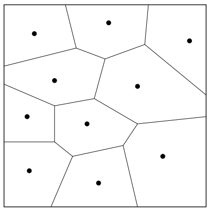
MAC 5747 GEOMETRIA COMPUTACIONAL

DIAGRAMAS DE VORONOI E DELAUNAY

1. Introdução



Dado um conjunto S de n pontos no plano queremos determinar para cada ponto p em S qual é a região $V(p)$ dos pontos do plano que estão mais perto de p do que de qualquer outro ponto em S. As n regiões $V(p)$ formam uma partição do plano chamada de *diagrama de Voronoi*.

Imaginemos uma vasta floresta contendo vários pontos de observação de incêndio. O conjunto das árvores que estão mais próximas de um determinada posto p determina a região $V(p)$ das árvores que são de responsabilidade do posto p.

Os pontos no conjunto S serão chamados de *sítios*. Dado um ponto p, qual a maneira mais eficiente de determinarmos os sítios de S mais próximos de p. Este problema foi chamado por Knuth [12] de *The Post-office Problem*, a idéia é que os sítios representam postos do correio e o ponto p é uma dada casa onde algo deve ser entregue.

A primeira solução que nos vem a cabeça para resolver o problema acima requer o cálculo $O(n)$ distâncias. Será que é possível fazermos melhor do que isto? A resposta é não; pelo menos se só estivermos interessados em resolver o problema para 'poucos' pontos. Entretanto, se o número de pontos p para os quais desejamos conhecer um sítio mais próximo de p for 'grande' então após um pré-processamento envolvendo o conjunto de sítios (que suporemos serem fixos) podemos responder a cada consulta em tempo $O(\log n)$. No caso unidimensional não é difícil ver como isto pode ser feito (ordenando os sítios e aplicando busca binária). Já no caso bidimensional, que é o caso que estaremos mais interessados, a maneira de responder as consultas em $O(\log n)$ é mais envolvente e creio que não chegaremos a estudá-la nesta disciplina (veja, por exemplo, Seção 2 do Capítulo 2 de Preparata e Shamos [16]).

O diagrama de Voronoi de um conjunto de n pontos pode ser contruído em $O(n \log n)$ por um (complexo) algoritmo do tipo divisão-e-conquista (cf. Shamos e Hoey [18]). Em 1985, Fortune [7] desenvolveu um algoritmo por varredura (*plane-sweep algorithm*) muito elegante e simples de complexidade de tempo é $O(n \log n)$.

A seguir, na Seção 2, definimos o diagrama de Voronoi de um conjunto. Na Seção 3 listaremos algumas propriedades importantes dos diagramas de Voronoi. Associado a cada diagrama de Voronoi temos um 'diagrama dual', o chamado diagrama de Delaunay $\mathrm{Del}(S)$ se S, que será apresentado na Seção 4. A cota inferior do pro é discutida na Seção 5. Um nova primitiva geométrica, a primitiva `InCircle`, será introduzida na Seção 6. Na

Date: 2 de julho de 2000.

Seção 7 estudaremos um algoritmo quadrático para construir o diagrama de Delaunay. Na Seção 8 veremos um algoritmo de divisão-e-conquista que constrói o diagrama de Delaunay de um conjunto S de n pontos em tempo $O(n \log n)$. Os diagramas de Voronoi e Delaunay podem ser usados para resolver diversos problemas. Na Seção 9 apresentaremos alguns destas aplicações. Finalmente, na Seção 10, temos uma lista de exercícios.

Observação. Os tópicos que cobriremos foram extraídos do: Guibas e Stolfi [10, 11], Capítulo 7 de De Berg, Van Kreveld, Overmars e Schwarzkoft [2] Capítulo 4 de Figueiredo e Carvalho [6]; Capítulo 8 de Laszlo [13], Capítulo 5 de O'Rourke [15]; Capítulos 5 de Preparata e Shamos [16]; e Capítulo 6 de Resende e Stolfi [3].

2. Definições

Seja $S = \{p_1, p_2, \ldots, p_n\}$ um conjunto de pontos em E^2 (ou seja, espaço Euclidiano de dimensão dois onde a disctância considerada é a Eucliadana). Os pontos em S serão chamados de *sítios*. A *região de Voronoi* $V(p_i)$ do sítio p_i consiste de todos pontos do plano que estão tão próximos de p_i quanto de qualquer outro sítio em S, ou seja:

$$V(p_i) := \{x \in E^2 \mid \text{dist}(p_i, x) \leq \text{dist}(p_j, x) \text{ para } j = 1, \ldots, n\},$$

onde $\text{dist}(p,q)$ denota a distância (Euclidiana) entre dois pontos p e q de E^2. $V(p_i)$ também é chamado de *polígono de Voronoi associado a* p_i (cf. Preparata e Shamos [16]). Abaixo chamaremos $V(p_i)$ de um polígono de Voronoi mesmo que $V(p_i)$ for uma região não-limitada (nesse caso algumas arestas de $V(p_i)$ podem ser semi-retas). A Figura 1 mostra um exemplo de uma região de Voronoi.

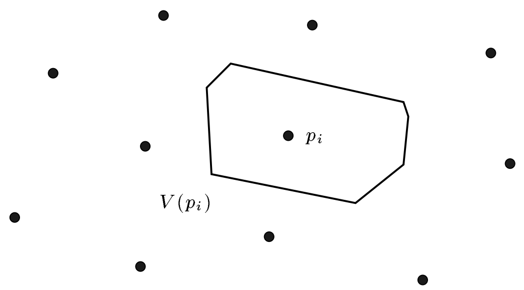


Figura 1. Exemplo de uma região (polígono) de Voronoi.

Observemos que alguns pontos do plano têm mais do que um sítio mais-próximo (pertencem a mais de uma região de Voronoi). Os pontos do plano que têm mais do que um sítio mais-próximo formam o chamado *diagrama de Voronoi* do conjunto S, denotado por $\text{Vor}(S)$. A Figura 2 mostra exemplos de diagramas de Voronoi.

Para um par de sítios distintos p_i e p_j denotaremos por $B(p_i, p_j)$ a reta bissetora ortogonal ao segmento $p_i p_j$ (a reta mediatriz do segmento $p_i p_j$). Os pontos em $B(p_i, p_j)$ são equidistantes de p_i e p_j. A Figura 3 mostra um diagrama de Voronoi para três pontos não colineares p_1, p_2 e p_3. Notemos que as retas mediatrizes se intersectam em um único ponto — o circuncentro

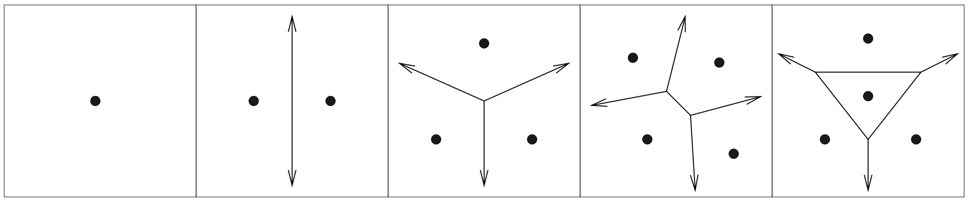


FIGURA 2. Alguns diagramas de Voronoi com até 4 sítios.

da circunferência circuscrita ao triângulo $\triangle(p_1, p_2, p_3)$; isto é, o centro da única circunferência que passa sobre os sítios p_1, p_2 e p_3.

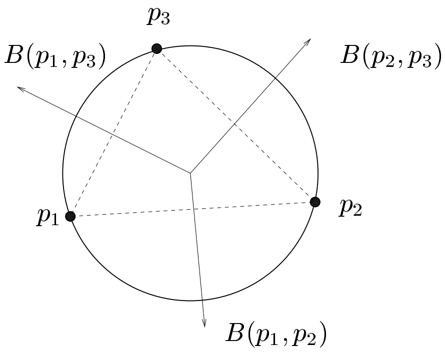


FIGURA 3. As retas mediatrizes $B(p_1, p_2)$, $B(p_1, p_3)$ e $B(p_2, p_3)$ se intersectam no centro da circunferência.

Seja $H(p_i, p_j)$, $i \neq j$, o semiplano fechado com fronteira $B(p_i, p_j)$ e contendo o sítio p_i. Logo, $H(p_i, p_j)$ é o conjunto dos pontos do plano que estão tão próximos de p_i quanto de p_j, ou seja

$$H(p_i, p_j) := \{x \in E^2 \mid \mathrm{dist}(p_i, x) \leq \mathrm{dist}(p_j, x)\}.$$

Como $V(p_i)$ é o conjunto dos pontos do plano que estão tão próximos de p_i quanto de qualquer outro sítio em S temos então que

$$V(p_i) = \cap_{j \neq i} H(p_i, p_j).$$

Desta igualdade concluímos uma propriedade importante dos diagramas de Voronoi, a saber, cada região de Voronoi é convexa (já que a intersecção de conjuntos convexo é um conjunto convexo). Quando a região é limitada ela forma um polígono convexo. As arestas das regiões são chamadas de *arestas de Voronoi* e os vértices de *vértices de Voronoi*. Observemos que um ponto no interior de uma aresta de Voronoi tem dois sítios mais-próximos e um vértice de Voronoi tem pelo menos três sítios mais-próximos.

Ao longo de toda a discussão que faremos suporemos que nenhum subconjunto de quatro pontos de S são cocirculares (pertencem a uma mesma circunferência) e nenhum conjunto de três pontos são colineares (não sei se precisamos desta última restrição). Estas restrições, a famosa hipótese de posição geral, é usada apenas para facilitar a discussão e não constituiem restrições 'sérias.'

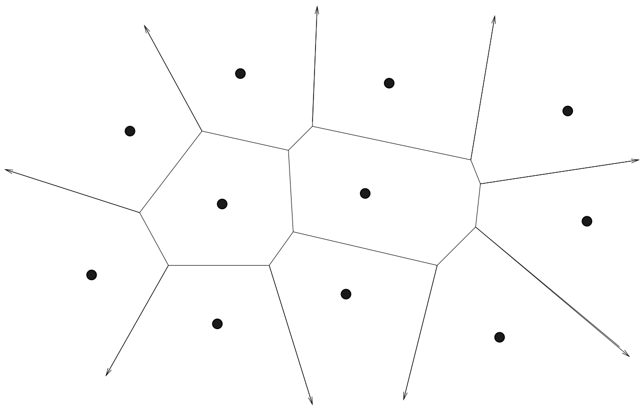


Figura 4. O diagrama de Voronoi de um conjunto S de sítios no plano.

3. Propriedades do diagrama de Voronoi

Neste seção listaremos algumas propriedades importantes dos diagramas de Voronoi. Suporemos que $S = \{p_1, \ldots, p_n\}$ é o conjunto dado de sítios.

Teorema 1. *Cada região de Voronoi $V(p_i)$ é convexa.* ∎

Teorema 2. *Se v é um vértice de Voronoi (ou seja, vértice de um diagrama de Voronoi) então v está na intersecção de três arestas de Voronoi, isto é, v está na junção de três regiões de Voronoi.*

Prova. Um vértice de Voronoi é uma intersecção de arestas de Voronoi. Sejam $e_0, e_1, \ldots, e_{k-1}$ $(k \geq 3)$ as arestas, em sentido anti-horário, incidentes a um vértice v (veja Figura 5). A aresta e_i é compartilhada pelos polígonos $V(p_{i-1})$ e $V(p_i)$ $(i = 0, 1, \ldots, k-1)$. Como v pertence a e_i então v é eqüidistante a p_{i-1} e p_i $(i = 0, \ldots, k-1)$. Logo v é eqüidistante a $p_1, p_2, \ldots, p_{k-1}$ e portanto estes pontos são cocirculares. Como assumimos que em S não existem quatro pontos cocirculares temos que $k \leq 3$. Suponha $k = 2$. Então e_0 e e_1 são arestas compartilhadas por $V(p_0)$ e $V(p_1)$ e ambas as arestas pertencem a reta mediatriz do segmento $p_1 p_2$ e portanto e_0 e e_1 não se intersectam no vértice v, uma contradição. Logo $k = 3$ e v está na intersecção de três arestas. ∎

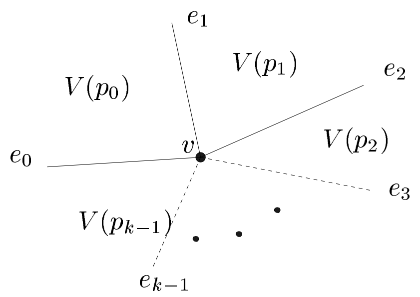


Figura 5. Arestas de Voronoi incidentes a um vértice de Voronoi.

O teorema acima é equivalente ao fato de que se v é um vértice de Voronoi na junção das regiões de Voronoi $V(p_1)$, $V(p_2)$ e $V(p_3)$ então v é o centro de um círculo $C(v)$ determinado por p_1, p_2 e p_3.

Teorema 3 (Círculo Circunscrito). *Para cada vértice de Voronoi v o interior do círculo $C(v)$ não contém sítio algum. (Observemos que a fronteira de $C(v)$ contém exatamente três sítios.)*

Prova. Sejam p_1 p_2 e p_3 os três sítios de S determinando o círculo $C(v)$. Se $C(v)$ contém outro sítio, digamos p_4, então v é mais-próximo de p_4 do que de p_1, p_2 e p_3. Uma contradição, pois assumimos que v era um ponto comum a $V(p_1), V(p_2)$ e $V(p_3)$. A situação está ilustrada na Figura 6. ∎

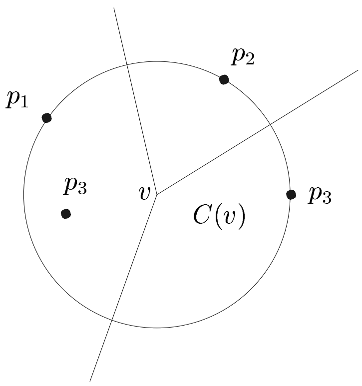


FIGURA 6. O círculo $C(v)$ não contém sítios em seu interior.

