorrer, pero que no es completamente plana.

¡Búsqueda del tesoro! 

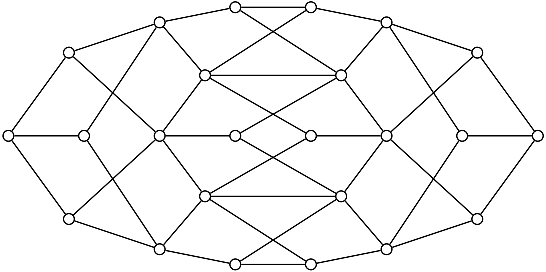
Se deja a un pirata con un GPS en algún punto de la ciudad de México, y se le dice que hay un tesoro en *alguna* calle de la ciudad. El pirata no tiene de otra más que recorrer *todas* las calles, y no puede dejar marcas.
Lamentablemente (para el pirata), la ciudad de México tiene cosas como esta:

Objetivo

Queremos extender nuestras técnicas de subdivisiones planas a subdivisiones no planas cuyas aristas tengan (intuitivamente) "pocas" intersecciones (o al menos un número manejable). Para ello, vamos a definir lo que llamaremos *subdivisiones quasi-planas*.

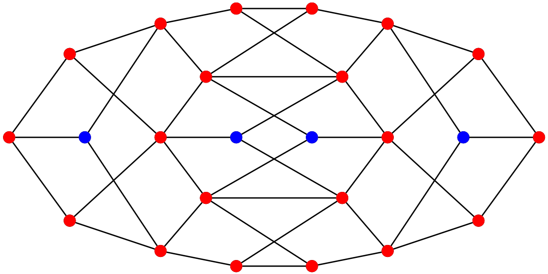
Subdivisión quasi-plana

Una *subdivisión quasi-plana* es una gráfica $G = (V, E)$ con vértices en el plano



Subdivisión quasi-plana

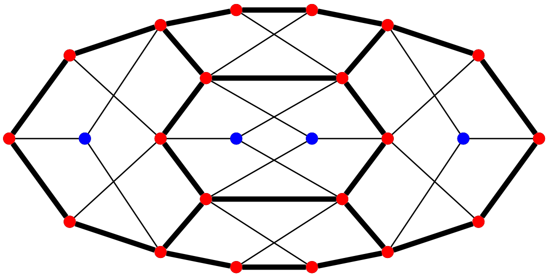
Una *subdivisión quasi-plana* es una gráfica $G = (V, E)$ con vértices en el plano, y particionada en $V_p \cup V_c = V$ de tal forma que:



Subdivisión quasi-plana

Una *subdivisión quasi-plana* es una gráfica $G = (V, E)$ con vértices en el plano, y particionada en $V_p \cup V_c = V$ de tal forma que:

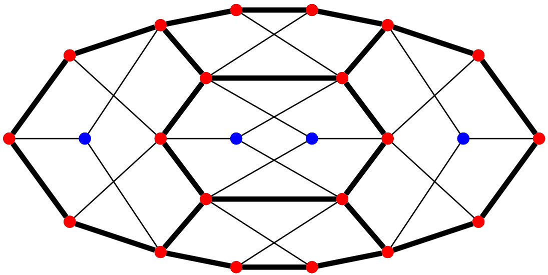
- los vértices en V_p inducen una gráfica plana P conectada,



Subdivisión quasi-plana

Una *subdivisión quasi-plana* es una gráfica $G = (V, E)$ con vértices en el plano, y particionada en $V_p \cup V_c = V$ de tal forma que:

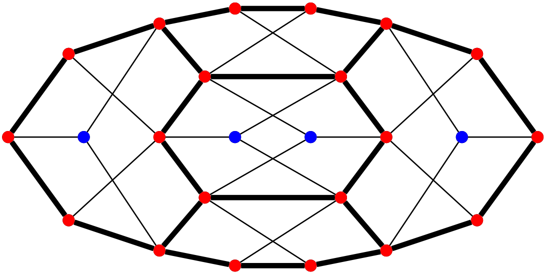
- los vértices en V_p inducen una gráfica plana P conectada,
- la cara externa de P no contiene ningún vértice de V_c o arista de $G - P$, y



Subdivisión quasi-plana

Una *subdivisión quasi-plana* es una gráfica $G = (V, E)$ con vértices en el plano, y particionada en $V_p \cup V_c = V$ de tal forma que:

- los vértices en V_p inducen una gráfica plana P conectada,
- la cara externa de P no contiene ningún vértice de V_c o arista de $G - P$, y
- ninguna arista de P es cruzada por ninguna otra arista de G.



Propiedades de las subdivisiones quasi-planas

Nos referiremos a la gráfica P como la *subgráfica plana subyacente*, y a las caras de P como *caras subyacentes*.

Propiedades de las subdivisiones quasi-planas

Nos referiremos a la gráfica P como la *subgráfica plana subyacente*, y a las caras de P como *caras subyacentes*.

Hacemos notar que el algoritmo no necesita saber de la partición $V = V_p \cup V_c$; ésta sólo la usamos para demostrar que funciona.

Propiedades de las subdivisiones quasi-planas

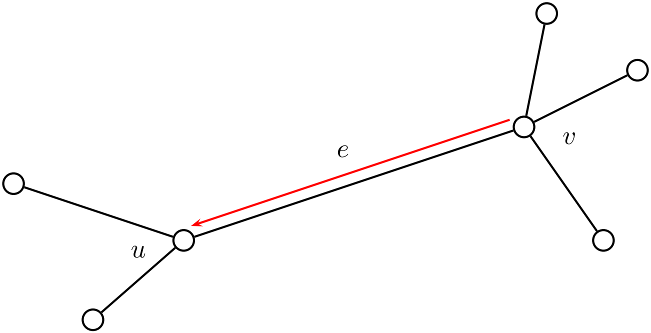
Nos referiremos a la gráfica P como la *subgráfica plana subyacente*, y a las caras de P como *caras subyacentes*.

Hacemos notar que el algoritmo no necesita saber de la partición $V = V_p \cup V_c$; ésta sólo la usamos para demostrar que funciona.

Para poder extender la noción de cara en subdivisiones quasi-planas, necesitamos definir ciertas funciones básicas.

