پیدا کردن کوتاه‌ترین مسیر روی سطوح نامنظّم مثلّث‌بندی شده‌ی وزن‌دار

محمّد قدسی
ghodsi@sharif.ac.ir

حمید ضرّابی‌زاده
zarrabi@ce.sharif.ac.ir

دانشکده‌ی مهندسی کامپیوتر
دانشگاه صنعتی شریف

چکیده

مسئله‌ی پیدا کردن کوتاه‌ترین مسیر یکی از مهم‌ترین مسئله‌های مطرح شده در زمینه‌ی هندسه‌ی محاسباتی و شاخه‌های دیگری هم‌چون الگوریتم‌های گراف، سیستم‌های اطّلاعات جغرافیایی (GIS)، برنامه‌ریزی حرکت در روبات‌ها و غیره می‌باشد. مسئله‌ی موردنظر ما حالت خاصّی از این مسئله است که هدف آن تعیین کوتاه‌ترین مسیر بین دو نقطه از یک سطح وزن‌دار در فضای سه بعدی است، به گونه‌ای که تمام مسیر روی سطح یک چندوجهی قرار گرفته باشد. در این مقاله پس از معرّفی دقیقِ مسئله و برخی از الگوریتم‌های مهمّ شناخته شده برای حلّ آن، الگوریتم‌های تقریبی نسبتاً ساده‌ای را برای به‌دست آوردن کوتاه‌ترین مسیر در سطوح نامنظم مثلث بندی شده‌ی وزن دار معرّفی می‌کنیم. سپس با بررسی و مقایسه‌ی نتایج به‌دست‌آمده از پیاده‌سازی این الگوریتم‌ها، پیش‌نهادهایی جهت بهبود سرعت آن‌ها ارائه خواهیم کرد.

کلیدواژه‌ها: هندسه‌ی محاسباتی، الگوریتم تقریبی، کوتاه‌ترین مسیر، سطوح نامنظّم مثلّث بندی شده (TIN).

۱ مقدّمه

تحقیقات اخیر در زمینه‌ی روباتیک، پیمایش زمین و اتوماسیون صنعتی، باعث پیش‌رفت تحقیقات در زمینه‌ی برنامه‌ریزی حرکت[۱] شده است. یک دسته از مسائل پایه‌ای در این زمینه، مسائل پیدا کردن کوتاه‌ترین مسیر هستند. این مسئله‌ها از مسائل پایه‌ای در هندسه‌ی محاسباتی و نظریّه‌ی گراف‌ها محسوب می‌شوند.

در این مقاله، به نوع خاصّی از این مسئله توجّه خواهیم کرد که هدف آن تعیین کوتاه‌ترین مسیر بین دو نقطه از یک سطح وزن‌دار در فضای سه بعدی است، به گونه‌ای که تمام مسیر روی سطح یک چندوجهی مورد نظر قرار گرفته باشد. سطح وزن‌دار سطحی است که به نواحی مختلف آن اعداد حقیقی غیر منفی‌ای نسبت داده شده است که نشان دهنده‌ی هزینه‌ی عبور از واحد طول در این ناحیه‌هاست. برای حالت بدون وزن این مسئله الگوریتم‌های با زمان اجرای از مرتبه‌ی چندجمله‌ای وجود دارد ولی در حالت کلّی و در فضای سه بعدی که در آن تعدادی مانع به شکل چندوجهی وجود دارند، این مسئله یک مسئله‌ی NP-Hard محسوب می‌شود.

در این مقاله ضمن بررسی کارهای انجام شده، به معرّفی چند الگوریتم تقریبی برای حلّ مسئله می‌پردازیم سپس با بررسی نتایج عملی حاصل از پیاده‌سازی هریک، به ارائه‌ی پیش‌نهادهایی جهت بهبود سرعت بهترین الگوریتم، هم از لحاظ عملی و هم از جنبه‌ی تئوری خواهیم پرداخت.