Notemos que o teorema acima estabelece uma relação bijetora entre centro de círculos contendo três sítios na fronteira e vértices de Voronoi. Isto nos fornece um algoritmo para encontrarmos todos os vértices de Voronoi de $\text{Vor}(S)$ para um dado conjunto S: para cada tripla de sítios (p_i, p_j, p_k), $1 \leq i < j < k \leq n$, se o círculo que contém p_i, p_j e p_k na sua fronteira não contém nenhum outro sítio em seu interior então o centro do círculo é um vértice de Voronoi.

Teorema 4 (Círculo Vazio). *Sejam p e q dois sítios distintos de S. As regiões $V(p)$ e $V(q)$ possuem uma aresta em comum se e somente se existe um círculo contendo p e q tal que todos os demais sítios de S são exteriores ao círculo.*

Prova. Os polígonos $V(p)$ e $V(q)$ possuem uma aresta em comum se e somente se existem pontos que são equidistantes de p e q e além disso são mais-próximos de p e q do que qualquer outro sítio de S. Isto implica que $V(p)$ e $V(q)$ têm uma aresta em comum se e somente se existe um círculo que contém p e q e exclui todos os outros demais sítios (considere um círculo $C(x)$ com centro em um ponto x no interior da aresta compartilhada por $V(p)$ e $V(q)$ e com raio $\text{dist}(x, p) = \text{dist}(x, q)$). ∎

Teorema 5. *Todo sítio mais-próximo de um sítio p_i de S determina uma aresta de Voronoi do polígono $V(p_i)$.*

Prova. Seja p_j um sítio mais-próximo de p_i e seja v o ponto do meio do segmento p_ip_j. Mostraremos que v pertence a $V(p_i)$. Suponha por contradição que v não pertence a $V(p_i)$. Neste caso temos que o segmento p_iv intersecta alguma aresta de $V(p_i)$, digamos que a aresta contida na mediatriz de p_ip_k em um ponto u. A situação é ilustrada pela Figura 7. Assim temos que $\text{dist}(p_i, u) < \text{dist}(p_i, v)$ e portanto $\text{dist}(p_i, p_k) \leq 2 \times \text{dist}(p_i, u) < 2 \times \text{dist}(p_i, v) = \text{dist}(p_i, p_j)$, uma contradição. Logo, o círculo com centro em v e raio $\text{dist}(p_i, v)$ não contém nenhum sítio além de p_i e p_j e portanto, pelo Teorema 4, os polígonos $V(p_i)$ e $V(p_j)$ possuem uma aresta em comum. ∎

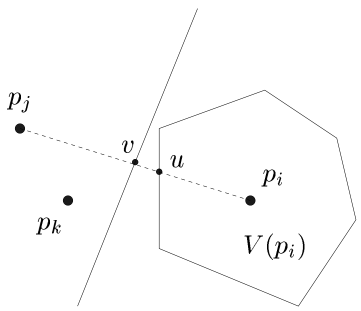


FIGURA 7. Cada sítio mais próximo de p_i determina uma aresta de $V(p_i)$.

Teorema 6. *A região de Voronoi $V(p_i)$ é limitada se e somente se p_i pertence ao interior do fecho convexo de S ($Conv(S)$).*

Prova. Suponhamos que p_i pertence ao interior de $\text{conv}(S)$. Então existem sítios p_1, p_2 e p_3 em S tais que p_i pertence ao interior do triângulo $\triangle(p_1, p_2, p_3)$. Consideremos os círculos $C_{1,2}, C_{1,3}$ e $C_{2,3}$ determinados por p_i e cada um dos três pares de sítios $\{p_1, p_2\}$, $\{p_1, p_3\}$ e $\{p_2, p_3\}$, respectivamente. (Veja Figura 8.) Cada um deste círculos tem raio finito. Seja $A_{j,k}$ o arco circular entre na fronteira de $C_{j,k}$ entre p_j e p_k que não contém p_i ($1 \leq j < k \leq 3$). Seja c um círculo que contém $C_{1,2}, C_{1,3}$ e $C_{2,3}$. Seja C um círculo contendo $C_{1,2}, C_{1,3}$ e $C_{2,3}$. Afirmamos que todo ponto x exterior a C está mais perto de um dos sítios p_1, p_2 e p_3 do que do sítio p_i. De fato, considere o segmento xp_i para algum ponto x que não pertença a C. Pelo Teorema de Jordan temos que xp_i intersecta um dos lados do triângulo $\triangle(p_1, p_2, p_3)$, digamos p_1p_2. Logo xp_i intersecta $A_{1,2}$ em um ponto u que está mais próximo de p_1 ou p_2 do que de p_i. Temos assim que todo ponto x externo a C é mais-próximo de um dos sítios p_1, p_2, p_3 do que de p_i, o que implica que a região $V(p_i)$ está contida em C e portanto é limitada.

Reciprocamente, suponha que a região $V(p_i)$ é limitada (i.e. $V(p_i)$ é um polígono) e sejam $e_1, e_2, \ldots, e_k$ ($k \geq 3$) as arestas de $V(p_i)$ listadas em sentido anti-horário. Cada aresta e_j ($j = 1, \ldots, k$) está contida em alguma mediatriz de um segmento p'_jp_i, $p'_j \in S$. Assim temos que p_i é um vértice no interior de $\text{conv}\{p'_1, p'_2, \ldots, p'_k\}$. ∎

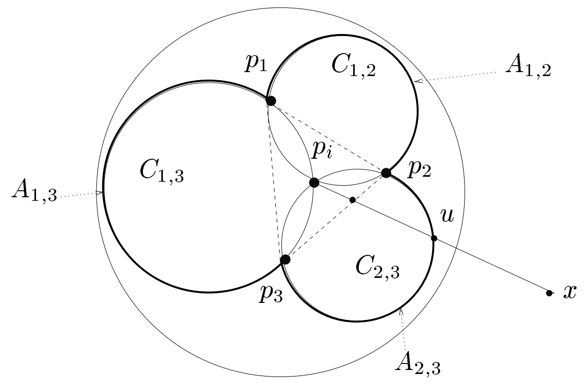


FIGURA 8. Como p_i é um ponto do interior de conv(S) temos que a região $V(p_i)$ é limitada.

4. Diagrama de Delaunay

O que veremos a seguir estabelece uma ligação fundamental entre o diagrama de Voronoi e uma triangularização especial.

Observação. O material desta seção foi extraído de Figueiredo e Carvalho [6].

4.1. **O diagrama.**

Nós já estudamos o grafo dual de uma triangularização (que era uma árvore), consideraremos agora o grafo dual do diagrama de Voronoi de um conjunto S de pontos em E^2. O grafo dual do diagrama de Voronoi Vor(S) é o grafo cujos vértices são os sítios de S e dois vértices são ligados por uma aresta se as regiões correspondentes compartilham uma aresta. Consideremos agora o diagrama obtido ao representarmos estas arestas por segmentos de retas. Chamaremos este diagrama de o[1] *diagrama de Delaunay* de S (cf. Figueiredo e Carvalho [6]) e o denotaremos por Del(S). A Figura 9 mostra o diagrama de Delaunay correspondente ao diagrama de Voronoi da Figura 4.

4.2. **Propriedades do diagrama de Delaunay.**

Devido a dualidade entre o diagrama de Voronoi e o diagrama de Delaunay, muitas das propridades do diagrama de Voronoi vistas na seção anterior têm uma tradução imediata em termos do diagrama de Delaunay. Abaixo listaremos algumas dessas propriedades como teoremas.

Teorema 7 (Círculo Circunscrito). *Três sítios p_1, p_2 e p_3 de um conjunto S definem uma face do diagrama de Delaunay de S se e somente se existe um círculo que contém p_1, p_2 e p_3 e não contém outro sítio de S.*

Prova. Segue do Teorema 3. (Veja Figura 10.) ∎

Teorema 8 (Círculo Vazio). *Dois sítios p_1 e p_2 de um conjunto S são adjacentes no diagrama de Delaunay de S se e somente se existe algum círculo que contém p_1 e p_2 e não contém outro sítio de S.*

[1] O artigo definido é conseqüência da nossa hipótese de não haver em S quatro sítios cocirculares—considere os sítios que são vértices de um quadrado.

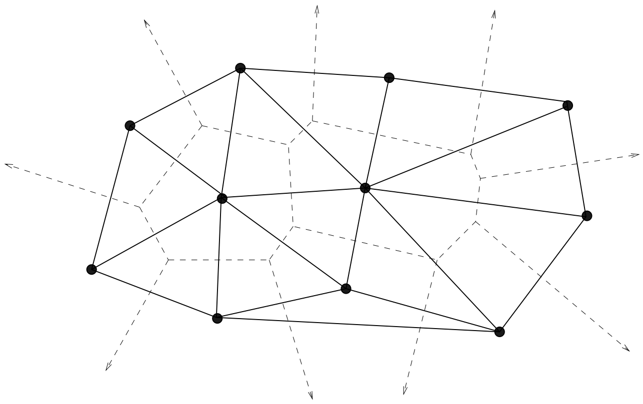


FIGURA 9. O diagrama de Delaunay do conjunto S da Figura 4.

Prova. Segue do Teorema 4. (Veja Figura 10.) ∎

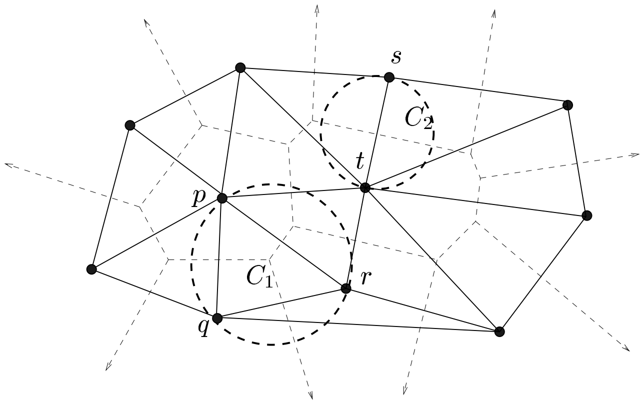


FIGURA 10. O círculo C_1 é um certificado de que $\triangle(p,q,r)$ é uma face de Delaunay. O círculo C_2 é um certificado de que a st é uma aresta de Delaunay.

Teorema 9. *Todo sítio mais-próximo de um sítio p de um conjunto S determina uma aresta do diagrama de Delaunay de S incidente a p.*

Prova. Segue do Teorema 5. ∎

Teorema 10. *O diagrama de Delaunay de um conjunto S de sítios é a realização de um grafo planar.*

Para provarmos o Teorema 10 precisaremos do seguinte lema:

Lema 1. *Sejam pq e rs segmentos que se intersectam em um único ponto o. Existe um círculo que contém p e q e não contém r e s se e somente se $\angle(qrp) + \angle(qsp) < \pi$ (equivalentemente, $\angle(rqs) + \angle(rps) > \pi$).*

Prova. Suponhamos que exista um círculo C que contém p e q e não contém r e s. Podemos assumir que p e q estão na fronteira de C. Sejam r' e s' as intersecção de rs com C (veja Figura 11). Então $\angle(qrp) + \angle(qsp) <$

$\angle(qr'p) + \angle(qs'p) = \pi$. Reciprocamente, se $\angle(qrp) + \angle(qsp) < \pi$ então existem pontos r' e s' sobre o segmento rs tal que $\angle(qr'p) + \angle(qs'p) = \pi$. Estes pontos, juntamente com p e q determinam um círculo que não contém r e s. ∎

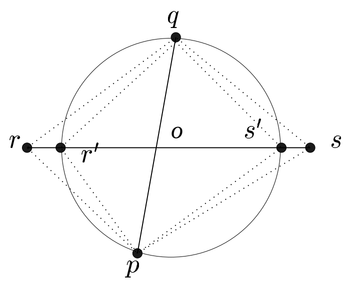


FIGURA 11. Círculo contendo p e q e excluindo r e s.

Prova do Teorema 10. Precisamos provar que as arestas de $\mathrm{Del}(S)$ não se intersectam a não ser nos vértices. Suponhamos que pq e rs sejam duas arestas de $\mathrm{Del}(S)$ que se intersectam em um ponto o. Como $V(p)$ e $V(q)$ compartilham uma aresta de Voronoi então existe um círculo contendo p e q que exclui r e s (cf. Teorema 8). Pelo lema anterior isto implica que $\angle(qrp) + \angle(qsp) < \pi$. Logo, $\angle(rqs) + \angle(rps) > \pi$ o que implica (ainda pelo lema anterior) que não existe um círculo contendo r e s que exclua p e q. Portanto, pelo Teorema 8, $V(r)$ e $V(s)$ não possuem uma aresta em comum. ∎

Contruiremos um conjunto de triângulos

$$\mathcal{T}(S) := \{T(v) \mid v \text{ é um vértice de Voronoi de } \mathrm{Vor}(S)\}$$

tal que os seus interiores sejam disjuntos e cada ponto no interior de $\mathrm{conv}(S)$ pertence a um destes triângulos.

Seja v um vértice de Voronoi e sejam $p_{v,1}, p_{v,2}, p_{v,3}$ os sítios de S tal que $V(p_{v,1}), V(p_{v,2})$ e $V(p_{v,3})$ são as três regiões de $\mathrm{Vor}(S)$ que compartilham o vértice v (cf. Teorema 2). (Veja Figura 12.) Definimos $T(v)$ como sendo o triângulo cujos vértices são $p_{v,1}, p_{v,2}, p_{v,3}$. Temos que $T(v)$ é um triângulo do diagrama de Delaunay e $T(v) \subseteq \mathrm{conv}(S)$ para todo vértice de Voronoi v, (prova deste fato usa o Teorema 8).

