
بسم الله الرحمن الرحیم

دانشکده علوم ریاضی

پایان‌نامه کارشناسی ارشد بهینه سازی

یافتن کوتاهترین مسیر در صفحه برای حرکت ربات با حذف موانع

نگارنده: محمد حسین تیموری

اساتید راهنما

دکتر جعفر فتحعلی
دکتر ابوالفضل پورعیدی

خرداد ۱۳۹۹

تقدیم به:

این پایان نامه را تقدیم می‌کنم به:

پدر و مادر عزیز و مهربانم
که در سختی‌ها و دشواری‌های زندگی همواره یاوری دلسوز و فداکار و پشتیبانی محکم و مطمئن برایم بوده‌اند.

و به همسرم
به پاس قدردانی از قلبی آکنده از عشق و معرفت که محیطی سرشار از سلامت و امنیت و آرامش و آسایش برای من فراهم آورده است

و در آخر
آموزگارانی که برایم، زندگی، بودن و انسان بودن را معنا کردند

حال این برگ سبزی است تحفه درویش تقدیم آنان...

سپاس‌گزاری...

از اساتید گرامی جناب آقایان دکترفتحعلی و دکتر پورعیدی بسیار سپاسگذارم به دلیل یاری‌ها و راهنمایی‌های بی‌چشمداشت ایشان که بسیاری از سختی‌ها را برایم آسان نمودند و بدون راهنمایی‌های ایشان تامین این پایان نامه بسیار مشکل مینمود.

محمد حسین تیموری
خرداد ۱۳۹۹

تعهد نامه

اینجانب **محمد حسین تیموری** دانشجوی کارشناسی ارشد رشته **ریاضی کاربردی** دانشکده **علوم ریاضی** دانشگاه صنعتی شاهرود، نویسنده پایان‌نامه با عنوان **یافتن کوتاهترین مسیر در صفحه برای حرکت ربات با حذف موانع** ، تحت راهنمایی **جعفر فتحعلی** و **ابوالفضل پورعیدی** متعهد می‌شوم:

- تحقیقات در این پایان‌نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است.
- در استفاده از نتایج پژوهش‌های دیگر پژوهش‌گران، به مرجع مورد استفاده استناد شده است.
- مطالب این پایان‌نامه، تا کنون توسط خود، یا فرد دیگری برای دریافت هیچ نوع مدرک یا امتیازی در هیچ‌جا ارایه نشده است.
- حقوق معنوی این اثر، به دانشگاه صنعتی شاهرود تعلق دارد، و مقالات مستخرج با نام ‘‘ دانشگاه صنعتی شاهرود ’’ یا “ Shahrood University of Technology ” به چاپ خواهد رسید.
- حقوق معنوی تمام افرادی که در به‌دست آوردن نتایج اصلی پایان‌نامه تاثیرگذار بوده‌اند، در مقالات مستخرج از پایان‌نامه رعایت می‌گردد.
- در تمام مراحل انجام این پایان‌نامه، در مواردی که از موجود زنده (یا بافت‌های آنها) استفاده شده است، ضوابط و اصول اخلاقی رعایت شده است.
- در تمام مراحل انجام این پایان‌نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته (یا استفاده شده است)، اصل رازداری و اصول اخلاق انسانی رعایت شده است.

محمد حسین تیموری
خرداد ۱۳۹۹

مالکیت نتایج و حق نشر

- تمام حقوق معنوی این اثر و محصولات آن (مقالات مستخرج، کتاب، برنامه‌های رایانه‌ای، نرم‌افزارها و تجهیزات ساخته شده) متعلق به دانشگاه صنعتی شاهرود می‌باشد. این مطلب باید به نحو مقتضی، در تولیدات علمی مربوطه ذکر شود.
- استفاده از اطلاعات و نتایج موجود در این پایان‌نامه بدون ذکر منبع مجاز نمی‌باشد.

چکیده

در این پایان نامه با گراف رؤیت پذیری آشنا خواهیم شد که برای محاسبه طول کوتاهترین مسیر بین دو نقطه در فضای آزاد بین موانع مورد استفاده قرار می‌گیرد.
سپس انواع مختلف کوتاهترین مسیر هندسی را در داخل یک چندضلعی ساده، که در آن اجازه داریم بخشی از مسیر به خارج از چندضلعی برود، مطالعه می‌کنیم. فرض کنید P یک چندضلعی ساده باشد که از n رأس تشکیل شده باشد و فرض کنید s و t دو نقطه در P باشند. برای یک عدد صحیح $k \geq ۰$، یک مسیر kـحذفی از s به t مسیری است که s و t را به هم وصل کند به طوری که تقاطع آن با فضای بیرونی P حداکثر k پاره‌خط است. از نظر تعداد پاره‌خط‌های مسیر در داخل P، هیچ محدودیتی وجود ندارد. هدف این است که مسیری با حداقل طول اقلیدسی در بین تمام مسیرهای $(\leq k)$ـحذفی ممکن از s تا t محاسبه شود. در این پایان نامه، این مسئله را برای $k = ۱$ مطالعه می‌کنیم و الگوریتمی را پیشنهاد می‌کنیم که کوتاهترین مسیر یکـحذفی را در زمان $O(n^۳)$ محاسبه می‌کند. نشان می‌دهیم که برای چندضلعی‌های مستطیل، حداقل طول مسیر یکـحذفی می‌تواند در زمان $O(n \log n)$ محاسبه شود.
همینطور طریقه یافتن نقشه کوتاهترین مسیر در میان h مانع محدب را بررسی می‌کنیم، طوریکه اجازه حذف حداکثر k مانع را داشته باشیم، برای $k \leq h$. با توجه به نقطه ثابت داده شده s، نحوه ساخت یک نقشه را با نام کوتاهترین kـمسیر نشان می‌دهیم، بطوریکه همه مقصدها در ناحیه‌ای از نقشه،دارای یک کوتاهترین مسیر ترکیبیاتی یکسان را که از بین حداکثر k مانع عبور خواهند کرد، هستند. محدودیت $\Theta(kn)$ را برای اندازه این نقشه اثبات می‌کنیم که می‌تواند در مدت زمان $O(k^۲ n \log n)$ محاسبه شود، که در آن n تعداد کل رئوس موانع است.

کلمات کلیدی: کوتاهترین مسیر، چندضلعی ساده، چندضلعی مستطیلی، الگوریتم دایکسترا، عبور از موانع، رؤیت پذیری

فهرست مطالب

فهرست تصاویر

فهرست جداول

فصل ۱

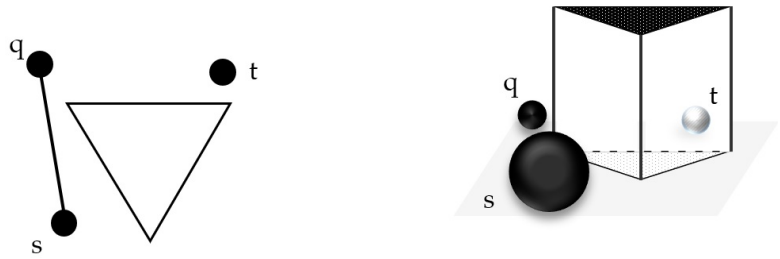
مفاهیم اولیه و پیش‌نیازها

در این فصل قرار است که با مفاهیم اولیه کوتاهترین مسیر بین دو نقطه در صفحه با وجود موانع محدب آشنا شویم. همینطور مفهوم گراف رؤیت پذیری و الگوریتم‌هایی که در این امر می‌توانند به ما کمک کنند از قبیل الگوریتم دایکسترا، الگوریتم پیوسته دایکسترا و نیز نمادهایی مجانبی که در رشد توابع غیرمنفی استفاده خواهیم کرد را معرفی می‌کنیم.

۱.۱ گراف رؤیت پذیری

فرض کنید S مجموعه‌ای شامل تعدادی چندضلعی جدا از هم به عنوان مانع در صفحه باشد و P_{start} و P_{goal} دو نقطه در بین چندضلعی‌ها باشند. در این بخش می‌خواهیم نقشه‌ای برای S بسازیم که با استفاده از آن می‌توانیم کوتاهترین مسیر بین هر دو نقطه P_{start} و P_{goal} را بدست آوریم. این نقشه را گراف رؤیت پذیری S می‌نامیم که آن را با نماد $\mathcal{G}_{\text{vis}}(S)$ نشان می‌دهیم. گره‌های $\mathcal{G}_{\text{vis}}(S)$ رأس‌های S است، و بین رأس های v و w یال وجود دارد، اگر آنها بتوانند یکدیگر را ببینند. به این معنی که پاره‌خط $\overline{vw}$ از فضای داخلی مانعی در S، عبور نمی‌کند. دو رأسی که بتوانند همدیگر را ببینند(متقابلا) دو رأس رؤیت پذیر گوییم، و پاره‌خط اتصال دهنده آنها یال رؤیت پذیر نامیده می‌شود. توجه داشته باشید که نقاط انتهایی یال یک مانع همیشه یکدیگر را می‌بینند. از این رو، یال‌های مانع زیرمجموعه‌ای از یالهای $\mathcal{G}_{\text{vis}}(S)$ را تشکیل می‌دهند. با توجه به شکل ۱.۱ رأس q از رأس s قابل رؤیت بوده زیرا می‌توان آنها را با یک یال به هم وصل کرد، همینطور رأس t برای رأس s

قابل رؤیت نیست چون نمی‌توان آنها را با یک یال به هم وصل کرد.



شکل ۱.۱: گراف رؤیت‌پذیری

۲.۱ الگوریتم دایکسترا

در نظریه گراف، الگوریتم دیکسترا [۱] یکی از الگوریتم‌های پیمایش گراف است که توسط دانشمند هلندی علوم رایانه، اِدْسْخِر دیْکْسْترا در سال ۱۹۵۹ ارائه شد. این الگوریتم یکی از الگوریتم‌های پیمایش گراف است که مسئلهٔ کوتاهترین مسیر از مبدأ واحد را برای گراف‌های وزن‌داری که یال با وزن منفی ندارند، حل می‌کند و در نهایت با ایجاد درخت کوتاه‌ترین مسیر، کوتاه‌ترین مسیر از مبدأ به همهٔ رأس‌های گراف را به دست می‌دهد. همچنین می‌توان از این الگوریتم برای پیدا کردن کوتاه‌ترین مسیر از مبدأ تا رأس مقصد به این ترتیب بهره جست که در حین اجرای الگوریتم به محض پیداشدن کوتاه‌ترین مسیر از مبدأ به مقصد، الگوریتم را متوقف کرد. الگوریتم دیکسترا یکی از الگوریتم‌های مورد استفاده برای محاسبه کوتاه‌ترین مسیر تک منبع [۲] است. در صورتی که گراف یال با وزن منفی داشته باشد، این الگوریتم درست کار نمی‌کند.

۱.۲.۱ روند الگوریتم دایکسترا

روند الگوریتم دایکسترا مطابق زیر می‌باشد:

۱. انتخاب راس مبدا

۲. مجموعهٔ S شامل رئوس گراف، معین می‌شود. در شروع، این مجموعه تهی بوده و با پیشرفت الگوریتم، این مجموعه رئوسی که کوتاه‌ترین مسیر به آن‌ها یافت شده‌است را در بر می‌گیرد.

۳. راس مبدأ با اندیس صفر را در داخل S قرار می‌دهد.

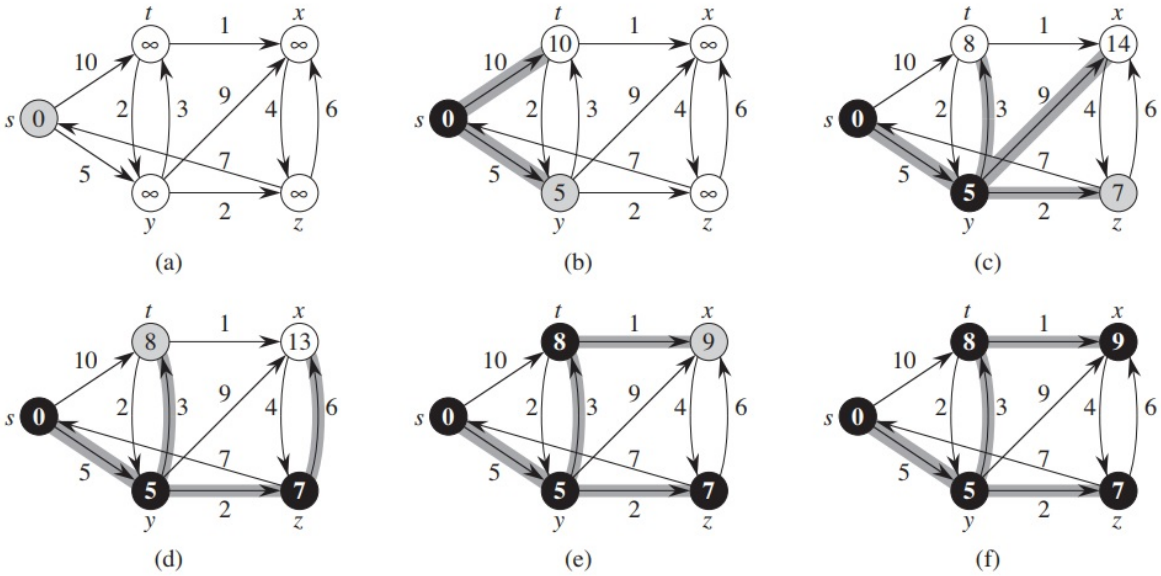
۴. برای رئوس خارج از S هم باید یک اندیس در نظر گرفت که اندیس آن از مجموع وزن یال و اندیس راس قبلی بدست می‌آید.

[1] Dijkstra's algorithm
[2] single-source shortest path

۵. از رئوس خارج مجموعه، راسی با کمترین اندیس انتخاب شده و به مجموعهٔ S اضافه می‌گردد.

۶. این کار را دوباره از مرحلهٔ ۴ ادامه داده تا راس مقصد وارد مجموعهٔ S شود.

در پایان اندیس بدست آمده در راس مقصد، نشان دهندهٔ مسافت بین مبدأ و مقصد می‌باشد.(اگر رأس مقصد دارای اندیس نباشد به این معنی است که بین رأس مبدأ و رأس مقصد هیچ مسیری وجود ندارد و در اصطلاح دو رأس مبدأ و مقصد را جدا از هم می‌نامیم).
همچنین برای پیدا کردن مسیر می‌توان اندیس دیگری برای هر راس در نظر گرفت که نشان دهندهٔ راس قبلی در مسیر طی شده باشد. بدین ترتیب پس از پایان اجرای الگوریتم، با دنبال کردن رئوس قبلی از مقصد به مبدا، کوتاه‌ترین مسیر بین دو نقطه نیز یافت می‌شود.



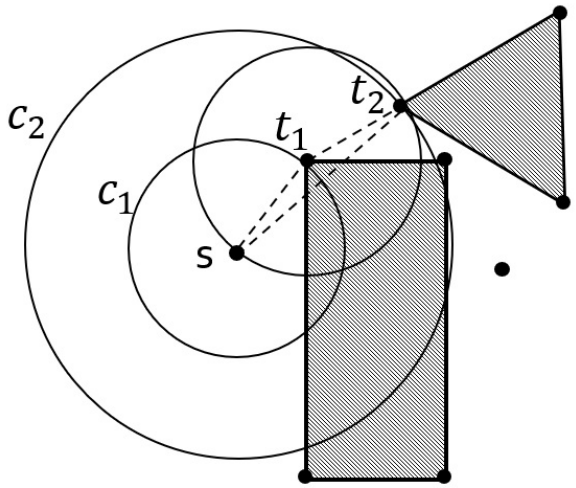
شکل ۲.۱: الگوریتم دایکسترا، رئوس مشکی به عنوان رئوس انتخابی و رئوس سفید، رئوس در صف انتظار هستند.

به عنوان مثال در شکل ۲.۱[۸]، در قسمت (a) رأس s به عنوان رأس ابتدایی انتخاب شده است. در قسمت (b) دو رأس y, t با توجه به ساختار گراف در این مرحله در نظر گرفته می‌شوند که با توجه به وزن یال‌ها، رأس y انتخاب می‌شود. همینطور در قسمت (c) برای رأس y نیز سه رأس در نظر گرفته می‌شود اما به دلیل داشتن فاصله کمتر از رأس s، رأس z انتخاب می‌شود. به همین روش پیش می‌رویم تا آخر که کوتاهترین فاصله هر رأس از مبدأ بدست آید.

۳.۱ الگوریتم پیوسته دایکسترا

مطالب این بخش از مقاله [۱۹] الگوریتم بهینه برای کوتاهترین مسیرهای اقلیدسی در صفحه انتخاب شده است. در این الگوریتم، جبهه موج از یک نقطه به عنوان مبدأ در حضور موانع چندضلعی، منتشر

می‌شود. خط مقدم موج زمان t شامل تمام نقاط صفحه بوده که کوتاهترین مسیر آن نقاط تا مبدأ، در مدت زمان t طی شود. مرز جبهه موج مجموعه‌ای از دایره‌ها بوده که هر کدام از آنها از کمانهایی پی‌درپی و متوالی تشکیل شده است. هر کدام از این کمانها را یک موجک می‌نامیم و این موجک‌ها توسط یک رأس مانع که قبلا توسط جبهه موج پوشش داده شده است، ساخته می‌شود، این رأس را مولد(ژنراتور) موجک می‌نامیم. همانطور که در شکل ۳.۱ مشاهده می‌کنید، طبق الگوریتم پیوسته دایکسترا، ابتدا از رأس s جبهه موج ایجاد کرده تا دایره c_1 ایجاد شده به اولین رأس یعنی t_1 برسد سپس پاره‌خط $\overline{st_1}$ کوتاهترین پاره‌خط بین این دو نقطه است. مجددا از رأس s با ایجاد جبهه موج سعی در پیدا کردن دومین رأس می‌کنیم. دایره c_2 در رأس دوم به t_2 می‌رسد. اما چون قسمتی از پاره‌خط $\overline{st_2}$ از فضای درون چندضلعی عبور می‌کند، این پاره‌خط، کوتاه‌ترین پاره‌خط نخواهد بود. در مرحله بعد، جبهه موج را از رأس t_1 شروع به ایجاد می‌کنیم تا پاره‌خط $\overline{t_1t_2}$ حاصل شود. مسیر $s \longrightarrow t_1 \longrightarrow t_2$ کوتاه‌ترین مسیر بین رأس‌های s و t_2 است.



شکل ۳.۱: کوتاه‌ترین مسیر بین دو رأس به کمک الگوریتم پیوسته دایکسترا

نقطه تلاقی دو موجک مجاور در امتداد مرکز منحنی، یک خط راست یا یک هذلولی است. شبیه‌سازی جبهه موج نیاز به پردازش رویدادهایی دارد که مکان‌شناسی آن را تغییر می‌دهد. این رویدادها در دو دسته قرار می‌گیرند: برخوردهای جبهه موج–جبهه موج و برخوردهای جبهه موج–موانع. توانایی پردازش این رویدادها بطور مؤثر کلید یک الگوریتم سریع برای کوتاهترین مسیر مسئله است. با این حال، به سرعت، تشخیص و پردازش این رویدادها کاملاً مشکل است و به جز نتیجه اخیر میشل [۳] [۲۸]، تمام الگوریتم‌های قبلی که از روش پیوسته دایکسترا استفاده کردند، نتیجه‌ای بهتر از این نداشتند که مدت زمان در بدترین حالت برابر $\Omega(n^2)$ است.

دو ایده جدید را برای تسریع در اجرای روش انتشار جبهه موج معرفی می کنیم: ایده اول سبک درخت زیرمجموعه‌ای از صفحه بوده، و ایده دوم تقریب زدن جبهه موج است. ایده اول ما این است

[3] Mitchell

که تشخیص دهیم که پیشروی یک خط مقدم موج از یک رویداد به رویداد دیگر می تواند بسیار مهم باشد بدون داشتن یک زیر مجموعه مناسب و خوش‌رفتار از صفحه که انتشار امواج را هدایت کند. یک زیرمجموعه خاص در اندازه $O(n)$ برای رئوس موانع می‌سازیم، موقتا پاره‌خط بین آنها را نادیده می‌گیریم. هر سلول از این زیر واحد، با نام زیرمجموعه سازگار، تعداد ثابتی یالهایی به صورت خط صاف دارد، دارای حداکثر یک رأس مانع است، و ویژگی اساسی زیر را نتیجه می‌دهد: برای هر یال e از این زیرمجموعه، سلول‌هایی با اندازه $O(1)$ وجود دارند که فاصله آنها از یال e برابر $2|e|$ است. سپس پاره‌خط‌های مانع را در زیر مجموعه قرار می‌دهیم، اما اندازه خطی زیرمجموعه و خاصیت انطباق آن را حفظ کنید اکنون یک یال غیر مانع e این خاصیت را دارد که سلولهای $O(1)$ در کوتاهترین مسیر با فاصله $2|e|$ از یال وجود دارند. این سلولها واحدهای الگوریتم انتشار ما را تشکیل می‌دهند: در هر مرحله، پیشروی جبهه موج را از طریق یک سلول پیش می‌بریم. از آنجا که هر سلول پیچیدگی واضح و روشنی دارد، ما قادر به انجام انتشار در یک سلول بطور دقیق هستیم.

در داخل یک سلول، جبهه موج ـ موانع یک رویدادی است که راه حل آن نسبتاً آسان است. همینطور جبهه موج ـ جبهه موج یک رویدادی پیچیده‌تر است. رویداد جبهه موج ـ جبهه موج به دو قسمت تقسیم می‌شود، بسته به اینکه آیا موجک‌های برخورد کننده در جبهه موج با هم همسایه(مجاور) هستند یا خیر. برخورد موجک‌های همسایه هنگامی اتفاق می‌افتد که یک موجک با گسترش موج‌های دو همسایه خود درگیر شود. این رویداد به راحتی قابل تشخیص و پردازش است. با این حال، برخورد بین موج‌های غیرهمسایه مشکل‌ساز است و برای پردازش آنها ایده دوم خود را معرفی می‌کنیم: تقریب زدن جبهه موج.

هنگام تلاش برای انتشار موج افقی در یال مرزی یک سلول، ایده محاسبه جبهه موج را دقیقاً کنار می‌گذاریم. درعوض، دو جبهه موج جداگانه را حفظ می‌کنیم، و از طرف مقابل به یال نزدیک می‌شویم. هر یک از این جبهه‌های موج یک جبهه موج تقریبی است، نشان می‌دهد که جبهه موج فقط از یک طرف به یال برخورد می‌کند.

از زمان‌سنج استفاده کرده تا کمترین زمان از لحظه‌ای که هر یال توسط جبهه موج درگیر است، انجام دهیم، و هر قسمت از جبهه موجی را که پس از طی کردن یک زمان در آن یال مرزی، وارد مرز سلول می‌شود، حذف کنید. وظیفه مهم این زمان‌سنج‌ها این است که اطمینان حاصل کند که برخورد جبهه موج کوتاه از نقشه کوتاهترین مسیر واقعی در طول انتشار تقریبی جبهه موج در یک همسایگی کوچک از محل واقعی آنها تشخیص داده می‌شود. الگوریتم جبهه موجی تقریبی را انتشار می‌دهد، به خاطر سپردن برخوردهای جبهه موج ـ جبهه موج و به روزرسانی جبهه موج به گونه‌ای که اطلاعات کافی در هر زمان به عنوان یک موجی تقریبی داشته باشد.

