Projeto e Análise da Varredura de Graham

Khalil Carsten do Nascimento
Departamento de Ciência da Computação
Universidade de Brasília
Brasília, Brasil
khalilcarsten@gmail.com

Renato Avelar Nobre
Departamento de Ciência da Computação
Universidade de Brasília
Brasília, Brasil
rekanobre@gmail.com

***Palavras-Chave*—algoritmo, varredura de Graham, geometria computacional, envoltória convexa, análise assintótica, complexidade de tempo, complexidade de espaço, correção, polígonos**

***Resumo*—Imagine o problema de saber se um veículo autônomo tem chance de colidir com alguma entidade do meio em que anda. Esse problema pode ser resolvido utilizando o algoritmo de Varredura de Graham encontrando a envoltória convexa do veículo. Esse trabalho apresenta o algoritmo e realiza uma análise da sua correção, utilizando a premissa de invariância da sua estrutura de repetição. Realiza-se também uma análise da complexidade temporal que foi observada como $O(n \log n)$, e espacial, que utiliza um total de $O(n)$ em seu pior caso. Comparamos o algoritmo de Graham com outros encontrados na literatura e percebemos que o mesmo é assintoticamente ótimo quando não se possui nenhuma informação a respeito da envoltória.**

I. Introdução

Veículos autônomos vem sendo um tópico crescente na literatura nos recentes anos. Imagine o problema de saber se alguma parte do veículo tem chance de colidir com alguma entidade do meio. Esse problema pode ser abordado encontrando a envoltória convexa do veículo.

Dizemos que a envoltória convexa é o menor polígono convexo que contém todos os pontos contidos no plano. Podemos visualizar a envoltória como um elástico esticado em volta dos pontos, que quando solto, cria-se um formato poligonal.

Existem diversos algoritmos capazes de calcular a envoltória para um dado conjunto de pontos, nesse artigo, descrevemos a Varredura Graham, mostramos que o algoritmo é correto, e fazemos uma análise de sua complexidade temporal e espacial. Por fim, é realizada uma comparação com outros algoritmos publicados na área.

Este artigo esta dividido da seguinte forma: A Seção II apresenta o problema da envoltória convexa e suas aplicações; a Seção III explica o algoritmo de Graham para exame de envoltórias convexas e realiza a devida análise de completude, corretude e complexidade de tempo e espaço; após a análise a Seção IV compara a eficiência da Varredura de Graham com outros algoritmos de envoltórias convexas; por ultimo, a Seção V conclui o trabalho.

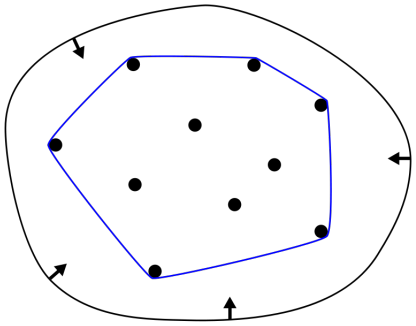

Figura 1. Envoltória Convexa de Pontos Q

II. Envoltória Convexa

Na geometria um polígono P é dito convexo se para qualquer dois pontos p_a e p_b internos à ele, o segmento de reta que passa pelos dois pontos se situa dentro do mesmo.

No caso de um plano cartesiano, contendo mais de três pontos não colineares entre si, haveria então a possibilidade de encontrar diversos polígonos. No entanto, as vezes precisa-se ter conhecimento apenas dos pontos mais externos no plano, apresenta-se então o *Problema da Envoltória Convexa*.

Seja um conjunto de pontos Q, dizemos que a envoltória convexa $EC(Q)$ é o menor polígono convexo que contém todos os pontos p_i de Q. No plano podemos visualizar a envoltória como um elástico esticado em volta dos pontos, que quando solto, cria-se um formato poligonal [3], como representado pela Figura 1. Os pontos extremos do conjunto Q representarão o polígono convexo.

A importância da análise das envoltórias convexas, se veem crescente com a necessidade de reconhecimento de padrões, como na aplicação para preprocessamento de numerais na aplicação à modelos de aprendizagem [5]. Outra aplicação crescente é em movimentação de veículos autônomos e precaução de colisões, devido ao fato de que se o menor polígono convexo, capaz de representar o carro, não colide com outros pontos no plano, logo o carro também não colidirá.