Teorema 11. *Sejam v e u vértices distintos de $\mathrm{Vor}(S)$. Então os interiores de $T(v)$ e $T(u)$ são disjuntos.*

Prova. Considere agora dois triângulos $T(v_1)$ e $T(v_2)$ de $\mathcal{T}(S)$, $v_1 \neq v_2$. Mostraremos que os interiors de $T(v_1)$ e $T(v_2)$ não se intersectam. Se $C(v_1)$ e $C(v_2)$ são disjuntos então $T(v_1)$ e $T(v_2)$ também são disjuntos. (Lembremos que $C(v)$ denota o círculo que tem $p_{v,1}, p_{v,2}$ e $p_{v,3}$ na fronteira.) Assuma então que $C(v_1)$ e $C(v_2)$ se intersectam. Pelo Teorema 7 temos que $C(v_1) \not\subseteq C(v_2)$ e $C(v_2) \not\subseteq C(v_1)$. Assim a fronteira de $C(v_1)$ e $C(v_2)$ se intersectam em dois pontos q_1 e q_2. Seja l a reta determinada pelo segmento q_1q_2; esta reta também separa v_1 de v_2. (Veja Figura 13.) Afirmamos que l também separa o interior de $T(v_1)$ do interior de $T(v_2)$. De fato, assuma, por contradição,

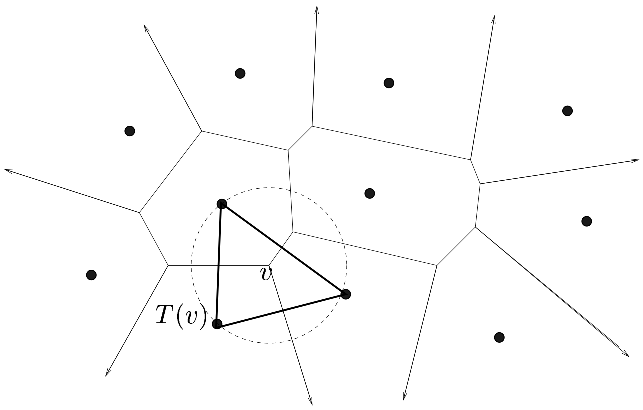


FIGURA 12. O triângulo $T(v)$ correspondente ao vértice de Voronoi v.

que existe um ponto de $T(v_1)$ no lado de l onde esta v_2. (Veja a área sombreada da Figura 13.) Como $T(v_1) \subset C(v_1)$ temos então que um vértice de $T(v_1)$ está no interior de $C(v_2)$ o que contradiz o Teorema 7. Logo, os interiores de $T(v_1)$ e $T(v_2)$ são disjuntos. ∎

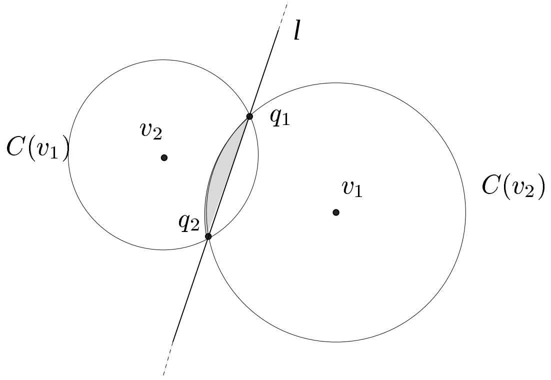


FIGURA 13. Não existe ponto de $T(v_1)$ no interior da região sombreada.

Teorema 12. *Seja pq uma aresta do diagrama de Delaunay e seja $r \in S$. Então os sítios p, q e r formam um triângulo de $\mathcal{T}(S)$ se e somente se $\angle(prq)$ é máximo dentre todos os ângulos da forma $\angle(psq)$, onde $s \in S$ está no mesmo semiplano de r em relação a pq.*

Prova. (A feramenta de gemeotrai elementar que utilizamos na prova deste teorema é o Teorema do Ângulo Inscrito.) Suponhamos que $s \in S$ está no mesmo semiplano de r em relação a pq e que $\angle(psq) > \angle(prq)$. Então , pelo Teorema do Ângulo Inscrito, s é interior ao círculo contendo p, q e r na fronteira, o que contradiz o fato de $\triangle(p, q, r)$ ser um triângulo de $\mathcal{T}(S)$. (Veja Figura 14.) Reciprocamente, se p, q e r forma um triângulo de $\mathcal{T}(S)$ então por definição de $\mathcal{T}(S)$ e pelo Teorema 7 sabemos que existe um existe um vértice de Voronoi v tal que $C(v)$ é um círculo vazio. Agora, pelo Teorema do Ângulo Inscrito (ou um corolário deste teorema) sabemos que um círculo $\angle(prq)$ é máximo dentre todos os ângulos da forma $\angle(psq)$, onde $s \in S$ está no mesmo semiplano de r em relação a pq ∎

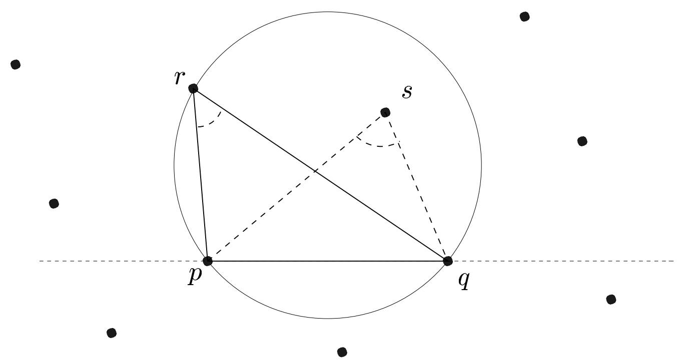


FIGURA 14. O triângulo $\triangle(p,q,r)$ pertence a $\mathcal{T}(S)$.

Teorema 13. *Os triângulos em $\mathcal{T}(S)$ cobrem o fecho convexo dos sítios em S.*

Prova. Notemos primeiro que $T(v) \subset \text{conv}(S)$, o que implica que a união dos triângulos em $\mathcal{T}(S)$ está contida em $\text{conv}(S)$.

Se p_ip_j é segmento na fronteira de $\text{conv}(S)$ então p_ip_j é uma aresta do diagrama de Delaunay (isto segue do Teorema 8).

Toda arestas p_ip_j do diagrama de Delaunay $\text{Del}(S)$ delimita uma ou duas regiões. Se p_ip_j delimita uma região do diagrama então existe um triângulo em $\mathcal{T}(S)$ incidente a p_ip_j: o triângulo formado por p_i, p_j e pelo sítio p_k que máxima o ângulo $\angle(p_ip_kp_j)$ (cf. Teorema 12). Se p_ip_j não fizer parte da fronteira de $\text{conv}(S)$ então existem dois triângulos de $\mathcal{T}(S)$ incidentes a p_ip_j, novamente determinados pelos sítios que formam ângulos máximos com p_ip_j; um em cada semiplano determinado pela reta que contém p_ip_j.

Seja agora p um ponto arbitrário de $\text{conv}(S)$. Pelo que foi exposto acima sabemos que se p pertence a alguma aresta de $\text{Del}(S)$ então p pertence a algum triângulo de $\mathcal{T}(S)$. Assim podemos supor que p não pertence a uma aresta de $\text{Del}(S)$. Seja l uma semi-reta que tenha a sua extremidade em p e não contenha nenhum outro sítio de S. Seja p_ip_j a 'primeira' arestas de $\text{Del}(S)$ que tem intersecção não vazia com l. (Veja Figura 15.) Afirmamos que p pertence ao triângulo T de $\mathcal{T}(S)$ formado pela aresta p_ip_j e pelo sítio p_k tal que $\angle(p_ip_kp_j)$ seja máximo. De fato, se p não pertence a T então a semi-reta l deveria intersectar alguma outra aresta de $T4$ e não somente p_ip_j. Este ponto de intersecção seria necessariamente anterior ao ponto de intersecção com p_ip_j, o que contradiz a escolha da aresta p_ip_j. Desta forma temos que os triângulos em $\mathcal{T}(S)$ cobrem $\text{conv}(S)$. ∎

Em 1934, B. Delaunay mostrou que o diagrama $\text{Del}(S)$ produz uma triangularização do conjunto S de sítios. Esta triangularização é chamada de *triangularização de Delaunay* de S.

Teorema 14. *(Delaunay* [4]*) O diagrama de Delaunay é uma triangularização do conjunto de sítios S.*

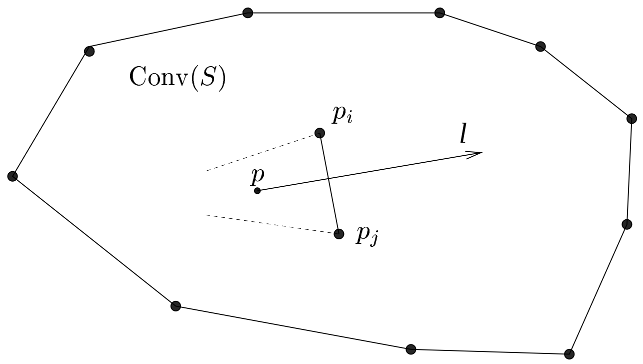


Figura 15. Ilustração da prova do Teorema 13.

Prova. Cada triângulo de $\mathcal{T}(S)$ é formado por arestas do diagrama de Delaunay de S. Pelo Teorema 11 a intersecção entre os interiores de dois triângulos distintos de $\mathcal{T}(S)$ é vazia e pelo Teorema 13 os triângulos em $\mathcal{T}(S)$ cobre conv(S). Logo as regiões (faces) limitadas do diagrama de Delaunay formam triângulos com interiores disjuntos que cobrem conv(S). ∎

Corolário 1. *O diagrama de Voronoi um conjunto S de n sítios tem no máximo $2n-5$ vértices e $3n-6$ arestas.*

Prova. Cada aresta do diagrama de Delaunay Del(S) corresponde a uma única arestas de Voronoi de Vor(S). Como Del(S) é uma triangularização de S temos, pela Fórmula de Euler, que Del(S) tem no máximo $3n-6$ arestas e $2n-4$ faces. Logo o número de arestas de Voronoi de Vor(S) é no máximo $3n-6$ e o número de vértices de Voronoi é no máximo $2n-5$ (a face ilimitada de del(S) não corresponde a um vértice de Voronoi). ∎

5. Cota inferior

Discutiremos nesta seção a cota inferior de qualquer algoritmo que constrói um diagrama de Voronoi ou uma triangularização de um dado conjunto S com n pontos.

Dado o diagrama de Voronoi Vor(S) não é difícil encontrarmos o par de pontos mais próximos em tempo $O(n)$: basta examinarmos as sítios cujas regiões compartilham uma aresta de Voronoi—pelo Corolário 1 sabemos que teremos que examinar $O(n)$ pares de sítios; o número de arestas de um diagrama de Voronoi de n sítios tem $O(n)$ arestas. Como sabemos que o Problema do Par Mais Próximo tem conta inferior $\Omega(n \log n)$ temos então o seguinte teorema.

Teorema 15. *Qualquer algoritmo para construir o diagrama de Voronoi de um dado conjunto com n sítios faz $\Omega(n \log n)$ operações elementares.* ∎

Como veremos, o diagrama de Delaunay Del(S) pode ser obtido em tempo linear a partir do diagrama de Voronoi Vor(S). O Teorema 15 já implica que o problema de construir o diagrama de Delaunay Del(S) de um conjunto com n sítios tem cota inferior $\Omega(n \log n)$. Entretanto, observaremos aqui que o Problema de Ordenação pode ser reduzido ao problema de construir uma triangularização (não necessariamente a de Delaunay) de um dado conjunto de pontos. Considere um conjunto $\{x_1, \ldots, x_n\}$ de n números. Dado este conjunto construiremos o conjunto S formado pelos $n+1$ pontos

$$\{(x_1, 1), (x_2, 1), \ldots, (x_n, 1), (0, 0)\}.$$

Esta transformação pode ser realizada em tempo linear. O conjunto S tem uma única triangularização (ilustrada na Figura 16). Dada esta triangularização (ou seja, uma estrutura de dados, como a winged-edge, representando o grafo planar assoaciado à esta triangularização) podemos em tempo linear obter a ordenação dos números $x_1, \ldots, x_n$. Resumindo, obtemos o seguinte teorema.

Teorema 16. *Qualquer algoritmo para construir a triangularização de um dado conjunto com n pontos faz $\Omega(n \log n)$ operações elementares.* ∎

O teorema acima implica que, em particular, qualquer algoritmo que construa uma triangularização de Delaunay de um dado conjunto com n sítios faz $\Omega(n \log n)$ operações elementares.

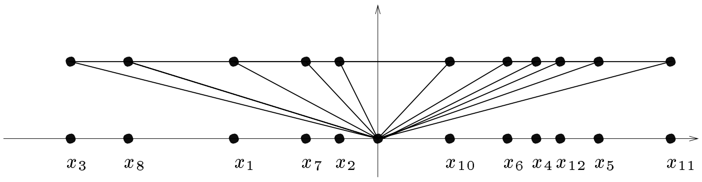


FIGURA 16. Ilustração da redução do Problema de Ordenação para o problema de construir uma triangularização.

6. A primitiva geométrica InCircle

Observação. O material desta seção foi extraído do artigo de Guibas e Stolfi [10].

Dados quatro pontos a, b, c, d o predicado (primitiva) `InCircle`(a, b, c, d) é verdadeiro se o ponto d está na face esquerda da circunferência orientada pelo triângulo $\triangle(a, b, c)$. A face esquerda é a face interior (limitada) da circunferência se o triângulo $\triangle(a, b, c)$ estiver orientado positivamente, caso contrário a face esquerda é o exterior da circunferência. Logo, se o tomamos a, b e c de tal forma que o triângulo $\triangle(a, b, c)$ esteja orientado positivamente então a primitiva `InCircle` decide se o ponto d está no círculo determinado por pelos pontos a, b e c, veja Figura 17.

