5 . Invólucros Convexos no Plano (cont...)

Antonio Leslie Bajuelos
Departamento de Matemática
Universidade de Aveiro

Mestrado em Matemática e Aplicações

Complexidade Algorítmica

Notação O

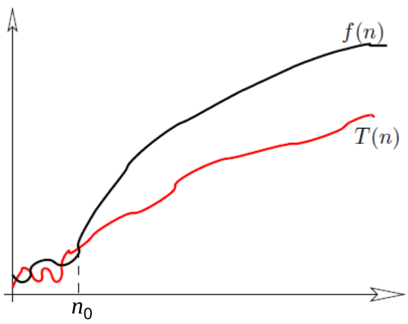
- Sejam T(n) e f(n) funções de $\aleph \rightarrow \Re$.

  T(n) é O(f(n)) se existem constantes positivas c e n_0 tq

  $$T(n) \leq c\, f(n)$$

  para todo $n \geq n_0$.

  i.e. para n grande, T(n) não cresce mais rápido que f(n)



Complexidade Algorítmica

Notação O

- Usada para análise de pior caso. Também chamado **limite superior**
- "T(n) é O(f(n))" deve ser entendido como "T(n) ∈ (f(n))".
- "T(n) = O(f(n))" deve ser entendido como "T(n) ∈ (f(n))".
- "T(n) é g(n) + O(f(n))" significa que existem constantes positivas c e n_0 tais que

  $T(n) \leq g(n) + c\, f(n);$ para todo $n \geq n_0$

3

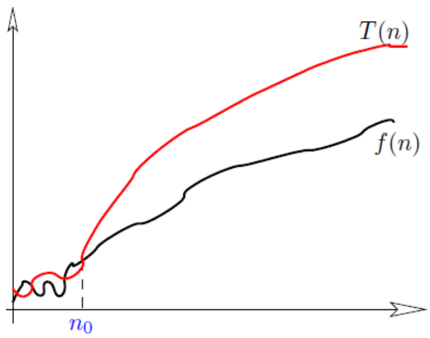
Complexidade Algorítmica

Notação Ω

- Dizemos que T(n) é Ω(f(n)) se existem constantes positivas c e n_0 t.q.

  $T(n) \geq c\, f(n)$

  para todo $n \geq n_0$



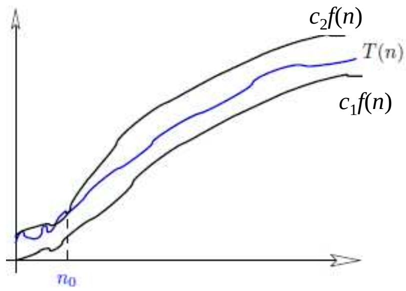
4

Complexidade Algorítmica

Notação Θ

- Dizemos que $T(n)$ é $\Theta(f(n)) \Rightarrow$

$\Theta(f(n)) = \{ T(n), \exists\, c_1, c_2, n_0, \text{ tq } c_1 f(n) \leq T(n) \leq c_2 f(n), \forall\, n \geq n_0 \}$



Complexidade Algorítmica

Notação assimtótica é limite

$\lim_{n \to \infty} T(n)/f(n) = c \Leftrightarrow T(n) = \Theta(f(n))$

$\lim_{n \to \infty} T(n)/f(n) = 0 \Leftrightarrow f(n) = O(f(n))$

$\lim_{n \to \infty} T(n)/f(n) = \infty \Leftrightarrow f(n) = \Theta(f(n))$

- **Analogias:**

  Dados números reais a e b é possível fazer as analogias

  - $f(n) = \Theta(g(n)) \approx a = b$
  - $f(n) = O(g(n)) \approx a \leq b$
  - $f(n) = \Omega(g(n)) \approx a \geq b$

Complexidade Algorítmica

Exemplo nº1

- Mostrar que $\log_a n = \Theta(\log_b n)$ para qualquer a, b > 1 constante.

