بسم الله الرحمن الرحیم

آماده‌شده توسط کلاس yazd-thesis، نسخه 0.1

دانشگاه یزد

دانشکده ریاضی

گروه علوم کامپیوتر

پایان‌نامه برای دریافت درجه کارشناسی ارشد

علوم کامپیوتر (محاسبات علمی)

بررسی مسئله k-مسیر با حداقل یال‌اشتراکی

استاد راهنما: دکتر محمد فرشی

استاد مشاور: دکتر مهدیه هاشمی‌نژاد

پژوهش و نگارش: علیرضا جلایق

اسفندماه ۱۳۹۴

تقدیم به

پدر و مادر عزیزم

و همه کسانی که درست اندیشیدن را به من آموختند.

سپاس‌گزاری

سپاس خداوند یکتای عزتمندی که رحمت و دانش او در سراسر گیتی گسترده شده، آسمان‌ها و زمین همه از آن اوست و علم و دانش حقیقی را بر هر که بخواهد موهبت می‌فرماید. رحمت و لطف او را بی‌نهایت سپاس می‌گویم چرا که فهم و درک مطالب این پژوهش را بر من ارزانی داشت و مرا به این اصل رساند که علم و ایمان دو بال یک پروازند. توفیق تلاش به من داد و هر بار که خطا کردم فرصتی دوباره، تا با امید، تلاشی تازه را آغاز کنم و به خواست او به نتیجه‌ی مطلوب نائل آیم. به‌راستی که همه چیز از آن اوست و همه‌چیز به خواست اوست.

بر خود وظیفه می‌دانم از زحمات بی‌دریغ استاد راهنمای ارجمند جناب آقای دکتر محمد فرشی، تشکر و قدردانی نمایم؛ ایشان در تمامی مراحل انجام پایان نامه با دقت نظر و به پشتوانه اندوخته‌های علمی، مرا یاری نمودند، بسیار خرسندم که این دوره را تحت رهنمودهای ارزشمند ایشان به پایان رساندم. همچنین از زحمات و راهنمایی‌های استاد گرانقدر، سرکار خانم مهدیه هاشمی‌نژاد به عنوان استاد مشاور، کمال تشکر و قدردانی را دارم.

بسمه تعالی

مدیریت تحصیلات تکمیلی	**صورتجلسه دفاعیه پایان‌نامه دانشجوی** **دوره کارشناسی ارشد**	شناسه: ب/ک/3

جلسه دفاعیه پایان‌نامه تحصیلی آقای/خانم: علیرضا جلایق دانشجوی کارشناسی ارشد

رشته/گرایش: علوم کامپیوتر/علوم کامپیوتر(محاسبات علمی)

تحت عنوان: بررسی مسئله k-مسیر با حداقل یال‌اشتراکی

و تعداد واحد: ۶ در تاریخ ۱۳۹۴/۱۲/۱۵ با حضور اعضای هیأت داوران (به شرح ذیل) تشکیل گردید.

پس از ارزیابی توسط هیأت داوران، پایان‌نامه با نمره به عدد ۱۷.۹۰ به حروف هفده و نود صدم و درجه خوب مورد تصویب قرار گرفت.

عنوان	نام و نام خانوادگی	امضاء
استاد/استادان راهنما:	دکتر محمد فرشی	
استاد/استادان مشاور:	دکتر مهدیه هاشمی‌نژاد	
متخصص و صاحبنظر داخلی:	دکتر سید ابوالفضل شاهزاده فاضلی	
متخصص و صاحبنظر خارجی:	دکتر اردشیر دولتی ملک آباد	

نماینده تحصیلات تکمیلی دانشگاه (ناظر)

نام ونام خانوادگی: دکتر منصور نمازیان

امضاء:

چکیده

برای دو راس داده شده s و t از یک گراف $G = (V, E)$، مسئله پیدا کردن k مسیر یال مجزا از راس s به راس t یکی از مسائلی است که در نظریه گراف مطرح می‌شود. k مسیر را یال‌مجزا گویند، اگر هر یال $e \in E$ حداکثر یک‌بار در مسیرها استفاده شده باشد. تعداد مسیرهای یال‌مجزا در گراف محدود است، فرض کنید حداکثر تعداد مسیرهای یال‌مجزا در G، k' است، در صورتی که $k > k'$ باشد، بدیهی است که پیدا کردن k مسیر یال‌مجزا در G ممکن نیست. یک راه‌حل جایگزین پیدا کردن k مسیر با حداقل یال‌اشتراکی، یعنی یالی که در بیش از یک مسیر ظاهر شده است، می‌باشد. در این پایان‌نامه مسئله حداقل یال‌اشتراکی را بررسی می‌کنیم که هدف آن پیدا کردن k مسیر از s به t با کم‌ترین تعداد یال‌اشتراکی می‌باشد. ثابت شده است که مسئله حداقل یال‌اشتراکی NP–سخت است و هیچ الگوریتم تقریبی با عامل $2^{\log^{1-\epsilon} n}$، به‌ازای‌هر $\epsilon > 0$ برای آن وجود ندارد. بهترین الگوریتم تقریبی که برای این مسئله ارائه شده، دارای عامل تقریب $\lfloor \frac{k}{2} \rfloor$ می‌باشد و تنها در حالتی که k ثابت باشد الگوریتم با زمان چندجمله‌ای برای مسئله وجود دارد.

مسئله دیگری که در این پایان‌نامه بررسی می‌شود، مسئله حداقل r–آسیب‌پذیری است. هدف از مسئله حداقل آسیب‌پذیری پیدا کردن k مسیر از s به t در G، با کم‌ترین تعداد یال آسیب‌پذیر می‌باشد. به یالی که بیش از r بار در مسیرها ظاهر شود، یال آسیب‌پذیر می‌گویند. r یک عدد ثابت است که جزء ورودی‌های مسئله حداقل آسیب‌پذیری می‌باشد و هرگاه $r = 1$ باشد، مسئله حداقل آسیب‌پذیری به مسئله حداقل یال‌اشتراکی تبدیل می‌شود.

فهرست مطالب

فهرست جداول

فهرست تصاویر

فصل ۱

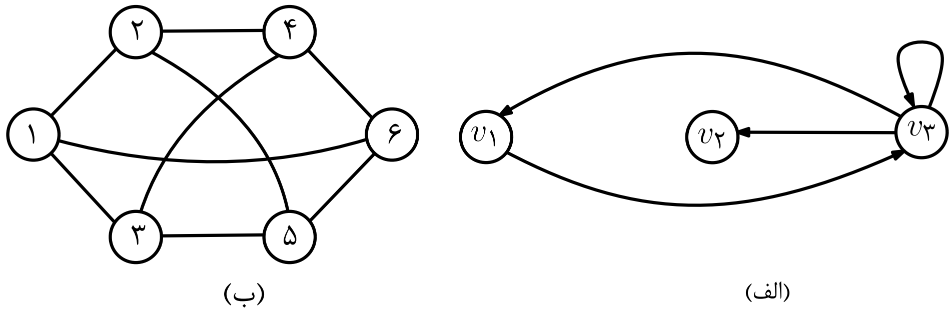
مقدمات

در این فصل، مفاهیمی که در فصل‌های آینده به آنها نیاز داریم، بیان می‌گردد. همچنین برخی از ویژگی‌ها، قضیه‌ها و مسائل مرتبط مورد بررسی قرار می‌گیرد. مسئله جریان با یال-هزینه کمینه و مسئله کوچکترین پوشش مجموعه از جمله مسائلی هستند که در این فصل بیان می‌شوند.

۱-۱ گراف‌ها

یک گراف، ساختاری است شامل دو مجموعه‌ی متناهی $V = \{v_۱, v_۲, \ldots, v_n\}$ به عنوان مجموعه **رئوس** و $E = \{e_۱, e_۲, \ldots, e_n\}$ به عنوان مجموعه **یال‌ها**. هر یال یک زوج مرتب از رئوس مجموعه V است به عنوان مثال: $e_i = (v_j, v_k)$ نشان دهنده یالی از v_j به v_k است. دراین‌صورت e_i یالی است که از راس v_j خارج شده و به راس v_k وارد می‌شود. چنین ساختاری، گراف جهت‌دار نامیده می‌شود، زیرا به‌ازای‌هر یال یک جهت (از v_j به v_k) مشخص شده است. به گرافی که یال‌های آن بدون‌جهت باشند، گراف بدون‌جهت می‌گویند.

برچسب‌گذاری یک گراف، نسبت دادن برچسب‌هایی به یال‌های گراف، یا به راس‌های گراف و یا به هر دوی آن‌ها است که به صورت معمول این برچسب‌ها را با اعداد صحیح و یا حروف نمایش می‌دهند.



شکل ۱-۱: نمونه‌ای از گراف

گراف را می‌توان به راحتی توسط نموداری نمایش داد که در آن راس‌ها با دایره و یال‌ها به شکل خط‌های جهت‌داری که راس‌ها را به یکدیگر متصل می‌کنند، مشخص کرد. یک گراف با رئوس $\{v_۱, v_۲, v_۳\}$ و یال‌های

$\{(v_۱, v_۳), (v_۳, v_۱), (v_۳, v_۲), (v_۳, v_۳)\}$ در شکل ۱-۱ (الف) نمایش داده شده است. به تعداد یال‌های ورودی به راس v، درجه ورودی v گفته می‌شود و با $\mathrm{indeg}(v)$ نمایش داده می‌شود، همچنین به تعداد یال‌های خروجی از راس v، درجه خروجی راس v گفته می‌شود و با $\mathrm{outdeg}(v)$ نمایش داده می‌شود.

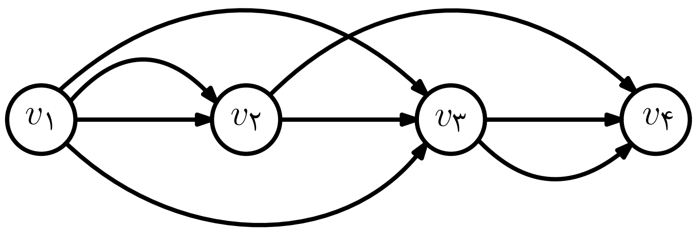
یک توالی از یال‌های $(v_i, v_j), (v_j, v_k), \ldots, (v_m, v_n)$ یک **گشت**[۱] از v_i به v_n نامیده می‌شود. طول یک گشت تعداد یال‌هایی است که با حرکت از راس اول تا راس نهایی پیموده می‌شود. گشتی که شامل راس تکراری (و طبعا یال تکراری) نباشد را **مسیر**[۲] می‌نامند. اگر یک مسیر شامل راس تکراری نباشد آن را **مسیر ساده**[۳] گویند. یک مسیر از v_i به خودش، بدون داشتن راس تکراری را یک **دور**[۴] با مرکزیت v_i می‌گویند. در یک گراف بدون‌جهت G، دو راس u و v را **متصل**[۵] می‌نامند اگر در گراف G حداقل یک مسیر از u به v و حداقل یک مسیر از v به u وجود داشته باشد. درغیراین‌صورت آنها **غیرمتصل**[۶] نامیده می‌شوند. اگر دو رأس با مسیری به طول یک (یک یال) به هم متصل باشند، رأس‌ها، **مجاور**[۷] نامیده می‌شوند. یک دور اگر غیر از راسی که دور را با آن آغاز شده است، راس تکراری دیگری نداشته باشد، **دور ساده**[۸] گفته می‌شود.

در شکل ۱-۱ (الف) $(v_۱, v_۳)$، $(v_۳, v_۲)$ یک مسیر ساده از $v_۱$ به $v_۲$ می‌باشد. در این شکل، دنباله‌ی یال‌های $(v_۱, v_۳)$،$(v_۳, v_۳)$ و $(v_۳, v_۱)$ یک دور است اما دور ساده نیست. اگر یال‌های گراف برچسب داشته باشند، می‌توانیم گام‌ها را با برچسب نیز مشخص کنیم. این برچسب یک توالی از برچسب یال‌هایی است که هنگام پیمایش یک مسیر پیموده می‌شود. در نهایت، یک یال از یک راس به خودش را **طوقه**[۹] می‌نامیم. در شکل ۱-۱ (الف) یک طوقه روی راس $v_۳$ دیده می‌شود.

یک **زیرگراف**[۱۰] برای G، گرافی است که راس‌های آن زیرمجموعه راس‌های G و یال‌های آن زیرمجموعه یال‌های G باشد و به‌ازای‌هر یالی که در زیرگراف قرار می‌گیرد باید راس‌های دو طرف آن نیز داخل مجموعه راس‌های زیرگراف باشد. به‌طور معکوس یک **ابرگراف**[۱۱] برای G، گرافی است که G زیرگراف آن است.

[۱] Walk
[۲] Path
[۳] Simple path
[۴] Cycle
[۵] Connected
[۶] Disconnected
[۷] Adjacent
[۸] Simple cycle
[۹] Loop
[۱۰] Subgraph
[۱۱] Supergraph

به‌عبارت‌دیگر اگر H زیرگراف G باشد، آنگاه G یک ابرگراف برای H می‌باشد. اگر در یک گراف جهت‌دار $G = (V, E)$، بتوان راس‌ها را به صورت $v_۱, v_۲, \ldots, v_n$ برچسب‌گذاری کرد طوری که به‌ازای‌هر یال $E \ni (v_i, v_j)$، $i < j$ باشد، به ترتیب راس‌ها، **ترتیب توپولوژیکی**[۱] می‌گویند. در گراف شکل ۱-۲، $v_۱, v_۲, v_۳, v_۴$، یک ترتیب توپولوژیکی است. گراف‌هایی که دور جهت‌دار دارند، ترتیب توپولوژیکی برای آنها وجود ندارد. به عنوان مثال گراف ۱-۱ (الف)، ترتیب توپولوژیکی ندارد.



