Primi elementi

di

Geometria Computazionale Piana

Adriano Pascoletti

a.a. 2011–12

Indice

1 Poligoni e triangolazioni **1**
- 1.1 Poligoni . . . 1
  - 1.1.1 Poligono generico . . . 1
  - 1.1.2 Poligono semplice . . . 2
  - 1.1.3 Alcuni problemi . . . 3
- 1.2 Diagonali . . . 4
  - 1.2.1 Esistenza . . . 4
  - 1.2.2 Triangolazione . . . 5
  - 1.2.3 Un algoritmo di triangolazione . . . 7

2 Operazioni di base **9**
- 2.1 Area segnata . . . 9
  - 2.1.1 Triangolo . . . 9
  - 2.1.2 Poligono semplice . . . 12
  - 2.1.3 Retta per due punti . . . 13
- 2.2 Point location . . . 14
  - 2.2.1 Rispetto a una retta . . . 14
  - 2.2.2 Rispetto a un poligono convesso . . . 14
  - 2.2.3 Rispetto a un segmento . . . 15
  - 2.2.4 Rispetto a un poligono semplice . . . 16
- 2.3 Intersezioni . . . 17
  - 2.3.1 Semiretta orizzontale-segmento . . . 17
  - 2.3.2 Segmento-segmento . . . 18
- 2.4 Ordinamento per angolo polare . . . 18

3 Grafi planari **21**
- 3.1 Introduzione . . . 21
- 3.2 Poliedri e grafi planari . . . 24
- 3.3 Duale di un poligono semplice triangolato . . . 25

4 DCEL **29**
- 4.1 Versione originale . . . 29
- 4.2 Versione con semispigoli . . . 30
  - 4.2.1 Tabella dei vertici . . . 31
  - 4.2.2 Tabella dei semispigoli . . . 31
  - 4.2.3 Tabella delle facce . . . 31
  - 4.2.4 Campi . . . 33
- 4.3 Costruzione . . . 33
  - 4.3.1 Preprocessing . . . 34

4.3.2 Costruzione: fase I 34
4.3.3 Costruzione: fase II 36
4.4 Caso generale 39

Bibliografia **43**

1 Poligoni e triangolazioni

1.1 Poligoni 1
1.1.1 Poligono generico 1
1.1.2 Poligono semplice 2
1.1.3 Alcuni problemi 3
1.2 Diagonali 4
1.2.1 Esistenza 4
1.2.2 Triangolazione 5
1.2.3 Un algoritmo di triangolazione 7

Sommario

Il poligono — introdotto nella Sez. 1.1 è un ente fondamentale nella geometria computazionale piana. Di non minore importanza il suo caso particolare, il poligono semplice (Sez. 1.1). Un poligono semplice partiziona il piano in tre regioni — interno, esterno e frontiera — e permette di introdurre il concetto di diagonale (Sez. 1.2). La dimostrazione — costruttiva — dell'esistenza di diagonali per ogni poligono semplice con almeno quattro vertici porta ad abbozzare un algoritmo quadratico di triangolazione di un poligono semplice e a formulare tutta una serie di problemi geometrici di base.

1.1 Poligoni

1.1.1 Poligono generico

L'idea di poligono piano inteso come spezzata che si chiude su se stessa può essere formalizzata con la seguente

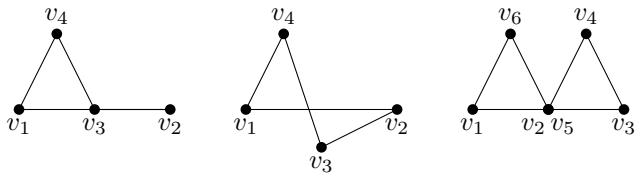
Definizione 1.1 Un poligono $\mathscr{P}$ nel piano è definito da una successione ordinata di un numero finito di tre o piú punti $v_1, v_2, \ldots, v_n$ (detti *vertici*) e dai segmenti $v_1v_2, \ldots, v_nv_1$ (detti *spigoli* o *lati*) con la seguente proprietà: due lati consecutivi $v_{i-1}v_i$ e v_iv_{i+1} si intersecano solo in corrispondenza del vertice v_i; formalmente

$$v_{i-1}v_i \cap v_iv_{i+1} = \{v_i\} \text{ con } v_0 = v_n \text{ e } v_{n+1} = v_1.$$

Da notare che la condizione di almeno tre vertici è ridondante: con due soli vertici l'intersezione dei lati consecutivi v_1v_2 e v_2v_1 non si riduce ad un vertice.

La definizione di poligono non richiede che i vertici siano distinti. Cosa succede se due vertici coincidono? Se due vertici consecutivi coincidono — ad esempio $v_i = v_{i+1}$ — allora l'intersezione $v_{i-1}v_i \cap v_iv_{i+1}$ non si riduce a v_i perché contiene anche v_{i+1}. Quindi nessun poligono può avere lati di lunghezza nulla.

Consideriamo qualche altro esempio: la figura (a) non rappresenta un poligono perché $v_1v_2 \cap v_2v_3$ non si riduce a $\{v_2\}$. La definizione di poligono non esclude l'intreccio di lati o la ripetizione di vertici non adiacenti, quindi gli oggetti rappresentati nelle figure (b) e (c) sono poligoni a tutti gli effetti.



(a) (b) (c)

La ripetizione di vertici non consecutivi e l'intreccio di lati sono proibiti nei poligoni semplici.

1.1.2 Poligono semplice

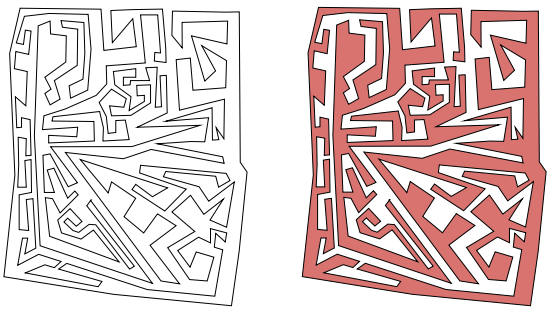
Definizione 1.2 Un poligono è detto semplice se ogni coppia di spigoli non consecutivi ha intersezione vuota.

Il poligono (b) non è semplice perchè gli spigoli non consecutivi v_1v_2 e v_3v_4 hanno intersezione non vuota; nemmeno (c) è semplice perché sono non vuote le seguenti quattro intersezioni di spigoli non consecutivi: $v_1v_2 \cap v_4v_5$, $v_1v_2 \cap v_5v_6$, $v_2v_3 \cap v_4v_5$ e $v_2v_3 \cap v_5v_6$.

Per i poligoni semplici vale il seguente, importantissimo, risultato che permette di parlare di *interno* ed *esterno*.

Teorema 1.3 (Teorema di Jordan, 1838–1922) *Un poligono semplice $\mathscr{P}$ divide il piano in due regioni o* facce*, una limitata (detta* interno *di $\mathscr{P}$) e una illimitata (detta* esterno *di $\mathscr{P}$). Percorrendo la frontiera di $\mathscr{P}$ in verso antiorario l'interno rimane a sinistra.*

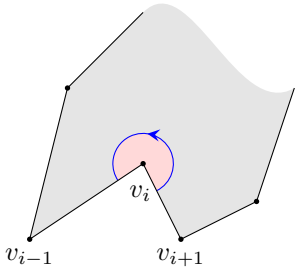
I poligoni semplici possono avere un aspetto tutt'altro che semplice: a destra c'è un poligono semplice con un centinaio di lati: non è semplice individuare l'interno senza la colorazione. Come faranno i programmi di grafica a decidere se un pixel è interno ad un poligono?



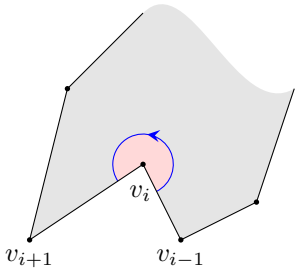
Il Teorema di Jordan ha introdotto tre coppie di termini — *interno-esterno*, *destra-sinistra*, *orario-antiorario* — che, come vedremo, sono indispensabili in molti argomenti della geometria computazionale. Un primo esempio è il concetto di angolo interno in corrispondenza di un vertice di un poligono: se il poligono non è semplice non è chiaro cosa si debba intendere con angolo al vertice.

Definizione 1.4 L'*angolo interno* in corrispondenza del vertice v_i di un poligono semplice $\mathscr{P}$ è definito come segue:

se $\mathscr{P}$ è assegnato tramite una lista ordinata di vertici $(v_1, \ldots, v_n)$ che determinano un percorso antiorario della frontiera, allora l'angolo interno in v_i è l'angolo di cui ruotare in verso antiorario il lato v_iv_{i+1} attorno a v_i per portare v_{i+1} sulla semiretta uscente da v_i e passante per v_{i-1}



se $\mathscr{P}$ è assegnato tramite una lista ordinata di vertici $(v_1, \ldots, v_n)$ che determinano un percorso orario della frontiera, allora l'angolo interno in v_i è l'angolo di cui ruotare in verso antiorario il lato v_iv_{i-1} attorno a v_i per portare v_{i-1} sulla semiretta uscente da v_i e passante per v_{i+1}



1.1.3 Alcuni problemi

Problema 1.5 È data una lista ordinata di punti distinti del piano $\mathscr{P} = (v_1, \ldots, v_n)$: può essere considerata come descrizione della frontiera di un poligono semplice? In caso affermativo stabilire se i vertici della frontiera sono elencati in verso antiorario oppure orario.

Soluzione. Posto $e_i = v_iv_{i+1}$ con le identificazioni $v_{n+1} = v_1$, $v_0 = v_n$, $e_0 = e_n$ e $e_{n+1} = e_1$, $\mathscr{P}$ è associabile a un poligono semplice se e solo se per ogni $i \in \{1, \ldots, n-2\}$ si ha

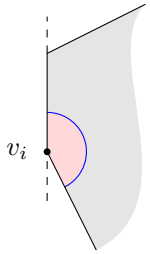
- $e_i \cap e_j = \emptyset$ $(\forall j \in J_i := \{j \in \{1, \ldots, n\} \setminus \{i-1, i, i+1\} \text{ e } j > i\})$;
- $e_i \cap e_j = \{v_i\}$ se $j = i - 1$;
- $e_i \cap e_j = \{v_{i+1}\}$ se $j = i + 1$. ∎

Questa soluzione, oltre ad essere poco efficiente perché richiede il calcolo di $O(n^2)$ intersezioni di coppie di segmenti, solleva un ulteriore problema, che sarà trattato nel prossimo capitolo:

Problema 1.6 Come trovare l'intersezione di due segmenti tenendo conto degli inconvenienti legati ad operazioni floating-point?

Problema 1.7 La Definizione 1.4 di angolo (interno) al vertice di un poligono semplice $\mathscr{P} = (v_1, \ldots, v_n)$ fa riferimento all'orientazione oraria o antioraria indotta dalla lista dei vertici. Come decidere se l'orientazione è antioraria oppure oraria?

Soluzione. Detto v_i il minimo (o il massimo) lessicografico dei punti di $\mathscr{P}$ si consideri la verticale per v_i. I vertici di $\mathscr{P}$ devono stare a destra (sinistra) della verticale per v_i, oppure su di essa ma piú in alto (basso) di v_i, l'angolo interno in v_i è minore di π.
Se la lista $\mathscr{P}$ è antioraria è necessaria una rotazione antioraria minore di π di v_iv_{i+1} attorno a v_i per portare v_{i+1} sulla semiretta uscente da v_i e passante per v_{i-1}. Se la lista è oraria la rotazione antioraria richiesta per arrivare alla stessa situazione è maggiore di π. ∎



Ed è cosí sorto il nuovo

Problema 1.8 Dati tre punti distinti v_{i-1}, v_i, v_{i+1} come determinare se la rotazione antioraria di v_iv_{i+1} attorno a v_i che porta v_{i+1} sulla semiretta uscente da v_i e passante per v_{i-1} è maggiore, minore, oppure uguale a π?

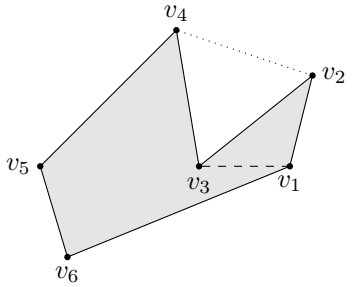
Soluzione. La soluzione è rimandata al prossimo capitolo. Qui anticipiamo che si riduce al calcolo del prodotto vettoriale di $v_i - v_{i-1}$ e $v_{i+1} - v_i$. Questo prodotto è eseguibile in $O(1)$ e il suo segno individua il tipo di svolta eseguita in v_i muovendosi da v_{i-1} a v_i e proseguendo poi verso v_{i+1}. ∎

1.2 Diagonali

Il Teorema di Jordan assicura che ogni poligono semplice definisce un interno, un esterno e, di conseguenza, una frontiera. Su questi concetti è basata la definizione di diagonale, la dimostrazione dell'esistenza di diagonali (Prop. 1.10) e di triangolazioni (Sez. 1.2.2).

Definizione 1.9 Il segmento v_iv_j $(i \neq j)$ è una diagonale del poligono semplice $\mathscr{P} = (v_1, \ldots, v_n)$ se i suoi estremi sono vertici di $\mathscr{P}$ e il suo interno è contenuto nell'interno di $\mathscr{P}$.

La figura mostra una diagonale (v_1v_3) e il segmento v_2v_4 che, contenendo punti esterni a $\mathscr{P}$, non è una diagonale. Se i vertici v_2, v_3 e v_6 fossero collineari v_3v_6 sarebbe una diagonale, ma non v_2v_6 perché conterrebbe punti della frontiera di $\mathscr{P}$ diversi dai suoi estremi.



1.2.1 Esistenza

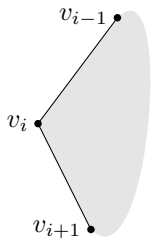
È stato osservato che un poligono semplice ha almeno tre lati. Un triangolo non ha diagonali e per i poligoni semplici con piú di tre lati si può dimostrare

che esiste una diagonale. La dimostrazione, costruttiva, sarà la chiave per lo sviluppo di un algoritmo di triangolazione.

Proposizione 1.10 *Ogni poligono semplice $\mathscr{P} = (v_1, \ldots, v_n)$ con $n \geq 4$ lati ha una diagonale.*

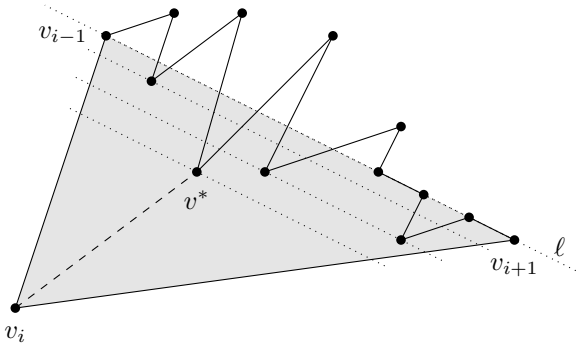
Dim.

Detto v_i il minimo lessicografico di $\mathscr{P}$ si supponga, senza perdita di generalità, che la lista dei vertici sia antioraria. Se il segmento $v_{i-1}v_{i+1}$ ha interno contenuto nell'interno di $\mathscr{P}$ allora il segmento $v_{i-1}v_{i+1}$ è una diagonale e la tesi è vera.
Se il segmento $v_{i-1}v_{i+1}$ non ha interno contenuto nell'interno di $\mathscr{P}$ allora, essendo l'angolo in v_i convesso, c'è almeno un vertice di $\mathscr{P}$ che sta sul segmento $v_{i-1}v_{i+1}$ oppure all'interno del triangolo $v_{i-1}v_iv_{i+1}$.



