

ABORDAGENS ALGORITMÍMICAS PARA O PROBLEMA DO FECHO CONVEXO NO PLANO. Renato de Jesus Manzoní, Marco Antônio Piteri. – Exatas - Ciência da Computação – Departamento de Matemática, Estatística e Computação – Faculdade de Ciências e Tecnologia – Campus de Presidente Prudente.

O problema de se encontrar o *fecho convexo* de um conjunto de n pontos no \mathbf{R}^n possui inúmeros interesses teóricos. Do ponto de vista prático, as aplicações mais imediatas estão associadas às dimensões 2 e 3, como por exemplo: *detecção de colisão em movimentos de robôs, área/volume mínimo necessário para envolver um objeto 2D/3D e análise de formas*, para citar algumas.

Um outro aspecto interessante relacionado a este problema é a sua utilização na obtenção da solução de outros problemas de natureza geométrica, ou seja, situações em que o *fecho convexo* aparece como um subproblema a ser resolvido. Um exemplo clássico é a obtenção do *diâmetro* de um conjunto de n pontos em \mathbf{R}^n .

Por outro lado, sua estrutura geométrica está diretamente relacionada a outras duas importantes subdivisões planares associadas a um conjunto de pontos arbitrários, mais especificamente, a *Triângulação de Delaunay* e o *diagrama de Voronoi*. Em alguns contextos mais específicos, esta última também é referenciada na literatura por *tecagem de Thiessen*.

Estas diferentes características, aliado ao fato deste problema ter sido um dos primeiros a ser estudado pela comunidade ligado à área de *Geometria Computacional*, fez com que ele recebesse uma atenção especial, o que acabou por refletir numa enorme variedade de abordagens algorítmicas existentes para sua solução no plano, nem todas passíveis de serem generalizadas para outras dimensões. Vale salientar que este trabalho está sendo desenvolvido no contexto de um projeto de iniciação científica suportado pela FAPESP e que objetiva discutir os múltiplos aspectos relativos aos mais variados paradigmas computacionais, entre eles, os algoritmos de *Graham*, *Andrew*, *Jarvis*, *Kirkpatrick* e *Seidel Chan*, *Inserção*, *MergeHull*, e *QuickHull*.

Devemos realçar ainda que há uma estreita relação do problema de *fecho convexo* com o problema de *ordenação*. Nesse sentido, os principais métodos de ordenação possuem seus homólogos algoritmos aplicados ao problema do fecho convexo. Na verdade, o limite inferior de complexidade de pior caso dos algoritmos de fecho convexo é condicionado por aqueles associadas aos melhores algoritmos de ordenação, ou seja, $\Omega(n \log n)$.

Embora muitas das abordagens listadas acima já tenham sido implementadas, neste trabalho especificamente objetivamos discutir em maior profundidade os algoritmos de Jarvis e o de Inserção, aplicados a conjunto de pontos no plano.

Existem diversas maneiras de se definir o fecho convexo de um conjunto S com n pontos no plano. Uma primeira definição formal é considerar o fecho convexo de S , denota-se $Conv(S)$, como o menor conjunto convexo K que contém todos os pontos de S . É possível observar na Figura 1c que este conjunto é dado por um polígono. Da observação da Figura 1a podemos ainda definir $Conv(S)$ como a intersecção de todos os conjuntos convexos que contém S . Esta segunda definição pode ser formalizada matematicamente pela equação (1).

$$Conv(S) = \bigcap_{K \supseteq S} K \quad (1)$$

Poderíamos definir ainda $Conv(S)$ como sendo o conjunto de todas as combinações lineares convexas dos pontos de S . A equação (2) captura a essência desta definição.

$$Conv(S) = \left\{ \sum_{i=1}^n \lambda_i p_i, \lambda_i \in \mathbb{R}, \lambda_i \geq 0, p_i \in S; \sum_{i=1}^n \lambda_i = 1 \right\} \quad (2)$$

As definições dadas acima embora consigam precisar matematicamente o conceito do fecho convexo, elas não oferecem nenhuma pista de como podemos usá-las para a elaboração de processos algorítmicos que consigam efetivamente encontrar o fecho convexo. Precisamos de uma caracterização de $Conv(S)$ discreta, finita e que seja passível de ser representada num computador.