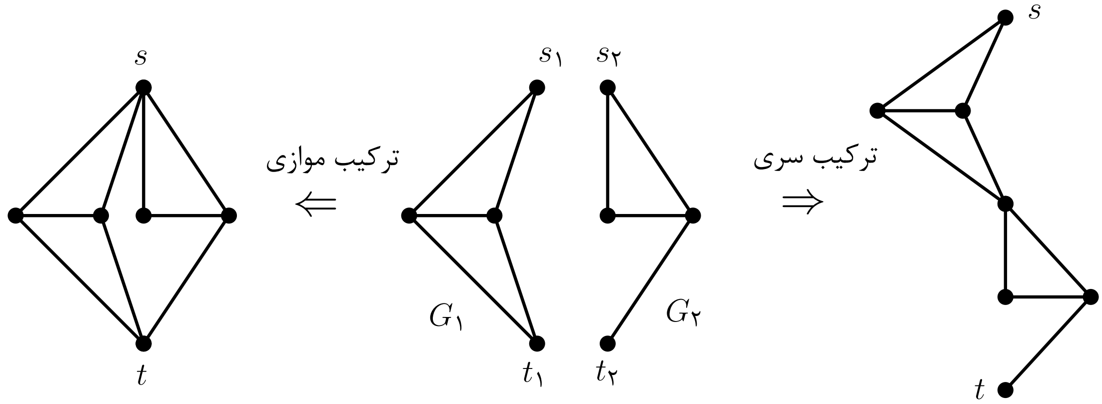
شکل ۱-۲: ترتیب $v_۱, v_۲, v_۳, v_۴$، یک ترتیب توپولوژیکی برای گراف است

گراف، انواع مختلفی دارد. در زیر، چند نمونه گراف که تعریف آن‌ها در این تحقیق مورد نیاز می‌باشد آورده شده است:

به یک گراف بدون‌جهت **همبند**[۲] می‌گویند، اگر به‌ازای‌هر دو راس دلخواه u و v، مسیری از u به v یا از v به u وجود داشته باشد. درغیراین‌صورت گراف را **ناهمبند**[۳] می‌نامند. اگر در یک گراف جهت‌دار، جایگزین کردن تمام یال‌های جهت‌دار با یال‌های بدون‌جهت منجر به ساخت یک گراف همبند (بدون‌جهت) شود، در این صورت به این گراف جهت‌دار، **همبند ضعیف** می‌گویند.

اگر در یک گراف برای هر راس یک نقطه و برای هر یال که دو راس گراف را به هم وصل می‌کند، یک خط بین نقاط متناظر آنها در صفحه کشیده شود، یک رسم از آن گراف به دست می‌آید. یک رسم مسطح از گراف، رسمی است که در آن هیچ دو یالی مگر در راس‌های انتهاییشان یکدیگر را قطع نکنند. گرافی که دارای رسم مسطح باشد **گراف مسطح**[۴] گوییم و در غیراین صورت به آن **گراف نامسطح** می‌گوییم. به عنوان مثال شکل ۱-۱ (الف) گراف مسطح است و شکل ۱-۱ (ب) گراف نامسطح را نمایش می‌دهد. هر رسم از گراف

[۱] Topological ordering
[۲] Connected
[۳] Disconnected
[۴] Planner graph

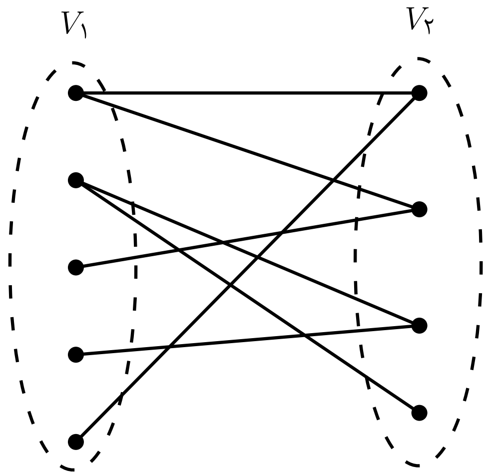


شکل ۱-۳: ترکیب سری و موازی دو گراف G_1 و G_2

مسطح روی یک صفحه، آن را به ناحیه‌هایی تقسیم می‌کند، هر کدام از آن نواحی یک **وجه** [1] از گراف نامیده می‌شود. به وجه نامتناهی، **وجه بیرونی** و دیگر وجه‌ها را **وجه داخلی** گویند. **گراف وزن‌دار** گرافی است که به هر یک از یال‌ها یا رئوس یک عدد یا وزن نسبت داده شود. رئوس s_1 و t_1 از گراف G_1 و رئوس s_2 و t_2 از گراف G_2 را در نظر بگیرید. s_iها و t_iها را به ترتیب رئوس مبدا و مقصد می‌گوییم. گراف G از ترکیب سری G_1 و G_2 به‌دست‌می‌آید اگر راس‌های t_1 و s_2 با هم ادغام شوند و تشکیل یک راس جدید در G بدهد و راس‌های s_1 و t_2 را به عنوان مبدا و مقصد در G انتخاب کنیم. گوییم گراف G از ترکیب موازی گراف‌های G_1 و G_2 به‌دست‌آمده، اگر راس‌های s_1 و s_2 با هم ادغام شوند و به عنوان مبدا G و راس‌های t_1 و t_2 با هم ادغام شوند و به عنوان مقصد G انتخاب کنیم. گراف $G = (V, E)$ با مجموعه راس‌های $\{s, t\}$ و مجموعه یال $\{(s, t)\}$ را یک گراف سری-موازی با مبدا s و مقصد t است و هر گرافی که به صورت بازگشتی توسط دو عمل ترکیب سری و یا موازی به‌دست‌آید را **گراف سری-موازی**[2] می‌گوییم (شکل ۱-۳ را ببینید).

گراف دوبخشی[3]، گرافی است که راس‌هایش را می‌توان به دو مجموعه مجزا مثل V_1 و V_2 افراز کرد، به طوری که هر یال از آن گراف، یک راس از V_1 را به یک راس از V_2 وصل می‌کند. معمولا گراف دوبخشی را به صورت $G = (V_1, V_2, E)$ نمایش می‌دهند که V_1 و V_2 دو بخش گراف و E مجموعه یال‌های گراف را

[1] Face
[2] Parallel series graph
[3] Bipartite graph



شکل ۱-۴: مثالی از یک گراف دوبخشی

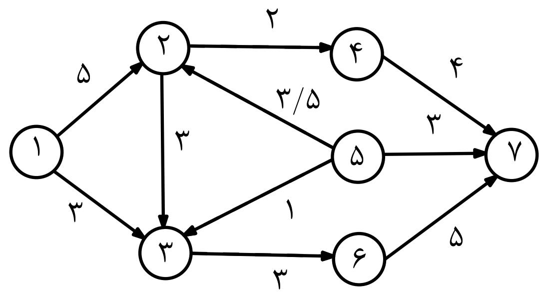
تشکیل می‌دهند. یک مثال از این گراف‌ها در شکل ۱-۴ نمایش داده شده است.

۱-۲ شبکه جریان

به یک گراف جهت‌دار با یال‌های وزن‌دار به طوری که حداقل یک راس به عنوان مبدا و حداقل یک راس به عنوان مقصد داشته باشد، **شبکه جریان**[۱] می‌گوییم. وزن هر یال نشان دهنده ظرفیت جریان عبوری از آن یال می‌باشد و جهت هر یال نشان دهنده‌ی جهت قابل عبور جریان است. در مسئله‌ی **جریان بیشینه**[۲] می‌خواهیم حداکثر جریانی را پیدا کنیم که می‌توان از راس‌های مبدا به راس‌های مقصد منتقل کرد، به طوری که از هر یال حداکثر به اندازه‌ی ظرفیت آن و در جهت مجاز جریان عبور کند. به عنوان مثال در گراف شکل ۱-۵ می‌خواهیم بیش‌ترین جریان از راس $s = ۱$ (مبدا) به راس $t = ۷$ (مقصد) برسد به طوری که از هر یال حداکثر به اندازه ظرفیتش جریان عبور کند. جریان بیشینه گراف شکل ۱-۵ در شکل ۱-۶ نمایش داده شده است، در این گراف اندازه جریان بیشینه برابر ۵ است و به عنوان مثال جریان با اندازه ۳ بر روی یال $(۳, ۶)$ قرار دارد.

تعریف ۱-۲-۱. جریان‌دهی برای گراف وزن دار $G = (V, E)$ با ظرفیت $c_{(u,v)}$ برای یال $e = (u, v) \in E$ به معنی پیدا کردن تابع $f : E \to R^{\geq ۰}$ است به طوری که:

[۱] Network flow
[۲] Maximum flow



شکل ۱-۵: یک شبکه جریان

۱. شرط محدودیت جریان برای هر یال برقرار باشد؛ یعنی $\forall (u,v) \in E : ۰ < f_{(u,v)} \leq c_e$.

۲. شرط بقای جریان برای هر راس به جز مبدا و مقصد برقرار باشد؛ یعنی : $\forall v \in V - \{s,t\}$ $\sum_u f_{(u,v)} = \sum_w f_{(v,w)}$.

به‌عبارت‌دیگر، جمع جریان‌های ورودی به یک راس باید با جمع جریان‌های خروجی از آن برابر باشد، به جز راس‌های s و t که به ترتیب راس‌های مبدا و مقصد هستند.

تعریف ۱-۲-۲. اندازه یک جریان برابر با میزان جریان خارج شده از راس مبدا منهای جریان وارد شده به مبدا می‌باشد:

$$|f| = \sum_{\substack{w \\ (s,w) \in E}} f_{(s,w)} - \sum_{\substack{w \\ (w,s) \in E}} f_{(w,s)}$$

تعریف ۱-۲-۳. افراز مجموعه راس‌های گراف G به دو مجموعه S و $\bar{S} = V - S$ را یک **برش**[1] گویند. هر برش یک مجموعه از یال‌ها را تعریف می‌کند که یک نقطه پایانی آن‌ها در S و نقطه پایانی دیگر آن‌ها در $\bar{S}$ قرار دارد، به این یال‌ها **یال عبوری**[2] گفته می‌شود و این برش را به صورت $[S, \bar{S}]$ نمایش می‌دهند.

اندازه برش در گراف‌های بدون‌وزن برابر با تعداد یال‌های عبوری و در گراف‌های وزن‌دار برابر با مجموع وزن یال‌های عبوری از S به $\bar{S}$ است. شکل ۱-۶ یک برش $[S, \bar{S}]$ که $S = \{۱,۲,۳\}$ و $\bar{S} = \{۴,۵,۶,۷\}$

[1] Cut
[2] Crossing edge

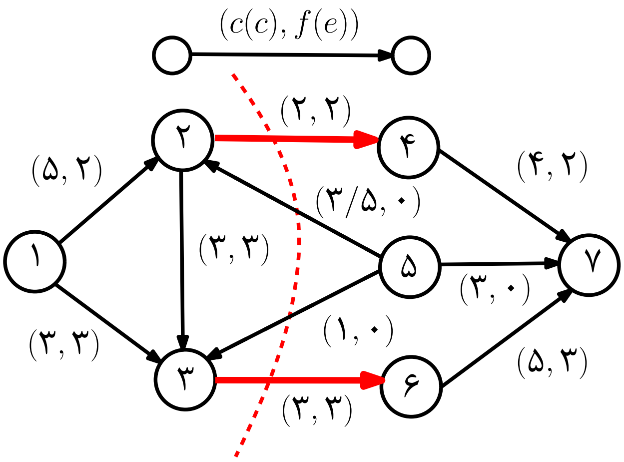
را نمایش می‌دهد. یال‌های عبوری در این برش مجموعه یال‌های $\{(۲,۴),(۳,۶)\}$ می‌باشد، در نتیجه اندازه برش ۵ است.

برشی که در میان همه برش‌ها کم‌ترین اندازه را داشته باشد، **برش‌کمینه** گویند. در شکل ۱-۶ اندازه برش ۵ می‌باشد و برش نمایش داده شده، یک برش‌کمینه نیز هست.

تعریف ۱-۲-۴. برای دو راس داده شده s و t از گراف G، برش $[S,\bar{S}]$ را یک $s-t$-برش می‌گوییم اگر $s \in S$ و $t \in \bar{S}$ باشد.

به عنوان مثال اگر $s = ۱$ و $t = ۷$ باشد، برش نمایش داده شده در شکل ۱-۶ یک $s-t$-برش است؛ ولی اگر $s = ۱$ و $t = ۳$ باشد، این برش یک $s-t$-برش نیست. یکی از قضیه‌ها در بحث برش‌کمینه قضیه شناخته شده **جریان بیشینه-برش‌کمینه** است. این قضیه که یکی از نتایج مهم به دست آمده در رابطه با برش‌کمینه است، توسط فورد[1] و فولکرسون[2] نخستین بار در سال ۱۹۵۶ مطرح شده و به شرح زیر می‌باشد.

قضیه ۱-۲-۵. *[۱۶] (جریان بیشینه-برش‌کمینه): بیش‌ترین میزان جریانی که می‌توان از یک راس مبدا s به یک راس مقصد t در یک شبکه‌ی دارای ظرفیت انتقال داد برابر با ظرفیت $s-t$-برش کمینه است.*



شکل ۱-۶: یک جریان‌دهی بیشینه

[1] Ford
[2] Fulkerson

در گراف‌های ناهمبند برش‌کمینه به سادگی به‌دست‌می‌آید، به این صورت که راس‌های یک مولفه همبندی گراف را در مجموعه S و راس‌های باقی مانده را در مجموعه $\bar{S}$ قرار می‌دهیم. چون در برش، هیچ یال عبوری نداریم، در نتیجه اندازه برش برابر صفر است. در ادامه این تحقیق فرض می‌کنیم که گراف‌های در نظر گرفته شده همگی همبند هستند.

۱-۳ انواع مسائل و کلاس‌های پیچیدگی

۱-۳-۱ مسائل بهینه‌سازی

بسیاری از مسائل در جهان مربوط به بهینه‌سازی می‌باشند. یک مسئله بهینه‌سازی، مسئله‌ای است که به دنبال کمینه کردن یا بیشینه کردن یک تابع هدف می‌باشد. مسائل بهینه‌سازی را می‌توان بر مبنای پیوسته یا گسسته بودن متغیرها، به دو دسته تقسیم‌بندی کرد. یک مسئله بهینه‌سازی با متغیرهای گسسته به عنوان یک مسئله بهینه‌سازی ترکیبیاتی شناخته می‌شود. در این تحقیق منظور از مسائل بهینه‌سازی، بهینه‌سازی ترکیبیاتی است. در ادامه چند نمونه از این مسائل نام برده می‌شود.