Se non ci sono vertici all'interno del triangolo $v_{i-1}v_iv_{i+1}$ e qualche vertice è interno al segmento $v_{i-1}v_{i+1}$ allora ogni segmento con un estremo in v_i e l'altro su uno dei vertici interni a $v_{i-1}v_{i+1}$ è una diagonale.

Se invece ci sono vertici all'interno di $v_{i-1}v_iv_{i+1}$ si consideri la retta ℓ per v_{i-1} e v_{i+1} e le sue parallele passanti per i vertici v interni al triangolo $v_{i-1}v_iv_{i+1}$: ognuna determina un triangolo con un vertice in v_i, uno sul lato v_iv_{i+1} e uno su $v_{i-1}v_i$.



Se la parallela a ℓ condotta per un vertice v^* individua un triangolo Δ di area minima allora all'interno di Δ non cadono vertici del poligono e il segmento v_iv^* è una diagonale.[1] ∎

1.2.2 Triangolazione

La diagonale costruita può essere di due tipi: dal predecessore al successore del vertice che realizza il minimo lessicografico, oppure da quest'ultimo ad un altro vertice v^* di $\mathscr{P}$.

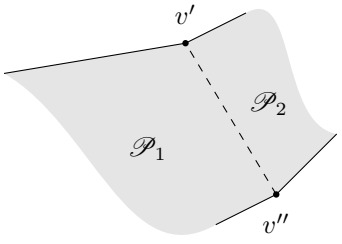
[1] Da un punto di vista algoritmico v^* è facilmente computabile come uno qualsiasi dei vertici che massimizzano l'area dei triangoli $v_{i+1}v_{i-1}v$ al variare di v tra i vertici che cadono internamente al triangolo $v_{i-1}v_iv_{i+1}$ o internamente al segmento $v_{i-1}v_{i+1}$.

Nel primo caso il triangolo $v_{i-1}v_iv_{i+1}$ è detto *orecchia* del poligono. Nel secondo caso nulla si può dire sul numero di lati dei due poligoni in cui $\mathscr{P}$ è spaccato dalla diagonale.

In generale un poligono semplice $\mathscr{P}$ è scisso da una diagonale $v'v''$ in due poligoni $\mathscr{P}_1$, $\mathscr{P}_2$ rispettivamente con n_1 e n_2 vertici.
Chiaramente $3 \leq n_1, n_2 < n$ e

$$n_1 + n_2 = n + 2$$

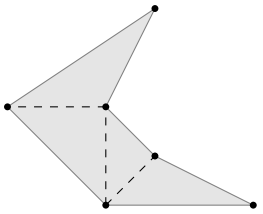
poiché i vertici comuni v' e v'' sono vertici sia di $\mathscr{P}_1$ che di $\mathscr{P}_2$.



Se n_1 e/o n_2 sono maggiori di tre possiamo costruire altre diagonali, e cosí via fino a quando $\mathscr{P}$ risulta suddiviso in triangoli. Questo procedimento ricorsivo porta ad una *triangolazione* di $\mathscr{P}$.

Definizione 1.11 Un poligono semplice $\mathscr{P}$ si dice triangolato se è unione di triangoli e i lati dei triangoli sono lati di $\mathscr{P}$ oppure diagonali.

Dalla seconda condizione si deduce che i vertici dei triangoli che compongono una triangolazione di $\mathscr{P}$ sono tutti vertici di $\mathscr{P}$ e ogni coppia di triangoli ha intersezione vuota o coincidente con un vertice o una diagonale.



È facile stabilire il numero di diagonali e triangoli presenti in ogni triangolazione di un poligono semplice con n lati.

Proposizione 1.12 *Ogni poligono semplice $\mathscr{P}$ con n lati può essere triangolato con $n-3$ diagonali e decomposto di conseguenza in $n-2$ triangoli.*

Dim. La tesi è vera per $n = 3$ perché un triangolo non ha diagonali. Supponiamola vera fino a $n-1$ ($n \geq 4$). È stato detto che una diagonale scinde $\mathscr{P}$ in due poligoni $\mathscr{P}_1$ e $\mathscr{P}_2$ rispettivamente con n_1 e n_2 lati ($3 \leq n_1, n_2 < n$, $n_1 + n_2 = n + 2$). Detti $d(n)$ e $t(n)$ il numero di diagonali e di triangoli di una triangolazione di $\mathscr{P}$ si ha

$$\begin{aligned} d(n) &= d(n_1) + d(n_2) + 1 \\ t(n) &= t(n_1) + t(n_2) \end{aligned}$$

e, usando l'ipotesi induttiva,

$$\begin{aligned} d(n) &= n_1 - 3 + n_2 - 3 + 1 = n + 2 - 6 + 1 = n - 3 \\ t(n) &= n_1 - 2 + n_2 - 2 = n + 2 - 4 = n - 2. \end{aligned}$$

■

1.2.3 Un algoritmo di triangolazione

I passi salienti di un algoritmo ricorsivo per la triangolazione di un poligono semplice $\mathscr{P}$ sono i seguenti:

Triangolazione basata sull'esistenza di diagonali.
Input: un poligono semplice $\mathscr{P} = \{v_1, \ldots, v_n\}$.
Output: la lista delle diagonali di una triangolazione di $\mathscr{P}$.

```
1  Triangolazione1(𝒫);
2  If |𝒫| = 3 then
3  Return(∅) (* un triangolo non ha diagonali *)
4  else begin
5     v_i v_j ← diagonale(𝒫);
6     𝒫_1 ← sottolista antioraria dei vertici v_i, ..., v_j;
7     𝒫_2 ← sottolista antioraria dei vertici v_j, ..., v_i;
8     Return(Join(Triangolazione1(𝒫_1),{v_i v_j}, Triangolazione1(𝒫_2)))
9  end;
10 end;
```

```
1  diagonale(𝒫);
2  v ← vertice convesso di 𝒫;
3  (v', v'') ← predecessore e successore di v; (* in verso antiorario *)
4  S ← vertici di 𝒫 \ {v', v, v''} interni al triangolo v'vv'' o su v'v'';
5  If S = ∅ then
6     Return(v'v'') (* orecchia *)
7  else begin
8     w ← un vertice di S che massimizzi l'area del triangolo v'wv'';
9     Return(vw)
10 end;
11 end;
```

Vediamo piú da vicino come funziona l'algoritmo di p. 7. Se $\mathscr{P}$ è un triangolo la lista è vuota; in caso contrario viene calcolata una diagonale (punto 5) e $\mathscr{P}$ è scisso in due poligoni $\mathscr{P}_1, \mathscr{P}_2$ come nella dimostrazione della Prop. 1.12. Le diagonali della triangolazione di $\mathscr{P}$ sono il risultato della concatenazione (Join, punto 9) di tre liste di diagonali: quella data da Triangolazione1($\mathscr{P}_1$), quella che contiene la sola diagonale trovata al punto 5 e quella data da Triangolazione1($\mathscr{P}_2$).

Il calcolo della diagonale (diagonale($\mathscr{P}$)) solleva vari problemi.

Il vertice convesso v di $\mathscr{P}$ può essere calcolato in $O(n)$ individuando il minimo (o il massimo) lessicografico.

Per quanto riguarda il punto 3 è già stato anticipato che la ricerca del predecessore e del successore di v in verso antiorario è fattibile in $O(1)$ calcolando un prodotto vettoriale.

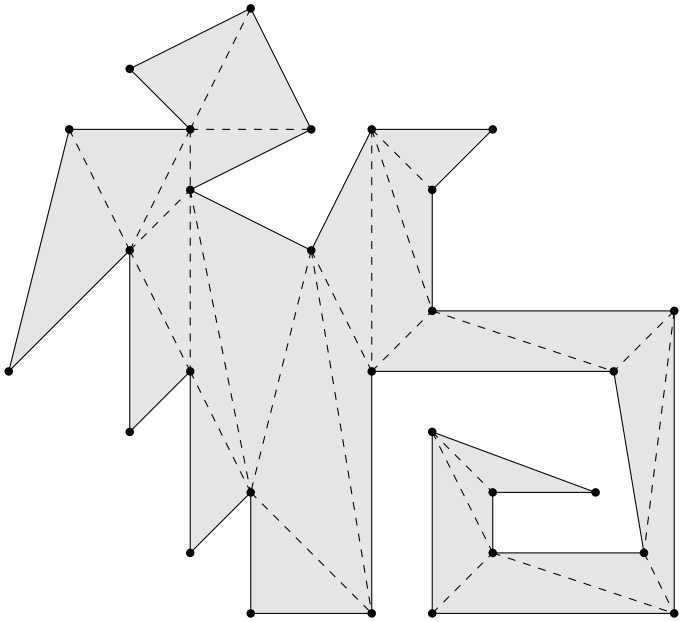
La grossa novità è il punto 4 in cui si richiede di trovare l'insieme dei vertici di $\mathscr{P}$, diversi da $\{v', v, v''\}$, interni al triangolo $v'vv''$ o sul segmento $v'v''$. Questo è un caso molto particolare di tutta una miriade di problemi di geometria computazionale che vanno sotto il nome di *Point Location*: *decidere*

se un dato punto di punto di interrogazione appartiene o meno a una prefissata regione del piano.

Nel nostro caso si devono localizzare i punti di $\mathscr{P} \setminus \{v', v, v''\}$ rispetto all'interno del triangolo $v'vv''$ e rispetto al segmento $v'v''$ e nel Cap. 2 si vedrà come tutte queste localizzazioni possano essere fatte in $O(n)$.

Il punto 8 richiede il calcolo dell'area di un triangolo di cui si conoscono le coordinate dei vertici: anche questo è fattibile in $O(1)$ come si vedrà nel prossimo capitolo.

Mettendo assieme tutte queste complessità si può dire che una diagonale di un poligono semplice di n lati è generabile in $O(n)$ e le $n-3$ diagonali della triangolazione sono generabili in $O(n^2)$.



Questo algoritmo di triangolazione non è ottimo. Nel corso sarà presentato un algoritmo che richiede $O(n \log n)$ che si basa sulla decomposizione di un poligono semplice in sottopoligoni monotoni triangolabili in tempo lineare.

Naturalmente ci sono casi particolari in cui la triangolazione è velocissima: sapendo che un poligono semplice $\mathscr{P}$ con n lati è convesso basta collegare il primo vertice con gli altri $n-3$ non adiacenti ($O(n)$).

Nel 1991 B. Chazelle [1] ha dimostrato la possibilità di triangolare un poligono semplice con n lati in tempo ottimo ($O(n)$) ma al momento non esiste un'implementazione adeguata.

2 Operazioni di base

2.1 Area segnata 9
2.1.1 Triangolo 9
2.1.2 Poligono semplice 12
2.1.3 Retta per due punti 13
2.2 Point location 14
2.2.1 Rispetto a una retta 14
2.2.2 Rispetto a un poligono convesso 14
2.2.3 Rispetto a un segmento 15
2.2.4 Rispetto a un poligono semplice 16
2.3 Intersezioni 17
2.3.1 Semiretta orizzontale-segmento 17
2.3.2 Segmento-segmento 18
2.4 Ordinamento per angolo polare 18

Sommario

In questo capitolo sono introdotte le operazioni di base della geometria computazionale piana la cui utilità è in buona parte emersa nel primo capitolo. Il concetto chiave è quello di area segnata — discusso nella Sez. 2.1 — che permette di risolvere alcuni problemi di point location, di intersezione, di orientazione e di ordinamento per angolo polare.

2.1 Area segnata

Aree segnate, prodotti vettoriali, determinanti sono concetti strettamente legati fra loro che consentono di rispondere agli interrogativi di base della geometria computazionale. Con la definizione di area segnata di un triangolo (Sez. 2.1.1) è possibile definire l'area segnata di un poligono (Sez. 2.1.2) ed esprimere la collinearità di punti del piano con una condizione di area nulla. In quest'ottica saranno discusse le equazioni cartesiane e parametriche della retta per due punti (Sez. 2.1.3).

2.1.1 Triangolo

Due punti v_1 e v_2 del piano determinano, assieme all'origine O, un triangolo Ov_1v_2, degenere quando i tre punti non sono distinti o quando v_2 è

proporzionale a v_1. L'area segnata della terna ordinata (O, v_1, v_2) è, per definizione,

$$\mathcal{A}(Ov_1v_2) = \frac{1}{2} v_1 \wedge v_2$$

dove $v_1 \wedge v_2$ è il prodotto vettoriale di v_1 e v_2. Se $v_i = (x_i, y_i)$ $(i = 1, 2)$ allora

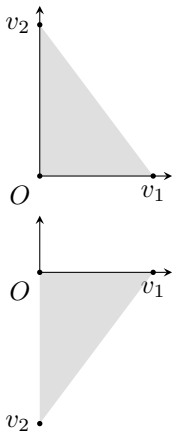
$$v_1 \wedge v_2 = \det \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} = x_1 y_2 - x_2 y_1.$$

Il determinante di una matrice quadrata cambia segno quando due righe (o due colonne) sono scambiate, quindi $\mathcal{A}(Ov_1v_2) = -\mathcal{A}(Ov_2v_1)$ ed è per questo motivo che si parla di area segnata di un triangolo e di area di una terna ordinata.

Esempio 2.1 Consideriamo un triangolo rettangolo con un vertice in O, uno sull'asse x ($v_1 = (x_1, 0)$) e l'altro sull'asse y ($v_2 = (0, y_2)$): la sua area segnata $x_1 y_2/2$ è positiva se e solo sono verificate due condizioni:

- $x_1, y_2 \neq 0$ (cateti non nulli, triangolo non degenere) e
- x_1, y_2 hanno lo stesso segno.

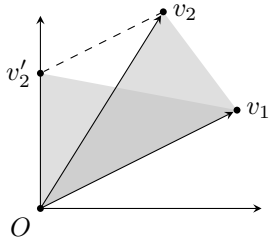
Quando questo succede il ciclo O, v_1, v_2 è antiorario.
Se l'area segnata è negativa il triangolo è non degenere e il ciclo O, v_1, v_2 è orario. ■



Esempio 2.2 Consideriamo il caso, un po' piú generale, di un triangolo con un vertice nell'origine e gli altri due generici e verifichiamo che l'area segnata è proprio $\mathcal{A}(Ov_1v_2) = \frac{1}{2}(x_1y_2 - x_2y_1)$. Se $x_1 \neq 0$ si può scrivere

$$\mathcal{A}(Ov_1v_2) = \frac{1}{2} x_1 (y_2 - \frac{x_2}{x_1} y_1).$$

Il termine tra parentesi è l'ordinata y_2' di v_2', punto d'intersezione dell'asse y con la parallela a Ov_1 condotta per v_2.
I triangoli Ov_1v_2 e Ov_1v_2' hanno la stessa area segnata; Ov_1v_2' ha base y_2' e altezza x_1. La sua area segnata, se diversa da zero, è positiva quando la terna (O, v_1, v_2) determina un percorso antiorario, negativa quando è orario.

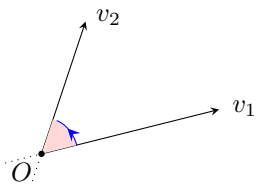


Se $x_1 = 0$ il triangolo Ov_1v_2 ha base y_1 e altezza x_2 e la formula conserva la sua validità.