در پایان مرحله انتشار، تمام اطلاعات مربوط به برخورد را جمع‌آوری می‌کنیم، سپس استفاده می‌کنیم تکنیک‌های نمودار ورونی [۴] در هر سلول برای محاسبه دقیق حوادث تصادفی در آن سلول است. برخوردها، یال‌های نقشه کوتاهترین مسیر را تعیین می‌کنند.

[4] Voronoi diagram

۴.۱ نمادهای مجانبی

نمادهایی که برای توصیف زمان اجرای تقریبی یک الگوریتم استفاده می‌کنیم را نمادهای مجانبی می‌گوییم، این نمادها را می‌توان به کمک توابع تعریف کرد. به خاطر داشته باشیم که دامنه این توابع مجموعه اعداد طبیعی می‌باشد. اینچنین نمادها برای توصیف بدترین حالت عملکرد تابع $T(n)$، که معمولا اندازه ورودی آنها عددی صحیح باشد تعریف می‌شوند. با این وجود، گاهی اوقات از علائم مجانبی به روشهای مختلف می‌توان استفاده کرد. به عنوان مثال، ممکن است نماد را به دامنه اعداد حقیقی گسترش دهیم یا برعکس، آن را به یک زیر مجموعه از اعداد طبیعی محدود کنیم. با این وجود باید اطمینان حاصل کنیم که معنای دقیق این علامت را بفهمیم تا هنگام استفاده، از آن سوء برداشت نکنیم. ضمنا در این بخش توابع غیرمنفی را بررسی خواهیم کرد. مطالب این بخش از کتاب آشنایی با الگوریتم‌ها [۸] انتخاب شده است.

۱.۴.۱ نماد Θ

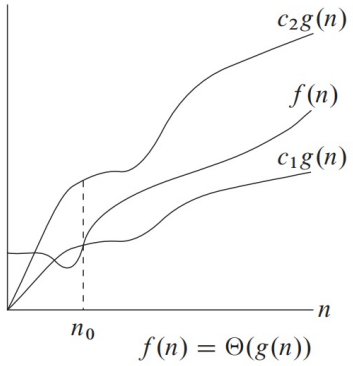
تابع $g(n)$ مفروض است. نماد $\Theta(g(n))$ را برای این تابع به صورت زیر تعریف می‌کنیم:

$$\Theta(g(n)) = \{f(n) : \circ \leq c_1 g(n) \leq f(n) \leq c_2 g(n), \quad \exists c_1, c_2, n_\circ \in \mathbb{R}^+, \forall n \geq n_\circ\}$$

تابع $f(n)$ به مجموعه $\Theta(g(n))$ تعلق دارد هرگاه مقادیر ثابت و مثبت c_1 و c_2 وجود داشته باشند بطوریکه در قضیه فشردگی (ساندویچ [۵]) بین دو مقدار $c_1 g(n)$ و $c_2 g(n)$ (برای مقدار به قدر کافی بزرگ n) صدق کند. از آنجایی که $\Theta(g(n))$ یک مجموعه است پس می‌توان نوشت $f(n) \in \Theta(g(n))$ و این یعنی $f(n)$ عضوی از مجموعه $\Theta(g(n))$ است. در عوض، برای بیان همین مفهوم معمولا از تساوی $f(n) = \Theta(g(n))$ استفاده می‌کنیم.

شکل ۴.۱ تصویر شهودی از توابع $f(n)$ و $g(n)$ را به ما می‌دهد بطوریکه $f(n) = \Theta(g(n))$ برای تمام مقادیر n و همینطور در سمت راست $n_\circ$، این نمودار بالاتر از $c_1 g(n)$ و پایین‌تر از $c_2 g(n)$ قرار دارد. به عبارت دیگر، برای هر $n \geq n_\circ$، تابع $f(n)$ در یک عامل ثابت با تابع $g(n)$ برابر است. تابع $g(n)$ یک محدوده مجانبی محکم برای $f(n)$ است.
تعریف $\Theta(g(n))$ نیازمند آن است که همه اعضای تابع $f(n) \in \Theta(g(n))$ مجانبی باشند، هر زمان n به اندازه کافی بزرگ باشد، $f(n)$ خواهد بود.(یک مجانبی مثبت تابع آن است که مقدار آن برای مقادیر بزرگ n مثبت باشد) در نتیجه، تابع $g(n)$ باید بدون علامت غیر منفی بوده یا اینکه مجموعه $\Theta(g(n))$ تهی باشد. بنابراین فرض می‌کنیم که هر تابعی که از نماد Θ استفاده کرد، مجانب است. این فرض سایر نمادهای مجانبی تعریف شده در این فصل را نیز شامل می شود.

[5] Sandwich Theorem



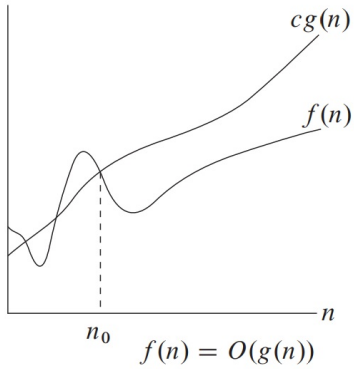
شکل ۴.۱: نماد Θ یک تابع را در محدوده عوامل ثابت محدود می‌کند.

۲.۴.۱ نماد O

نماد Θ محدوده مجانب وار یک تابع از بالا و پایین است. اما زمانی که فقط محدوده مجانب وار از بالا داشته باشیم، از نماد O استفاده می‌کنیم. تابع $g(n)$ مفروض است، مجموعه‌ای از توابع $O(g(n))$ (دقت کنیم که O را بصورت O بزرگ نوشته و تلفظ می‌کنیم) را به صورت زیر نشان می‌دهیم:

$$O(g(n)) = \{f(n) : \circ \leq f(n) \leq cg(n), \quad \exists c, n_\circ \in \mathbb{R}^+, \quad \forall n \geq n_\circ\}$$

از نماد O استفاده می‌کنیم تا حد بالایی را در مورد یک تابع قرار دهیم. شکل ۵.۱ درک بهتری از نماد O به ما می‌دهد. برای همه مقادیر n که در سمت راست $n_\circ$ هستند، مقدار تابع $f(n)$ برابر یا کمتر از مقدار $cg(n)$ است. عبارت $f(n) = O(g(n))$ نشان می‌دهد که تابع $f(n)$ یک عضو از مجموعه $O(g(n))$ است. از تساوی $f(n) = \Theta(g(n))$ می‌توان تساوی $f(n) = O(g(n))$ را نتیجه گرفت. زیرا نماد Θ نماد قویتری نسبت به نماد O است. به عبارت دیگر می‌توان گفت که $\Theta(g(n)) \subseteq O(g(n))$.



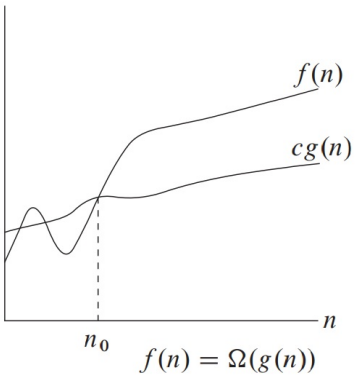
شکل ۵.۱: نماد O محدوده بالایی را برای یک تابع در یک عامل ثابت ایجاد می کند.

۳.۴.۱ نماد Ω

نماد O محدوده مجانبی از بالای یک تابع است اما نماد Ω محدوده مجانبی از پایین یک تابع است. تابع $g(n)$ را در نظر بگیرید، مجموعه توابع $\Omega(g(n))$ را به صورت زیر نشان می‌دهیم:

$$\Omega(g(n)) = \{f(n) : \circ \leq cg(n) \leq f(n), \quad \exists c, n_\circ \in \mathbb{R}^+, \quad \forall n \geq n_\circ\}$$

شکل ۶.۱ درک بهتری از نماد Ω به ما می‌دهد. برای همه مقادیر n که در سمت راست $n_\circ$ هستند، مقدار تابع $f(n)$ برابر یا بیشتر از مقدار $cg(n)$ است.



شکل ۶.۱: نماد Ω برای یک تابع از پایین با یک عامل ثابت محدودیت ایجاد می‌کند.

وقتی می گوییم که زمان اجرا یک الگوریتم برابر $\Omega(g(n))$ است، منظور ما این است که مهم نیست که ورودی خاص اندازه n برای هر مقدار از n برای چه مقدار انتخاب شود، زمان اجرا روی آن ورودی حداقل زمان $g(n)$ است. به عبارت دیگر، به بهترین حالت زمان اجرای یک الگوریتم محدودیت کمتری داده و آن را به عنوان زمان اجرا در نظر می‌گیریم.

۴.۴.۱ نماد o

نماد O برای نمایش محدوده مجانبی از بالا بود که ممکن است مجانبی چسبیده باشد یا چسبیده نباشد. محدوده $2n^2 = O(n^2)$ مجانبی چسبیده است اما محدوده $2n = O(n^2)$ مجانبی چسبیده نیست. از نماد o زمانی استفاده می‌کنیم که نشان دهیم محدوده ما مجانبی چسبیده نیست. مجموعه $o(g(n))$ (البته در نظر داشته باشید که حرف o کوچک است) را به صورت زیر تعریف می‌کنیم:

$$o(g(n)) = \{f(n) : \circ \leq f(n) < cg(n) \mid \forall c > \circ, \exists n_\circ > \circ, \forall n \geq n_\circ\}$$

به عنوان مثال، $2n = o(n^2)$، اما $2n^2 \neq o(n^2)$.

تعاریف نماد O و نماد o مشابه یکدیگر هستند. تفاوت اصلی این دو نماد در این است که اگر

$f(n) = O(g(n))$، آنگاه نامعادله $0 \leq f(n) \leq cg(n)$ برای بعضی از مقادیر ثابت c که $c > 0$ برقرار است، اما نامعادله $0 \leq f(n) < cg(n)$ برای همه مقادیر ثابت c که $c > 0$ برقرار است. بطور شهودی، در نماد o با نزدیک شدن به بی نهایت، مقدار تابع $f(n)$ نسبت به مقدار تابع $g(n)$ ناچیز می‌شود، به این معنی که:

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

در برخی منابع از حد فوق به عنوان تعریف نماد o استفاده می‌شود اما در این منبع تعریف o شامل همه توابعی می‌شود که مجانبی نامنفی دارند.

۵.۴.۱ نماد ω

بر اساس قیاس، نماد ω به نماد Ω به مانند نماد o به نماد O است. از نماد ω زمانی استفاده می‌کنیم که بخواهیم محدوده پایین رو نشان دهیم بطوریکه مجانبی چسبیده نباشد. یکی از راههای تعریف این نماد به صورت زیر است: $f(n) \in \omega(g(n)) \iff g(n) \in o(f(n))$
مجموعه $\omega(g(n))$ را به صورت زیر تعریف می‌کنیم:

$$\omega(g(n)) = \{f(n) : 0 \leq cg(n) < f(n) \mid \quad \forall c > 0, \exists n_0 > 0, \forall n \geq n_0\}$$

به عنوان مثال، $n^2/2 = \omega(n)$، اما $n^2/2 \neq \omega(n^2)$. رابطه $f(n) = \omega(g(n))$ نشان می‌دهد که

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

یعنی با نزدیک شدن به بی‌نهایت، $f(n)$ به طور دلخواه بزرگتر از $g(n)$ می‌شود.

در این پایان‌نامه، جنبه‌های ساختاری و محاسباتی کوتاهترین kـ مسیر را بررسی می‌کنیم. این مسأله با مسأله 0ـ مسیر در صفحه متفاوت است. دو تا کوتاهترین صفرـ مسیر با یک نقطه مشترک بنا به نامساوی مثلثی، نمی‌توانند همدیگر را قطع کنند، و این یک ویژگی اساسی برای محاسبه آنها در زمان بهینه است [۱۹]. اما، دو تا کوتاهترین kـ مسیر$(k > 0)$ می‌توانند همدیگر را قطع کنند. مسائل هندسی kـ مسیر از نظر تئوری، بعنوان قسمتی از دسته‌بندی وسیع بهینه‌سازی با موانع [۶، ۲۵] یا مسئله تقویت شبکه [۲، ۱۲]، جالب هستند و بطور عملی برای برنامه‌هایی مانند طراحی حرکت روبات، جایی که ممکن است تغییر محیط روبات برای مسیرهای کوتاه شده‌ای که استفاده شده است، مفید باشد.(مسائل هندسی kـ مسیر به نظر می‌رسند که نسبت به تقویت شبکه پیچیده‌تر هستند، زیرا حذف یک مانع تنها می‌تواند یالهای اضافی زیادی در فضای صفحه ایجاد کند.) علاوه بر این، در مسیریابی حرکت روبات، مسئله می‌تواند به این صورت مدل شود که می‌توان از موانع با صرف هزینه برای ساختن پل یا احداث تونل اجتناب کرد.
رویکرد ما برای حل مسئله kـ مسیر محاسبه‌ی نقشه کوتاهترین kـ مسیر یا SPM_k است، که افرازی از صفحه به کلاسهای هم‌ارزی از سلولها(ناحیه‌ها) است به طوریکه همه نقاط پایانی داخل یک ناحیه

دارای کوتاهترین kـمسیر یکسان از نظر ساختار ترکیبیاتی هستند. هنگامی که نقشه را داشته باشیم، کوتاهترین kـمسیر به هر مقصد را می توان با انجام یک پرس و جوی مکانی به روی نقشه محاسبه کرد [۱۰، ۲۲].

کارهای مرتبط

مسئله محاسبه کوتاهترین مسیرها با موانع، یک تاریخچه طولانی در هندسه محاسباتی دارد که قدمت آن به دهه ۱۹۷۰ باز می‌گردد. مخصوصا، مورد موانع چندضلعی در صفحه، به صورت گسترده بررسی شده است [۳، ۴، ۱۳، ۲۱، ۲۶، ۲۷، ۲۹، ۳۰، ۳۲]، که موجب رسیدن به چارچوب پیوسته دایکسترا با زمان بهینه‌ی $O(n \log n)$ شده است [۱۹]. بسیاری از گونه‌های دیگر مسئله، از جمله کوتاهترین مسیرهای داخل چندضلعی ساده [۱۴، ۱۶، ۲۳]، در بین ناحیه‌های وزندار [۲۸]، و همچنین در میان موانع منحنی [۶] [۷، ۲۰]، مورد مطالعه قرار گرفته است. کلیت مسئله ما مربوط به بهینه‌سازی هندسی است، جایی که تعداد محدودی از قیدها قابل حذف هستند. این نمونه کار در [۶، ۱۵، ۲۵، ۳۱] و همینطور در زمینه برنامه‌نویسی خطی با ابعاد کم، جدایی با محوطه، و بهینه‌سازی هندسی، دنبال شده است. مسئله ما همچنین می‌تواند به عنوان یک شکل از تقویت شبکه مشاهده شود، که در آن هدف اضافه کردن یال‌ها به شبکه برای بهبود اتصال، قطر، یا نسبت طول و غیره است [۱، ۲، ۵، ۱۲]. کار قبلی که از نزدیک با تحقیقات ما بیشتر مرتبط است، نتیجه جدیدی است که توسط ماهشواری [۷] و همکاران انجام شده است [۲۴]، که یک الگوریتم زمان $O(n^3)$ را برای محاسبه مسیر ۱ـحذفی در داخل یک چندضلعی ساده ارائه می دهد: یعنی کوتاهترین مسیر در داخل یک nـضلعی ساده که مجاز به حذف یک چندضلعی است. کار ما در یافتن مسیرهای kـحذفی ، برای k دلخواه، در یک محیط شامل احتمالاً $O(n)$ موانع محدب است.

نتایج ما

برای یک عدد صحیح $k \geq 0$، یک مسیر kـحذفی را از s تا t تعریف می‌کنیم تا مسیری باشد که s و t را به هم وصل کند به طوری که مسیری با حداقل طول اقلیدسی در بین تمام مسیرهای $(\leq k)$ـحذفی ممکن از s تا t محاسبه شود و الگوریتمی را پیشنهاد می‌کنیم که کوتاهترین مسیر یکـحذفی را در زمان $O(n^3)$ محاسبه کند. همینطور برای چندضلعی‌های مستطیل، حداقل طول مسیر یکـحذفی می‌تواند در زمان $O(n \log n)$ محاسبه می‌شود.

از طرفی نشان می‌دهیم که SPM_k دارای $O(kn)$ ناحیه و $O(kn)$ یال می‌باشد (فصل۵). یک الگوریتم زمان $O(k^2 n \log n)$ برای محاسبه SPM_k ارائه می‌دهیم (فصل ۵ و در بخش ۴.۵)، که با استفاده از چارچوب پیوسته دایکسترا، SPM_j را به ترتیب برای هر $0 \leq j \leq k$ می‌سازیم. زمان اجرای الگوریتم برای $k = O(1)$ بهینه است.

---

[6] Curved Obstacles

[7] Maheshwari

فصل ۲

محاسبه گراف رویت پذیری

در این فصل می‌خواهیم گراف رؤیت پذیری و انواع الگوریتم‌های مربوط به آن و همینطور مسائل کوتاهترین مسیرهای یک حذفی را مورد بررسی قرار دهیم.
فرض کنید S مجموعه موانع چندضلعی‌های از هم جدا در صفحه بوده که در حالت کلی دارای n یال می‌باشند. (الگوریتم کوتاهترین مسیر نیازمند محاسبه گراف رؤیت پذیری مجموعه‌ی S^*، که شامل نقاط ابتدایی و انتهایی است. حضور رئوس تنها، هیچ مشکلی را در حل مسائل بوجود نمی‌آورد، به همین دلیل از این رئوس صرف نظر می‌کنیم.) برای محاسبه‌ی گراف رؤیت پذیری S، باید جفت رئوسی را پیدا کنیم که برای یکدیگر قابل مشاهده باشند. به عبارت دیگر، برای هر جفت رأس این آزمایش را می‌کنیم که پاره‌خطی واصل بین این دو رأس آیا موانع را قطع می‌کند. آزمایشاتی از این قبیل، زمان $O(n)$ تا زمان $O(n^۳)$ را منجر شود. به زودی خواهیم دید که اولویت بندی انتخاب جفت رأس‌ها از اهمیت بالایی برخوردار است بطوریکه اگر این جفت‌ها را به ترتیب دلخواه درنظر نگیریم، می‌توان آزمایش را با کارآیی بیشتری انجام داد، اما در یک زمان بر روی یک راس متمرکز شویم و مانند همه الگوریتم‌های زیر، همه رأس‌های قابل مشاهده از آن را شناسایی کنیم.
مطالب بخش‌های ۱.۲ و ۲.۲ این فصل، از کتاب هندسه محاسباتی [۹] انتخاب شده است.

۱.۲ الگوریتم رئوس رؤیت پذیر S

ورودی این الگوریتم، مجموعه S که شامل موانع چندضلعی ناپیوسته است.
خروجی این الگوریتم، گراف رؤیت پذیری $\mathcal{G}_{\text{vis}}(S)$ است.
اینک به صورت زیر عمل کنید:

۱. با گراف $\mathcal{G} = (V, E)$ شروع کرده که V مجموعه همه‌ی رئوس چندضلعی‌های S است و $E = \emptyset$.

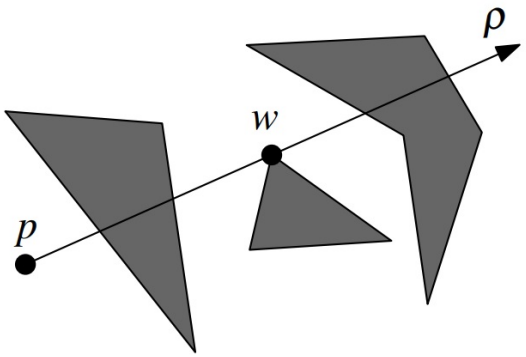
۲. برای همه‌ي رئوس $v \in V$.

۳. همه‌ی رئوسی که از رأس v قابل رؤیت هستند را مجموعه W بنامید.

۴. برای هر رأس $w \in W$، یال (v, w) را در E اضافه کنید.

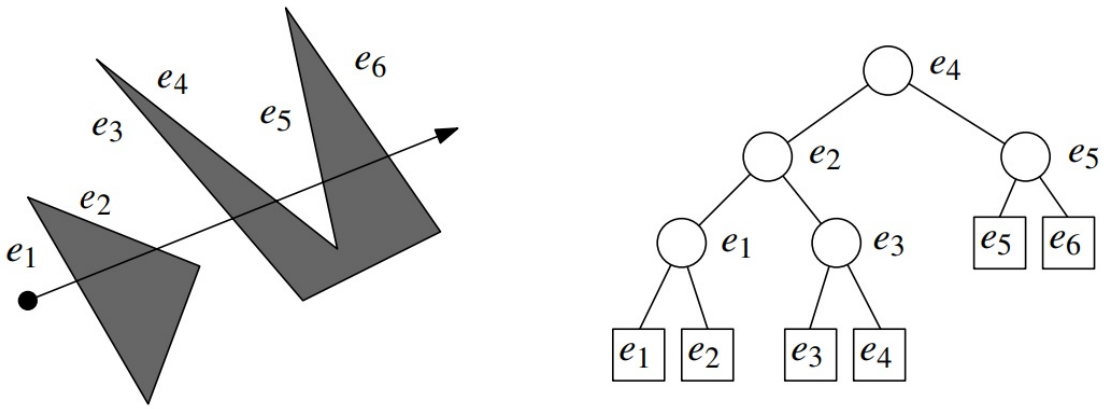
۵. به گراف $\mathcal{G}$ برگردید.