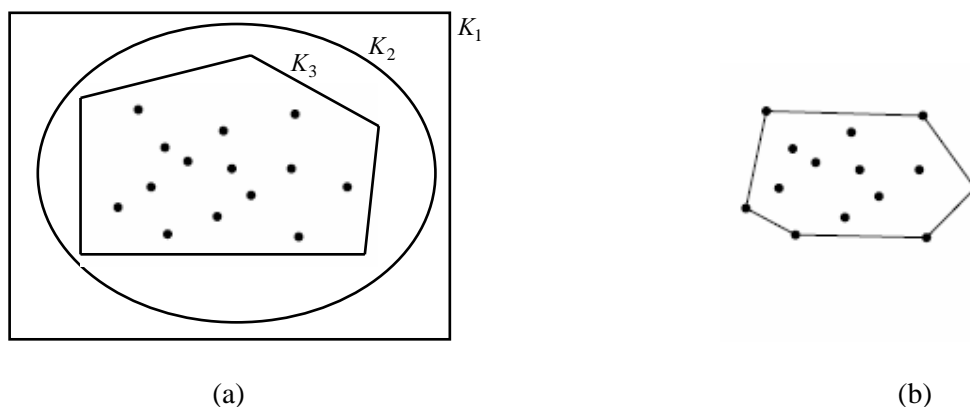


Figura 1: (a) Conjunto S de pontos no plano e diferentes conjuntos C_i , ($1 \leq i \leq 3$) que o contém; (b) $Conv(S)$ dado por um polígono.

A partir da Figura 1 observamos que $Conv(S)$ é um polígono convexo, cujos vértices pertencem ao conjunto dado. Logo, é suficiente identificar estes vértices e organizá-los numa lista, por exemplo, no sentido anti-horário. Sabemos que em qualquer polígono convexo a reta suporte passando por quaisquer de seus lados (segmentos orientados) deixa todos os demais pontos num mesmo lado da reta. Entretanto, o número total de segmentos é dado por $\binom{n}{2}$. Além disso, para cada um dos segmentos teríamos que classificar $n-2$ pontos para verificar a que lados eles pertencem em relação ao segmento orientado. Um possível algoritmo explorando estas idéias seria muito ineficiente, já que seria de ordem cúbica.

O conceito de ponto extremo pode nos ajudar. Dizemos que $p \in S$ é um *ponto extremo* de S , se não existir dois pontos $a, b \in S$, tal que p esteja no interior do segmento \overline{ab} . Em outras palavras, quando p for escrito somente por meio de uma combinação convexa trivial de pontos de S . Quando $n = 2$ que $p \in S$ é um *ponto extremo* se e somente se ele não está no interior do segmento de reta que une este dois pontos. Além disso, podemos provar que os *pontos extremos* de $Conv(S) \in S$ e se localizam na fronteira de $Conv(S)$. Logo, para resolvermos computacionalmente o problema de fecho convexo de um conjunto finito de pontos S , é suficiente identificar os *pontos extremos* de $Conv(S)$ dentre os pontos de S .

Antes de iniciar a descrição do princípio computacional associado ao algoritmo de *Jarvis*, vamos apelar um pouco mais para o nosso senso comum. Se alguém lhe entregasse um lápis, uma régua e uma folha de papel com um conjunto de pontos S desenhados e lhe pedisse para encontrar o fecho convexo, muito provavelmente os passos construtivos que você usaria seriam muito próximos da técnica elaborada por *Jarvis*.

Para simplificar a descrição da técnica, vamos admitir que S não tenha pontos coincidentes e que não haja três pontos colineares. Inicialmente devemos escolher um ponto p_i , ($1 \leq i \leq n$); $p_i \in Conv(S)$. Isso pode ser facilmente obtido se considerarmos o ponto de menor ordenada em S . Vamos referenciar este ponto por p . Agora, vamos imaginar uma reta L paralela ao eixo- x , fixa e passando por p . Ao rotacionarmos L no sentido anti-horário ela irá interceptar o ponto q . Observem que os pontos $p, q \in Conv(S)$. O próximo passo consiste em fixar L no ponto q e repetir a rotação desta no sentido anti-horário até encontrar o próximo ponto, que no caso da Figura 2a é s . Acabamos de encontrar mais um *ponto extremo* de $Conv(S) \subseteq S$. A repetição sistemática deste processo conduz a completa obtenção de todos os pontos de $Conv(S)$, de acordo com a Figura 2b. Podemos observar ainda que quando L está fixa sobre um ponto, por exemplo p , e, procuramos pelo próximo, o ponto escolhido será aquele associado ao ângulo mínimo em relação à reta L e as semi-retas iniciando em p e passando pelos demais pontos