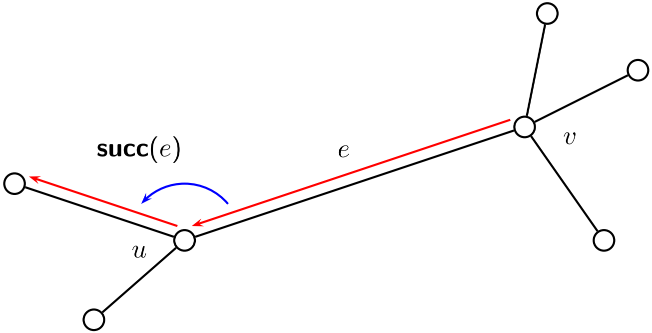
Funciones básicas de subdivisiones quasi-planas

Dada una arista dirigida e, definimos las siguientes funciones:



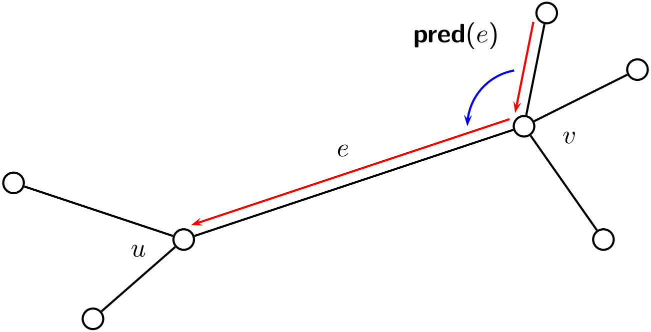
Funciones básicas de subdivisiones quasi-planas

Dada una arista dirigida e, definimos las siguientes funciones:



Funciones básicas de subdivisiones quasi-planas

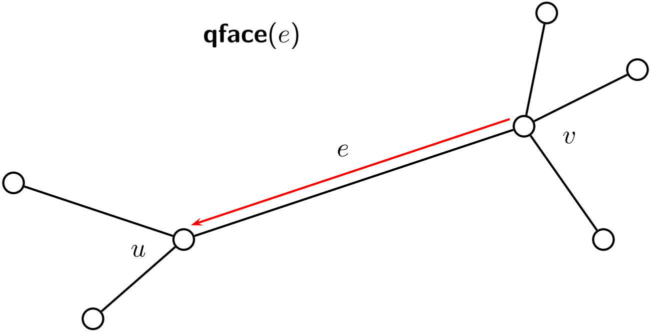
Dada una arista dirigida e, definimos las siguientes funciones:



Funciones básicas de subdivisiones quasi-planas

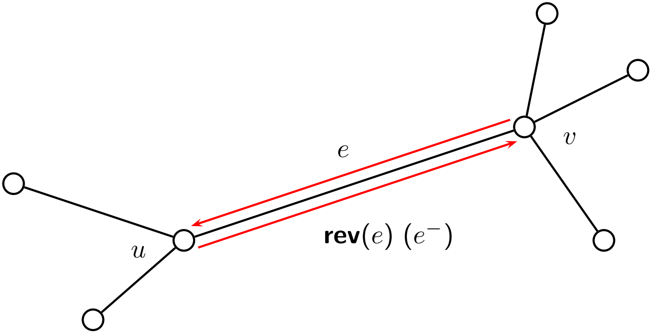
Dada una arista dirigida e, definimos las siguientes funciones:

qface(e)



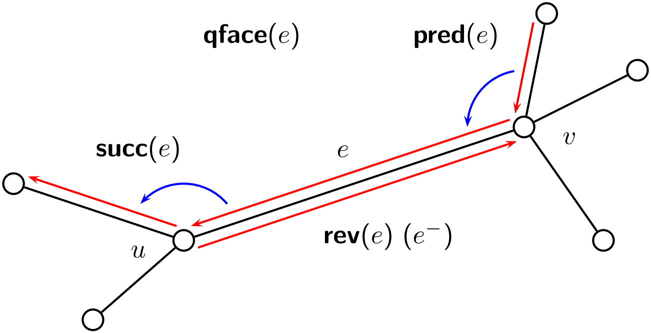
Funciones básicas de subdivisiones quasi-planas

Dada una arista dirigida e, definimos las siguientes funciones:



Funciones básicas de subdivisiones quasi-planas

Dada una arista dirigida e, definimos las siguientes funciones:

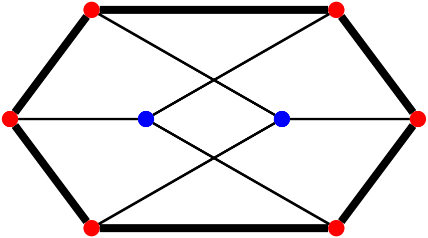


Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.

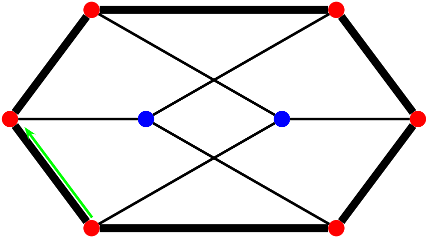
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



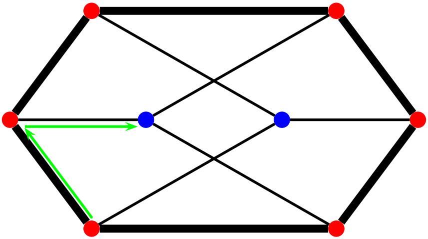
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



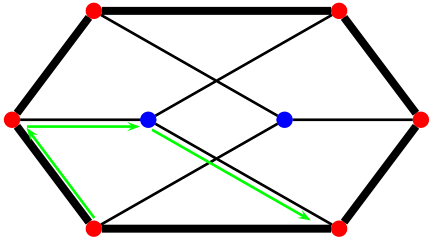
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



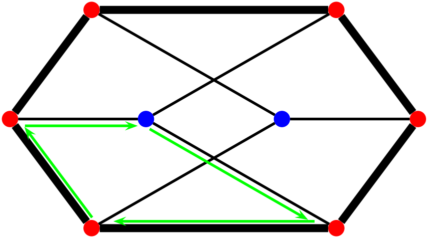
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



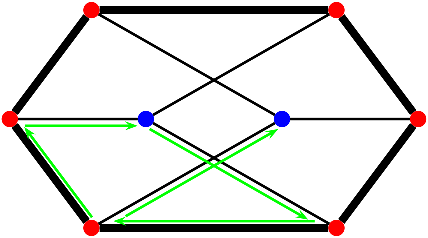
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



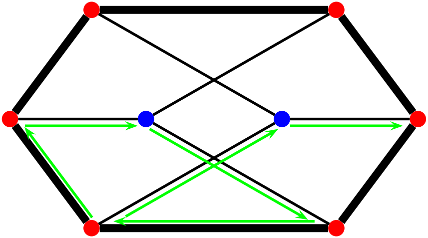
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



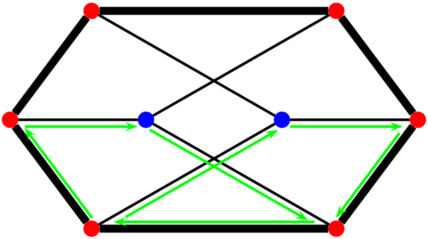
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



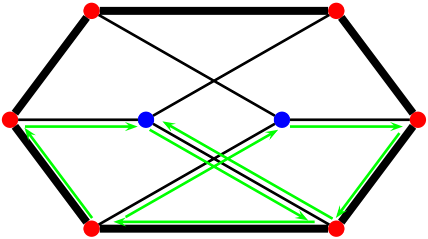
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



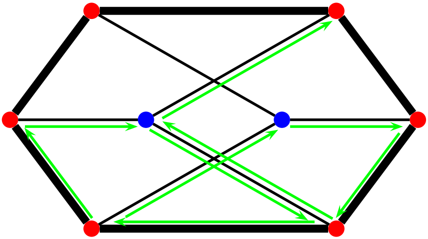
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



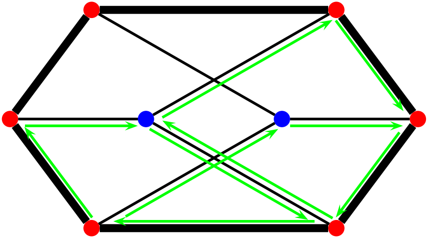
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



