بسم الله الرحمن الرحیم

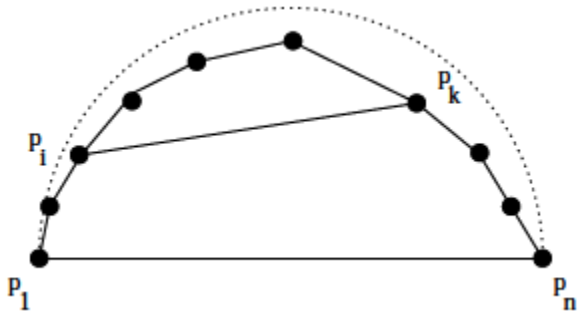
مثلث بندی با بیشترین وزن برای برخی چند ضلعی های محدب

تعریف مسئله:

پیدا کردن ماکسیمم وزن مثلث بندی نوع خاصی از چند ضلعی های محدب تحت عنوان چند ضلعی محدب نیمه دایره ای[1].

تعریف چند ضلعی نیم دایره ای:

چند ضلعی محدب P را نیم دایره ای میگوییم اگر ضلعی از P وجود داشته باشد بطوریکه Pداخل نیم دایره ای به قطر آن ضلع قرار بگیرد.



شکل 1:یک نمونه از چند ضلعی محدب نیم دایره ای

ویژگی های چند ضلعی محدب نیم دایره ای:

اگر چند ضلعی محدب نیم دایره ای Pرا با محموعه نقاط $P = (p_1, p_2, \dots, p_k, \dots, p_n)$ نشان دهیم به جز ویژگی های یک چندضلعی محدب دارای 2 ویژگی دیگر نیز می باشد که این دو ویژگی به راحتی قابل اثبات است.

[1] Semi circle

ویژگی1: اگر $\overline{p_1p_k}$ برای 1≤i<k≤n یالی از چندضلعی P باشد آنگاه چند ضلعی محدود به $\overline{p_1p_k}$ و زنجیره $(p_i, \dots, p_k)$ یک چند ضلعی محدب نیم دایره ای می باشد.

به منظور اثبات این خاصیت از لم ساده زیر استفاده می کنیم

لم1 :فرض کنید C نیم دایره ای به قطر AB و نقطهD داخل C است اگر و فقط اگر $\widehat{ADB}$ زاویه ای منفرجه باشد.

حال فرض کنید i<j<k باشد.با توجه به لم بالا p_j داخل نیم دایره است اگر $\widehat{p_ip_jp_k}$ >90. چون $\widehat{p_ip_jp_k}$ >$\widehat{p_1p_jp_n}$ و همچنین بنابر لم چون p_jداخل نیم دایره p_1p_nاست پس داریم$\widehat{p_1p_jp_n}$>90 . در نتیجه $\widehat{p_ip_jp_k}$>90 که خاصیت مطلوب ماست. این نشان می دهد که تمام نقاط چندضلعی محدب ($p_i,\dots,p_k$) داخل نیم دایره به قطر $\overline{p_ip_k}$ قرار می گیرند.این اثبات را کامل می کند.

ویژگی 2 : در چند ضلعی P یال $\overline{p_ip_j}$ برای1 ≤ i < j<n یا 1 < i < j ≤ n کوچکتر از $\overline{p_1p_n}$ است

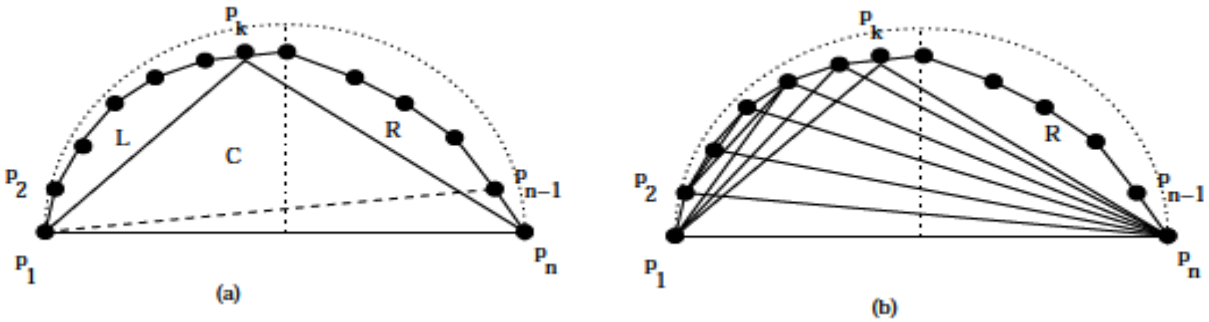
این ویژگی به راحتی قابل اثبات است.چون $\overline{p_1p_n}$ قطر دایره فرض شده و در یک دایره بیشترین طول برابر قطر دایره می باشد پس هیچ یالی در P نمی تواند بزرگتر از قطر دایره یا همان $\overline{p_1p_n}$ باشد.