  Resp. Devemos mostrar que existem c, c′ t.q.

  i) $\log_a n \le c \log_b n$

  ii) $\log_a n \ge c' \log_b n$

  i) $\log_a n \le c \log_b n \Leftrightarrow \log_a n \le c(\log_a n / \log_a b)$

  Tomando $c = \log_a b$ temos o resultado desejado.

  ii) TI

Propriedade

$$\log_b n = \frac{\log_a n}{\log_a b}$$

Complexidade Algorítmica

Exemplo nº2

- Mostrar que $nO(n) = O(n^2)$

  Resp. "$nO(n) = O(n^2)$" significa "$nO(n) \subset O(n^2)$".
  Ou seja, se T(n) é O(n), então nT(n) é $O(n^2)$.
  De fato, se T(n) é O(n) então existem constantes, digamos $c=10$ e $n_0=10^{100}$, tais que

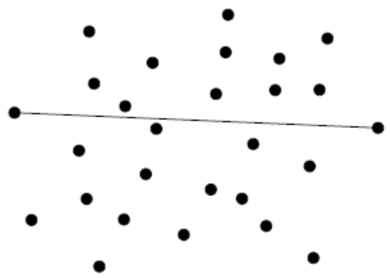
  $T(n) \le 10n$; para todo $n \ge 10^{100}$.

  Desta forma,
  $nT(n) \le n10n \le 10\, n^2$
  para todo $n \ge 10^{100}$. Logo nT(n) é $O(n^2)$.

Invólucro Convexo

Uma aplicação: diâmetro

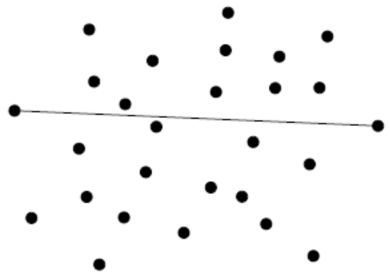
- *Suponhamos que são dados n pontos no plano e queremos encontrar a maior distancia determinada por dois deles.*



Invólucro Convexo

Uma aplicação

- **Solução ingénua**: para cada um dos pontos, se calcula a distancia entre ele e os restantes pontos. Ficamos com a maior dessas distancias. O custo desta solução é $O(n^2)$ já que o número de pares possíveis é $\binom{n}{2} = \frac{n(n-1)}{2}$



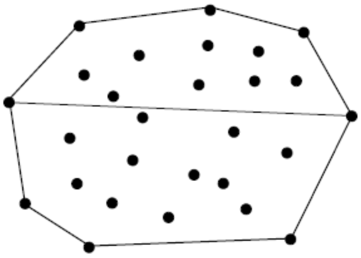
Invólucro Convexo

Uma aplicação: diâmetro

- **A solução tem um alto custo e também não utiliza a geometria do problema.**

  Uma proposta: Explorar o seguinte facto:

  a maior distancia entre pares de pontos só pode estar definida entre pontos "exteriores" e mais precisamente entre os vértices do invólucro convexo do conjunto.



Solução: O(n log n)

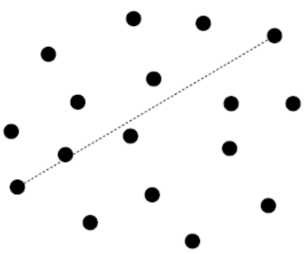
11

Invólucro Convexo

Diâmetro: formalizando

Definição: O diâmetro dum conjunto de n pontos $S = \{p_1, p_2, \ldots, p_n\}$ é a maior distancia entre dois deles:

$$diam(S) = \max_{1 \le i, j \le n} d(p_i, p_j)$$



12

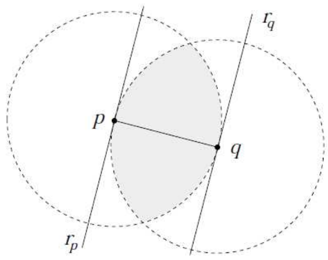
Invólucro Convexo

Diâmetro: formalizando

Proposição: O diâmetro de S se alcança em vértices do conv(S).

Prova:

Sejam p, q ∈ S dos pontos onde se realiza o diâmetro de S: diam(S) = d(p, q). Então o círculo com centro em p e raio d(p, q) contem a S já que se algum ponto q' de S quedara fora então d(p, q') > d(p, q) = diam(S), o que é um absurdo. Utilizando o mesmo raciocínio podemos afirmar que S está contido no círculo com centro em q e raio d(p,q). Por tanto, S está na intersecção de ambos círculos (lua).



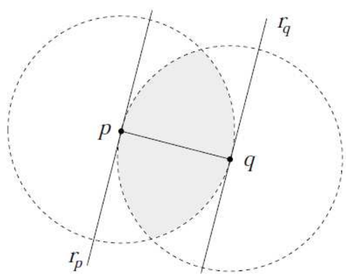
Invólucro Convexo

Diâmetro: formalizando

Proposição: O diâmetro de S se alcança em vértices do conv(S).