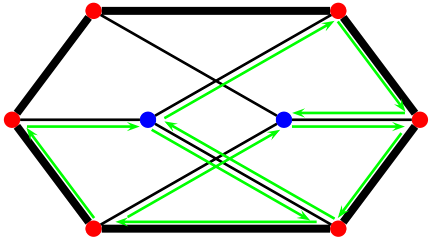
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



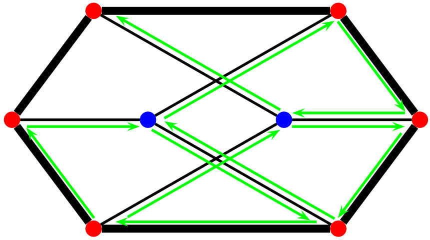
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



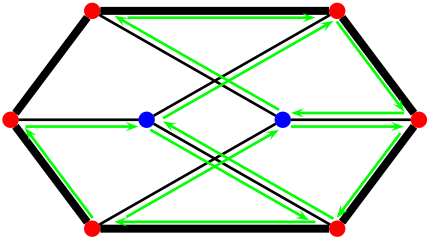
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



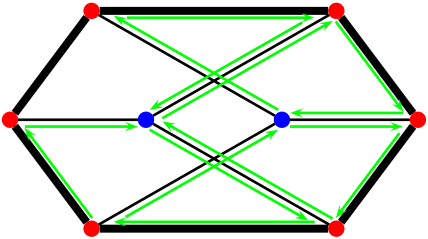
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



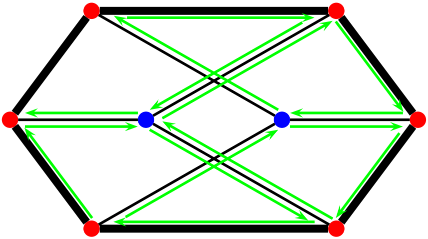
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



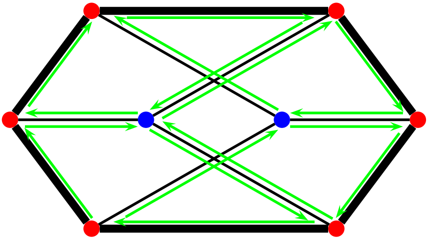
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



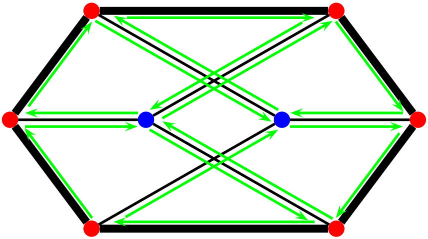
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



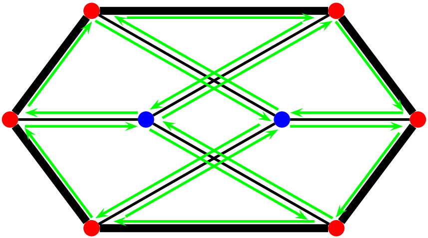
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



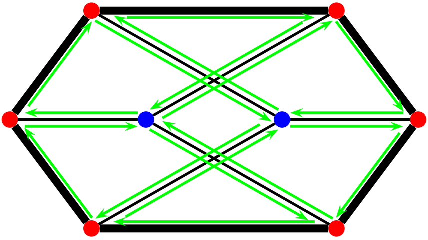
Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



Quasi-caras

Como **succ** y **pred** son inyectivas, podemos definir un camino cerrado para cada arista e de G, aplicando repetidamente la función **succ** hasta obtener e de nuevo. A este camino le llamamos una *quasi-cara* de G, que es lo que nos regresa **qface**(e). F es el conjunto de todas las quasi-caras.



Si G originalmente era una subvidision **plana**, entonces una quasi-cara se convierte en una cara regular.

Regla del Vecino Izquierdo

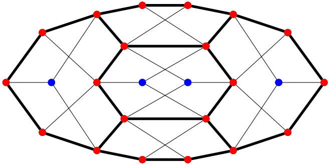
Queremos un algoritmo que recorra una subdivisión quasi-plana, de tal forma que se reporte cada vértice, arista (no dirigida) y quasi-cara *exactamente* una vez, en algún orden, y sin dejar marcas en los vértices. Sorprendentemente, se puede en *todas* las subdivisiones quasi-planas que cumplan una sencilla condición geométrica.

Regla del Vecino Izquierdo

Queremos un algoritmo que recorra una subdivisión quasi-plana, de tal forma que se reporte cada vértice, arista (no dirigida) y quasi-cara *exactamente* una vez, en algún orden, y sin dejar marcas en los vértices. Sorprendentemente, se puede en *todas* las subdivisiones quasi-planas que cumplan una sencilla condición geométrica.

Definición 1

Decimos que una subdivisión quasi-plana G satisface la Regla del Vecino Izquierdo si todo vértice $v \in V_c$ tiene un vecino u tal que u aparece a la izquierda de v.

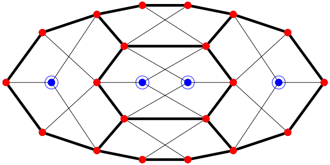


Regla del Vecino Izquierdo

Queremos un algoritmo que recorra una subdivisión quasi-plana, de tal forma que se reporte cada vértice, arista (no dirigida) y quasi-cara *exactamente* una vez, en algún orden, y sin dejar marcas en los vértices. Sorprendentemente, se puede en *todas* las subdivisiones quasi-planas que cumplan una sencilla condición geométrica.

Definición 1

Decimos que una subdivisión quasi-plana G satisface la Regla del Vecino Izquierdo si todo vértice $v \in V_c$ tiene un vecino u tal que u aparece a la izquierda de v.

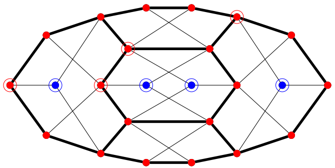


Regla del Vecino Izquierdo

Queremos un algoritmo que recorra una subdivisión quasi-plana, de tal forma que se reporte cada vértice, arista (no dirigida) y quasi-cara *exactamente* una vez, en algún orden, y sin dejar marcas en los vértices. Sorprendentemente, se puede en *todas* las subdivisiones quasi-planas que cumplan una sencilla condición geométrica.

Definición 1

Decimos que una subdivisión quasi-plana G satisface la Regla del Vecino Izquierdo si todo vértice $v \in V_c$ tiene un vecino u tal que u aparece a la izquierda de v.



El orden $\preceq$

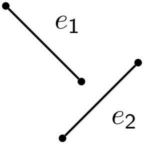
Dada una arista e, definimos $\mathbf{l}(e)$ como el vértice de e más a la izquierda (o más hacia abajo, si ambos vértices tienen la misma coordenada x).