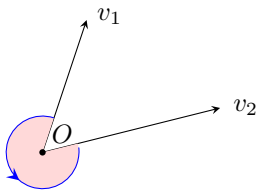
Se $x_1 = 0$ e $x_2 > 0$ l'area è positiva quando $y_1 > 0$ $(O, (0, y_1), (x_2, y_2)$ antiorario) e negativa quando $y_1 < 0$ (percorso antiorario). ■

▶ **Aspetti geometrici.** Prima di passare all'area di un triangolo generico è opportuno soffermarsi per riassumere alcuni aspetti geometrici del prodotto vettoriale.

- $v_1 \wedge v_2 > 0$ se e solo se andando da O a v_1 e poi a v_2 la frontiera del triangolo Ov_1v_2 è percorsa in verso antiorario.
  Questa condizione è anche equivalente al fatto che v_2 giace nel semipiano aperto a sinistra della retta orientata $\overrightarrow{Ov_1}$, e anche al fatto che una rotazione antioraria minore di π di Ov_1 attorno all'origine porta v_1 sulla semiretta che esce da O e passa per v_2.



- $v_1 \wedge v_2 < 0$ se e solo se andando da O a v_1 e poi a v_2 la frontiera del triangolo Ov_1v_2 è percorsa in verso orario. Condizioni equivalenti: v_2 giace nel semipiano aperto a destra della retta orientata $\overrightarrow{Ov_1}$ e una rotazione antioraria maggiore di π di Ov_1 attorno a O porta v_1 sulla semiretta uscente da O e passante per v_2.



- $v_1 \wedge v_2 = 0$ se e solo se v_1 e/o v_2 sono nulli, oppure v_1 e v_2 sono allineati con l'origine. In quest'ultimo caso serve una rotazione antioraria di 0 o π di Ov_1 attorno a O per portare v_1 sulla semiretta uscente da O e passante per v_2.

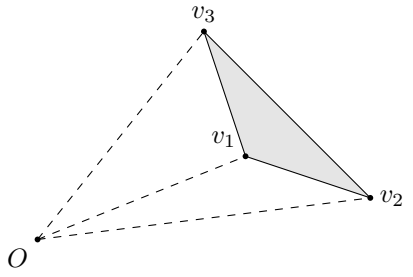
▶ **Area segnata di un triangolo generico.** Passiamo ora al calcolo dell'area del triangolo individuato da tre punti v_1, v_2, v_3 del piano. Poiché le traslazioni non alterano le aree possiamo scegliere uno dei tre vertici, ad esempio v_1, e portarlo nell'origine. I vertici del nuovo triangolo sono O, $v_2 - v_1$, $v_3 - v_1$ e la sua area segnata è data da

$$\frac{1}{2}(v_2 - v_1) \wedge (v_3 - v_1) = \frac{1}{2}(v_1 \wedge v_2 + v_2 \wedge v_3 + v_3 \wedge v_1)$$

grazie alla distributività del prodotto vettoriale rispetto alla somma e alla sua alternanza ($v_i \wedge v_j = -v_j \wedge v_i$) che comporta $v_i \wedge v_j = 0$ quando $j = i$.

L'area trovata è una somma di tre termini facili da ricordare $(1,2;\ 2,3;\ 3,1)$ e interpretabili come aree di altrettanti triangoli con un vertice in O: Ov_1v_2, Ov_2v_3, Ov_3v_1.

Nella figura a fianco la prima e la terza area sono negative, la seconda è positiva e la loro somma è l'area segnata di v_1, v_2, v_3, positiva perché il triangolo è antiorario.



Quando si ha a che fare con un'aritmetica imprecisa (floating-point) è preferibile un'altra espressione che prevede il calcolo del determinate di una matrice di ordine tre i cui elementi hanno il pregio di non richiedere operazioni

preliminari sulle componenti dei punti, contrariamente al caso di

$$\det \begin{bmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{bmatrix}.$$

Proposizione 2.3 *L'area del triangolo con vertici $v_i = (x_i, y_i)$ $(i = 1, 2, 3)$ è*

$$\mathcal{A}(v_1, v_2, v_3) = \frac{1}{2} \det \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix}.$$

Dim. Sviluppando il determinante lungo la terza colonna si ha $\frac{1}{2}(v_2 \wedge v_3 + v_3 \wedge v_1 + v_1 \wedge v_2)$. ∎

2.1.2 Poligono semplice

Siamo anche in grado di calcolare l'area segnata di un poligono semplice usando una sua triangolazione e la formula per l'area segnata di un triangolo generico.

Proposizione 2.4 *Se $\mathscr{P} = (v_1, \ldots, v_n)$ è un poligono semplice la sua area è*

$$\mathcal{A}(\mathscr{P}) = \frac{1}{2} \sum_{i=1}^{n} v_i \wedge v_{i+1} \quad (v_{n+1} = v_1).$$

Anche questa formula è facile da ricordare: $1, 2;\ 2, 3;\ \ldots;\ n-1, n;\ n, 1$.

Dim. La formula proposta è vera per $n = 3$. Supponiamola vera fino a $n - 1$. Se $n > 3$ il poligono $\mathscr{P}$ ammette una diagonale che ne provoca la scissione in due poligoni $\mathscr{P}_1, \mathscr{P}_2$ rispettivamente con n_1 e n_2 lati. Chiaramente

$$3 \leq n_1, n_2 \leq n - 1, \quad n_1 + n_2 = n + 2 \quad \text{e} \quad \mathcal{A}(\mathscr{P}) = \mathcal{A}(\mathscr{P}_1) + \mathcal{A}(\mathscr{P}_2).$$

Senza perdere in generalità si può supporre che la diagonale sia $v_1 v_i$ e usare l'ipotesi induttiva

$$\begin{aligned} \mathcal{A}(\mathscr{P}_1) &= \frac{1}{2} \sum_{h=1}^{i-1} v_h \wedge v_{h+1} + \frac{1}{2} v_i \wedge v_1 \\ \mathcal{A}(\mathscr{P}_2) &= \frac{1}{2} \sum_{h=i}^{n} v_h \wedge v_{h+1} + \frac{1}{2} v_1 \wedge v_i \quad (v_{n+1} = v_1) \\ \mathcal{A}(\mathscr{P}) &= \frac{1}{2} \sum_{h=1}^{i-1} v_h \wedge v_{h+1} + \frac{1}{2} \sum_{h=i}^{n} v_h \wedge v_{h+1} \end{aligned}$$

perché $v_i \wedge v_1 = -v_1 \wedge v_i$. ∎

2.1.3 Retta per due punti

Il prodotto vettoriale è utilissimo anche in molte altre occasioni, come avremo modo di vedere. Qui ci limitiamo al seguente

Problema 2.5 Dati due punti distinti del piano (v_1 e v_2) trovare l'equazione della retta r che passa per essi.

Soluzione. Basta osservare che ogni punto v di r è allineato con v_1 e v_2, quindi $v - v_1$ e $v_2 - v_1$ sono proporzionali e, di conseguenza, $(v - v_1) \wedge (v_2 - v_1) = 0$. Usando la definizione di prodotto vettoriale si ottiene l'equazione cartesiana di r

$$(x - x_1)(y_2 - y_1) = (y - y_1)(x_2 - x_1). \tag{2.1}$$

Se r non è verticale ($x_2 \neq x_1$) si ha

$$y = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1).$$

Se r è verticale $x_1 - x_2 = 0$ e si ottiene $(x - x_1)(y_2 - y_1) = 0$. Ma $y_2 \neq y_1$ se $v_1 \neq v_2$, quindi

$$x = x_1.$$

■

Al variare di t in $\mathbb{R}$ il punto $v(t) = (x(t), y(t))$ con

$$\begin{aligned} x(t) &= x_1 + t(x_2 - x_1) \\ y(t) &= y_1 + t(y_2 - y_1) \end{aligned} \tag{2.2}$$

è un punto di r perché soddisfa la (2.1) e, viceversa, per ogni punto (x, y) che soddisfa la (2.1) esiste $t \in \mathbb{R}$ per cui $x = x(t)$ e $y = y(t)$. Le equazioni (2.2) sono dette equazioni parametriche di r.

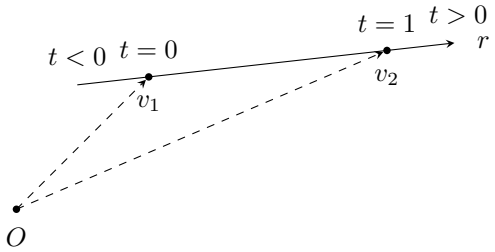
Ogni punto della retta è somma di v_1 e di un vettore proporzionale a $v_2 - v_1$, quindi $v(t) = v_1 + t(v_2 - v_1)$. In particolare $v(0) = v_1$, $v(1) = v_2$ e al variare di t da zero a uno viene descritto il segmento v_1v_2 orientato da v_1 a v_2:

$$v_1v_2 = \left\{ v \in \mathbb{R}^2 : v = (1 - t)v_1 + tv_2,\ t \in [0, 1] \right\}.$$

Questo è l'insieme delle combinazioni convesse di v_1 e v_2 (combinazioni lineari del tipo $t_1v_1 + t_2v_2$ con $t_1, t_2 \in \mathbb{R}$, $t_1, t_2 \geq 0$ e $t_1 + t_2 = 1$). La retta r, esprimibile come

$$\left\{ v \in \mathbb{R}^2 : v = (1 - t)v_1 + tv_2,\ t \in \mathbb{R} \right\},$$

è un insieme di combinazioni di v_1 e v_2 — piú generali delle combinazioni convesse — che vanno sotto il nome di combinazioni affini.



Per $t \geq 0$ si ottengono i punti della semiretta uscente da v_1 e passante per v_2, per $t \leq 0$ quelli dell'altra semiretta uscente da v_1.

2.2 Point location

Una retta orientata r definisce due semipiani e il prodotto vettoriale risolve il problema della localizzazione di un punto di interrogazione q rispetto ad essa: sta sulla retta o fuori? se sta fuori e la retta è orientata come dire se q sta a sinistra o a destra? La risposta è contenuta nella Sez. 2.2.1. Una serie di localizzazioni di questo tipo permette di risolvere il problema della localizzazione rispetto ad un poligono convesso (Sez. 2.2.2), ad un segmento (Sez. 2.2.3) e ad un poligono semplice (Sez. 2.2.4).

2.2.1 Rispetto a una retta

Una coppia di punti distinti del piano (v_1, v_2) determina una retta che può essere vista come frontiera comune di due semipiani aperti: H_ℓ formato dai punti a sinistra della retta r orientata da v_1 a v_2 e H_r formato dai punti a destra di essa. Chiaramente

$$H_\ell = \left\{v \in \mathbb{R}^2 : \mathcal{A}(v_1, v_2, v) > 0\right\}, \quad H_r = \left\{v \in \mathbb{R}^2 : \mathcal{A}(v_1, v_2, v) < 0\right\}$$

perché un punto q appartiene a H_ℓ (H_r) se e solo se il triangolo $\Delta(v_1, v_2, q)$ è antiorario (orario). Quindi

Proposizione 2.6 *Un punto q sta sulla retta r individuata da due punti distinti v_1 e v_2 se e solo se $\mathcal{A}(v_1, v_2, q) = 0$ (collinearità) e sta fuori da essa se e solo se $\mathcal{A}(v_1, v_2, q) \neq 0$. Se la retta è orientata da v_1 a v_2 allora q sta nel semipiano sinistro se e solo se $\mathcal{A}(v_1, v_2, q) > 0$ e in quello destro se e solo se $\mathcal{A}(v_1, v_2, q) < 0$.*

Molto utile anche la seguente interpretazione dell'area segnata di un triangolo in termini di svolte. Se il triangolo $\Delta(v', v, v'')$ determinato da una terna ordinata di punti distinti ha area positiva allora nel cammino che parte da v' e raggiunge v'' passando per v c'è una svolta a sinistra in corrispondenza di v. Se l'area è negativa c'è una svolta a destra. Se invece l'area è nulla allora v'' è allineato con v' e v e in v il cammino può proseguire senza svolte a con un'inversione a U (svolta di π).

Dato un poligono abbiamo già osservato che l'angolo al vertice in corrispondenza del minimo (o del massimo) lessicografico v è convesso. Ora possiamo dire che la frontiera del poligono è antioraria se e solo se in v c'è una svolta a sinistra.

2.2.2 Rispetto a un poligono convesso

Esistono moltissimi modi, tutti equivalenti, di definire un poligono convesso. Scegliamo quello piú intuitivo:

Definizione 2.7 Un poligono semplice $\mathscr{P} = (v_1, \ldots, v_n)$ è convesso se e solo se tutti gli angoli interni sono convessi.

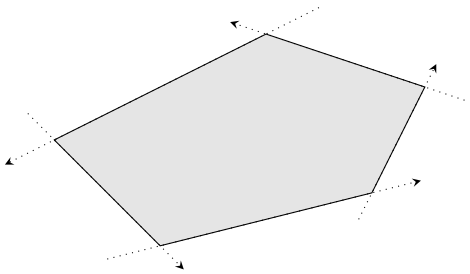
Supponendo che la lista ordinata che definisce $\mathscr{P}$ induca un attraversamento antiorario della frontiera di $\mathscr{P}$ la convessità di ogni angolo interno è equivalente

alla condizione

$$(v_i - v_{i-1}) \wedge (v_{i+1} - v_i) > 0 \quad (i = 1, \ldots, n)$$

con $v_1 = v_{n+1}$, come al solito.

La retta determinata da due vertici consecutivi v_i e v_{i+1} orientata dal primo al secondo ($r_i := \overrightarrow{v_i v_{i+1}}$) ha la seguente proprietà:



l'interno di $\mathscr{P}$ sta nel semipiano sinistro di r_i e l'intersezione della frontiera di $\mathscr{P}$ con r_i si riduce al lato $v_i v_{i+1}$. E viceversa:

- q è interno a $\mathscr{P}$ se e solo sta nel semipiano sinistro di r_i per $i = 1, \ldots, n$, cioè $\mathcal{A}(v_i, v_{i+1}, q) > 0$ con $1 \leq i \leq n$;
- q è interno o sulla frontiera di $\mathscr{P}$ se e solo se sta nell'intersezione dei semipiani chiusi

  $$\bigcap_{i=1}^{n} \left\{ v \in \mathbb{R}^2 : \mathcal{A}(v_i, v_{i+1}, v) \geq 0 \right\}$$

  ossia $\mathcal{A}(v_i, v_{i+1}, q) \geq 0$ con $1 \leq i \leq n$;
- è un punto della frontiera se e solo se $\mathcal{A}(v_i, v_{i+1}, q) \geq 0$ per $1 \leq i \leq n$ e la disuguaglianza è un'uguaglianza per almeno un indice i;
- è esterno a $\mathscr{P}$ se e solo se esiste almeno un indice $i \in \{1, \ldots, n\}$ per cui $\mathcal{A}(v_i, v_{i+1}, q) < 0$ (sta nel semipiano destro di r_i per almeno un indice i).

Presentando l'algoritmo per la triangolazione di un poligono semplice basato sull'esistenza di diagonali erano comparsi due problemi: come determinare se un punto q è interno ad un triangolo $v'vv''$ e come determinare se q appartiene ad un segmento $v'v''$. Abbiamo la soluzione del primo: q è interno al triangolo antiorario $v'vv''$ se e solo se $\mathcal{A}(v'vq) > 0$, $\mathcal{A}(vv''q) > 0$, $\mathcal{A}(v''v'q) > 0$.

2.2.3 Rispetto a un segmento

Dato un segmento $v'v''$ non degenere e un punto di interrogazione q dire se tale punto sta sul segmento oppure no.