روش رئوس رؤیت پذیر در صورتی که S مجموعه‌ای از موانع چندضلعی و p نقطه‌ای در صفحه باشد، کاربرد دارد، اما اگر p یک رأس از مجموعه‌ی S باشد (p رأسی از موانع چندضلعی مجموعه S باشد)، این الگوریتم قابل استفاده نیست. باید همه‌ی رأس‌های مانع قابل مشاهده از p را برگرداند.
اگر فقط بخواهیم آزمایش کنیم که یک رأس w از رأس p قابل مشاهده است، کار چندان سختی نیاز نیست انجام دهیم: باید پاره‌خط $\overline{pw}$ را در برابر همه‌ی موانع بررسی کنیم. اما اگر بخواهیم تمام رأس‌های S را بررسی کنیم، انتظار می‌رود اطلاعاتی که از آزمایش یک رأس بدست می‌آید، برای آزمایش رأس دیگر نیز مورد استفاده قرار گیرد. اکنون همه‌ی پاره‌خط‌های $\overline{pw}$ را در نظر بگیرید. دستورالعمل مناسبی برای بررسی آنها خواهد بود تا بتوانیم در هنگام بررسی‌های بعدی از اطلاعات یک راس استفاده کنیم؟ یک انتخاب منطقی انتخاب دایره‌وار و چرخه‌ای رأس‌های اطراف p است. بنابراین آنچه ما انجام خواهیم داد، بررسی رأس‌ها به ترتیب چرخه‌ای است، در عین حال نگه داشتن اطلاعاتی که به ما کمک می‌کند تا در مورد رؤیت پذیری رأس بعدی که مورد بررسی قرار می‌گیرد تصمیم بگیریم.
رأس w از رأس p رؤیت شده هرگاه پاره‌خط $\overline{pw}$ از فضای داخلی هر مانع عبور نکند. نیم‌خط ρ با نقطه ابتدایی p که از نقطه w می‌گذرد را در نظر بگیرید. اگر از رأس p رأس w رؤیت نشود، آنگاه نیم‌خط ρ قبل از اینکه به رأس w برسد، به یال یک مانع برخورد خواهد کرد.
برای بررسی این مسئله، یک جستجوی دوتایی با رأس w در یال‌های مانع که توسط ρ قطع شده است، انجام می‌دهیم. با این روش می‌توان فهمید که آیا w در پشت هر یک از این یال‌ها قرار دارد. همانطور که در شکل ۱.۲ می‌بینید. (اگر خود رأس p نیز یک رأس از مانع باشد، در این صورت تحت یک شرط رأس w باز قابل رؤیت نیست و آن شرط این است که رئوس p و w رأس‌های یک مانع باشند و پاره‌خط $\overline{pw}$ درون مانع قرار داشته باشد. این موارد را می‌توان با نگاه کردن به یال‌های به سمت w بررسی کرد که آیا نیم‌خط ρ قبل از رسیدن به w در قسمت داخلی آن مانع است یا خیر. در حال



شکل ۱.۲: رؤیت پذیر بودن دو رأس

حاضر ما اینگونه موارد خاص را نادیده می‌گیریم.)
در ضمن بررسی رأس‌ها به صورت دایره‌وار و چرخه‌ای پیرامون رأس p، یال‌های موانع در تقاطع با نیم‌خط ρ را به صورت نمودار درختی τ رسم می‌کنیم. (همانطور که بعدا خواهیم دید، یال‌هایی که در نیم‌خط ρ وجود دارد، نیازی به ذخیره‌سازی در نمودار درختی τ ندارند.) برگ‌های نمودار درختی τ یال‌های تقاطع را به ترتیب نگه می‌دارند: سمت چپ برگ اولین پاره‌خط قطع شده توسط نیم‌خط ρ، برگ بعدی نمودار درختی، پاره‌خطی است که تقاطع بعدی را با نیم‌خط ρ دارد و همینطور الی آخر. گره‌های داخلی، که جستجو را در نمودار درختی τ انجام می‌دهند، یال‌ها را نیز ذخیره می کنند. به طور دقیق تر، یک گره داخلی v، یال سمت راست را در زیر شاخه سمت چپ خود ذخیره می‌کند، به طوری که تمام یال‌های موجود در زیر درخت راست آن از نظر شماره، از پاره‌خط e_v بزرگتر هستند (با توجه به نظم در طول نیم‌خط ρ) و همه بخش‌های موجود در زیربنای چپ آن از نظر شماره، کوچکتر یا برابر با e_v است. (با توجه به نظم همراه نیم‌خط ρ) شکل ۲.۲ مثالی از این بحث است.



شکل ۲.۲: نمودار درختی در یال‌های قطع شده

بررسی رئوس به ترتیب چرخه‌ای به معنای مؤثر بودن این است که نیم‌خط ρ را در حدود p بچرخانیم. بنابراین رویکرد ما شبیه به الگوی رفتاری صفحه است که در مکان‌های مختلف دیگری از آن استفاده

کردیم. تفاوت این است که به جای استفاده از یک خط افقی که به سمت پایین حرکت می‌کند تا صفحه را جارو کنیم، از یک نیم‌خط ساعتگرد به مبدأ نقطه p، استفاده می‌کنیم.
وضعیت جارو کردن صفحه چرخشی ما ترتیب توالی یال‌های مانع در تقاطع با نیم‌خط ρ است. در نمودار درختی τ حفظ می‌شود. رویدادهای موجود در این جارو کردن رأس‌های S است. برای مقابله با یک رأس w باید تصمیم بگیریم که آیا رأس w از رأس p با جستجو در ساختار نمودار درختی τ، رؤیت می‌شود، و باید نمودار درختی τ را با اضافه کردن و یا با حذف کردن یال‌های مانع با w تلاقی پیدا کند.
الگوریتم رؤیت رئوس جارو کردن صفحه چرخشی را خلاصه می‌کند. جارو کردن به وسیله‌ی نیم‌خط ρ با حرکت در جهت مثبت محور Xها شروع شده و در جهت عقربه‌های ساعت حرکت می‌کند. بنابراین الگوریتم ابتدا رئوس را در جهت حرکت عقربه‌های ساعت، زاویه‌ای که پاره‌خط از p به هر رأس با قسمت مثبت محور Xها می‌سازد، مرتب می‌کند. برای اینکه بتوانیم در مورد رؤیت پذیری یک رأس w تصمیم بگیریم، باید بدانیم که آیا پاره‌خط $\overline{pw}$ از فضای داخلی هر مانعی عبور می‌کند یا خیر. از این رو، انتخاب واضح این است که قبل از اینکه رأس w را بررسی کنیم، هر گونه رأس که ممکن است در قسمت داخلی $\overline{pw}$ باشد وجود دارد. به عبارت دیگر، رأس‌هایی که زاویه آنها یکسان است به منظور افزایش فاصله تا p بررسی می‌شوند. این الگوریتم به شرح زیر است:

۲.۲ الگوریتم رئوس رؤیت پذیر (p, S)

ورودی الگوریتم: مجموعه S شامل موانع چندضلعی و نقطه p که در فضای داخلی هیچ مانعی قرار ندارد.
خروجی الگوریتم: مجموعه همه‌ی رئوس موانع که از نقطه p قابل مشاهده هستند.

۱. رأس‌های موانع را مطابق با زاویه عقربه ساعت مرتب سازی کنید که نیم‌خط از p به هر راس با جهت مثبت محور xها ایجاد کند. در صورت برخورد، رأس‌های نزدیک‌تر به p باید قبل از رأس دورتر از p بیایند. فرض کنید w_n و . . . و w_1 لیست مرتب شده باشد.

۲. فرض کنید نیم‌خط ρ با نقطه ابتدایی p، موازی با جهت مثبت محور xها باشد. یال‌های موانع را که به طور صحیح توسط نیم‌خط ρ قطع شده‌اند، پیدا کنید و آنها را در یک نمودار درختی τ به ترتیبی که توسط نیم‌خط ρ قطع شده‌اند، یادداشت کنید.

۳. $W \longleftarrow \emptyset$

۴. برای $i \longleftarrow 1$ تا n

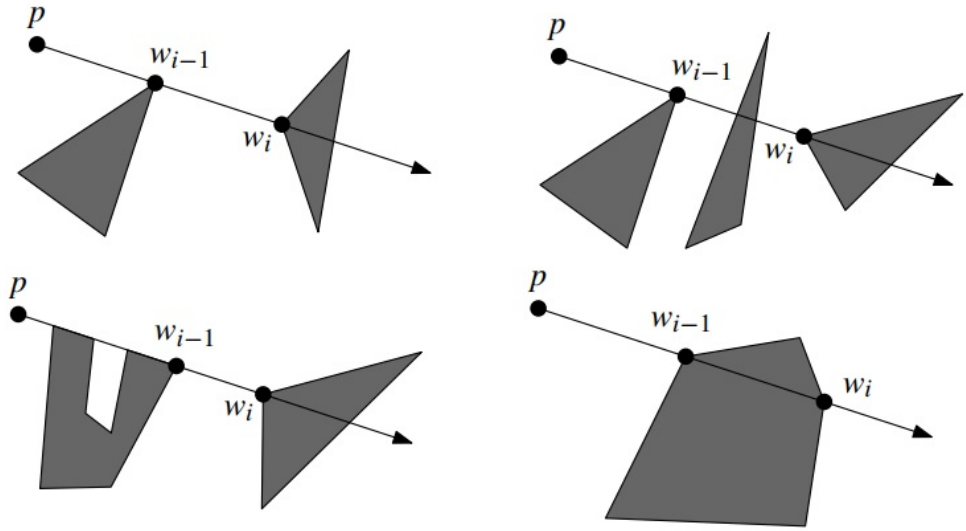
۵. اگر w_i قابل مشاهده بود، آنگاه w_i را به W اضافه کنید.

۶. یال‌های موانعی که به w_i در جهت حرکت عقربه‌های ساعت نیم‌خط با نقطه ابتدایی p و گذرنده از w_i، برخورد می‌کنند را در نمودار درختی τ قرار دهید.

۷. یال‌های موانعی که به w_i در جهت عکس حرکت عقربه‌های ساعت نیم‌خط با نقطه ابتدایی p و گذرنده از w_i، برخورد می‌کنند را از نمودار درختی τ حذف کنید.

۸. به W برگردید.

قابل مشاهده بودن رأس w_i توسط زیرمجموعه‌ی قابل مشاهده شده‌ها مشخص می‌گردد. به طور معمول، این فقط شامل جستجو در نمودار درختی τ می‌شود که آیا یال نزدیک به p، که در پایین‌ترین برگ یادداست شده است، پاره‌خط $\overline{pw_i}$ را قطع می‌کند. اما باید دقت کنیم که پاره‌خط $\overline{pw_i}$ شامل رئوس دیگر باشد. آیا w_i در چنین مواردی قابل مشاهده است یا خیر؟ که بستگی دارد. در شکل ۳.۲ برای بعضی از مواردی که ممکن است رخ دهد. پاره‌خط $\overline{pw_i}$ ممکن است فضای داخلی موانعی که بر روی این رأس‌ها قرار دارد، قطع نکند. به نظر می‌رسد که باید همه یال‌ها را با یک رأس در $\overline{pw_i}$ بررسی کنیم تا متوجه شویم که آیا w_i قابل مشاهده است یا خیر. خوشبختانه در حال حاضر رأس‌های قبلی که در پاره‌خط $\overline{pw_i}$ قرار دارند را بررسی کرده‌ایم. بنابراین برای رؤیت پذیر بودن w_i به صورت زیر می‌توان عمل کرد. اگر w_{i-1} قابل مشاهده نباشد، w_i نیز قابل مشاهده نیست. اگر w_{i-1} قابل مشاهده باشد، دو روش وجود دارد که w_i می‌توانند قابل مشاهده نباشند. یا پاره‌خط $\overline{w_{i-1}w_i}$ درون مانعی قرار گرفته که w_{i-1} و w_i رأس‌های این مانع هستند، و یا پاره‌خط $\overline{w_{i-1}w_i}$ توسط یک یال نمودار درختی τ قطع شده است. (از آنجا که در حالت دوم، این یال بین w_{i-1} و w_i قرار دارد، باید به درستی با $\overline{w_{i-1}w_i}$ قطع شده باشد.) این آزمایش درست است زیرا $\overline{pw_i} = \overline{pw_{i-1}} \cup \overline{w_{i-1}w_i}$ (اگر $i = 1$، آنگاه رأسی بین p و w_i وجود ندارد، بنابراین فقط باید به پاره‌خط $\overline{pw_i}$ توجه کنیم). حالات مختلف بصورت زیر است:



شکل ۳.۲: برخی از مثالها که ρ شامل چندین رئوس است. در همه این موارد w_{i-1} قابل مشاهده است. در دو حالت چپ w_i نیز قابل مشاهده است و در دو سمت راست w_i قابل مشاهده نیست.

رؤیت w_i

۱. اگر پاره‌خط $\overline{pw_i}$ در قسمت داخلی مانعی باشد که w_i رأس آن است، در محلی w_i را قطع کند. سپس مسیر دیگر را انتخاب کنید.

۲. در غیر اینصورت اگر $i = ۱$ یا $w_{i-۱}$ روی پاره‌خط $\overline{pw_i}$ قرار ندارد.

۳. اگر e وجود داشته باشد و پاره‌خط $\overline{pw_i}$، e را قطع کرده باشد سپس مسیر دیگری را انتخاب کنید، در غیر اینصورت ادامه دهید.

۴. و اگر $w_{i-۱}$ دیده نشد، سپس مسیر دیگری را انتخاب کنید.

۵. و اگر نمودار درختی τ از یال e که پاره خط $w_{i-۱}w_i$ را قطع کرده است.

۶. اگر e وجود داشته باشد، سپس مسیر دیگری را انتخاب کنید، در غیر اینصورت ادامه دهید.

این شرح الگوریتم رئوس قابل مشاهده برای محاسبه رأس‌های قابل مشاهده در یک نقطه مشخص به پایان می‌رسد.
زمان اجرای الگوریتم رأس‌های رؤیت پذیر چقدر است؟ زمانی که قبل از خط ۴ صرف کردیم، با زمان مرتب سازی رأس‌ها به ترتیب دایره‌وار و چرخه‌ای در اطراف p که به آنها تسلط داریم، برابر $O(n \log n)$ است. هر اجرای حلقه شامل تعداد ثابت عملیات بر روی نمودار درختی τ است، که زمان $O(\log n)$ را می‌گیرد، به علاوه تعداد ثابت آزمایش‌های هندسی که زمان ثابت را می‌برد. از این رو، یک اجرای زمان $O(\log n)$ را می‌گیرد و منجر به زمان کلی اجرای $O(n \log n)$ می‌شود.
به یاد بیاورید که برای محاسبه کل گراف رؤیت پذیر، باید رئوس رؤیت پذیر را در هر یک از n رأس از مجموعه S اعمال کنیم. قضیه زیر را بدست می‌آوریم:

قضیه ۱.۲.۲. گراف رؤیت پذیر مجموعه‌ی S که شامل موانع چندضلعی ناپیوسته با تعداد کل n یال، در زمان $O(n^۲ \log n)$ محاسبه می‌شود.

فصل ۳

کوتاهترین مسیر kـ حذفی

۱.۳ مسائل مسیر هندسی با حذف موانع

مطالب این بخش از مقاله مسائل مسیر هندسی با حذف [۲۴] انتخاب شده است.
فرض کنید P یک چندضلعی ساده شامل n رأس بوده و s,t یک جفت از نقاط در Pباشد. فرض کنید $int(P)$ نمایش درون P و $\overline{P}$ نمایش بیرون P باشد، یعنی : $int(P) = P\backslash\partial(P), \overline{P} = \mathbb{R}^2\backslash int(P)$.
در مسئله کوتاهترین مسیر در داخل چندضلعی ساده، ورودی شامل P و یک جفت نقطه $s,t \in P$ است. هدف وصل کردن s به t با یک مسیر در P با کوتاهترین طول می‌باشد. در اینجا یک مسیر دنباله‌ای است از پاره‌خط‌ها که به این پاره خط‌ها یال‌های مسیر می‌گویند، مسیر فقط در رأس‌های P تغییر می‌کند (یا می‌چرخد). طول یک مسیر برابر با مجموع طول همه‌ی یال‌های آن مسیر است. مسئله کوتاهترین مسیر درون یک چندضلعی ساده پیشینه تاریخی طولانی و غنی دارد. به خوبی شناخته شده است که کوتاهترین نقشه مسیر از یک نقطه تا کلیه رأس‌ها در P می‌تواند در زمان $O(n)$ محاسبه شود [۱۴].
می‌خواهیم مسائل مختلف کوتاهترین مسیر هندسی را بررسی کنیم که قسمتی از مسیر از P خارج شود. برای هر عدد صحیح $k \geq 0$، مسیر kـحذفی بین نقاط s و t که s و t را به هم وصل کند بطوریکه تقاطع این مسیر با $\overline{P}$ شامل حداکثر k پاره‌خط باشد. از نظر تعداد پاره‌خط‌های مسیر درون P، هیچ محدودیتی نداریم. هدف ما محاسبه کوتاهترین مسیرهای (از نظر طول اقلیدسی) در بین همه‌ی مسرهای $(\leq k)$ـحذفی از s به t است. یکی از موارد مختلف این موضوع به صورت زیر است:

مسئله مسیر ۱ـ حذفی: چندضلعی ساده P و جفت نقطه $s,t \in P$ داده شده است، کوتاهترین مسیر ۱ـحذفی بین s و t را محاسبه کنید بطوریکه مسیر ۱ـحذفی و $\overline{P}$ حداکثر در یک نقطه همدیگر را قطع کرده باشند (برای تصویرسازی بهتر شکل ۱.۳ را ببینید).

۲.۳ کوتاهترین مسیر با حذف در گراف

ابتدا باید مسئله کوتاهترین مسیر kـحذفی در یک گراف ساده $G = (V, E = E_1 \cup E_2)$ را در نظر بگیریم که دو مجموعه‌ی ناپیوسته از یال‌های E_1 و E_2 به ترتیب با نام‌های خوب و بد آشنا می‌شویم. هر یال مانند $(u,v) \in E_1 \cup E_2$ با هزینه غیرمنفی $c(u,v)$ نشان داده می‌شود. می توان با انتخاب بسیاری از یال‌های خوب، مسیری را از s به t محاسبه کنیم اما مجاز به استفاده در اکثر یال‌های بد برای کاهش هزینه‌های مسیر هستیم. هدف این است که مسیری با حداقل هزینه از s به t داشته باشیم. نشان خواهیم داد که الگوریتم دایکسترا را می‌توان برای کار با این مسأله با استفاده از گره‌های فیبوناتچی به عنوان ساختار داده‌های اضافی تنظیم کرد، طوریکه هر عضو گره فیبوناتچی به صورت یک سه‌تایی (a,b,x) است، که در آن a تالی و b مقدم است. هر عضو $a \in \{a_1, a_2, ..., a_N\}$ حداکثر یک نماینده در گره‌ها دارند. هر عضو در گره‌ها ممکن است شامل اطلاعات x دیگری باشند که می‌تواند تهی باشد.

۱.۲.۳ الگوریتم دایکسترا kـ حذفی (G,s,t):

ورودی: گراف جهتدار وزنی $G = (V,E)$ و یک عدد صحیح $k \geq 0$. هر یال (u,v) یک وزن برابر $c(u,v)$ داشته و یک علامت برابر $f(u,v)$ که نشان دهنده‌ی خوب یا بد بودن آن است، دارد.
خروجی: کوتاهترین مسیر kـحذفی از s به t.

۱. قرار دهید $((s,0,0),H)$. (عنصر H را مقدار دهی اولیه کنید)

۲. برای هر $v \in V \setminus \{s\}$ و هر $\mu = 0,1,...,k$ قرار دهید $((v,\mu,\infty),H)$ سپس این عمل را تکرار کنید.

۳. $(u,\mu,\chi) = delete-min(H)$ // // عنصر ریشه را از H حذف کنید.

۴. اگر $u = t$ آنگاه χ را گزارش داده و خارج کنید.

۵. در غیر اینصورت برای هر $v \in Adj(u)$ // // $Adj(u)$ برابر است با لیست گره‌های مجاور گره u.

۶. یال (u,v) را پردازش کنید.

۷. اگر $f(u,v)$=''خوب'' سپس چندتایی $(v,\mu',\chi')=(v,\mu,\chi+c(u,v))$ را ایجاد کنید.

۸. اگر $f(u,v)$=''بد'' سپس چندتایی $(v,\mu',\chi')=(v,\mu+۱,\chi+c(u,v))$ را ایجاد کنید.

۹. اگر $\mu'\leq k$ سپس مقدار $((v,\mu',\chi'),H)$ را کاهش دهید تا زمانی که H تهی شود.

عملیات قرار دادن به این معنی که قرار دادن (a,b,x) قرار دادن یک چندتایی در گره H است، کاهش دادن مقدار به این معنی که افزایش مقدار (a,b') اگر مقدار b موجود در گره دارای کلید a بیشتر از b باشد، مقدار b متناسب با کلید a را در گره H بروزرسانی کنید. اگر بروزرسانی انجام شود، گره H تنظیم می‌شود، یافتن کمترین به این معنی که یافتن کمترین مقدار (H) ورودی را با حداقل اولویت (b) در گره H باز می‌گرداند، را می‌توان در زمان $O(۱)$ انجام داد، و عملیات حذف کمترین مقدار (H) حذف عناصر با حداقل اولویت‌ها از H است (و سپس گره H را بروزرسانی می‌کنیم)، نیازمند زمان لگاریتمی روی ابعاد گروه است.

در حین اجرای الگوریتم، از صف اولویت H که به عنوان یک گروه فیبوناچی نگهداری می‌شود، استفاده می‌کنیم. هر گروه از H یک سه‌تایی (v,μ,χ) است که مطابق است با مسیر از s به گروه v با تعداد حذف شده‌ی μ، و χ نشان دهنده‌ی طول این مسیر است. در اینجا (v,u) نقش کلیدی را بازی می‌کند و χ نقش اولویت‌دار را در گروه فیبوناتچی H بازی می‌کند. علاوه بر این، تعداد k نشانگر C_v را حفظ می‌کنیم (در یک آرایه) با هر گره $v\in V$. با توجه به ورودی μـام از C_v، موقعیت مقدار کلیدی (v,μ) در H را یادداشت می‌کنیم. ورودی‌های $\{C_v,v\in V\}$ در هنگام اجرای عملیات قرار دادن، کاهش دادن، حذف کمترین در H بروزرسانی می‌شوند. در ابتدا، گروه H شامل تعداد $n(k+۱)$ عنصر از تعداد $n(k+۱)$ مقدار کلیدی ممکن مجموعه‌ی $\{(v,\mu)|v\in V,\mu=۰,۱,۲,...,k\}$ است. مقدار χ از هر عنصر یک مجموعه‌ای است که به ∞ میل می‌کند. مکان C_v، $v\in V$ مجموعه‌ای برابر با آن است.

در هر تکرار از الگوریتم دایکسترا تعریف شده، عنصر H که کمترین مقدار χ که با استناد به حذف کمترین (H) را دارد، انتخاب می‌کنیم. آن را با گره $u\in V$ مطابقت دهید. گره u را رها کنید، یا به عبارت دیگر، یال (u,v) برای همه‌ی رأس‌های درون V که مجاور به u هستند را پردازش کنید. برای هر یال (u,v)، یک چندتایی (v,μ',χ') را تولید می‌کند، بطوریکه $\chi'=\chi+c(u,v)$ و $\mu'=\mu$ یا $\mu+۱$ که بستگی دارد به یال (u,v) که ''خوب'' یا ''بد'' هستند. اگر $\mu'\leq k$ و χ' کمتر از مقدار χ متصل به گره (v,μ')ـام از گره $H^۶$ باشد، به کاهش کلیدی (v,μ',χ') استناد می‌کنیم. الگوریتم زمانی که برای اولین بار به t برسد خاتمه می‌یابد، به عبارت دیگر، زمانی یک چندتایی (v,μ,χ) از گروه با $v=t$ گرفته می‌شود. شبه کد روش پیشنهادی در الگوریتم ۱.۲.۳ آورده شده است.