Dadas $e_1, e_2 \in E$, se define el orden total $\preceq$ de la siguiente manera; $e_1 \preceq e_2$ si (lo primero que se cumpla):

El orden $\preceq$

Dada una arista e, definimos $\mathbf{l}(e)$ como el vértice de e más a la izquierda (o más hacia abajo, si ambos vértices tienen la misma coordenada x).
Dadas $e_1, e_2 \in E$, se define el orden total $\preceq$ de la siguiente manera; $e_1 \preceq e_2$ si (lo primero que se cumpla):

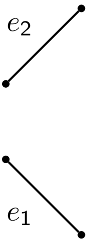
- la coordenada x de $\mathbf{l}(e_1)$ es menor que la coordenada x de $\mathbf{l}(e_2)$,



El orden $\preceq$

Dada una arista e, definimos $\mathbf{l}(e)$ como el vértice de e más a la izquierda (o más hacia abajo, si ambos vértices tienen la misma coordenada x).
Dadas $e_1, e_2 \in E$, se define el orden total $\preceq$ de la siguiente manera; $e_1 \preceq e_2$ si (lo primero que se cumpla):

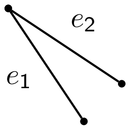
- la coordenada x de $\mathbf{l}(e_1)$ es menor que la coordenada x de $\mathbf{l}(e_2)$,
- la coordenada y de $\mathbf{l}(e_1)$ es menor que la coordenada y de $\mathbf{l}(e_2)$,



El orden $\preceq$

Dada una arista e, definimos $\mathbf{l}(e)$ como el vértice de e más a la izquierda (o más hacia abajo, si ambos vértices tienen la misma coordenada x).
Dadas $e_1, e_2 \in E$, se define el orden total $\preceq$ de la siguiente manera; $e_1 \preceq e_2$ si (lo primero que se cumpla):

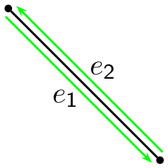
- la coordenada x de $\mathbf{l}(e_1)$ es menor que la coordenada x de $\mathbf{l}(e_2)$,
- la coordenada y de $\mathbf{l}(e_1)$ es menor que la coordenada y de $\mathbf{l}(e_2)$,
- la pendiente de e_1 es menor que la pendiente de e_2,



El orden $\preceq$

Dada una arista e, definimos $\mathbf{l}(e)$ como el vértice de e más a la izquierda (o más hacia abajo, si ambos vértices tienen la misma coordenada x).
Dadas $e_1, e_2 \in E$, se define el orden total $\preceq$ de la siguiente manera; $e_1 \preceq e_2$ si (lo primero que se cumpla):

- la coordenada x de $\mathbf{l}(e_1)$ es menor que la coordenada x de $\mathbf{l}(e_2)$,
- la coordenada y de $\mathbf{l}(e_1)$ es menor que la coordenada y de $\mathbf{l}(e_2)$,
- la pendiente de e_1 es menor que la pendiente de e_2,
- e_1 va a la "derecha" ("abajo")



Definiciones en quasi-caras

Dada f una quasi-cara de G, definimos:

Definiciones en quasi-caras

Dada f una quasi-cara de G, definimos:

- **entry**$(f) = e \in f : e \preceq e'$ para toda $e \neq e' \in f$,

Definiciones en quasi-caras

Dada f una quasi-cara de G, definimos:

- **entry**$(f) = e \in f : e \preceq e'$ para toda $e \neq e' \in f$,
- la función **ismin**(e), que regresará V si $e =$ **entry**(**qface**(e)) y $e^- =$ **entry**(**qface**(e^-)) y $e \preceq e^-$; y F en otro caso.

Definiciones en quasi-caras

Dada f una quasi-cara de G, definimos:

- **entry**$(f) = e \in f : e \preceq e'$ para toda $e \neq e' \in f$,
- la función **ismin**(e), que regresará V si $e =$ **entry**(**qface**(e)) y $e^- =$ **entry**(**qface**(e^-)) y $e \preceq e^-$; y F en otro caso.
- definimos $e_0 = (u_0, v_0)$ como la arista mínima en el orden $\preceq$.

Definiciones en quasi-caras

Dada f una quasi-cara de G, definimos:

- **entry**$(f) = e \in f : e \preceq e'$ para toda $e \neq e' \in f$,
- la función **ismin**(e), que regresará V si $e =$ **entry**(**qface**(e)) y $e^- =$ **entry**(**qface**(e^-)) y $e \preceq e^-$; y F en otro caso.
- definimos $e_0 = (u_0, v_0)$ como la arista mínima en el orden $\preceq$.

Lema 1

Si una subdivisión quasi-plana G satisface la Regla del Vecino Izquierdo, entonces la función **ismin**(e) *= V si y sólo si* $e = e_0$.

Definiciones en quasi-caras

El Lema 1 nos garantiza que podemos comprobar si una arista es e_0 usando sólo las funciones básicas. Esto nos permite definir la siguiente función con sólo $O(1)$ de memoria extra:

Definiciones en quasi-caras

El Lema 1 nos garantiza que podemos comprobar si una arista es e_0 usando sólo las funciones básicas. Esto nos permite definir la siguiente función con sólo $O(1)$ de memoria extra:

- **parent**(f) = **qface**(**rev**(**entry**(f))), si **entry**$(f) \neq e_0$, y NULL en otro caso.

Definiciones en quasi-caras

El Lema 1 nos garantiza que podemos comprobar si una arista es e_0 usando sólo las funciones básicas. Esto nos permite definir la siguiente función con sólo $O(1)$ de memoria extra:

- **parent**(f) = **qface**(**rev**(**entry**(f))), si **entry**$(f) \neq e_0$, y NULL en otro caso.

Definimos $G(F)$ como la gráfica $(F, E(F))$ donde

$$E(F) = \{(f,' f) \ : \textbf{parent}(f) = f'\}.$$

Definiciones en quasi-caras

El Lema 1 nos garantiza que podemos comprobar si una arista es e_0 usando sólo las funciones básicas. Esto nos permite definir la siguiente función con sólo $O(1)$ de memoria extra:

- **parent**(f) = **qface**(**rev**(**entry**(f))), si **entry**$(f) \neq e_0$, y NULL en otro caso.

Definimos $G(F)$ como la gráfica $(F, E(F))$ donde

$$E(F) = \{(f,' f) \ : \textbf{parent}(f) = f'\}.$$

Lema 2

Si una subdivisión quasi-plana G satisface la Regla del Vecino Izquierdo, entonces para cada quasi-cara f tal que **entry**$(f) \neq e_0$*, se cumple que* **entry**(**parent**$(f)) \prec$ **entry**(f).

Definiciones en quasi-caras

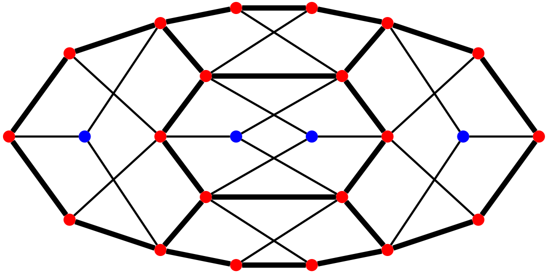
Corolario 1

Si una subdivisión quasi-plana de G satisface la Regla del Vecino Izquierdo, entonces para toda quasi-cara f se cumple que **parent**$(f) \neq f$.