Prova (cont…):

Consideremos agora a recta r_p (respectivamente r_q) que passa por p (resp. por q) e é ao segmento pq. A recta r_p é tangente à lua em p e é por tanto uma recta de suporte de S. Então p (q) ∈ conv(S) e p (q) deve ser um vértice do conv(S). □



Invólucro Convexo

Algoritmo Scan

- Entrada: Lista dos pontos de S, ordenados em relação às suas x-coordenadas: $L = \{v_1, v_2, \ldots, v_n\}$

  Passo 1: $C = \{v_1, v_2\}$; $q = v_2$

  Passo 2: *(Scan)*

  For i = 3 to n do

  1. p = prev(q) (anterior em C)
  2. While $\Delta(p, q, v_i) \leq 0$,
     (a) Delete q de C
     (b) Update q = p; p = prev(q)
  3. $C = C \cup \{v_i\}$; $q = v_i$

Invólucro Convexo

Algoritmo Graham (variante)

- **Entrada:** Lista S: $S = (p_1, \ldots, p_n)$.
- **Saída:** Lista cíclica, duplamente ligada C com os vértices do conv(S) orientados positivamente.

  Passo 1: Ordenar os pontos de S com relação às x-coordenadas: $L = \{v_1, v_2, \ldots, v_n\}$

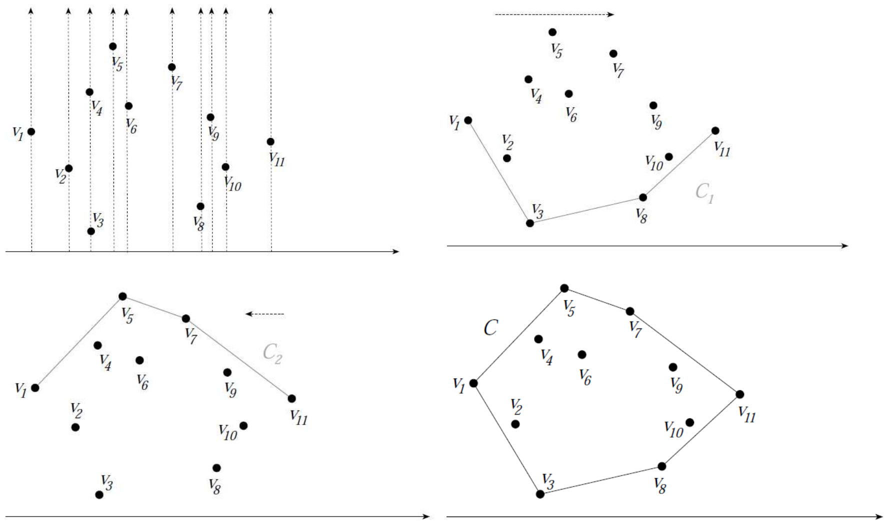
  Passo 2: Aplicar o **scan** a L e obter **C_1 - cadeia inferior do conv(S)**

  Passo 3: Inverter a ordem em L obtendo L* . Aplicar novamente o **scan** a L* e obter **C_2 - cadeia superior do conv(S)**

  Passo 4: Concatenar C_1 e C_2 e obter o conv(S)

Invólucro Convexo

Algoritmo Graham (variante)



Invólucro Convexo: Complexidade

Preliminares

- Encontrar cotas inferiores para resolver um dado problema é, de forma geral, uma questão delicada.
- Estratégia: Suponha que temos dois problemas P_1 e P_2, ambos sobre n dados de entrada tais que:
  - para o problema P_1 é conhecida uma cota inferior P_1 que é f(n) e,
  - sabemos reduzir P_1 a P_2 em tempo g(n) e g(n) « f(n)

  Então podemos afirmar que:
  - Podemos transformar em tempo g(n) a entrada de P_1 na entrada de P_2
  - Podemos transformar a solução de P_2 numa solução para P_1

$$P_1 \propto_{g(n)} P_2$$

18

Invólucro Convexo: Complexidade

Preliminares

$$P_1 \propto_{g(n)} P_2$$

- Seja A - qualquer algoritmo A que resolve P_2 então: podemos construir um algoritmo B que resolve P_1.
  Como?
  - Transformando, em tempo g(n), a entrada de P_1 na entrada de P_2
  - Utilizar o algoritmo A para resolver P_2