---

[۱] Motion Planning

Algorithm	Running Time
Mitchell and Papadimitriou	$O(n^{8}\log(\frac{nK}{N}))$
Lanthier et al.	$O(n^{5})$
Mata and Mitchell	$O(n^{3}K)$
Aleksandrov et al.	$O(n(K\log K)\log(nK\log K))$

جدول ۱: مقایسه‌ی الگوریتم‌های کوتاه‌ترین مسیر روی سطوح وزن‌دار

۲ تعاریف اولیه

سطحِ P که مساحت درون آن به n ناحیه‌ی بسته‌ی چندضلعی شکل تقسیم شده، به‌همراه دو نقطه‌ی s و t (مبدأ و مقصد) روی این سطح داده شده است؛ هدف پیدا کردن کوتاه‌ترین مسیر بین این دو نقطه روی سطح می‌باشد. بدون این‌که از کلّیت مسئله کاسته شود، می‌توانیم فرض کنیم که تمامی ناحیه‌ها به‌شکل مثلّث‌اند و نقاط s و t دو رأس از این مثلّث‌بندی هستند. اگر برای هر یک از رأس‌های مثلّث‌بندی، یک عدد به‌عنوان ارتفاع آن نقطه از سطح در نظر بگیریم، سطح حاصل یک تین (TIN[۲]) نامیده می‌شود.

به هر وجه f، یک عدد مثبتِ w_f به‌عنوان وزنِ آن وجه اختصاص داده شده است. در حقیقت این عدد برابر هزینه‌ی عبور از واحد طول در وجه می‌باشد. اگر فاصله‌ی اقلیدسی بین دو نقطه‌ی x و y را با $|xy|$ نمایش دهیم، فاصله‌ی وزن‌دار آن‌ها در یک وجه به وزن w_f برابر $w_f|xy|$ خواهد بود که این فاصله‌ی وزن‌دار را به‌صورت $\|xy\|$ نشان خواهیم داد. کوتاه‌ترین مسیر اقلیدسی بین نقاط s و t را با $\pi(s,t)$ و طول آن را با $|\pi(s,t)|$ نمایش می‌دهیم. همچنین، کوتاه‌ترین مسیر وزن‌دار بین نقاط s و t را با $\Pi(s,t)$ و طول آن را با $\|\Pi(s,t)\|$ نمایش خواهیم داد.

۳ بررسی کارهای انجام‌شده

با این که برای مسئله‌ی کوتاه‌ترین مسیر در سطوح وزن‌دار هیچ الگوریتم دقیقی وجود ندارد، در عوض الگوریتم‌های تقریبی گوناگونی برای یافتن کوتاه‌ترین مسیر روی سطوح وزن‌دار ارائه شده‌اند.

اولین بررسی کلّی در مورد مسئله‌ی ناحیه‌های وزن‌دار توسطMitchell و Papadimitriou [۴] صورت گرفته است. آن‌ها علاوه بر تعریف دقیق خصوصیات مربوط به کوتاه‌ترین مسیرها و مسیرهای بهینه در سطوح وزن‌دار، یک الگوریتم ϵ —تقریبی مبتنی بر روش دایسترای پیوسته ارائه نموده‌اند که دارای زمان اجرای $O(n^{8}L)$ می‌باشد. (منظور از L ضریب پیچیدگی بیتی مسئله‌است.)

پس از آن Mata و Mitchell [۳] الگوریتمی را ارائه کرده‌اند که در زمان $O(kn^{3})$ کوتاه‌ترین مسیر تقریبی را تعیین می‌کند. (k یکی از پارامترهای ورودی الگوریتم است). با کم و زیاد کردن k در این الگوریتم می‌توان دقّت نتیجه‌ی نهایی را کاهش یا افزایش داد.

در سال ۱۹۹۷ Lanthier، Maheshwari و Sack [۲] یک الگوریتم تقریبی ارائه نمودند که اساس کار این الگوریتم ساخت یک گراف از روی سطح مورد نظر به‌وسیله‌ی قرار دادن تعدادی نقطه‌ی کمکی روی یال‌ها می‌باشد. زمان اجرای این الگوریتم در بدترین حالت $O(n^{5})$ و فضای حافظه‌ی موردنیاز آن $O(n^{3})$ است. همچنین وزن مسیر تقریبیِ به‌دست آمده از این الگوریتم، حدّاکثر به اندازه‌ی $|W|L|$ از وزن مسیر بهینه بیش‌تر خواهد بود. (منظور از L، بزرگ‌ترین یال و منظور از W بیش‌ترین وزن وجه‌ها است.)