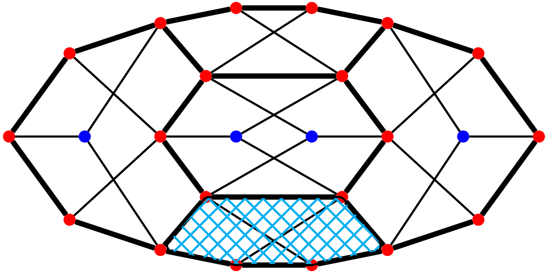
Teorema 1

Si una subdivisión quasi-plana G satisface la la Regla del Vecino Izquierdo, entonces la gráfica $G(F)$ es un árbol con raíz **qface**(e_0).

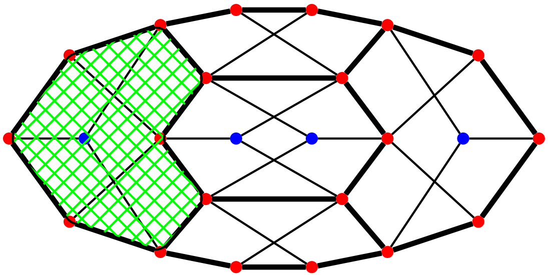
$G(F)$ es un árbol con raíz



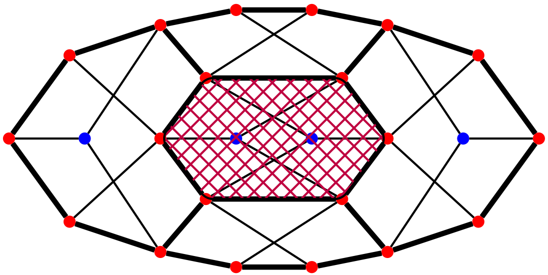
$G(F)$ es un árbol con raíz



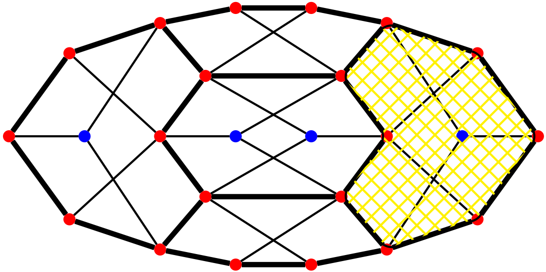
$G(F)$ es un árbol con raíz



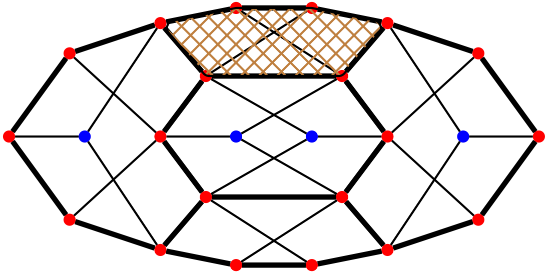
$G(F)$ es un árbol con raíz



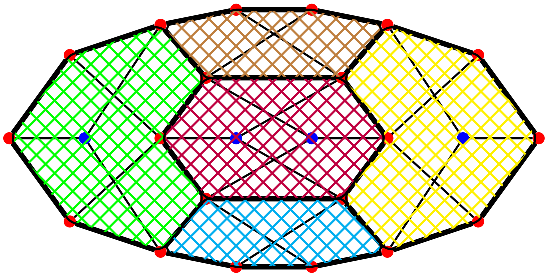
$G(F)$ es un árbol con raíz



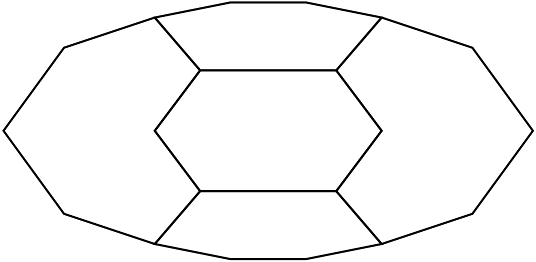
$G(F)$ es un árbol con raíz



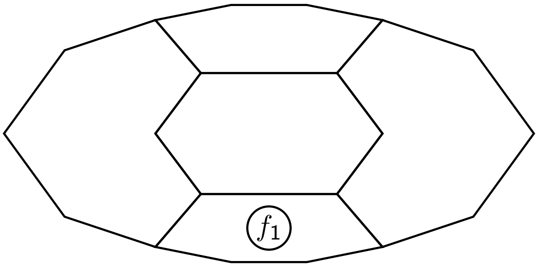
$G(F)$ es un árbol con raíz



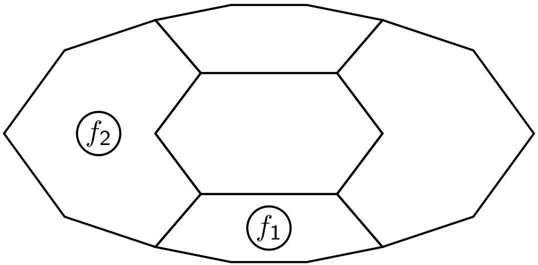
$G(F)$ es un árbol con raíz



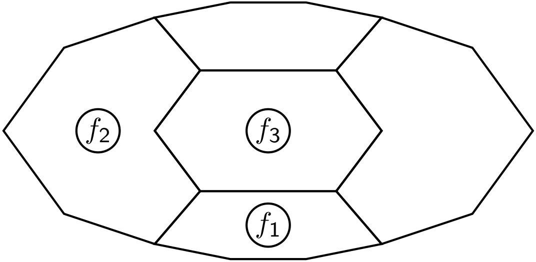
$G(F)$ es un árbol con raíz



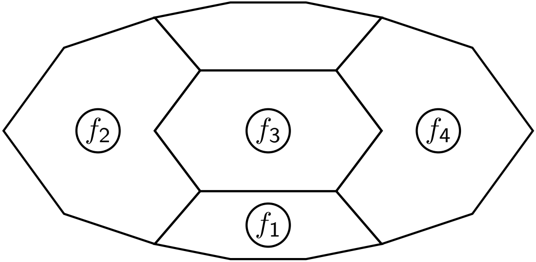
$G(F)$ es un árbol con raíz



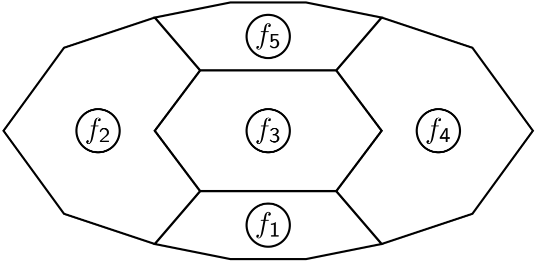
$G(F)$ es un árbol con raíz



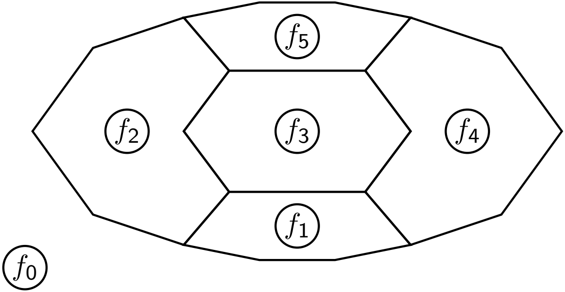
$G(F)$ es un árbol con raíz



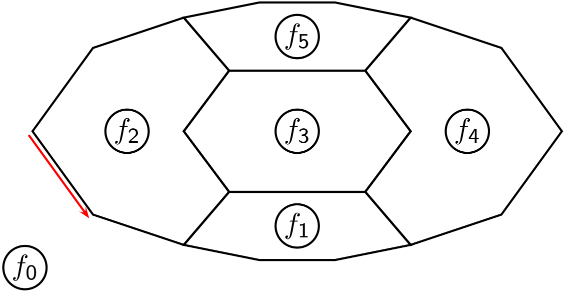
$G(F)$ es un árbol con raíz



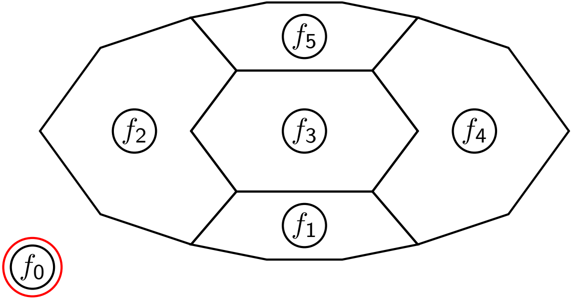
$G(F)$ es un árbol con raíz



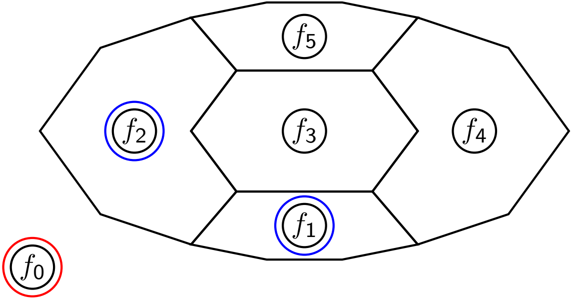
$G(F)$ es un árbol con raíz



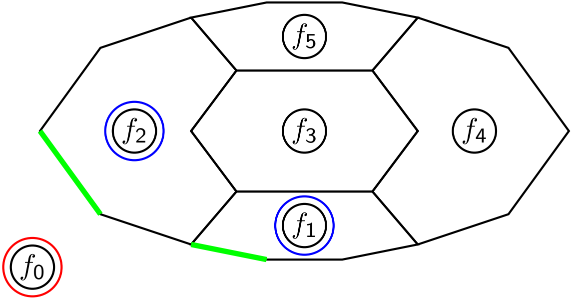
$G(F)$ es un árbol con raíz



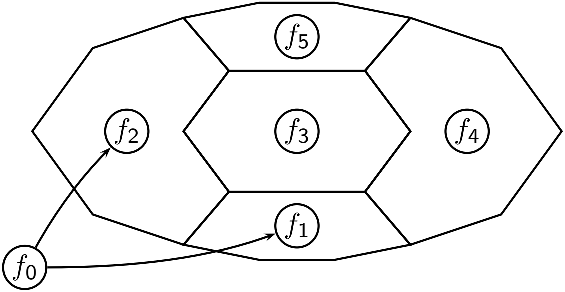
$G(F)$ es un árbol con raíz



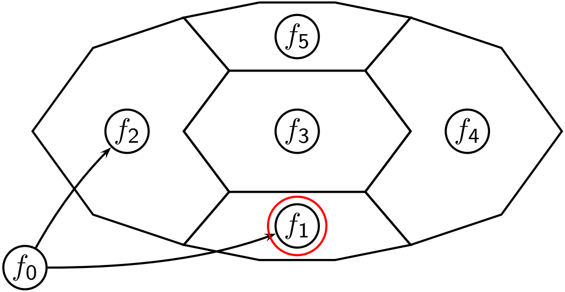
