نگاهی الگوریتمی به مسائل شمارشی در هندسه‌ی اعداد

نوید علامتی

چکیده

در نگاهی الگوریتمی به هندسه‌ی اعداد، ابتدا الگوریتمی را مطرح می‌کنیم که از پیچیدگی الگوریتم تعمیم‌یافته‌ی اقلیدس برای یافتن ب.م.م می‌باشد و این الگوریتم تعداد نقاط مشبکه‌ای را روی یک پاره‌خط محاسبه می‌کند. سپس الگوریتمی با همین پیچیدگی برای شمارش نقاط مشبکه‌ای در یک مثلث ذکر می‌کنیم و نهایتا با ارایه‌ی الگوریتمی برای چندضلعی‌ها این بخش را به پایان می‌رسانیم. الگوریتم‌هایی که در این بخش مطرح می‌شوند، ضمن سادگی نسبت به موارد مشابه خود، از پیچیدگی زمانی یکسانی نسبت به آنها برخوردار هستند.شمارش نقاط مشبکه‌ای در چندضلعی‌ها در دو بعد و بالاتر یکی از مسائل بنیادین ریاضی است که برای مدت طولانی مورد مطالعه قرار گرفته است. شمارش نقاط مشبکه‌ای در حالت ساده یعنی حالتی که مختصات صحیح باشند به طور موفقیت‌آمیزی توسط پیک در قرن نوزدهم انجام گرفته است. شمارش نقاط مشبکه‌ای در چندضلعی‌هایی که رئوسشان مختصات گویا دارند نیز برای مدت طولانی مورد بحث بوده است. زمانی که بعد ثابت باشد، Barvinok نشان داد که الگوریتم چندجمله‌ای برای شمارش نقاط مشبکه‌ای در چندضلعی‌هایی با مختصات رئوس گویا وجود دارد. در دو بعد، Beck و Robins الگوریتم کارایی ارائه دادند که از مثلث‌بندی چندضلعی و شمارش نقاط مشبکه‌ای در مثلث قائم‌الزاویه استفاده می‌کرد. اما، در مقام عمل الگوریتم‌های موجود برای مختصات گویا کارایی چندانی ندارند. به عنوان مثال الگوریتمی که توسط Barvinok در سال ۱۹۹۳ ارائه شد، سختی بسیاری برای پیاده‌سازی داشت و تا سال ۲۰۰۴ اصلا پیاده‌سازی نشده بود!

در ادامه‌ی بحث، الگوریتمی کارا و راحت برای پیاده‌سازی جهت شمارش نقاط مشبکه‌ای در چندضلعی‌هایی با مختصات گویا ارائه خواهیم داد. این الگوریتم تا حد زیادی ساده‌تر از الگوریتم Beck و Robins بوده، در حالی که هر دو الگوریتم از پیچیدگی زمانی یکسانی برخوردارند. سادگی الگوریتم بسیار مهم است، چرا که سادگی باعث راحتی در پیاده‌سازی می‌شود و امکان بروز باگ را در برنامه‌ها پایین می‌آورد. از طرفی دیگر منطقی است وقتی که هردو از لحاظ نظری پیچیدگی زمانی یکسانی دارند، برنامه‌ی ساده در عمل سریع‌تر از دیگری اجرا شود. این الگوریتم توسط Yanagisawa در پژوهشکده‌ی شرکت IBM منتشر شده است.

لازم به ذکر است که در ادامه‌ی بخش منظور از $\{x\}$ در واقع، $x - \lfloor x \rfloor$ یعنی جزء اعشاری می‌باشد.

۱ نگاهی الگوریتمی به هندسه‌ی اعداد

۱.۱ الگوریتم شمارش نقاط مشبکه‌ای برای پاره‌خط‌ها

در این بخش، الگوریتمی برای محاسبه‌ی تعداد نقاط مشبکه‌ای بر روی یک پاره‌خط داده شده ارائه می‌گردد. برای ساخت الگوریتم از تعریف و لم‌های زیر استفاده خواهیم کرد.