Notemos que uma situação degenerada ocorre se d pertence a circunferência determinada por a, b e c. Curiosamente a situação não é degenerada se a, b e c são colineares e d esta fora da reta determinada por estes pontos.

Teorema 17. *O valor da primitiva* ***`InCircle`***(a, b, c, d) *é dado pelo sinal do determinante*

$$\mathcal{D}(a, b, c, d) := \begin{vmatrix} x_a & y_a & x_a^2 + y_a^2 & 1 \\ x_b & y_b & x_b^2 + y_b^2 & 1 \\ x_c & y_c & x_c^2 + y_c^2 & 1 \\ x_d & y_d & x_d^2 + y_d^2 & 1 \end{vmatrix}.$$

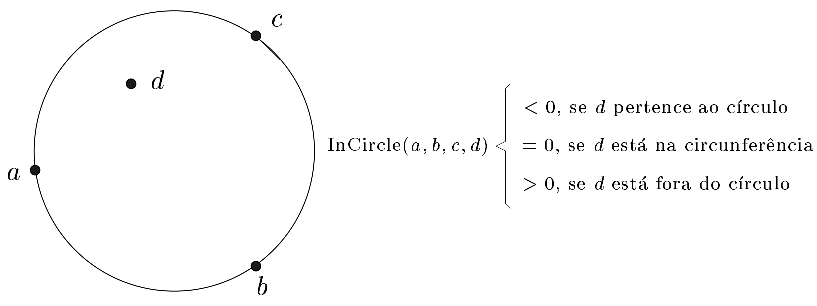


FIGURA 17. A primitiva `InCircle`.

Prova. Considere a seguinte função que leva pontos do plano xy no parabolóide $z = x^2 + y^2$ (A Figura 18 ilustra esta função.)

$$\lambda : (x, y) \mapsto (x, y, x^2 + y^2).$$

Primeiro mostraremos que a, b, c e d são cocirculares se e somente se $\lambda(a)$, $\lambda(b)$, $\lambda(c)$ e $\lambda(d)$ são coplanares.

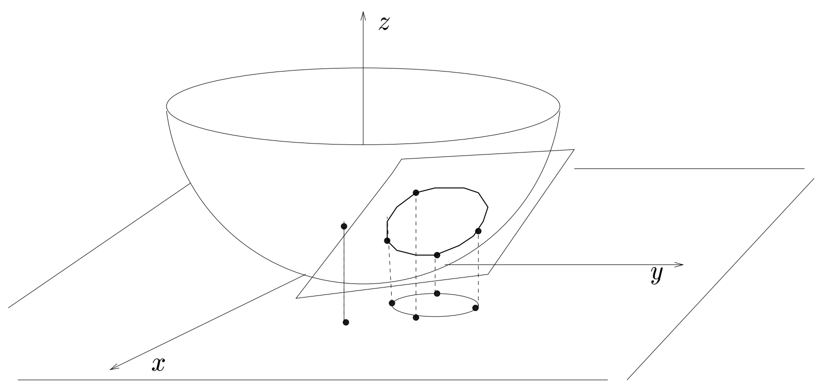


FIGURA 18. O parabolóide usado para calcula a primitiva `InCircle`.

Suponha primeiro que a, b, c e d sejam cocirculares. Se nós temos o caso degenerado onde os pontos são colineares então o determinante é zero como podemos verificar através da expansão da terceira coluna. Mas o valor do determinante também é igual ao volume sinalizado do tetraedro definido por $\lambda(a)$, $\lambda(b)$, $\lambda(c)$ e $\lambda(d)$. Como o volume é zero então os pontos devem ser coplanares. Caso contrário seja (p, q) o centro da circunferência de raio r que contém os pontos a, b, c e d. Nos temos que

$$(x_a - p)^2 + (y_a - q)^2 = r^2,$$

ou equivalentemente

$$-2p \cdot x_a - 2q \cdot y_a + 1 \cdot (x_a^2 + y_a^2) + (p^2 + q^2 - r^2) \cdot 1 = 0. \tag{1}$$

Relações análogas se verificam para os pontos b, c e d, e portanto nós temos uma dependência linear entre as colunas da matriz acima, o que implica que $\mathcal{D}(a, b, c, d) = 0$. Assim, concluímos novamente que $\lambda(a), \lambda(b), \lambda(c)$ e $\lambda(d)$ são coplanares.

Agora, reciprocamente, suponha que $\lambda(a), \lambda(b), \lambda(c)$ e $\lambda(d)$ são coplanares. Se os pontos a, b, c e d são colineares não há o que provar. Logo suponha que (digamos) a, b, c são não-colineares. Seja (p, q) o centro da circunferência de raio r determinada pelo triângulo $\triangle(a, b, c)$. Então a, b e c satisfazem a Equação 1 (as respectivas versões). Como a, b e c são não-colineares então as três linhas correspondentes da matriz acima são linearmente independentes. Como o determinante é nulo então a quarta linha da matriz (correspondente ao ponto d) deve ser combinação linear das três primeiras linhas, e portanto d também satisfaz a Equação 1, ou seja, d está na circunferência determinada por a, b e c.

O que acabamos de provar mostra que a projeção de seções planares do parabolóide $z = x^2 + y^2$ no plano xy determinam circunferências. O parabolóide é uma superfície convexa (para 'cima') então em uma seção de um plano com o parabolóide a parte do parabolóide abaixo do plano tem sua projeção no plano xy no interior do círculo determinado por a, b e c e os pontos do parabolóide tem sua projeção fora deste círculo. A partir disto vemos que para decidir se um ponto está dentro ou fora do círculo determinado pelos pontos a, b, c e d basta aplicarmos a primitiva `Teste-de-Orientação`$(\lambda(a), \lambda(b), \lambda(c), \lambda(d))$, que corresponde ao volume sinalizado do tetraedro determinado pelos pontos $\lambda(a), \lambda(b), \lambda(c)$ e $\lambda(d)$. O resultado da primitiva `Teste-de-Orientação`$(\lambda(a), \lambda(b), \lambda(c), \lambda(d))$ é dado através do sinal do determinante $\mathcal{D}(a, b, c, d)$. ∎

7. Um algoritmo quadrático

Nesta seção estudaremos um algoritmo quadrático para construir o diagrama de Delaunay de um dado conjunto $S = \{p_1, \ldots, p_n\}$ de sítios. O algoritmo que veremos faz uma 'mímica' do algoritmo `Embrulho-para-presente` para construir o fecho convexo de um conjunto de pontos em $\mathbb{R}^3$. Notemos ainda que dado Del(S) podemos, em tempo linear, construir Vor(S).

7.1. **A idéia do algoritmo.** A idéia básica, análoga a do Algoritmo `Embrulho-para-Presente`, é usar as arestas do diagrama de Delaunay (*arestas de Delaunay*) dos triângulos que já foram determinados para encontrar os outras triângulos. Cada arestas do diagrama de Delaunay é compartilhada por um ou dois triângulos (*triângulo de Delaunay*—triângulos da triangularização de Delaunay). Uma aresta de Delaunay tem apenas um triângulo do diagrama incidente a ela quando está na fronteira do fecho convexo de S. Chamaremos uma aresta de Delaunay de *livre* se algum dos triângulos que compartilham esta aresta ainda não foi determinado pelo algoritmo. Dada uma aresta livre p_ip_j de um triângulo de Delaunay podemos determinar o triângulo que compartilha esta aresta da seguinte maneira: encontre o ponto p_k, do lado apropriado do segmento p_ip_j, para o qual o $\angle(p_ip_kp_j) > \angle(p_ip_sp_j)$ para todo ponto p_s que esta do mesmo lado de p_k (cf. Teorema 12). Este é o passo básico do algoritmo e pode ser feito usando-se as primitivas `Left` e `InCircle`.

7.2. **O Algoritmo.** Como fizemos no Algoritmo `Embrulho-para-Presentes` uma estrutura de adjacência (como *winged-edge*) pode ser utilizada para armazenarmos o diagrama de Delaunay. A medida que as faces são introduzidas no diagrama suas arestas são examinadas para verificarmos se elas já fazem parte da estrutura (se são livres ou não). Em caso da aresta examinada não fazer parte da estrutura (i.e. a aresta é livre) ela deverá ser explorada mais tarde.

Como começar o algoritmo já que no início não temos nenhuma face (triângulo) de Del(S)? Basta procedermos de maneira análoga ao que fizemos no Algoritmo de Jarvis: determinar uma aresta de conv(S). Esta aresta é com certeza uma aresta de Del(S) (para a qual ainda resta contruirmos o triângulo de Del(S) incidente a ela—i.e., esta aresta é livre). Com esta aresta podemos, usando o Teorema 12, encontrar uma face de Del(S).

O algoritmo empregará uma fila Q para armazenar as faces que vão sendo encontradas ao longo da execução do algoritmo.

Algoritmo `Delaunay`(S)
`Entrada:` Um conjunto finito S de pontos em E^2.
`Saída:` O diagrama de Delaunay dos pontos em S.
1. $Q \leftarrow \emptyset$; [Fila com as faces encontradas.]
2. $f \leftarrow$ uma face inicial de del(S);
3. Insira f na fila Q e na winged-edge estrutura T;
4. **while** $Q \neq \emptyset$ **do**
5. $f \leftarrow$ primeira face na fila Q;
6. **for** cada aresta livre e de f **do**
   [Passo correspondente ao Teorema 12.]
7. $f' \leftarrow$ face do diagrama de Delaunay que compartilha e com f;
8. Insira f' na fila Q;
9. Insira f' na estrutura winged-edge T, conectando-a com as faces já geradas que lhe são adjacentes e determinando suas arestas livres.
10. **return** T.

7.3. **Análise do algoritmo.** O algoritmo pára somente quando toda a triangularização de Delaunay tiver sido obtida. Observe que se as faces geradas (inclusive a face externa) não cobrissem o plano, alguma aresta de uma das faces estaria livre, o que não ocorre já que o algoritmo só termina quando todas as arestas livres foram exploradas. O Passo 2 pode ser feito em tempo $O(n)$. Utilizando a primitiva `InCircle` podemos executar o Passo 7 em tempo $O(n)$. A complexidade de tempo do Passo 9 também é $O(n)$ (precisamos varrer toda a estrutura T para determinarmos se uma aresta esta presente ou não). Como os Passos 7 e 9 serão executados $O(n)$ vezes (cf. Corolário 1) concluímos então que complexidade de tempo do Algoritmo `Delaunay` é $O(n^2)$.

8. Um algoritmo de divisão-e-conquista

Neste seção estudaremos um algoritmo de divisão-e-conquista que constrói o diagrama de Delaunay de um dado conjunto S de n pontos no plano em tempo $O(n \log n)$ (i.e. um algoritmo ótimo para o problema). O método

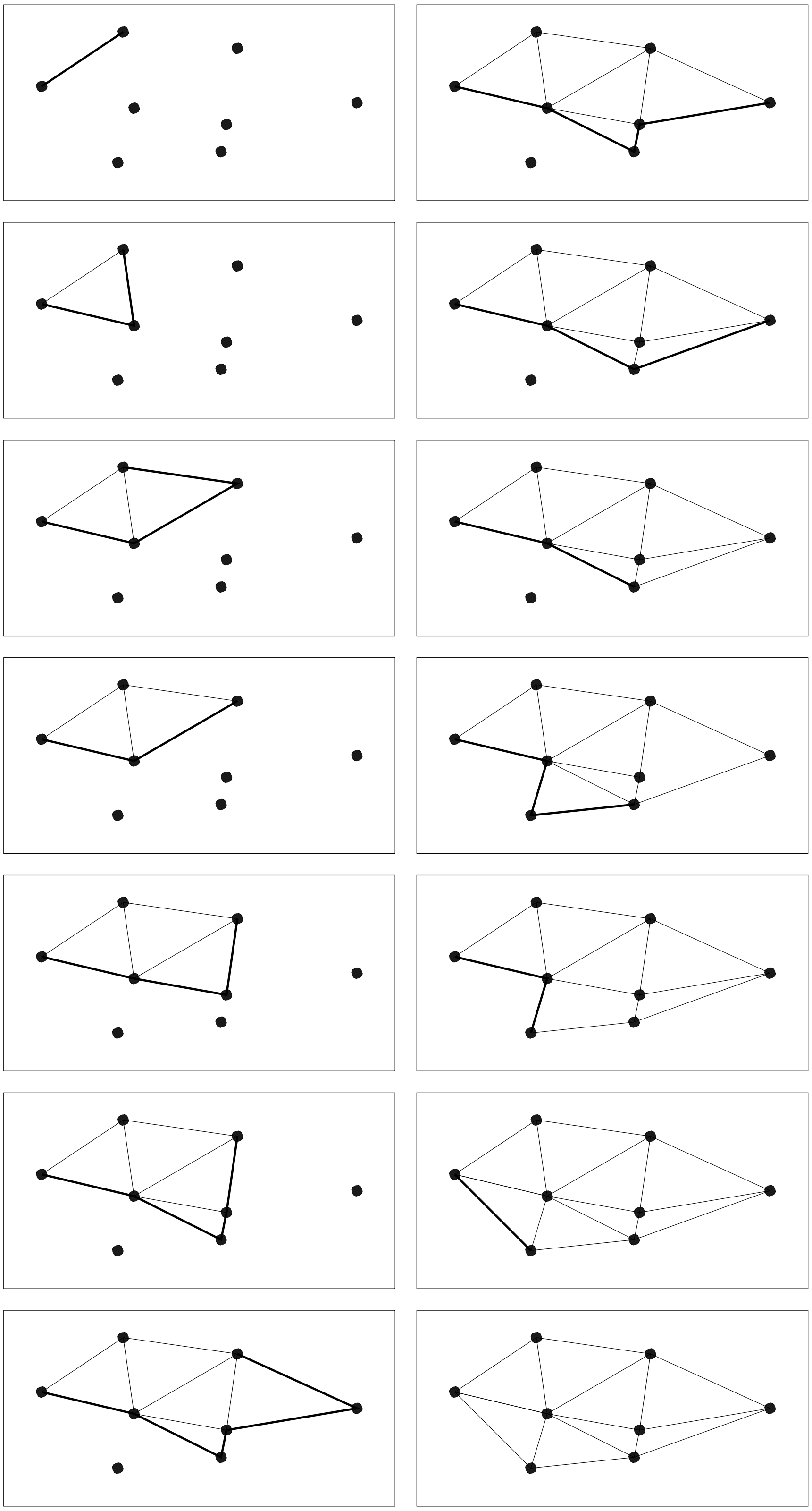


FIGURA 19. Ilustração do Algoritmo `Delaunay` em ação.

de divisão-e-conquista já foi visto várias vezes durante o andamento desta disciplina: algoritmo de complexidade de tempo $O(n \log n)$ para o Problema do Par Mais Próximo; Algoritmo `QuickHull`; e Algoritmo `MergeHull`.