مسئله ۱-۳-۱. *مسئله کوله‌پشتی[۱]: فرض کنید که مجموعه‌ای از اشیاء که هرکدام دارای وزن و ارزش خاصی هستند در اختیار دارید. اشیاء را به گونه‌ای انتخاب کنید که مجموع وزن اشیاء انتخاب شده، کوچک‌تر یا مساوی حدی از پیش تعیین شده باشد و ارزش آن‌ها بیشینه شود.*

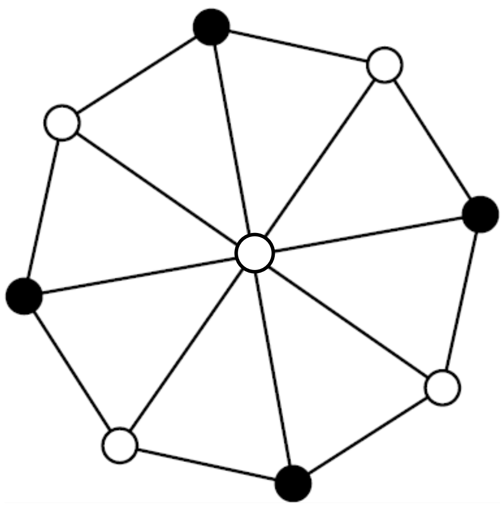
مسئله ۱-۳-۲. *مسئله یافتن کوتاه‌ترین مسیر[۲]: یافتن مسیری بین دو رأس در گراف به گونه‌ای که مجموع وزن یال‌های تشکیل دهنده آن کمینه شود. برای مثال می‌توان مسأله یافتن سریع‌ترین راه برای رفتن از یک مکان به مکان دیگر روی نقشه را، در نظر گرفت؛ در این حالت رأس‌ها نشان دهنده مکان‌ها و یال‌ها نشان دهنده بخش‌های مسیر هستند که برحسب زمان لازم برای طی کردن آن‌ها وزن‌گذاری شده‌اند.*

مسئله ۱-۳-۳. *مسئله فروشنده دوره‌گرد[۳]: تعدادی شهر داریم و هزینه رفتن مستقیم از یکی به دیگری را می‌دانیم. مطلوب است کم‌هزینه‌ترین مسیری که از یک شهر شروع شود و از تمامی شهرها دقیقا یک‌بار عبور*

[۱] Knapsack problem
[۲] Shortest path problem
[۳] Travelling salesman problem

کند و به شهر شروع بازگردد.

مسئله ۱-۳-۴. *مسئله مجموعه مستقل راسی[1]: در این مسئله هدف یافتن بزرگ‌ترین زیرمجموعه از رئوس گراف است به طوری که هیچ دو راسی از مجموعه مجاور نیستند. به عنوان مثال رئوس سیاه رنگ در شکل ۱-۷ یک مجموعه مستقل راسی بیشینه را تشکیل می‌دهد.*



شکل ۱-۷: مجموعه مستقل راسی؛ رئوس سیاه رنگ یک مجموعه مستقل راسی بیشینه را تشکیل می‌دهند.

۱-۳-۲ مسائل تصمیم‌گیری

برای هر مسئله بهینه‌سازی یک مسئله تصمیم‌گیری متناظر وجود دارد که بررسی می‌کند آیا یک راه‌حل کم‌تر یا مساوی برای یک کران وجود دارد یا خیر، مسائل تصمیم‌گیری فقط دارای خروجی بلی یا خیر می‌باشند. در ادامه دو نمونه از این مسائل را معرفی می‌کنیم.

مسئله ۱-۳-۵. *مسئله تصمیم‌گیری مجموعه مستقل راسی: گراف G و عدد k داده شده است، آیا در گراف G مجموعه مستقل راسی به اندازه حداکثر k وجود دارد یا خیر؟*

[1] Independent set problem

مسئله ۱-۳-۶. *مسئله تصمیم‌گیری یافتن کوتاه‌ترین مسیر: گراف G، عدد k و دو راس مشخص به عنوان ابتدا و انتهای مسیر را دریافت کرده و بررسی می‌کند مسیری به وزن حداکثر k بین دو رأس درگراف وجود دارد.*

اگر الگوریتمی وجود داشته باشد که بتواند مسئله تصمیم‌گیری را حل کند، آنگاه الگوریتمی وجود خواهد داشت که مسئله بهینه‌سازی متناظر با آن را حل کند [۵]. به عنوان مثال در مسئله یافتن کوتاه‌ترین مسیر، اگر وزن یال‌ها عدد صحیح باشند، ابتدا طول کوتاه‌ترین مسیر را با استفاده از جستجوی دودویی در پرسش‌های متوالی می‌یابیم، بعد از یافتن اندازه کوتاه‌ترین مسیر بین دو راس s و راس t، هر بار یک یال e را حذف می‌کنیم و الگوریتم تصمیم‌گیری را برای گراف با اندازه کوتاه‌ترین مسیر اجرا می‌کنیم، اگر پاسخ بلی بود، یعنی یال e در مسیر بهینه قرار ندارد و اگر پاسخ خیر بود، یعنی یال e در مسیر بهینه قرار دارد. این روش را برای تمام یال‌های گراف به کار می‌بریم تا مجموعه یال‌های مسیر بهینه به‌دست‌آید.

۱-۳-۳ کلاس‌های پیچیدگی

مواردی هست که می‌دانیم مسئله جواب دارد ولی راه‌حل و جواب آن ارائه نشده است و گاهی علاوه بر مشکل مذکور حتی با در دست داشتن راه‌حل بهینه، منابع و ابزار لازم جهت پیاده‌سازی آن مسئله را نداریم. بعد از این که متوجه شدیم کدام مسائل قابل حل و کدام غیرقابل حل هستند، به طور طبیعی این سوال به ذهن می‌رسد که درجه سختی مسئله چقدر است. برای سادگی کار مسائل به کلاس‌هایی تقسیم می‌شوند که از حیث زمان یا فضا با هم مشابه هستند. این کلاس‌ها در اصطلاح **کلاس‌های پیچیدگی** خوانده می‌شوند. می‌توان دو مورد پیچیدگی زمانی و پیچیدگی فضایی را مورد بحث قرار داد. در اینجا پیچیدگی زمانی را بررسی می‌کنیم، هر چند نتایج بسیاری در مورد پیچیدگی فضایی وجود دارد، اما مطالعه پیچیدگی زمانی مهم‌تر و مفیدتر است. معروف‌ترین این کلاس‌ها، کلاس‌های P و NP هستند، که مسائل را از نظر زمان، تقسیم‌بندی می‌کنند.

از دیگر کلاس‌هایی که در این تحقیق با آن‌ها برخورد می‌کنیم، کلاس‌های NP-سخت، NP-کامل، $DTIME$ و $NTIME$ است. برای تعریف این کلاس‌ها نیاز داریم که با مفهوم کاهش و ماشین تورینگ آشنا باشیم. تعریف کاهش در ادامه بیان می‌شود و برای آشنایی با ماشین تورینگ به [۱۹] مراجعه کنید.

کلاس پیچیدگی NP مجموعه‌ای از همه مسائل تصمیم‌گیری است می‌توان در زمان چندجمله‌ای درستی جواب را بررسی کرد. **کلاس پیچیدگی** P را مجموعه‌ای از مسائل تصمیم‌گیری تعریف می‌کنیم که با استفاده از الگوریتم‌های زمان چندجمله‌ای قابل حل باشند. هر مسئله‌ای که عضو کلاس P باشد، حتما عضو کلاس NP نیز است، به‌عبارت‌دیگر کلاس P زیرمجموعه کلاس NP می‌باشد.

مسئله A به مسئله دیگری مانند B کاهش می‌یابد، اگر هر نمونه از A بتواند به یک نمونه از B تبدیل شود. با این تبدیل یک راه‌حل برای B، راه‌حلی برای A فراهم می‌کند. یک کاهش از یک مسئله به مسئله دیگر ممکن است برای نشان دادن سخت بودن مسئله دوم نسبت به مسئله اول استفاده شود. اغلب یک راه‌حل سریع حل مسئله جدید این است که یک نمونه از مسئله جدید را به نمونه‌ای از مسئله قدیمی تبدیل می‌کنیم و سپس با استفاده از راه‌حل‌های موجود آن را حل می‌کنیم و جواب آن را برای مشخص کردن جواب مسئله جدید به کار می‌بریم.

اگر بتوانیم مسئله $L_۱$ را در زمان چندجمله‌ای به مسئله $L_۲$ کاهش دهیم، آنگاه اگر مسئله $L_۱$ سخت باشد، آنگاه مسئله $L_۲$ نیز سخت است. فرض کنید ما یک مسئله $L_۱$ داریم که اثبات شده حل آن سخت است، و یک مسئله $L_۲$ شبیه به آن داریم. ممکن است حدس بزنیم که حل مسئله $L_۲$ سخت باشد. با استفاده از برهان خلف می‌توان اثبات کرد که مسئله $L_۲$ سخت است. فرض می‌کنیم که حل مسئله $L_۲$ آسان باشد، اگر ما نشان بدهیم که هر نمونه از مسئله $L_۱$ که می‌دانیم سخت است به راحتی با تبدیل آن به مسئله $L_۲$ قابل حل است و آن را حل کنیم، به تناقض می‌رسیم. این اثبات می‌کند که مسئله $L_۲$ نیز سخت می‌باشد.

تعریف ۱-۳-۱. کلاس NP-سخت: یک مسئله تصمیم‌گیری در کلاس NP-سخت قرار دارد اگر هر مسئله عضو کلاس NP در زمان چندجمله‌ای قابل کاهش به آن باشد.

تعریف ۱-۳-۲. کلاس مسائل تمام سخت یا NP-کامل: یک مسئله تصمیم‌گیری L در کلاس پیچیدگی NP-کامل قرار دارد هرگاه در شرایط زیر صدق کند:

۱) مسئله L عضو کلاس NP-سخت باشد

۲) مسئله L عضو کلاس NP باشد.

برای اثبات NP-کامل بودن یک مسئله Q، ابتدا نشان می‌دهیم که مسئله Q در کلاس پیچیدگی NP

قرار دارد و سپس مسئله Q' که یک مسئله شناخته شده در کلاس NP-سخت است را در نظر می‌گیریم و ثابت می‌کنیم که می‌توان مسئله Q' را در زمان چندجمله‌ای به مسئله Q کاهش داد.

تعریف ۱-۳-۳. $DTIME(T(n))$: اشاره به کلاسی از مسائل دارد که برای آن‌ها الگوریتم قطعی در زمان $T(n)$ وجود دارد. به عبارت دیگر $DTIME(T(n))$ کلاس زبان‌هایی است که می‌توان به وسیله یک ماشین تورینگ قطعی در زمان $O(T(n))$ تصمیم‌گیری کرد.

این مشاهده ما را بر آن می‌دارد تا بیان دیگری از کلاس پیچیدگی P به صورت

$$P = \cup_{i \geq 1} DTIME(n^i)$$

در نظر بگیریم. به‌عبارت‌دیگر کلاس P شامل تمام زبان‌هایی است که توسط یک ماشین تورینگ قطعی در زمان چندجمله‌ای پذیرفته می‌شوند، ضمن این که درجه این چندجمله‌ای دارای اهمیت نمی‌باشد.

تعریف ۱-۳-۴. $NTIME(T(n))$: اشاره به کلاسی از مسائل دارد که برای آن‌ها الگوریتم غیرقطعی در زمان $T(n)$ وجود دارد. به عبارت دیگر $NTIME$ کلاس زبان‌هایی است که می‌توان به وسیله یک ماشین تورینگ غیرقطعی در زمان $O(T(n))$ تصمیم‌گیری کرد. الگوریتم‌های غیرقطعی در بخش ۱-۴-۱ بیان شده است.

همچنین می‌توان تعریف دیگری از کلاس پیچیدگی NP به صورت

$$NP = \cup_{i \geq 1} NTIME(n^i)$$

بیان کرد. به‌عبارت‌دیگر کلاس NP شامل تمام زبان‌هایی است که توسط یک ماشین تورینگ غیرقطعی در زمان چندجمله‌ای پذیرفته می‌شوند، ضمن این که درجه این چندجمله‌ای نیز دارای اهمیت نمی‌باشد.

۱-۴ انواع الگوریتم‌ها

۱-۴-۱ الگوریتم‌های قطعی و غیرقطعی

یک **الگوریتم قطعی**[1]، الگوریتمی است که رفتاری قابل پیش‌بینی دارد، با دادن یک ورودی خاص به آن، همیشه خروجی مشخصی را تولید کند به‌عبارت‌دیگر الگوریتم قطعی یک مسیر مشخص را برای رسیدن به جواب طی می‌کند. در مقابل **الگوریتم غیرقطعی**[2] برای یک ورودی خاص ممکن است در اجراهای متفاوت، خروجی یکسان تولید نکند. الگوریتم غیرقطعی چندین مسیر را برای رسیدن به جواب طی می‌کند، با توجه به این نمی‌توان حالت بعدی اجرای الگوریتم را مشخص کرد. به عنوان مثال الگوریتم ۱٫۱ را در نظر بگیرید که عدد طبیعی را دریافت می‌کند و مشخص می‌کند اول است یا خیر. در صورتی که خروجی الگوریتم برای n، خیر باشد، قطعا عدد n اول نیست ولی اگر خروجی الگوریتم برای n بلی باشد، به احتمال زیاد n، عدد اول است. همچنین اگر الگوریتم را چندین بار برای ورودی n اجرا کنیم، ممکن است خروجی‌ها یکسان نباشد.

الگوریتم ۱٫۱: الگوریتم غیرقطعی بررسی اول بودن عدد n

ورودی: عدد طبیعی n
خروجی: تصمیم‌گیری در مورد اول بودن عدد n

برای *۷۰ بار تکرار کن*
  یک عدد تصادفی a انتخاب کن که $1 < a < n$
  اگر $n \% a == 0$ **آنگاه**
    خیر **را برگردان**
بلی **را برگردان**

۲-۴-۱ الگوریتم‌های تقریبی

حل کردن خیلی از مسائل بهینه‌سازی به صورت بهینه کار سختی است. در حقیقت بسیاری از این مسائل NP-سخت هستند. این بدین معنی است که هیچ الگوریتم چندجمله‌ای وجود ندارد که جواب‌بهینه مسئله را

[1] Deterministic algorithm
[2] Nondeterministic algorithm

پیدا کند، مگر این که $P=NP$ باشد. به عنوان مثال برای مسئله فروشنده دوره‌گرد و مسئله یافتن مجموعه مستقل بیشینه، هنوز الگوریتمی با زمان چند جمله ای پیدا نشده است و از جمله مسائل NP–سخت هستند.