تعریف ۱ *فرض کنید x_1 ، y_1 ، x_2 و y_2 اعداد گویایی باشند. تعداد نقاط مشبکه‌ای روی پاره‌خط تعریف شده با نقاط (x_1, y_1) و (x_2, y_2) را با $L(x_1, y_1, x_2, y_2)$ نشان می‌دهیم.*

لم ۱ *در صورتی که اعداد صحیح نامنفی a و b داده شده باشند، الگوریتمی برای محاسبه‌ی مقسوم‌علیه مشترک آنها وجود دارد که به تبع آن اعداد x و y نیز می‌توانند محاسبه گردند به‌طوری که:*

$$ax + by = gcd(a, b)$$

جزئیات پیچیدگی زمانی چنین الگوریتمی در مرجع [۱۲] ذکر شده است.

لم ۲ *معادله‌ی دیوفانتی خطی $ax + by = c$ جواب دارد اگر و فقط اگر c بر d بخش‌پذیر باشد که در آن $d = gcd(a, b)$. همچنین اگر (x_0, y_0) جوابی از معادله باشد، مجموعه جواب‌های معادله شامل جفت‌های صحیح (x, y) است که:*

$$x = x_0 + \frac{b}{d}k, \quad y = y_0 - \frac{a}{d}k, \quad k \in \mathbb{Z}$$

حال می‌توانیم الگوریتم را بسازیم.

قضیه ۳ *$L(x_1, y_1, x_2, y_2)$ می‌تواند در $O(max\{\log|x_1|, \log|y_1|, \log|x_2|, \log|y_2|\})$ گام محاسبه گردد.*

برهان. بدون از دست دادن کلیت می‌توان فرض کرد که $0 \leq x_1 \leq x_2$ و $y_1 \geq y_2 \geq 0$ ((x_1, y_1) و (x_2, y_2) را در صورتی که $0 < x_1 \leq x_2$ و $y_1 \geq y_2 > 0$ باشد، به ترتیب به $(-x_1, y_1)$ و $(-x_2, y_2)$ تغییر می‌دهیم.) چون x_1 ، y_1 ، x_2 و y_2 اعدادی گویا هستند، اعداد صحیح a ، b و c را می‌توان چنان پیدا کرد که خط $ax+by = c$ از نقاط (x_1, y_1) و (x_2, y_2) بگذرد. در صورتی که $a = 0$ یا $b = 0$ باشد، $L(x_1, y_1, x_2, y_2)$ به طور بدیهی قابل محاسبه است. در غیر این صورت با استفاده از لم ۱ می‌توان بزرگ‌ترین مقسوم‌علیه

مشترک a و b و اعداد صحیح p و q را چنان پیدا کرد که $ap+bq=d$. چون $(x,y)=(\frac{c}{d}p,\frac{c}{d}q)$ در معادله‌ی $ax+by=c$ صدق می‌کند، با استفاده از لم ۲، همه‌ی جواب‌های صحیح معادله را به شکل زیر می‌توان بیان کرد:

$$x=\frac{c}{d}p+\frac{b}{d}k,\quad y=\frac{c}{d}q-\frac{a}{d}k,\quad k\in\mathbb{Z}$$

چون $x_1\le x\le x_2$ نابرابری زیر باید برقرار باشد:

$$(x_1-\frac{c}{d}p)(\frac{d}{b})\le k\le(x_2-\frac{c}{d}p)(\frac{d}{b})$$

بنابراین،

$$L(x_1,y_1,x_2,y_2)=\left\lfloor(x_2-\frac{c}{d}p)\frac{d}{b}\right\rfloor-\left\lceil(x_1-\frac{c}{d}p)\frac{d}{b}\right\rceil+1$$

زمان‌برترین قسمت الگوریتم همان استفاده از الگوریتم تعمیم‌یافته‌ی اقلیدسی است که در لم ۱ اشاره گردید. بنابراین در $O(max\{\log a,\log b\})$ گام می‌توان نقاط را شمارش کرد. ■

۲.۱ الگوریتمِ شمارشِ نقاطِ مشبکه‌ای برایِ مثلث‌ها

در این بخش، الگوریتم بازگشتی کوتاهی برای شمارش نقاط مشبکه‌ای در یک مثلث قائم‌الزاویه ارائه می‌کنیم. برای ساخت الگوریتم ابتدا به بیان چند تعریف و لم می‌پردازیم.

تعریف ۲ *فرض کنید a ، b و c اعداد صحیح مثبت باشند. مثلث قائم‌الزاویه‌ی $T(a,b,c)$ را به صورت زیر تعریف می‌کنیم:*

$$T(a,b,c)=\{(x,y)\in\mathbb{R}^2|ax+by\le c,x>0,y>0\}$$

تعریف ۳ *فرض کنید a ، b و c اعداد صحیح مثبت باشند. تعداد نقاط مشبکه‌ای با مختصات صحیح و مثبت درون و روی مثلث $T(a,b,c)$ را با $N(a,b,c)$ نشان می‌دهیم.*

از تعریف، لم زیر به وضوح برقرار است.