Aleksandrov و بقیه [۱] در سال ۱۹۹۸ توانستند با استفاده از ایده‌ی به‌کار رفته در الگوریتم Lanthier، یک الگوریتم جدید ϵ —تقریبی برای مسئله ارائه کنند. زمان اجرایی این الگوریتم $O(nm\log mn + nm^{2})$ است که در آن منظور از m، متوسط تعداد نقاط کمکی است که توسّط الگوریتم، روی یال‌ها افزوده می‌شود.

در جدول ۱، الگوریتم‌های تقریبی موجود به‌همراه زمان اجرایی هر یک نشان داده شده است. برای این‌که بتوان زمان اجرای این الگوریتم‌ها را به‌راحتی با یک‌دیگر مقایسه نمود، تمامی زمان‌های اجرا بر اساس یک پارامتر $K = O(\frac{N^{2}W}{\epsilon w})$ بازنویسی شده‌اند. در این پارامتر، N برابر بزرگ‌ترین مختصات صحیح نقاط سطح، W و w به‌ترتیب برابر بیش‌ترین و کم‌ترین وزن صحیح ناحیه‌ها و ϵ درجه‌ی دقّتِ تعریف‌شده توسّط کاربر است.

۴ الگوریتم‌های تقریبی

در این بخش به توضیح چند الگوریتم تقریبی برای یافتن کوتاه‌ترین مسیر بین دو نقطه روی سطح تین می‌پردازیم.

[۲] Triangulated Irregular Network

تمامی این الگوریتم‌ها پیاده‌سازی شده و نتایج حاصل از عمل‌کرد آن‌ها در بخش ۶ مقایسه و ارزیابی گردیده است. خصوصیت مشترک این الگوریتم‌ها این است که همگی برای تعیین کوتاه‌ترین مسیر تقریبی بین دو رأس، تین موردنظر را به یک گراف تبدیل کرده و از الگوریتم‌های متداول یافتن کوتاه‌ترین مسیر در گراف برای یافتن کوتاه‌ترین مسیر بین نقاط مبدأ و مقصد استفاده می‌نمایند.

۱.۴ استفاده از گراف ساده‌ی تقریب

شاید ساده‌ترین کاری که برای تعیین کوتاه‌ترین مسیر بین دو رأس به‌طور تقریبی می‌توان انجام داد، این باشد که هر رأس از گراف G را متناظر با یکی از رأس‌های تین T در نظر بگیریم و بین هر دو رأس از G که رأس‌های متناظرشان در T به هم متّصل باشند، یک یال با وزنی معادل با فاصله‌ی اقلیدسی آن‌دو ضرب‌در کم‌ترین وزن دو وجه مجاور یال قرار دهیم. به این ترتیب می‌توانیم $\Pi'(s,t)$ را با استفاده از الگوریتم دایسترا به‌دست آوریم. این روش یال‌های مسیر تقریبی را به یال‌های تین اوّلیّه محدود می‌کند و خروجی آن بسته به نحوه‌ی مثلّث‌بندی تین ورودی می‌تواند بسیار بد باشد.

۲.۴ روش نقاط کمکی ثابت

در این روش، خطای زیاد روش قبلی را با استفاده از رأس‌ها و یال‌های جدیدی که به گراف G می‌افزاییم و آن‌ها را رأس‌ها و یال‌های کمکی[۳] می‌نامیم، تا حدّ خوبی بهبود می‌بخشیم. برای این کار، روی هر یال از T، m نقطه را به‌طور یک‌نواخت اضافه می‌کنیم (m یک عدد مثبت دل‌خواه است). سپس به‌ازای هر وجه، تمام نقاط کمکی قرار گرفته روی یال‌های آن را دوبه‌دو به یک‌دیگر وصل می‌کنیم. وزن هر یال جدید، حاصل‌ضرب فاصله‌ی اقلیدسی بین رأس‌های دو سر آن و وزن وجهی می‌باشد که در آن قرار گرفته است. برای تعیین کوتاه‌ترین مسیر تقریبی روی T، از یکی از الگوریتم‌های کوتاه‌ترین مسیر روی G استفاده می‌کنیم و مسیر حاصل را دوباره روی T منتقل می‌کنیم.