در مواجه با این نوع از مسائل سخت، وقتی نمی‌توانیم انتظار داشته باشیم که یک الگوریتم چندجمله‌ای برای حل آن به دست بیاوریم، چه کاری می‌توانیم انجام دهیم؟ در دنیای امروز، تعداد مسائل بهینه‌سازی در حال افزایش است و لازم است که حل شوند و بیشتر آن‌ها NP–سخت هستند. در بعضی موارد که به جواب بهینه نیاز نیست، الگوریتم‌های تقریبی را به کار می‌بریم و به جای پیدا کردن جواب‌های بهینه، بر روی جواب‌های خوب (نزدیک به بهینه) تمرکز می‌کنیم. در این صورت هدف ما پیدا کردن جوابی است که به جواب بهینه نزدیک باشد. برای مثال ما می‌خواهیم، یک الگوریتم برای مسئله فروشنده دوره‌گرد پیدا کنیم که همیشه یک گشتی را پیدا کند که اندازه آن حداکثر α برابر اندازه گشت کمینه باشد.

از نماد $OPT(I)$ برای نمایش جواب‌بهینه مسئله با ورودی I استفاده می‌کنیم. به عنوان مثال در مسئله فروشنده دوره‌گرد، $OPT(P)$ اشاره به طول کوتاه‌ترین گشت روی مجموعه شهرهای P است و وقتی مسئله مجموعه‌های مستقل را بررسی می‌کنیم، $OPT(G)$، اندازه بزرگ‌ترین مجموعه مستقل از گراف G است. برای سادگی گاهی به جای $OPT(I)$، از نماد OPT استفاده می‌کنیم.

در **مسئله کمینه‌سازی**[1] ، ما می‌خواهیم جوابی با کم‌ترین مقدار برای مسئله پیدا کنیم، مسئله فروشنده دوره‌گرد یک مسئله کمینه‌سازی است. یک الگوریتم برای مسئله کمینه‌سازی، االگوریتم α–تقریبی خوانده می‌شود، اگر الگوریتم برای هر ورودی I یک جواب تولید کند که انداره آن حداکثر $\alpha \cdot OPT(I)$ باشد. α به عنوان عامل تقریب برای الگوریتم نامیده می‌شود.

مسئله بیشینه‌سازی[2] ، مسئله‌ای است که ما می‌خواهیم جوابی با بیشترین مقدار برای آن پیدا کنیم. مجموعه‌های مستقل یک مثال از مسئله بیشینه‌سازی است. یک الگوریتم برای مسئله بیشینه‌سازی α–تقریبی نامیده می‌شود اگر الگوریتم برای هر ورودی I جوابی تولید کند که مقدار آن حداقل $\alpha \cdot OPT(I)$ باشد. فرض می‌کنیم در مسائل کمینه‌سازی $\alpha > ۱$ و در مسائل بیشینه‌سازی $\alpha < ۱$ است. بدیهی است که هر چه α به یک نزدیک‌تر باشد، الگوریتم تقریب بهتری از جواب‌بهینه به‌دست می‌آورد.

در بعضی از موارد که جواب بهینه را نمی‌دانیم، می‌توان کران پایین (یا، کران بالا در مسائل بیشینه‌سازی)

[1] Minimization problem
[2] Maximization problem

جواب بهینه را به‌دست‌آورد و اثبات کرد که الگوریتم همیشه جواب α–تقریبی تولید می‌کند. بنابراین پیداکردن کران برای جواب بهینه یکی از مراحل مهم در تحلیل الگوریتم‌های تقریبی است.

یکی از روش‌های طراحی الگوریتم‌های تقریبی، استفاده از برنامه‌ریزی خطی و به کار گیری روش اولیه–دوگان است. در این روش ابتدا یک جواب‌شدنی مسئله دوگان به‌دست‌آورده می‌شود، سپس دنباله‌ای از جواب‌های شدنی و همگرا به نقطه بهینه به نحوی تولید می‌شود که اختلاف بین جواب بهینه مسئله اولیه و جواب‌شدنی دوگان کاهش یابد. از خاصیت دوگانگی ضعیف برای محاسبه عامل تقریب در روش اولیه–دوگان استفاده می‌شود.

قضیه ۱-۴-۱. *(قضیه دوگانگی ضعیف): یک برنامه‌ریزی خطی کمینه‌سازی (اولیه) و دوگان آن را در نظر بگیرید و فرض کنید برنامه‌ریزی خطی و دوگان به صورت زیر باشد:*

$$\begin{array}{lll} P = \min cx & , & D = \max yb \\ Ax \geq b & & yA \leq c \\ x \geq 0 & & y \geq 0 \end{array}$$

اگر $X = (x_1, x_2, ..., x_n)$ *یک جواب شدنی اولیه و* $Y = (y_1, y_2, \ldots, y_m)$ *یک جواب شدنی دوگان باشد، آنگاه:*

$$\sum_{i=1}^{m} b_i y_i \leq \sum_{j=1}^{n} c_j x_j$$

که c_i *و* b_j *ضرایب تابع هدف در مسئله اولیه و دوگان هستند، به رابطه فوق، رابطه دوگانگی ضعیف می‌گوییم و در صورتی که داشته باشیم:*

$$\sum_{i=1}^{m} b_i y_i^* = \sum_{j=1}^{n} c_j x_j^*$$

به آن دوگانگی قوی می‌گوییم که X^* *و* Y^* *جواب‌های بهینه هستند [۲۶].*

۱-۴-۳ الگوریتم‌های ابتکاری

همانطور که در بخش قبل بیان شد، برای حل مسائل سخت می‌توان از الگوریتم‌های تقریبی استفاده کرد. این الگوریتم‌ها می‌توانند پیدا کردن جواب خوب در فاصله‌ی مشخصی از جواب بهینه را تضمین کنند. روش دیگر برای حل مسائل سخت، استفاده از **الگوریتم‌های ابتکاری**[۱] می‌باشد. این الگوریتم‌ها به دنبال جوابی قابل

[۱] Heuristic algorithms

قبول برای مسئله می‌گردند که الزاما بهترین جواب برای آن مسئله نیست و هیچ تضمینی برای خوبی جواب ندارند. اما براساس شواهد و نتایج تجربی، جواب‌ها نزدیک به جواب بهینه هستند. الگوریتم‌های ابتکاری از دسته الگوریتم‌های غیرقطعی هستند و برای یافتن جواب ممکن است مسیرهای متفاوتی را طی کنند و به‌ازای یک ورودی مشخص، خروجی یکتایی ندارند.

۱-۵ مسئله پوشش مجموعه

پوشش مجموعه[1] یکی از مسائل مهم بهینه‌سازی ترکیبیاتی است که مفهوم ساده‌ای دارد و از مسائل پرکاربرد می‌باشد که مطالعه زیادی در مورد آن انجام شده است.

مسئله ۱-۵-۱. *مجموعه مرجع X و مجموعه C شامل دسته‌ای از زیرمجموعه‌های X که اجتماع آن‌ها برابر با X است را در نظر بگیرید که به هر مجموعه C یک وزن اختصاص داده شده است. مسئله پوشش مجموعه، زیرمجموعه‌ای از اعضای C را مشخص می‌کند که اجتماع آن‌ها برابر با X می‌شود و مجموع وزن آن‌ها کمینه است.*

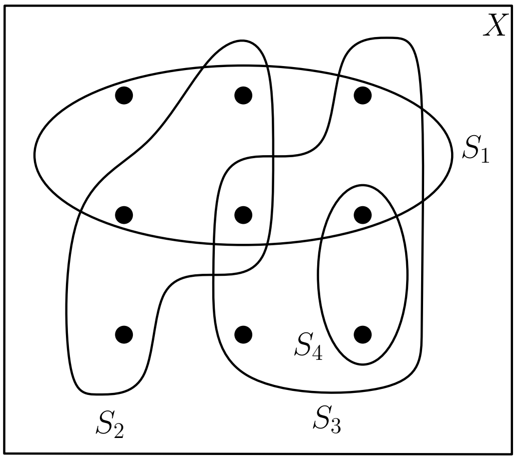
به عنوان مثال یکی از کاربردهای پوشش مجموعه، در تشکیل گروه برای حل مسئله می‌باشد. فرض کنید X مجموعه‌ای از مهارت‌های مورد نیاز برای حل یک مسئله باشد، و مجموعه‌ای از افراد برای کار کردن بر روی این مسئله وجود دارند و هر فرد در یک یا چند زمینه مهارت دارد. می‌خواهیم گروهی را تشکیل دهیم که شامل کم‌ترین تعداد ممکن از افراد باشد، به طوری که برای هر مهارت مورد نیاز در X، عضوی از گروه دارای آن مهارت باشد.

کارپ و همکاران در سال ۱۹۷۲ نشان دادند که مسئله پوشش مجموعه، یک مسئله NP-کامل است [۱۱]. از آنجایی که بعید است این مسئله در زمان چندجمله‌ای حل شود، پژوهش‌های زیادی برای به‌دست‌آوردن الگوریتم‌های تقریبی که جواب آن‌ها نزدیک به جواب بهینه باشد، انجام شده است.

همان‌طور که در بخش ۱-۳-۲ توضیح داده شد است، برای هر مسئله بهینه‌سازی یک مسئله تصمیم‌گیری متناظر با آن نیز وجود دارد. در نسخه تصمیم‌گیری مسئله پوشش مجموعه، ورودی سه‌تایی $\langle X, C, l\rangle$ است که

[1] Set cover

X مجموعه مرجع، C شامل دسته‌ای از زیرمجموعه‌های X و l یک عدد صحیح می‌باشد. این مسئله بررسی می‌کند که آیا پوششی مانند $C' \subset C$ برای X وجود دارد که اندازه آن حداکثر l باشد.



شکل ۱-۸: یک مثال از مسئله پوشش مجموعه

در شکل ۱-۸، یک مثال (X, C) از مسئله پوشش مجموعه نمایش داده شده است که X شامل ۹ عنصر و C شامل ۴ زیرمجموعه $\{S_۱, S_۲, S_۳, S_۴\}$ می‌باشد. کوچکترین زیرخانواده C که مجموعه X را پوشش دهد، برابر است با $C' = \{S_۱, S_۲, S_۳\}$. اگر مسئله تصمیم‌گیری پوشش مجموعه برای مثال شکل ۱-۸ به صورت $< X, C, ۳ >$ مطرح شود، بدین معنی است که آیا زیرخانواده‌ای از C با اندازه حداکثر ۳ وجود دارد که عناصر X را پوشش دهد؟ که جواب مسئله بلی می‌باشد، زیرا مجموعه $C_۱ = \{S_۱, S_۲, S_۳\}$ را می‌توان در نظر گرفت که اندازه آن‌ها حداکثر ۳ است و همه عناصر X را پوشش می‌دهند. ولی اگر مسئله به صورت $< X, C, ۲ >$ مطرح شود، آنگاه هیچ زیرخانواده‌ای وجود ندارد که اندازه آن حداکثر ۲ باشد و همه عناصر X را پوشش دهد، در نتیجه جواب مسئله $< X, C, ۲ >$، خیر می‌باشد.

۱-۶ مسئله جریان با یال-هزینه کمینه

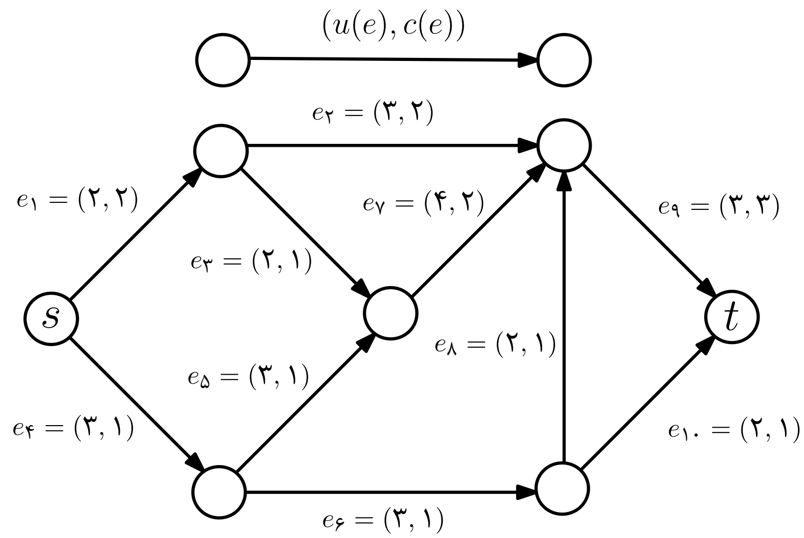
مسئله جریان با یال-هزینه کمینه[1] یکی از مسائل بنیادی شبکه جریان با کاربردهای فراوان است. از جمله کاربردهای مسئله در بهینه‌سازی شبکه‌های غیرهمزمان[2]، حمل‌ونقل[3]، برنامه‌ریزی[4]، مسیریابی[5] و طراحی شبکه[6] است [۷].

مسئله ۱-۶-۱. *مسئله جریان با یال-هزینه کمینه ($MECF$): گراف $G = (V, E)$ با ظرفیت $u(e) \in Z^+$ و هزینه $c(e) \in Z^+$ برای هر یال $e \in E$ در نظر بگیرید. یک جریان f با اندازه F از راس مشخص s به راس مشخص t پیدا کنید به طوری که مجموع هزینه یال‌هایی که جریان از آن‌ها عبور کرده حداقل شود یا به عبارت دیگر اندازه* $\sum_{e\in E, f(e)>0} c(e)$ کمینه شود.

به‌عبارت‌دیگر $MECF$ یک جریان f با هزینه کمینه را در گراف پیدا می‌کند به‌طوری که هزینه جریان برابر است با جمع هزینه یال‌هایی که از آنها جریان عبور می‌کند. به عنوان مثال گراف G در شکل ۱-۹ را در نظر بگیرید. در صورتی که بخواهیم مسئله $MECF$ برای جریان f با اندازه $F = ۳$ حل کنیم، هزینه جریان f برابر با $C = ۷$ می‌شود، که از جمع هزینه روی یال‌های $e_۶, e_۸, e_۹, e_{۱۰}$ و $e_۴$ به دست آمده است، و اگر بخواهیم مسئله $MECF$ برای جریان با اندازه $F = ۵$ حل کنیم، هزینه جریان f برابر با $C = ۱۱$ می‌شود، که از جمع هزینه روی یال‌های $e_{۱۰}, e_۶, e_۴, e_۹, e_۲, e_۱$ و $e_۸$ به‌دست‌آمده است.