لم ۴ *فرض کنید a، b و c اعداد صحیح مثبت باشند. در این صورت،* $N(a,b,c)=N(b,a,c)$

لم ۵ *فرض کنید a و c اعداد صحیح مثبت باشند. در این صورت:* $N(a,a,c)=\frac{1}{2}\left\lfloor\frac{c}{a}\right\rfloor(\left\lfloor\frac{c}{a}\right\rfloor-1)$.

لم ۶ *فرض کنید a ، b و c اعداد صحیح مثبت باشند و $a>b$. فرض کنید $m=\left\lfloor\frac{c}{a}\right\rfloor$ ، $h=\frac{c-am}{b}$ ، $k=\left\lfloor\frac{a-1}{b}\right\rfloor$ و $c'=c-b(km+\lfloor h\rfloor)$. در این صورت معادله‌ی زیر برقرار است:*

$$N(a,b,c)=N(a-bk,b,c')+\frac{1}{2}km(m-1)+m\lfloor h\rfloor$$

برهان. ابتدا، به بیان مساحت مثلث $T(a,b,c)$ با استفاده از $N(a,b,c)$ می‌پردازیم. در صورتی که با دقت به مربع‌های واحد در شکل صفحه‌ی بعد نگاه کنیم، یعنی مربع‌هایی واقع در مثلث قائم‌الزاویه که چهار رأس آن نقاط مشبکه‌ای می‌باشند، تعداد مربع‌های واحد در $T(a,b,c)$ با $N(a,b,c)$ برابر است. حال بقیه‌ی مثلث $T(a,b,c)$ را به چند ذوزنقه‌ی S_i و یک مثلث R تقسیم می‌کنیم. برای هر عدد صحیح i که $0<i\le m$ ذوزنقه‌ی S_i را به صورت زیر تعریف می‌کنیم:

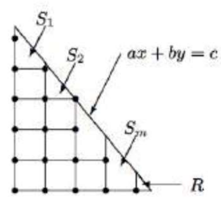
$$S_i=\{(x,y)\in\mathbb{R}^2|ax+by\le c,i-1\le x\le i,$$
$$y\ge\left\lfloor\frac{c-ai}{b}\right\rfloor\}$$

مثلث R نیز به صورت زیر تعریف می‌شود:

$$R=\{(x,y)\in\mathbb{R}^2|ax+by\le c,x\ge m,y\ge 0\}$$

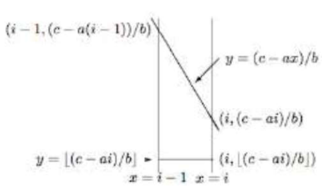
پس، مساحت مثلث $T(a,b,c)$ برابر است با:

$$|T(a,b,c)|=N(a,b,c)+\sum_{i=1}^{m}|S_i|+|R|$$



حال با توجه به اینکه مثلث R شامل سه رأس (m,h) ، $(m,0)$ و $(\frac{c}{a},0)$ است، به راحتی قابل بررسی است که $|T(a,b,c)|=\frac{c^2}{2ab}$ و $|R|=\frac{1}{2}h(\frac{c}{a}-m)$ می‌باشد. مساحت S_i نیز به راحتی از طریق رابطه‌ی زیر قابل محاسبه است:

$$|S_i|=\frac{1}{2}(\frac{c-a(i-1)}{b}-\left\lfloor\frac{c-ai}{b}\right\rfloor+\frac{c-ai}{b}-\left\lfloor\frac{c-ai}{b}\right\rfloor)$$
$$=\frac{1}{2}(\frac{a}{b}+2\{\frac{c-ai}{b}\})$$



بنابراین، $N(a,b,c)$ از طریق رابطه‌ی زیر به‌دست می‌آید:

$$N(a,b,c)=|T(a,b,c)|-|R|-\sum_{i=1}^{m}|S_i|$$

$$= \frac{c^۲}{۲ab} - \frac{h}{۲}(\frac{c}{a} - m) - \frac{۱}{۲}\sum_{i=۱}^{m}(\frac{a}{b} + ۲\{\frac{c-ai}{b}\})$$

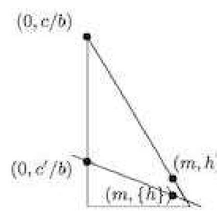
$$= \frac{cm}{۲b} + \frac{h}{۲}m - \frac{۱}{۲}\sum_{i=۱}^{m}(\frac{a}{b} + ۲\{\frac{c-ai}{b}\})$$

برابری سوم با توجه به اینکه $c - bh = am$ است، برقرار می‌باشد، چرا که خط $ax + by = c$ از نقطه‌ی (m, h) می‌گذرد.