O algoritmo que veremos é recursivo e terá os seguintes três passos (os típicos passos de um algoritmo de divisão-e-conquista) em cada nível da recursão:

Dividir: O problema é dividido em subproblemas de tamanho menor.

Conquistar: Os subproblemas são resolvidos recursivamente. Entretanto, se algum subproblema é suficientemente pequeno ele é resolvido diretamente.

Combinar: As soluções dos subproblemas são combinadas para obtermos uma solução do problema original.

Observação. O material desta seção foi extraído do Capítulo 4 de Figueiredo e Carvalho [6] e do artigo de Guibas e Stolfi [10]. O algoritmo que veremos é devido a Lee e Schachter [14] e é dual do algoritmo de Shamos e Hoey [17] (veja também Preparata e Shamos [16] páginas 211–220).

8.1. **A descrição do algoritmo.** Para facilitar a nossa exposição do algoritmo suporemos que qualquer subconjunto de quatro pontos de S são não-cocirculares e que qualquer dois pontos não têm a mesma x-coordenada. O algoritmo está descrito logo abaixo. Notemos que, para que o o diagrama de Delaunay de S seja contruído em tempo $O(n \log n)$, necessitamos que a fase `Combinar` seja feita em tempo $O(n)$.

Algoritmo `Delaunay2`(S)
`Entrada:` Um conjunto S de sítios em E^2.
`Saída:` O diagrama de Delaunay $\mathrm{Del}(S)$ de S.
1. **if** $|S| \leq 3$ construa $\mathrm{Del}(S)$ e **return**.
2. (`Dividir`) Particione o conjunto S em conjuntos S_1 com $\lceil n/2 \rceil$ pontos e S_2 com $\lfloor n/2 \rfloor$ pontos.
3. (`Conquistar`) Recursivamente construa os diagramas de $\mathrm{Del}(S_1)$ e $\mathrm{Del}(S_2)$.
4. (`Combinar` ou `Merge`) Tendo os diagramas $\mathrm{Del}(S_1)$ e $\mathrm{Del}(S_2)$ construa $\mathrm{Del}(S_1 \cup S_2)$.

8.2. **Pré-processamento: 'sorting step'.** Para facilitar os passos `Dividir` e `Conquistar`, antes de aplicarmos o algoritmo `Delaunay2`, ordenaremos o conjunto de pontos dado em relação as suas x-coordenadas.

8.3. **Dividir.** O conjunto de pontos S será particionado em dois subconjunto em relação a mediana das x-coordenas. Graças ao pré-processamento este passo pode ser feito rapidamente. Em outras palavras, o conjunto S será divido com relação a uma reta vertical: seja L o conjunto formado pelos sítios de S que têm x-coordenada menor ou igual a mediana das x-coordenadas dos pontos e seja R o conjunto formado pelos demais pontos de S (pontos à direita de reta vertical passando pela mediana das x-coordenadas). (Escreveremos L (*left*) e R (*right*) em vez de de S_1 e S_2, respectivamente.) Notemos que esta escolha de conjuntos L e R, juntamente com a nossa hipótese de que S não tem dois pontos com a mesma x-coordenada, garante que os diagramas Del(L) e Del(R) são disjuntos — se permitissemos vários pontos com a mesmo coordenada teríamos apenas que tormar um certo cuidado para definirmos L e R, não queremos que um conjunto fique com todos os sítios e o outro fique vazio.

8.4. **Conquistar.** O passo conquistar corresponde as chamadas recursivas onde cada subproblema será resolvido independentemente. A base da recursão é atinginda quando o conjunto de sítios em questão tem no máximo três sítios. Na base da recursão o diagrama de Delaunay desses (no máximo três) sítios pode ser facilmente construído. Por exemplo, o diagrama de Delaunay de três sítios corresponde a um triângulo que tem esses sítios como vértices ou a dois segmentos ligando os sítios (no caso do sítios serem colineares).

8.5. **Combinar: 'the merging step'.** A maior parte do trabalho deste algoritmo de divisão-e-conquista está neste passo (o mesmo fenômeno se verificou quando estudamos outros algoritmos de divisão-e-conquista). Precisamos ser cuidadosos para realizarmos este passo em tempo linear: um algoritmo ingênuo pode atingir a complexidade de tempo $O(n^2)$ para este passo, o que não nos permitiria obter a complexidade de tempo de $O(n \log n)$ para todo o algoritmo.

Chamaremos as arestas de Del(L) e Del(R) de L-L-arestas e R-R-arestas, respectivamente. As arestas de Del(S) que têm um extremo em um sítio de L e o outro extremo em um sítio de R serão ditas L-R-aresta (ou *crossing edge*). O passo `Combinar` consistirá em removermos algumas L-L e R-R-arestas e incluirmos algumas L-R-aresta.

Todas as L-R-arestas cruzam uma reta vértical l que separa o conjunto L e o conjunto R. Estes pontos de interseção de l com as L-R-arestas de Del(S) induzem uma ordem total no conjunto das L-R-arestas. Chamaremos esta ordem total de y-ordem. A L-R-aresta mais 'baixa' (menor na y-ordem) e mais 'alta' (maior na y-ordem) corresponde a aresta tangente inferiore e superior a Conv(L) e Conv(R), respectivamente. (Veja Figura 20.) O seguinte teorema nos fornece mais informações sobre a estrutura das L-R-arestas.

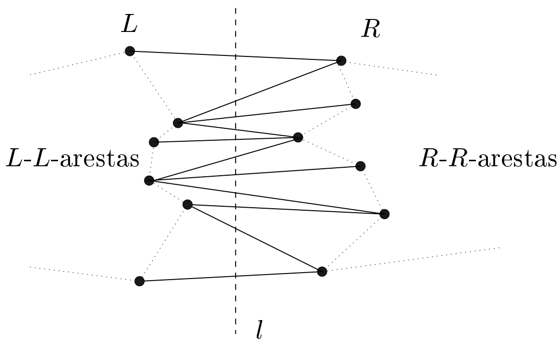


FIGURA 20. As L-R-arestas de Del(S).

Teorema 18. *Qualquer duas L-R-arestas consecutivas na y-ordem têm um vértice em comum. Ademais, o terceiro lado do triângulo que estas arestas determinam é uma L-L-aresta ou uma R-R–aresta.*

Prova. A primeira parte do teorema segue do fato de $\mathrm{Del}(S)$ ser uma triangularização dos pontos em S (cf. Teorema 14). Logo, o terceiro lado do triângulo tem ambos os seus vértices em L ou em R. Sejam p, q e r os sítios do triângulo em questão e suponhamos, sem perda de generalidade, que $p \in L$, $q, r \in R$ (ou seja, pq e pr são as L-R-arestas consecutivas na y-ordem). Precisamos mostrar que qr é uma aresta de $\mathrm{Del}(L)$. Como o triângulo $\triangle(p, q, r)$ é um triângulo de $\mathrm{Del}(L \cup R)$ temos então que qr é uma aresta de $\mathrm{Del}(L \cup R)$. Pelo Teorema 8 existe um círculo C contendo q e r tal que todos os demais sítios de S são exteriores a C. Logo, em particular, C é um círculo que contém q e r e não contém nenhum outro sítio de R. Portanto, novamente pelo Teorema 8, temos que qr é uma aresta de $\mathrm{Del}(R)$. ∎

Suponha que tenhamos encontrado todas as L-R-arestas na y-ordem entre a tangente inferior a $\mathrm{conv}(L)$ e $\mathrm{conv}(R)$ até uma aresta $p_l p_r$, $p_l \in L$ e $p_r \in R$. (Veja Figura 21.) Pelo Teorema 18 sabemos que a próxima L-R-aresta na y-ordem é incidente a p_l ou p_r. Para determinarmos esta aresta basta examinarmos os sítios de L adjacentes a p_l em $\mathrm{Del}(L)$ e os sítios de R adjacentes a p_r em R. Por outro lado, pelo Teorema 12, sabemos que a próxima L-R-aresta na y-ordem tem uma ponta em p_l ou p_r e a outra ponta em um sítios p tal que o ângulo $\angle(p_l p p_r)$ é máximo entre todos os sítios que estão à esquerda do segmento (orientado) $p_l p_r$.

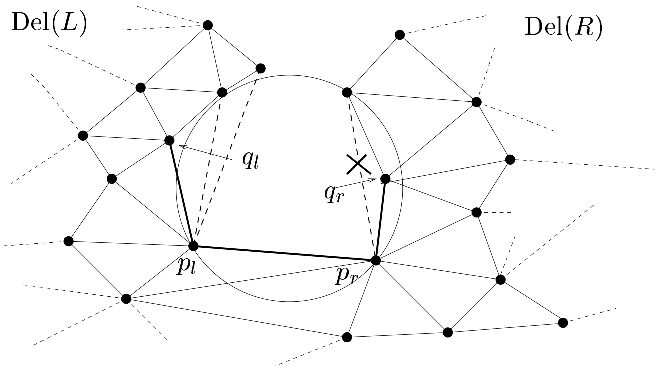


FIGURA 21. Determinando a próxima L-R-aresta de $\mathrm{Del}(L \cup R)$.

Abaixo encontra-se uma descrição do Algoritmo `MergeDelaunay` que encontra as L-R-arestas e remove as L-L e R-R-arestas que deixam de ser arestas do diagrama $\mathrm{Del}(L \cup R)$. Em cada iteração o algoritmo começa com uma L-R-aresta $p_l p_r$ e trabalha no sentido de encontrar a próxima L-R-aresta (próxima em relação a y-ordem). Dentre os sítios adjacentes a p_l (p_r, respectivamente) o algoritmo determina um sítio q_l (q_r, respectivamente) candidato a ponta da próxima L-R-aresta. (Veja Figura 21.) No final da iteração, o algoritmo determina, entre q_l e q_r qual sítio que será a ponta da próxima L-R-aresta (a próxima L-R-arestas será $p_l q_r$ ou $p_r q_l$).

Para encontrar q_l o algoritmo procederá como segue. Sejam $q_1, q_2, \ldots, q_k$ os sítios de L adjacentes a p_l em Del(L) listados no sentido anti-horário. Se o círculo determinado pelo triângulo $\triangle(p_l, p_r, q_1)$ contém o ponto q_2 concluímos imediatamente que $p_l q_1$ não pode ser uma aresta de Del($L \cup R$). Neste caso, eliminamos a L-L-aresta $p_l q_1$ e repetimos o mesmo teste com os sítios q_2 e q_3 no lugar de q_1 e q_2, respectivamente. O sítio procurado q_l é o primeiro sítios q_t tal que o círculo determinado pelo triângulo $\triangle(p_l, p_r, p_t)$ não contém p_{t+1}. (Por que isto funciona?) A determinação de q_r é análoga, a única diferença é que examinamos os sítios adjacentes a p_r em Del(R) no sentido horário ao ínves de anti-horário.

Para garantirmos que a determinação de q_l e q_r seja feita em tempo constante por aresta incidente a p_l e p_r, suporemos que Del(L) e Del(R) sejam dadas através de suas estruturas *winged-edge* (ou qualquer outra estrutura que represente um grafo planar de uma maneira 'descente' — ou seja, faz as operações necessárias rapidamente.

Notação. Se u e v são vértices adjacentes de um diagrama planar H então denotaremos por $\texttt{esq}(u, v)$ o vértice de H adjacente a u que está imediatamente à esquerda do segmento (orientado) uv no sentido anti-horário. Analogamente, denotaremos $\texttt{dir}(u, v)$ o vértice de H adjacente a u que está imediatamente à direita do segmento uv no sentido horário. (Veja Figura 22.) Notemos que se H é representado por uma estrutura *winged-edge* então as operações `esq` e `dir` podem ser feitas em tempo constante (i.e. $O(1)$).

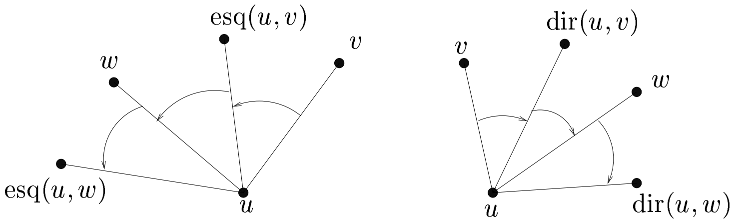


FIGURA 22. Ilustração da definição de $\texttt{esq}(u, v)$ e $\texttt{dir}(u, v)$.