قضیّه‌ی ۱. با استفاده از روش نقاط کمکی ثابت می‌توان مسیر تقریبی وزن‌دار $\Pi'(s,t)$ را به عنوان تقریبی برای کوتاه‌ترین مسیر $\Pi(s,t)$ به‌دست آورد به‌طوری که $|\Pi'(s,t)| \leq |\Pi(s,t)| + W.|L|$. در این رابطه $|L|$ طول بلندترین یال در تین مورد نظر و W بیش‌ترین وزن در بین وزن وجه‌های تین می‌باشد. [۲]

[۳] Steiner points and edges

۳.۴ روش بازه‌های ثابت

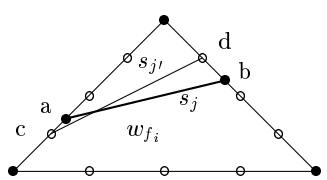
یکی از مشکلات روش قبل این است که در صورت یکنواخت نبودن طول یال‌ها، طول بازه‌های ایجاد شده روی یال‌ها نیز غیریکنواخت خواهد بود و این مسئله بر دقت مسیر ایجاد شده تأثیر نامطلوبی دارد.

در روش بازه‌های ثابت، تعداد نقاط کمکی اضافه شده به هر یال را طوری انتخاب کنیم که طول بازه‌ها ایجادشده روی یال‌ها تقریباً مساوی شود. هر چند این روش در بدترین حالت همانند روش قبلی عمل می‌کند، ولی به‌طور متوسّط، این روش تعداد رأس‌ها، یال‌ها و در نتیجه زمان اجرای الگوریتم کوتاه‌ترین مسیر را تا حدّ زیادی کاهش می‌دهد.

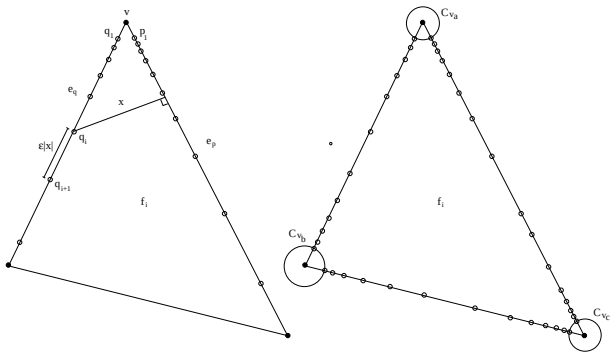
۴.۴ روش بازه‌های نمایی

الگوریتم بازه‌های نمایی با دریافت یک تین وزن‌دار و عدد مثبت ϵ از ورودی، مسیر تقریبی $\Pi'(s,t)$ را طوری پیدا می‌کند که $\frac{\|\Pi'(s,t)\|}{\|\Pi(s,t)\|} \leq 1+\epsilon$.

این الگوریتم تا حدّی شبیه الگوریتم معرّفی شده در بخش ۲.۴ عمل می‌کند و از نقاط کمکی و ساختن گرافی متناظر با تین اوّلیّه استفاده می‌کند، با این تفاوت که این الگوریتم به‌جای این که تعداد ثابتی نقطه‌ی کمکی را روی هر یال با فاصله‌های مساوی پخش کند، نقاط را با فاصله‌های نمایی (از رأس‌های یال) پخش می‌نماید. بدیهی است برای این که خطای مسیر تقریبی حاصل از الگوریتم از ϵ برابر کوتاه‌ترین مسیر کم‌تر باشد، کافی است طول هر یک از پاره‌خط‌های تشکیل دهنده‌ی مسیر تقریبی از $1+\epsilon$ برابر طول پاره‌خطّ متناظری در کوتاه‌ترین مسیر کم‌تر باشد. همان طور که در شکل ۱ مشاهده می‌شود، تفاوت میان s'_i و s_i حدّاکثر به اندازه‌ی $ac+bd$ است. اگر بخواهیم s'_i از $1+\epsilon$ برابر s_i کوتاه‌تر باشد، کافی است که $\frac{ac+bd}{s_i} \leq \epsilon$. به این ترتیب واضح است که هر چه s_i بزرگ‌تر باشد، می‌توانیم $ac+bd$ را بزرگ‌تر انتخاب کنیم، بدون آن که خطای الگوریتم از ϵ بیش‌تر شود. در بخش بعد به تشریح دقیق‌تر این الگوریتم خواهیم پرداخت.