به راحتی می‌توان مشاهده کرد که $a - bk > ۰$ ، چون $k = \lfloor \frac{a-۱}{b} \rfloor$. زمانی که خط $ax + by = c$ از نقاط $(۰, \frac{c}{b})$ و (m, h) می‌گذرد، خط $(a - bk)x + by = c'$ از نقاط $(۰, \frac{c'}{b})$ و $(m, \{h\})$ عبور می‌کند. بنابراین، مشابه با بحثی که برای $N(a, b, c)$ داشتیم، برای $N(a - bk, b, c')$ خواهیم داشت:

$$N(a - bk, b, c') =$$

$$\frac{c'm}{۲b} + \frac{\{h\}}{۲}m - \frac{۱}{۲}\sum_{i=۱}^{m}(\frac{a - bk}{b}) + ۲\{\frac{c' - (a - bk)i}{b}\})$$



در نتیجه:

$$N(a, b, c) - N(a - bk, b, c') =$$

$$\frac{cm}{۲b} - \frac{c'm}{۲b} + \frac{h}{۲}m - \frac{\{h\}}{۲}m$$

$$-\frac{۱}{۲}\sum_{j=۱}^{m}((\frac{a}{b} + ۲\{\frac{c - aj}{b}\}) - (\frac{a - bk}{b} + ۲\{\frac{c' - (a - bk)j}{b}\}))$$

$$= \frac{cm}{۲b} - \frac{(c - b(km + \{h\}))m}{۲b} + \frac{۱}{۲}m\lfloor h \rfloor$$

$$-\frac{۱}{۲}\sum_{j=۱}^{m}((\frac{a}{b} + ۲\{\frac{c - aj}{b}\}) - (\frac{a}{b} - k + ۲\{\frac{c - aj}{b}\}))$$

$$= \frac{(km + \lfloor h \rfloor)m}{۲} + \frac{۱}{۲}m\lfloor h \rfloor - \frac{۱}{۲}\sum_{j=۱}^{m} k$$

$$= \frac{k}{۲}m(m - ۱) + m\lfloor h \rfloor$$

لذا لم مورد نظر اثبات می‌شود. ■

قضیه ۷ *فرض کنید a، b و c اعداد صحیح مثبت باشند. $N(a, b, c)$ را می‌توان در $O(\max\{\log a, \log b\})$ گام محاسبه کرد.*

برهان. فرض کنید که $k = \lfloor \frac{a-۱}{b} \rfloor$ باشد. به راحتی قابل بررسی است که $a - bk$ زمانی با b برابر است که a بر b بخش‌پذیر باشد و $a - bk$ زمانی با $a \mod b$ برابر است که a بر b بخش‌پذیر نباشد. بنابراین، $۰ < a - bk \le b$ برقرار است.

با استفاده از لم ۴، بدون کاستن از کلیت می‌توان فرض کرد که $a \ge b$ در بحث زیر برقرار است. با استفاده از لم ۵ و لم ۶ و این واقعیت که $۰ < a - bk \le b$ ، می‌توان از الگوریتم زیر برای محاسبه‌ی $N(a, b, c)$ استفاده کرد.

Algorithm ۱ calcN(a, b, c)

a, b and c are positive integers such that $a \ge b$

  $m \leftarrow \lfloor \frac{c}{a} \rfloor$
  if $a = b$ **then**
    return $\frac{1}{2}m(m-1)$
  else
    $k \leftarrow \lfloor \frac{a-1}{b} \rfloor$
    $h' \leftarrow \lfloor \frac{c-am}{b} \rfloor$
    return calcN$(b, a - bk, c - b(km + h')) + \frac{1}{2}m(m-1) + mh'$
  end if