Se il segmento è verticale ($v'_x = v''_x$) allora $q \in v'v''$ se e solo se

$$v'_y \leq q_y \leq v''_y \quad \text{oppure} \quad v''_y \leq q_y \leq v'_y.$$

Altrettanto facile il caso di segmento orizzontale ($v'_y = v''_y$): $q \in v'v''$ se e solo se

$$v'_x \leq q_x \leq v''_x \quad \text{oppure} \quad v''_x \leq q_x \leq v'_x.$$

Se $v'v''$ non è né orizzontale, né verticale devono essere verificate due condizioni: la collinearità ($\mathcal{A}(v', v'', q) = 0$) e l'appartenenza alla fascia verticale determinata dagli estremi del segmento ($v'_x \leq q_x \leq v''_x$ oppure $v''_x \leq q_x \leq v'_x$).

Abbiamo ora a disposizione tutti gli strumenti per triangolare in $O(n^2)$ un poligono semplice con n lati.

2.2.4 Rispetto a un poligono semplice

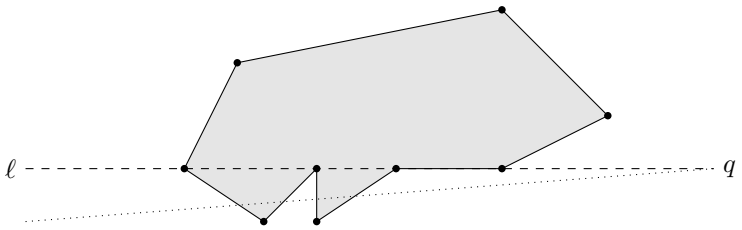
Ci occupiamo ora di un altro importante problema di point location:

Problema 2.8 Dato un poligono semplice $\mathscr{P}$ con n vertici $\{v_1, \ldots, v_n\}$ e un punto di interrogazione q dire dove sta q rispetto a $\mathscr{P}$: all'interno, all'esterno o sulla sua frontiera?

L'appartenenza alla frontiera è facilmente liquidabile localizzando q rispetto ai lati v_1v_2, v_2v_3,...,v_nv_1: q appartiene alla frontiera di $\mathscr{P}$ se e solo se appartiene ad almeno un lato.

Supponiamo ora che q non appartenga alla frontiera di $\mathscr{P}$. L'idea è sostanzialmente quella che ha portato Jordan alla dimostrazione che ogni poligono semplice ha un interno ed un esterno, oltre, naturalmente, alla frontiera.

Si consideri una semiretta orizzontale ℓ che parte da $-\infty$ e termina in q e si contino le sue intersezioni con la frontiera di $\mathscr{P}$. Il punto all'infinito non appartiene a $\mathscr{P}$. Procedendo lungo ℓ la prima intersezione — se esiste — comporta verosimilmente un ingresso all'interno di $\mathscr{P}$. Ogni altra intersezione provoca, in generale, un passaggio dall'interno all'esterno o viceversa e si può congetturare che se il numero di intersezioni è pari allora q è all'esterno di $\mathscr{P}$ ed è all'interno se tale numero è dispari. Questa è l'idea generale ma è facile produrre esempi che destano perplessità sul reale numero di intersezioni.



Il numero di lati è finito ed è finito anche il numero delle loro direzioni: esiste quindi una leggera perturbazione di ℓ — ad esempio una rotazione in verso antiorario attorno a q — che fa sí che l'intersezione della nuova semiretta con $\mathscr{P}$ non contenga vertici di $\mathscr{P}$. In questo caso ogni intersezione provoca un passaggio dall'interno all'esterno o viceversa.

Il fatto importante è che non è necessario perturbare effettivamente la semiretta orizzontale; è sufficiente immaginarla perturbata e comportarsi di conseguenza (*perturbazione simbolica*). Infatti,

- se ℓ contiene un lato orizzontale la sua versione leggermente ruotata non interseca tale lato;

- per i lati ab non orizzontali vanno distinti due casi patologici. Detto a l'estremo alto e b quello basso ($y_a > y_b$)
  - se $\ell \cap ab = \{a\}$ la perturbazione fa sí che l'intersezione cada non piú su a, ma all'interno di ab,
  - se invece $\ell \cap ab = \{b\}$ la perturbazione fa sparire l'intersezione.

Proposizione 2.9 *Se q non è un punto della frontiera di un poligono semplice $\mathscr{P}$ il numero di intersezioni della semiretta orizzontale ℓ (da q a $-\infty$) con la frontiera di $\mathscr{P}$ è uguale al numero di lati intersecati da ℓ in un punto del loro interno o nell'estremo piú alto. Se questo numero è pari q sta all'esterno di $\mathscr{P}$, se è dispari all'interno.*

È chiaro la localizzazione di q rispetto alla frontiera di un poligono semplice con n vertici è fattibile in $O(n)$ e se q non sta sulla frontiera anche la localizzazione rispetto all'interno di $\mathscr{P}$ richiede $O(n)$. In totale $O(n)$ sia per lo spazio che per il tempo.

2.3 Intersezioni

In questa sezione vengono trattati alcuni semplici problemi di intersezione: tra un segmento e una semiretta orizzontale (Sez. 2.3.1) e tra due segmenti (Sez. 2.3.2). Sono problemi facilmente risolubili analiticamente ma, nella pratica, ci si deve confrontare con le imprecisioni dell'aritmetica in virgola mobile. L'approccio geometrico-computazionale a questi importanti problemi — spesso mal condizionati — cerca di evitare operazioni critiche come la divisione per un determinante prossimo a zero e anche in questo caso il prodotto vettoriale gioca un ruolo determinante.

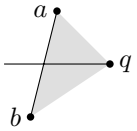
2.3.1 Semiretta orizzontale-segmento

Per completare l'algoritmo di point location rispetto ad un poligono semplice dobbiamo risolvere ancora un problema:

Problema 2.10 data una semiretta orizzontale ℓ da $-\infty$ a un punto prefissato q e un segmento ab, dire se ℓ interseca ab oppure no e, in caso affermativo, dire se $q \in ab$.

Se il segmento è orizzontale è ovvio che c'è intersezione solo se l'ordinata di q coincide con quella dei punti della semiretta ($q_y = a_y\ (= b_y)$) e in tal caso l'intersezione è non vuota se e solo se $q_x \geq \min\{a_x, b_x\}$.

Se ab non è orizzontale si può supporre, senza perdita di generalità, che sia $a_y > b_y$. C'è intersezione se e solo se $y_b \leq y_q \leq y_a$ e $\mathcal{A}(b, q, a) \geq 0$.

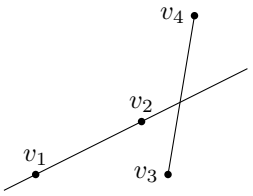


Se queste due condizioni sono soddisfatte con $\mathcal{A}(b, q, a) = 0$ allora $q \in ab$.

2.3.2 Segmento-segmento

Prima di trattare il problema dell'intersezione di due segmenti analizziamo quello, molto piú semplice, tra la retta r determinata da due punti v_1 e v_2 ed un segmento v_3v_4.

C'è intersezione se e solo se v_3 e v_4 non stanno nello stesso semipiano aperto individuato dalla retta per v_1 e v_2, cioè se e solo se $\mathcal{A}(v_1, v_2, v_3)\mathcal{A}(v_1, v_2, v_4) \leq 0$. Se la disuguaglianza è stretta le due aree hanno segni opposti (v_3 sta in un semipiano aperto, v_4 nell'altro), se è un'uguaglianza o v_3 o v_4 o entrambi sono sulla retta e l'intersezione non è vuota. Se il prodotto delle aree fosse positivo v_3 e v_4 starebbero nello stesso semipiano aperto e non ci sarebbero intersezioni.



Siamo molto vicini alla soluzione del problema dell'intersezione di due segmenti. Questa intersezione può presentarsi in varie forme:

- un unico punto interno ai due segmenti (la cosiddetta intersezione *forte* o *generica*),
- un estremo di un segmento che appartiene all'altro segmento,
- un segmento (eventualmente vuoto) quando gli estremi di un segmento stanno sulla retta determinata dall'altro segmento.

In questi ultimi due casi parleremo di intersezione *debole* perché esistono perturbazioni arbitrariamente piccole degli estremi che distruggono tali intersezioni (un'*instabilità strutturale*).

Detti v_1v_2 e v_3v_4 i due segmenti, l'intersezione forte è equivalente al verificarsi delle due condizioni

$$\mathcal{A}(v_1, v_2, v_3)\mathcal{A}(v_1, v_2, v_4) < 0 \quad \text{e} \quad \mathcal{A}(v_3, v_4, v_1)\mathcal{A}(v_3, v_4, v_2) < 0.$$

La prima dice che l'estremo v_3 appartiene ad uno dei semipiani aperti individuati da v_1v_2 e v_4 all'altro. Analogo il significato della seconda: v_1 e v_2 appartengono ai semipiani aperti opposti determinati dalla retta v_3v_4.

2.4 Ordinamento per angolo polare

Il capitolo si chiude con il problema dell'ordinamento di un insieme di punti in base al loro angolo polare. Questo ordinamento è di fondamentale importanza in molti algoritmi di geometria computazionale piana: mi limito a citare la costruzione dell'inviluppo convesso di un insieme di punti e la costruzione della struttura dati DCEL (*doubly-connected edge list* trattata nel Capitolo 4) a partire dalla descrizione di un grafo planare per mezzo di liste di adiacenza.

Problema 2.11 Si consideri un insieme di punti del piano diversi dall'origine $S = \{v_1, \ldots, v_n\}$; si desidera una sua permutazione $S' = \{v'_1, \ldots, v'_n\}$ con la seguente proprietà: *l'angolo polare di v'_i non supera quello di v'_{i+1}* ($1 \leq i \leq n-1$).

La formulazione del problema coinvolge punti del piano $v = (x, y)$ e angoli polari: immediatamente balza alla mente un possibile algoritmo basato sul-

l'arcotangente, perché l'angolo polare θ di v è uguale a arctan $\frac{y}{x}$. Le possibili limitazioni di questo approccio sono evidenti:

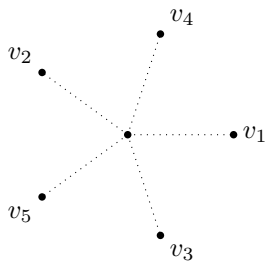
- eventuali punti con ascissa nulla possono essere gestiti dalle varianti di arctan con due argomenti (ascissa e ordinata) presenti in ogni libreria numerica seria;
- la funzione arctan è periodica di periodo π mentre l'angolo polare è compreso tra zero (incluso) e 2π (escluso) quindi il calcolo di arctan deve essere seguito da considerazioni sui segni di x e y;
- arctan è calcolato ricorrendo al troncamento di una serie, quindi con un numero — anche molto grande — di operazioni in virgola mobile (se due punti sono vicini tra loro e molto distanti dall'origine l'arcotangente dev'essere calcolato con grande precisione).

L'alternativa da adottare è basata sul prodotto vettoriale. Se S_+ è un insieme di punti (x, y), diversi dall'origine e con ordinata positiva oppure o nulla ma con ascissa positiva ($y > 0$ o $y = 0$ e $x > 0$) allora gli angoli polari di due punti $v_i, v_j \in S_+$ sono facilmente confrontabili: *l'angolo polare di v_i non supera quello di v_j se e solo se $v_i \wedge v_j \geq 0$.*

Il complementare di S_+ nel piano privato dell'origine è l'insieme S_- ($S_- = \mathbb{R}^2 \setminus (\{O\} \cup S_+)$). I punti di S_- hanno ordinata negativa oppure nulla e ascissa negativa ($y < 0$ oppure $y = 0$ e $x < 0$) e anche per due punti di S_- si può dire che *l'angolo polare di v_i non supera quello di v_j se e solo se $v_i \wedge v_j \geq 0$.*

Poiché gli angoli polari dei punti di S_+ appartengono all'intervallo semiaperto $[0, \pi)$ e quelli di S_- a $[\pi, 2\pi)$ basta far seguire alla versione ordinata per angolo polare dei punti di S_+ quella dei punti di S_- per ottenere il risultato desiderato.

Se la scissione in S_+, S_- desta perplessità si consideri la figura a destra in cui $v_1 \wedge v_2 > 0$, $v_2 \wedge v_3 > 0$, $v_3 \wedge v_4 > 0$, $v_4 \wedge v_5 > 0$, $v_5 \wedge v_1 > 0$. Si deduce che due vertici consecutivi in $(v_1, v_2, v_3, v_4, v_5)$ sono ordinati anche per angolo polare, ma non si può dedurre che l'ordine corretto è $(v_1, v_2, v_3, v_4, v_5)$.



L'ordine corretto è $(v_1, v_4, v_2, v_5, v_3)$. Infatti la versione ordinata di $S_+ = \{v_1, v_2, v_4\}$ è $\{v_1, v_4, v_2\}$, quella di $S_- = \{v_3, v_5\}$ è $\{v_5, v_3\}$ e quella dell'insieme considerato è $\{v_1, v_4, v_2, v_5, v_3\}$.

Spesso è utile considerare gli angoli polari non nell'intervallo $[0, 2\pi)$ ma in $[-\pi, +\pi)$. In questo caso gli angoli polari dei punti di S_- sono minori di quelli di S_+ e l'ordinamento per angolo polare crescente in $[-\pi, +\pi)$ è immediato: basta far seguire alla versione ordinata di S_- la versione ordinata di S_+.

3 Grafi planari

3.1 Introduzione 21
3.2 Poliedri e grafi planari 24
3.3 Duale di un poligono semplice triangolato 25

Sommario

Dopo l'introduzione del concetto di grafo planare sono esposte la formula di Eulero (Teor. 3.4) e le connessioni tra poliedri, grafi planari e solidi platonici. La Sez. 3.3 introduce il grafo duale di un grafo planare e ne dimostra la 3-colorabilità che ha, come conseguenza, la soluzione del celebre problema della 'galleria d'arte'.

3.1 Introduzione

Ricordiamo che un grafo è una coppia $\mathscr{G} = (V, E)$ con V insieme dei vertici ed $E \subseteq V \times V$ insieme degli spigoli. Nulla è specificato sulla natura degli elementi di V o sul modo in cui due elementi di V sono accoppiati per formare uno spigolo. Usatissima la rappresentazione di V come insieme di punti del piano e quella di E come insieme di curve semplici che connettono i due vertici della coppia di vertici.

Un grafo è detto *planarizzabile* se esiste una disposizione di V nel piano e di E con curve semplici che si intersecano solo in corripondenza di un vertice. Un grafo con una rappresentazione di questo tipo è detto *grafo planare*.

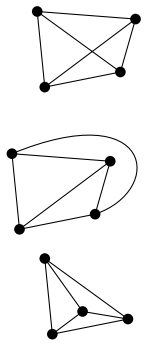
Siamo però abituati a rappresentare i grafi planari con punti (vertici) e segmenti (spigoli) piuttosto che curve semplici: questo è lecito grazie al seguente

Teorema 3.1 (I. Fary, 1948) *Ogni grafo planare ammette una rappresentazione con spigoli rettilinei.*

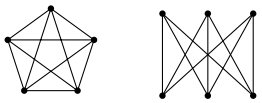
I grafi planari a spigoli rettilinei sono spesso indicati con l'acronimo *PSLG, Planar Straight-Line Graph*.

Esempio 3.2 Consideriamo il grafo completo con quattro vertici.