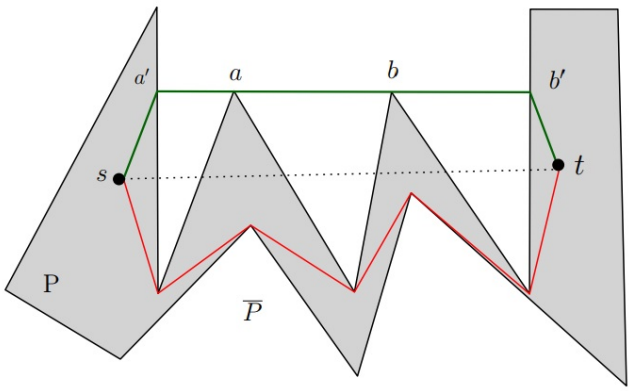
قضیه ۱.۲.۳. الگوریتم دایکسترا kـحذفی به درستی یک مسیر شامل کمترین هزینه با حداکثر kـحذف یال در زمان $O(mk+nk\log n)$ و حافظه $O(m+nk)$ محاسبه می‌کند، بطوریکه $n=|V|$ و $m=|E_۱|+|E_۲|$.

برهان. درستی الگوریتم تبعیت می‌کند از الگوریتم کوتاهترین مسیر دایکسترا، و واقعیت این است

که از دوتایی (v, μ) به عنوان کلید صف اولویت H استفاده خواهیم کرد.
قسمت اول پیچیدگی زمانی از این واقعیت ناشی می‌شود که ممکن است هر یال در اکثر k زمان پردازش شود و برای اجرای صف اولویت از گره فیبوناچی استفاده می‌کنیم. قسمت دوم پیچیدگی زمانی از این واقعیت ناشی می‌شود که ممکن است یک گره از گراف برای رها شدن در حداکثر زمان k با تعداد k حذف متمایز در نظر گرفته شود. عملیات حذف کمترین در زمان $O(nk)$ فراخوانی شده و عملیات هر حذف کمترین نیازمند زمان $O(\log(nk))$ است، که nk ابعاد H می‌باشد.
به غیر از ذخیره گراف، که نیازمند فضای $O(max(m, n))$ بوده، فضای مورد نیاز برای ذخیره C_v برای همه‌ی $v \in V$ برابر $O(nk)$ است. اغلب صف اولویت H نیازمند فضای $O(nk)$ است. □

۳.۳ مسئله مسیر یک‌ـ حذفی در چندضلعی ساده

چندضلعی ساده P شامل n رأس و یک جفت نقطه $s, t \in P$ داده شده است. هدف محاسبه یک مسیر با کمترین طول از s به t است که از حداکثر یک یال مسیر یا قسمتی از آن که در خارج از چندضلعی P عبور می‌کند. از این به بعد، از نماد $\overline{P} = \mathbb{R}\backslash int(P)$ که نشان دهنده‌ی ناحیه خارج از چندضلعی P است، و یالی از مسیر s به t که از میان $\overline{P}$ عبور می‌کند و به آن یال حذف شده می‌گوییم. توجه داشته باشید که برخلاف کوتاهترین مسیر در داخل یک چند ضلعی ساده، در اینجا ممکن است شکستگی در یک مسیر همیشه در رأس چندضلعی قرار نگیرد (شکل ۱.۳ را ببینید)، که یال حذفی کوتاهترین مسیر از s به t در نقطه‌ای از قسمت داخلی یک یال چندضلعی، ممکن است در جایی از مسیر وارد چندضلعی شود (از سمت چپ). در همه جای این بخش، از نماد "$x \rightsquigarrow y$" برای نمایش یک مسیر که مجموعه‌ای از خطوط به هم پیوسته است، از x به y استفاده می‌کنیم، و از نماد $x \longrightarrow y$ برای نمایش یک یال از x به y استفاده می‌کنیم. همینطور از نماد $\prod_{in}(x, y)$ برای نمایش کوتاهترین مسیر بین جفت نقاط $x, y \in P$ درون چندضلعی استفاده می‌کنیم.



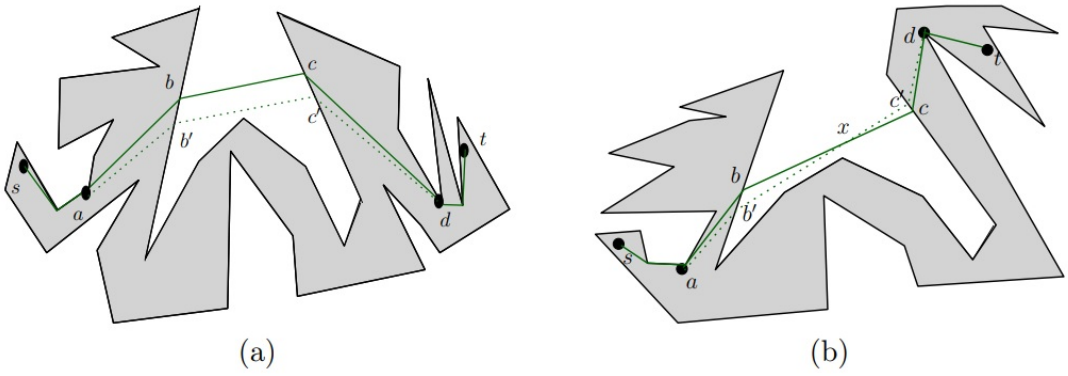
شکل ۱.۳: تصویر سازی $\prod_{in}(s, t)$ (مسیر قرمز رنگ) و کوتاهترین مسیر یک‌ـ حذفی بین مسیر s و t (مسیر سبز رنگ).

نتیجه ۱.۳.۳. در صورتیکه پاره‌خط کاملاً در داخل P قرار داشته باشد، یک مسیر انحصاری از s به t وجود دارد که کوتاهتر از $\prod_{out}(s,t)$ است.

برهان. از آنجاییکه پاره‌خط st کامل درون چندضلعی P قرار ندارد، حداقل یک چرخش در یک رأس مانند v از $\prod_{in}(s,t)$ وجود دارد. دو نقطه α و β به فاصله خیلی کم ϵ از v روی دو پاره‌خط از $\prod_{in}(s,t)$ مجاور به v انتخاب کرده به طوریکه پاره‌خط $\alpha\beta \cap \overline{P}$ یک جزء متصل را تشکیل می‌دهد (یعنی $\alpha\beta$ یک یال حذفی است). به کمک نامساوی مثلثی، مسیر بدست آمده با جایگزین کردن پاره‌خط $\alpha \to v \to \beta$ از $\prod_{in}(s,t)$ با $\alpha \to \beta$ مسیر کوتاهتر خواهد شد، و یک مسیر حذفی ممکن از s به t به ما می‌دهد. □

لم ۱.۳.۳. فرض کنید $\prod = s \rightsquigarrow a \to b \to c \to d \rightsquigarrow t$ مسیر یک‌ـحذفی مطلوب از s به t شامل حداقل سه پاره‌خط باشد، که یال bc یال حذفی و رأس‌های a و b رأس‌های P و نقاط b و c نقاط روی محیط P هستند. سپس، یال حذفی bc یا عبور می‌کند از رأس a که رأس P است یا نقاط $\{a,b,c\}$ یا نقاط $\{b,c,d\}$ بر امتداد یک خط واقع‌اند.

برهان. از طریق برهان خلف اثبات می‌نماییم. فرض کنید $\prod = s \rightsquigarrow a \to b \to c \to d \rightsquigarrow t$ مسیر یک‌ـحذفی مطلوب که یال bc از رأس‌های P عبور نکند و هیچکدام از $\{a,b,c\}$ یا هیچکدام از $\{b,c,d\}$ بر امتداد یک خط نیستند. بستگی به شیب یال‌های ab و cd از $\prod$، باید دو مورد را در شکل‌های ۲.۳ بررسی کنیم.
این را در نظر بگیرید که یال‌های ab و cd از $\prod$ شیب‌های متفاوتی دارند (یعنی ab و bc و cd مسیر محدب را تشکیل می‌دهند). در این مورد، یال حذفی bc می‌تواند به موقعیت جدید $b'c'$ انتقال یابد، موازی با bc (شکل ۲.۳ قسمت a را ببینید)، بطوریکه $b'c'$ یک یال حذفی است و مسیر جدید $\prod' = s \rightsquigarrow a \to b' \to c' \to d \rightsquigarrow t$ کوتاهتر از مسیر $\prod$ است. این موضوع با می‌نیمم بودن $\prod$ تناقض دارد. توجه داشته باشید که می‌توانیم این حرکت را تا زمانی که bc یا یکی از رأس‌های P را لمس کند یا b و c و a و یا d در امتداد یک خط باشند. □



شکل ۲.۳: تصویر اثبات لم ۱.۳.۳

نقشه کوتاهترین مسیر از s به t را که به ترتیب SPM_s و SPM_t نامیده، استفاده کرده تا مسیر یک‌ـحذفی مطلوب را محاسبه کنیم. نقشه‌های کوتاهترین مسیر به طور معمول استفاده می شوند که کوتاهترین مسیر از همه‌ی نقاط در P از s که در P بمانند را محاسبه کنیم. این نقشه‌ها محدوده‌ی چندضلعی را به فواصل باز تقسیم می‌کنند که به آنها ''بخش‌های جزیی'' گفته می‌شود. هیچ یک از بخش‌های جزیی حاوی هیچ نقطه‌ای از P نیست.

تعریف ۱.۳.۳. هر بخش جزیی وابسته به یک رأس چندضلعی است، که به آن رأس منبع گفته می‌شود. به گونه‌ای که برای هر نقطه از آن بخش، کوتاهترین مسیر از s از رأس منبع آن است.

SPM_sها را می‌توان بصورت یک درخت تجسم کرد. ریشه آن s است، و بخش‌های جزیی آن برگ‌ها هستند.
رأس‌های P ممکن است به صورت گره‌های برگی یا غیر برگی در درخت ظاهر شود. هر گره P از SPM_sها، که یک رأس P است، طول کوتاهترین مسیر را از s به p در داخل چندضلعی ذخیره می‌کند. یک ساختار داده مشابه برای SPM_t نیز آماده شده است.

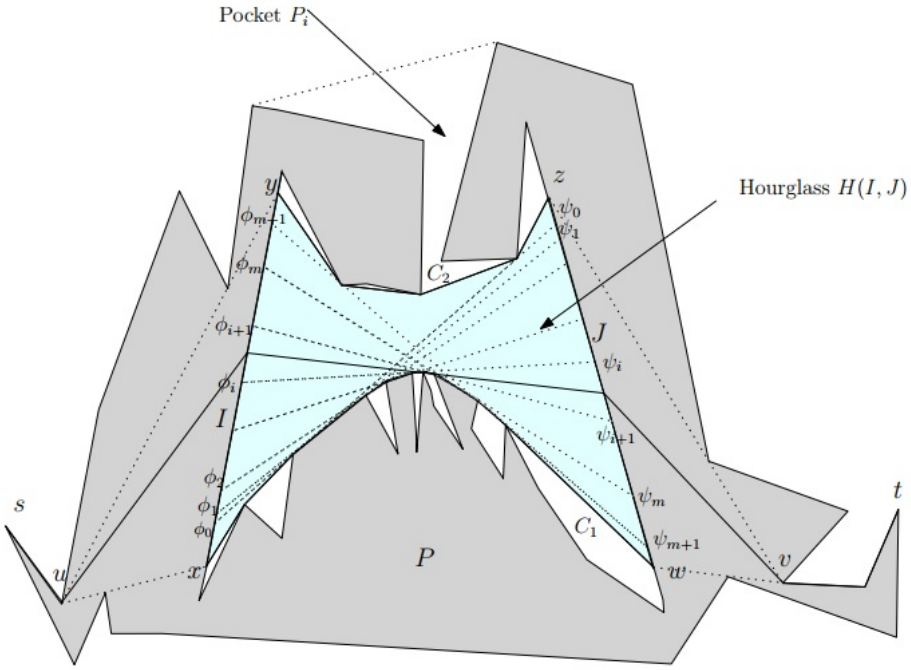
لم ۲.۳.۳. پس از یک پردازش زمان خطی، طول کوتاهترین مسیرهای $\prod_{in}(s,p)$ و $\prod_{in}(t,p)$ برای هر نقطه p روی یک بخش جزیی در زمان $O(1)$ محاسبه می‌شود.

پوسته محدب $CH(P)$ از P را در نظر بگیرید. هر یال $CH(P)$، که یالی از P نیست، بسته را نسبت به P به صورت زیر تعریف می‌کنیم.

تعریف ۲.۳.۳. بسته چندضلعی ساده P به کمک یال e از $CH(P)$ تعریف می‌شود که یال P نیست. این منطقه چندضلعی خارج از P اما داخل $CH(P)$ است. با e و دنباله‌ای از یال‌های متوالی P محدود می‌شود که اولین و آخرین یال دنباله در دو نقطه انتهایی e رخ می‌دهد. دو نقطه انتهایی e به عنوان مرزهای بسته نامگذاری خواهد شد.

فرض کنید این بسته‌ها با $P_1, P_2, ..., P_k$ نشان داده شوند. هر بسته به تنهایی متناظر با یال $CH(P)$ خارج از چندضلعی ساده P تعریف می‌شود. فرض کنید $\prod = s \rightsquigarrow a \to b \to c \to d \rightsquigarrow t$ یک مسیر یک‌ـحذفی شامل حداقل سه پاره‌خط که در لم ۱.۳.۳ بیان شده است. در این نماد، a و b رأس‌های P بوده، bc یال حذفی، که b و c نقاط روی مرز P هستند. فرض کنید b به بخش جزیی I از SPM_s تعلق داشته و فرض کنید c به بخش جزیی J از SPM_t تعلق داشته باشد. توجه داشته باشید که a منبع I در SPM_s بوده و d منبع J در SPM_t است. مشاهده کنید که $\prod$ از پنج قسمت تشکیل شده است، یعنی قسمت اول آن $\prod_{in}(s,a)$ و قسمت دوم آن $\prod_{in}(t,d)$ و قسمت سوم پاره‌خط bc بطور کامل درون بسته‌های p_i قرار دارند، جایی که b و c روی مرز P هستند، قسمت چهارم پاره‌خط ab در P و قسمت پنجم پاره‌خط cd در P است. ممکن است پاره‌خط های ab و bc در امتداد یک خط باشند و یا پاره‌خط‌های bc و cd در امتداد یک خط باشند.
فرض کنید $I = [xy]$ و $J = [wz]$ دو بخش جزیی از بسته P_i باشند، که منبع آنها به ترتیب رأس‌های

u و v با نسبت به ترتیب SPM_s و SPM_t. می‌خواهیم مسیر یک‌ـحذفی مطلوب بین s و t را محاسبه کنیم، طوریکه یال حذفی bc محدود به $b \in I$ و $c \in J$ است. فرض کنید که در تقاطع مرز P_i خلاف جهت عقربه‌های ساعت با شروع در y، نقاط x و y و z به ترتیب ظاهر می‌شوند (شکل ۳.۳ را ببینید). کوتاهترین مسیرهای بین x و w و بین y و z را محدود کنید تا در P_i قرار بگیرید. این دو کوتاهترین مسیر به همراه بخش‌های جزیی I و J یک ساعت شنی $H(I,J)$ [1] را در P_i تعریف می‌کنند. اگر کوتاهترین مسیر مربوطه هیچ رأس به اشتراک نگذارد، گفته می‌شود ساعت شنی باز است. در غیر این صورت بسته است. مشاهدات زیر را بر اساس ارتباط خوب بین کوتاهترین مسیرها و ساعت شنی انجام می‌دهیم.



شکل ۳.۳: پردازش یک جفت از بخش‌های جزیی (I,J)

نتیجه ۲.۳.۳. برای دو بخش جزیی I و J از بسته‌ی P_i، وجود دارد یک پاره‌خط واصل بین I و J که بطور کامل درون P_i قرار گرفته اگر و فقط اگر $H(I,J)$ باز است.

برای جفت بخش‌های جزیی I و J در بسته‌ی P_i، ابتدا بررسی می‌کنیم که $H(I,J)$ با محاسبه کوتاهترین مسیر مربوط به P_i باز است یا خیر. اگر $H(I,J)$ باز باشد، در اینصورت گوییم که بخش جزیی I و J با اعتبار هستند، در غیراینصورت آنها بی اعتبارند. از این پس فرض کنید که بخش‌های جزیی معتبر هستند. در این حالت کوتاهترین مسیر بین x و w در P_i زنجیره محدب است (آن را C_1 بنامید). به طور مشابه، کوتاهترین مسیر بین y و z در P_i یک زنجیره محدب است (آن را C_2

[1] Hourglasses

بنامید).
برای محاسبه یال حذفی bc، که $b \in I$ و $c \in J$، به صورت زیر ادامه می‌دهیم. بخش‌های جزیی I و J را به فواصل تقسیم می‌کنیم، به طوری که برای همه نقاط در یک بازه، رأس که در C_1 (و C_2) مماس است یکسان است. این کار با اسکن C_1 با شروع از x و خطوط ترسیم که با یال‌های C_2 تراز شده است، بدست می‌آید. خطوطی که C_2 را قطع می‌کنند نادیده گرفته می‌شوند و سایر مواردی که wz را قطع می‌کنند حفظ می‌شوند.(خطوطی که wz را قطع می‌کنند با یال‌های متوالی C_1 مطابقت دارند.) فرض کنید این خطوط xy را به ترتیب در نقاط $\phi_1, \phi_2, ..., \phi_m$ و wz را به ترتیب در نقاط $\psi_1, \psi_2, ..., \psi_m$ قطع کنند (شکل ۳.۳ را ببینید). همچنین یک جفت مماس مشترک C_1 و C_2 ترسیم می‌کنیم. فرض کنید این خطوط xy را به ترتیب در نقاط ϕ_0, ϕ_{m+1} و wz را به ترتیب در نقاط ψ_0, ψ_{m+1} قطع کنند. به کمک لم ۱.۳.۳، یال حذفی کوتاهترین مسیر یک‌ـحذفی که I را در بازه‌ی $[\phi_i, \phi_{i+1}]$ و J را در بازه‌ی $[\psi_i, \psi_{i+1}]$ قطع کنند، از همان رأس w از P عبور خواهد کرد.
برای هر بازه‌ی $[\phi_i, \phi_{i+1}]$ و جفت مربوط به آن یعنی بازه‌ی $[\psi_i, \psi_{i+1}]$ می‌توان طول کوتاهترین مسیر یک‌ـحذفی با عملکرد مطلوب که با اضافه کردن طول (آ) $\prod_{in}(s, u)$، که u رأس منبع از I در SPM_s است، (ب) $\prod_{in}(t, v)$ که v رأس منبع از J در SPM_t، (پ) پاره‌خط bc که $b \in [\phi_i, \phi_{i+1}]$ و $c \in [\psi_i, \psi_{i+1}]$، (ت) پاره‌خط ub، (ث) و پاره‌خط cv، را محایبه کرد. زمانی که مکان $b \in [\phi_i, \phi_{i+1}]$ را ثابت کنیم، سایر موارد را می‌توان در زمان $O(1)$ از SPM_s و SPM_t تعیین کرد. به هر حال برای هر $b \in [\phi_i, \phi_{i+1}]$، کمترین مقدار $|ub| + |bc| + |cv|$ در زمان $O(1)$ محاسبه می‌شود. از این رو طول کوتاهترین مسیر یک‌ـحذفی محدود به فواصل $b \in [\phi_i, \phi_{i+1}]$ و $c \in [\psi_i, \psi_{i+1}]$ در زمان $O(1)$ محاسبه می‌شود. همان روش برای محاسبه کوتاهترین مسیر یک‌ـحذفی که از رأس C_2 عبور می‌کند، استفاده می‌شود. سرانجام، کوتاهترین مورد برای یک جفت معتبر از بخش‌های جزیی (I, J) در نظر گرفته شده است. از این رو، طول کوتاهترین مسیر یک‌ـحذفی به I و J را می‌توان در زمان متناسب با تعداد رأس بسته‌ي P_i محاسبه کرد. با در نظر گرفتن تمام جفت‌های ممکن بخش‌های جزیی، می‌توانیم کوتاهترین مسیر یک‌ـحذفی از s تا t عبور از بسته‌ی P_i را شناسایی کنیم. با تکرار این محاسبه برای هر بسته، می‌توانیم کوتاهترین مسیر یک‌ـحذفی را بین s و t محاسبه کنیم. نتیجه را در قضیه زیر خلاصه می‌کنیم.

قضیه ۱.۳.۳. کوتاهترین مسیر یک‌ـحذفی بین یک جفت نقاط در چندضلعی ساده P شامل تعداد n رأس در زمان $O(n^3)$ و با استفاده از فضای $O(n)$ محاسبه می‌شود.

برهان. کوتاهترین مسیر یک‌ـحذفی شامل یک پاره‌خط مستقیم بین s و t، یا یک مسیر شامل دو پاره‌خط، یا یک مسیر شامل سه یا بیشتر پاره‌خط است.
ابتدا معتبر بودن مسیر یک‌ـحذفی پاره‌خط st را آزمایش می‌کنیم. این کار می‌تواند به کمک وارسی مرز P و کنترل تعداد تقاطع‌های بین مرز و پاره‌خط st،در زمان $O(n)$ انجام شود. حال موردی را در نظر بگیرید که کوتاهترین مسیر حذفی که از دو پاره‌خط تشکیل شده باشد. نقطه برگشت مسیر نمی‌تواند در قسمت بیرونی P باشد. علاوه بر این، نقطه برگشت باید در یک رأس P قرار داشته باشد،

در غیر این صورت طول مسیر می‌تواند به بهینگی نزدیک‌تر شود. برای هر رأس v از P، می‌توان از ساختار داده پرتاب پرتویی استفاده کرد که پی ببریم آیا پاره‌خط sv و vt حداکثر یکبار از نمای خارجی P عبور می‌کند. در میان همه مسیرهای دو پاره‌خطی معتبر، یک مورد را پیدا می‌کنیم که دارای حداقل طول باشد. ساختار داده پرتاب پرتویی به زمان $O(n)$ نیاز دارد و برای هر رأس v می‌توانیم بفهمیم که sv یا tv بیش از یک بار در زمان $O(\log n)$ قطع می‌کند.
اگر کوتاهترین مسیر یک‌ـحذفی شامل سه یا بیشتر پاره‌خط باشد، لم ۱.۳.۳ را اتخاذ می‌کنیم. نقشه‌های کوتاهترین مسیر SPM_s و SPM_t می‌توان در زمان و مکان $O(n)$ توسط الگوریتم‌ها محاسبه کرد. هر بسته‌ی P_i را بطور جداگانه پردازش می‌کنیم. فرض کنید تعداد رأس‌ها در P_i برابر n_i، تعداد بخش‌های جزیی در SPM_s و SPM_t به ترتیب برابر μ و ν باشد. جفت‌های $\mu \times \nu$ را از بخش‌های جزیی (I, J) در نظر گرفته‌ایم، بطوریکه $I \in SPM_s$ و $J \in SPM_t$. برای هر جفت (I, J)، همه‌ی P_i را در زمان $O(n_i)$ پیمایش کرده تا زنجیره‌های محدب C_1 و C_2 را تشکیل دهیم. اگر یک جفت از بخش‌های جزیی (I, J) معتبر باشند، C_1 و C_2 را با هم ادغام کرده و دوباره پیمایش را انجام می‌دهیم تا دو تکه I و J را در بدترین حالت حداکثر فواصل n_i تقسیم بندی کنیم. همانطور که قبلا ذکر شد، زمان پردازش هر جفت فواصل $[\phi_i, \phi_{i+1}] \in I$ و $[\psi_i, \psi_{i+1}] \in J$ نیازمند زمان $O(1)$ است، پردازش جفت بخش‌های جزیی (I, J) در بسته P_i نیازمند زمان $O(n_i)$ است. نتیجه نهایی پس از در نظر گرفتن تمام جفت‌های ممکن در مورد بخش‌های جزیی معتبر گزارش می‌شود. نتیجه از این واقعیت بهره می‌گیرد که تعداد جفت بخش‌های جزیی (I, J)، $I \in SPM_s$ و $J \in SPM_t$ برابر $O(n^2)$ با توجه به تمام بسته‌های P است. □

۴.۳ مسئله کوتاهترین مسیر یک‌ـحذفی در چندضلعی با خطوط مستقیم (مستطیل شکل)

چندضلعی مستطیلی شکل[2] P و جفت نقاط $s, t \in P$ داده شده است، مسیر مستطیلی شکل[3] یک‌ـحذفی یک مسیر مستطیلی از s به t بطوریکه یک یال e افقی یا عمودی که از P خارج شود، و تقاطع e با $\overline{P}$ حداکثر در یک پاره‌خط باشد. هدف ما محاسبه یک مسیر مستطیلی یک‌ـحذفی با حداقل طول است. از نماد $\Psi_{in}(s, t)$ برای نمایش کوتاهترین مسیر مستطیلی از s به t که داخل P قرار بگیرد استفاده کرده، همینطور از نماد $\Psi_{one}(s, t)$ برای نمایش کوتاهترین مسیر یک‌ـحذفی مستطیلی از s به t استفاده می‌کنیم. خط یا پاره‌خط ℓ را همراستا با یال e از چندضلعی P گفته هرگاه بخشی از ℓ یا بخشی از e همپوشانی داشته باشد.