پیدا کردن مثلث بندی با بیشترین وزن برای چند ضلعی محدب نیم دایره ای:

مثلث بندی با بیشترین وزن چند ضلعی نیم دایره ای P را MAT(P) می نامیم.و قبل از بدست آوردن MAT(P) لم زیر را اثبات می کنیم.

لم2 : اگر$P=(p_1,p_2,\dots,p_k,\dots,p_n)$ یک چند ضلعی محدی نیم دایره ای باشد.آنگاه یکی از یالهای$\overline{p_1p_{n-1}}$ و $\overline{p_np_2}$ باید متعلق به مثلث بندی با بیشترین وزن MAT(P) باشد

اثبات: (اثبات با برهان خلف)فرض کنیم که هیچ کدام از یالهای $\overline{p_1p_{n-1}}$ و $\overline{p_np_2}$ متعلق به MAT(P) نباشد. آنگاه باید یک نقطه مانندp_k برای 2<k<n-1 وجود داشته باشد که هر دو یال $\overline{p_1p_k}$ و $\overline{p_np_k}$ متعلق به MAT(P)باشد.



شکل 2 : برای اثبات لم 2

بدون کم شدن کلیت اصل مسئله فرض میکنیم که p_k در یک سمت عمود منصف ضلع $\overline{p_1p_n}$ قرار دارد, مثلا با توجه به شکل2-a, p_k در سمت چپ عمود منصف قرار دارد,دو یال $\overline{p_1p_k}$ و $\overline{p_np_k}$ چند ضلعی P را به 3 ناحیه R,L و C افراز می کنند.دو ناحیه L و R با توجه به ویژگی1 یک چند ضلعی محدب نیم دایره ای است.(ارجاع به قسمت (a) شکل 2).

اگر یالهای MAT(P) که در ناحیه Lقرار دارند برابر $(e_1, e_2, \dots\dots, e_{k-3})$ باشد و فرض کنیم که $E_L = (e_1, e_2, \dots\dots, e_{k-3}, \overline{p_1 p_k})$و اگر E_L^* به وسیله یالهای $(\overline{p_n p_2}, \overline{p_n p_3}, \dots\dots, \overline{p_n p_{k-1}})$ مشخص شود.آنگاه یک تناظر کامل بین E_L و E_L^* وجود دارد.زیرا E_L و E_L^* ناحیه LUC را مثلث بندی می کند,بنابراین تعداد یالهای داخلی دو مثلث بندی باید باهم برابر باشند.حال به آسانی می توان دید که در تناظری که به صورت $(e_i, \overline{p_n p_j})$ برای 1<i,j<kاست, $w(e_i) < w(\overline{n,j})$. زیرا با توجه به ویژگی 2هریال داخل E_L کوچکتر از $\overline{p_1 p_k}$ است و با توجه به اینکه p_k سمت چپ عمودمنصف $\overline{p_1 p_n}$ قرار دارد همه یالهای E_L^* از $\overline{p_n p_k}$ و $\overline{p_n p_k}$ از $\overline{p_1 p_k}$ بزرگتر است در نتیجه $w(E_L)$ از $w(E_L^*)$ کوچکتر است. اکنون ما یک مثلث بندی جدید می سازیم و نام آن را $T(P)$ قرار می دهیم که شامل همه یالهای $MAT(P)$ می باشد به جز یالهای داخل E_L که به جای آن یالهای داخل E_L^* قرار می گیرد.که اکنون به دلیل $w(E_L^*) > w(E_L)$، $w(T(P)) > w(MAT(P))$ می باشد که با فرض ماکسیمم بودن وزن $MAT(P)$ در تناقض است بنابراین نقطه ای مانند p_k نمی تواند وجود داشته باشد و یکی از یالهای $\overline{p_1 p_{n-1}}$ و $\overline{p_n p_2}$ باید متعلق به $MAT(P)$ باشد. □

با توجه به لم بالا سه الگوریتم برای بدست آوردن $MAT(P)$ پیشنهاد میشود:

1-راه حل کورکورانه:

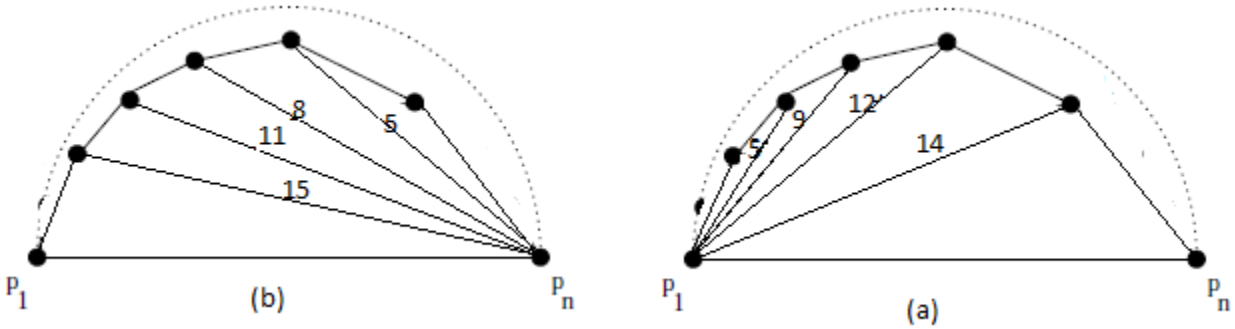
در این الگوریتم ما تمام حالات ممکن برای ساخت $MAT(P)$ با توجه به لم2 بدست می آوریم در این حالات در هر مرحله برای انتخاب یالها 2 انتخاب وجود دارد تا تعداد راسها برابر 3 شود که دیگر مثلث بندی امکان نداشته باشد پس تعداد حالات ممکن برابر است با

$$2 \times 2 \times \dots \times 2 = 2^{n-3} \in O(2^n)$$

که این عبارت دارای پیچیدگی نمایی است و به همین دلیل با وجودی که جواب درست را به ما می دهد راه حل مناسبی نیست

2-راه حل حریصانه:

در روش حریصانه اینگونه عمل می کنیم که در هر مرحله بین دویالی که با توجه به لم 2 بدست می آوریم بزرگترین طول را انتخاب میکنیم که در این روش می توان مثلثی بندی ای پیدا کرد که با توجه به این روش درست باشد ولی ماکسیمم وزن را به ما ندهد مانند شکل زیر



شکل4:مثال نقض برای روش حریصانه

برای مثال بالا با توجه به راه حل حریصانه شکل 4-(b) انتخاب می شود چون 15>14 در صورتی که وزن شکل 4- (a) دارای وزن بیشتری است.بنابراین از روش حریصانه برای بدست آوردن ماکسیمم وزن نمیتوان استفاده کرد.

$$w\big(MAT(a)\big) = w\left(\text{ضلعها}\right) + (14 + 13 + 11 + 9) = w\left(\text{ضلعها}\right) + 40$$

$$w\big(MAT(b)\big) = w\left(\text{ضلعها}\right) + (15 + 12 + 10 + 8) = w\left(\text{ضلعها}\right) + 39$$

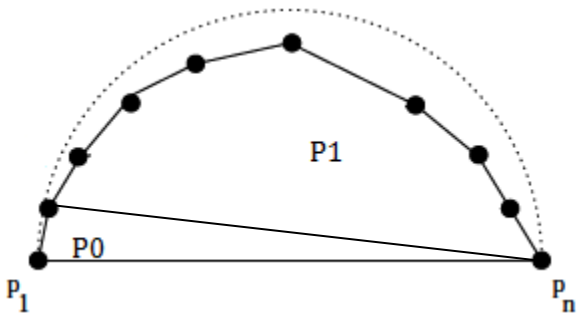
$$w\big(MAT(a)\big) > w(MAT(b))$$

3-راه حل پویا :

حال با استفاده مراحل زیر راه حل پویا را ارائه می دهیم

3-1:تعیین زیرساختار بهینه :

برای تعیین زیرساختار بهینه فرض می کنیم که مثلث بندی ما برای چند محدب نیم دایره ای$P = (p_1, p_2, \dots., p_n)$ دارای بیشترین وزن است باید نشان دهیم که همان مثلث بندی برای چندضلعی محدب نیم دایره ای$P_1 = (p_2, \dots., p_n)$یا$P_1 = (p_1, \dots., p_{n-1})$ نیز دارای بیشترین وزن است(با توجه به لم 2)