È istintivo tracciare un quadrilatero e poi le due diagonali ottenendo una rappresentazione non planare.
Ma basta trasformare il segmento che unisce il vertice sud-est con quello nord-ovest in una curva semplice per ottenere un grafo planare con spigoli *non* rettilinei.
Il Teor. 3.1 assicura l'esistenza di una rappresentazione con spigoli rettilinei. Per ottenerla basta spostare verso sud-ovest il vertice nord-est. ■



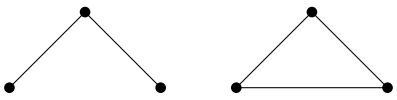
Teorema 3.3 (K. Kuratowski, 1930) *Un grafo ammette una rappresentazione planare se e solo se non contiene sottografi omeomorfi a $K_{3,3}$ o a K_5 (grafo bipartito con 6 vertici e grafo completo con 5 vertici).*



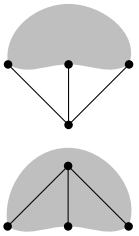
Nella trattazione dei grafi planari è importante considerare le facce, oltre a vertici e spigoli. In assenza di vertici isolati o con un solo spigolo incidente (vertici di *grado* minore di due) le facce sono delimitate da poligoni semplici, e viceversa. Un grafo planare è *connesso* se per ogni coppia di vertici esiste un cammino che li congiunge. La celebre formula di Eulero coinvolge tutte queste quantità:

Teorema 3.4 (Formula di Eulero) *In un grafo planare connesso e non vuoto con $v0$ vertici, e spigoli, f facce si ha $v - e + f = 2$.*

Dim. È facile verificare la validità del risultato per grafi con un solo vertice ($v = 1$, $e = 0$, $f = 1$) o con due vertici connessi da uno spigolo ($v = 2$, $e = 1$, $f = 1$). Con tre vertici sono possibili solo due grafi connessi: si passa dal primo al secondo aggiungendo uno spigolo e questa aggiunta provoca la comparsa di una faccia limitata



Supponiamo che la formula valga per i grafi connessi con non piú di $v - 1$ vertici e consideriamo un nuovo vertice di grado g (cioè con g spigoli incidenti). Questo nuovo vertice può stare nella faccia illimitata o in una faccia limitata, ma in ogni caso introduce g nuovi spigoli che generano $g - 1$ nuove facce limitate.



Prima dell'aggiunta di questo vertice la formula era valida (ipotesi induttiva): $(v-1)-e+f=2$; ora il numero di vertici è cresciuto di uno, quello degli spigoli di g, quello delle facce di $g-1$ e per il nuovo grafo planare si ha

$$((v-1)+1)-(e+g)+(f+g-1)=v-e+f=2. \qquad \blacksquare$$

Esempio 3.5 In un poligono semplice $\mathscr{P}$ con n vertici ci sono n spigoli. $\mathscr{P}$ è un grafo planare con due facce (interno ed esterno di $\mathscr{P}$). La formula di Eulero dice che $n-n+2=2$. Triangolando $\mathscr{P}$ si ottiene un altro grafo planare, ancora con n vertici, ma gli spigoli sono $n+d$ (n lati e d diagonali) e le facce sono $t+1$ (la faccia illimitata e t triangoli). In questo caso la formula di Eulero dice che $n-(n+d)+(t+1)=2$ cioè $t=d+1$ (cfr. Proposizione 1.12). Per arrivare al risultato $t=n-2$ e $d=n-3$ si può ricorre al seguente trucco: si immagini di visitare uno spigolo e di collocare un contrassegno sulla faccia a sinistra e uno su quella a destra. Se ogni spigolo viene visitato una volta sola allora i contrassegni utilizzati sono $2(n+d)$; ogni triangolo contiene tre contrassegni e la faccia illimitata ne contiene n. Quindi $2(n+d)=3t+n$ e da $d=t-1$ si ricava $t=n-2$ e $d=n-3$. $\blacksquare$

La formula di Eulero può essere estesa al caso piú generale di un grafo planare con un numero arbitrario ($c\geq 1$) di componenti connesse. Se la i-esima componente connessa ha v_i vertici, e_i spigoli e f_i facce deve essere $v_i-e_i+f_i=2$. La faccia illimitata è comune ad ogni componente connessa quindi il numero totale di facce è $\sum_{i=1}^{c} f_i-(c-1)$, con $\sum_{i=1}^{c} v_i$ vertici e $\sum_{i=1}^{c} e_i$. Quindi

Proposizione 3.6 *In un grafo planare con v vertici, e spigoli, f facce e c componenti connesse si ha*

$$v-e+f=c+1. \tag{3.1}$$

Illustriamo ora una serie di legami tra numero v di vertici, e di spigoli e f di facce di un PSLG.

Proposizione 3.7 *In un PSLG, non necessariamente connesso con v vertici di grado $g_1, g_2, \ldots, g_v$ si ha*

$$2e=\sum_{i=1}^{v} g_i. \tag{3.2}$$

Dim. Si immagini di visitare gli e spigoli e di contrassegnare, durante la visita del generico spigolo, i suoi due vertici. Alla fine saranno stati utilizzati $2e$ contrassegni e il vertice v_i, in cui ci sono g_i spigoli incidenti, ha esattamente g_i contrassegni. $\blacksquare$

Proposizione 3.8 *Per un grafo planare con e spigoli, f facce, c componenti connesse e v vertici di grado $\geq g$ valgono le seguenti disuguaglianze:*

$$\begin{gathered} 2e\geq 3v, \quad (g-2)v\leq 2(f-(c+1))\leq 2(f-2), \\ (g-2)e\leq g(f-(c+1)\leq g(f-2). \end{gathered} \tag{3.3}$$

Dim. Dalla (3.2) con $g_i \geq g$ si ha $2e \geq gv$. Dalla (3.1) moltiplicata per due si ha $2v + 2f - 2(c+1) = 2e \geq gv$ e $(g-2)v \leq 2(f-(c+1)) \leq 2(f-2)$. Moltiplicando la (3.1) per g si ha $gv - ge + gf = g(c+1)$ con $gv \leq 2e$, quindi $(g-2)e \leq g(f-(c+1)) \leq g(f-2)$. ∎

Proposizione 3.9 *In un grafo planare con almeno tre vertici ($v \geq 3$), e spigoli ed f facce si ha $e \leq 3v - 6$ e $3f \leq 2e$.*

Dim. Il numero di spigoli è massimo quando il grafo è connesso, senza vertici di grado uno e con facce limitate triangolari. Se ogni spigolo emette un contrassegno a destra ed uno a sinistra in totale vengono emessi $2e$ contrassegni ed ogni faccia ne riceve un numero uguale al numero di lati che la delimitano: $2e = \sum_{i=1}^{f} l_i = 3(f-1) + l_\infty$ dove l_∞ è il numero di lati della faccia illimitata. Da $l_\infty \geq 3$ si ha $3f \leq 2e$.

Sostituendo questa disuguaglianza nella formula di Eulero $e = v + f - 2$ si ottiene $e \leq v - 2 + 2/3\, e$ e $e \leq 3v - 6$. ∎

3.2 Poliedri e grafi planari

Possiamo informalmente definire un poliedro come un solido delimitato da un insieme finito di facce poligonali (poligoni semplici).

Un poliedro è detto semplice se non ha 'buchi'. Cubo, tetraedro, prisma sono esempi di poliedri semplici. Immaginiamo un poliedro appoggiato su un tavolo in modo che la superficie di contatto poliedro-tavolo sia una faccia del poliedro, quindi un poligono semplice. Praticando un forellino all'interno di questa faccia e deformando la superficie rimanente in maniera graduale, senza strappi o pieghe, si arriva ad un grafo planare con lo stesso numero di vertici, spigoli e facce (la faccia rimossa dal poliedro di partenza diventa la faccia illimitata). Se gli spigoli ottenuti non sono rettilinei il Teorema di Fary (Teor. 3.1) ne permette la trasformazione in un PSLG.

Platone (427?–347 a. C.) si interessò della ricerca dei poliedri semplici e regolari; regolari nel senso che

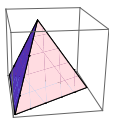
- le facce sono poligoni regolari congruenti con ℓ lati e
- ogni vertice ha grado g.

In un solido con queste proprietà — spesso detto *solido platonico* — dev'essere $\ell, g \geq 3$. Se ci sono f facce e v vertici, dev'essere $2e = gv = \ell f$ per la (3.2) e perché ogni spigolo è incidente in due facce.

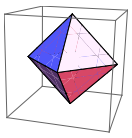
Sostituendo $v = 2e/g$ e $f = gv/\ell$ nella formula di Eulero $v - e + f = 2$ si ottiene un'espressione di v in funzione di ℓ e g:

$$v = \frac{4\ell}{2\ell - (\ell-2)g}. \tag{3.4}$$

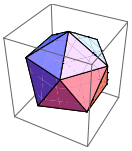
Interessano valori interi di v al variare di ℓ e g con il vincolo $\ell, g \geq 3$. Con $\ell = 3$ si ha $v = 12/(6-g)$ e gli unici valori possibili per g sono 3, 4, 5. $g = 3$ implica $v = 4$, $f = 4$, $e = 6$. Il poliedro regolare con quattro vertici di grado tre, quattro facce triangolari e sei spigoli è il tetraedro.



$g = 4$ implica $v = 6$, $f = 8$, $e = 12$. Questo è un poliedro con sei vertici di grado quattro, otto facce triangolari e dodici spigoli: l'ottaedro.

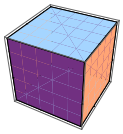


Con $g = 5$ si ottiene un poliedro con venti facce triangolari, dodici vertici, trenta spigoli (icosaedro).

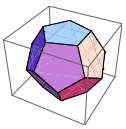


Per valori di ℓ superiori a tre il numero di possibilità cala drasticamente.

$\ell = 4$ implica $v = 16/(8-2g)$ e l'unico modo di ottenere v intero è $g = 3$. In questo caso ci sono otto vertici, sei facce quadrate e dodici spigoli (cubo, detto anche esaedro).



$\ell = 5$ implica $v = \frac{20}{10-3g}$. Anche in questo caso $g = 3$ è l'unica possibilità e questo poliedro ha dodici facce pentagonali, venti vertici e trenta spigoli. È il penta-dodecaedro.



Con $\ell \geq 6$ e $g \geq 3$ il denominatore della (3.4) non può essere positivo, quindi si può concludere che esistono esattamente cinque solidi platonici.

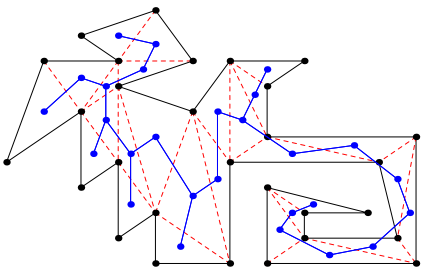
3.3 Duale di un poligono semplice triangolato

Si è visto che la triangolazione di un poligono semplice $\mathscr{P}$ con n vertici è un PSLG $\mathscr{T}$ con $2n-3$ spigoli (gli n lati di $\mathscr{P}$ e le sue $n-3$ diagonali) e $n-1$ facce ($n-2$ triangoli e la faccia illimitata).

Il grafo duale $\mathscr{D}$ di $\mathscr{T}$ è definito come segue:

- al triangolo τ di $\mathscr{T}$ viene associato biiettivamente un vertice v_τ di $\mathscr{D}$;
- ad una diagonale δ di $\mathscr{T}$ è associato uno spigolo in $\mathscr{D}$ in questo modo: se δ è comune ai triangoli τ' e τ'' allora lo spigolo associato collega $v_{\tau'}$ e $v_{\tau''}$.

La faccia illimitata di $\mathscr{T}$ e i lati di $\mathscr{P}$ non hanno alcun corrispettivo in $\mathscr{D}$.



Proposizione 3.10 *$\mathscr{D}$ è un grafo planarizzabile, connesso con la struttura di albero.*

Dim. Si collochi v_τ nel baricentro del triangolo τ e alla diagonale δ comune ai triangoli τ', τ'' si associ la spezzata da $v_{\tau'}$ al punto medio di δ e da questo a $v_{\tau''}$. È chiaro che tali spezzate possono incontrarsi solamente in corrispondenza

di vertici di $\mathscr{D}$, quindi $\mathscr{D}$ è planarizzabile e trasformabile in un PSLG grazie al teorema di Fary (Teor. 3.1).

È facile vedere che il duale $\mathscr{D}$ di un poligono semplice con 3 o 4 lati è connesso. Se i duali dei poligoni semplici triangolati con $3, 4, \ldots, n-1$ vertici siano connessi si consideri la triangolazione $\mathscr{T}$ di un poligono semplice con n vertici. Una diagonale $v'v''$ di $\mathscr{T}$ partiziona $\mathscr{T}$ in due triangolazioni $(\mathscr{T}_1, \mathscr{T}_2)$ di poligoni semplici con meno di n lati e i loro duali $\mathscr{D}_1, \mathscr{D}_2$ sono connessi grazie all'ipotesi induttiva. Alla diagonale $v'v''$ corrisponde in $\mathscr{D}$ uno spigolo con un estremo in $\mathscr{D}_1$ e uno in $\mathscr{D}_2$ e per ogni coppia di vertici esiste un cammino che li connette. $\mathscr{D}$ è connesso, ha $n-2$ vertici, $n-3$ spigoli e la sola faccia illimitata ($(n-2)-(n-3)+f = 2$ dalla formula di Eulero (3.1)). Pertanto $\mathscr{D}$ è un grafo connesso e aciclico, cioè un albero. ∎

Si può dire qualcosa di piú.

Proposizione 3.11 *L'albero $\mathscr{D}$ ha vertici di grado compreso tra uno e tre e, se $n \geq 4$, esistono almeno due vertici di grado uno (orecchie).*

Dim. In $\mathscr{D}$ connesso non ci sono vertici isolati e il grado di un vertice è uguale al numero di lati del corrispondente triangolo che sono diagonali di $\mathscr{P}$, perciò non piú di tre.

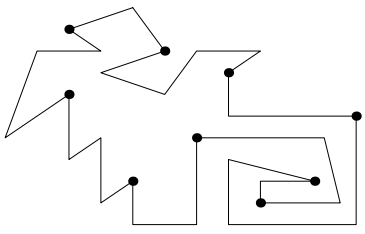
$\mathscr{D}$ ha $n-3$ spigoli, $n-2$ vertici e se ogni vertice avesse grado ≥ 2 la (3.2) porterebbe alla contraddizione $2(n-3) = \sum_{i=1}^{n-2} g_i \geq 2(n-2)$. Deve perciò esistere un vertice w di grado uno in $\mathscr{D}$.

Per scovare l'altro vertice di grado uno iniziamo una visita di $\mathscr{D}$ da w, marcandolo e spostandoci nel suo unico adiacente w'. Quando la visita raggiunge un nuovo vertice, il vertice viene marcato e se ne esamina il grado. Se è uno è stato trovato un secondo vertice di grado uno e la visita si arresta. Se il grado è maggiore di uno esiste almeno uno spigolo che lo connette a un vertice non marcato[1] e la visita può proseguire verso un vertice non marcato.

La visita prima o poi deve arrestarsi perché il numero di vertici è finito: non può arrestarsi su un vertice di grado maggiore di uno perché, come già osservato, l'albero avrebbe un ciclo. Allora la visita deve arrestarsi su un vertice di grado uno diverso da quello di partenza. ∎

Possiamo ora dimostrare, costruttivamente, che ogni triangolazione di un poligono semplice è 3-colorabile, cioè che è possibile colorare i vertici di $\mathscr{T}$ con tre colori e in modo che vertici adiacenti abbiano colori diversi. Basta iniziare una ricerca in profondità (DFS) da un vertice di $\mathscr{D}$ di grado uno e procedere come in Fig. 3.1.