لم ۱.۴.۳. اگر s و t روی مرز چندضلعی P باشند، آنگاه کمترین مسیر مستطیلی از s به t وجود دارد بطوریکه هر پاره‌خط از این مسیر با بعضی از یال‌های چندضلعی P همراستا باشند.

[2] Rectilinear Polygon
[3] Rectilinear Path

برهان. فرض کنید Ψ کوتاهترین مسیر مستطیلی از s به t باشد، و دارای یک یال عمودی e است که هیچ بخشی از آن با یال‌های چندضلعی همراستا نیست. در اینجا باید دو مورد را در نظر بگیریم که آیا یال عمودی ℓ' از Ψ همراستا با بعضی یال‌های P است یا خیر. در حالت مثبت، اگر یال e را به صورت افقی به سمت e' حرکت کنیم، طول مسیر بدون تغییر باقی می‌ماند. بعد از همراستا شدن با e'، نتیجه را نگه می‌داریم. در حالت منفی، یال ترکیبی $e \bigoplus e'$ در حال حرکت در امتداد یال عمودی بعدی خود می‌باشد و طول کل آن را بدون تغییر نگه می‌دارد. به طور مشابه ادامه یافته، نتایج را نگه می‌داریم. به طور مشابه، اگر یک یال افقی از Ψ با یک یال افقی از چندضلعی همراستا نباشد، می‌تواند به روشی مشابه اصلاح شود تا با یال P همراستا گردد. □

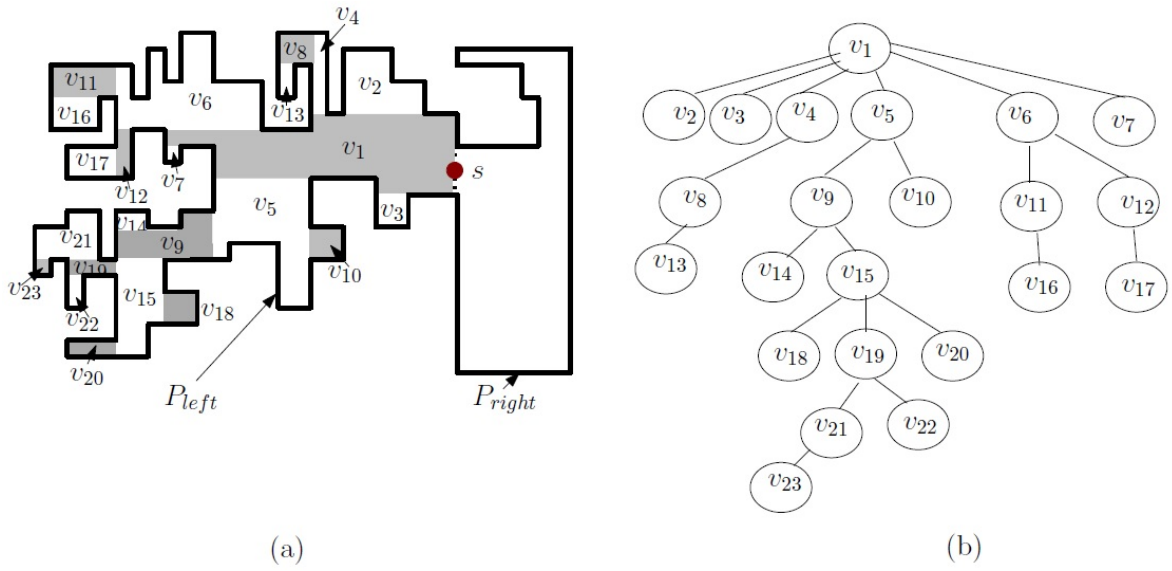
اگر s و t روی مرز P نباشند، سپس کمترین مسیر مستطیلی از s به t وجود دارد بطوریکه تمام یال‌ها با یال‌های چندضلعی P همراستا شده باشند به جز احتمالاً یال‌هایی که روی s یا t قرار دارند.

لم ۲.۴.۳. اگر مسیر مستطیلی یک‌ـ حذفی وجود داشته باشد که طول آن کوتاهتر از $\Psi_{in}(s,t)$ باشد آنگاه می‌توان کوتاهترین مسیر مستطیلی یک‌ـ حذفی $\Psi_{one}(s,t)$ را بدست آورد بطوریکه یال حذفی از امتداد یک یال P در خارج چندضلعی حاصل می‌شود.

برهان. فرض کنید Ψ کوتاهترین مسیر مستطیلی یک‌ـ حذفی باشد که یال $e = [a,b]$ افقی باشد، و با هر یال P همراستا نباشد. یال‌های عمودی مجاور آن در a و b باید به صورت عمودی در جهت مخالف باشد. در غیر اینصورت، می توان با حرکت $[a,b]$ در آن جهت (معمول)، طول مسیر را کاهش دهیم. اکنون، $[a,b]$ را بالاتر یا پایین‌تر نگه می‌داریم تا همان طول را حفظ کنیم تا اینکه با یال افقی قبلی یا بعدی همراستا شود، آن را e' از Ψ می‌نامیم. اگر یال افقی $e \oplus e'$ همراستا با یک یال از چندضلعی نباشد، سپس می‌توان آن را در یک جهت حرکت داد (بالا یا پایین) طول آن بدون تغییر تا زمانی که با یال دیگری از Ψ همراستا شود یا آن را یالی از چند ضلعی قرار می دهد. در اولین مورد، روند جابجایی یال ادغام شده ادامه می‌یابد و در حالت دوم، روند اثبات نتیجه متوقف می‌شود. □

ابتدا کوتاهترین مسیر مستطیلی به روش کوتاهترین مسیر مستطیلی $\Psi_{in}(s,t)$ در چندضلعی مستطیلی P را محاسبه می‌کنیم. یک جفت از پاره‌خط‌های متعامد $h(s)$ و $v(s)$ (به ترتیب $h(t)$ و $v(t)$) در نقطه s (به ترتیب t) را رسم می‌کنیم. فرض کنید چندضلعی به دو قسمت P_{left} و P_{right} در دو طرف $v(s)$ تقسیم‌بندی شود. از نماد $\hat{P}_{left}$ و نماد $\hat{P}_{right}$ برای نمایش رأس‌های چندضلعی به ترتیب P_{left} و P_{right} استفاده می‌شود. افراز تصویری P_{left} و P_{right} را محاسبه کنید. هر پنجره از یک نمودار تصویری، پایه یک نمودار تصویری مجاور است، و ارتباط مجاورت بین نمودار تصویری در هر طرف از $v(s)$ را می‌توان به عنوان یک درخت جهتدار نشان داد. از نماد τ_{left} برای نمایش ”نمودار درختی“ برای چندضلعی P_{left} استفاده می‌کنیم. نمودار گره $v \in \tau_{left}$ با نماد $H(v)$ نمایش داده می‌شود. در شکل ۴.۳، یک افراز نمودار تصویری و یک نمای درختی برای چندضلعی P_{left} نشان داده شده است. اینک می‌خواهیم روشی برای محاسبه کوتاهترین مسیر از s به t در نظر بگیریم. فرض کنید $t \in P_{left}$. از $H(s)$ و $H(t)$ برای نمایش نمودار تصویری شامل به ترتیب s و

t استفاده می‌کنیم، ریشه‌ّای نمودار درختی P_{left} را $H(s)$ می‌نامیم. کوتاهترین مسیر از t به s در نمودار درختی از $H(t)$ به $H(s)$ حرکت خواهد کرد، و و مسیر مربوطه همانطور که در شکل ۴.۳ نشان داده شده است در طرح یک یال نمودار تصویری به پایه آن شکسته (خم) خواهد شد. افراز نمودار تصویری P و محاسبه‌ی نمودار درختی نیازمند زمان $O(n)$ است. نمودار تصویری بر اساس نمودار تصویری اصلی و منبع خود برنامه ریزی می‌شود. اینها یک مجموعه نقاط اشنایدر[۴] به نام Q_{left} را ایجاد می‌کنند. شکستگی در رأس‌های $\hat{P}_{left}$ از چندضلعی P_{left} و نقاط اشنایدر در Q_{left} ممکن است اتفاق بیافتد.



شکل ۴.۳: بخش (a) تجزیه نمودار تصویری P_{left} و بخش (b) نمودار درختی قسمت (a).

اکنون، اگر s و t در نمودار تصویری یکسان قرار داشته باشند، کوتاهترین مسیر آنها یک محور موازی Lـمسیر (یک Lـمسیر شامل یک پاره‌خط افقی و عمودی در یک نقطه خم (چرخش) مشترک است) است که آنها را به هم وصل می‌کند. در غیر این صورت، اولین خم مسیر از s به t در تصویر t از پایه $H(t)$ قرار خواهد گرفت. دفعه بعدی خم‌ها در نقاط $\hat{P}_{left} \cup Q_{left}$ هستند. نمودار $G = (V, E)$ را می‌سازیم، بطوری که

$$V = \hat{P}_{left} \cup Q_{left} \cup \{s, s_i, i \in \{N, S, E, W\}\} \cup \{t, t_i, i \in \{N, S, E, W\}\}$$

که $\{s_i, t_i, i \in N, S, E, W\}$ پیش‌بینی‌های متعامد از s و t در مرز P در چهار جهت (یعنی به ترتیب شمال، جنوب، شرق، غرب). یال $e \in E$ به جفت نقاط $\alpha, \beta \in \hat{P}_{left} \cup Q_{left}$ می‌پیوندد، بطوریکه α و β دارای مؤلفه x (یا y) یکسان هستند و هیچ نقطه دیگری از $\hat{P}_{left} \cup Q_{left}$ در بازه‌ي $[\alpha, \beta]$ نیست. کوتاهترین مسیر از s به t می‌تواند به کمک الگوریتم دایکسترا روی گراف G محاسبه گردد.

[4] Steiner Points

مشاهده کنید که $|V| = O(n)$ از آنجایی که هر پنجره حداکثر دو عضو در Q دارد و تعداد پنجره‌های موجود در نمودار تصویری از P_{left} برابر است با تعداد گره‌های در τ_{left}، که تعداد رأس‌های P_{left} حداکثر برابر $|\hat{P}_{left}|$ است. تعداد یال‌ها اغلب برابر $O(n)$ است زیرا رأس P حداکثر ۲ یال دارد و هر عضو Q حداکثر در ۳ یال اتفاق می‌افتد. الگوریتم دایکسترا در مدت زمان $O(n \log n)$ اجرا می‌کند.
اکنون یک روش دیگر برای محاسبه مسیر یکـحذفی با کمترین طول را شرح می‌دهیم. به کمک ۲.۴.۳ یال حذف شده می‌تواند پسوند $\hat{e}$ یک یال e از چندضلعی P باشد. پسوند $\hat{e}$ مرز P را در بیرون از آن ملاقات می‌کند. توجه داشته باشید که پسوند e می‌تواند با یال e' در داخل به هم برخورد کنند، سپس به بیرون برود، و سپس آن را با یال دیگر e'' قطع کند. در اینجا لازم به ذکر است که یال حذفی از مرز P_{left} ممکن است به نقطه‌ای در مرز P_{right} برسد. بنابراین در قاعده سازی محاسبه‌ی گراف $\Psi_{one}(s,t)$، نیاز داریم که جفت P_{left} و P_{right} را با هم کنترل کنیم. مانند مسأله کوتاهترین مسیر، گراف $G_{one} = (V_{one}, E_{one})$ را ایجاد می‌کنیم، بطوریکه

$$V_{one} = \hat{P} \cup Q \cup R \cup \{s, (s_i, \sigma_i), i \in \{N,S,E,W\}\} \cup \{t, (t_i, \tau_i), i \in \{N,S,E,W\}\}$$

اینجا (آ) $\hat{P} = \hat{P}_{left} \cup \hat{P}_{right}$، (ب) Q مجموعه رأس‌هایی است که به ترتیب از تصویر اساسی نمودار تصویری در پایه منبع(سرچشمه) تشکیل شده است، همانطور که برای مسأله کوتاهترین مسیر توضیح داده شده است، (پ) R تصاویر متعامد رأس‌های $\hat{P}$ روی مرز P از بیرون آن است، (ت) s_i (به ترتیب t_i) تصاویر متعامد از s (به ترتیب t) روی مرز چندضلعی از درون در طرف iـام است و (ث) σ_i (به ترتیب τ_i) نقطه تقاطع خط گسترش یافته از s (به ترتیب t) با مرز چندضلعی از خارج در طرف iـام است. همینطور برای یال‌ها داریم $E_{one} = \hat{E} \cup E_{violation}$. یال‌های موجود در $\hat{E}$ با توجه به رأس‌های $\hat{P} \cup Q$ در کوتاهترین مسیر تعریف شده و برچسب گذاری تحت عنوان خوب می‌شوند. یال‌های $E_{violation}$ مجموعه همه‌ی یال‌های حذفی و با برچسب تحت عنوان بد می‌باشند.
برای هر رأس P دو یال حذفی ممکن است وجود داشته باشد. هر یال $E_{violation}$ به یک رأس $\hat{P}$ وصل شده است و تصاویر متعامد آن ($\in R$) روی مرز P از طرف خارج است. رأس‌های درون R و یال‌های درون $E_{violation}$ از طریق جابجایی یک خط افقی (به ترتیب عمودی) روی چندضلعی در زمان $O(n \log n)$ تولید می‌شوند. توجه داشته باشید، یک یال در $E_{violation}$ ممکن است از یال‌های بسیاری در $E_{violation}$ عبور کند.
بنابراین، بر خلاف G، ممکن است G_{one} یک گراف مسطح نباشد. به هر حال، (آ) تعداد رأس‌های درون R برابر $O(n)$ است زیرا هر یال در P حداکثر در دو جهت گسترش می‌یابد، و (ب) تعداد یال‌های درون E_{one} بصورت $E_{one} = O(n)$ است زیرا هر رأس P در حداکثر دو یال از $E_{violation}$ می‌تواند تعریف شود. الگوریتم ۱.۲.۳ را اجرا می‌کنیم، با تعداد یال‌های حذفی $k = ۱$، که $\Psi_{one}(s,t)$ را محاسبه کند و نتیجه بعدی را بدست آوریم.

قضیه ۱.۴.۳. فرض کنید یک چندضلعی مستطیلی ساده P با تعداد n رأس و دو نقطه s و t در داخل آن، کوتاهترین مسیر مستطیلی یکـحذفی از s تا t را می‌توان در زمان $O(n \log n)$ با استفاده از فضای $O(n)$ محاسبه کرد.

فصل ۴

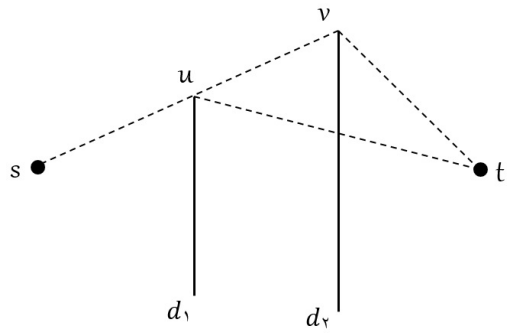
kـ مسیر و ویژگی‌های آن

در این فصل به تعریف گراف رؤیت پذیر و قوانین آن می‌پردازیم و ویژگی‌های دو مسیر متمایز با ویژگی‌های مشترک را بیان خواهیم کرد. همینطور با استفاده از مفهوم رؤیت پذیر می‌خواهیم یک مسیر را به دو یا چند مسیر جزیی تقسیم کنیم.
مطالب این فصل را از مقاله [۱۸] استخراج کرده‌ایم.

۱.۴ تعریف kـ مسیر

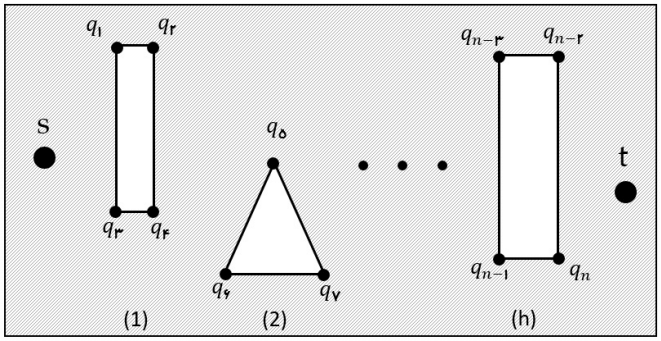
مجموعه‌ای از موانع چندضلعی در صفحه و عدد صحیح k داده شده است، کدام k مانع را حذف کنیم تا کوتاهترین مسیر بین دو نقطه s و t داده شده، بدست آید؟

به عبارت دیگر، کوتاهترین مسیری که اجازه عبور از درون حداکثر k مانع داریم کدام است؟ مسیری که در آن از درون حداکثر k مانع عبور کنیم یک kـ مسیر می‌نامیم. بنا به قرار داد مسیر بدون مانع را یک صفرـ مسیر می‌نامیم. به عنوان مثال در شکل ۱.۴، دو خط d_1 و d_2 به عنوان موانع و نقطه s نقطه آغازین و نقطه t نقطه پایانی است. در اینصورت مسیر $s \to u \to v \to t$ کوتاهترین مسیر از s به t بوده که از درون هیچ مانعی عبور نمی‌کند (کوتاهترین ۰ـ مسیر از s به t)، همینطور مسیر $s \to u \to t$ کوتاهترین مسیر از s به t بوده که مجاز است از یک مانع عبور کند (کوتاهترین ۱ـ مسیر از s به t).
بطور دقیق تر، فرض کنید چندضلعی P شامل h مانع محدب در صفحه است که تعداد کل رأس‌های



شکل ۱.۴: نمایش کوتاهترین ۰ـ مسیر و ۱ـ مسیر از s به t

چندضلعی n تا بوده بطوریکه همگی درون مستطیل R قرار گرفته باشند. (به شکل ۲.۴ توجه کنید) متمم موانع درون R را فضای آزاد می‌نامیم. فرض کنید نقطه شروع s در فضای آزاد داده شده و ثابت است، می‌خواهیم کوتاهترین kـ مسیر برای $k \leq h$ به دیگر نقاط این فضای آزاد را محاسبه کنیم.



شکل ۲.۴: مستطیل R با h مانع محدب و n رأس، قسمتهای هاشورخورده فضای آزاد می‌باشند

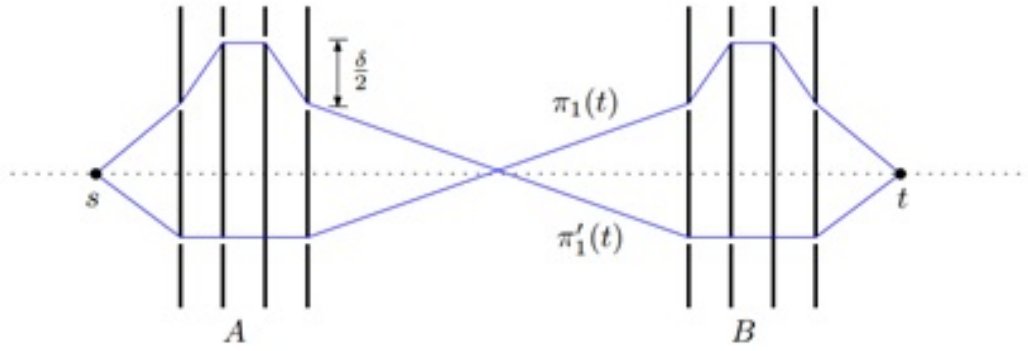
توصیف این کوتاهترین مسیرها که می‌توانند بصورت افرازی از صفحه باشند، نقشه کوتاهترین kـ مسیر نامیده می‌شود. از نماد $\pi_k(t)$ برای نمایش کوتاهترین kـ مسیر از s به t، که s نقطه ثابت ابتدایی است، استفاده می‌کنیم و طول این مسیر را با $d_k(t)$ نشان می‌دهیم.

۲.۴ ویژگی های kـ مسیر

فرض کنید یک نقطه p در فضای آزاد داده شده است، کوتاهترین kـ مسیر که آن را با نماد $\pi_k(t)$ نشان می‌دهیم، مسیری است که نقطه s را به p وصل می‌کند و از درون حداکثر k مانع عبور کند، و کمترین طول را بین همه مسیرها دارد. در ادامه، به مسیرهایی که دقیقا k مانع را قطع می‌کنند، به

عنوان $(= k)$ـ مسیر اشاره می‌کنیم. با این تصور که مسئله می‌تواند در زمان چندجمله ای (درجه دوم) حل شود، با استفاده از جستجوی شبه دایکسترا روی ''گراف رؤیت پذیری'' شروع می‌کنیم.

قضیه ۱.۲.۴. [۱۹]فرض کنید P یک چندضلعی است که دارای h مانع محدب با تعداد n راس است و فرض کنید نقطه شروع (آغاز) s و نقطه هدف (نهایی) t داده شده است. کوتاهترین kـ مسیر از s به t، در زمان $O((kn + h^2) \log n + kh^2)$ محاسبه می‌شود.