    Custo: $T_B(n) = g(n) + T_A(n)$
- Tendo em conta que P_1 é $\Omega(f(n))$ então:

  $\Rightarrow \exists\, c > 0$ tq $T_B(n) = g(n) + T_A(n) \geq cf(n)$

  $\overset{g(n) \ll f(n)}{\Rightarrow}$ $f(n) + T_A(n) \geq c'f(n)$

  $\Rightarrow$ f(n) é também uma cota inferior para o problema P_2

  $\Rightarrow T_A(n) = \Omega(f(n))$

19

Invólucro Convexo: Complexidade

Teorema:

O problema de calcular o invólucro convexo de n pontos no plano se reduz, em tempo linear, ao problema de ordenar n números reais:

Ordenar $\propto_n$ Invólucro Convexo

20

Invólucro Convexo: Complexidade

Teorema: Ordenar $\propto_n$ Invólucro Convexo

Prova:

Seja A um algoritmo que, dado um conjunto S de n pontos no plano calcula conv(S). A partir de A podemos construir um algoritmo B para ordenar um conjunto de n números reais como segue:

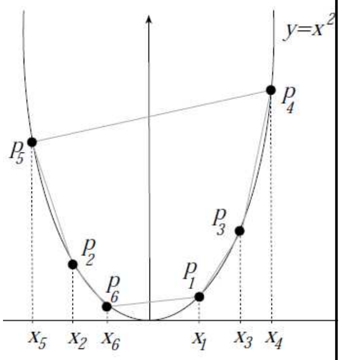
Algoritmo B

Entrada: N = $\{x_1, \ldots, x_n\}$ conjunto de n números reais

Saída: Lista com os números ordenados: $(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$ de m a M

Passo 1: Considerar os pontos do plano $p_i = (x_i, x_i^2)$, $\forall$ i = 1..n

$\{p_i\}$ – pontos da parábola $y = x^2$. Como a curva duma parábola define a fronteira dum conjunto convexo então no conv(S), S = $\{p_1, \ldots, p_n\}$ estão todos os pontos como vértices.



21

Invólucro Convexo: Complexidade

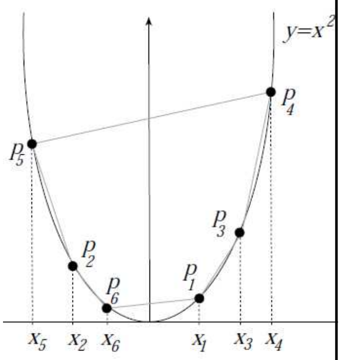
Teorema: Ordenar $\propto_n$ Invólucro Convexo

Prova (cont…)**:**

Passo 2: Aplicar o algoritmo A para calcular o conv(S). Este algoritmo vai devolver uma lista cíclica com os vértices do conv(S) com orientação positiva: L = $(p_{\sigma(1)}, \ldots, p_{\sigma(n)})$.

Passo 3: Localizar em L o ponto de menor x-coordenada, $p'_{\sigma(1)}$. Intercambiar os elementos de L até que $p'_{\sigma(1)}$ figure como o primeiro elemento da lista L.

Passo 4: Projectar os pontos de L, na nova ordem, sobre o eixo OX: $(x'_{\sigma(1)}, \ldots, x'_{\sigma(n)})$ que é lista ordenada que queremos construir.



(Exemplo com N = { -1, 2, 0 , -2, 1})

22

Invólucro Convexo: Complexidade

Teorema: **Ordenar $\propto_n$ Invólucro Convexo**

Prova (cont…)**:**

O custo do algoritmo B, $T_B(n)$ será: O(n) mais o tempo de execução do algoritmo A mais o tempo para encontrar o mínimo, a rotação e projecção de L no eixo OX:

$$T_B(n) = \Theta(n) + T_A(n)$$

É conhecido que a complexidade do problema da ordenação é $\Theta(n \log n)$ então $T_B(n) = \Omega(n \log n)$.

Tendo em conta que $n \ll n \log n$ então podemos concluir que:

$$T_A(n) = \Omega(n \log n) \qquad \square$$

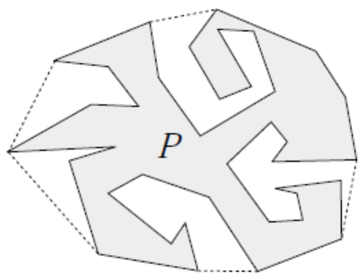
Corolário: A complexidade do problema de calcular o invólucro convexo de n pontos no plano é $\Omega(n \log n)$.