حال زمان اجرای محاسبه‌ی calcN را بررسی می‌کنیم. مشاهده می‌شود که زمان اجرا متناسب با تعدا فراخوانی بازگشتی الگوریتم calcN می‌باشد. زمانی که a بر b بخش‌پذیر نباشد، یک فراخوانی بازگشتی انجام می‌دهد که پارامترهای (a, b) را به $(b, a \mod b)$ تغییر می‌دهد. زمانی که a بر b بخش‌پذیر است، یک فراخوانی نهایی بازگشتی انجام می‌دهد که پارامترهای (a, b) را به (b, b) تغییر می‌دهد. بنابراین تعداد فراخوانی‌های بازگشتی الگوریتم calcN برابر با تعداد فراخوانی‌های بازگشتی در الگوریتم اقلیدسی برای محاسبه‌ی بزرگ‌ترین مقسوم‌علیه مشترک است. در نتیجه این الگوریتم $O(\log a)$ مرحله انجام می‌دهد. ■

توجه داشته باشید که الگوریتم calcN فقط با اعداد صحیح سر و کار دارد و این سادگی الگوریتم کلی را میسر می‌سازد.

۳.۱ الگوریتمی برای چندضلعی‌ها

در این بخش، الگوریتمی برای شمارش تعداد نقاط مشبکه‌ای در یک چندضلعی داده شده (درون و روی اضلاع) ارائه می‌دهیم. ابتدا ایده‌های اصلی این الگوریتم را توصیف می‌کنیم و سپس به جزئیات می‌پردازیم.

یک چندضلعی به طور صوری به صورت زیر تعریف می‌شود: فرض کنید P یک چندضلعی گویای دوبعدی با رئوس $p_۰, p_۱, \ldots, p_n$ باشد

به‌طوری که $p_n = p_0$ و اضلاع آن پاره‌خط‌هایی باشند که هر پاره‌خط p_i را به p_{i+1} متصل می‌سازد. هر رأس p_i مختصات (x_i, y_i) دارد که در آن x_i و y_i اعداد گویای مثبت هستند. فرض می‌کنیم که رئوس به ترتیب ساعتگرد باشند و چندضلعی در حالت خاص نباشد، به عبارت دیگر دو ضلع فقط در یک نقطه می‌توانند متقاطع باشند و هیچ نقطه‌ای به عنوان رأس بیش از یک‌بار ظاهر نگردد. توجه داشته باشید که چندضلعی لزوما محدب نیست.

حال به تشریح ایده‌های اصلی الگوریتم می‌پردازیم. برای سادگی، فرض کنید که هیچ نقطه‌ی مشبکه‌ای روی اضلاع چندضلعی نباشد و هیچ یک از رئوس P مختصات صحیحی نداشته باشند. برای شمارش تعداد نقاط مشبکه‌ای، از رابطه‌ای مشابه رابطه‌ای که برای محاسبه‌ی مساحت آنها استفاده می‌گردد، بهره می‌بریم. مساحت یک چندضلعی P می‌توان با رابطه‌ی زیر محاسبه کرد:

$$area(P) = |\sum_{i=1}^{n} \frac{(x_i - x_{i-1})(y_i - y_{i-1})}{2}|$$

چرا که اگر فرض کنیم که ناحیه‌ی D_i ذوزنقه‌ای باشد که رئوس آن (x_{i-1}, y_{i-1}) ، $(x_{i-1}, 0)$ ، $(x_i, 0)$ و (x_i, y_i) باشد. در نتیجه مساحت چندضلعی P از طریق رابطه‌ی زیر می‌تواند محاسبه شود:

$$area(P) = |\sum_{i=1}^{n} sgn(x_i - x_{i-1})|D_i||$$

به‌طوری که $sgn(x)$ در صورتی که $x > 0$ باشد، برابر با یک و در غیر این‌صورت برابر با -1 است. تعداد نقاطه‌های مشبکه‌ای در چندضلعی P می‌تواند از رابطه‌ی زیر محاسبه گردد:

$$num(P) = |\sum_{i=1}^{n} sgn(x_i - x_{i-1})(\#\{(x,y) \in \mathbb{Z}^2 | (x,y) \in D_i\})|$$

مسئله‌ی باقی‌مانده نحوه‌ی محاسبه‌ی تعداد نقاط مشبکه‌ای در ذوزنقه‌ی D_i می‌باشد. تمام کاری که بایستی انجام داد این است که ذوزنقه‌ی D_i را به یک مستطیل و یک مثلث قائم‌الزاویه تقسیم کنیم، چرا که شمارش تعداد نقاط مشبکه‌ای مستطیل ساده است و تعداد نقاط مشبکه‌ای در یک مثلث قائم‌الزاویه نیز می‌تواند از طریق الگوریتم calcN که در بخش قبل ارائه شد، محاسبه گردد.