شکل ۳.۴: دو تقاطع ۱ـ مسیر

حل مسئله با استفاده از گراف رؤیت پذیری در بدترین حالت می‌تواند درجه دوم باشد به این دلیل که تعداد موانع می‌تواند $h = \Omega(n)$ باشد. همچنین برای محاسبه کوتاهترین kـ مسیر، محدود به تنها یک نقطه (یا مجموعه ای ثابت از نقاط) در یک زمان است، اگرچه پس از پیش پردازش درجه دوم می‌توان آن را تعمیم داد تا پرس و جوهایی در مدت زمان $O(h(k + \log n))$ را پشتیبانی کند.
نتیجه اصلی در این پایان‌نامه یک الگوریتم برای محاسبه کوتاهترین kـ مسیرها از s تا همه ی نقاط فضای آزاد در یک زمان کمتر از مربعی $O(k^2 n \log n)$ است. این کار را با محاسبه نقشه کوتاهترین kـ مسیر از فضای آزاد انجام می‌دهیم، همچنین یک محدودیت $\Theta(kn)$ روی پیچیدگی‌های ساختاری SPM_k را ثابت می‌کنیم. توجه داشته باشید که طول کوتاهترین kـ مسیر تا یک نقطه، منحصر به فرد است، اگرچه برخی از نقاط (در امتداد نیمسازهای مرزهای ناحیه‌های ساخته شده در نقشه کوتاهترین مسیر) با چندین کوتاهترین kـ مسیر قابل دسترس هستند. برای سادگی کار، با این حال، فرض می‌کنیم که موانع در موقعیت کلی قرار دارند ، به طوری که کوتاهترین kـ مسیر به هر راس مانع، منحصر به فرد است. (در غیر این صورت، اگر برای یک رأس چند kـ مسیر وجود داشته باشد، یکی از آنها را به دلخواه انتخاب می‌کنیم.)
با برجسته کردن یک مشکل در رابطه با کوتاهترین kـ مسیرها شروع می‌کنیم. کوتاهترین مسیرها به دو مقصد مختلف می‌توانند یکدیگر را قطع کنند، که یک مشکل اصلی برای چارچوب دایکسترا پیوسته برای کوتاهترین مسیرهای هندسی است [۱۹]، زیرا این روش بستگی دارد به این واقعیت که دو مسیر کوتاه اقلیدسی از یک مبدأ مشترک نمی‌توانند همدیگر را قطع کنند.

لم ۱.۲.۴. یک حالت برای موانع وجود دارد به طوری که برای دو مقصد t_1 و t_2، کوتاهترین مسیر $\pi_k(t_1)$ و $\pi_k(t_2)$ برای $k > 0$ همدیگر را قطع می‌کنند.

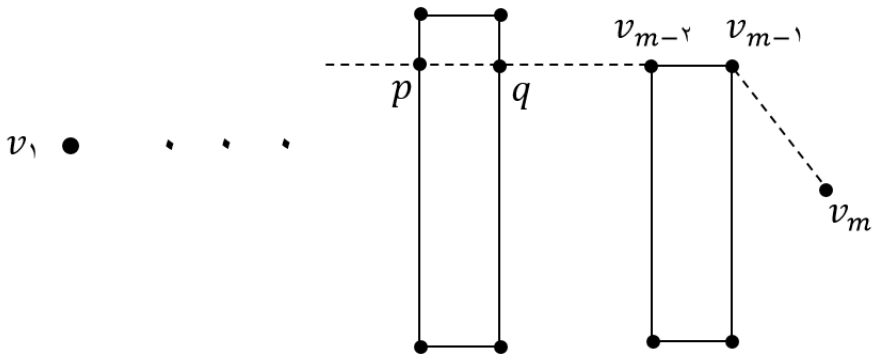
برهان. با توجه به شکل ۳.۴، ساختار دو بسته مانع یکسان A و B موازی محور y ها قرار دارد. هر بسته شامل چهار نوار عمودی با سوراخ است، (حفره های تک نقطه‌ای که نوار اصلی را به نوارهای جداشده تقسیم می‌کند). فضای افقی بین نوارها در بسته‌ها خیلی کم است اما در شکل جدا از هم هستند. نقاط s و t هر دو روی محور x به ترتیب در فاصله ۱ واحد به چپ و راست بسته‌های A و B قرار دارند. نشان می‌دهیم که دو کوتاهترین ۱ـ مسیر از s به t وجود دارد، که یکدیگر را قطع می‌کنند، همانطور که در شکل نشان داده شده است. سپس نتیجه می‌گیریم که با لغزش بسیار اندک t به بالا و پایین دو نقطه انتهایی t_1 و t_2 را با کوتاهترین ۱ـ مسیرها که همدیگر را قطع می‌کنند، به دست می‌آوریم، همانطور که ادعا شده است.
در هر بسته، دهانه‌ها یک گروه بالایی و پایینی تشکیل می‌دهند. در گروه بالایی، نوارهای ۲ و ۳ دارای دهانه در $y = (1 + \delta/2)$ هستند، و نوارهای ۱ و ۴ دارای دهانه‌هایی در $y = 1$ هستند. در گروه پایین، همه به جز نوار ۳ دارای دهانه در $y = 1$ هستند. اگر فاصله بین بسته‌ها برابر D، سپس کوتاهترین ۰ـ مسیر دارای طول $2\sqrt{2} + D + 2\delta$ و کوتاهترین ۲ـ مسیر دارای طول $2\sqrt{2} + D$ است. مسیری که دقیقاً یک تقاطع از یک گروه بالایی دارد حداقل $2\sqrt{2} + D + 3\delta/2$ و کوتاهترین مسیر با یک تقاطع در یک گروه پایین‌تر دارای طول $2\sqrt{2} + D + 2/D + \delta$ $< 2\sqrt{2} + \sqrt{D^2 + 4} + \delta$ است. با انتخاب $D = 10$، و $\delta = 4/D$، کوتاهترین ۱ـ مسیر را می‌توانیم مجبور کنیم تا دقیقاً یک گروه از هر نوع را طی کند. این، دو کوتاهترین kـ مسیرهای $\pi_1(t)$ و $\pi'_1(t)$ که همدیگر را قطع می‌کنند، ایجاد می‌کنند. اکنون فرض کنید t_1 (یا t_2) یک نقطه مقصد باشد که با جابجایی بسیار اندک t به صورت عمودی به سمت بالا (یا به طور عمودی به سمت پایین) بدست می‌آید. سپس به سهولت می‌توان دید که کوتاهترین مسیرهای $\pi_1(t_1)$ و $\pi_1(t_2)$ یکدیگر را قطع می‌کنند.
□

خوشبختانه، همانطور که در این بخش نشان می‌دهیم، کوتاهترین kـ مسیرها همیشه می‌توانند به مسیرهای جزئی که همدیگر را قطع نمی‌کنند، تجزیه شوند که می‌توان چارچوب پیوسته دایکسترا را برای آنها استفاده کرد، برای این منظور مفهوم پارکینگ kـ سطحی را معرفی می‌کنیم. برای رسیدن به این هدف، در ادامه تعدادی از لم ها را بررسی می کنیم.

لم ۲.۲.۴. یک کوتاهترین مسیر با دقیقاً k تقاطع، می تواند به کوتاهترین مسیر با دقیقاً $k-1$ تقاطع، یک پاره خط مستقیم درون یک مانع و کوتاهترین مسیر با صفر تقاطع، تجزیه شود.

برهان. فرض کنید $\pi = (v_1, v_2, ..., v_m)$ یک kـ مسیر از v_1 به v_m باشد. با برگشت به عقب از v_m در امتداد π، v_i اولین رأسی باشد که پاره خط $\overline{v_{i-1}v_i}$ یک یا چند مانع را قطع می‌کند. فرض کنید H نزدیکترین مانع به v_i در امتداد پاره خط $\overline{v_{i-1}v_i}$ باشد. با محدب بودن H پاره خط $\overline{v_{i-1}v_i}$، H را در دو نقطه قطع می‌کند که این دو نقطه را p, q می‌نامیم، و پاره خط $\overline{pq}$ به طور کامل درون H قرار

می‌گیرد. با توجه به بهینه بودن زیر مسیرها، مسیر از v_1 به p کوتاهترین مسیر با دقیقا $k-1$ تقاطع است، با توجه به ساختار گفته شده، پاره خط $\overline{pq}$ درون مانع قرار می‌گیرد و مسیر جزئی q به v_m هیچ مانعی را قطع نمی‌کند.به شکل زیر دقت کنید:



شکل ۴.۴: تفکیک کوتاهترین kـ مسیر از v_1 به v_m

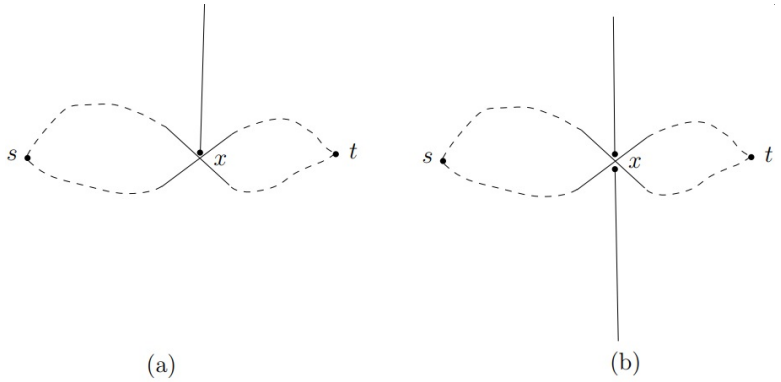
□

نتیجه ۱.۲.۴. در یک کوتاهترین kـ مسیر، پاره‌خط‌های قبل و بعد هر مانع، که مانع را قطع می‌کنند، با پاره خط‌های درون مانع هم‌راستا هستند.

لم ۲.۲.۴ به ما این امکان را می‌دهد که هر $\pi_k(t)$ را به یک $(k-1)$ـ مسیر $\pi_{k-1}(p)$، یک پاره‌خط $\overline{pq}$ و یک زیرمسیر بدون برخورد با موانع بین q و t تقسیم کنیم. دو زیرمسیر آخر را با تعداد موانع قطع شده با قسمت از پیشوند مسیر برچسب گذاری می‌کنیم، و این برچسب‌ها را با پیشوند شمارشی نامگذاری می‌کنند. به عبارت دیگر، پیشوند شمارشی برای مسیر جزئی $\overline{pq}$ برابر $(k-1)$ است و پیشوند شمارشی برای مسیر جزئی از q به t برابر k است. با یک استفاده بازگشتی از لم ۲.۲.۴ می‌توانیم $\pi_k(t)$ را به $2k+1$ مسیر جزئی که برچسب‌های آنها کاهشی نیستند، تجزیه کنیم.
نتیجه کلیدی این تجزیه، لم زیر است که می‌گوید مسیر جزئی با پیشوند شمارشی مشابه نمی‌توانند همدیگر را قطع کنند.

لم ۳.۲.۴. فرض کنید $\pi_k(t)$ و $\pi'_k(t')$ دو مسیر جزئی باشند که پیشوند شمارشی آنها یکسان است. سپس $\pi_k(t)$ و $\pi'_k(t')$ همدیگر را قطع نمی‌کنند.

برهان. اثبات از کاربرد ساده نابرابری مثلث نتیجه می‌شود (با توجه به شکل ۵.۴): اگر دو مسیر جزئی با پیشوند شمارشی یکسان داشته باشیم، چنانچه به یکدیگر برسند (در نقطه t) در اینصورت می‌توانیم پیشوند هر مسیر را به قسمت پسوند مسیر دیگر وصل کنیم و به احتمال زیاد با رسم پاره‌خط میانبر، می‌توان یا حداقل یک مسیر را کوتاهتر کرد یا با همان طول مسیر قبلی ولی بدون برخورد بدست آورد.



شکل ۵.۴: نامساوی مثلثی و عدم برخورد دو زیرمسیر با پیشوند شمارشی یکسان

به عنوان مثال در شکل ۳.۴، یال‌های دو kـ مسیر که همدیگر را قطع کرده‌اند، در مسیرهای جزئی خود، پیشوندهای شمارشی متفاوتی دارند.

□

دو لم بعدی خواص کوتاهترین kـ مسیر را ارائه می‌کنند که در ادامه مفید خواهند بود.

تعریف ۱.۲.۴. نقطه‌ی p از مبدأ s را kـ رؤیت پذیر [1] گوییم اگر پاره‌خط $\overline{sp}$ حداکثر k مانع را قطع کند. یک یال kـ رؤیت پذیر یک کوتاهترین kـ مسیر با دقیقا یک یال است.

لم ۴.۲.۴. اگر p از s، $(k-1)$ـ رؤیت پذیر نباشد، آنگاه مسیر $\pi_k(p)$ باید یک $(=k)$ـ مسیر باشد.

برهان. به کمک برهان خلف، فرض کنید $\pi_k(p)$ از تعداد موانع کمتر از k عبور کند. از آنجایی که p از s، $(k-1)$ـ رؤیت پذیر نیست، $\pi_k(p)$ باید حداقل یک شکستگی داشته باشد. سپس مسیر کوتاهتر را می‌توان با قطع کردن مانعی که باعث این شکستگی شده است، ایجاد کرد و در نتیجه تعداد مسیرها را از یک واحد افزایش می‌دهد. مسیر حاصل،از مسیر $\pi_k(p)$ کوتاهتر است و حداکثر k تقاطع دارد که با بهینگی $\pi_k(p)$ متناقض است.

□

فرض کنید $d_k(p)$ طول کوتاهترین kـ مسیر به نقطه p باشد. واضح است مسیری که j مانع را قطع می‌کند و دارای حداقل دو پاره‌خط است کوتاهتر می‌شود حتی اگر اجازه قطع تعداد موانع بیشتری را داشته باشد. بنابراین، این بدان معنی است که برای هر نقطه p که از s، $(k-1)$ـ رؤیت پذیر نیست، $d_j(p) > d_{(j+1)}(p)$، برای هر $j < k$ برقرار است.

لم ۵.۲.۴. برای هر نقطه p که $(k-1)$ـ رؤیت پذیر از s نیست، طول کوتاهترین jـ مسیرها دنباله‌ای نزولی را تشکیل می‌دهند:

$$d_0(p) > d_1(p) > ... > d_j(p) > ... > d_k(p)$$

[1] $k-visible$

فصل ۵

نقشه کوتاهترین مسیر، SPM_K

در این فصل به ویژگی‌های kـ مسیر خواهیم پرداخت و با مفاهیم اختصاصی کوتاهترین kـ مسیر از قبیل SPM_K، kـ راس قبلی، kـ پارکینگ و ناحیه V_k آشنا خواهیم شد.
مطالب این فصل را از مقاله [۱۸] استخراج کرده‌ایم.

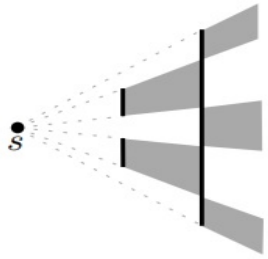
۱.۵ خواص و محدودیت ها

پس از محرز شدن ویژگی‌های پایه ای کوتاهترین kـ مسیر، در حال حاضر بحث خود را از نقشه کوتاهترین kـ مسیر SPM_K آغاز می‌کنیم.

تعریف ۱.۱.۵. **Kـ رأس قبلی:** با توجه به کوتاه‌ترین kـ مسیر $\pi_k(p)$، Kـ رأس قبلی رأس p رأسی از مجموعه رئوس P (از جمله s) که مجاورت به p در $\pi_k(p)$ است، تعریف می‌کنیم. افراز فضای خالی به ناحیه‌های همبند با Kـ رأس قبلی یکسان، کوتاهترین kـ مسیر نامیده می‌شود و با SPM_K نشان داده می‌شود. زیرمجموعه از SPM_K که کوتاهترین مسیر $\pi_k(p)$ برای هر نقطه p دقیقا k مانع را قطع می‌کند، کوتاهترین kـ مسیر نامیده و با $SPM_{=k}$ نشان داده می‌شود. برای مثال به شکل ۱.۵ مراجعه کنید.

برخلاف $SPM_\circ$، که در آن رأس قبلی یک ناحیه همیشه داخل یا روی مرز ناحیه است، رأس قبلی یک ناحیه در SPM_K ممکن است خارج از ناحیه قرار داشته باشد. علاوه بر این، چندین ناحیه در SPM_K ممکن است رأس قبلی یکسان داشته باشند. (شکل ۱.۵ را ببینید) بنابراین نیاز داریم تا

اطلاعات اضافی را با رأس چندضلعی حفظ کنیم تا رابط رأس قبلی را تغییر دهیم. به طور خاص، فرض کنید که v رأس kـ قبلی از رأس P باشد، یعنی رأس مجاور به P در $\pi_k(p)$ باشد. فرض کنید پاره خط $\overline{vp}$ تعداد $(k-i)$ مانع را قطع کند، برای یک مقدار $0 \leq i \leq k$. سپس طول $d_k(p)$ از $\pi_k(p)$ برابر با مجموع طول iـ مسیر به v و طول پاره خط $\overline{vp}$ است.



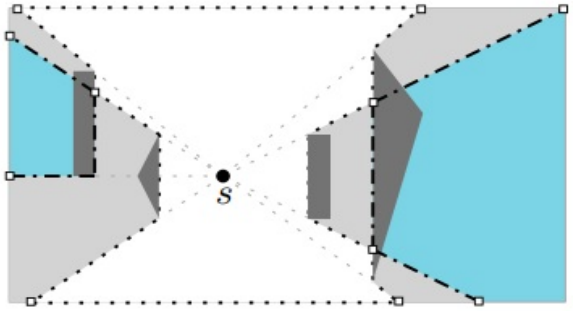
شکل ۱.۵: ۱ـ رأس قبلی همه نقاط در ناحیه سایه دار SPM_1 نقطه s است

باید مقادیر $d_i(v)$ برای تمام راس‌های موانع v و همه اعداد صحیح $i = 0, 1, ..., k$ را ذخیره کنیم. به عبارت دیگر، برای یک نقطه p در $SPM_{=K}$، kـ قبلی از p را توسط زوج (v, i) معرفی می‌کنیم که در آن v راس P و $i \in \{0, 1, ..., k\}$ بطوریکه $d_k(p) = d_i(v) + |\overline{vp}|$ و پاره‌خط $\overline{vp}$ تعداد $(k-i)$ مانع را قطع می‌کند. بنابراین تعداد کل kـ رأس قبلی برابر $O(kn)$ است. با این حال، این به تنهایی تعداد ناحیه‌ها در $SPM_{=k}$ را محدود نمی‌کند چون چندین ناحیه می‌توانند kـ رأس قبلی یکسان داشته باشند. در راستای محدود کردن پیچیدگی ترکیبیاتی نقشه، در ادامه مفهوم kـ رؤیت پذیر را تعریف می‌کنیم.

تعریف ۲.۱.۵. ناحیه V_k (kـ رؤیت پذیر): ناحیه V_K را ناحیه‌ای شامل نقاط kـ رؤیت پذیر تعریف می‌کنیم. نقطه‌ای مانند s برابر است با ناحیه قابل دید از نقطه‌ی s با پاره خطی که حداکثر k مانع را قطع می‌کند.که متشکل از نقاط kـ رؤیت پذیراست (شکل ۲.۵). حالا اگر $\pi_k(p)$ کم‌تر از k مانع را قطع کند، با استفاده از لم ۴.۲.۴، p باید در V_{k-1} قرار گیرد. مسیر $\pi_k(p)$ یک پاره‌خط مستقیم است و kـ رأس قبلی نقطه‌ی p، نقطه s است. همینطور مجموعه یال های ∂V_k یال هایی هستند که همه رئوس k مانع محدب را قطع می‌کنند.

لم ۱.۱.۵. تمام نقاط p به گونه‌ای که $\pi_k(p)$ کمتر از k تقاطع داشته باشد، در V_{k-1} قرار دارند. چنانچه نقطه‌ای خارج از V_{k-1} قرار داشته باشند، SPM_k همان $SPM_{=k}$ است، یعنی نقشه کوتاهترین مسیر با دقیقا k تقاطع دارد.

این لم کار را آسان می‌کند و این امکان را می‌دهد که SPM_k را به دو ناحیه مجزای V_{k-1} و $SPM_{=k}$ تجزیه کنیم. در ادامه، خواص ساختاری این ناحیه‌ها را مورد مطالعه قرار داده و از آنها استفاده می‌کنیم برای محاسبه کران های بالا در اندازه نقشه مربوطه استفاده می‌کنیم.



شکل ۲.۵: ناحیه $V_۲$ کل شکل بوده و ناحیه $V_۱$ قسمت خاکستری و سفید، همینطور ناحیه $V_۰$ قسمت سفید رنگ است.(به عبارت دیگر قسمت آبی رنگ ناحیه $V_۲ \backslash V_۱$، قسمت خاکستری رنگ ناحیه $V_۱ \backslash V_۰$ است)

لم ۲.۱.۵. تعداد یال‌های مرز ∂V_k برابر $O(n+h) = O(n)$ است.

برهان. هر رأس ∂V_k یکی از رأس‌های P است یا تصویری از $۲h$ مماس های از s تا یک مانع از P است. یال های موجود در مجموعه ∂V_k از زیر پاره‌خط‌های مماس یا قسمت‌هایی از مرز موانع هستند. هر راس تصویر شده متعلق به یک پاره خط از ∂V_k خطی همراستا با s است، و نقطه انتهایی x دورتر از s انتهای یک پاره‌خط $\overline{sx}$ است که دقیقاً k مانع را قطع می‌کند. بنابراین، هر یک از $۲h$ مماس حداکثر یک پاره‌خط از ∂V_k و حداکثر دو راس ایجاد می‌کند.

□

لم ۳.۱.۵. تعداد همه‌ی یالهای روی ∂V_i برای $۰ \leq i \leq k$، برابر $O(n+hk)$ است.

با اتصال s به تمامی رئوس در یال $\partial V_{k-۱}$، می‌توانیم به سادگی $V_{(k-۱)}$ را به نواحی پیچیدگی ثابت در SPM_K تجزیه کنیم.

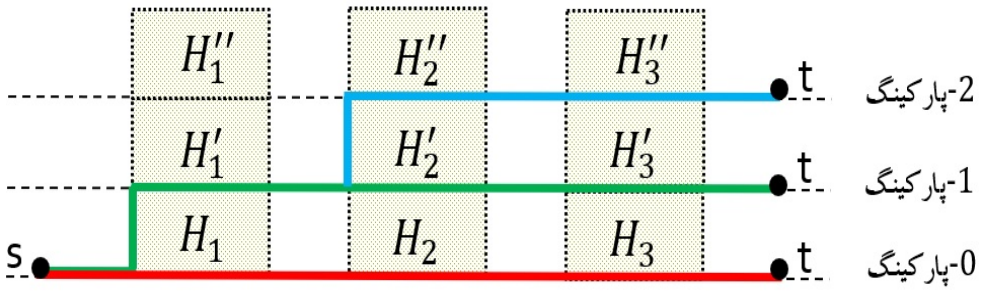
۲.۵ پارکینگ سطح k و ساختار $SPM_{=k}$

اکنون ایده اصلی خود را برای محاسبه کوتاه‌ترین kـ مسیر معرفی می‌کنیم. با توجه به لم ۲.۲.۴ ، یک $(=k)$ـ مسیر از s به نقطه p تشکیل شده است از اتصال یک $(k-۱)$ـ مسیر به مرز یک یال مانع H، یک کوتاهترین مسیر در داخل H و کوتاهترین مسیر در فضای خالی از سوی دیگر H تا p است. این امر یک ساختار افزایشی $SPM_{=k}$ از $SPM_{=(k-۱)}$ را نشان می‌دهد. این ساختار را با استفاده از یک پارکینگ kـ طبقه [۱] توصیف می‌کنیم. ایده این است که چندین نمونه از چندضلعی ورودی ایجاد کرده و آن‌ها را روی هم قرار دهیم که کوتاهترین مسیر در هر طبقه دارای تعداد پیشوند شمارشی

[1] k-garage

یکسان بوده و بنابراین یکدیگر را قطع نمی‌کنند، ناحیه مسطح فضای خالی در طبقه بالا $SPM_{=k}$ است.