شکل ۱: پاره‌خطی از یک کوتاه‌ترین مسیر تقریبی



(الف) (ب)

شکل ۲: الف) افزودن نقاط کمکی روی یال‌ها ب) ادغام نقاط کمکی روی یک یال

۵ الگوریتم بازه‌های نمایی

فرض کنید v یک رأس از سطح مثلث‌بندی‌شده‌ی T باشد. h_v را به‌صورت کوتاه‌ترین فاصله از v تا مرز حاصل از اجتماع وجه‌های مجاور آن تعریف می‌کنیم. یک سطح کروی چندوجهی C_v که آن را یک «گوی»[۴] حول v خواهیم نامید، به‌صورت زیر تعریف می‌شود: فرض کنید به‌ازای یک مقدار مثبت ϵ ، $r_v = \epsilon h_v$. r را به‌عنوان کوچک‌ترین r_v روی تمام v ها تعریف می‌کنیم. اگر vuw یک سطح مثلثی مجاور v باشد، u' (w') را یک نقطه روی vu (vw) با فاصله‌ی r_v از v درنظر می‌گیریم. بدین ترتیب یک زیروجه مثلثی‌شکل $vu'w'$ به‌دست می‌آید. گوی C_v حول v از اجتماع تمام این زیروجه‌های مجاور v به‌دست می‌آید.

واضح است که فاصله‌ی بین هر دو گوی C_{v_a} و C_{v_b} بزرگ‌تر از $(۲\epsilon - ۱)h_{v_a}$ است، لذا از این پس جهت سادگی کار، ϵ را کوچک‌تر از $\frac{۱}{۴}$ در نظر می‌گیریم. فرض کنید θ_v کوچک‌ترین زاویه‌ی (محاسبه‌شده در فضای سه‌بعدی) بین هر دو یال مجاور v در T باشد. θ را نیز به‌عنوان کوچک‌ترین θ_v تعریف می‌کنیم.

۱.۵ الگوریتم محاسبه‌ی گراف

به‌ازای هر رأس v از وجه f_i اعمال زیر را انجام می‌دهیم: فرض کنید e_p و e_q رأس‌های مجاور v از وجه f_i هستند. ابتدا دو نقطه‌ی کمکی روی e_p و e_q در فاصله‌ی r_v از v قرار می‌دهیم و آن‌ها را به‌ترتیب $p_۱$ و $q_۱$ نام‌گذاری می‌کنیم. طبق تعریف داریم: $|\overline{vp_۱}| = |\overline{vq_۱}| = r_v$. اگر $\theta_v < \frac{\pi}{۲}$، تعریف می‌کنیم: $\delta = ۱ + \epsilon \sin\theta_v$ و درغیراین‌صورت $\delta = ۱ + \epsilon$. نقاط کمکی $p_۲, p_۳, \ldots, p_{\mu_p - ۱}$ را روی e_p طوری قرار می‌دهیم که $|\overline{vp_j}| = r_v \delta^{j-۱}$ و $\mu_p = \log_\delta(|e_p|/r_v)$. به همین ترتیب نقاط کمکی $q_۲, q_۳, \ldots, q_{\mu_q - ۱}$ را روی e_q با شرط $\mu_q = \log_\delta(|e_q|/r_v)$ اضافه می‌کنیم. بدین ترتیب مجموعه‌هایی از نقاط کمکی روی یال‌های e_p و e_q ایجاد می‌شوند. (شکل ۲—الف را مشاهده نمایید.)

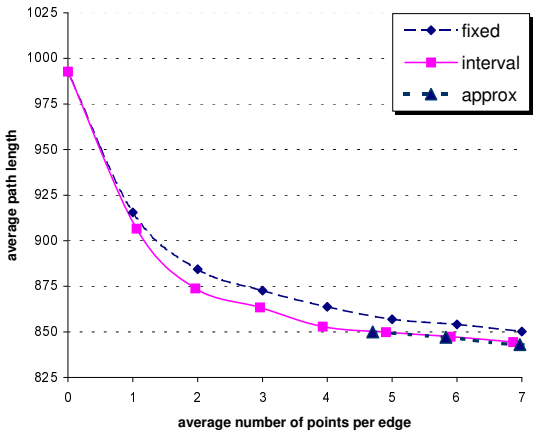
اگر کوتاه‌ترین فاصله‌ی بین یک نقطه‌ی a تا یک یال e را $d(a,e)$ بنامیم، این فاصله معادل با یک خط عمودی از a به e خواهد بود.