مسئله $MECF$ به عنوان یک مسئله NP-سخت شناخته شده و یک الگوریتم F-تقریبی برای آن ارائه شده است[۱۴]. در این جا F نشان دهنده مقدار جریانی است که در مسئله $MECF$ می‌خواهیم از راس s به راس t انتقال دهیم. به عنوان مثال اگر $F = ۳$ باشد، آنگاه برای مسئله $MECF$ یک الگوریتم تقریبی با عامل ۳ وجود دارد. در [۳] ثابت شده است برای مسئله $MECF$ الگوریتم تقریبی با عامل $۲^{\log^{۱-\epsilon} n}$ به‌ازای هر $\epsilon > ۰$ وجود ندارد، مگراینکه $NP \subset DTIME(n^{polygon\, n})$ باشد، این نتیجه با این فرض به‌دست آمده که $F = O(n^۲)$ باشد و n تعداد راس‌های گراف است.

[1] Minimum edge-cost flow
[2] Synchronous networks
[3] Transportation
[4] Scheduling
[5] routing
[6] Network Design



شکل ۱-۹: یک شبکه با هزینه و ظرفیت برای هر یال

قضیه ۱-۶-۱. *مسئله $MECF$ در صورتی‌که هزینه همه یال‌ها توزیع یکنواخت[۱] باشد، هیچ الگوریتم تقریبی با عامل تقریب $۲^{\log^{۱-\epsilon} n}$، به‌ازای‌هر $\epsilon > ۰$ وجود ندارد، مگراین‌که $NP \subset DTIME(n^{polygon\, n})$ باشد. حتی اگر ظرفیت یال‌ها از بین مجموعه $u(e) \in \{۱, poly(n)\}$ انتخاب شود، همچنان مسئله سخت می‌باشد.*

در این فصل، مفاهیمی که در فصل‌های بعد به آنها نیاز داریم، بیان شد. همچنین برخی از ویژگی‌ها، قضیه‌ها و مسائل مرتبط مورد بررسی قرار گرفت. مسئله جریان با یال-هزینه کمینه و مسئله کوچک‌ترین پوشش مجموعه از جمله مسائلی هستند که در این فصل بیان شد. برای اثبات قضیه‌ها و بررسی درستی الگوریتم‌ها از این مسائل استفاده می‌شود.

[۱] Uniform distribution

فصل ۲

مسئله حداقل یال‌اشتراکی

در این فصل، ابتدا مسئله مسیرهای مجزا در گراف را تعریف می‌کنیم و برخی از نتایجی که برای آن وجود دارد را بیان می‌کنیم. سپس مسئله k مسیر با حداقل یال‌اشتراکی را بررسی می‌کنیم. مسئله حداقل یال‌اشتراکی در یک حالت خاص تبدیل به مسئله مسیرهای مجزا می‌شود و یک مسئله NP-سخت است، با این وجود چند الگوریتم تقریبی و روش ابتکاری برای حل آن معرفی می‌شود.

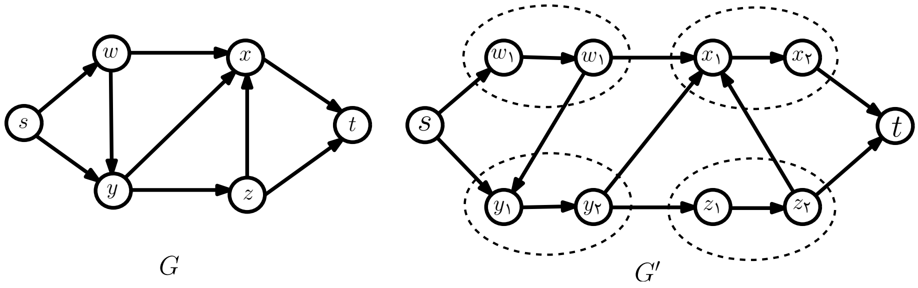
۲-۱ مسیرهای مجزا در گراف

دو مسیر در گراف را **یال‌مجزا** گویند اگر دو مسیر، هیچ یال مشترکی نداشته باشند. در صورتی که دو مسیر هیچ راس مشترکی به غیر از رئوس ابتدا و انتهایشان نداشته باشند به آن‌ها مسیرهای **راس‌مجزا** گویند. دو مسیر در صورتی که راس‌مجزا باشند، یال‌مجزا نیز هستند ولی عکس آن لزوما برقرار نیست. در این پژوهش منظور از یک مسیر مجزا، مسیر راس‌مجزا می‌باشد.

تعریف ۲-۱-۱. به k مسیر در گراف که دو به دو با هم یال‌مجزا باشند، را k مسیر یال‌مجزا گویند و k مسیر که به صورت دو به دو با هم راس‌مجزا باشند را k مسیر راس‌مجزا گویند.

مسئله پیدا کردن k مسیر راس‌مجزا (یال‌مجزا) در گراف G از راس s به راس t یکی از مسائلی است که در نظریه گراف مطرح می‌شود. تعداد مسیرهای راس‌مجزا (یال‌مجزا) در گراف محدود است، بنابراین به‌ازای‌هر k، k مسیر راس‌مجزا (یال‌مجزا) در گراف وجود ندارد. حداکثر تعداد مسیرهای یال‌مجزا از راس s به راس t در گراف $G = (V, E)$ با جریان بیشینه از s به t در شبکه $N = (V, E)$ برابر است، که شبکه N از روی گراف $G = (V, E)$ و نسبت دادن ظرفیت یک واحد به هر یال می‌سازیم. با استفاده از مسئله مسیرهای یال‌مجزا، می‌توان تعداد مسیرهای راس‌مجزا در گراف را نیز به‌دست‌آورد. به‌عبارت‌دیگر مسئله مسیرهای راس‌مجزا در زمان چندجمله‌ای قابل تبدیل شدن به مسئله مسیرهای یال‌مجزا می‌باشد. برای به‌دست‌آوردن حداکثر مسیرهای راس‌مجزا در گراف G، گراف G' را از روی G بدین صورت می‌سازیم؛ ابتدا هر راس $v \in V \backslash \{s, t\}$ را به دو راس $v_۱$ و $v_۲$ تبدیل می‌کنیم و از $v_۱$ به $v_۲$ یک یال $(v_۱, v_۲)$ را به گراف اضافه می کنیم و برای هر یال (u, v) در G، یال $(u_۲, v_۱)$ را به G' اضافه می‌کنیم. برای راحتی کار فرض کنید $s = s_۱ = s_۲$ و $t = t_۱ = t_۲$ باشد (شکل ۲-۱ را ببینید). با این تبدیل تعداد مسیرهای یال‌مجزا در G' برابر با تعداد مسیرهای راس‌مجزا در G'

است [۶].



شکل ۲-۱: جریان بیشینه در G' برابر است با حداکثر تعداد مسیرهای راس‌مجرا در G

مسئله ۲-۱-۱. *گراف $G=(V,E)$ و k جفت از راس‌های $(s_1,t_1),(s_2,t_2),\ldots,(s_k,t_k)$ از گراف G را در نظر بگیرید. هدف از مسئله k مسیر مجزا، پیدا کردن k مسیر مجزای $p_1,p_2,\ldots,p_k$ در صورت وجود است، به طوری که مسیر p_i، مسیری باشد که راس s_i را به t_i وصل می‌کند.*

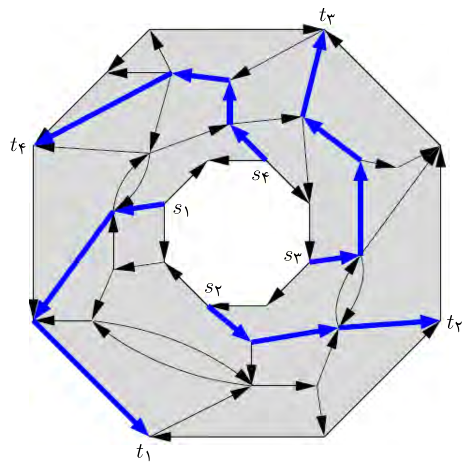
مسئله مسیرهای مجزا، یک مسئله NP-سخت است [۱۲]. اگر k جزء ورودی‌ها نباشد و یک مقدار ثابت باشد، می‌توان k مسیرمجزا را در زمان چندجمله‌ای در گراف‌های مسطح جهت‌دار [۲۳] و همچنین در گراف‌های جهت‌دار و بدون‌دور [۱۵] به‌دست‌آورد، در حالی که مسئله در حالت کلی حتی اگر $k=2$ باشد، برای گراف‌های جهت‌دار NP-سخت است [۱۵]. اگر گراف بدون‌جهت باشد، می‌توان مسیرهای مجزا را در زمان چندجمله‌ای برای مقدار ثابت k به‌دست‌آورد [۲۲، ۲۴، ۲۵]. کرامر[1] و وان لیووین[2] در سال ۱۹۸۴ اثبات کردند که وقتی k ثابت نباشد، مسئله مسیرهای مجزا NP-سخت است، حتی اگر گراف بدون‌جهت و مسطح باشد.

در صورتی که گراف مسطح باشد، در دو حالت خاص می‌توان در زمان چندجمله‌ای مسئله را حل کرد، حتی اگر k ثابت نباشد. حالت اول اگر همه راس‌های انتهایی s_i و t_i بر روی وجه بیرونی گراف باشند. این مسئله را می‌توان با کاهش به مسئله جریان بیشینه در زمان چندجمله‌ای حل کرد [۴]. در حالت دوم، اگر راس‌های انتهایی $s_1,\ldots,s_k$ در یک وجه مشترک S و راس‌های انتهایی $t_1,\ldots,t_k$ در یک وجه مشترک T

[1] Kramer
[2] Van Leeuwen

قرار داشته باشند، به طوری که وجه‌های S و T مجزا باشند (شکل ۲-۲ را ببینید). اریک کولین [۱] و همکاران نشان داده‌اند که می‌توان این مسئله را (در این حالت) نیز در زمان $O(kn \log n)$ حل کرد [۴]. الگوریتم آن‌ها علاوه بر این که k مسیر مجزا را مشخص می‌کند، به ما این تضمین را می‌دهد که مجموع طول مسیرها نیز کمینه شود.

در نسخه بهینه‌سازی مسئله مسیرهای مجزا، هدف به‌دست‌آوردن مسیرهای مجزا با کم‌ترین طول است.



شکل ۲-۲: یک مثال از مسئله مسیرهای مجزا درگراف مسطح و راه‌حل آن (خطوط ضخیم).

این مسئله در حالت‌های مختلف بررسی می‌شود، ممکن است هدف، کمینه کردن مجموع طول مسیرها و یا این که هدف کمینه کردن طولِ طولانی‌ترین مسیر باشد. مسئله در انواع دیگری هم بررسی شده ولی دو نوع بیان شده بیشترین کاربرد را دارد. در ادامه این دو مسئله و چند نوع دیگر از آن بیان و نتایج به دست آمده برای آن‌ها بررسی می‌شود[۱۳].

مسئله ۲-۱-۲. *مسئله k مسیر مجزا با حداقل طول $(Min-Sum)$: درگراف $G=(V,E)$، k جفت از راس $(s_۱,t_۱),(s_۲,t_۲),\dots,(s_k,t_k)$ را در نظر بگیرید. k مسیر مجزا $p_۱,p_۲,\dots,p_k$ به‌گونه‌ای پیدا کنید که مجموع طول مسیرهای $\sum_{i=۱}^{k} l(p_i)$ کمینه شود. $l(p_i)$ طول مسیر p_i درگراف G است، یا به عبارت دیگر $l(p_i)=\sum_{e\in p_i} w(e)$، برای یال e مقدار $w(e)$ درگراف‌های بدون‌وزن، یک و درگراف‌های وزن‌دار برابر با وزن یال e می‌باشد.*

[۱] Eric Colin

	شرایط گراف ورودی و رئوس انتهایی	پیچیدگی
$k = 2$	گراف جهت‌دار	NP-سخت
	جهت‌دار، مسطح، یک وجه	مسئله باز
	بدون‌جهت	مسئله باز
	بدون‌جهت، مسطح، دو وجه	راه‌حل چندجمله‌ای
$k = 3$	بدون‌جهت، مسطح، یک وجه	راه‌حل چندجمله‌ای
k: ثابت باشد	بدون‌جهت	مسئله باز
k: در حالت کلی	بدون‌جهت	NP-سخت
	$s_1 = \cdots = s_k$ و $t_1 = \cdots = t_k$	راه‌حل چندجمله‌ای (MCF)
	مسطح، یک وجه، خوش‌ترتیب	راه‌حل چندجمله‌ای (MCF)

جدول ۲-۱: نتایج به دست آمده برای مسئله k مسیر مجزا با حداقل طول.

اگر $k = 2$ باشد، مسئله $Min - Sum$ در گراف‌های جهت‌دار NP-سخت است [۱۳]، و برای گراف‌های بدون‌جهت و گراف‌های مسطح جهت‌دار به طوری که s_1, s_2, t_1 و t_2 بر روی یک وجه قرار داشته باشند، یک مسئله باز است. در صورتی که گراف، مسطح و بدون‌جهت باشد و راس‌های s_1 و s_2 در وجه S و راس‌های t_1 و t_2 در وجه T قرار داشته باشد به طوری که S و T مجزا باشند، برای مسئله $Min - Sum$ راه‌حل با زمان چندجمله‌ای وجود دارد. اگر $k = 3$ باشد، برای مسئله $Min - Sum$ در گراف‌های بدون‌جهت و مسطح، به طوری که همه راس‌های انتهایی s_i ها و t_i ها در یک وجه قرار داشته باشند، راه‌حل چندجمله‌ای وجود دارد [۱۳].

اگر k یک مقدار ثابت باشد، مسئله $Min - Sum$ برای گراف‌های بدون‌جهت، حل نشده است [۱۳].