Algoritmo `MergeDelaunay`(Del(L),Del(R))
`Entrada:` Dois diagramas de Delaunay Del(L) e Del(R) tal que os sítios de L estão à esquerda dos sítios de R.
`Saída:` O diagrama de Delaunay Del($L \cup R$) de $L \cup R$.
1. $T \leftarrow$ Del(L) $\cup$ Del(R);
   [T é uma estrutura winged-edge.]
2. Determine o segmento $p_l p_r$ tangente inferior a conv(L) e conv(R).
   [Veja o Algoritmo `Tangente-Inferior`.]
3. **while** $p_l p_r$ não é a tangente superior a conv(L) e conv(R) **do**
   [**Determinação do candidato** q_l.]
4. $\quad q_l \leftarrow \texttt{esq}(p_l, p_r)$; $q \leftarrow \texttt{esq}(p_l, q_l)$;
5. $\quad$ **while** $\texttt{Left}(p_l, p_r, q_l)$ e q pertence ao círculo determinado por p_l, p_r e q_l **do**
6. $\qquad$ Remova a L-L-aresta $p_l q_l$ de T;

```
7.        q_l ← q; q ← esq(p_l, q);
8.     if Left(p_r, p_l, q_l)
9.        then não temos candidado a q_l.
   [Determinação do candidato q_r.]
10.    q_r ← dir(p_r, p_l); q ← dir(p_r, q_r);
11.    while Left(p_l, p_r, q_r) e q pertence ao círculo determinado
          por p_l, p_r e q_r do
12.       Remova a R-R-aresta p_r q_r de T;
13.       q_r ← q; q ← dir(p_r, q);
14.    if Left(p_r, p_l, q_r)
15.       then não temos candidado a q_r.
   [Determinação da próxima L-R-aresta.]
16.    Seja p o sítio entre os candidatos cujo ângulo ∠(p_l p p_r) é máximo.
   [Se não existirem candidatos, então p_l p_r é tangente superior.]
17.    if p ∈ L
18.       then insira p_r p em T;
19.       else insira p_l p em T;
20. Return T.
```

As operações primitivas usadas pelo algoritmo são `Lefton` e `InCircle`. Nós podemos intuitivamente ver o algoritmo como uma bolha (bidimensional) que vai subindo entre Del(L) e Del(R) e que a cada instante ocupa a maior área possível. Cada vez que a bolha atinge um novo sítio a próxima L-R-aresta é determinada (e talvez algumas L-L- ou R-R-arestas tenham que ser removidas).

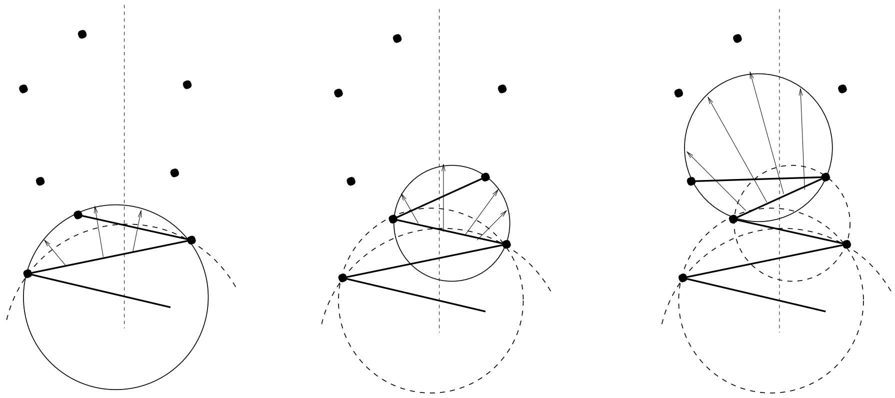


Figura 23. Ilustração do algoritmo `MergeDelaunay` através de 'bolhas subindo.'

8.6. **Corretude do Algoritmo `MergeDelaunay`.** Para mostrarmos a corretude do algoritmo precisamos verificar que:

1. o algoritmo produz todas as L-R-arestas de Del($L \cup R$);
2. as L-L e R-R-arestas removidas pelo algoritmo não são arestas de Del($L \cup R$); e
3. ao final do Algoritmo `MergeDelaunay` o diagrama resultante corresponde a triangularização de Delaunay de $L \cup R$.

Teorema 19. *Seja S um conjunto de sítios em E^2 e considere um sítio particular p e uma reta l passando por p. Assuma por conveniência que l é uma reta horizontal. Sejam $p_1, p_2, \dots, p_k$ $(k \geq 1)$ os sítios de S acima de l que são vizinhos de p em $Del(S)$ listados em ordem anti-horária. Seja q um ponto em l à direita de p e seja $\alpha_i := \angle(pp_iq)$ $(i = 1, \dots, k)$. Então existe um índice j, $1 \leq j \leq k$ tal que*

1. $\alpha_i < \alpha_{i+1}$ *para* $i = 1, \dots, j-1$; *e*
2. $\alpha_i > \alpha_{i+1}$ *para* $i = j, \dots, k$.

Prova. Primeiro mostraremos que se q_i denota a intersecção da circunferência determinada por triângulo $\triangle(p, p_i, p_{i+1})$ $(i = 1, \dots, k-1)$ com a reta l então a seqüência $q_1, q_2, \dots, q_{k-1}$ se move monotonicamente para a esquerda. (Veja Figura 24.)

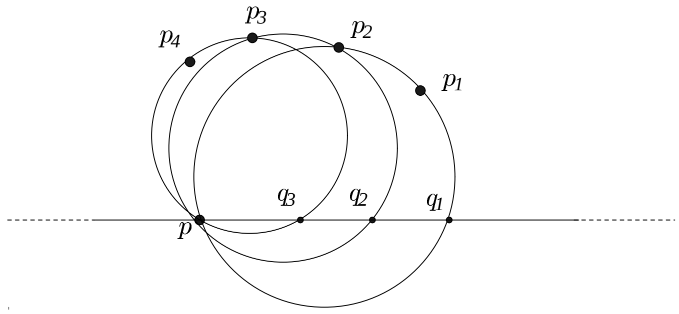


FIGURA 24. A seqüência $q_1, \dots, q_{k-1}$ se move monotonicamente para a esquerda.

Para $i = 1, \dots, k-3$ temos que o sítio p_{i+2} não pertence ao círculo determinado por $\triangle(p, p_i, p_{i+1})$ (cf. Teorema 2). Logo, $\angle(pp_{i+2}p_{i+1}) + \angle(pq_ip_{i+1}) < \pi$ Portanto, q_i não pertence ao círculo determinado por $\triangle(p, p_{i+1}, p_{i+2})$ o que implica que q_{i+1} está à esquerda de q_i.

Para terminarmos a prova do teorema basta notarmos que se q_i está à direita de q então q pertence ao círculo determinado por $\triangle(p, p_i, p_{i+1})$ (o que implica que $\angle(pqp_i) + \angle(pp_{i+1}p_i) > \pi$, ou, equivalentemente, $\angle(p_{i+1}pq) + \angle(p_{i+1}p_iq) < \pi$). Logo, p_{i+1} pertence ao círculo determinado por $\triangle(p, q, p_i)$. Analogamente, se q_i esta à esquerda de q então p_{i+1} não pertence ao círculo determinado por $\triangle(p, p_i, q)$. Com isto completamos a prova do teorema. ∎

A primeira L-R-aresta determinada pelo algoritmo na Linha 2 é certamente uma aresta de $\text{Del}(L \cup R)$ (cf. Teorema 8). Os Teorema 18 e 19 implicam que as Linhas 5–9 do Algoritmo `MergeDelaunay` encontra o sítios $q_l \in L$ tal que o ângulo $\angle(p_lq_lp_r)$ é máximo. Analogamente, os Linhas 10–15 determinam o sítio $q_r \in R$ tal que o ângulo $\angle(p_lq_rp_r)$ é máximo. A Linha 16 decide qual sítios $p \in \{q_l, q_r\}$ máximiza o ângulo $\angle(p_lpp_r)$. Logo, pelo Teorema 12, a L-R-aresta inserida na estrutura T pelas Linhas 17–19 é a aresta de $\text{Del}(L \cup R)$ sucessora da L-R-aresta p_lp_r na y-ordem. Isto implica que o Algoritmo `MergeDelaunay` encontra todas as L-R-arestas de $L \cup R$.

Teorema 20. *As L-L e R-R-arestas removidas pelo algoritmo não são arestas do diagrama de Delaunay de $(L \cup R)$.*

Prova. Uma L-L-aresta $p_l q_l$ é removida pelo algoritmo na Linha 6 somente se o círculo determinado por q_l está à esquerda do segmento $p_l p_r$ e o sítio q imediatamente à esquerda do segmento $p_l q_l$ no sentido anti-horário pertence ao círculo determinado por $\triangle(p_l, p_r, q_l)$. Como $p_l p_r$ é uma aresta da triangularização de Delaunay de $L \cup R$ temos que se $p_l q_l$ é uma aresta de $\mathrm{Del}(L \cup R)$ então $\triangle(p_l, p_r, q_l)$ é um triângulo de $\mathrm{Del}(L \cup R)$. Isto contradiz o Teorema 7, já que q é um sítio contido no interior do $C(v)$ onde v é o vértice de Voronoi correspondente ao triângulo $\triangle(p_l, p_r, q_l)$.

Analogamente mostra-se que as L-L-arestas removidas pela Linha 12 do algoritmo não são arestas do diagrama de Delaunay de $L \cup R$. ∎

Teorema 21. *Ao final do Algoritmo* `MergeDelaunay` *o diagrama resultante corresponde a triangularização de Delaunay de* $L \cup R$

Prova. Primeiro mostraremos que o diagrama obtido é uma triangularização de $L \cup R$. No ínicio do algoritmo $p_l p_r$ corresponde a tangente inferior a $\mathrm{conv}(L)$ e $\mathrm{conv}(R)$ e depois de cada iteração o segmento $p_l p_r$ terá uma de uma extremidade mais alta (maior y-coordenada). O laço das Linhas 3–19 termina quando $p_l p_r$ alcança a tangente superior a $\mathrm{conv}(L)$ e $\mathrm{conv}(R)$. As faces interiores do diagrama serão ou faces de $\mathrm{Del}(L)$ ou $\mathrm{Del}(R)$ ou faces formadas por duas L-R-arestas e um outra L-L- ou R-R-aresta. Em ambos os casos as faces do novo diagrama formam triângulos.

Precisamos verificar agora que a triangularização obtida de $L \cup R$ é de fato uma triangularização de Delaunay, ou seja, que as arestas do diagrama resultante são arestas de $\mathrm{Del}(L \cup R)$. Devido aos Teoremas 19 e 20 precisamos apenas verificar que arestas do diagrama resultante que são L-L- ou R-R-arestas são arestas de $\mathrm{Del}(L \cup R)$. Seja pq uma L-L-aresta que não foi removida pelo algoritmo. Se pq não é uma aresta de $\mathrm{Del}(L \cup R)$ então existe uma aresta rs de $\mathrm{Del}(L \cup R)$ que tem intersecção não vazia com pq. Já que pq é uma L-L-aresta então um dos sítios r ou s deve ser um sítios de R. Mas isto implica que rs é uma L-R-aresta e portanto foi corretamente encontrada pelo algoritmo. Logo, se pq não é uma aresta de $\mathrm{Del}(L \cup R)$ ela deve ter sido removida pelas Linhas 5–7 do algoritmo. Analogamente se verifica que toda R-R-aresta do diagrama resultante ao final do algoritmo é uma aresta de $\mathrm{Del}(L \cup R)$. ∎

Estes teoremas completam a prova da corretude do Algoritmo `MergeDelaunay`—o que implica a corretude do Algoritmo `Delaunay2`.

8.7. **Análise do algoritmo.** Seja n o número de sítios de S. A complexidade de tempo do pré-processamento (ordenação dos sítios) que é $O(n \log n)$.

No Algoritmo `MergeDelaunay` (passo `Combinar`) a Linha 2 pode ser executada em tempo $O(n)$ (veja o Algoritmo `Tangente-Inferior`). O laço entre as Linhas 3–19 são executados $O(n)$ vezes. Em cada iteração do laço das Linhas 5–7 uma L-L-arestas é removida. Logo, esse laço será executado $O(n)$ vezes. Analogamente, o laço das Linhas 11–13 também será executado $O(n)$ vezes. Como a execução de cada Linha i, $(i = 3, \ldots, 19)$ gasta tempo $O(1)$ concluímos que a complexidade de tempo do Algoritmo `MergeDelaunay` é $O(n)$.

Se $T(n)$ é a função complexidade de tempo do Algoritmo `Delaunay2`, sem contar o custo do pré-processamento, temos que

$$T(n) \leq T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + cn = O(n \log n),$$

onde c é uma constante. Logo, a complexidade de tempo do Algoritmo `Delaunay2` é $O(n \log n)$.

9. Aplicações

Os diagramas de Voronoi e Delaunay podem ser usados para resolver diversos problemas. Nesta seção apresentaremos alguns destas aplicações.

Observação. O material desta seção foi extraído do: Capítulo 4 de Figueiredo e Carvalho [6]; Capítulos 5 e 6 de Preparata e Shamos [16]; e Capítulo 5 de O'Rourke [15].

9.1. **O Problema do Par Mais Próximo.** O *Problema do Par Mais Próximo* consiste em: *dado*: um conjunto S de n pontos em E^2; *encontrar:* um par de pontos de S cuja distância entre eles é mínima. Um algoritmo ingênuo resolveria este problema em tempo $O(n^2)$. Como já vimos este problema pode ser resolvido em tempo ótimo $O(n \log n)$ por um algoritmo de divisão-e-conquista. Pelo Teorema 5 temos que cada par mais próximo corresponde a uma aresta de $\mathrm{Vor}(S)$ e a uma aresta de $\mathrm{Del}(S)$. Para encontrarmos um par mais próximo entre os pontos de S basta então examinarmos todas as arestas de $\mathrm{Vor}(S)$ ou $\mathrm{Del}(S)$. Este algoritmo gasta tempo $O(n)$ já que número de arestas em ambos os diagramas é $O(n)$ (cf. Corolário 1). Se combinarmos este algoritmo com um algoritmo que constroi $\mathrm{Vor}(S)$ ou $\mathrm{Del}(S)$ em $O(n \log n)$ teremos um algoritmo ótimo para o Problema do Par Mais Próximo.