شکل 4:برای اثبات تعیین زیرساختار بهینه

برای اثبات زیرساختار بهینه اگر فرض کنیم شکل4 دارای مثلث بندی بهینه باشد که با یال$\overline{p_2p_n}$ چندضلعی P را به مثلث P_0 و چند ضلعی نیم دایره ای P_1 تقسیم کرد که باید نشان داد که مثلث بندی برای P_1 نیز بیشترین مقدار است که با روش cut & pasteفرض می کنیم که مثلث بندی برای P_1 ($MAT(P_1)$)ماکسیمم مقدار نباشد پس یک مثلث بندی با ماکسیمم مقدار وجود دارد که جایگزین($MAT(P_1)$) میکنیم و نام این مثلث بندی را $T(P_1)$ می نامیم که روابط زیر برقرار است

$$w\big(T(P_1)\big) > w\big(MAT(P_1)\big) \longrightarrow w\big(T(P_1)\big) + w(P_0) > w\big(MAT(P_1)\big) + w(P_0) \longrightarrow w\big(T(P)\big) > w(MAT(P))$$

که عبارت بالا با ماکسیمم بودن وزن $MAT(P)$ در تناقض است بنابراین مثلث بندی ($MAT(P_1)$) نیز ماکسیمم است.

3-2:جواب بازگشتی:

با توجه به زیر ساختار بهینه و لم 2 می توان رابطه بازگشتی زیر را برای بدست آوردن ماکسیمم وزن بدست آورد

$$\omega(i,j)=\begin{cases}\omega(\overline{i,i+1}) & j=(i+1)\\ max\{\omega(i,j-1)+\omega(\overline{j-1,j}),\omega(i+1,j)+\omega(\overline{i,i+1})\}+\omega(\overline{i,j}) & \text{otherwise}\end{cases}$$

3-3:الگوریتم پویا

با توجه به رابطه بازگشتی بالا میتوان جواب ها را به صورت بالا مثلثی در آرایه w به ابعاد $n \times n$ به صورت قطری ذخیره کرد و همچنین می توان از آرایه کمکی s به ابعاد $n \times n$ برای ساخت جواب بهینه استفاده کرد

و همچنین $w(\overline{p_i p_{i+1}})$ برابر طول ضلع $\overline{p_i p_{i+1}}$ است.

ALGORITHM $MAT-FIND(P)$

Input: P $//a\ semicircled\ convex\ n-sided\ polygon: (p_1, p_2, \dots, p_n).$

Output: w, s

$n = length(P);$

for $i = 1$ ***to*** $n-1$ ***do***

$w(i, i+1) = w(\overline{i, i+1});$

end

for $l = 2$ ***to*** $n-1$ ***do***

for $i = 1$ ***to*** $n-1$ ***do***

if $w(i, i+l-1) + w(\overline{i+l-1, i+l}) > w(i+1, i+l) + w(\overline{i, i+1})$ ***Then***

$w(i, i+l) = w(i, i+l-1) + w(\overline{i+l-1, i+l}) + w(\overline{i, i+l});$

$s(i, i+l) = 1;$

else

$w(i, i+l) = w(i+1, i+l) + w(\overline{i, i+1}) + w(\overline{i, i+l});$

$s(i, i+l) = 0;$

end

end

زمان اجرای الگوریتم بالا می باشد به صورت زیر بدست می آید

$$\sum_{l=2}^{n-1}\sum_{i=1}^{n-l} c = \sum_{l=2}^{n-1} c(n-l-1+1) = cn(n-1-2+1) - c\sum_{l=2}^{n-1} l = cn^2 - 2cn - \frac{c(n-2)(n-1)}{2} \epsilon \, O(n^2)$$

که به روش ساده تر میتوان گفت هر مرحله حلقه i به تعداد n-l بار اجرا میشود که جواب به صورت زیر است

3-4:ساخت جواب بهینه:

برای ساخت جواب بهینه از آرایه کمکی S استفاده می کنیم و الگوریتم آن به صورت زیر است

ALGORITHM: $Print - MAT(s,i,j,N)$

Input: s,i,j,N

Output: مجموعه یالها در مثلث بندی

if $j = i+1$ ***Then***

for $k = 1$ ***to*** $N-1$ ***do***

$print\ '(''p'_k 'p'_{k+1}')';$

end

else

if $s(i,j) = 1$ ***Then***

$Print - MAT(s,i,j-1,N)$

else

$Print - MAT(s,i+1,j,N)$

end

$print\ ('p'_i 'p'_j);$

end

end

الگوریتم بالا در ابتدا با $Print - MAT(s,1,n,n)$ فراخوانی می شود

پایان