Il problema della galleria d'arte è risolto: basta prendere il colore meno usato e sistemare una guardia su ogni vertice con quel colore. Nel caso peggiore bastano $\lfloor n/3 \rfloor$ guardie. Il solito poligono ha 29 vertici e può essere sorvegliato da 9 guardie disposte come in figura.



[1] in caso contrario l'albero avrebbe un ciclo!

Tricolorazione di un poligono semplice triangolato.
Input: la triangolazione $\mathscr{T}$ di un poligono semplice $\mathscr{P}$ e il suo albero duale $\mathscr{D}$;
Output: a ogni vertice di $\mathscr{P}$ è associato un colore $\in \{R, G, B\}$

```
1 Tricolorazione(𝒯, 𝒟);
2 ∀w ∈ 𝒟 do visited (w) ← False;
3 scegli un vertice w ∈ 𝒟 di grado uno;
4 colora i vertici del triangolo 𝒯_w associato con R, G e B;
5 visited (w) ← True;
6 ∀w' ∈ 𝒟 adiacente a w t.c. visited(w) = False do DFS(w', 𝒟, 𝒯);
7 end;
```

```
1 DFS(v, 𝒟, 𝒯); (* v ∈ 𝒟 vertice da cui proseguire la DFS *)
2 (* il triangolo τ ∈ 𝒯 associato a v ha due vertici colorati *)
3 visited(v) ← True;
4 colora il vertice non colorato di τ nell'unico modo possibile
5 ∀w ∈ 𝒟 adiacente a v t.c. visited(w) = False do DFS(w, 𝒟, 𝒯);
6 end
```

Figura 3.1: tricolorazione della triangolazione di un poligono semplice.

4 DCEL

4.1 Versione originale 29
4.2 Versione con semispigoli 30
4.2.1 Tabella dei vertici 31
4.2.2 Tabella dei semispigoli 31
4.2.3 Tabella delle facce 31
4.2.4 Campi 33
4.3 Costruzione 33
4.3.1 Preprocessing 34
4.3.2 Costruzione: fase I 34
4.3.3 Costruzione: fase II 36
4.4 Caso generale 39

Sommario

Quando si considerano grafi planari è spesso importante disporre di informazioni relative non solo a vertici e spigoli, ma anche alle facce. Tali informazioni sono implicite in ogni descrizione di un grafo — ad esempio nelle liste di adiacenza vertici–vertici — ma la loro manipolazione non è agevole.

La DCEL (Doubly-Connected Edge List) è una struttura dati che esplicita le relazioni tra vertici e spigoli e tra spigoli e facce. La sua costruzione per un grafo connesso con n vertici descritto dalle liste di adiacenza tra vertici è fattibile in tempo e spazio lineari (quindi ottimi) dopo un preprocessing che richiede $O(n \log n)$.

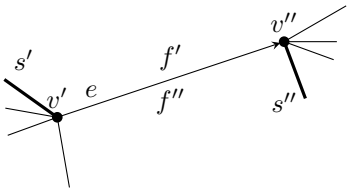
L'idea originaria della DCEL è esposta nella Sez. 4.1. Le Sez. 4.2 e 4.3 trattano la piú recente versione con semispigoli (esposta, senza troppi dettagli, in [2]) e la sua costruzione sotto certe ipotesi, di comodo; ipotesi discusse e rimosse nella Sez. 4.4.

4.1 Versione originale

La DCEL (acronimo di *Doubly-Connected Edge List*, lista di spigoli doppiamente connessa) è un'efficiente struttura dati per la rappresentazione di un grafo planare $\mathscr{G}$ con spigoli rettilinei (PSLG). Rispetto alle strutture tradizionali, quali liste di adiacenza, matrici di adiacenza,... ha il pregio di mettere in luce esplicitamente le facce e di facilitare la soluzione di problemi che le coinvolgono, quali, ad esempio: *fornire una lista ordinata dei vertici e/o degli spigoli che delimitano una faccia; elencare in verso orario o antiorario le facce incidenti in un vertice; dato un punto di interrogazione che non coincide con un vertice e non appartiene a uno spigolo, a quale faccia appartiene?*

La versione originale della DCEL ([3], [4] Sez. 1.2.3.2) è la seguente: una lista doppiamente legata di record (un record per ogni spigolo di $\mathscr{G}$) con record composti da sei campi. Orientando arbitrariamente uno spigolo e restano definite le seguenti sei quantità

- vertici sorgente (v') e destinazione (v'') di e;
- faccia sinistra (f') e destra (f'') di e;
- s' è il primo spigolo incidente in v' incontrato con una rotazione antioraria di e attorno al suo vertice sorgente;
- s'' è il primo spigolo con un estremo in v'' incontrato con una rotazione antioraria di e attorno al suo vertice destinazione.



Disponendo delle liste di adiacenza non è facile fornirne la lista degli spigoli che delimitano una faccia di $\mathscr{G}$. Disponendo della DCEL le cose sono notevolmente piú semplici:

---

fSpigoli: determinazione degli spigoli di una faccia f di $\mathscr{G}$.
Input: DCEL $\mathscr{D}$ di $\mathscr{G}$ e una faccia f di $\mathscr{G}$;
Output: una lista oraria $\mathscr{L}$ degli spigoli che delimitano f.

```
1  fSpigoli(𝒟, f);
2  scandisci 𝒟 alla ricerca di uno spigolo e t.c. f'(e) = f ∨ f''(e) = f;
3  ℒ ← {}; e₀ ← e; (* ricorda lo spigolo iniziale *)
4  repeat
5     ℒ ← Append(ℒ, e);
6     If f = f'(e) then e ← s'(e)
7     else e ← s''(e)
8  until e = e₀;
9  Return(ℒ)
10 end;
```

4.2 Versione con semispigoli

Questa versione della DCEL ([2] Sez. 2.2) differisce da quella originale per vari motivi: ad ogni spigolo è associata una coppia di semispigoli gemelli; prevede tre tabelle (con le informazioni su vertici, semispigoli e facce).

I semispigoli associati ad uno spigolo (non orientato) $v'v''$ sono due spigoli orientati: uno va da v' a v'', l'altro — detto gemello — va da v'' a v'.

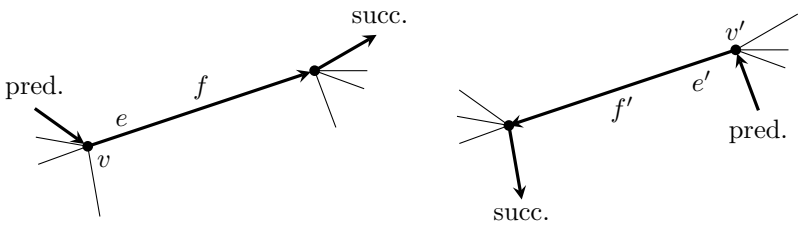
Nelle Sez. 4.2.1, 4.2.2 e 4.2.3 vengono descritti i campi delle tabelle e le loro principali caratteristiche, mentre la Sez. 4.2.4 è dedicata al riassunto di tutti i campi dei record, all'introduzione di alcune funzioni ausiliarie di etichettatura e alla valutazione dello spazio occupato dalle tre tabelle.

4.2.1 Tabella dei vertici

La tabella dei vertici contiene un record per vertice; ogni record è composto da tre campi: l'etichetta del vertice v in questione, le coordinate (x, y) della sua posizione nel piano e l'etichetta di un semispigolo uscente da v.

4.2.2 Tabella dei semispigoli

La tabella dei semispigoli contiene un record per ogni semispigolo e ogni record ha sei campi.



In figura alcune quantità relative ad un semispigolo e (a sinistra) e al suo gemello e' (a destra).

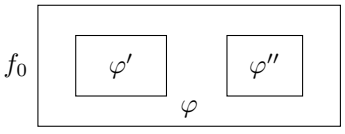
I primi tre campi di un record di semispigolo rappresentano l'etichetta del *semispigolo* stesso, quella del suo vertice *origine* e quella del suo *gemello*. Seguono le etichette del semispigolo predecessore e di quello successore — fondamentali nella descrizione delle facce: il *predecessore* di un semispigolo e è il semispigolo che termina nel vertice origine di e e con la stessa faccia sinistra di e; il successore è il semispigolo uscente dal vertice di arrivo di e e con la stessa faccia sinistra di e. L'ultimo campo contiene l'etichetta della *faccia sinistra* f di e.

Per descrivere il *semispigolo gemello* e' di e bastano poche osservazioni: l'orientazione è opposta a quella di e, quindi il vertice destinazione di e è il vertice origine di e'; la faccia destra di e è la faccia sinistra di e'. Predecessore e successore di e' delimitano la faccia *destra* di e. Passando da un semispigolo e al suo successore viene percorsa la frontiera della faccia sinistra (in verso antiorario), mentre passando dal gemello e' di e al suo successore viene percorsa la faccia destra di e, sempre in verso antiorario.

4.2.3 Tabella delle facce

La descrizione delle facce coinvolge qualche sottigliezza. Per semplificare la presentazione escludiamo per il momento la presenza di vertici di grado zero (vertici isolati) o di grado uno e supponiamo che le frontiere delle facce siano frontiere di poligoni semplici. Queste ipotesi verranno completamente rimosse nella Sez. 4.4.

È chiaro che ogni grafo planare ha almeno una faccia, quella illimitata (f_0). Nella figura a destra ci sono anche tre facce limitate $(\varphi, \varphi', \varphi'')$.

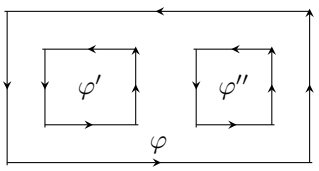


Per la faccia limitata φ possiamo distinguere una frontiera esterna (il rettangolo piú grande) e due frontiere interne (rettangoli interni) che corrispondono a

'buchi' di φ. Le facce φ' e φ'' non presentano buchi e sono completamente individuate dalla loro frontiera esterna.

Per la faccia illimitata la situazione si ribalta: manca una frontiera esterna ed è internamente delimitata da φ. Da questa discussione poco formale sono emersi i due importanti concetti di frontiera esterna e frontiere interne.

Queste frontiere possono essere orientate in modo che una faccia sia la regione di piano che sta a sinistra di ogni spigolo della sua frontiera esterna e, sempre a sinistra, degli spigoli di ogni sua frontiera interna.



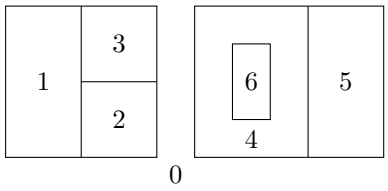
Pertanto le frontiere esterne sono orientate in verso antiorario e quelle interne in verso orario.

È stato detto che la faccia illimitata non ha frontiera esterna, mentre ogni faccia limitata ha una sola frontiera esterna. Il numero di frontiere interne di una faccia non è superiormente limitabile a priori, quindi la tabella delle facce della DCEL sarà costituita da tanti record quante sono le facce e per ogni faccia f saranno presenti i seguenti campi:

- l'etichetta della faccia,
- l'etichetta di un semispigolo della frontiera esterna che abbia f come faccia (sinistra),
- una lista di 'buchi' (ognuno individuato da un'etichetta di un semispigolo che abbia f come faccia sinistra).

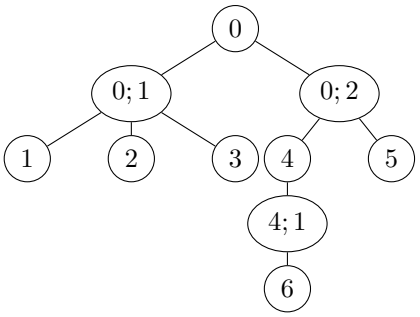
Se si tiene presente il problema di point location in cui si vuole determinare la faccia che contiene un punto di interrogazione q, emerge la possibilità di una tabella delle facce leggermente piú complessa, ma che può portare a significativi risparmi.

Nella figura a fianco la faccia illimitata (0) ha due buchi (che indicheremo con $0;1$ e $0;2$). Il buco $0;1$ contiene a sua volta tre facce prive di buchi (1, 2 e 3); il buco $0;2$ contiene la faccia 5 (senza buchi) e la faccia 4 che, a sua volta, contiene la faccia 6 (senza buchi).



Emerge una struttura ad albero in cui la radice 0 rappresenta la frontiera esterna della faccia illimitata e i suoi due figli ne rappresentano i buchi.

Il vertice $0;1$, contenendo tre facce prive di buchi, ha tre figli che sono foglie dell'albero e che rappresentano le frontiere esterne delle facce 1, 2 e 3.
Il buco $0;2$ contiene due facce: il vertice $0;2$ ha due figli associabili alle frontiere esterne di 4 e 5.
La frontiera interna di 4, $4;1$, è rappresentata da un figlio del vertice 4 ed ha a sua volta un figlio, 6, che rappresenta la frontiera esterna della faccia 6. Questa faccia, non vendo buchi, è associata ad una foglia dell'albero.



4.2.4 Campi

Una DCEL è completamente specificata dalle seguenti funzioni, interpretabili anche come campi di record, array, puntatori,...

Per il generico vertice con etichetta ℓ: $\mathsf{vCoords}(\ell)$ contiene le coordinate del vertice, $\mathsf{vOutEdge}(\ell)$ l'etichetta di un semispigolo uscente da ℓ.

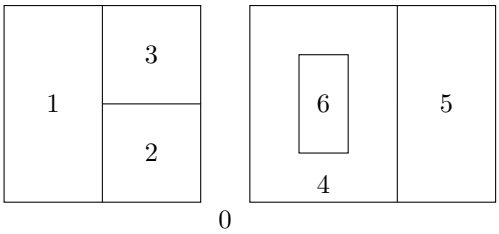
Per il generico semispigolo con etichetta ℓ: $\mathsf{eSource}(\ell)$ è l'etichetta del vertice da cui il semispigolo esce; $\mathsf{eTwin}(\ell)$ è quella del semispigolo gemello; $\mathsf{ePred}(\ell)$ è l'etichetta dell'unico semispigolo che termina in $\mathsf{eSource}(\ell)$ e ha la stessa faccia sinistra e $\mathsf{eSucc}(\ell)$ è quella dell'unico semispigolo uscente dal vertice destinazione di ℓ e con la stessa faccia sinistra; $\mathsf{eFace}(\ell)$ è l'etichetta della faccia sinistra di ℓ.

Per la generica faccia con etichetta ℓ: $\mathsf{fOuter}(\ell)$ è l'etichetta di un semispigolo della frontiera esterna di ℓ (non definito per la faccia illimitata); $\mathsf{fInner}(\ell)$ è una lista che, se non vuota, ha la seguente struttura: l'elemento i-esimo della lista è del tipo $b_i \to \{f_{i1}, f_{i2}, \ldots\}$: b_i è l'etichetta di un semispigolo sulla i-esima frontiera interna (buco) di ℓ e $\{f_{i1}, f_{i2}, \ldots\}$ è la lista delle facce con frontiera esterna connessa con la frontiera del buco.

