5. O método do embrulho para presente (gift wrapping)

O algoritmo que veremos agora também é conhecido como algoritmo de Jarvis (cf. [9], veja também a Seção 3 do Capítulo 35 de [3] e a Seção 3 do Capítulo 3 de [6]). O algoritmo pode ser visto como uma versão 2-dimensional do método sugerido por Chand & Kapur [2] para construir o fecho convexo de um conjunto de pontos em dimensões arbitrárias. A complexidade de tempo do algoritmo é $O(nh)$, onde h é o número de arestas (ou vértices) de $\text{conv}(S)$, i.e. o algoritmo é *output-sensitive*.

Intuitivamente o algoritmo simula o enrolar do conjunto S por um barbante e pode também ser visto como uma melhoria do Algoritmo `Arestas-Extremas`: a idéia é usar a aresta extrema que acabou de ser encontrada para encontrar a próxima aresta. O método funciona, pois sabemos que no fecho convexo uma aresta (extrema) esta ligada a outra, que por sua vez esta ligada a uma outra aresta e assim por diante. Para cada aresta extrema encontrada testaremos $O(n)$ candidatas a próxima aresta extrema, logo a complexidade total do algoritmo será $O(nh) = O(n^2)$, que reduz por um fator de n a complexidade do Algoritmo `Arestas-Extremas`.

Para facilitar a exposição vamos assumir que os pontos de S estão em posição geral: não existem três pontos em S que sejam colineares. Seja $e = xy$ uma aresta extrema de $\text{conv}(S)$ (assuma y tem o valor de sua abscissa maior do que o valor da abscissa de x; y está a direita de x em relação à reta vertical). Sabemos que $\text{conv}(S)$ possui uma outra aresta com uma extremidade em y. Para todo ponto $z \in S$ nos testamos se todos os pontos de S estão de um mesmo lado em relação à reta que passa pelo segmento yz (candidata a aresta de $\text{conv}(S)$): este teste pode ser feito em tempo $O(n)$—o que continuaria a nos deixar com um algoritmo de complexidade de tempo $O(n^3)$. A observação chave em relação a complexidade de tempo é escolher um ponto z cujo ângulo polar θ_z com relação ao ponto y e a reta passando pelo segmento xy seja mínimo. Isto garante que todos os pontos de S estão de um mesmo lado da reta passando por yz (veja Figura 3) e portanto não precisamos fazer uma verificação para cada ponto de S.

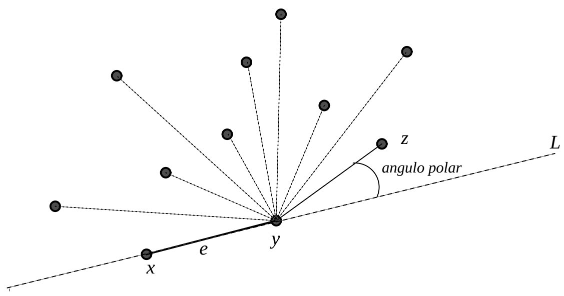

FIGURA 3. A próxima aresta yz tem o menor ângulo polar com relação à aresta anterior e.

Como começar o algoritmo já que no início não temos nenhuma aresta de $\text{conv}(S)$? Basta começarmos com (digamos) um vértice p_o mais baixo em S (menor y-coordenada) e encontramos o ponto p_1 em S com o menor ângulo polar (sentido anti-horário) com relação ao ponto p_0 e a reta horizontal (veja Figura 4).

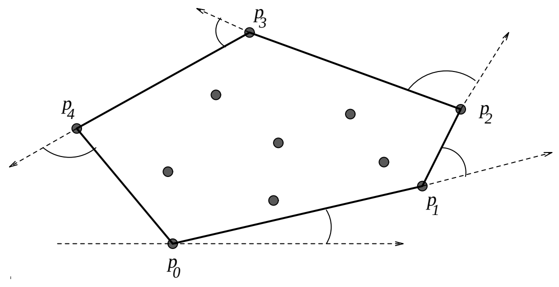

FIGURA 4. Operações feitas pelo algoritmo de Jarvis. O primeiro ponto é um ponto mais baixo. O próximo ponto tem o menor ângulo polar em relação ao ponto p_0 e a reta horizontal.

Algoritmo `Embrulho-Para-Presente`
`Entrada:` Um conjunto finito $S = \{p_1, \dots, p_n\}$ de pontos no plano.
`Saída:` As arestas extremas de S a medida que elas aparecem
na fronteira de conv(S) no sentido anti-horário.
Encontre o ponto mais baixo de S. [custa tempo $O(n)$]
Seja i_0 o índice desse ponto e $i \leftarrow i_0$.
Seja L a reta horizontal passando por p_{i_0}.
repeat
  for cada $j, j \neq i$ **do**
    Calcule o ângulo polar θ_j do ponto p_j em relação ao
    ponto p_i e a reta L.
  Seja k o índice do ponto p_k tal que $\theta_k = \min_j \theta_j$.
  Output $p_i p_k$ como uma aresta de conv(S).
  $i \leftarrow k$.
  $L \leftarrow$ reta passado por $p_i p_k$.
until $i = i_0$. [Fechou o polígono.]

Se h é o número de arestas extremas de conv(S) então a complexidade de tempo do algoritmo de Jarvis no pior caso é $O(hn) = O(n^2)$: um custo de $O(n)$ por aresta encontrada. Note que quando o número de arestas extremas de conv(S) é pequeno, digamos $h = O(\log n)$, então o algoritmo de Jarvis é tão rápido quanto o algoritmo ótimo de Graham que veremos logo adiante (ótimo em termos de complexidade de tempo).