III. Varredura de Graham

O problema da envoltória convexa pode ser resolvido com o algoritmo de Varredura de Graham, que utiliza uma pilha de pontos do plano cartesiano. A varredura insere cada ponto do

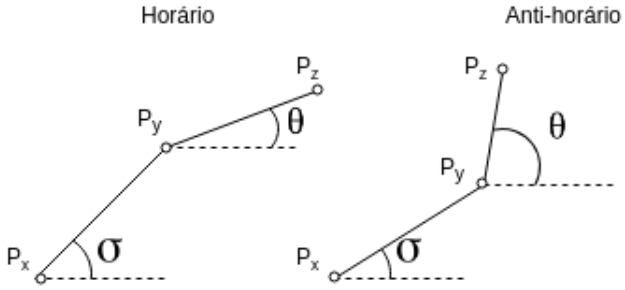

Figura 2. Visualização das rotações horária e anti-horária

conjunto de entrada Q na pilha uma vez e, durante a execução do algoritmo, os pontos que não pertencem a envoltória vão sendo desempilhados.

Tendo isso em vista a nossa entrada para o algoritmo é um conjunto Q de pontos, onde $|Q| \geq 3$, e temos como saída uma pilha S, contendo todos os pontos, em ordem anti-horária que formam a envoltória convexa.

O algoritmo se inicia com a escolha de um ponto p_0 onde este ponto deve ser o ponto mais em baixo e mais a esquerda possível [1]. Agora usando p_0 como o ponto referência faremos uma ordenação do conjunto Q utilizando como métrica as coordenadas polares de cada ponto. Aqui nota-se a necessidade de escolher um ponto como referência, tendo em vista que para calcular uma coordenada polar é necessário dois pontos do espaço euclidiano. Podemos converter uma coordenada cartesiana para uma coordenada polar utilizando a função $atan2(x, y)$ definida em [3]. Caso dois pontos assumam a mesma coordenada polar aquele que tem a menor distância para o ponto de referência é eliminado.

Agora em uma pilha vazia S empilharemos os pontos p_0, p_1 e p_2 de Q. E faremos uma iteração para cada ponto em $\langle p_3, p_4, \ldots, p_m \rangle$, onde m é a quantidade de pontos na lista ordenada. Para cada interação o algoritmo checa se os dois últimos pontos de S e o ponto p_i, o atual ponto da iteração, executam uma rotação horária ou anti-horária. Enquanto for encontrado rotações horárias o algoritmo desempilha o ponto do topo de S e faz a comparação de rotação novamente. Caso uma anti-horária seja encontrada o ponto p_i é empilhado.

Para o teste da direção da rotação utiliza-se as equações 1 e 2. Caso $\theta > \sigma$ significa que temos uma rotação anti-horária e $\theta < \sigma$ uma rotação horária.

Ao final das iterações restará em S somente os pontos que formam a envoltória convexa.

$$\sigma = \frac{(y_2 - y_1)}{(x_2 - x_1)} \tag{1}$$

$$\theta = \frac{(y_3 - y_1)}{(x_3 - x_1)} \tag{2}$$

$$atan2(x, y) = \begin{cases} arctan(\frac{y}{x}), & \text{se } x > 0 \\ arctan(\frac{y}{x}) + \pi, & \text{se } x < 0 \text{ e } y \geq 0 \\ arctan(\frac{y}{x}) - \pi, & \text{se } x < 0 \text{ e } y < 0 \\ \frac{\pi}{2}, & \text{se } x = 0 \text{ e } y > 0 \\ -\frac{\pi}{2}, & \text{se } x = 0 \text{ e } y < 0 \\ \text{indefinido }, & \text{se } x = 0 \text{ e } y = 0 \end{cases} \tag{3}$$

Algoritmo 1: Varredura de Graham

```
1  p_0 ← ponto com a menor coordenada y, caso empate,
     escolha aquele com a menor coordenada x;
2  Ordene(Q, p_0);
3  S ← pilha();
4  S.empilha(p_0);
5  S.empilha(p_1);
6  se m ≥ 2 então
7  |  S.empilha(p_2);
8  fim
9  para i ← 3 até m faça
10 |  enquanto horario(S.penultimo(), S.topo(),
   |   p_i) faça
11 |  |  S.desempilha()
12 |  fim
13 |  S.empilha(p_i)
14 fim
15 retorna S
```

A. *Correção*

Um aspecto importante na hora de realizar a análise de um algoritmo projetado é executar a prova formal do mesmo. A prova formal é capaz de dizer se um algoritmo funciona da maneira adequada para todos os possíveis casos de seu escopo. Para o algoritmo da Varredura de Graham, o seguinte teorema define sua correção:

Teorema: Se a Varredura de Graham é executada em um conjunto Q de pontos, onde $|Q| \geq 3$, então, no termino, a pilha S consiste, de baixo para cima, exatamente nos vértices de $EC(Q)$ em ordem anti-horária [1].