23

Invólucro Convexo dum Polígono

Problema:

Dado um polígono simples, P, determinar o conv(P).



24

Invólucro Convexo dum Polígono

Proposição: **Seja P um polígono simples e S o conjunto dos seus vértices. Então conv(P) = conv(S).**

Prova: É necessário provar cada uma das "$\subset$"

"conv(S) $\subset$ conv(P)"

S $\subset$ P $\subset$ conv(P). Como conv(P) é um conjunto e S $\subset$ conv(P) então conv(S) $\subset$ conv(P)

"conv(P) $\subset$ conv(S)"

conv(S) é um conjunto convexo por tanto contem todo segmento determinado por dois pontos de S. Em particular conv(S) contem a todas as arestas de P. Logo ∂P $\subset$ conv(S). Por outro lado se um polígono Q contem a fronteira doutro polígono R então é obvio que R $\subset$ Q. Logo P $\subset$ conv(S). Por último como conv(S) é um conjunto convexo que contem P então necessariamente conv(P) $\subset$ conv(S).

□

25

Invólucro Convexo dum Polígono

Proposição: **Seja P um polígono simples e S o conjunto dos seus vértices. Então conv(P) = conv(S).**

- **Algoritmo trivial** – utilizar apenas o conjunto dos pontos formado pelos vértices do polígono e para esse conjunto determinar o invólucro convexo (algoritmos de Graham, Jarvis, etc.).
- **Contudo NÃO estamos a utilizar o facto de que o conjunto dos pontos a tratar não é um conjunto arbitrário senão uma lista de vértices.**
- Será possível diminuir o custo do conv(P) de O(n log n) para O(n)?

Resposta: É possível mas não é imediato

26

Invólucro Convexo dum Polígono

- Será possível diminuir o custo do conv(P) de O(n log n) para O(n)?

  Resposta: É possível mas não é imediato
- Una primeira aproximação: **adaptar o Algoritmo de Graham**.
- O algoritmo de Graham consta de dois passos:
  (i) A ordenação dos pontos.
  (ii) O *Scan*
- O processo de **ordenação angular** pode ser interpretado como uma **poligonização estrelhada** do conjunto de ponto.
- O passo (ii), o ***Scan***, actua sobre a lista ordenada de pontos, i.e., sobre o polígono estrelhado que foi construído, verificando em cada vértice se o angulo interior é menor que π.

Invólucro Convexo dum Polígono

- Adaptação: "esquecer" o passo de ordenação que é o de maior custo computacional e aplicar directamente o Scan que é O(n).
- Nova versão do **Algoritmo de Graham** (para construir o conv(P))

Passo 1: Localizar em P o vértice com x-min e colocar este vértice como o primeiro vértice da lista.

Passo 2: Aplicar a P o processo *Scan*

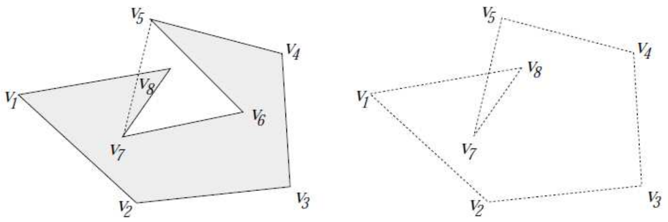
Processo *Scan*

1. Inicializar C, C $\leftarrow$ v1
2. (Scan) For i = 2 to n do
   (a) C $\leftarrow$ C $\cup$ $\{v_i\}$
   (b) q = v_i; p = prev(q)
   (c) While $\Delta(p, q, v_{i+1}) \leq 0$ and p = v_1 (Obs.: $v_{n+1} = v_1$)
   C $\leftarrow$ C \ {q}
   q = p; p = prev(q)
   (d) if $\Delta(p, q, v_{i+1}) \leq 0$ and p = v_1 then C $\leftarrow$ C \ {q}

Passo 3: Return C

Invólucro Convexo dum Polígono

- Esta nova versão do **Algoritmo de Graham** (para construir o conv(P)) **NÃO** calcula correctamente o invólucro de todos os polígonos simples.
- **Exemplo (TI verificar):**



- **Algoritmos para calcular o conv(P) em O(n):**
  - Algoritmo de Lee
  - Algoritmo de Melkman

29