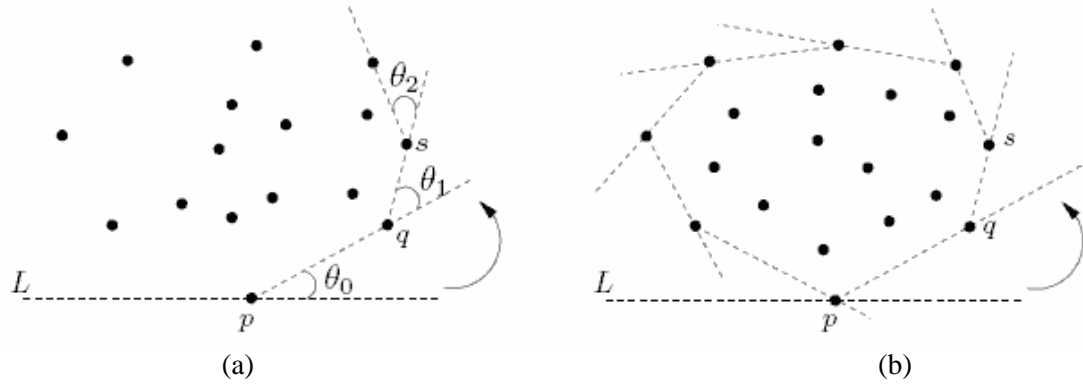


Figura 2: Ilustração da técnica *Jarvis*. (a) Conjunto de pontos S , ponto extremo p e reta L . O ponto q é escolhido, porque o segmento definido pelos pontos p e q possui o menor ângulo θ com a reta L ; (b) obtenção do $Conv(S)$ após a repetição sistemática do princípio de *Jarvis*.

O algoritmo de *Jarvis* possui uma característica importante, ele é sensível à saída, ou seja, o tempo de execução do algoritmo depende do número de vértices h pertencentes à $Conv(S)$. Na verdade, este algoritmo possui complexidade $O(n.h)$. Se $h \cong n$, então teremos um tempo de execução quadrático. Entretanto, na prática este algoritmo pode ser considerado eficiente.

A próxima abordagem a ser descrita mantém permanentemente a fronteira do fecho convexo atualizada e se baseia num paradigma muito usado na área de projeto e análise de algoritmos que é denominado inserção, daí o seu nome, algoritmo de *Inserção*. Este algoritmo inclusive pode ser facilmente generalizado para outras dimensões.

A idéia básica é que se sabemos construir $Conv(S)$ onde $|S| = 3$ (cardinalidade do conjunto), podemos construir sistematicamente para $|S| = 4$, $|S| = 5$ e assim sistematicamente, até $|S| = n$.

Para facilitar a discussão, vamos imaginar que já construímos $Conv(S)$ para $|S| = k$. A cada novo ponto p_{k+1} a ser analisado, pode ocorrer duas situações: ou $p_{k+1} \in Conv(S)$ e neste caso não há nada a fazer, já que $Conv(S)$ está atualizado; ou, situa-se no exterior do fecho convexo, ver Figura 3a. Neste caso, precisamos atualizar a fronteira de $Conv(S)$ acrescentando o ponto $s = p_{k+1}$.

Esta tarefa pode ser realizada encontrando-se inicialmente os pontos p e q . Observem na Figura 3a que estes pontos dividem a seqüência de vértices de $Conv(S)$ em duas cadeias, aqueles que estão do mesmo lado de s em relação ao segmento pq e os que estão do lado oposto de s em relação ao mesmo segmento. Podemos observar ainda que $Conv(S) \cup s$ introduz dois novos segmentos ao polígono associado à $Conv(S)$ que são \overline{sp} e \overline{sq} . A questão fundamental agora é remover os vértices que deixam de pertencer a fronteira do fecho convexo, que são exatamente os da cadeia que estão do mesmo lado de s em relação ao segmento pq , com exceção dos extremos. O fecho convexo atualizado agora para $k+1$ pontos pode ser observado na Figura 3b.