Realizaremos a prova desse teorema, utilizando como base o Algoritmo 1, que representa o pseudocódigo do algoritmo de Graham.

*1) **Prova:**:* No Algoritmo 1 obtemos após a execução da linha 2, uma sequencia de pontos $\langle p_1, p_2, \ldots, p_m \rangle$. Para auxiliar na prova, vamos definir :

$$Q_i = \{p_0, p_1, \ldots, p_i\} \ \forall i = 2, 3, \ldots, m \tag{4}$$

Observe que os pontos em $Q - Q_m$ são aqueles que foram removidos porque tinham o mesmo ângulo polar em relação a p_0 que algum ponto em Q_m; esses pontos não estão em $EC(Q)$ e, portanto, $EC(Q_m) = EC(Q)$.

Portanto, precisamos provar apenas que, quando o Algoritmo 1 termina, a pilha S consiste nos vértices de $EC(Q_m)$

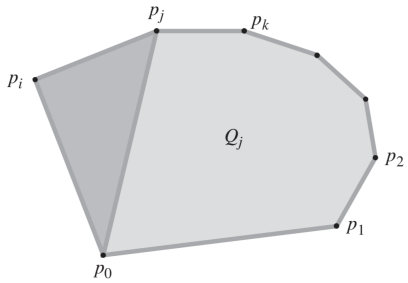

Figura 3. Representação da pilha S após empurrarmos p_i

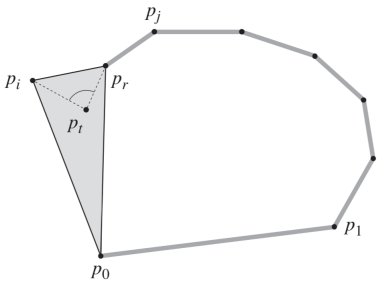

Figura 4. Representação de p_t no interior do triângulo, não sendo vértice de $EC(Q_i)$

em ordem anti-horária, quando apresentados de baixo para cima. Observe que, exatamente como p_0, p_1 e p_m são vértices de $EC(Q)$, os pontos p_0, p_1 e p_i são vértices de $EC(Q_i)$

Para provar utilizaremos o seguinte invariante de estrutura de repetição:

No início de cada iteração da estrutura de repetição das linhas 9-14, a pilha S consiste, de baixo para cima, exatamente nos vértices de $EC(Q_{i-1})$ em ordem anti-horária.

*2) **Inicialização**:* Observe que o invariante inicializa válido na primeira vez que executamos a linha 9, já que, nesse instante, a pilha S consiste exatamente nos vértices de $Q_2 = Q_{i-1}$, e esse conjunto de três vértices forma sua própria envoltória convexa. E completando, eles aparecem em ordem anti-horária de baixo pra cima.

*3) **Manutenção**:* Ao entrar em uma iteração da estrutura de repetição *para* na linha 9, o ponto que está no alto da pilha S é p_{i-1}, que foi inserido no final da iteração anterior (ou antes da primeira iteração, quando $i = 3$).

Seja p_j o ponto que está no alto da pilha S após a execução do laço *enquanto* nas linhas 10-12, mas antes de a linha 13 inserir p_i, e seja p_k o ponto imediatamente abaixo de p_j em S. No momento em que p_j é o ponto que está no alto da pilha S e ainda não empurramos p_i a pilha S contém exatamente os mesmos pontos que continha depois da iteração j do laço *para* da linha 9. Então, pelo invariante de laço S contém exatamente os vértices de $EC(Q_j)$ naquele momento e eles aparecem em ordem anti-horária de baixo para cima.

Entrando em detalhes do porque a *manutenção* se mantém verdadeira, focalizaremos no momento antes momento imediatamente antes de p_i ser colocado na pilha. Sabemos que o angulo polar de p_i em relação a p_0 é maior que o ângulo polar de p_j em e que o ângulo $\angle p_k p_j p_i$ vira no sentido anti-horário, pois caso o contrário teríamos extraído p_j. Portanto, como S contém exatamente os vértices de $EC(Q_j)$, vemos, que uma vez empurrado p_i, S passará a conter os vértices de $EC(Q_j \cup \{p_i\})$, de ordem anti-horária de baixo para cima, como representado na Figura 3.