برمبنای ایده‌های فوق، الگوریتم مورد نظر را می‌سازیم. اگر نقاط مشبکه‌ای روی اضلاع چندضلعی P وجود داشته باشند یا برخی از رئوس P مختصات صحیح داشته باشند، باید الگوریتم را تغییر دهیم. قبل از تشریح الگوریتم، ابتدا یک نمادگذاری را تعریف می‌کنیم.

تعریف ۴ *فرض کنید که x_{i-1} ، y_{i-1} ، x_i و y_i اعداد صحیح مثبتی باشند که $x_i \neq x_{i-1}$. حال $N(x_{i-1}, y_{i-1}, x_i, y_i)$ را تعداد نقاط مشبکه‌ای که در داخل یا مرز ذوزنقه‌ی D_i به رئوس (x_{i-1}, y_{i-1}) ، $(x_{i-1}, 0)$ ، $(x_i, 0)$ و (x_i, y_i) قرار دارد، تعریف می‌کنیم.*

ابتدا الگوریتم را برای یک چندضلعی محدب بیان می‌کنیم. تعداد نقاط مشبکه‌ای در چندضلعی P می‌تواند از رابطه‌ی زیر محاسبه گردد:

$$num(P) = |\sum_{i=1}^{n}(N(D_i) - u(P,i))|$$

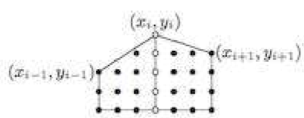
به‌طوری‌که:

$$N(D_i) = \begin{cases} N(x_{i-1}, y_{i-1}, x_i, y_i) & x_{i-1} < x_i \\ L(x_i, y_i, x_{i-1}, y_{i-1}) - N(x_i, y_i, x_{i-1}, y_{i-1}) & x_{i-1} > x_i \\ 0 & x_{i-1} = x_i \end{cases}$$

و

$$u(P,i) = \begin{cases} \lfloor y_i + 1 \rfloor & \text{if} \quad x_i \text{ is integer and } x_{i-1} < x_i < x_{i+1} \\ \lceil -y_i \rceil & \text{if} \quad x_i \text{ is integer and } x_{i-1} > x_i > x_{i+1} \end{cases}$$

به راحتی می‌توان بررسی کرد که رابطه‌های داده شده به درستی تعداد نقاط مشبکه‌ای را در یک چندضلعی P نشان می‌دهند. توجه داشته باشید که $u(P,i)$ گنجانیده شده تا از شمارش دوباره اجتناب گردد. در صورتی که $u(P,i)$ وجود نداشته باشد، رابطه‌ی مورد نظر ممکن است برخی نقاط را دوبار در نظر بگیرد.

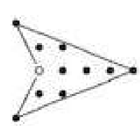


برای چندضلعی‌های نامحدب، باید در رابطه تغییراتی را انجام دهیم چرا که برخی رئوس P ممکن است اشتباها به عنوان نقاط داخلی چندضلعی و نه رئوس آن‌ها شمارش گردند. برای اجتناب از این حالت، $v(P,i)$ را به صورت زیر تعریف می‌کنیم:

$$v(P,i) = \begin{cases} 1 & \text{if} \quad (x_i, y_i) \in \mathbb{Z}^2, x_{i-1}, x_{i+1} < x_i \text{ and} \\ & \begin{vmatrix} x_i - x_{i-1} & x_{i+1} - x_i \\ y_i - y_{i-1} & y_{i+1} - y_i \end{vmatrix} > 0 \\ -1 & \text{if} \quad (x_i, y_i) \in \mathbb{Z}^2, x_{i-1}, x_{i+1} > x_i \text{ and} \\ & \begin{vmatrix} x_i - x_{i-1} & x_{i+1} - x_i \\ y_i - y_{i-1} & y_{i+1} - y_i \end{vmatrix} < 0 \end{cases}$$

به‌طوری که $|A|$ دترمینان ماتریس A می‌باشد و قرار دهید:

$$num(P) = |\sum_{i=1}^{n}(N(D_i) - u(P,i) - v(P,i))|$$



بنابراین، یک الگوریتم خطی از تعداد رئوس چندضلعی برای شمارش

تعداد نقاط مشبکه‌ای در یک چندضلعی داریم. با اینکه این الگوریتم فرض می‌کند که رئوس چندضلعی باترتیب ساعتگرد داده شده‌اند، می‌توان مشاهده کرد که این الگوریتم تعداد نقاط مشبکه‌ای در داخل چندضلعی را در حالتی که با ترتیب پادساعتگرد داده شوند نیز، محاسبه می‌کند.