ادّعای ۱ . به‌ازای $i < \mu_q$ و $j < \mu_p$ داریم: $|\overline{q_i q_{i+۱}}| = \epsilon \cdot d(q_i, e_p)$ و $|\overline{p_j p_{j+۱}}| = \epsilon \cdot d(p_j, e_q)$. [۱]

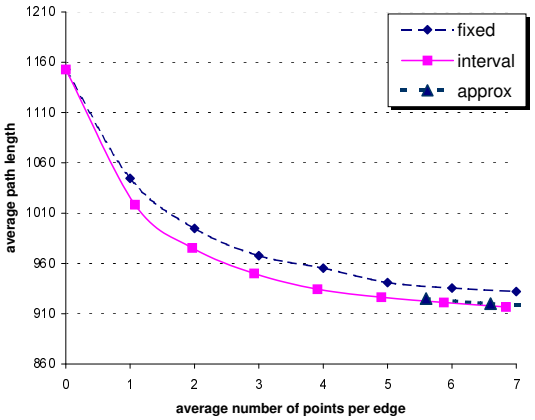
از آن‌جایی‌که نقاط کمکی را براساس کم‌ترین زاویه‌ی θ_v حول v اضافه کرده‌ایم، به یک «جبهه‌ی موج موازی هم‌مرکز» از نقاط کمکی روی یال‌های مجاور v خواهیم رسید. با توجّه به این که عملیات افزدون نقاط کمکی را برای هر یک از رأس‌های f_i به‌طور مجزّا انجام داده‌ایم، درنتیجه روی هر یال f_i با دو مجموعه‌ی هم‌پوشان از نقاط کمکی روبرو خواهیم شد. برای رفع این هم‌پوشانی، تعدادی از نقاط کمکی روی هر یال را طی یک مرحله‌ی «پیرایش» حذف می‌کنیم. به‌ازای هر دو مجموعه از نقاط کمکی که از رأس‌های دو سر یال e نشأت گرفته‌اند، نقطه‌ای را روی e مشخص خواهیم کرد که در آن نقطه، طول بازه‌ی به‌وجود آمده از هر دو مجموعه برابر باشد و سپس تمام بازه‌های بزرگ‌تر را حذف می‌کنیم. بدین‌ترتیب یک بازه از یک مجموعه درصورتی حذف خواهد شد که یک بازه‌ی کوچک‌تر از مجموعه‌ی دیگر با آن هم‌پوشانی ایجاد کرده باشد. (شکل ۲—ب را مشاهده نمایید.)

قضیّه‌ی ۲ . کوتاه‌ترین مسیر ϵ—تقریبی وزن‌دار در یک تین با n وجه، در زمان $O(mn \log mn + nm^۲)$ قابل محاسبه است. ($m = \log_\delta(|L|/r)$ و $\delta = ۱ + \epsilon \cdot \sin\theta$). [۱]

[۴] Sphere



(الف)



(ب)

شکل ۳: نتایج آزمون (الف) تین ۵۰۰ رأسی (ب) تین ۱۰۰۰ رأسی

۶ بررسی نتایج

نتایج حاصل از اجرای الگوریتم‌ها بر روی دو تین ۵۰۰ و ۱۰۰۰ رأسی به‌صورت نمودارهای خطی در شکل ۳ نمایش داده شده است. برای این که بتوان نتایج حاصل از الگوریتم‌های مختلف را با یک‌دیگر مقایسه کرد، این نمودارها بر اساس متوسط تعداد نقاط کمکی ایجاد شده روی هر یال رسم شده‌اند. بدیهی است که اگر متوسط تعداد نقاط کمکی ایجاد شده روی یال‌ها در دو آزمونِ مختلف یکسان باشد، درنتیجه تعداد کل یال‌ها در گراف ایجاد شده برابر خواهد بود و بنابراین زمان اجرای هر کدام از الگوریتم‌ها که معادل زمان اجرای الگوریتم دایسترا بر روی گراف حاصل است، برابر خواهد بود. لذا نمودارهایی که در شکل ۳ نشان داده شده‌اند، بیان‌گر میانگین طول کوتاه‌ترین مسیرِ به‌دست‌آمده از الگوریتم‌های مختلف، در زمان‌های یکسان می‌باشند.