Visto isso, precisamos mostrar agora que $EC(Q_j \cup \{p_i\})$ é o mesmo conjunto de pontos que $EC(Q_i)$. Considere qualquer ponto p_i que tenha sido extraído durante a iteração i do laço *para* da linha 9, e seja p_r, o ponto imediatamente abaixo de p_t na pilha S no momento em que p_t foi extraído (p_r podia ser p_j). O ângulo $p_r p_t p_i$ faz uma curva não para a esquerda e o ângulo polar de p_t em relação a p_0 é maior que o ângulo polar de p_r.

Perceba que o ponto p_t, deve se localizar no interior, ou em um dos lados, do triângulo formado por $\angle p_0 p_r, p_i$, como representado na Figura 4. E como p_t está dentro do triângulo formado por três outros pontos de Q_i, ele não pode ser um vértice $EC(Q_i)$. Portanto, visto que p_t não é vértice de $EC(Q_i)$ temos que:

$$EC(Q_i - \{p_t\}) = EC(Q_i) \tag{5}$$

Observe que a igualdade (5) se aplica a todos os pontos que foram extraídos durante a iteração de i da estrutura de repetição *para* da linha 9, denominaremos esses pontos de E_i. Podemos então aplicar a igualdade repetidamente para mostrar que $EC(Q_i - E_i) = EC(Q_i)$, porém, $Q_i - E_i = Q_j \cup \{p_i\}$ e, assim, concluímos que $EC(Q_j\{p_i\}) = EC(Q_i - E_i) = EC(Q_i)$.

Concluindo, mostramos que, uma vez que p_i é empurrado, a pilha S contém exatamente os vértices de $EC(Q_i)$ em ordem anti-horária de baixo para cima. Então, incrementar i tornará o invariante de estrutura de repetição válido para a próxima iteração.

*4) **Término**:* Quando a estrutura de repetição termina, temos $i = m + 1$ e, portanto, o invariante de laço implica que a pilha S consiste exatamente nos vértices de $EC(Q_m)$, que é $EC(Q)$, em ordem anti-horária de baixo para cima.

Portanto, concluímos assim a prova do teorema proposto como definição da correção, e assim, pode se dizer que o algoritmo é correto.

B. Análise Temporal

Tendo em vista o Algoritmo 1 faremos agora uma análise temporal de suas operações. A linha 1 pode ser vista como uma busca do menor valor em um vetor, podemos concluir isso com uma complexidade no pior caso de $O(n)$. A linha 2 se trata de uma ordenação, onde podemos utilizar algoritmos como Heap Sort e Merge Sort que possuem a complexidade de $O(n \log n)$. Observando a linha 9 temos um laço que será executado $n-3$ vezes. Na linha 10 temos um laço aninhado que é executado no máximo $m-2$ vezes. Cada ponto p_i é empilhado somente uma vez e depois de desempilhado nunca volta a pilha, podemos concluir que o laço da linha 10 é executado em um tempo

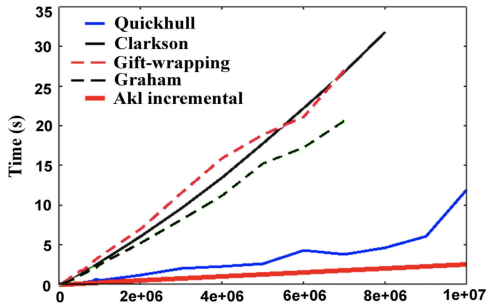

Figura 5. Comparação de Execução de Algoritmos

global de $O(n)$. Todas operações que envolvem operações com pilha são executadas em $O(1)$.

Somando as complexidades temos $O(n) + O(n \log n) + O(1) = O(n \log n)$ o que nos leva a concluir que o tempo de execução da varredura de Graham é $O(n \log n)$.

É importante notar que utilizar algoritmos de ordenação que possuem a complexidade maior que $O(n \log n)$ pode gerar uma complexidade de maior ordem. Isso descarta a opção de se utilizar algoritmos como *Quick Sort* que alcança complexidade de $O(n^2)$ no pior caso.

C. Análise Espacial

A varredura de Graham utiliza somente uma pilha como espaço adicional para desenvolver a lógica do algoritmo. Essa pilha no pior caso pode chegar ao tamanho da entrada o que nos leva a um complexidade espacial de $O(n)$.

IV. Comparações do Algoritmo de Graham

Assim como o Algoritmo de Graham, existem outras diversas abordagens na literatura para resolver o problema de envolturas convexas. Levaremos em conta algoritmos determinísticos de duas dimensões para realizar as comparações. Mas tenha em mente que existe algoritmos estocásticos e de aproximações, e algoritmos para mais de dois planos.