نتیجه‌گیری و کارهای آتی

در بخش الگوریتمی هندسه‌ی اعداد، الگوریتمی از پیچیدگی پیدا کردن ب.م.م دو عدد صحیح برای شمارش نقاط مشبکه‌ای در یک پاره‌خط ارائه گردید. الگوریتمی که در صورتی که از ویژگی‌های اثبات شده از هندسه‌ی اعداد استفاده نمی‌کردیم، چه بسا مرتبه‌ی زمانی آن بسیار افزایش می‌یافت. با استفاده از همین تکنیک و از همین پیچیدگی زمانی، الگوریتمی برای شمارش نقاط مشبکه‌ای در یک مثلث ارائه گردید و نهایتا یک الگوریتم خطی از تعداد رئوس چندضلعی برای شمارش تعداد نقاط مشبکه‌ای در یک چندضلعی ارائه شد. از طرفی چون این الگوریتم نیازی به ذخیره‌ی مختصات همه‌ی رئوس چندضلعی داده شده در یک زمان ندارد فضای کمتری نسبت به الگوریتم‌های شمارشی که از یک الگوریتم مثلث‌بندی استفاده می‌کنند، نیاز دارد. این کار، الگوریتم را برای پیاده‌سازی نیز ساده می‌کند. کارهای آتی برای این مسئله می‌تواند تعمیم الگوریتم به ابعاد بالاتر از جمله بعد سوم باشد. از طرفی دیگر، با تجمیع ابزارهایی که هندسه‌ی اعداد به‌دست می‌دهد، شاید بتوان به پیچیدگی محاسباتی کمتری نسبت به پیچیدگی محاسباتی الگوریتم‌های گفته شده دست یافت.

مراجع

[1] A.K.Lenstra, Lattices and factorization of polynomialsm Report IW, Amsterdam (1981).

[2] A.I. Barvinok, A polynomial-time algorithm for counting integral points in poly- hedra when the dimension is fixed, in Proceedings of 34th Symposium on the Foun- dations of Computer Science (FOCS '93), pp. 566–572, IEEE Computer Society Press, New York, 1993.

[3] M. Beck and S. Robins, Explicit and efficient formulas for the lattice point count inside rational polygons using Dedekind - Rademacher sums, Discrete and Com- putational Geometry, Vol. 27, No. 4, pp. 443–459, 2002.

[4] B. Chazelle, Triangulating a simple polygon in linear time Discrete and Computational Geometry, Vol. 6, No. 5, pp. 485–524, 1991.

[5] J.A. De Loera, The many aspects of counting lattice points in polytopes, Mathematische Semesterberichte manuscript (http://www.math.ucdavis.edu/˜ deloera/RECENT WORK/semesterberichte.pdf).

[6] J.A. De Loera, R. Hemmecke, J. Tauzer, and R. Yoshida, Effective lattice point counting in rational convex polytopes, The Journal of Symbolic Computation, Vol.38, No. 4, pp. 1273–1302, 2004.

[7] K. Kolodziejczyk, Hadwiger - Wills-type higher dimensional generalizations of Pick's theorem, Discrete and Computational Geometry, Vol. 24, No. 2-3, pp. 355– 364, 2000.

[8] G. Pick, Geometrisches zur Zahlenlehre Sitzungber. Lotos, Naturwissen Zeitschrift Prague, Vol. 19, pp. 311–319, 1899.

[9] Gruber, Peter, Convex and discrete geometry, Springer Grundlehren Series (vol.336) 2007.

[10] . J. Hans-Gill, M.Raka and R. Sehmi, On Conjectures of Minkowski and Woods for n=7, Journal of Number Theory, Vol 129 (2009), 1011-1033.

[11] C. T. McMullen, Minkowski's conjecture, well rounded lattices and topological dimension, Journal of the American Mathematical Society 18(2005)711-734.

[12] T.H. Cormen, C.E. Leiserson, and R.L. Rivest, Introduction to Algorithms, MIT Press, 1990.