Teorema 22. *O Problema do Par Mais Próximo pode ser resolvido em tempo $O(n \log n)$. Ademais, dado o diagrama de Voronoi Vor(S) ou o diagrama de Delaunay Del(S) de um conjunto de pontos S podemos encontrar o par mais-próximo de S em tempo linear.* ∎

9.2. **Todos os vizinhos mais próximos.** O *Problema de Todos os Vizinhos Mais Próximos* (*All Nearest Neighbors*) consiste em: *dado*: um conjunto S de n pontos em E^2; *encontrar*: para cada ponto $p \in S$ um ponto em S que seja mais-próximo de p. Novamente, devido ao Teorema 5, para encontrarmos um (ou todos) os pontos mais próximos de um ponto $p \in S$ basta procurarmos entres seus vizinhos em $\mathrm{Del}(S)$ (ou, equivalentemente, examinar as arestas da região de Voronoi $V(p)$). Como o número de arestas é $O(n)$, a determinação de todos os vizinhos mais próximos é feita em $O(n)$. (Note que aqui estamos fazendo análise amortizada.)

Teorema 23. *O Problema de Todos os Vizinhos Mais Próximos pode ser resolvido em tempo $O(n \log n)$. Ademais, dado o diagrama de Voronoi Vor(S) ou o diagrama de Delaunay Del(S) de um conjunto de pontos S podemos encontrar o ponto mais-próximo de cada ponto de S em tempo linear.* ∎

∎

[Uma maneira de capiturar a essência do algoritmo para o Problema de Todos os Vizinhos Mais Próximos é através do *grafo dos vizinhos mais próximos* (*Nearest Neighbor Graph*) de S, denotado por NNG(S). Este grafo tem como vértices os pontos de S e tem uma aresta entre dois pontos se um dos pontos é o vizinho mais próximo do outro. Devido ao Teorema 5 podemos verificar que este grafo pode ser representado como um diagrama planar que esta contido em Del(S).]

9.3. **Fecho convexo planar.** O *Problema do Fecho Convexo Planar* consiste em: *dado*: um conjunto S de n pontos em E^2; *construir*: o fecho convexo Conv(S) dos pontos em S. Devido ao Teorema 6 temos que o fecho convexo de S é determinado pelo diagrama de Voronoi de S ou, equivalentemente, pelo diagrama de Delaunay de S (as arestas do fecho convexo são arestas de Delaunay de Del(S)). Se o diagram de Voronoi de S é dado através de, digamos, uma estrutura winged-edge então examinar as arestas de Voronoi até que um raio e (aresta de Voronoi que é uma semireta) de uma região de Voronoi $V(p)$ seja encontrado. Depois, examinando as arestas de $V(p)$ podemos obter um novo raio e' de Vor(S) e assim por diante. Desta forma obteremos todos os sítios que têm as suas respectivas regiões de Voronoi ilimitadas. Estes sítios são os vértices extremos de Conv(S).

Teorema 24. *O Problema do Fecho Convexo Planar pode ser resolvido em tempo $O(n \log n)$. Ademais, dado o diagrama de Voronoi Vor(S) ou o diagrama de Delaunay Del(S) de um conjunto de pontos S podemos construir o fecho convexo Conv(S) em tempo linear.* ∎

9.4. **Triangularização maximizando o ângulo mínimo.** Uma *seqüência de ângulos de uma triangularização T* é uma seqüência $(\alpha_1, \alpha_2, \ldots, \alpha_{3t})$ dos ângulos dos triângulos que fazem parte de T em ordem crescente. Sejam T e T' triangularizações de um conjunto de pontos S, diremos que $T \succeq T'$ se a seqüência de ângulos de T é lexicograficamente maior ou igual a seqüência de ângulos de T': ou $\alpha_1 > \alpha'_1$ ou $\alpha_1 = \alpha'_1$ e $\alpha_2 > \alpha'_2$ ou $\alpha_1 = \alpha'_1$ e $\alpha_2 = \alpha'_2$ e $\alpha_3 > \alpha'_3$, ou ... Em Edelsbrunner [5] páginas 302 e 303 o seguinte teorema é provado.

Teorema 25. *A triangularização de Delaunay $T :=$ Del(S) é maximal em relação a ordem $\succeq$ das triangularizações de um conjunto S de pontos de E^2.* ∎

Em particular o teorema acima implica que a triangularização de Delaunay maximiza o menor ângulo.

Teorema 26. *O problema de encontrar uma triangularização T de um conjunto S de pontos tal que o menor ângulo de um triângulo em T seja máximo pode ser resolvido em tempo $O(n \log n)$.* ∎

9.5. **Maior círculo vazio.** O *Problema do Maior Círculo Vazio* (*Largest Empty Cycle*) consiste em: *dado*: um conjunto S de n pontos (sítios) em E^2; *encontrar:* o círculo com maior raio que tem o seu centro no fecho convexo de S e não contém nenhum ponto de S em seu interior.

Para cada ponto $p \in \operatorname{conv}(S)$ denotaremos por $f(p)$ o valor do maior raio de um círculo com centro em p que não contenha nenhum sítio de S em seu interior. (Existe um número potencialmente infinito de pontos em conv(S)

que são candidatos a maximizar a função $f(p)$.) Chamaremos de *círculo vazio* a todo círculo que não contenha em seu interior nenhum ponto do conjunto dado S.

9.5.1. *Centro no interior de conv(S).* Seja p no interior de conv(S) e seja $C(p)$ o círculo (vazio) de centro p e raio $f(p)$. Suponha primeiro que $C(p)$ contém exatamente um ponto $q \in S$. Se movermos o ponto p para um ponto p' um pouco mais distante de q na direção do vetor (p,q) teremos que $f(p') > f(p)$. Logo, se p é um ponto de conv(S) que maximiza $f(p)$ então $C(p)$ não pode conter um único ponto de S. (Veja Figura 25.) Suponha agora que $C(p)$ contém dois pontos q e r de S. Se movermos o ponto p para o ponto p', mais distante do segmento qr ao longo da mediana do segmento qr teremos que $f(p') > f(p)$. (Veja Figura 25.) Assim, se p é um ponto de S que maximiza $f(p)$ então $C(p)$ não pode conter somente dois pontos de S. Agora se $C(p)$ contém três pontos de S não é possível movermos p um pouco e obter um ponto p' tal que $f(p') > f(p)$. Portanto, devido ao Teorema 3 podemos concluir o seguinte:

Lema 2. *Seja o p é um ponto no interior de conv(S) tal que o maior raio $f(p)$ (o maior raio de um círculo vazio com centro em p) é máximo. Então p é um vértice de Voronoi de Vor(S).* ∎

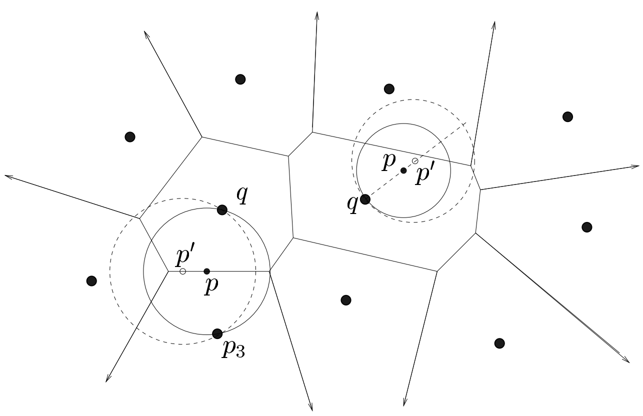


FIGURA 25. Círculos de centro p no interior de conv(S): círculo contendo exatamente um sítio e círculo contendo exatamente dois sítios.

9.5.2. *Centro na fronteira de conv(S).* Seja p um ponto na fronteira de conv(S) e seja $C(p)$ o círculo de centro p e raio $f(p)$. Analogamente ao que foi feito acima podemos concluir que se p é um ponto da fronteira de conv(S) que maximiza $f(p)$ então $C(p)$ não pode conter um único ponto de S. (Veja Figura 26.) O raciocínio que usamos para mostrar que $C(p)$ deve ter mais do que dois pontos de S não funciona para um ponto p na fronteira de conv(S) (pois ao movermos p para p' podemos obter um ponto p' fora de conv(S)).

Lema 3. *Seja p um ponto da fronteira do fecho convexo tal que $f(p)$ (o maior raio de um círculo vazio com centro em p) é máximo. Então p pertence a uma aresta de Voronoi.* ∎

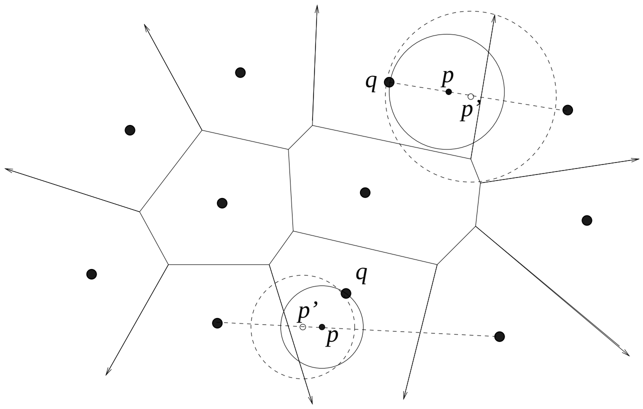


FIGURA 26. Círculo de centro p na fronteira de conv(S): círculo contendo exatamente um sítio.

Devido aos lemas anteriores vemos que é possível resolver o Problema do Maior Círculo Vazio em tempo $O(n \log n)$ se numa fase de pré-processamento construirmos Vor(S). O algoritmo está descrito abaixo.

Algoritmo `Maior-Circulo-Vazio`(S)
`Entrada:` Um conjunto S de n pontos em E^2.
`Saída`: Um ponto $p \in$ conv(S) tal que o raio $f(p)$ do círculo vazio de centro p é máximo.
1. Construa o diagrama de Voronoi de S. [Custa tempo $O(n \log n)$.]
2. $max \leftarrow 0$; $p \leftarrow$ `null`;
3. **for** cada vértice de voronoi v de Vor(S) **do**
4. **if** v está em P
5. **then** Compute o maior círculo vazio de centro v e atualize p e max (se necessario).
6. **for** cada aresta de Voronoi e de Vor(S) **do**
7. Calcule a intersecção q de e com a fronteira de P.
8. Calcule $f(q)$ o raio do círculo vazio com centro em q e atualize p e max (se necessario).
9. **return** p e max.

Teorema 27. *O Problema do Maior Círculo Vazio pode ser resolvido em tempo $O(n \log n)$.* ∎

9.6. **Árvore geradora Euclidiana mínima.** Uma *árvore geradora Euclidiana mínima* (*Euclidean minimum spanning tree*) de um conjunto S de pontos em E^2 é uma árvore que tem como vértices os ponto de S, tem como arestas segmentos de reta com extremos em S e que entre as árvores que interligam todos os pontos de S e uma cujo comprimento total de suas

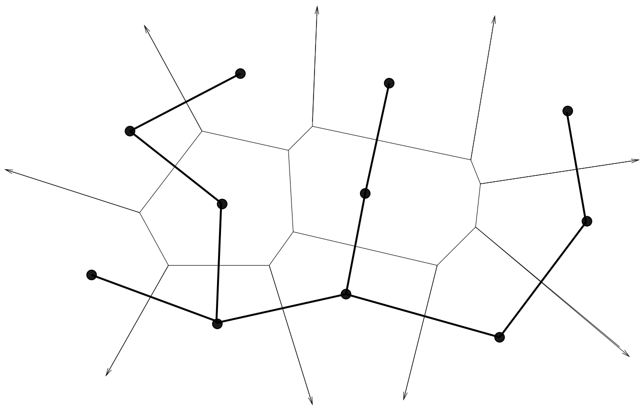


FIGURA 27. Uma árvore geradora mínima dos pontos de S

aresta é mínimo. Um exemplo de árvore geradora mínima e mostrado na Figura 27.

Consideraremos aqui o *Problema da Árvore Geradora (Euclidiana) Mínima*: *dados:* uma conjunto S de n pontos em E^2; *encontrar:* uma árvore gerado de S cujo comprimento de suas arestas é mínimo. Um algoritmo conhecido para resolver este problema é o seguinte Algoritmo de `Kruskal` (que resolve um problema mais geral: *dados*: um grafo $G = (V, E)$ e uma função 'comprimento' $l : E \to \mathbb{R}$; *encontrar*: uma árvore geradora T de G de comprimento mínimo (i.e. $\sum_{e \in T} l(e)$ é mínimo)):

Algoritmo `Kruskal`(S)
`Entrada:` Um conjunto S de pontos em E^2.
`Saída:` Uma árvore geradora T de S de comprimento mínimo.
1. Sejam todos os segmentos $e_1, e_2, \ldots, e_k$ com extremidades em S
   em ordem crescente dos seus comprimentos.
2. $T \leftarrow \emptyset$; $i \leftarrow 1$;
3. **while** T não é uma árvore geradora **do**
4.   **if** $T + e_i$ é acíclico
5.   **then** $T \leftarrow T + e_i$
6. $i \leftarrow i + 1$;
7. **return** T.

A complexidade de tempo do algoritmo acima é $O(n^2 \log n)$. O tempo para ordenar os $\binom{n}{2}$ segmentos com extremidades no pontos de S. O tempo total gasto pelas Linhas 3–6 também é $O(n \log n)$). Como observaremos a seguir, com o auxílio do diagrama de Delaunay para S, o algoritmo acima pode ser acelerado para uma complexidade de tempo de $O(n \log n)$.