اگر k در حالت کلی قرار داشته باشد، یا به عبارت دیگر بتوان هر مقدار بزرگ‌تر از صفر برای k اختیار کرد، در این حالت مسئله $Min - Sum$ برای گراف‌های بدون‌جهت، NP-سخت است. در صورتی که $s_1 = \cdots = s_k$ و $t_1 = \cdots = t_k$ باشد، می‌توان مسئله $Min - Sum$ را در زمان چندجمله‌ای با استفاده از مسئله جریان با هزینه کمینه (MCF) حل نمود [۱۳]. در این حالت به هر مسیر از s به t یک $s - t$-مسیر می‌گوییم. همچنین می‌توان این مسئله را با استفاده از الگوریتم گولدبرگ[1] و روس [2] که در [۸] ارائه شده،

[1] Goldberg
[2] Raos

در زمان $O(m\ \min(n^{\frac{2}{3}}, m^{\frac{1}{2}} \log(n^2/m) \log k))$ در گراف $G = (V, E)$ به‌دست‌آورد، که $m = |E|$ و $n = |V|$. در صورتی که گراف ورودی مسطح باشد و تمام راس‌های انتهایی به ترتیب $s_1, \ldots, s_k, t_1, \ldots, t_k$ بر روی یک وجه ظاهر شده باشند، مسئله $Min - Sum$ را می‌توان با استفاده از مسئله جریان با هزینه کمینه در زمان چندجمله‌ای حل کرد. در یک گراف مسطح، به ترتیبی که راس‌های ابتدایی و انتهایی بر روی یک وجه ظاهر می‌شوند، یک ترتیب، **خوش‌ترتیب** [1] گویند. در گراف‌های مسطح در زمانی که $t_1, \ldots, t_k$ روی وجه T و $s_1, \ldots, s_k$ روی وجه S قرار دارند، می‌توان مسئله $Min - Sum$ را در زمان $O(kn \log n)$ حل کرد، که n تعداد راس‌های گراف ورودی می‌باشد [۱۳]. نتایج به دست آمده برای مسئله $Min - Sum$ به صورت خلاصه در جدول ۲-۱ نمایش داده شده است.

مسئله ۲-۱-۳. *مسئله k مسیر مجزا به طوری که طول ِطولانی‌ترین مسیر کمینه شود ($Min - Max$): در گراف $G = (V, E)$، k مسیر مجزا $p_1, p_2, \ldots, p_k$ به گونه‌ای پیدا کنید به طوری که $\max_i(l(p_i))$ کمینه شود.*

مسئله $Min - Max$، در گراف‌های جهت‌دار و بدون‌دور NP-سخت است، با این وجود در [۱۷] الگوریتم تقریبی با فاکتور ۲ برای آن ارائه شده است. برای مشاهده کلی نتایج جدول ۲-۱ را ببینید.

همان‌طور که قبلا بیان شد، مسئله مسیرهای مجزا در حالت‌های دیگری نیز بررسی شده است. شیلوچ[2] و همکاران در [۱۰] نشان داده‌اند که مسئله پیدا کردن دو $s - t$-مسیر به طوری که طول کوتاه‌ترین و طول ِطولانی‌ترین مسیر به ترتیب با δ_1 و δ_2 کران دار شود، NP-سخت است، δ_1 و δ_2 به ترتیب کران بالا و کران پایین در مسئله هستند. مسئله پیداکردن دو $s - t$-مسیر که طول کوتاه‌ترین مسیر کمینه شود نیز NP-سخت است [۲۷]. لی[3] و همکاران در [۱۸] یک حالت کلی‌تر از مسئله مسیرهای مجزا را بررسی کرده‌اند.

مسئله ۲-۱-۴. *در گراف $G = (V, E)$ دو راس ِمجزای s و t را در نظر بگیرید، به طوری که به ازای هر یال $(u, v) \in E$، k هزینه $c_{uv}^{(1)}, c_{uv}^{(2)}, \ldots, c_{uv}^{(k)}$، با مقدار نامنفی اختصاص داده شده است. هزینه jامین*

[1] Well ordered
[2] Shiloach
[3] Li

مسیر p_j به صورت زیر به دست می‌آید.

$$C^{(j)}(p_j) = \sum_{(u,v)\,\in\,p_j} c^j_{uv}$$

k مسیر مجزا از s به t طوری پیدا کنید که مجموع هزینه‌ها، یعنی $\sum_{j=1}^{k} C^{(j)}(p_j)$، کمینه شود.

به عنوان یک کاربرد از مسئله، فرض کنید در یک شبکه مخابراتی قصد داریم داده[1] و صوت[2] را با کمترین هزینه از مکانی به مکان دیگر هر کدام در مسیری مجزا انتقال دهیم. از آنجایی که هزینه عبور داده و صوت در شبکه با هم برابر نیست، بنابراین باید به‌ازای‌هر یال، یک نوع هزینه برای عبور صوت و یک نوع هزینه برای عبور داده در نظر گرفت. فرض کنید $c^{(1)}_{uv}$، هزینه عبور یک واحد صوت و $c^{(2)}_{uv}$، هزینه عبور یک واحد داده از یال uv باشد. در این صورت می‌توان از مسئله مسیرهای مجزا با کمترین هزینه استفاده کرد و مسیرهای بهینه را برای $k=2$ به‌دست‌آورد. این مسئله در صورتی که گراف ورودی جهت‌دار یا بدون‌جهت باشد، NP-کامل است حتی اگر $k=2$ باشد. در [۱۸] یک الگوریتم ابتکاری با زمان چندجمله‌ای و همچنین یک الگوریتم چندجمله‌ای برای گراف‌های بدون‌جهت و بدون‌دور ارائه شده است.

در صورتی که در مسئله $MECF$، ظرفیت همه کمان‌ها مقدار واحد باشد و بخواهیم جریان f با اندازه k در گراف G از راس s به راس t به دست بیاوریم، آنگاه این مسئله برابر با مسئله kامین کوتاه‌ترین مسیر می‌شود. در مسئله kامین کوتاه‌ترین مسیر هدف یافتن k مسیر مجزا از راس s به راس t به گونه‌ای است که مجموع طول مسیرها کمینه شود. می‌توان این مسئله را به صورت یک برنامه‌ریزی خطی نوشت. فرض کنید که برای یال $e \in E$، $f(e)$ جریان روی یال e و $c(e)$ هزینه یال e باشد و همچنین $\delta^+(i)$، مجموعه یال‌های خروجی از راس i و $\delta^-(i)$، مجموعه یال‌های ورودی به راس i باشد.

$$\begin{array}{lll}
\min & \sum_e c(e)f(e) & \\
s.t. & \displaystyle\sum_{\delta^+(i)} c(e) - \sum_{\delta^-(i)} c(e) & = \begin{cases} k & i = s \\ 0 & i \neq s,t \\ -k & i = t \end{cases} \\
& 0 \le c(e) \le 1 &
\end{array}$$

یک جواب برنامه‌ریزی خطی فوق متناظر با یک جواب برای مسئله kامین کوتاه‌ترین مسیر می‌باشد.

[1] Data
[2] Voice

۲-۲ تعریف مسئله حداقل یال‌اشتراکی

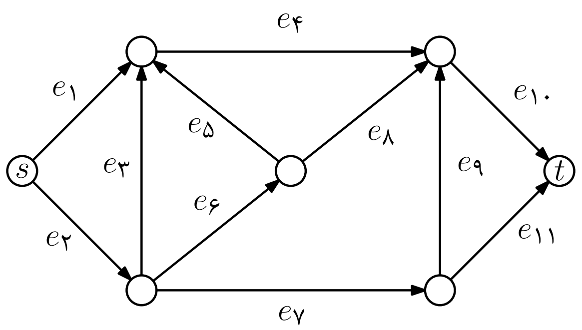
گراف $G = (V, E)$ و دو راس مشخص s و t از آن را در نظر بگیرید، در مسئله مسیرهای یال‌مجزا، هدف پیدا کردن k مسیر یال‌مجزا از s به t بود. تعداد مسیرهای یال‌مجزا در گراف محدود هستند و می‌توان در زمان چندجمله‌ای حداکثر تعداد آن‌ها را به‌دست‌آورد. فرض کنید حداکثر تعداد مسیرهای یال‌مجزا در G برابر با k' باشد. اگر $k > k'$ باشد بدیهی است که پیدا کردن k مسیر یال‌مجزا در G غیرممکن است. این حالت ممکن است در گراف‌های تنک[۱] رخ دهد و از طرفی در عمل بسیاری از گراف‌های کاربردی تنک هستند. در صورتی که لازم باشد همیشه به‌ازای‌هر k، k مسیر از s به t داشته باشیم، **مسئله حداقل یال‌اشتراکی** (MSE)[۲] را بررسی می‌کنیم. این مسئله تعمیمی از مسئله مسیرهای یال‌مجزا است و در یک حالت خاص به مسئله مسیرهای یال‌مجزا تبدیل می‌شود. به یالی که بیش از یک‌بار در مسیرها استفاده شود، **یال‌اشتراکی** می‌گوییم.

مسئله ۲-۲-۱. *مسئله حداقل یال‌اشتراکی (MSE): گراف $G = (V, E)$، دو راس $s, t \in V$ و عدد $k > 0$ را در نظر بگیرید. مجموعه P شامل k مسیر از s به t در G به گونه‌ای پیدا کنید که تعداد یال‌های اشتراکی کمینه شوند.*

مسئله MSE در طراحی بعضی از شبکه‌های ارتباطی و حمل‌ونقل کاربرد دارد. به عنوان مثال فرض کنید یک نهاد امنیتی می‌خواهد برنامه حرکت یک شخص خیلی مهم[۳] که قصد دارد بین دو نقطه در شبکه حرکت کند را مخفی نماید. استراتژی[۴] به این صورت است؛ بسته به این که چقدر تضمین امنیتی لازم است، k مسیر را از قبل به عنوان مسیر پیشنهادی در نظر می‌گیرند. سپس به صورت تصادفی یک مسیر را از بین k مسیر به عنوان مسیر نهایی انتخاب می‌کنند. برای این که احتمال حمله دشمن (که استراتژی و مسیرها را می‌داند) را کاهش دهیم، باید برای یال‌هایی که ریسک بالایی دارند یک نگهبان در نظر بگیریم. به‌عبارت‌دیگر، یال‌هایی که بیش از یک‌بار در مسیرها استفاده شده‌اند، نیاز به محافظت دارند. برای این که هزینه محافظت از یال‌ها را کاهش دهیم، ما باید k مسیر را با حداقل یال اشتراکی پیدا کنیم. بدون از دست دادن کلیت مسئله فرض

[۱] Sparse
[۲] Minimum Shared Edge
[۳] Very important person
[۴] Strategy

می‌کنیم گراف ورودی جهت‌دار است. شکل ۲-۳ یک مثال مسئله MSE را در گراف ساده نمایش می‌دهد. تعداد مسیرهای ممکن از s به t برابر ۶ است که به صورت $\pi_1, \pi_2, \ldots, \pi_6$ نمایش داده شده، برای $k = 2$، دو مسیر از s به t می‌توان به‌دست‌آورد که حداقل تعداد ممکن یال‌اشتراکی، برابر صفر باشد، و دو مسیر π_1 و π_2 به‌دست‌می‌آید. برای $k = 3$، حداقل تعداد یال‌های اشتراکی برابر ۲ است، که به وسیله مجموعه مسیرهای π_1, π_2 و π_3 به‌دست‌آمده، در این مجموعه مسیرها فقط یال‌های e_2 و e_{10} بیش از یک‌بار در مسیرها ظاهر شده‌اند و بر طبق تعریف فقط یال‌های e_2 و e_{10} یال‌اشتراکی هستند. بنابراین برای اینکه سه مسیر از s به t در گراف G به‌دست‌آوریم به حداقل دو یال‌اشتراکی نیاز داریم.



$$\pi_1 = \langle e_1, e_4, e_{10} \rangle$$
$$\pi_2 = \langle e_2, e_7, e_{11} \rangle$$
$$\pi_3 = \langle e_2, e_6, e_8, e_{10} \rangle$$
$$\pi_4 = \langle e_2, e_3, e_4, e_{10} \rangle$$
$$\pi_5 = \langle e_2, e_7, e_9, e_{10} \rangle$$
$$\pi_6 = \langle e_2, e_6, e_5, e_4, e_{10} \rangle$$

شکل ۲-۳: شش $s - t$-مسیر در گراف G، که با π_1 تا π_6 مشخص شده‌اند.

مجموعه مسیرهای P در راه‌حل مسئله MSE یکتا نیست، به عنوان مثال برای $k = 3$ مجموعه $P' = \{\pi_2, \pi_3, \pi_4\}$ نیز می‌تواند جواب باشد، زیرا این مسیر نیز دو یال اشتراکی دارد. ولی هر ترکیب دیگری از مسیرها که در نظر بگیریم، نمی‌توان مجموعه مسیرهایی را انتخاب کرد که کمتر از دو یال‌اشتراکی داشته باشد. در یک حالت خاص اگر تعداد یال‌های اشتراکی مورد نیاز صفر باشد، مسئله حداقل یال‌اشتراکی به مسئله مسیرهای یال‌مجزا تبدیل می‌شود. در مثال شکل ۲-۳ برای $k = 2$ به هیچ یال‌اشتراکی نیاز نداشتیم، به عبارت دیگر می‌توان دو مسیر یال‌مجزا در گراف به‌دست‌آورد. ولی برای $k = 3$ به دو یال‌اشتراکی نیاز بود، این بدین معنی است که نمی‌توان در گراف شکل ۲-۳، سه مسیر یال‌مجزا به‌دست‌آورد.

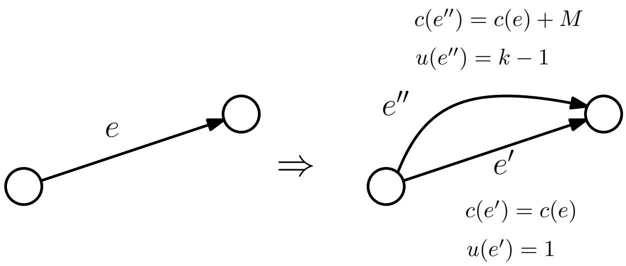
همان‌طور که قبلا توضیح داده شد، در مسئله MSE برای مقدار k محدودیت نداریم و می‌توان به ازای هر عدد صحیح $k > 0$، k مسیر به‌دست‌آورد. در گراف شکل ۲-۳ می‌توان برای $k = 4$، مجموعه مسیرهای

$\{\pi_1, \pi_2, \pi_3, \pi_5\}$ با ۳ یال‌اشتراکی به‌دست‌آورد. هر چه مقدار k افزایش می‌یابد، تعداد یال‌های اشتراکی نیز افزایش پیدا می‌کند. در مسئله MSE هیچ قیدی روی مسیرهای انتخابی نداریم و ممکن است یک مسیر دو یا چند بار در مجموعه مسیرهای راه‌حل انتخاب شده باشد. به عنوان مثال در گراف شکل ۲-۳ برای $k = 4$، مجموعه $\{\pi_1, \pi_1, \pi_1, \pi_1\}$ نیز یک جواب برای مسئله MSE است. هر چه مقدار k بیشتر باشد، احتمال این که مسیرها شبیه به هم باشند نیز افزایش پیدا می‌کند تا جایی که اگر $k > |E|$ باشد، آنگاه حداقل تعداد یال‌های اشتراکی برابر با اندازه کوتاه‌ترین مسیر از s به t می‌شود و همه k مسیر راه‌حل برابر با کوتاه‌ترین مسیر از s به t می‌شود. این نتیجه در بخش ۲-۵-۲ بیان و اثبات می‌شود. در ادامه دو مورد از مسائلی که بسیار شبیه به مسئله حداقل یال‌اشتراکی هستند بیان می‌شود.