Si suppone inoltre che siano definite tre funzioni iniettive per la creazione delle etichette: $\mathsf{vLabel}(i)$ associa all'intero positivo i l'etichetta da attribuire all'i-esimo vertice di $\mathscr{V}$; $\mathsf{eLabel}(i, j)$ associa alla coppia ordinata di interi positivi (i, j) l'etichetta del semispigolo che va dal vertice i al vertice j. $\mathsf{fLabel}(i)$ associa all'intero positivo i l'etichetta della faccia i-esima. L'etichetta $\mathsf{fLabel}(0)$ è riservata alla faccia illimitata. Spesso vLabel e fLabel sono la funzione identica e $\mathsf{eLabel}(i, j)$ un'espressione del tipo e_k, $\overrightarrow{e_k}$, $\overrightarrow{e_{i,j}}$, $e_{i,j}$.

Per quanto riguarda le dimensioni delle tabelle della DCEL di un PSLG con n vertici e m spigoli si hanno n record associati ai vertici e $2m$ ai semispigoli. Il numero di facce dipende dal numero c di componenti connesse:[1] dalla formula di Eulero $n - m + f = c + 1$, si ha $f = m - n + c + 1$.

Nell'esempio a fianco $n = 18$, $m = 21$, $c = 3$ e $f = 7$.



4.3 Costruzione

Abbozziamo un algoritmo per la costruzione della DCEL di un grafo planare descritto per mezzo di

- una lista $\mathscr{V} = \{(x_1, y_1), (x_2, y_2), \ldots\}$ di coordinate dei vertici e di
- un insieme di liste di adiacenza $\mathscr{A}$. $\mathscr{A}(i) = \{j_{i1}, j_{i2}, \ldots\}$ è una lista di interi corrispondenti ai vertici adiacenti al vertice i: esiste uno spigolo da (x_i, y_i) a (x_j, y_j) se e solo se j compare nella i-esima lista di adiacenza ($j \in \mathscr{A}(i)$) e i compare nella j-esima lista di adiacenza ($i \in \mathscr{A}(j)$).

[1] facilmente calcolabile in tempo lineare con una ricerca in profondità

Se n è il numero di vertici e m_i la cardinalità di $\mathscr{A}(i)$ allora il numero m di spigoli è $\frac{1}{2}\sum_{i=1}^{n} m_i$ e $m \in O(n)$ in ogni grafo planare.

Nella descrizione della costruzione della DCEL continueremo a supporre che ogni vertice abbia grado ≥ 2 e che le facce siano delimitate da poligoni semplici.

4.3.1 Preprocessing

La costruzione della DCEL a partire da $\mathscr{V}$ e $\mathscr{A}$ procede abbastanza speditamente se preceduta da un preprocessing che riordini per angolo polare crescente θ ($0 \leq \theta < 2\pi$) ogni lista di adiacenza e associ ad ogni elemento il predecessore e il successore (in verso antiorario).

Sia $\mathscr{S}$ la versione di $\mathscr{A}$ cosí ordinata per angolo polare crescente; ccwPred e ccwSucc le funzioni che, dato un vertice v ed uno spigolo e incidente in v, danno lo spigolo incidente in v che precede o segue e in verso antiorario.

---

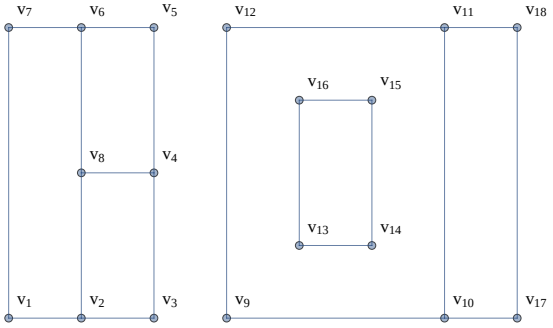
preprocess: preliminari per la costruzione della DCEL
Input: vertici $\mathscr{V}$ e liste di adiacenza $\mathscr{A}$ di $\mathscr{G}$;
Output: liste riordinate $\mathscr{S}$ e funzioni ccwPred, ccwSucc.

```
 1 preprocess(𝒱, 𝒜);
 2 For i ← 1 to n do begin
 3    𝒮(i) ← PolarSort (𝒜(i)); (* angoli polari in [0, 2π) *)
 4    For h ← 1 to m_i do begin
 5       ℓ ← eLabel(i, 𝒮(i, h)); (* ℓ variabile locale *)
 6       If h > 1 then ccwPred(ℓ) ← eLabel(i, 𝒮(i, h − 1))
 7       else ccwPred(ℓ) ← eLabel(i, 𝒮(i, m_i))
 8       If h = m_i then ccwSucc(ℓ) ← eLabel(i, 𝒮(i, 1));
 9       else ccwSucc(ℓ) ← eLabel(i, 𝒮(i, h + 1))
10    end;
11 end;
12 Return(𝒮, ccwPred, ccwSucc);
13 end;
```

Il punto 3 richiede $O(m_i \log m_i)$, il ciclo 4–10 $O(m_i)$, quindi il tempo per il preprocessing è $\sum_{i=1}^{n} O(m_i \log m_i) = O(\sum m_i \log m_i) = O(m \log m)$.

4.3.2 Costruzione: fase I

Dopo questo preprocessing la costruzione delle tabelle dei vertici e dei semispigoli diventa piuttosto semplice, almeno per i campi diversi da eFace. Durante questa costruzione faremo spesso riferimento a questo grafo:



► **Vertici.** Le informazioni sui vertici sono facilmente costruibili in $O(n)$ con l'algoritmo che segue in cui si sfrutta l'ipotesi dell'assenza di vertici di grado zero (se i ha grado 0 la lista $\mathscr{S}(i)$ è vuota e l'assegnamento al Punto 5 privo di senso). Da ricordare anche che vOutEdge(v) individua il semispigolo uscente da v e con angolo polare massimo.

buildVertexTable: costruzione della tabella dei vertici
Input: vertici $\mathscr{V}$ e liste di adiacenza $\mathscr{S}$;
Output: tabella dei vertici.

```
1 buildVertexTable(𝒱, 𝒮);
2 For i ← 1 to n do begin
3    ℓ ← vLabel(i); (* ℓ variabile locale *)
4    vCoords(ℓ) ← (x_i, y_i); (* i-esimo elemento di 𝒱 *)
5    vOutEdge(ℓ) ← eLabel(i, 𝒮(i, m_i))
6 end;
7 Return(vCoords, vOutEdge);
8 end;
```

v	vCoords	vEdge	v	vCoords	vEdge
v_1	{0, 0}	e_{v_1,v_7}	v_{10}	{6, 0}	e_{v_{10},v_9}
v_2	{1, 0}	e_{v_2,v_1}	v_{11}	{6, 4}	$e_{v_{11},v_{10}}$
v_3	{2, 0}	e_{v_3,v_2}	v_{12}	{3, 4}	e_{v_{12},v_9}
v_4	{2, 2}	e_{v_4,v_3}	v_{13}	{4, 1}	$e_{v_{13},v_{16}}$
v_5	{2, 4}	e_{v_5,v_4}	v_{14}	{5, 1}	$e_{v_{14},v_{13}}$
v_6	{1, 4}	e_{v_6,v_8}	v_{15}	{5, 3}	$e_{v_{15},v_{14}}$
v_7	{0, 4}	e_{v_7,v_1}	v_{16}	{4, 3}	$e_{v_{16},v_{13}}$
v_8	{1, 2}	e_{v_8,v_2}	v_{17}	{7, 0}	$e_{v_{17},v_{10}}$
v_9	{3, 0}	$e_{v_9,v_{12}}$	v_{18}	{7, 4}	$e_{v_{18},v_{17}}$

► **Semispigoli.** Le informazioni relative ai semispigoli — a parte eFace — vengono costruite in due passi. Nel primo vengono definiti i valori dei campi eSource e eTwin per ogni semispigolo, nel secondo vengono riempiti i campi ePred e eSucc sfruttando pesantemente le informazioni ricavate nel preprocessing.

buildHalfEdgeTableI: prima fase della costruzione della tabella dei semispigoli
Input: vertici $\mathscr{V}$ e liste di adiacenza $\mathscr{S}$;
Output: eSource,eTwin.

```
1  buildHalfEdgeTableI(𝒱, 𝒮);
2  For i ← 1 to n do begin
3     For h ← 1 to m_i do begin
4        ℓ ← eLabel(i, 𝒮(i,h)); (* ℓ variabile locale *)
5        ℓ' ← eLabel(𝒮(i,h), i); (* ℓ' variabile locale *)
6        eSource(ℓ) ← vLabel(i);
7        eTwin(ℓ) ← ℓ'
8     end;
9     end;
10 Return(eSource, eTwin);
11 end;
```

buildHalfEdgeTableII: seconda fase della costruzione della tabella dei semispigoli
Input: liste di adiacenza $\mathscr{S}$, eTwin, ccwPred, ccwSsucc;
Output: ePred,eSucc.

```
1  buildHalfEdgeTableII(𝒱, 𝒮);
2  For i ← 1 to n do begin
3     For h ← 1 to m_i do begin
4        ℓ ← eLabel(i, 𝒮(i,h)); (* ℓ locale *)
5        ePred(ℓ) ← eTwin(ccwSucc(ℓ));
6        eSucc(ℓ) ← ccwPred(eTwin(ℓ));
7     end;
8  end;
9  Return(ePred, eSucc);
10 end:
```

4.3.3 Costruzione: fase II

Restano da riempire il campo eFace della tabella dei semispigoli e la tabella delle facce. La tabella delle facce del grafo a p. 35 è

f	fOuter	fInner
f_0		$\{e_{v_{18},v_{17}} \to \{f_1, f_2\}, e_{v_5,v_4} \to \{f_4, f_5, f_6\}\}$
f_1	$e_{v_{17},v_{18}}$	{}
f_2	$e_{v_{10},v_{11}}$	$\{e_{v_{15},v_{14}} \to \{f_3\}\}$
f_3	$e_{v_{14},v_{15}}$	{}

f	fOuter	fInner
f_4	e_{v_4,v_5}	{}
f_5	e_{v_8,v_6}	{}
f_6	e_{v_8,v_2}	{}

con la faccia illimitata f_0 che contiene due buchi (innescati da $e_{v_{18},v_{17}}$ e e_{v_5,v_4}). Le facce limitate sono sei e l'unica con buchi è f_2 (buco innescato da $e_{v_{15},v_{14}}$).

FinalStep: completamento semispigoli e tabella delle facce.
Input: DCEL corrente, $\mathscr{V}$, $\mathscr{S}$;
Output: eFace, fInner, fOuter.

```
1  FinalStep(𝒱, 𝒮);
2  v* ← max. lessic. di 𝒱; φ ← NewFaceLabel(); (* v*, φ variabili locali *)
3  fOuter(φ) ← Nil, fInner(φ) ← {Nil}; (* inizializzazione faccia illimitata *)
4  NewInnerBoundary(v*, φ);
5    While ∅ ≠ (𝒱* := {v ∈ 𝒱 : ∃ e t.c. eSource(e) = v ∧ eFace(e) = Nil}
6    begin
7      v* ← max. lessic. di 𝒱*;
8      φ ← faccia della DCEL corrente che contiene v*;
9      NewInnerBoundary(v*, φ)
10   end;
11 Return(eFace, fInner, fOuter);
12 end;
```

Il massimo lessicografico v^* di $\mathscr{V}$ è senz'altro un vertice della faccia illimitata; la faccia illimitata riceve l'etichetta φ ed è inizializzata al Punto 3. v^* appartiene alla frontiera di un buco della faccia illimitata; l'attraversamento (orario) di tale frontiera e di quelle esterne contenute nel buco e nella stessa componente connessa di v^* è realizzato da NewInnerBoundary(v^*, φ). Alla fine del Punto 4 tutti i semispigoli appartenenti alla componente connessa di v^* hanno il campo eFace ben definito. Al punto 5 viene considerata la possibilità di vertici non raggiunti da questa esplorazione perché appartenenti ad altre componenti connesse. Questi vertici sono origine di semispigoli per cui eFace non è ancora definito (=Nil). Individuata la faccia φ che contiene il massimo lessicografico v^* di questi vertici si può dire che v^* è sulla frontiera di un buco di φ e chiamare ancora NewInnerBoundary(v^*, φ),... e cosí via fino a che ogni semispigolo ha il campo eFace corretto. Questo procedimento, applicato al PSLG di p. 35, determina le seguenti chiamate: NewInnerBoundary(v_{18}, f_0), NewInnerBoundary(v_{15}, f_2), NewInnerBoundary(v_5, f_0).

▶ **NewInnerBoundary.** Il semispigolo al Punto 2 è dato da vOutEdge(v) (semispigolo uscente da v con angolo polare massimo). vOutEdge(v) orienta in verso orario la frontiera del buco di φ in cui sta v^*; fInner(φ) è aggiornato di conseguenza (Punto 3).

NewInnerBoundary: gestione di una nuova frontiera interna.
Input: DCEL corrente, una faccia φ, un vertice $v \in \varphi$;
Output: nessuno (viene aggiornata la DCEL).

```
1  NewInnerBoundary(v, φ);
2  e ← semisp. uscente da v con angolo polare massimo; (* e locale *)
3  fInner(φ) ← Append(fInner(φ), {e, {}});
4  segui e fino a ritornare in e ponendo eFace(·) ← φ;
5  While
6     ∃ e' t.c. eFace(e') = Nil ∧ eTwin(e') incontrato al punto 4 begin
7     NewOuterBoundary(φ, e')
8  end;
9  end;
```

Partendo da e e seguendo i semispigoli dati da eSucc la faccia sinistra è sempre la stessa (φ, cfr. Punto 4). Se un gemello e' di un semispigolo appena attraversato ha eFace non definito allora e' sta sulla frontiera esterna di una nuova faccia, che viene esplorata da NewOuterBoundary(φ, e').

► **NewOuterBoundary.** NewOuterBoundary(φ, e) scatta quando in corrispondenza del semispigolo e inizia una nuova frontiera esterna contenuta in un buco φ e appartenente alla stessa componente connessa. Per prima cosa viene generata una nuova etichetta φ' per la nuova faccia ed inizializzati i valori di fOuter e fInner.

Poi φ' è aggiunta alla lista delle facce dell'ultimo buco φ e la frontiera esterna di φ' viene attraversata, in verso antiorario, ponendo eFace$(e) \leftarrow \varphi'$ per ogni semispigolo incontrato.

Fatto ciò, non è da escludere la possibilità che semispigoli gemelli di quelli appena attraversati abbiano il campo eFace ancora non definito. Se questo accade per un gemello e' allora e' appartiene alla frontiera esterna di una nuova faccia e non resta che chiamare NewOuterBoundary(φ, e').

NewOuterBoundary: trattamento di una nuova frontiera esterna.
Input: DCEL corrente, una faccia φ e un semispigolo e;
Output: nessuno (viene aggiornata la DCEL).

```
1 NewOuterBoundary(φ, e);
2 φ' ← NewFaceLabel(); (* φ' variabile locale *)
3 fOuter(φ') ← e; fInner(φ') ← {};
4 aggiungi φ' alla lista di facce contenute nell'ultimo buco di φ;
5 segui e fino a ritornare in e ponendo eFace(·) ← φ';
6 While ∃e' = eTwin(e) t.c. eFace(e') = Nil ∧ e incontrato al punto 5
7    NewOuterBoundary(φ, e')
8 end;
9 end;
```

Osservando attentamente la tabella delle facce a p. 36 si ha un'idea dei vari passi della costruzione. La prima frontiera interna della faccia illimitata f_0 è scoperta partendo da $e_{v_{18},v_{17}}$. L'attraversamento assegna f_0 al campo eFace di $e_{v_{18},v_{17}}, e_{v_{17},v_{10}}, e_{v_{10},v_9}, e_{v_9,v_{12}}, e_{v_{12},v_{11}}, e_{v_{11},v_{18}}$.