[Note a semelhança do algoritmo acima e do Algoritmo `Guloso` (da Lista de Exercícios 5) para encontrar uma triangularização de peso mínimo de um dado conjunto S de pontos de E^2. O Algoritmo `Kruskal` encontra uma árvore geradora (Euclidiana) mínima de S enquanto o Algoritmo `Guloso` falha na sua missão de encontrar um triangularização de S de peso mínimo.]

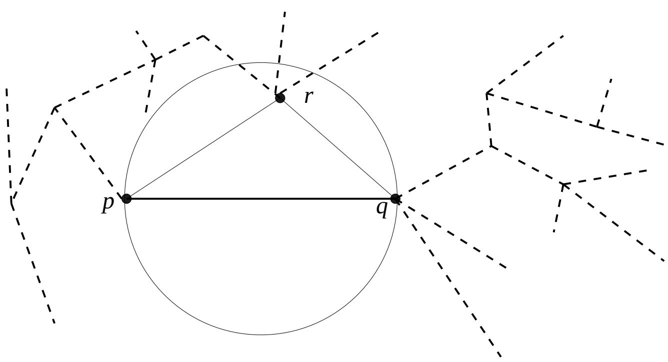


FIGURA 28. Ilustração da prova do Teorema 28.

A observação chave para obtermos um algoritmo mais rápido para o Problema da Árvore Geradora (Euclidiana) Mínima é o seguinte teorema

Teorema 28. *Seja T uma árvore geradora mínima de um conjunto S de pontos em E^2. Então as arestas T são arestas do diagrama de Delaunay de S.*

Prova. Seja $e = pq$ uma aresta de uma árvore geradora mínima de S. Se pq não pertence a Del(S) então, pelo Teorema 4, o círculo que tem pq como diâmetro deve conter algum ponto de S em seu interior. Seja r um tal ponto. Quando removemos e de T o conjunto S é particionado em duas componentes conexas que podem ser ligadas pela aresta pr ou qr. Como ambas estas arestas são mais curtas que a aresta e nós temos uma contradição. (Veja Figura 28.) ∎

O teorema acima mostra como podemos melhorar o algoritmo de Kruskal. Primeiro, em um pré-processamento, construirmos em $O(n \log n)$ o diagrama de Delaunay de S e ordenamos as $O(n)$ arestas do diagrama (cf. Corolário 1) em tempo $O(n \log n)$. Agora, as Linhas 2–7 do algoritmo de Kruskal podem ser aplicadas nesta arestas ordenadas de Del(S) para obtermos uma árvore geradora mínima de S em tempo $O(n \log n)$ (veja, por exemplo, o Capítulo 24 de [1]).

Teorema 29. *O Problema da Árvore Geradora Mínima pode ser resolvido em tempo $O(n \log n)$. Ademais, dado o diagrama de Voronoi Vor(S) ou o diagrama de Delaunay Del(S) de um conjunto de pontos S podemos encontrar uma árvore geradora Euclidiana mínima de S em tempo linear.* ∎

10. Exercícios

1. (Estrutura *winged-edge*) Considere um politopo representado por uma estrutura winged-edge.
   (a) Escreva um algoritmo que, dada uma face f, obtém todos os vértices desta face em tempo linear no número vértices de f.
   (b) Escreva uma algoritmo que, dado um vértice v, obtém todos os vértices adjacentes a v em tempo linear no número de arestas incidentes a v.

2. (Del(S) $\Rightarrow$ Vor(S)) Descreva um algoritmo que: *dado*: o diagrama de Delaunay Del(S) de um conjunto de sítios S; *constrói*: o diagrama de Voronoi Vor(S). Tente fazer com que seu algoritmo tenha complexidade de tempo linear.
3. (Vértice de Delaunay de grau grande) Descreva conjunto S de n pontos (n arbitrário), que não contenha quatro pontos cocirculares, tal que o diagrama de Delaunay Del(S) tem um vértice (sítio) de grau $n-1$.
4. (Uma aresta de Del(S) a partir de S_1 e S_2) Seja S um conjunto de pontos (sítios) de E^2 e sejam S_1 e S_2 subconjuntos não-vazios de S tal que $S_1 \cup S_2 = S$ e $S_1 \cap S_2 = \emptyset$ (ou seja, (S_1, S_2) é uma partição de S). Prove que se uv é um segmento de menor comprimento entre os segmentos em $\{s_1 s_2 \mid s_1 \in S_1 \text{ e } s_2 \in S_2\}$, então uv é uma aresta do diagrama de Delaunay.
5. (Exercício 5.3.3.1 de [15] — Polígono regular) Descreva os diagramas de Voronoi e Delaunay dos vértices de um polígono regular.
6. (Exercício 5.4.5.2 de [15] — Diagrama de Voronoi unidimensional) Um diagrama de Voronoi unidimensional de um conjunto $S = \{p_1, \ldots, p_n\}$ de sítios na reta (digamos, no eixo das abscissas) é um conjunto de pontos Vor(S) $= \{x_1, \ldots, x_{n-1}\}$ tal que x_i é o ponto médio do segmento $p_i p_{i+1}$.

   Descreva um critério que permite que determinemos se um dado conjunto $\{x_1, \ldots, x_{n-1}\}$ de pontos é um diagrama de Voronoi unidimensional. Qual é a complexidade do algoritmo resultante do critério que você obteve?
7. Prove que se S é um conjunto de n pontos de E^2 com no máximo k pontos cocirculares então cada vértice do diagrama de Voronoi Vor(S) de S tem grau entre 3 e k.
8. (Grafo de vizinhanca relativa) Uma *grafo de vizinhanca relativa* (*relative neighborhood graph*) de um conjunto S de n pontos de E^2, denotado por RNG(S), é um grafo cujo conjunto de vértices é S e existe uma aresta ligando dois pontos p e q, $\{p, q\} \subseteq S$, se

$$\text{distância}(p,q) \leq \min_{r \in S \setminus \{p,q\}} \max\{\text{distância}(p,r), \text{distância}(q,r)\}.$$

   Esta desigualdade determina uma região 'proibida' para o ponto r se p e q são adjacentes no grafo de RNG(S). Esta região é formada pela intersecção dos círculos de centro p e q e raio distância(p,q).
   (a) Prove que toda aresta de RNG(S) é uma aresta de Del(S).
   (b) (MST $\subseteq$ RNG) Prove que toda aresta de uma árvore geradora Euclidiana mínima de S (MST(S)) é uma aresta de RNG(S).
   (c) Descreva um algoritmo de complexidade de tempo $O(n^2)$ que constrói o grafo de vizinhança relativa de um dado conjunto S de n pontos. Mostre a corretude e analise a complexidade do seu algoritmo.
9. (Atualização dinâmica do diagrama de Delaunay) Dado o digrama de Delaunay Del(S) de um conjunto S de n pontos de E^2 e um ponto $p \in S$ descreva um algoritmo que constrói o diagrama de Delaunay de $S \setminus \{p\}$. A complexidade de tempo do seu algoritmo deve ser $O(k \log k)$

onde k é o número de arestas adjacentes ao sítio p. Mostre a corretude e analise a complexidade do seu algoritmo.

10. (Construção do diagrama de Delaunay *on-line*) Seja S um conjunto de n sítios em E^2 e seja p um sítio tal que $p \notin S$ e $p \in \mathrm{conv}(S)$.
    (a) Mostre que se p é um sítio no interior do triângulo $\triangle(a, b, c)$ do diagrama $\mathrm{Del}(S)$ então pa, pb e pc são arestas de $\mathrm{Del}(S \cup \{p\})$.
    (b) Mostre que se p é um sítio da aresta ab do diagrama $\mathrm{Del}(S)$ que é compartilhada pelos triângulos $\triangle(a, b, c)$ e $\triangle(d, b, a)$ de $\mathrm{Del}(S)$ então pa, pb, pc e pd são arestas do digrama $\mathrm{Del}(S \cup \{p\})$.
    (c) Chamaremos de *suspeitas* as arestas de $\mathrm{Del}(S)$ que suspeitamos que não fazem parte de $\mathrm{Del}(S \cup \{p\})$. Após as operações descritas nos itens (a) e (b) quais arestas de $\mathrm{Del}(S)$ são *supeitas*? Mostre como testar em tempo constante se uma aresta suspeita pertence ou não a $\mathrm{Del}(S \cup \{p\})$.
    (d) Se constatarmos que uma aresta suspeita ab não pertence a $\mathrm{Del}(S \cup \{p\})$ então qual aresta deverá ser inserida no diagrama? Depois de inserirmos esta aresta quais arestas que eram `não-suspeitas` passam a ser `suspeitas`?
    (e) Quantas vezes uma aresta `suspeita` será testada?
    (f) Dados o diagrama de Delaunay de um conjunto S de n sítios de E^2, um sitio $p \notin S$ e o triângulo $\triangle(a, b, c)$ do diagrama de Delaunay de S que contém p, descreva um algoritmo que constrói o diagrama de Delaunay de $S \cup \{p\}$. A complexidade do seu algoritmo deve ser $O(k)$ onde k é o número de arestas inseridas mais o número de arestas removidas. (Se você não conseguir um algoritmo $O(k)$ então tente descrever um algoritmo $O(n)$.) Mostre a corretude e analise a complexiade do seu algoritmo. (Se $p \notin \mathrm{conv}(S)$ então $\mathrm{Del}(S \cup \{p\})$ pode ser construído através do Algoritmo `MergeDelaunay`.)
    [Note que o algoritmo pedido por este item é o passo central de um algoritmo *on-line* que constrói o diagrama de Delaunay de um conjunto de n sítios em tempo $O(n^2)$.]

11. (Exercício 5.4.5.3 de [15] — Diagrama de Voronoi cinético) Imagine um conjunto de pontos movendo-se no plano, cada ponto com uma direção e velocidade fixas. Seja $V(t)$ o diagrama de voronoi deste pontos no instante t. É um problema em aberto obter um limite *tight* para o número de diagrmas combinatorialmente distintos que podemos obter ao longo do tempo. Tente obter a conhecida cota inferior de $\Omega(n^2)$. Em outra palavras, encontre um conjunto de n pontos tal que $V(t)$ muda a sua estrutura combinatória cn^2 vezes onde c é um constante.

    Ninguém foi capaz até agora de encontrar um exemplo onde mais de n^2 mudanças são necessárias, mas o melhor limite superior conhecido é $O(n^3)$ (cf. Fu e Lee [8], veja também Guibas, Mitchell e Roos [9]).

Referências

[1] T.H. Cormen, C.E. Leiserson e R.L. Rivest, *Introduction to algorithms*, The MIT Electrical Engineering and Computer Scienece Series, The MIT Press, MacGraw-Hill Book Company, 1990, QA758 C811i.

[2] M. de Berg, M. van Kreveld, M. Overmars e O. Schwarzkopf, *Computational geometry: Algorithms and applications*, Springer, 1997, Second Edition, 2000.

[3] P.J. de Resende e J. Stolfi, *Fundamentos de geometria computacional*, IX Escola de Computação, 1994.

[4] B. Delaunay, Sur la sphère vide, *Bull. Acad. Sci. USSR(VII)* (1934), 793–800, Classe Sci. Mat. Nat.

[5] H. Edelsbrunner, *Algorithms in combinatorial geometry*, EATCS Monographs on Theoretical Computer Science, no. 10, Springer-Verlag, Berlin, 1987, QA758 E21a.

[6] L.H. Figueiredo e P.C.P. Carvalho, *Introdução à geometria computacional*, 18º Colóquio Brasileiro de Matemática, IMPA, 1991, QA758 F475i.

[7] S. Fortune, A sweepline algorithm for Voronoi diagrams, *Algorithmica* **2** (1987), 153–174.

[8] J.-J. Fu e R.C.T. Lee, Voronoi diagrams of moving points in the plane, *International Journal on Computational Geometry and Applications* **1** (1991), no. 1, 23–32.

[9] L.J. Guibas, J.S.B. Mitchell e T. Roos, Voronoi diagrams of moving points in the plane, *Proceedings of the 17th International Workshop Graph-Theoretic Concepts in Computer Science*, Lecture Notes Computer Science, vol. 570, Springer-Verlag, 1991, pp. 113–125.

[10] L.J. Guibas e J. Stolfi, Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams, *ACM Transactions on Graphics* **4** (1985), no. 2, 74–123.

[11] L.J. Guibas e J. Stolfi, Ruler, compass and computer: The design and analysis of geometric algorithms, *Theoretical Foundations of Computer Graphics and CAD* (R.A. Earnshaw, ed.), NATO ASI Series, vol. F40, Springer-Verlag, 1988, pp. 111–165.

[12] D.E. Knuth, *The art of computer programming. Volume I: Fundamental algorithms*, Addison-Wesley, Reading, MA, 1968.

[13] M.J. Laszlo, *Computational geometry and computer graphics in C++*, Prentice Hall, Upper Saddle River, NJ, 1996.

[14] D.T. Lee e B.J. Schachter, Two algorithms for constructing the Delaunay triangulation, *Int. J. Comput. Inf. Sci.* **9** (1980), no. 3, 219–242.

[15] J. O'Rourke, *Computational geometry in C*, Cambridge University Press, Cambridge, 1993, Second Edition, 1998.

[16] F.P. Preparata e M.I. Shamos, *Computational geometry: An introduction*, Texts and Monographs in Computer Science, Springer-Verlag, New York, 1985, QA758 P927c.

[17] M.I. Shamos e D. Hoey, Closest point problems, *Proc. 16th Annual IEEE Symposium in Foundations of Computer Science*, 1975, pp. 151–162.

[18] M.I. Shamos e D. Hoey, Geometric intersection problems, *Proceedings of the 17th Annual IEEE Symposium on Foundations of Computer Science*, 1976, pp. 208–215.