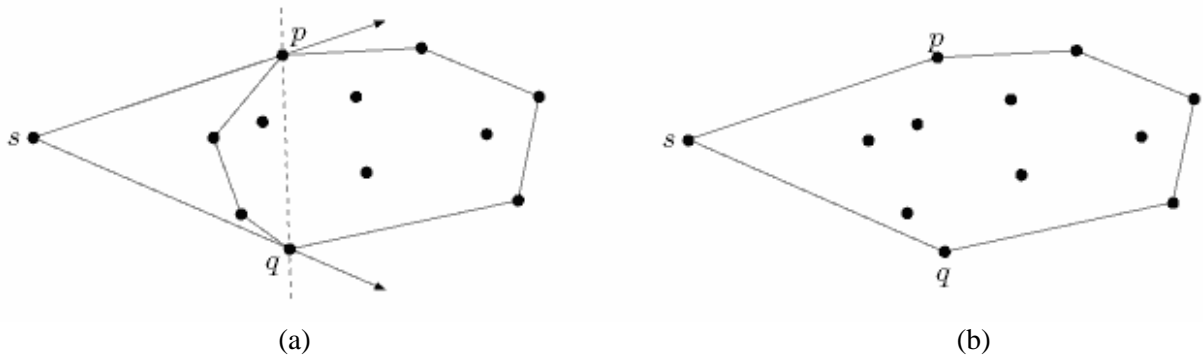


Figura 3: (a) Acrescentando um novo ponto s ao $Conv(S)$ onde encontramos os pontos p e q que dão origem aos segmentos tangentes a $Conv(S)$; (b) $Conv(S)$ já atualizado com o ponto s .

Até o presente momento já realizamos os estudos teóricos dos algoritmos de *Graham*, *Andrew Jarvis*, *QuickHull*, *Inserção* e *MergeHull*. No que se refere à implementação, todos eles foram implementados com sucesso e neste momento estamos nos concentrando em aumentar a consistência e robustez do sistema, tratando os casos singulares relativos a cada um dos algoritmos.

As implementações estão sendo realizadas usando-se a ferramenta Dev-C++, que é um ambiente integrado de desenvolvimento de 32 bits. A parte gráfica relativa à visualização de todos os aspectos geométricos do problema, incluindo a interface está sendo elaborada pela biblioteca multiplataforma *allegro*.

O conjunto de pontos para nossos testes é obtido a partir da geração de números randômicos no interior de uma área retangular. Com o intuito de simular ao máximo a existência de casos singulares (colinearidade), os pontos são gerados também sobre linhas poligonais e sobre os segmentos que constituem a fronteira de uma região retangular.

Este trabalho está inserido num contexto mais amplo de um projeto de iniciação científica financiado pela **FAPESP**, a quem agradecemos o apoio. Além dos tópicos aqui apresentados, existem outras componentes integrantes do projeto e que consistem em estudar os aspectos teóricos e práticos de algoritmos para encontrar o *fecho convexo* de *linhas poligonais* e *polígonos simples* arbitrários, bem como algoritmos aproximados para este mesmo problema.

Em consonância com os estudos que estão sendo realizados e em continuidade ao projeto, pretendemos generalizar os estudos para o espaço euclidiano de dimensão 3.

Uma outra questão que não pode ficar a margem de qualquer projeto versando sobre a temática de algoritmos geométricos são aquelas relacionadas à aritmética de ponto flutuante. Problemas geométricos são extremamente susceptíveis a aritmética finita, em particular o problema de fecho convexo, que utiliza sobremaneira de um predicado geométrico de classificação de pontos em relação a segmentos orientados, que é, na verdade, obtido a partir do cálculo do sinal de um determinante de ordem 3 (coordenadas absolutas), ou de maneira análoga e com maior precisão por um de ordem 2 (coordenadas relativas).

Considerando ainda que a correção e a robustez de algoritmos para o problema de fecho convexo são fortemente dependentes do predicado citado, pretendemos fazer uso de alguma abordagem de aritmética exata entre as inúmeras existentes e utilizadas pela comunidade.

Referências Bibliográficas

AVIS, D.; BREMNER, D.; SEIDEL, R. How good are convex hull algorithms? **Computer Geometry Theory and Applications**, v. 7, n. 5, p. 265-305, abr. 1997.

BERG, M.; KREVELD, M. V.; OVERMARS, M.; SCHWARZKOPF, O. **Computational geometry: algorithms and applications**. Berlin: Springer-Verlag, 1997. 365p.

BREMNER, D. Incremental convex hull algorithms are not output sensitive. **Discrete Computational Geometry**, v. 21, n.1, p. 57-68, jan. 1999.

GREEN, P. J.; SILVERMAN, B. W. Construction the convex hull of a set of points in the plane. **Computer Journal**, v. 22, n. 3, p. 262-266, 1979.

JARVIS, R. A. On the identification of the convex hull of a finite set of points in the plane, *Information Processing Letters*, v. 2, n. 1, p. 18-21, mar. 1973.

O'ROURKE, J. **Computational geometry in C**. 2 ed. Cambridge: Cambridge University Press, 1994. 376p.

Bolsa: FAPESP