Poi si scopre che il gemello di $e_{v_{18},v_{17}}$ ha il campo eFace uguale a Nil: la chiamata di NewOuterBoundary$(f_0, e_{v_{17},v_{18}})$ esplora la nuova faccia f_1 ponendo eFace$(\cdot) \leftarrow f_1$ per $e_{v_{17},v_{18}}, e_{v_{18},v_{11}}, e_{v_{11},v_{10}}, e_{v_{10},v_{17}}$.

A questo punto si vede che $e_{v_{10},v_{11}}$, gemello di un semispigolo sulla frontiera esterna di f_1 ha eFace non definito: NewOuterBoundary$(f_0, e_{v_{10},v_{11}})$ ed esplorazione della frontiera esterna della nuova faccia f_2. Alla fine tutti i semispigoli della componente connessa del massimo lessicografico v_{18} sono stati esplorati.

Viene poi considerato il massimo lessicografico dei vertici non ancora toccati dalle precedenti esplorazioni; il massimo lessicografico è v_{15} (contenuto in f_2), eOutEdge$(v_{15}) \leftarrow e_{v_{15},v_{14}}$ semispigolo di innesco di un buco di f_2 esplorato da newInnerBoundary(v_{15}, f_2). Durante questa esplorazione si scopre la faccia f_3 (faccia di $e_{v_{14},v_{15}}$ con la stessa frontiera del buco.

All'uscita da newInnerBoundary(v_{15}, f_2) la seconda delle tre componenti connesse è stata esplorata.

I vertici non ancora incontrati hanno il massimo lessicografico in v_5, vertice appartenente a f_0: newInnerBoundary(v_{15}, f_0) assegna f_0 ai campi eFace di $e_{v_5,v_4}, e_{v_4,v_3}, e_{v_3,v_2}, e_{v_2,v_1}, e_{v_1,v_7}, e_{v_7,v_6}, e_{v_6,v_5}$.

Alla fine si vede che eFace$(e_{4,5})$ = Nil e si scopre la faccia f_4. Finito l'attraversamento di f_4 si scopre che il gemello di e_{v_6,v_8} ha il campo eFace non settato: e_{v_8,v_6} diventa il semispigolo di innesco della faccia limitata f_5.

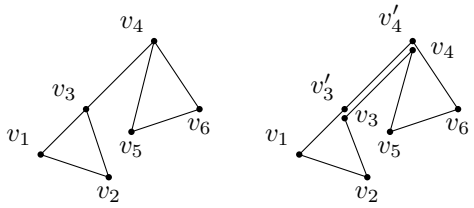
Dopo l'attraversamento il gemello di e_{v_2,v_8} ha eFace non settato. Viene cosí scoperta l'ultima faccia f_6 e dopo il suo attraversamento la DCEL è completa e la sua tabella dei semispigoli è

e	eFrom	eTwin	ePrev	eNext	eFace	e	eFrom	eTwin	ePrev	eNext	eFace
e_{v_1,v_2}	v_1	e_{v_2,v_1}	e_{v_7,v_1}	e_{v_2,v_8}	f_5	e_{v_{10},v_9}	v_{10}	$e_{v_9,v_{10}}$	$e_{v_{17},v_{10}}$	$e_{v_9,v_{12}}$	f_0
e_{v_2,v_1}	v_2	e_{v_1,v_2}	e_{v_3,v_2}	e_{v_1,v_7}	f_0	$e_{v_9,v_{12}}$	v_9	e_{v_{12},v_9}	e_{v_{10},v_9}	$e_{v_{12},v_{11}}$	f_0
e_{v_1,v_7}	v_1	e_{v_7,v_1}	e_{v_2,v_1}	e_{v_7,v_6}	f_0	e_{v_{12},v_9}	v_{12}	$e_{v_9,v_{12}}$	$e_{v_{11},v_{12}}$	$e_{v_9,v_{10}}$	f_2
e_{v_7,v_1}	v_7	e_{v_1,v_7}	e_{v_6,v_7}	e_{v_1,v_2}	f_5	$e_{v_{10},v_{17}}$	v_{10}	$e_{v_{17},v_{10}}$	$e_{v_{11},v_{10}}$	$e_{v_{17},v_{18}}$	f_1
e_{v_2,v_3}	v_2	e_{v_3,v_2}	e_{v_8,v_2}	e_{v_3,v_4}	f_6	$e_{v_{17},v_{10}}$	v_{17}	$e_{v_{10},v_{17}}$	$e_{v_{18},v_{17}}$	e_{v_{10},v_9}	f_0
e_{v_3,v_2}	v_3	e_{v_2,v_3}	e_{v_4,v_3}	e_{v_2,v_1}	f_0	$e_{v_{10},v_{11}}$	v_{10}	$e_{v_{11},v_{10}}$	$e_{v_9,v_{10}}$	$e_{v_{11},v_{12}}$	f_2
e_{v_2,v_8}	v_2	e_{v_8,v_2}	e_{v_1,v_2}	e_{v_8,v_6}	f_5	$e_{v_{11},v_{10}}$	v_{11}	$e_{v_{10},v_{11}}$	$e_{v_{18},v_{11}}$	$e_{v_{10},v_{17}}$	f_1
e_{v_8,v_2}	v_8	e_{v_2,v_8}	e_{v_4,v_8}	e_{v_2,v_3}	f_6	$e_{v_{11},v_{18}}$	v_{11}	$e_{v_{18},v_{11}}$	$e_{v_{12},v_{11}}$	$e_{v_{18},v_{17}}$	f_0
e_{v_3,v_4}	v_3	e_{v_4,v_3}	e_{v_2,v_3}	e_{v_4,v_8}	f_6	$e_{v_{18},v_{11}}$	v_{18}	$e_{v_{11},v_{18}}$	$e_{v_{17},v_{18}}$	$e_{v_{11},v_{10}}$	f_1
e_{v_4,v_3}	v_4	e_{v_3,v_4}	e_{v_5,v_4}	e_{v_3,v_2}	f_0	$e_{v_{11},v_{12}}$	v_{11}	$e_{v_{12},v_{11}}$	$e_{v_{10},v_{11}}$	e_{v_{12},v_9}	f_2
e_{v_4,v_5}	v_4	e_{v_5,v_4}	e_{v_8,v_4}	e_{v_5,v_6}	f_4	$e_{v_{12},v_{11}}$	v_{12}	$e_{v_{11},v_{12}}$	$e_{v_9,v_{12}}$	$e_{v_{11},v_{18}}$	f_0
e_{v_5,v_4}	v_5	e_{v_4,v_5}	e_{v_6,v_5}	e_{v_4,v_3}	f_0	$e_{v_{13},v_{14}}$	v_{13}	$e_{v_{14},v_{13}}$	$e_{v_{16},v_{13}}$	$e_{v_{14},v_{15}}$	f_3
e_{v_4,v_8}	v_4	e_{v_8,v_4}	e_{v_3,v_4}	e_{v_8,v_2}	f_6	$e_{v_{14},v_{13}}$	v_{14}	$e_{v_{13},v_{14}}$	$e_{v_{15},v_{14}}$	$e_{v_{13},v_{16}}$	f_2
e_{v_8,v_4}	v_8	e_{v_4,v_8}	e_{v_6,v_8}	e_{v_4,v_5}	f_4	$e_{v_{13},v_{16}}$	v_{13}	$e_{v_{16},v_{13}}$	$e_{v_{14},v_{13}}$	$e_{v_{16},v_{15}}$	f_2
e_{v_5,v_6}	v_5	e_{v_6,v_5}	e_{v_4,v_5}	e_{v_6,v_8}	f_4	$e_{v_{16},v_{13}}$	v_{16}	$e_{v_{13},v_{16}}$	$e_{v_{15},v_{16}}$	$e_{v_{13},v_{14}}$	f_3
e_{v_6,v_5}	v_6	e_{v_5,v_6}	e_{v_7,v_6}	e_{v_5,v_4}	f_0	$e_{v_{14},v_{15}}$	v_{14}	$e_{v_{15},v_{14}}$	$e_{v_{13},v_{14}}$	$e_{v_{15},v_{16}}$	f_3
e_{v_6,v_7}	v_6	e_{v_7,v_6}	e_{v_8,v_6}	e_{v_7,v_1}	f_5	$e_{v_{15},v_{14}}$	v_{15}	$e_{v_{14},v_{15}}$	$e_{v_{16},v_{15}}$	$e_{v_{14},v_{13}}$	f_2
e_{v_7,v_6}	v_7	e_{v_6,v_7}	e_{v_1,v_7}	e_{v_6,v_5}	f_0	$e_{v_{15},v_{16}}$	v_{15}	$e_{v_{16},v_{15}}$	$e_{v_{14},v_{15}}$	$e_{v_{16},v_{13}}$	f_3
e_{v_6,v_8}	v_6	e_{v_8,v_6}	e_{v_5,v_6}	e_{v_8,v_4}	f_4	$e_{v_{16},v_{15}}$	v_{16}	$e_{v_{15},v_{16}}$	$e_{v_{13},v_{16}}$	$e_{v_{15},v_{14}}$	f_2
e_{v_8,v_6}	v_8	e_{v_6,v_8}	e_{v_2,v_8}	e_{v_6,v_7}	f_5	$e_{v_{17},v_{18}}$	v_{17}	$e_{v_{18},v_{17}}$	$e_{v_{10},v_{17}}$	$e_{v_{18},v_{11}}$	f_1
$e_{v_9,v_{10}}$	v_9	e_{v_{10},v_9}	e_{v_{12},v_9}	$e_{v_{10},v_{11}}$	f_2	$e_{v_{18},v_{17}}$	v_{18}	$e_{v_{17},v_{18}}$	$e_{v_{11},v_{18}}$	$e_{v_{17},v_{10}}$	f_0

4.4 Caso generale

La DCEL è stata costruita ipotizzando che ogni vertice abbia grado ≥ 2 e che la frontiera di ogni faccia sia costituita da uno o piú poligoni semplici. Queste ipotesi sono spesso ragionevoli, ma non sempre accettabili. Cominciamo a discutere l'ultima ipotesi ragionando su un esempio: le frontiere esterne delle due facce limitate sono poligoni semplici, ma la frontiera interna della faccia

illimitata non lo è perché descritta da una lista circolare (oraria) di vertici $(v_1, v_3, v_4, v_6, v_5, v_4, v_3, v_2)$ in cui il secondo ed il sesto lato coincidono. Il grafo di sinistra può essere perturbato in quello di destra, che non dà problemi:



Siamo in presenza di un poligono senza intrecci di lati ma con ripetizioni di vertici. Poligoni di questo tipo saranno in seguito detti *poligoni quasi semplici*, perché separando leggermente i vertici ripetuti si ottiene un poligono semplice.

La determinazione della faccia cui appartiene un punto di interrogazione è cruciale nella costruzione della DCEL con piú di una componente connessa.

Per risolvere il problema del point location rispetto a poligoni quasi semplici vediamo dove può fallire l'algoritmo per il point location considerando un punto di interrogazione q come in figura. Non ci sono problemi per il poligono semplice perturbato: l'algoritmo di point location:

(1) prende in ingresso la lista circolare di vertici $(v_1, v'_3, \ldots, v_2)$;

(2) considera i lati non orizzontali: $\mathscr{C} = \{(v_1, v'_3), (v'_3, v'_4), \ldots, (v_2, v_1)\}$;

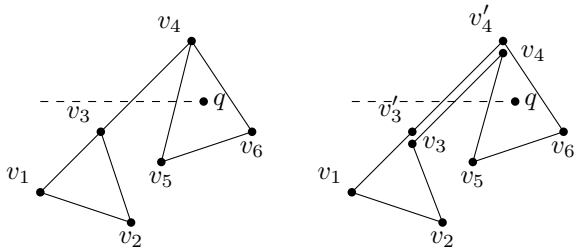
(3) considera il sottoinsieme $\mathscr{C}'$ di $\mathscr{C}$ formato dai lati con il vertice piú alto (a) di ordinata $a_y \geq q_y$ e vertice piú basso (b) di ordinata $b_y < q_y$:

$$\mathscr{C}' = \{(v'_3, v'_4), (v'_4, v_6), (v_5, v_4), (v_4, v_3)\};$$

(4) infine viene considerato il sottoinsieme $\mathscr{C}''$ di $\mathscr{C}'$ formato dai segmenti intersecati dalla semiretta orizzontale da $-\infty$ a q:

$$\mathscr{C}'' = \{(v'_3, v'_4), (v_5, v_4), (v_4, v_3)\}.$$

Se la cardinalità di $\mathscr{C}''$ è pari allora q è esterno al poligono ed è interno se dispari (come nel caso in esame).



Se i vertici v'_3 e v'_4 tendono rispettivamente v_3 e v_4 si vede che il lato v_3, v_4 contribuisce con due intersezioni: una quando viene percorso da v_3 a v_4 e una quando viene percorso da v_4 a v_3.

Si può concludere che l'algoritmo lineare per il point location rispetto a un poligono semplice è valido anche per un poligono quasi semplice.

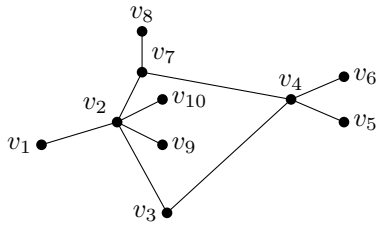
La DCEL può essere costruita nel modo visto nella Sez. 4.3 anche per grafi planari in cui le facce sono delimitate da poligoni quasi semplici. La tabella delle facce per il poligono considerato è riportata a fianco.

f	fOuter	fInner
f_0		$\{e_{v_6,v_5} \to \{f_1, f_2\}\}$
f_1	e_{v_5,v_6}	{}
f_2	e_{v_2,v_3}	{}

Piú o meno inconsciamente è risolto anche il problema legato a vertici di grado 1. Il segmento in figura può essere pensato come derivato da un triangolo con due vertici che si avvicinano fino a coincidere.

f	fOuter	fInner
f_0		$\{e_{v_2,v_1} \to \{\}\}$

Anche il prossimo esempio — un po' piú complesso — può essere visto come caso limite di un poligono semplice in cui alcuni vertici diventano coincidenti.



f	fOuter	fInner
f_0		$\{e_{v_6,v_4} \to \{f_1\}\}$
f_1	e_{v_3,v_4}	{}

In generale un poligono quasi semplice con n vertici può essere visto come derivato da un poligono semplice con piú di n vertici.

► **Vertici isolati.** L'unica restrizione ancora da affrontare è quella relativa a vertici di grado 0. Questi vertici sono isolati: vOutEdge non è definibile e non c'è modo di accedere a vertici del genere sfruttando le informazioni presenti nella tabella dei semispigoli o delle facce.

In presenza di un grafo con vertici isolati basta costruire la DCEL del sottografo indotto dai vertici non isolati e poi localizzare i vertici isolati rispetto al complesso delle facce della DCEL, come si è visto per i poligoni quasi semplici.

Bibliografia

[1] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete Computational Geometry*, 6:485–524, 1991.

[2] M. de Berg, M. van Kreveld, M. Overmars, e O. Scwarzkopf. *Computational Geometry - Algorithms and Applications.* Springer-Verlag, 1997.

[3] D.E. Muller e F.P. Preparata. Finding the intersection of two convex polyhedra. *Theoretical Computer Science*, 7(2):217–236, October 1978.

[4] F.P. Preparata e M.I. Shamos. *Computational Geometry - An introduction.* Springer-Verlag, 1985.