تعریف ۱.۲.۵. kـ پارکینگ: ساختار kـ پارکینگ را با قرار دادن k نمونه (یا طبقه) دامنه چند ضلعی ورودی P در بالای یکدیگر، با اتصالات ویژه در مرز موانع ایجاد می‌کنیم. مانع H روی کف طبقه i را به طرف مقابل خود در کف طبقه $i+1$ متصل می‌کنیم به طوری که هر مسیری که به مانع H روی کف طبقه i می‌رسد، می‌تواند تنها در طبقه بالاتر بعدی خارج شود. به عبارت دیگر، موانع به عنوان بالابرنده عمل می‌کنند. با توجه به شکل ۳.۵ که از نمای جانبی به موانع نگاه می‌کند، ابتدا نقاط s و t و موانع H_1 و H_2 و H_3 را روی طبقه همکف (طبقه صفرم) در نظر گرفته سپس دو نمونه دیگر از این موانع را روی همین موانع به عنوان طبقه اول و دوم قرار می‌دهیم. از نقطه s با عبور از کنار موانع و رسیدن به نقطه t به اصطلاح ساختار 0ـ پارکینگ بوجود می‌آید. در مرحله بعد با شروع از نقطه s می‌خواهیم از درون مانع H_1عبور کنیم. مانع H_1 به عنوان بالابرنده عمل کرده و ادامه مسیر را با یک طبقه بالاتر امکان پذیر می‌سازد و در ادامه اگر بخواهیم از کنار دو مانع دیگر عبور کنیم، در اینصورت به ساختار ۱ـ پارکینگ رسیده‌ایم.



شکل ۳.۵: ساختار kـ پارکینگ

الگوریتم مورد نظر ما برای ساخت $SPM_{=k}$ بوده که انبساط جبهه موج از مبدأ s در فضا را شبیه‌سازی می‌کند. جبهه موج در زمان T شامل تمام نقاط p است که کوتاهترین مسیر آن‌ها T است. مرز جبهه موج مجموعه‌ای از قوس‌های دایره‌ای به نام موجک است که هر کدام توسط یک راس مانع (شامل s) که قبلا توسط جبهه موج پوشانده شده بود، ایجاد شده است. راس مولد v، مولد موجک نامیده می‌شود و توسط زوج $(v.w)$ مشخص می‌شود، که در آن w زمانی است که در آن v توسط جبهه موج بدست می‌آید. مولدها را می‌توان به عنوان مبدأ که با تاخیر اضافه شده‌اند در نظر گرفت، چون آن‌ها شروع به شبیه‌سازی و انتشار موجک در زمان پس از شروع می‌کنند. مکان نقطه تلاقی دو موجک مجاور منحنی دایره است. در کنار یال مانع، منحنی دایره فضای خالی را به نواحی کوتاهترین مسیر افراز می‌کند.

چارچوب پیوسته دایکسترا را به ساختار kـ پارکینگ تعمیم می‌دهیم. هر سطح از پارکینگ، صفحه‌ای با موانع چند ضلعی است که در آن جبهه موج به صورت معمول منتشر می‌شوند، اما موجک‌ها

می‌توانند با ورود به موانع (آسانسور) به طبقه بالاتر بروند. به طور دقیق‌تر، هنگامی که جبهه موج با مانع H برخورد می‌کند، آن توسط یال بیرونی H جذب می‌شود و بلافاصله به داخل فضای درونی H گسیل می‌شود. هنگامی که جبهه موج به مرز داخلی سمت دیگر H می‌رسد، جذب می‌شود و بلافاصله در طبقه بالاتر پارکینگ انتشار می‌یابد. بنابراین این حرکت عمودی هیچ تاخیری نخواهد داشت. در این تنظیمات اصلاح‌شده، جبهه موج در زمان T شامل نقاط روی تمام طبقات است که در فاصله T از مبدأ قرار دارند.
منطقه V_{k-1} از دامنه چند ضلعی بر روی طبقه k ـ پارکینگ کوتاهترین k ـ مسیر برای هر نقطه p در V_{k-1} ـ قسمتی از پاره خط $\overline{sp}$ ـ و با حذف این نقاط در دامنه چند ضلعی بر روی طبقه k تعداد نسخه‌های اضافی از این مسیر را ایجاد می‌کند. ما جزئیات دقیق الگوریتم خود را در بخش ۴.۵ به تعویق می‌اندازیم. در ادامه به این نکته اشاره می‌کنیم که در الگوریتم ما برخی از ویژگی‌های ساختار k ـ پارکینگ مفید هستند.

۱. اگر π کوتاه‌ترین مسیر $s-t$ از s روی طبقه صفر تا t روی طبقه k باشد، آنگاه تصویر عقب‌گرد $\pi^{\downarrow}$ از π ، که با تصویر کردن π در دامنه مسطح P بدست می‌آید، کوتاهترین k ـ مسیر به t است. (برای مشاهده این، فرض کنید برای تناقض ما یک k ـ مسیر π_c از s به t داریم که کوتاهتر است. سپس با استفاده از لم ۲.۲.۴ به صورت بازگشتی، می‌توانیم π_c را به $2k+1$ زیرمسیر جدا از هم که با توجه به شمارنده پیشوند مرتب شده‌اند، تقسیم کنیم. اکنون مسیرها را به سطوح پارکینگ می‌بریم و آن‌ها را به ترتیب زیر متصل می‌کنیم: اگر شمارنده پیشوند زیرمسیر فعلی و بعدی یکسان باشد، نقطه پایانی مشترک آن‌ها در همان سطح به عنوان شمارنده پیشوند ملحق شود؛ در غیر این صورت به نقطه پایانی مشترک آن‌ها در سطح بعدی ملحق می‌شود. این امر مسیر π_c را به کوتاهترین مسیر $\pi_c^{\uparrow}$از s بر روی طبقه صفر تا t روی طبقه k تبدیل می‌کند. از آنجا که جابجایی عمودی بین طبقات پارکینگ هیچ تاخیر ایجاد نمی‌کند، مسیر جایگذاری شده $\pi_c^{\updownarrow}$ کوتاهتر از π است که یک تناقض است).

۲. از آنجا که جبهه موج بر روی طبقه i تنها توسط موجک که از طبقه زیر آن می‌آیند تحت تاثیر قرار می‌گیرد، می‌توانیم انتشار جبهه موج در طبقه i به عنوان یک دامنه چندضلعی با چندین مبدأ در نظر بگیریم. در طبقه $i > 0$، تمام مبادی مربوط به مولد موجک هستند که از طبقات پایین تر می‌آیند.

۳. برای محاسبه نقاط مبدأ در طبقه $i > 0$، باید تنها موجک‌هایی که از طبقه $i-1$ می‌آیند را در نظر بگیریم. این از لم ۵.۲.۴ پیروی می‌کند، که نشان می‌دهد حتی اگر به موجک اجازه داده شود که چندین طبقه را در بالابرنده بالا رود، یک موجک از طبقه $i-1$ می‌تواند به طبقه i برسد نه موجک‌هایی که از طبقه‌هایی پایینتر می‌آیند.

۴. زیرتقسیم بندی‌های مسطح متشکل از عمودمنصف‌های موجکهای دارای برخورد روی طبقه

i، نقشه کوتاهترین مسیر برای $(= i)$ـ مسیرها یا به عبارت دیگر برای $SPM_{=i}$ است. توجه داشته باشید که از آنجا که موانع محدب هستند، کوتاهترین مسیر به نقطه‌ای در طبقه i نمی‌تواند یک مانع (در هر طبقه) را بیشتر از یک‌بار قطع کند و یا حتی می‌تواند کوتاهتر شود.

این یک روش طبیعی برای محاسبه کوتاهترین مسیر $SPM_{=k}$ است. نقشه‌ها را برای $SPM_{=i}$ برای $i = ۰, ۱, ..., k$ به صورت تکراری می‌سازیم. هر تکرار $i > ۰$ به وسیله انتشار کوتاهترین مسیر با مجموعه‌ای از مبدأیی که از تکرار قبلی می‌آیند، تعریف می‌شود. در بخش زیر از این مشاهدات برای محاسبه یک کران روی اندازه نقشه کوتاهترین kـ مسیر SPM_k استفاده می‌کنیم.

۳.۵ پیچیدگی‌های SPM_k

نقشه کوتاهترین kـ مسیر SPM_k در طبقه بالای kـ پارکینگ دقیقا $SPM_{=k}$ در قسمتی از فضای خالی که بیرون $V_{k-۱}$ است همانطور که در لم ۴.۲.۴ نشان‌داده شده است. مرز $V_{k-۱}$ دارای اندازه خطی است و بنابراین ما فقط باید پیچیدگی $SPM_{=k}$ را محدود کنیم. برای محدود کردن پیچیدگی $SPM_{=k}$، نمودار مسطح جایگذاری شده G_k تشکیل شده توسط $SPM_{=k}$، $V_{k-۱}$ و موانع چندضلعی را بررسی می‌کنیم. به ویژگی‌های زیر گراف‌های مسطح توجه کنید که نتیجه مستقیم فرمول اویلر است.

لم ۱.۳.۵. فرض کنید f تعداد وجه ها در یک گراف مسطح $G = (V, E)$ باشد. اگر درجه همه رئوس G سه یا بیشتر باشند، اندازه G برابر $O(f)$ است.

مشاهده کنید که رئوس ''مورد توجه'' در G_k نقاطی هستند که یا عمودمنصف‌های مانع را قطع کرده یا همدیگر را قطع می‌کنند و بنابراین حداقل سه درجه دارند. اگر f تعداد وجه‌ها باشد، آنگاه با لم ۱.۳.۵ پیچیدگی نقشه به خاطر این راس $O(f)$ است. علاوه بر این، G_k همچنین می‌تواند تعداد $O(n)$ رئوس درجه دو مطابق با راس چندضلعی موانع داشته باشد که در نتیجه کل پیچیدگی برابر $O(f + n)$ است.
بنابراین به منظور محاسبه کران پیچیدگی $SPM_{=k}$، حد کافی برای محدود کردن تعداد وجه‌های f در نمودار، G_k کفایت می‌کند. ما با نتیجه خوب زیر شروع می‌کنیم [۱۹].

لم ۲.۳.۵. نقشه کوتاهترین مسیر از m نقطه مبدأ وزن دار شده با تأخیرهایشان در یک دامنه چندضلعی با n راس و h سوراخ دارای $f \leq m + n + h \leq m + ۲n$ وجه است. با توجه به مسطح بودن کل پیچیدگی نقشه برابر $O(f + n)$ است.

نکته اصلی اثبات لم قبلی این است که هر منطقه نقشه کوتاهترین مسیر ستاره‌ای شکل و متصل به همه نقاط قبلی منطقه است. از آنجایی که تعداد کل قبلی‌ها حداکثر برابر $(m + n)$ است، تعداد وجه‌های ناشی از این مناطق نیز حداکثر برابر $(m + n)$ است. از همه مهمتر، این لم فورا در مورد $SPM_{=k}$ اعمال نمی شود، زیرا برخی از قبلی‌ها در طبقه kـ ام متعلق به مناطقی از زیر طبقه kـ ام

هستند. به این معنا که، برخی از m نقطه مبدأ در دامنه چندضلعی قرار ندارند، بنابراین این استدلال که هر منطقه که به قبل خود متصل شده، اتفاق نمی‌افتد. خوشبختانه استدلال لم ۲.۳.۵ یک توپولوژیکی است، و می توانیم یک حوزه توپولوژیکی (مکان شناسی) ایجاد کنیم که استدلال در آن اعمال شود.

هر نقطه $p \in \partial P$ خارج از $V_{(k-1)}$ با فاصله $(k-1)$ـ تقاطع دارای فاصله $d_{k-1}(p)$ برچسب گذاری شده است. اگر p متعلق به یک مانع H است، و $q \in \partial H$ به طوریکه $d_{k-1}(q) + |\overline{qp}| < d_{k-1}(p)$ وجود دارد، سپس $\pi_k(p)$ ممکن است با قطع کردن H به p برسد. موجک که با $SPM_{=k}$ تعیین می کند مقدار دهی اولیه می‌شود به یک منبع وزنی که توسط ''بالابرنده'' به p می‌رسد و H را قطع می کند. اگر $q \in \partial H$ کمینه کند $d_{k-1}(q) + |\overline{qp}|$، قبلی q در $\pi_{k-1}(q)$ مولد موجک است که ابتدا به p در جبهه موج می‌رسد. هر یال از ∂H را به زیر یال‌های حداکثر با قبلی یکسان تقسیم می‌کنیم. برای هر زیر یال با رأس قبلی v، ما یک ''دهانه'' مثلثی را با کشیدن پاره خط ها از نقاط انتهایی زیر یال به v ایجاد می‌کنیم. کوتاهترین مسیر از v به سمت پارکینگ k^{th} در داخل دهانه گسترش پیدا می‌کند و در دامنه شبه چندضلعی با استفاده از چسباندن تمام دهانه‌ها بر روی مرز فضای آزاد بدست می‌آید که هر یک از نقشه کوتاهترین مسیر به قبلی خود متصل شده است. اگر این دهانه‌ها در صفحه تصویر شوند، احتمالا با هم همپوشانی خواهند داشت، اما از نظر هندسی آن‌ها ساختار دامنه را تغییر نمی‌دهند و فقط دو یال در هر دهانه اضافه می‌کنند.

لم ۳.۳.۵. فرض کنید P یک دامنه چندضلعی با n راس و h سوراخ باشد. اگر P با چسباندن حداکثر m دهانه مثلثی به مرزش بسط داده شود، سپس نقشه کوتاهترین مسیر m نقطه مبدأ وزندار شده با تاخیر آن‌ها در این حوزه چند ضلعی گسترش یافته، دارای تعداد $f \leq m + n + h \leq m + 2n$ و پیچیدگی کلی $O(m+n)$ است.

لم قبلی شامل انتشار کوتاهترین مسیر در هر طبقه از kـ پارکینگ و همچنین انتشار درون موانع (بالابرنده) می‌شود. در هر دو حالت نکته کلیدی برای محدود کردن پیچیدگی ساختار تکرار شونده، محدود کردن تعداد نقاط مبدأ است که به سطح بعدی انتشار می‌یابند، چه با استفاده از بالابرنده و چه از طریق طبقه پارکینگ. در هر بالابرنده و در هر پارکینگ سطح $i > 0$، نقاط مبدأ روی مرز دامنه قرار دارند. برای سادگی، نقاط مبدأ را در رئوس موانع افراز می‌کنیم، بنابراین هر منبع یک (ـ زیر)یال ماکسیمال l در برخی از یالهای مانع ∂H با یک مولد همراه (v, w) است. به چنین منبعی به عنوان یک منبع مرزی اشاره می‌کنیم و آن را با سه‌گانه (v, w, l) نشان می‌دهیم. کوتاهترین مسیر از یک منبع (v, w, l) وارد دامنه از طریق یال l می‌شوند و قبلی آن راس v با وزن (تاخیر) w است. همانطور که در بالا اشاره شد، هر یال مرزی یک دهانه مثلثی را که به مرز دامنه انتشار چسبیده است، تعریف می‌کند. این دهانه، بدنه محدب l و v می‌باشد.

هنگامی که یال مرزی به یک دامنه (یا درون P و یا درون یک مانع) پخش می‌شوند، نقشه کوتاهترین مسیر S را در دامنه تعریف می‌کنند. اگر منطقه S مطابق با مبدأ $s = (v, w, l)$ یک یال دامنه را قطع کند، سپس فاصله تقاطع در آن یال را اندازه می‌گیرد. یک مبدأ ورودی خواسته شده‌ی (v, w, l) یک

خواسته بر روی یال l خودش است. ورودی های خواسته شده را می توان برای انتشار بیشتر نادیده گرفت، زیرا مسیری که از طریق آن وارد یال l می‌شود، می‌تواند از همان یال خارج شده و کوتاهتر شود. ادعاهای خروجی (بر روی یالهای غیر از l) نقاط مبدأ را برای سطح بعدی انتشار کوتاهترین مسیر تعریف می‌کنند. در هر یال، حداکثر خروجی های خواسته شده با همان مبدأ، یک دسته خروجی خواسته شده نامیده می‌شود. اگر یک دسته خروجی خواسته شده بر روی یال e دارای مبدأ (v,w,l) باشد مبدأ مرزی مربوطه در سطح بعدی، (v,w,l') است، که در آن l'، کوچکترین پاره‌خط شامل دسته e است. همانطور که اشاره شد، ادعاهای ورودی در l' بر کوتاهترین مسیر انتشار در سطح بعدی تاثیر نمی‌گذارد.

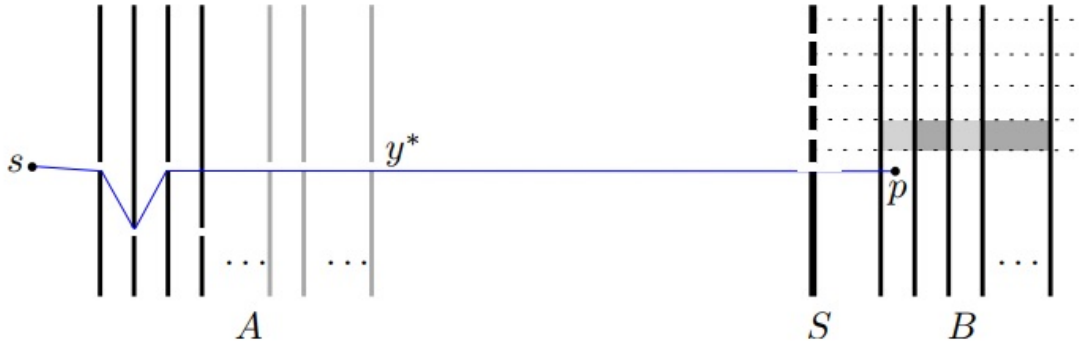
لم ۴.۳.۵. فرض کنید که S نقشه کوتاهترین مسیر بوده که از طریق انتشار m نقطه ابتدایی مرزی به داخل یک چندضلعی با n راس، بدست آمده باشد. سپس تعداد دسته‌های خروجی S در حال حاضر حداکثر $m+O(n)$ است .

اکنون آماده ایم که پیچیدگی $SPM_{=k}$ را محدود کنیم.

لم ۵.۳.۵. تعداد وجه های f_k در $SPM_{=k}$ برابر $O(n(k+1))$ است. پیچیدگی $SPM_{=k}$ محدودیت مجانبی مشابهی دارد.

برهان. اثبات به وسیله استقرا است. هدف ما این است که نشان دهیم ثابت C وجود دارد که تعداد وجه‌های f_k در $SPM_{=k}$ در اکثر موارد برابر $Cn(k+1)$ است برای همه $k \geq 0$. مراحل استقرایی را شروع می‌کنیم. فرض کنید m تعداد دسته‌های خروجی خواسته شده در $SPM_{(k-1)}$ باشد. این تعداد نقاط ابتدایی مرزی در انتشار (بالابرنده) درون فضای داخلی مانع است، که از سطح $k-1$ به سطح k می‌روند. به کمک لم ۴.۳.۵ ، تعداد نتایج گرفته شده برابر $m'=m+O(n)$ است. اما m' تعداد نقاط ابتدایی مرزی در ساختار $SPM_{=k}$ است، و دوباره به کمک لم ۴.۳.۵ ، تعداد نتایج گرفته شده برابر $m''=m'+O(n)=m+O(n)$ است، که $m'' \leq m+c_1 n$ برای یک مقدار ثابت c_1.
برای برقرار کردن حالت پایه، نقشه کوتاهترین مسیر بدون تقاطع (SPM_0) که پیچیدگی $O(n)$ دارد را به یاد بیاوریم، که اشاره دارد به اینکه تعداد نتایج خروجی روی یال برابر $O(n)$ است، به عبارت دیگر، حداکثر $c_2 n$ برای بعضی ثابت c_2 است. ترکیب حالت پایه و مرحله استقرایی، نشان می‌دهد که که تعداد نتایج خروجی یال های $SPM_{=k}$ حداکثر برابر $c_2 n + k.c_1 n$ است. تعداد وجه‌های $SPM_{=k}$ حداکثر برابر با تعداد یال‌های ابتدایی است، که حداکثر $Cn(k+1)$، برای $C=max(c_1,c_2)$ می باشد. لم ۱.۳.۵ پیچیدگی کلی را ثابت می کند. □

تطابق کران پایین : اکنون یک کران پایین برای اندازه SPM_k با ساختن یک نقشه با تعداد $\Omega(nk)$ ناحیه بدست می‌آوریم. این ساختار از موانع در شکل ۴.۵ نشان داده شده است. با دو دسته موانع A و B که موازی محور yها قرار گرفته اند، شروع می‌کنیم. درون هر دسته، فضای افقی بین نوارها بسیار کوچک هستند، اما برای وضوح بیشتر بزرگ نشان داده شده است. نقطه ابتدایی s روی محور xها قرار دارد و بسته A سمت راست آن قرار گرفته است.



شکل ۴.۵: نقشه کوتاهترین kـ مسیر با پیچیدگی $\Omega(nk)$. بسته A دارای تعداد $2k$ نوار سیاه و k نوار خاکستری است و بسته B، تعداد k نوار دارد. نوار ضخیم S به تعداد $\Omega(n)$ ورودی دارد. کوتاهترین kـ مسیر $\pi(p)$ از s نشان داده شده است. مشاهده می‌کنیم که $\pi(p)$ تعداد $k-1$ نوار در بسته A را قطع می‌کند و می‌تواند فقط اولین نوار در بسته B را قطع کند.