نکته‌ی مهمی که در نمودارها به چشم می‌خورد این است که مسیر تقریبیِ حاصل از تمامی الگوریتم‌های بررسی‌شده، با افزودن تنها چند نقطه‌ی کمکی (در این‌جا ۷ نقطه) به‌سرعت به مسیر دقیق همگرا می‌شوند که این مطلب نسبت به کران بالایی که از تحلیل نظری به‌دست می‌آید، به‌طور قابل ملاحظه‌ای بهتر است.

۷ بهبود الگوریتم ϵ —تقریبی

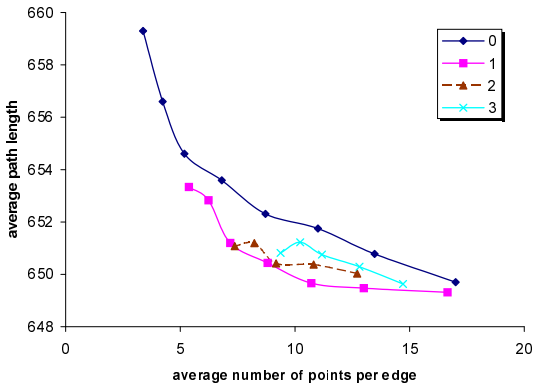
در این قسمت سعی خواهیم کرد با اعمال روش‌هایی، زمان اجرای الگوریتم بازه‌های نمایی را، هم از دو جنبه‌ی عملی و نظری بهبود ببخشیم.

۱.۷ استفاده از نقاط کمکی در گوی

در تعریف الگوریتم بازه‌های نمایی، در بخش ۵، دور هر رأس v از سطح مثلث‌بندی‌شده‌ی T، یک سطح کروی چندوجهی C_v که آن را یک «گوی» حول v نامیدیم، با شعاع $r_v = \epsilon h_v$ تعریف نمودیم. همان‌طور که دیدیم، تمامی نقاط کمکی قرار داده شده روی یال‌ها، خارج از این گوی قرار می‌گرفتند. همان‌طور که در شکل ۲ دیده می‌شود، به‌ازای مقادیر بزرگ ϵ، این‌کار باعث ایجاد تکه یال‌های بزرگ در گوی‌ها شده و درنتیجه دقّت مسیر به‌دست‌آمده، نسبت به مسیر مشابه در روش بازه‌های ثابت کاهش می‌یابد.

به‌نظر می‌رسد که قرار دادن تعدادی نقطه‌ی کمکی روی یال‌های قرارگرفته در درون محدوده‌ی گوی‌ها، بتواند در ایجاد مسیرهای بهتر (در زمان یکسان) مؤثّر باشد. برای بررسی این موضوع، به‌ترتیب تعداد یک، دو و سه نقطه‌ی کمکی روی یال‌های درون گوی قرار داده، نتایج را روی تین‌های مختلف آزمایش نمودیم. نتیجه‌ی این آزمایش به‌ازای یک تین ۵۰۰ رأسی در نمودار شکل ۴ نشان داده شده است.

همان‌طور که در نمودار دیده می‌شود، نتیجه‌ی اجرای الگوریتم بازه‌های نمایی با افزودن این نقاط بر روی یال‌های درون گوی بهبود می‌یابد. از طرفی هرچه تعداد نقاط کمکی روی یال‌های درون گوی افزایش یابد، میانگین تعداد نقاط روی هر یال و درنتیجه زمان اجرای الگوریتم افزایش می‌یابد. بنابراین باید مصالحه‌ای بین دقّت و سرعت برقرار نماییم. آزمایش‌های تجربی نشان می‌دهد که افزودن تنها یک نقطه‌ی کمکی به یال‌های درون هر گوی، بهترین نتیجه را از لحاظ سرعت و دقّت محاسبه‌ی مسیر در بر خواهد داشت. (شکل ۴ را ملاحظه نمایید.)