$G(F)$ es un árbol con raíz



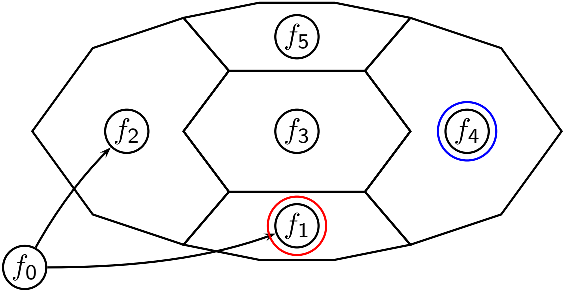
$G(F)$ es un árbol con raíz



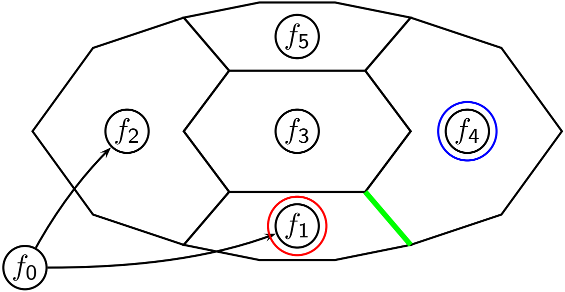
$G(F)$ es un árbol con raíz



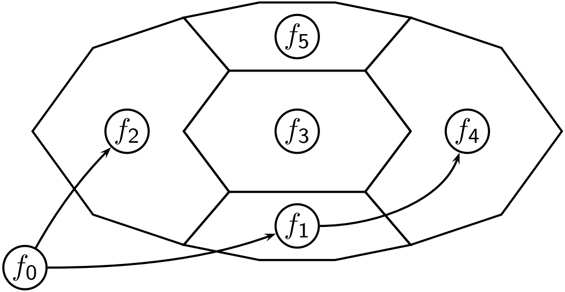
$G(F)$ es un árbol con raíz



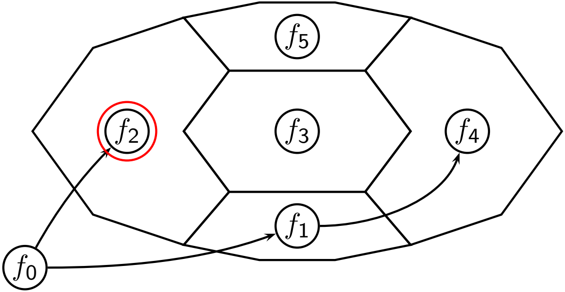
$G(F)$ es un árbol con raíz



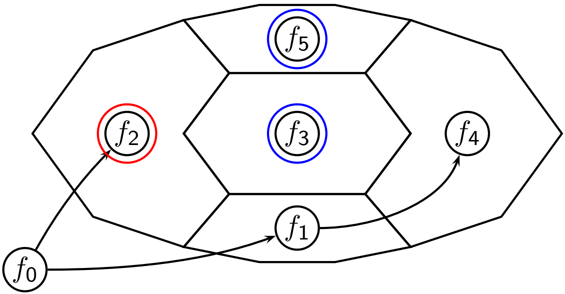
$G(F)$ es un árbol con raíz



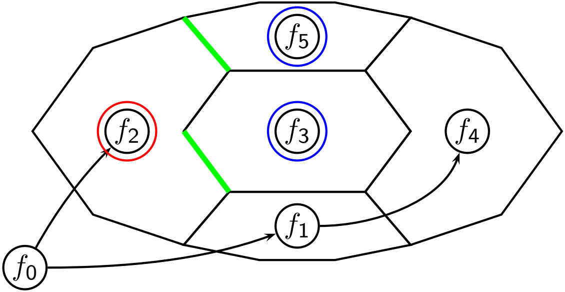
$G(F)$ es un árbol con raíz



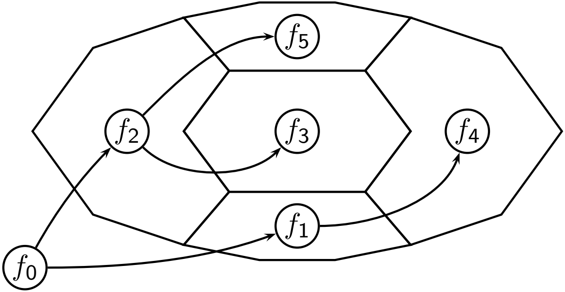
$G(F)$ es un árbol con raíz



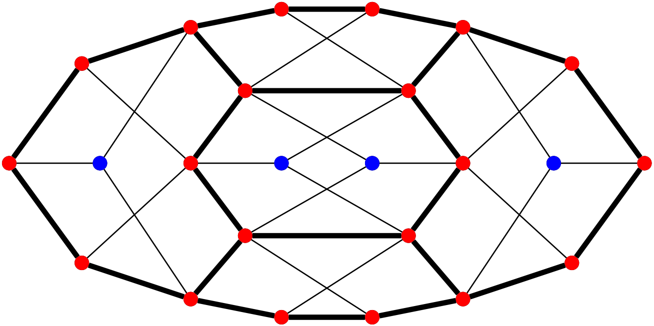
$G(F)$ es un árbol con raíz



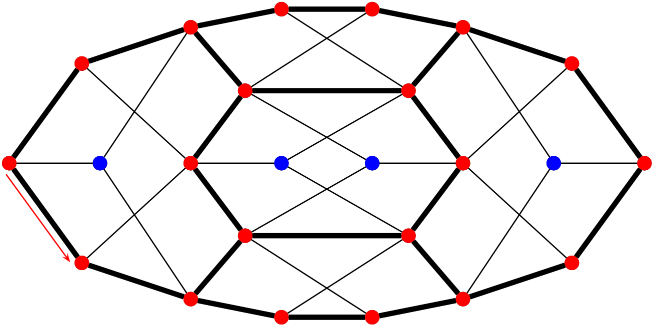
$G(F)$ es un árbol con raíz



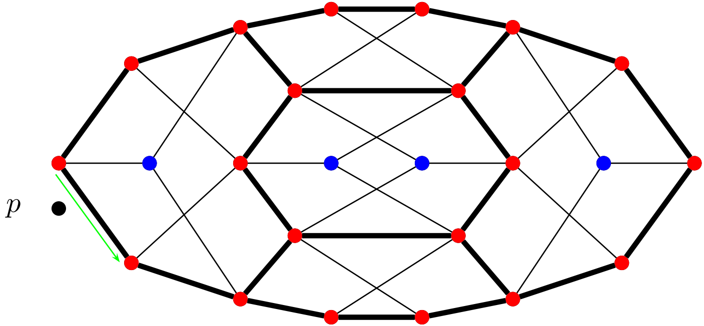
El algoritmo



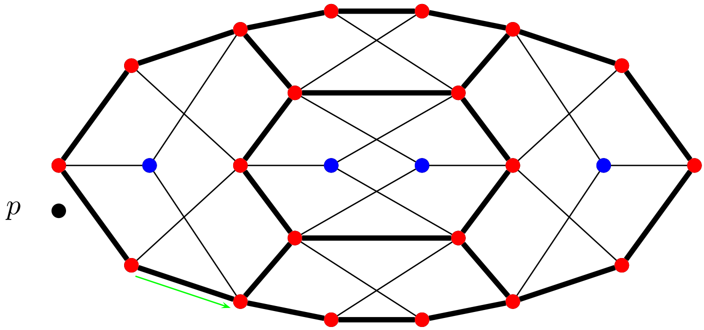
El algoritmo



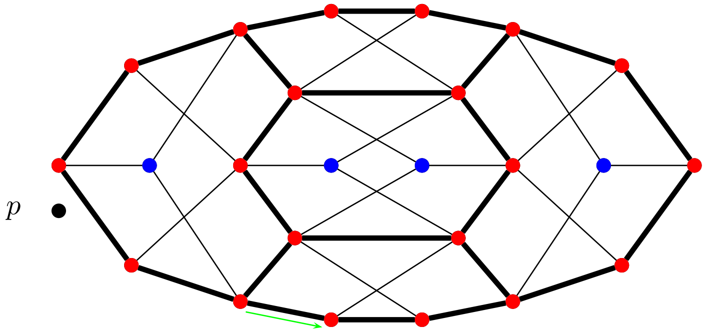
El algoritmo



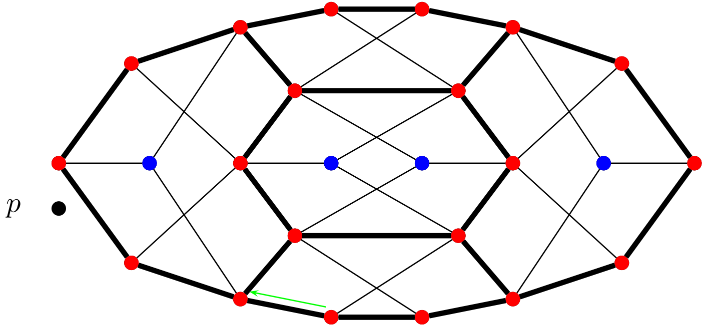
El algoritmo



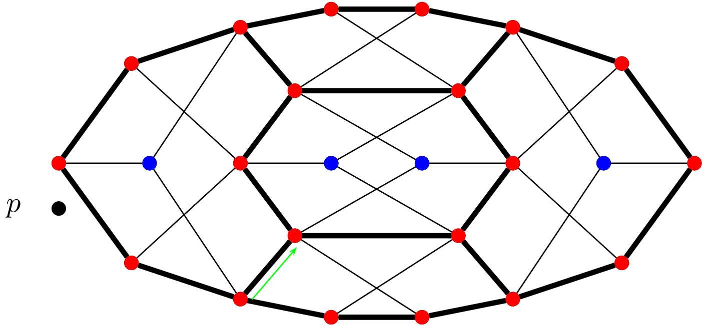
El algoritmo



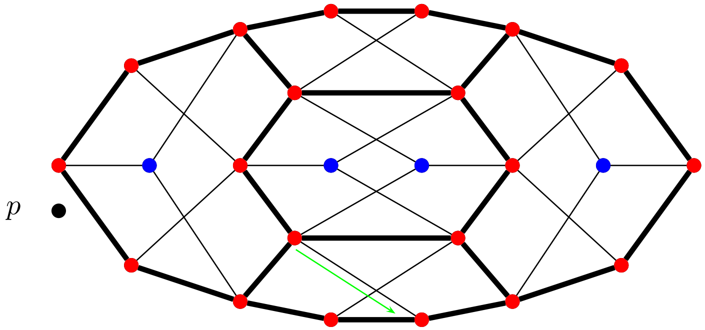
El algoritmo



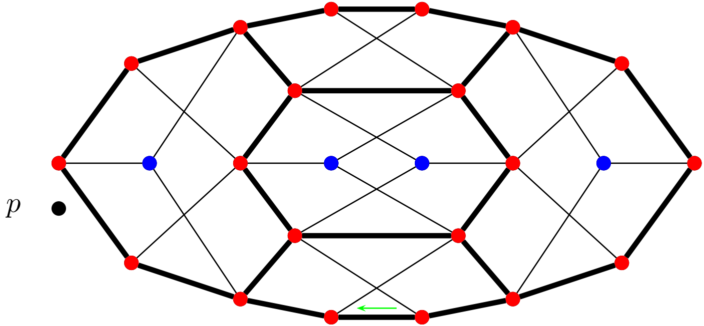
El algoritmo



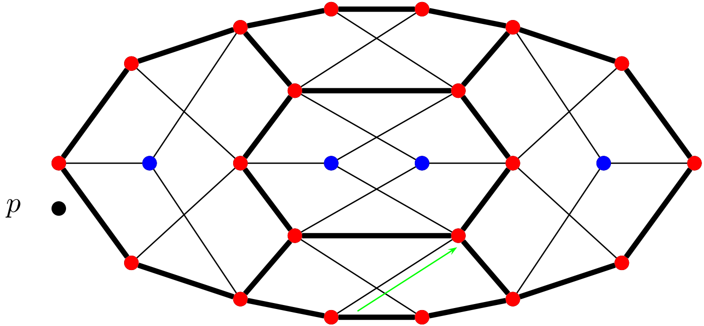
El algoritmo



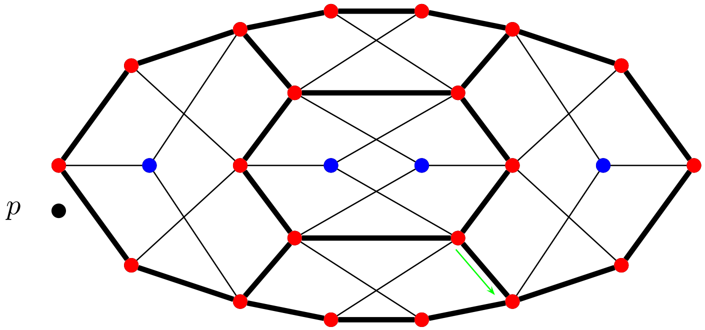
El algoritmo



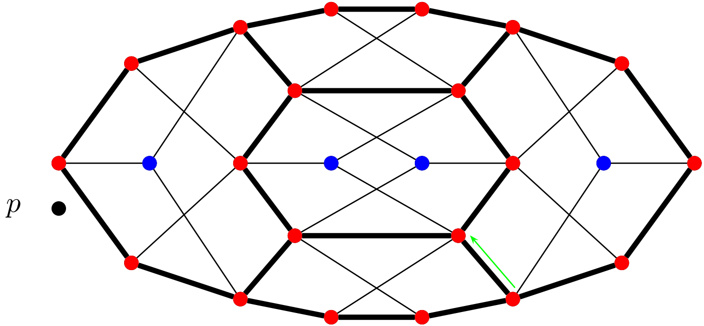
El algoritmo



El algoritmo



El algoritmo



Complejidad y correctez del algoritmo

Teorema 2

El algoritmo reporta cada vértice, arista no dirigida, y quasi-cara de una subdivisión quasi-plana G que satisfaga la Regla del Vecino Izquierdo exactamente una vez en tiempo $O(|E| \log |E|)$.