بسته A شامل تعداد $3k$ نوارهای سوراخ‌دار است. در اولین $2k$ نوار، نوارهای با شماره فرد در $y=0$ یک ورودی دارند و نوارهای با شماره زوج $y=-0/5$ یک ورودی دارند. k نوار بعدی در $y=0$ سوراخ دار می‌شوند. بسته B در فاصله D سمت راست A قرار گرفته و شامل k نوار بدون سوراخ است.
تعداد k نوار آخر در بسته A تضمین می‌کنند که کوتاهترین kـ مسیر که از s شروع می‌شود باید از سوراخ آخرین نوار A (با y^* نشان داده می شود) خارج شود، مسیری که آخرین نوار در A را در نقطه‌ای غیر از y^* قطع می‌کند، با حفظ همان تعداد تقاطع‌ها، ممکن است کوتاه‌تر شود.
مشاهده کوتاهترین مسیر که از s شروع شده می‌تواند با i تقاطع به شرطی که $0 \le i \le k$، به y^* برسد. با این حال، هر تقاطع، از منجر شدن به طول اضافی یک واحد جلوگیری می کند. بنابراین کوتاهترین مسیر با i تقاطع در y^* طول اضافی $(k-i)$ واحد دارد. همچنین توجه داشته باشید که کوتاهترین مسیر با iـ تقاطع قبل از y^* می‌تواند از اولین $(k-i)$ نوار، kـ نوار در مجموعه B را قطع کند، اما نمی‌تواند بعدی‌ها را قطع کند. بنابراین، در سمت راست نوار j در مجموعه B، یک منطقه با kـ قبلی $(y^*, k-j)$ و طول کل مسیر (تا نقطه‌ای روی محور xها) $D+j$ را بدست می‌آوریم. این به ما تعداد k منطقه را می‌دهد.
این ساختار را به $\Omega(nk)$ منطقه با اضافه کردن نوار عمودی S که به عنوان یک شکاف دهنده مسیر عمل می‌کند، گسترش می‌دهیم. این نوار خاص دارای مجموع m تک‌ـ نقطه دهانه در $y=0,1,...,m$ است که با y_i نشان داده می‌شود. S را در فاصله‌ای بسیار کوچک به سمت چپ دسته B قرار می‌دهیم و برای هر ورودی S یک ناحیه جدید ایجاد می‌کنیم. توجه داشته باشید که در دامنه $0 \le y \le m$، مسیری که S را به جز در یکی از سوراخ‌های y_i قطع کند، می‌تواند با دور زدن از طریق نزدیک‌ترین y_i کوتاهتر گردد و یک نقطه تقاطع بیشتر قبل از y^* ایجاد می‌کند. از این رو، کوتاهترین kـ مسیر همیشه از میان y_i می‌گذرد. این تعداد $O(mk)$ منطقه ایجاد می‌کند: kـ رأس قبلی منطقه در $y=i$ و

در سمت راست نوار j از بسته B نقاط $(y_i, k-j)$ خواهند بود و طول کل مسیر آن برابر $\sqrt{D^۲+i^۲}+j$ خواهد شد. تعداد کل رئوس در این ساختار برابر با

$$۳k \times ۴ + k \times ۲ + (m+۱) \times ۲ = ۱۴k + ۲m + ۲$$

است. با انتخاب $m = (n - ۱۴k - ۲)/۲$ و فرض $k < n/۲۸$، به $m = \Theta(n)$ می‌رسیم و تعداد کل مناطق در SPM_k برابر $\Omega(nk)$ است. این مطلب لم زیر را نتیجه می‌دهد.

لم ۶.۳.۵. بدترین حالت پیچیدگی SPM_k برابر $\Omega(nk)$ است.

ترکیب لم ۲.۱.۵ و ۵.۳.۵ و ۶.۳.۵ ، نتیجه اصلی این بخش را می گیریم.

لم ۷.۳.۵. نقشه کوتاهترین kـ مسیر از SPM_k دارای اندازه $\Theta(kn)$ است.

۴.۵ محاسبه ی یک الگوریتم برای کوتاهترین مسیر

در این بخش الگوریتم $O(k^۲ n \log n)$ برای ساخت SPM_k را شرح می‌دهیم. تعریف kـ پارکینگ را به یاد بیاورید (تعریف ۱.۲.۵)، $SPM_{=k}$ را به صورت تکراری می‌سازیم، هر سطح را در یک زمان ایجاد کنیم. برای محاسبه نقشه در هر سطح، نقاط مبدأ را از سطح قبلی منتشر می کنیم و سپس خط جبهه موج را در سطح فعلی انتشار می‌دهیم. برای این منظور از الگوریتم کوتاهترین مسیر در حضور موانع چندضلعی توسط سوری و هرشبرگر [۲] [۱۹] به عنوان یک زیرروال (رویه فرعی) استفاده می کنیم. به جز چند تغییر کوچک جزئی که لازم است، بیشتر الگوریتم بدون تغییر باقی می‌ماند. در ادامه، به طور خلاصه ایده‌های کلیدی را مرور می کنیم و درباره اصلاحات لازم بحث می کنیم. الگوریتم هرشبرگرـ سوری از روش دایکسترا پیوسته استفاده می کند که انتشار یک خط جبهه موج در فضای آزاد را شبیه‌سازی می کند. جبهه موج مجموعه‌ای از موجک‌های مدور است. شکل آن همانطور که حرکت می کند و به موانعی برخورد می کند، تغییر می کند. هر موجک از یک مولد سرچشمه می گیرد که ممکن است یک نقطه مبدأ یا یک راس مانع باشد(یک مبدأ حد وسط). یک γ مولد برای یک موجک توسط جفت (v, w)، مشخص می‌شود، که در آن v یک راس ورودی است و w زمانی است که در آن v شروع به انتشار γ می کند.

الگوریتم هرشبرگرـ سوری انتشار خط جبهه موج را در یک زیرمجموعه مسطح شبیه سازی کرده که آن را بخش منطبق بر فضای آزاد می‌گوییم. برای هر یال e، و هر نقطه $p \in e$، این الگوریتم، مولدی را که موجک آن به p می‌رسد، شناسایی می کند. ترکیب این نتایج برای همه $p \in e$، جبهه موج را برای هر رأس e به همراه دارد. ایده اصلی این الگوریتم، مکان‌یابی رویدادی جالب (مانند برخورد موجک‌ها) در تعداد ثابتی از سلول‌ها در زیرمجموعه است. هر یال آزاد فضای آزاد این زیرمجموعه در اتحاد تعداد ثابتی از سلول‌ها قرار دارد، که منطقه پوشش خوب آن $U(e)$ نامیده می‌شود. جبهه

[2] Suri, Hershberger

موج با ترکیب و انتشار موج مخالف از طریق $U(e)$ محاسبه می‌شود. سپس برای محاسبه کوتاه‌ترین مسیر ادغام می‌شوند. این نتیجه اصلی مربوط به الگوریتم ما است:

لم ۱.۴.۵. [۱۹] یک مجموعه از موانع چندضلعی با n راس و مجموعه‌ای از $O(n)$ نقطه مبدأ با تاخیر داده شده‌اند، می‌توان نقشه کوتاهترین مسیر را در زمان $O(n \log n)$ و فضای $O(n \log n)$ محاسبه کرد.

با توجه به مباحث قبل از لم ۴.۳.۵، به یاد می‌آوریم که نقاط ابتدایی موجود در طبقه i توسط سه‌گانه (v, w, l)، که در آن یک یال (زیر) از یک مانع H است، و موجک تولید شده توسط (v, w)، وارد کف طبقه $j < i$ از داخل H یک آسانسور می‌شود γ که از لبه عبور می‌کند. هر مبدأ (v, w, l) یک دهانه مثلثی را که بر روی مرز فضای آزاد قرار دارد، تعریف می‌کند. بطور مفهومی، ما به γ تبدیل موجک از (v, w, l) به عنوان انتشار در دهانه قبل از ورود به کف طبقه i فکر می‌کنیم. از نظر الگوریتمی، می‌توانیم دهانه را نادیده بگیریم و انتشار را در فضای آزاد از لبه شروع کنیم. این یک اصلاح جزئی در مرحله آغازین الگوریتم هرشبرگر–سوری است.
به طور خاص، الگوریتم هرشبرگر–سوری برای هر یال e به صورت زیر است:

۱. تمام نقاط ابتدایی مرزی (v, w, l)، را به گونه‌ای پیدا کنید که منطقه تحت پوشش $U(e)$ حاوی l باشد.

۲. $covertime(e)$ را مقدار دهی اولیه کنید، این همان زمانی است که e می تواند با جبهه موج درگیر شود، به حداقل رساندن تمام نقاط ابتدایی مرزی (v, w, l) با $l \in U(e)$ و برای هر مبدأ از این دست در نظر گرفتن مسیرهای از v با تأخیر w، محدود به عبور از l.

۳. برای هر مبدأ (v, w, l) با $l \in U(e)$ ، موجک آن را تا γ در داخل $U(e)$ پخش کنید.

در لم زیر نحوه محاسبه نقاط ابتدایی مرزی برای هر مرحله انتشار موج را نشان می دهیم.

لم ۲.۴.۵. در دامنه یک چندضلعی با m نقطه مبدأ مرزی و n رأس، می‌توان نتایج خروجی از مبدأ را در زمان و حافظه $O((m+n)\log(m+n))$ محاسبه کرد.

برهان. از الگوریتم هرشبرگر–سوری که برای نقاط مرزی اصلاح شده و در بالا شرح داده شده است، استفاده می‌کنیم. این الگوریتم کوتاهترین نقشه مسیر را برای نقاط ابتدایی داخل دامنه چندضلعی در کل زمان و حافظه $O((m+n)\log(m+n))$ محاسبه می‌کند. نقشه کوتاهترین مسیر افرازی است به فواصل $O(m+n)$، هر کدام توسط مبدأ خود بدست می آید. نقاط ابتدایی مرزی مجموعه دیگری از m فاصله را تشکیل می‌دهند. پوشش این دو مجموعه فواصل در خط اضافی زمان و حافظه، نتایج خروج را مشخص می‌کنیم، یعنی آنهایی را که دارای نتایج اولیه دیگری با بخش های متفاوت هستند. □

با وجود این اطلاعات اولیه، آماده هستیم تا الگوریتم خود را توصیف کنیم. ورودی یک دامنه چندضلعی P با موانع محدب است. از M برای مشخص کردن مجموعه نقاط مبدأ مرزی استفاده خواهیم کرد به عنوان ورودی به الگوریتم هرشبرگر–سوری این الگوریتم دو چیز را محاسبه می‌کند: منطقه $(K-۱)$–رؤیت پذیری V و نقشه $(=k)$ مسیر $SPM_{=k}$ که با هم ترکیب می‌شوند و SPM_k را تشکیل می دهند. طول کوتاهترین مسیر تا هر نقطه p می‌تواند به راحتی محاسبه شود ابتدا با یافتن منطقه حاوی p در نقشه SPM_k و سپس اتصال p به k–قبلی این منطقه همانطور که در ابتدای بخش ۵ شرح داده شده است.

تعریف ۱.۴.۵. الگوریتم برای ساخت SPM_k

۱. قرار می‌دهیم $M=\{s\}$ و از الگوریتم هرشبرگر–سوری استفاده کرده تا $SPM_۰$ را برای چندضلعی P محاسبه کند. V را به منطقه $\emptyset$ مقدار دهی اولیه دهید.

۲. برای هر $i \in ۱,۲,...,k$ تکرار کنید:

(آ) با استفاده از لم ۲.۴.۵ ، نقاط مبدأ موجود در $SPM_{i-۱}$ را از طریق موانع موجود در P انتشار داده تا مجموعه نقاط مبدأ مرزی M_{new} را برای SPM_i محاسبه کند.

(ب) تمام مناطق موجود در $SPM_{(i-۱)}$ را که رأس قبلی آن s است، شناسایی کنید. مشاهده می‌کنید که این دقیقاً منطقه $V'=V_{i-۱}\backslash V_{i-۲}$ است. P را با منطقه حذف شده‌ی قبلی، یک دامنه چندضلعی جدید در نظر بگیرید.

(ج) اگر $V=\emptyset$ است، سپس قرار دهید $V=V'$. در غیر این صورت، V را با V' در رأس‌های مشترک ادغام کنید.

(د) قرار دهید $M=M_{new}$ و از الگوریتم هرشبرگر–سوری استفاده کرده تا SPM_i را برای دامنه چندضلعی P محاسبه کنید.

۳. ناحیه $SPM_{=k}$ را با V در مرز ناحیه‌های $SPM_{=k}$ ادغام کنید که s رأس قبلی آن باشد (یعنی $V'=V_k\backslash V_{k-۱}$) تا $SPM_{=k}$ بدست آید.

مشاهده کنید که بعد از مرحله ۲ج تکرار i، ناحیه V برابر با $V_{(i-۱)}$ است. زیرا $V_{(i-۱)}$ شامل $V_{(i-۲)}$ است و از آنجا که هر دو منطقه اندازه خطی دارند توسط (لم ۲.۱.۵)، مرحله ۲ج زمان خطی می‌گیرد. بنابراین، زمان کل اجرا بستگی به k داشته که به آن فراخوانی الگوریتم هرشبرگر–سوری با تعداد $O(nk)$ نقطه مبدأ دارد (قضیه ۷.۳.۵). در ادامه نتیجه زیر را خواهیم داشت.

قضیه ۱.۴.۵. اگر P یک دامنه چندضلعی با موانع محدب به تعداد کل n رأس باشد، نقشه کوتاهترین k–مسیر برای P با توجه به یک مبدأ می‌تواند در زمان $O(k^۲ n \log n)$ و در حافظه $O(kn \log n)$ محاسبه شود.

فصل ۶

نتایج

با توجه به تعریف گراف رؤیت پذیری فرض کردیم S مجموعه موانع چندضلعی‌های از هم جدا در صفحه بوده که در حالت کلی دارای n یال می‌باشند. برای محاسبه‌ی گراف رؤیت پذیری S، جفت رئوسی را پیدا کردیم که برای یکدیگر قابل مشاهده بودند. به عبارت دیگر، برای هر جفت رأس این آزمایش را انجام دادیم که پاره‌خطی واصل بین این دو رأس آیا موانع را قطع کند. آزمایشاتی از این قبیل، زمان $O(n)$ تا زمان $O(n^۳)$ را منجر شد. اگر در یک زمان بر روی یک رأس متمرکز شویم و مانند همه الگوریتم‌های زیر، همه رأس‌های قابل مشاهده از آن را شناسایی کردیم. و با استفاده از این تعریف نشان دادیم که گراف رؤیت پذیر مجموعه‌ی S که شامل موانع چندضلعی ناپیوسته با تعداد کل n یال، در زمان $O(n^۲ \log n)$ محاسبه می‌شود.

در این پایان نامه مفهوم جدیدی از کوتاهترین مسیر با حذف موانع و با هدف کاهش هزینه‌های مسیر برای بررسی این انواع مشکلات هندسی کلاسیک کوتاهترین مسیر را معرفی کرده‌ایم. . یک الگوریتم زمان $O(n^۳)$ برای محاسبه کوتاهترین مسیر یک‌ـحذفی بین یک جفت نقاط در داخل یک چندضلعی ساده ارائه کرده ایم. مسئله جالب دیگر در این زمینه محاسبه کوتاهترین نقشه مسیر حذفی برای نقطه s است که در آن ورودی چند ضلعی ساده P و نقطه s است و خروجی ساختار داده‌ای است که برای یک پرس و جو داده شده از نقطه $q \in P$، می‌تواند طول کوتاهترین مسیر یک‌ـ حذفی از s تا q را به طور بهینه گزارش کند. ضمنا در مورد مسائل کوتاهترین مسیر یک‌ـحذفی و دو‌ـحذفی را در بین مجموعه‌ای به تعداد n مانع موازی مستطیل شکل جداکننده، فرض کردیم که نقاط s و t دو نقطه از فضای $\mathbb{R}^۲$ باشد که درون هیچ یک از موانع نیستند. در مورد یک مسیر

مستطیل شکل از s تا t از یال‌های افقی تشکیل شده است که به سمت یال‌های راست و یال‌های عمودی که به سمت بالا هدایت می‌شوند. به طور مشابه، در مسیر مستطیلی حذفی، همه یال‌های افقی (به جز حداکثر یکی) به سمت راست هدایت می‌شوند و همه یال‌های عمودی (به جز حداکثر یکی) به سمت بالا حرکت می‌کنند. الگوریتم‌های ما برای هر دوی این مسائل در زمان $O(n \log n)$ اجرا شدند.

همینطور مسئله یافتن کوتاهترین مسیری را که مجاز به حذف تعداد محدودی از موانع محدب آن هستیم، بررسی کردیم. نشان دادیم که اگرچه دو kـ‌مسیر ممکن است همدیگر را قطع کنند، براساس پیشوندها می‌توان آنها را به مسیرهای جزئی که همدیگر را قطع نمی‌کنند تجزیه کرد. این تجزیه این امکان را می‌دهد که کوتاهترین kـ‌مسیرها را به صورت کارایی با استفاده از چارچوب پیوسته دایکسترا محاسبه کنیم. نشان دادیم که اندازه نقشه کوتاهترین kـ‌مسیر برابر $\Theta(kn)$ است و اینکه می توان آن را در بدترین حالت در زمان $O(k^۲ n \log n)$ و با استفاده از پیچیدگی حافظه $O(kn \log n)$ محاسبه کرد. وقتی $k = O(۱)$ پیچیدگی زمانی الگوریتم بهینه است.

مراجع

[1] M. Abellanas, A. García, F. Hurtado, J. Tejel, and J. Urrutia. Augmenting the connectivity of geometric graphs. Computational Geometry, 40(3):220–230, 2008.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. Network Flows: Theory, Algorithms, and Applications. Prentice hall, 1993.

[3] T. Asano. An efficient algorithm for finding the visibility polygon for a polygonal region with holes. IEICE TRANSACTIONS (1976-1990), 68(9):557–559, 1985.

[4] T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai. Visibility of disjoint polygons. Algorithmica, 1(1-4):49–63, 1986.

[5] J.-L. De Carufel, C. Grimm, A. Maheshwari, and M. Smid. Minimizing the continuous diameter when augmenting paths and cycles with shortcuts. In 15th Scandinavian Symposium and Workshops on Algorithm Theory, pages 27:1–27:14, 2016.

[6] T. M. Chan. Low-dimensional linear programming with violations. SIAM Journal on Computing, 34(4):879–893, 2005.

[7] D. Z. Chen and H. Wang. Computing shortest paths among curved obstacles in the plane. ACM Trans. Algorithms, 11(4):26:1–26:46, 2015.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein. Introduction to Algorithms. The MIT Press Cambridge, 2009

[9] M. de Berg, O. Cheong, M. van Kreveld, M. Overmars. Computational Geometry Algorithms and Applications. Springer-Verlag Berlin, 3nd edition, 2008.

[10] H. Edelsbrunner, L. J. Guibas, and J. Stolfi. Optimal point location in a monotone subdivision. SIAM Journal on Computing, 15(2):317–340, 1986.

[11] S. Eriksson-Bique, J. Hershberger, V. Polishchuk, B. Speckmann, S. Suri, T. Talvitie, K. Verbeek, and H. Yıldız. Geometric k shortest paths. In Proceedings of the TwentySixth Annual ACM-SIAM Symposium on Discrete Algorithms, pages 1616–1625, 2015.

[12] M. Farshi, P. Giannopoulos, and J. Gudmundsson. Improving the stretch factor of a geometric network by edge augmentation. SIAM Journal on Computing, 38(1):226–240, 2008.

[13] S. K. Ghosh and D. M. Mount. An output-sensitive algorithm for computing visibility graphs. SIAM Journal on Computing, 20(5):888–910, 1991.

[14] L. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. E. Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. Algorithmica, 2(1-4):209–233, 1987.

[15] S. Har-Peled and V. Koltun. Separability with outliers. 16th International Symposium on Algorithms and Computation, pages 28–39, 2005.

[16] J. Hershberger and J. Snoeyink. Computing minimum length paths of a given homotopy class. Computational Geometry, 4(2):63–97, 1994.

[17] S. Schuierer. An optimal data structure for shortest rectilinear path queries in a simple rectilinear polygon. International Journal of Computational Geometry , Applications, 6(2):205–226, 1996.

[18] J. Hershberger, N. Kumar, S. Suri, Shortest Paths in the Plane with Obstacle Violations. Algorithmica, 82(7):1813–1832, 2020.

[19] J. Hershberger and S. Suri. An optimal algorithm for Euclidean shortest paths in the plane. SIAM Journal on Computing, 28(6):2215–2256, 1999.

[20] J. Hershberger, S. Suri, and H. Yıldız. A near-optimal algorithm for shortest paths among curved obstacles in the plane. In Proceedings of the Twenty-Ninth Annual Symposium on Computational Geometry, pages 359–368, 2013.

[21] J. Hershberger, S. Suri, and H. Yıldız. A near-optimal algorithm for shortest paths among curved obstacles in the plane. In Proceedings of the Twenty-Ninth Annual Symposium on Computational Geometry, pages 359–368, 2013.

[22] D. Kirkpatrick. Optimal search in planar subdivisions. SIAM Journal on Computing, 12(1):28–35, 1983.

[23] D. T. Lee and F. P. Preparata. Euclidean shortest paths in the presence of rectilinear barriers. Networks, 14(3):393–410, 1984.

[24] A. Maheshwari, S. C. Nandy, D. Pattanayak, S. Roy, and M. Smid. Geometric path problems with violations. Algorithmica, 1–24, 2016.

[25] J. Matoušek. On geometric optimization with few violated constraints. Discrete , Computational Geometry, 14(4):365–384, 1995.

[26] J. S. B. Mitchell. A new algorithm for shortest paths among obstacles in the plane. Annals of Mathematics and Artificial Intelligence, 3(1):83–105, 1991.

[27] J. S. B. Mitchell. Shortest paths among obstacles in the plane. International Journal of Computational Geometry , Applications, 6(3):309–332, 1996.

[28] J. S. B. Mitchell and C. H. Papadimitriou. The weighted region problem: finding shortest paths through a weighted planar subdivision. Journal of the ACM (JACM), 38(1):18–73, 1991.

[29] M. H. Overmars and E. Welzl. New methods for computing visibility graphs. In Proceedings of the Fourth Annual Symposium on Computational Geometry, 164–171, 1988.

[30] H. Rohnert. Shortest paths in the plane with convex polygonal obstacles. Information Processing Letters, 23(2):71–76, 1986.

[31] T. Roos and P. Widmayer. k-violation linear programming. Information Processing Letters, 52(2):109–114, 1994.

[32] J. A. Storer and J. H. Reif. Shortest paths in the plane with polygonal obstacles. Journal of the ACM (JACM), 41(5):982–1012, 1994.

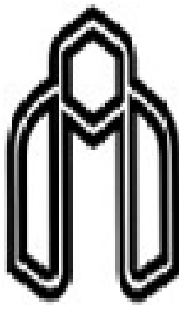
Shortest Paths in the Plane with Obstacle Violations

In this thesis, we first study variants of the classical geometric shortest path problem inside a simple polygon. In this version we allow to go outside the polygon. Let P be a simple polygon with n vertices and let s and t be two points in P. For an integer k ≥ 0, we define a k-violation path from s to t to be a connected path between s and t whose intersection with outside P consists of at most k segments. The number of segments of the path inside P. has no restriction The goal is to compute a path with minimum Euclidean length among all (≤ k)-violation paths from s to t. In this thesis, we study this problem for k = 1 and give an algorithm to compute the shortest one-violation path in $O(n^3)$ time. We show that this problem for rectilinear polygons, is solvable in $O(n \log n)$ time.

Then, we study the problem of computing shortest paths in the plane among h convex obstacles, where the path is allowed to go through (violate) up to k obstacles, for k ≤ h. Given a fixed source point s, we show how to construct a map, called a shortest k-path map, so that all destinations in the same region of the map have the same combinatorial shortest path passing through at most k obstacles. We prove a tight bound of Θ(kn) on the size of this map, and show that it can be computed in $O(k^2 n \log n)$ time, where n is the total number of obstacle vertices.

Finally, we will study visibility graphs, which will be used to compute the length of the shortest path between two points inside a given polygon..

Keywords: Shortest paths, Simple polygons, Rectilinear polygons, Continuous Dijkstra, Obstacle crossing, Visibility



Shahrood University of Technology

Faculty of Mathematical Sciences

M.Sc. Thesis in Optimization

Shortest Paths in the Plane with Obstacle Violations

By: Mohammad Hossein Teymouri

Supervisors

Dr. Jafar Fathali
Dr. Abolfazl Poureidi

June 2020