شکل ۴: نتیجه‌ی اجرای الگوریتم بازه‌های نمایی با افزودن ۰، ۱، ۲ و ۳ نقطه‌ی کمکی روی یال‌های درون هر گوی

۲.۷ کاهش حجم گراف تقریب

یکی دیگر از راه‌های بهبود الگوریتم این است که بتوانیم با حفظ ویژگی ϵ —تقریبی بودن آن، مرتبه‌ی زمان اجرای آن را از لحاظ نظری پایین بیاوریم. این کار با کاهش تعداد یال‌های کمکی قرار داده شده درونِ هر وجه گراف امکان‌پذیر است. همان‌طور که می‌دانیم، در حالت عادی هر وجه مثلثی شکل از تین اولیه در گراف تقریب، به یک گراف کامل تبدیل می‌گردد، به‌عبارتی دیگر تمامی نقاط کمکی قرار گرفته روی یال‌های یک وجه، دوبه‌دو به هم متصل می‌شوند. در مرجع [۵] نشان داده‌ایم که برخی از یال‌های گراف G را می‌توان به‌گونه‌ای حذف کرد که در گراف G' به‌دست آمده، تنها $O(nm \log mn)$ یال باقی بماند و درنتیجه زمان اجرای کلّ الگوریتم به $O(nm \log mn)$ کاهش یابد. البته با این کار، ضریب تقریب اندکی افزایش می‌یابد.

در این روش، هر یال e از G، توسّط یک مسیر e' از G' قابل تقریب است، به‌نحوی که $\|e'\|$ یک ϵ —تقریب از $\|e\|$ باشد. توضیحات دقیق‌تر مربوط به این روش در مرجع [۵] موجود است.

۸ نتیجه‌گیری

در این مقاله، مسئله‌ی یافتن کوتاه‌ترین مسیر بین دو نقطه از سطح یک تین وزن‌دار را مورد بررسی قرار دادیم و ضمن بررسی کارهای انجام شده در این زمینه، به معرّفی چند الگوریتم تقریبی برای حلّ مسئله پرداخته، نتایج عملی حاصل از پیاده‌سازی هریک را مقایسه و ارزیابی نمودیم. همچنین با ارائه‌ی پیش‌نهادهایی جهت بهبود بهترین الگوریتم موجود، سرعت اجرای آن را هم از لحاظ عملی و هم از جنبه‌ی تئوری بهبود بخشیدیم. به‌نظر می‌رسد که استفاده از روش‌های پردازش موازی نیز می‌تواند در به‌دست‌آوردن سرعت بیش‌تر مؤثّر باشد. موازی نمودن الگوریتم در مرجع [۵] مورد بررسی بیش‌تر قرار گرفته است.

مراجع

[1] L. Aleksandrov, M. Lanthier, A. Maheshwari and J. R. Sack, *"An ϵ - Approximatino Algorithm for Weighted Shortest Paths on Polyhedral Surfaces"*, SWAT 98, Stockholm, Sweden, July 1998.

[2] M. Lanthier, *"Shortest Path Problems on Polyhedral Surfaces"*, Ph.D. Thesis in progress, School of Computer Science, Carleton University, Ottawa, Canada, 1998.

[3] C. S. Mata and J. S. B. Mitchell, ,*"A New Algorithm for Computing Shortest Paths in Weighted Planar Subdivisions"*, Proc. 13th Annu. ACM Sympos. Computational Geometry, 1997, pp. 264-273.

[4] J. S. B. Mitchell and C. H. Papadimitriou, *"The Weighted Region Problem: Finding Shortest Paths Through a Weighted Planar Subdivision"*, Journal of the ACM, 38, January 1991, pp. 18-73.

[۵] حمید ضرّابی‌زاده، *«پیدا کردن موازی کوتاه‌ترین مسیر در سطوح نامنظّم مثلّث‌بندی‌شده‌ی وزن‌دار»*، پایان‌نامه‌ی کارشناسی ارشد، دانشکده‌ی مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران، آبان ماه ۱۳۷۹.

[۶] یاشار گنجعلی، *«تعیین کوتاه‌ترین مسیر تقریبی در سطوح نامنظم مثلث بندی شده‌ی وزن‌دار»*، پایان‌نامه‌ی کارشناسی، دانشکده‌ی مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران، شهریور ماه ۱۳۷۸.