Complejidad y correctez del algoritmo

Teorema 2

El algoritmo reporta cada vértice, arista no dirigida, y quasi-cara de una subdivisión quasi-plana G que satisfaga la Regla del Vecino Izquierdo exactamente una vez en tiempo $O(|E| \log |E|)$.

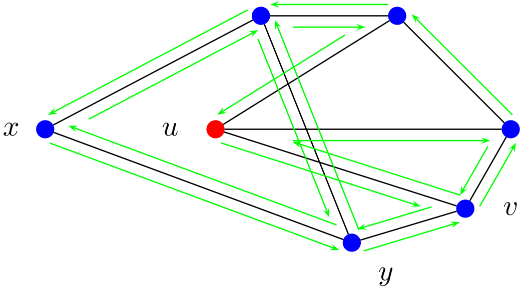
Si G era una subdivisión plana para empezar, entonces $|E| = O(|V|)$ y el tiempo del algoritmo es $O(|V| \log |V|)$, que era el tiempo del algoritmo para subdivisiones planas.

La importancia de ser de izquierda

Si la Regla del Vecino Izquierdo no se cumple, el algoritmo no funciona porque el orden $\preceq$ no puede garantizarse que sea total.

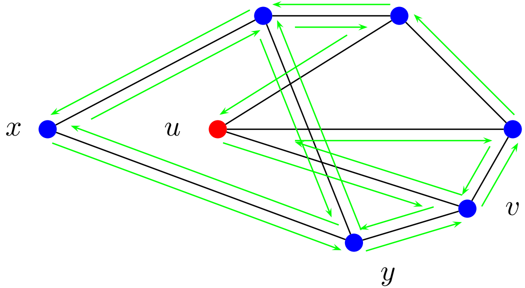
La importancia de ser de izquierda

Si la Regla del Vecino Izquierdo no se cumple, el algoritmo no funciona porque el orden $\preceq$ no puede garantizarse que sea total.



La importancia de ser de izquierda

Si la Regla del Vecino Izquierdo no se cumple, el algoritmo no funciona porque el orden $\preceq$ no puede garantizarse que sea total.



La función **ismin** regresa V para (u, v) y para (x, y).

Conclusiones

- Las técnicas aquí presentadas generalizan técnicas similares que se usan para recorrer subdivisiones planas a una clase muy amplia de subdivisiones no planas.

Conclusiones

- Las técnicas aquí presentadas generalizan técnicas similares que se usan para recorrer subdivisiones planas a una clase muy amplia de subdivisiones no planas.