مسئله ۲-۲-۲. *مسئله حداقل‌اشتراکی: گراف $G = (V, E)$، دو راس $s, t \in V$، عدد صحیح $k > 0$ و به‌ازای‌هر یال $e \in E$ هزینه $c(e)$ را در نظر بگیرید. مجموعه P شامل k مسیر از s به t به‌گونه‌ای انتخاب کنید که هزینه یال‌های اشتراکی کمینه شود. اگر یال e، λ بار در مسیرها ظاهر شود، هزینه این یال برابر با $(\lambda - 1)c(e)$ است.*

اگر هزینه همه یال‌ها با هم برابر باشد، فرق مسئله حداقل‌اشتراکی با مسئله MSE در این است که اگر یال e، λبار $(\lambda > 1)$ در مسیرها ظاهر شود، در مسئله حداقل‌اشتراکی، $(\lambda - 1)$ بار شمرده می‌شود، ولی در مسئله MSE یک‌بار شمرده می‌شود. مسئله حداقل‌اشتراکی در زمان چندجمله‌ای و با استفاده از جریان بیشینه قابل حل است. در [۲۸] یک الگوریتم دقیق با زمان $O(k|E| + |V| \log |V|)$ پیشنهاد شده است. به طور کلی این الگوریتم از سه گام تشکیل شده است. در گام اول؛ گراف $G' = (V', E')$ را از روی گراف ورودی $G = (V, E)$ بدین صورت می‌سازیم: به‌ازای‌هر یال $e \in E$، دو یال موازی e' و e'' به G' اضافه می‌کنیم و همان‌طور که در شکل ۲-۴ نشان داده شده است هزینه یال‌های e' و e'' را به ترتیب $c(e)$ و $M + c(e)$ و ظرفیت آنها را به ترتیب ۱ و $k - 1$ قرار می‌دهیم، فرض کنید M یک عدد بزرگ‌تر از k باشد. در گام دوم؛ با استفاده از الگوریتم جریان با هزینه کمینه، یک جریان با اندازه k در G' به‌دست‌می‌آوریم، و در گام سوم؛ با استفاده از تجزیه جریان به مسیر، k مسیر تولید می‌شود.

مسئله ۲-۲-۳. *مسئله جریان با هزینه ثابت و متغیر: در گراف $G = (V, E)$، به هر یال که یک هزینه ثابت و متناسب با هر واحد جریان عبوری از آن یک هزینه متغیر اختصاص می‌دهیم. هدف، پیدا کردن جریان با*



شکل ۲-۴: در گام اول الگوریتم حداقل‌اشتراکی هر یال e به دو یال e' و e'' تبدیل می‌شود.

اندازه k، از s به t می‌باشد به طوری که مجموع هزینه‌ها کمینه شود.

بهترین الگوریتم تقریبی برای این مسئله دارای ضریب تقریب $\beta(G) + ۱ + \epsilon$ به‌ازای‌هر $\epsilon > ۰$ می‌باشد [۲]، که در آن $\beta(G)$، $s-t$-برش بیشینه در گراف است. در گراف به $s-t$-برشی که بیش‌ترین اندازه را داشته باشد، $s-t$-برش بیشینه می‌گویند.

۲-۳ اثبات NP-سختی مسئله MSE

در ادامه ثابت می‌شود که نسخه تصمیم‌گیری مسئله حداقل یال‌اشتراکی NP-کامل است. برای اثبات، نشان داده می‌شود که مسئله تصمیم‌گیری مسئله حداقل یال‌اشتراکی در کلاس NP قرار دارد و همچنین می‌توان یک مسئله NP-سخت را به آن کاهش داد.

قضیه ۲-۳-۱. *نسخه تصمیم‌گیری مسئله MSE، NP-کامل است [۲۱].*

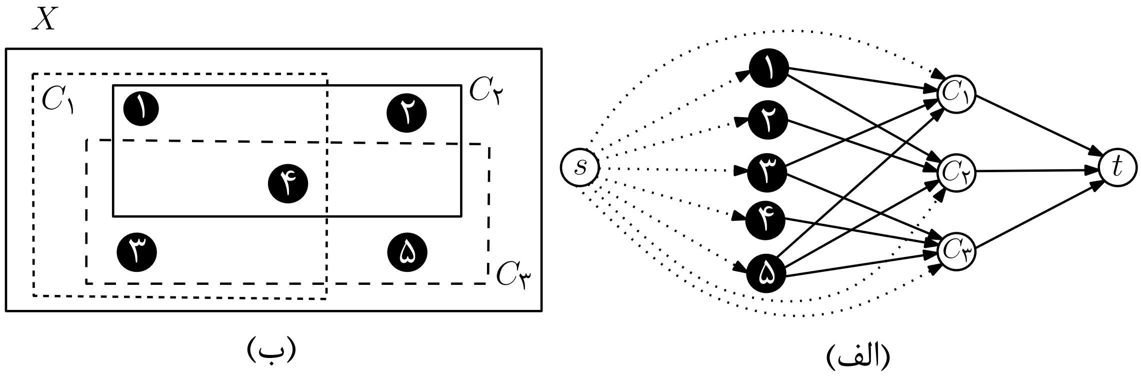
اثبات. نسخه تصمیم‌گیری مسئله MSE به این صورت بیان می‌شود: سه‌تایی $\langle G, k, h\rangle$، که G یک گراف بدون‌جهت با دو راس s و t و $k, h \in \mathbb{N}$ دو عدد می‌باشند، را در نظر بگیرید. آیا مجموعه P شامل k مسیر از s به t وجود دارد به طوری که تعداد یال‌های اشتراکی مسیرهایی که در P قرار دارند، حداکثر h باشد؟ در صورتی که مجموعه P شامل k تا $s-t$-مسیر را داشته باشیم می‌توان در زمان چندجمله‌ای تحقیق کرد که آیا تعداد یال‌های اشتراکی حداکثر h است یا خیر. هر یال می‌تواند حداکثر k بار در $s-t$-مسیرها ظاهر شود

بنابراین اندازه مجموعه P می‌تواند حداکثر k برابر مرتبه G باشد، در نتیجه مسئله MSE یک مسئله NP است.

روش اثبات قضیه بدین صورت است که مسئله تصمیم‌گیری پوشش مجموعه را به مسئله تصمیم‌گیری MSE کاهش می‌دهیم و سپس با استفاده از این مطلب که مسئله تصمیم‌گیری پوشش مجموعه NP-کامل است، نتیجه می‌گیریم که مسئله تصمیم‌گیری حداقل یال‌اشتراکی نیز یک مسئله NP-کامل است. در نسخه تصمیم‌گیری مسئله پوشش مجموعه، ورودی سه‌تایی $\langle X, C, l\rangle$ است که X مجموعه مرجع، C شامل دسته‌ای از زیرمجموعه‌های X و l یک عدد می‌باشد. با استفاده از انتقال، هر نمونه $\langle X, C, l\rangle$ از مسئله تصمیم‌گیری مسئله پوشش مجموعه را به یک نمونه $\langle G, k, h\rangle$ از مسئله MSE کاهش می‌دهیم. برای این کار ابتدا لازم است روش ساخت گراف G را بیان کنیم. به صورت شهودی گراف G را بدین صورت می‌سازیم: به‌ازای‌هر عضو $x \in X$ یک راس v_x و به‌ازای‌هر زیرمجموعه C_i یک راس v_{C_i} را به G اضافه می‌کنیم. در صورتی که عضو x داخل زیرمجموعه C_i قرار داشت، یال (v_x, v_{C_i}) را به گراف G اضافه می‌کنیم. در پایان از s به هر راس v_x، یک یال (s, v_x) و از هر راس v_{C_i} به t یک یال (v_{C_i}, t) اضافه می‌کنیم (شکل ۲-۵ را ببینید). در ادامه روش ساخت به صورت کامل و به زبان ریاضی بیان می‌شود: برای ساخت گراف G ابتدا راس‌های $V = V_X \cup V_C \cup \{t, s\}$ ، که $V_X = \{v_x | x \in X\}$ و $V_C = \{v_{C_i} | C_i \in C\}$ را به G اضافه می‌کنیم. سپس هر راس $v_x \in V_X$ را به هر راس $v_{C_i} \in V_C$ در صورتی که $x \in C_i$ باشد وصل می‌کنیم. علاوه‌براین از هر راس $v_{C_i} \in V_C$ یک یال جهت‌دار به راس t اضافه می‌کنیم، و در پایان از راس s به تمام راس‌های $v \in V_X \cup V_C$ یک مسیر با $l + ۱$ یال اضافه می‌کنیم، به هر کدام از این مسیرها یک **زنجیر**[1] گفته می‌شود. شکل ۲-۵ یک نمونه از ساخت گراف از روی مسئله پوشش مجموعه را نمایش می‌دهد. با قرار دادن $k = |X| + |C|\ ,\ h = l$ ساخت نمونه $\langle G, k, h\rangle$ را از نمونه پوشش مجموعه $\langle X, C, l\rangle$ کامل می‌کنیم. در ادامه نشان می‌دهیم دو نمونه مسئله $\langle G, k, h\rangle$ و $\langle X, C, l\rangle$ هم‌ارز هستند.

فرض کنید مجموعه P شامل k تا $s - t$-مسیر با حداکثر h یال‌اشتراکی در G باشد. نشان می‌دهیم که یک مجموعه $C' \in C$ با $|C'| \leq l$ وجود دارد به طوری که X را پوشش می‌دهد. هر زنجیر در حداکثر یک $s - t$-مسیر ظاهر می‌شود زیرا درغیراین صورت بیشتر از $h = l$ یال‌اشتراکی خواهیم داشت. از آنجایی

[1] Chain



شکل ۲-۵: یک مثال از انتقال مسئله پوشش مجموعه به مسئله MSE

که درجه خروجی s برابر با تعداد مسیرها ($k = |X| + |C|$) می‌باشد، بنابراین هر زنجیر فقط در یک مسیر ظاهر می‌شود. علاوه‌براین، هر راس $v_x \in V_X$ دقیقا در یک $s-t$-مسیر وجود دارد و تنها یک یال خروجی از هر راس $v_x \in V_X$ در P وجود دارد. در نتیجه یال‌های اشتراکی در بین یال‌هایی هستند که یک سر آن‌ها به t وصل است. این یال‌ها را در مجموعه V' قرار می‌دهیم به عبارت دیگر { یک یال‌اشتراکی است $.V' = \{v \in V_C|(v,t)$

یک $s-t$-مسیر را در نظر بگیرید که از راس $v_x \in V_X$ و از راس $v \in V_C$ می‌گذرد. ادعا می‌کنیم که $v \in V'$ است یا به‌عبارت‌دیگر یال (v,t)، یال‌اشتراکی است، زیرا راس v در حداقل دو $s-t$-مسیر ظاهر شده است: یک مسیر $s - v_x - v - t$ که به واسطه‌ی راس V_x، و یک مسیر $s - v - t$ که به واسطه‌ی یک زنجیر از s به v به وجود آمده‌است. بنابراین یال (v,t) در دو مسیر ظاهر شده است و یک یال‌اشتراکی محسوب می‌شود.

زیرگرافی از G شامل تنها یال‌هایی که در P قرار دارند را در نظر بگیرید. در این زیر گراف هر راس $v_x \in V_X$ به یک راس $v \in V'$ وصل شده است و در نتیجه مجموعه $C' = \{C_i | v_{C_i} \in V'\}$ یک پوشش با $|C'| = l$ برای X است. تا اینجا نشان داده شد که می‌توان از یک جواب $\langle G, k, h\rangle$، مسئله MSE به یک جواب از نمونه $\langle X, C, l\rangle$ رسید. در ادامه انتقال را به صورت معکوس بررسی می‌کنیم. فرض کنید مجموعه $C' \subset C$ وجود دارد به طوری که $|C'| \leq l$ و X را پوشش می‌دهد. نشان می‌دهیم در گراف متناظر G، مجموعه P با k، $s-t$-مسیر با حداکثر $h = l$ یال‌اشتراکی وجود دارد.

فرض کنید $V' = \{v_{C_i} \in V_C | C_i \in C'\}$. برای هر $x \in X$ یک $s-t$ –مسیر به نام P_x را به شرح ذیل تعریف می‌کنیم. از راس s شروع می‌کنیم و زنجیری که به v_x وصل است را دنبال می‌کنیم. از آنجایی که x در یکی از مجموعه‌های $C_i \in C$ قرار دارد بنابراین یال (v_x, v_{C_i}) که در گراف وجود دارد را دنبال می‌کنیم. در ادامه یالی که v_{C_i} را به t وصل می‌کند، دنبال می‌کنیم. چون هر راس x در حداقل یک مجموعه $C_i \in C'$ قرار دارد و تعداد آن‌ها $|X|$ است، مجموعه $P_X = \{P_x | x \in X\}$ شامل $|X|$ تا $s-t$ –مسیر است. سپس مجموعه P_C را تعریف می‌کنیم که به‌ازای‌هر $C_i \in C$ یک مسیر شامل زنجیر (s, v_{C_i}) و یال (v_{C_i}, t) می‌باشد، بنابراین P_C شامل $|C|$ تا $s-t$ –مسیر است. اگر $P = P_X \cup P_C$ را در نظر بگیریم، به سادگی مشاهده می‌شود، هر یال (v_{C_i}, t) که $C_i \in C'$ می‌تواند در حداقل دو مسیر در P ظاهر شود: (v_{C_i}, t) حداقل یک‌بار در مسیر $s - v_x - v_{C_i} - t$ ظاهر می‌شود زیرا مجموعه $C_i \in C'$ حداقل یک عضو x دارد و طبق تعریف مجموعه P_C به‌ازای‌هر $C_i \in C$ یک مسیر $s - v_{C_i} - t$ در P_C وجود دارد. از آنجایی که راس‌های $V_C \setminus V'$ در مسیرهای P_X ظاهر نشده‌اند، هر یال (v, t) که $v \in V_C \setminus V'$ باشد دقیقا در یک مسیر P استفاده شده است، در نتیجه تعداد یال‌های اشتراکی در P برابر با $h = |V'|$ است. بنابراین می‌توان هر جواب $\langle X, C, l\rangle$ از مسئله تصمیم‌گیری پوشش مجموعه را به یک جواب از نمونه $\langle G, k, h\rangle$ مسئله MSE تبدیل کرد.