Alguns desses algoritmos para o plano cartesiano, são: *Algoritmo de Jarvis*, *Algoritmos de Dividir e Conquistar*, *Quickhull* e *Kirkpatrick e Seidel*.

O algoritmo de Jarvis [8], também chamado de algoritmo de embrulhar pacotes é capaz de encontrar a envoltória convexa em $O(nh)$ onde h é o número de vértices da envoltória. Quando o h é conhecido, o algoritmo de poda e busca de Kirkpatrick e Seidel [9] com complexidade temporal $O(n \lg h)$ é assintoticamente ótimo. O algoritmo *Quickhull* é uma analogia ao algoritmo de *Quick Sort*, podendo ser rodado em $O(n \log n)$ [3]. A Figura 5 representa uma análise desses algoritmos, onde o eixo Y é o tempo de execução e o eixo X o número de pontos no conjunto. Observe que o Akl Incremental é uma heurística, que pode ser aplicada aos algoritmos para executar o algoritmo de forma mais rápida, esta análise é realizada em [6].

Interessante mencionar também que, usando um modelo de computação de árvore de decisão Yao [10] provou o limite inferior de $(n \lg n)$ para o tempo de execução de qualquer algoritmo de envoltórias convexas.

Tabela I
Análise de Complexidade Temporal de Pior Caso

Varredura de Graham	$O(n \log n)$
Algoritmo de Jarvis	$O(n^2)$
Quickhull	$O(n^2)$
Dividir e Conquistar	$O(n \log n)$
Kirkpatrick e Seidel	$O(n \lg h)$

Um último fato que devemos analisar é que existe na literatura um algoritmo antecessor ao Algoritmo de Graham, criado por Bass e Schubert [11], que sua interpretação correta é executada em $O(n \log n)$, precedendo Graham em 5 anos [7]. No entanto, devido ao algoritmo ser descrito de forma vaga e incorreta, e não ter sido provado seu tempo de execução e correção, o algoritmo foi esquecido e não creditado.

V. Conclusão

Apresentou-se o algoritmo de Varredura de Graham como uma possível solução para o problema de envoltura convexa. Realizou-se então uma análise extensiva de sua correção que levou em consideração a invariância de estrutura de repetição para provar o teorema proposto como descritor do funcionamento do algoritmo.

Após a análise do funcionamento foi feita uma análise de sua complexidade para visualizar a viabilidade aplicação do algoritmo. Obtemos que os valor de complexidade de tempo é de $O(n \log n)$, definido por seu algoritmo de ordenação, e da complexidade de espaço $O(n)$.

Finalizamos a análise realizando uma breve comparação entre os outros algoritmos existentes na literatura, e percebemos que o algoritmo de Graham é assintoticamente ótimo na sua complexidade temporal quando não temos nenhuma informação a respeito da envoltória.

Referências

[1] Cormen, Thomas H., et al. Introduction to algorithms. MIT press, 2009.
[2] Halim, Steven, et al. Competitive Programming 3. Lulu Independent Publish, 2013.
[3] Bayer, Valentina. "Survey of algorithms for the convex hull problem." preprint (1999).
[4] Graham, Ronald L. "An efficient algorithm for determining the convex hull of a finite planar set." Info. Pro. Lett. 1 (1972): 132-133.
[5] Das, Nibaran, et al. "Design of a novel convex hull based feature set for recognition of isolated handwritten Roman numerals." arXiv preprint arXiv:1501.05494 (2015).
[6] Souviron, Jean. "Convex hull: Incremental variations on the Akl-Toussaint heuristics Simple, optimal and space-saving convex hull algorithms." CoRR abs/1304.2676 (2013): n. pag.
[7] Toussaint, Godfried T. "A historical note on convex hull finding algorithms." Pattern Recognition Letters 3.1 (1985): 21-28.
[8] Jarvis, Ray A. "On the identification of the convex hull of a finite set of points in the plane." Information processing letters 2.1 (1973): 18-21.
[9] Kirkpatrick, David G., and Raimund Seidel. "The ultimate planar convex hull algorithm?." SIAM journal on computing 15.1 (1986): 287-299.
[10] Yao, Andrew Chi-Chih. "A lower bound to finding convex hulls." STANFORD UNIV., COMPUTER SCIENCE Department, SCHOOL OF HUMANITIES AND SCIENCES, 1979.
[11] Bass, L. J., and S. R. Schubert. "On finding the disc of minimum radius containing a given set of points." Mathematics of Computation 21.100 (1967): 712-714.