- La idea principal es extender el concepto de cara en una subdivisión plana a un camino cerrado en gráficas simétricamente dirigidas.

Conclusiones

- Las técnicas aquí presentadas generalizan técnicas similares que se usan para recorrer subdivisiones planas a una clase muy amplia de subdivisiones no planas.
- La idea principal es extender el concepto de cara en una subdivisión plana a un camino cerrado en gráficas simétricamente dirigidas.
- Con esto, el algoritmo puede recorrer (en tiempo polinomial) una amplia clase de subdivisiones no planas que satisfagan una sencilla condición geométrica.

Gracias

Gracias

Bibliografía

C. Gold, U. Maydell, and J. Ramsden.
Automated contour mapping using triangular element data structures and interpolant over each irregular triangular domain.
Computer Graphic, 11(2):170-175, 1977.

D. Avis and K. Fukuda.
A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra.
In *Proc. of 7th Annu. ACM Sympos. Comput. Geom.*, pages 98-104, 1991.

M. de Berg, M. van Kreveld, R. van Oostrum, and M. Overmars.
Simple traversal of a subdivision without extra storage.
International Journal of Geographic Information Systems, 11:359-373, 1997.

P. Bose and P. Morin.
An improved algorithm for subdivision traversal without extra storage.
Internat. J. Comput. Geom. Appl., 12(4):297-308, 2002.
Annual International Symposium on Algorithms and Computation (Taipei, 2000).