در روند بالا نشان داده شد، که مسئله تصمیم‌گیری پوشش مجموعه را می‌توان به مسئله تصمیم‌گیری MSE، در زمان چندجمله‌ای کاهش داد، بنابراین مسئله تصمیم‌گیری MSE یک مسئله NP –کامل است.

$\square$

می‌توان مسئله حداقل یال‌اشتراکی را با استفاده نسخه تصمیم‌گیری آن حل نمود، در نتیجه مسئله حداقل یال‌اشتراکی به سختی نسخه تصمیم‌گیری آن می‌باشد، از طرفی اگر یک جواب از مسئله حداقل یال‌اشتراکی را داشته باشیم، نمی‌توان در زمان چندجمله‌ای تصمیم‌گیری کرد که جواب بهینه است یا خیر، زیرا باید به صورت بورت فورس تمام حالت‌ها را بررسی کنیم و تعداد حالت‌ها نمایی است، در نتیجه مسئله حداقل یال‌اشتراکی، یک مسئله NP –سخت است.

۲-۴ الگوریتم تقریبی برای مسئله MSE

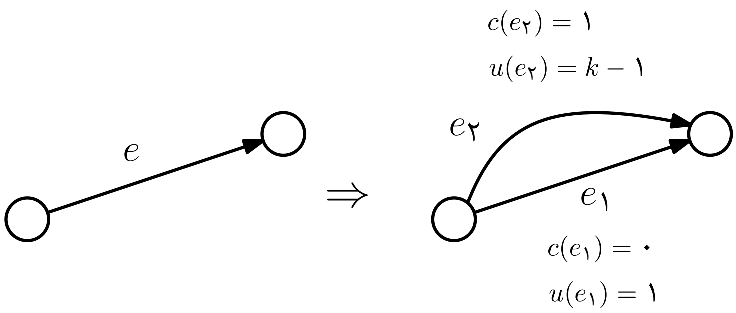
۲-۴-۱ الگوریتم k-تقریبی

همانطور که در بخش قبل بیان شد، مسئله MSE یک مسئله NP-سخت است و الگوریتم چندجمله‌ای تاکنون برای آن بدست نیامده است. معمولا برای حل این‌گونه از مسئله‌ها از الگوریتم‌های تقریبی استفاده می‌شود و به جای جواب‌بهینه، یک تقریبی از جواب‌بهینه به‌دست‌می‌آورند. در ادامه یک الگوریتم تقریبی برای مسئله حداقل یال‌اشتراکی با استفاده از مسئله جریان با یال-هزینه کمینه ارائه می‌شود [۲۱].

قضیه ۲-۴-۱. *مسئله MSE را می‌توان به مسئله $MECF$ کاهش داد.*

اثبات. هر نمونه از مسئله MSE در گراف $G = (V, E)$ را به یک نمونه از مسئله $MECF$ در گراف $G' = (V', E')$ تبدیل می‌کنیم. دقت کنید که گراف‌های G و G' جهت‌دار هستند. کاهش بدین صورت انجام می‌شود، مجموعه $V' = V$ و به‌ازای‌هر یال $e \in E$، دو یال جهت‌دار $e_۱$ و $e_۲$ در همان جهت e در گراف G به مجموعه یال E' اضافه می‌کنیم و هزینه و ظرفیت یال‌های $e_۱$ و $e_۲$ را به صورت $c(e_۱) = ۰$ ، $u(e_۱) = ۱$ ، $c(e_۲) = ۱$ و $u(e_۲) = k - ۱$ قرار می‌دهیم (شکل ۲-۶ را ببینید).

هر راه‌حلی با هزینه L برای $MECF$ در گراف G' متناظر با پیدا کردن $k = F$ مسیر در گراف G با L یال‌اشتراکی است. مجموعه یال‌هایی را در نظر بگیرید که در راه‌حل $MECF$، جریان روی آن‌ها مثبت است و هزینه ۱ دارند. یال‌های متناظر آن‌ها در گراف G، دقیقا همان یال‌هایی هستند که در راه‌حل MSE یال‌اشتراکی هستند. برعکس، هر راه‌حل MSE با L یال‌اشتراکی روی گراف متناظر با یک راه‌حل $MECF$ با هزینه L روی گراف G' است. □



شکل ۲-۶: ساختن یال‌های $e_۱$ و $e_۲$ در گراف G' از روی یال e در گراف G.

اگر یک الگوریتم تقریبی برای مسئله $MECF$ وجود داشته باشد، آنگاه می‌توان یک الگوریتم تقریبی برای مسئله MSE نیز به‌دست‌آورد. کرامک در [۱۴]، یک الگوریتم F-تقریبی برای مسئله $MECF$ بیان کرده است. اگر یک راه‌حل تقریبی با جریان f و با اندازه F از مسئله $MECF$ در گراف G' به دست آوریم، می‌توان با استفاده از معکوس روش انتقالی که در قضیه ۲-۴-۱ بیان شده، گراف G را از روی G' به‌دست‌آورد و یک راه‌حل تقریبی برای مسئله MSE در G' با $k = F$ به‌دست‌آورد که حداکثر L یال‌اشتراکی دارد.

تاکنون یک الگوریتم تقریبی با عامل k برای مسئله MSE بیان شده است. در ادامه یک انتقال دیگر از مسئله MSE به مسئله $MECF$ مطرح می‌شود که در مسئله $MECF$ حداکثر ظرفیت یال‌ها $k - ۱$ باشد. بنابراین می‌توان یک الگوریتم تقریبی بهتر برای مسئله MSE با عامل $k - ۱$ به‌دست‌آورد. به عبارت دیگر می‌توان عامل تقریب را از k به $k - ۱$ کاهش داد.

مسئله ۲-۴-۱. *مسئله جریان با هزینه کمینه[۱] (MCF): گراف $G = (V, E)$ با ظرفیت $u(e) \in Z^+$ و هزینه $c(e) \in Z^+$ را برای هر یال در نظر بگیرید. جریان f با اندازه F از راس مبدا s به راس مقصد t به گونه‌ای پیدا کنید که حاصل $\sum_{e \in E} c(e)f(e)$ کمینه شود و به‌ازای‌هر یال $f(e) \leq u(e)$ باشد.*

به عنوان مثال گراف G در شکل ۱-۹ را در نظر بگیرید. در صورتی که مسئله MCF را برای جریان f با اندازه $F = ۳$ حل کنیم، هزینه جریان f برابر با $C = ۷$ می‌شود، که جریان روی یال‌های $e_۴, e_۶, e_{۱۰}, e_۸$ و $e_۹$ به ترتیب $۳, ۳, ۲, ۱, ۱$ به دست می‌آید.

همان طور که قبلا ذکر شده است، مسئله $MECF$ به عنوان یک مسئله NP-سخت شناخته شده

[۱] Minimum-cost flow

است و یک اثبات برای آن در [۱۴] ارائه شده است. می‌توان با توجه به مطالب بیان شده، اثبات دیگر برای NP-سخت بودن مسئله $MECF$ ارائه داد. در قضیه ۲-۴-۱ بیان شد که می‌توان در زمان چندجمله‌ای مسئله MSE را به مسئله $MECF$ کاهش داد، از طرفی در قضیه ۲-۳-۱ بیان شده که مسئله MSE، یک مسئله NP-سخت است. در صورتی که این دو قضیه را با هم در نظر بگیریم، یک اثبات جدید برای NP-سخت بودن مسئله $MECF$ به‌دست‌می‌آید.

۲-۴-۲ الگوریتم $(k-1)$-تقریبی

فرض کنید $G' = (V', E')$ گرافی باشد که از G با استفاده از قضیه ۲-۴-۱ به دست آمده باشد. گراف G'' از روی G' به این صورت ساخته می‌شود: هزینه هر یال را در G'' برابر با $c(e)/u(e)$ قرار می‌دهیم به طوری که $c(e)$ و $u(e)$ به ترتیب هزینه و ظرفیت یک یال در G' است و ظرفیت هر یال در G'' را همان ظرفیت یال در G' قرار می‌دهیم. فرض کنید OPT' هزینه راه‌حل بهینه مسئله $MECF$ روی G' باشد و OPT'' هزینه راه‌حل بهینه مسئله MCF روی گراف G'' باشد. f را جریان متناسب با جواب بهینه OPT'' روی G'' در نظر بگیرید. از آنجایی که ظرفیت یال‌ها در G' و G'' با هم مشابه است، بنابراین جریان f در G' نیز معتبر می‌باشد. مجموعه یال‌هایی که جریان روی آنها مثبت است را با E^+ نمایش می‌دهیم. در نتیجه داریم:

$$\begin{aligned} OPT' = \sum_{e\in E^+} c(e) &= \sum_{e\in E^+} u(e)\frac{c(e)}{u(e)} \\ &\leq (k-۱)\sum_{e\in E^+}\frac{c(e)}{u(e)} \\ &\leq (k-۱)\sum_{e\in E^+}\frac{c(e)}{u(e)}f(e) = (k-۱)OPT''. \end{aligned}$$

اولین نابرابری برقرار است زیرا ظرفیت هر یال، بر اساس نحوه ساخت G' در قضیه ۲-۴-۱ حداکثر $k-۱$ است و دومین نابرابری نیز برقرار می‌باشد زیرا برای همه یال‌هایی که در E^+ قرار دارند $f(e) \geq ۱$. فرض کنید OPT هزینه راه‌حل بهینه در مسئله MSE روی G باشد، از ترکیب با قضیه ۲-۴-۱ داریم

$$OPT = OPT' \leq (k-۱)OPT''.$$

بنابراین، هر جواب بهینه برای MCF یک جواب $(k-1)$-تقریبی برای مسئله MSE است. الگوریتم‌های کارایی برای مسئله MCF وجود دارد. بهترین الگوریتم برای شرایط ما توسط آهوجا[1] در سال ۱۹۹۲ ارائه شده [۱] و در زمان $O(nm\log(nC)\log\log U)$ اجرا می‌شود که در آن C, m, n و U به ترتیب تعداد راس‌ها، یال‌ها، حداکثر هزینه یال و حداکثر ظرفیت یال می‌باشد. از آنجایی که در تبدیل ما $C = ۱$ و $U = k - ۱$ است، نتیجه زیر به دست می‌آید.

قضیه ۲-۴-۲. *یک الگوریتم $(k-1)$-تقریبی برای مسئله حداقل یال‌اشتراکی وجود دارد که در زمان $O(nm\log n\log\log k)$ اجرا می‌شود.*

در صورتی که گراف سری-موازی ۱-۱ باشد، برای مسئله $MECF$ یک الگوریتم $(1+\epsilon)$-تقریبی با زمان اجرای چندجمله‌ای ارائه شده است[۱۴]. بنابراین وقتی گراف سری-موازی باشد برای مسئله MSE، یک الگوریتم $(1+\epsilon)$-تقریبی با زمان $O(m^۳(۱+۱/\epsilon)\log k)$ وجود دارد.

۲-۵ کران‌پایین تقریب برای مسئله MSE

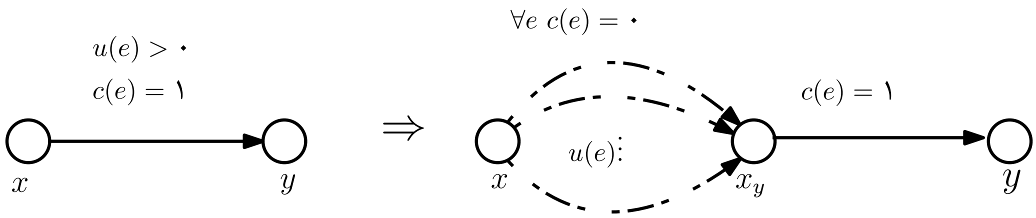
قضیه ۲-۵-۱. *مسئله MSE به‌ازای‌هر $\epsilon > ۰$ تقریب $۲^{\log^{۱-\epsilon} n}$ را نمی‌پذیرد، مگر این که $NP \subseteq DTIME(n^{Polylog\, n})$.*

اثبات. برای اثبات قضیه یک نمونه از مسئله $MECF$ که هزینه یال‌های آن یکسان است را به یک نمونه از مسئله MSE تبدیل می‌کنیم و نشان می‌دهیم که حل دو مسئله با هم معادل هستند. فرض کنید $\mathcal{P}$ یک مسئله $MECF$ باشد که می‌خواهیم در گراف G یک $s-t$-جریان f پیدا کنیم، به طوری که مجموع هزینه یال‌هایی که در جریان f از آن‌ها استفاده شده، کمینه شود یا به عبارت دیگر $\sum_{e\in E, f(e)>۰} c(e)$ حداقل شود. مقدار جریان

[1] Ahuja

f را با F و هزینه آن را با C نمایش می‌دهیم. درگراف G به هر یال یک ظرفیت $u(e) \in \{۱, poly(n)\}$ که $n = |V|$ است و یک هزینه یکسان $c(e) = ۱$ به همه یال‌ها اختصاص می‌دهیم.

گراف $G' = (V', E')$ را از روی $G = (V, E)$ به صورتی‌که در ادامه بیان می‌شود، می‌سازیم. به ازای هر راس $x \in V$ یک راس x متناظر با آن در V' اضافه می‌کنیم و سپس یک مجموعه یال بین راس‌های x و y به گراف G' اضافه می‌کنیم. مجموعه یال شامل $u(e)$ تا زنجیر از x به x_y و یک یال جهت‌دار از x_y به y است. هر زنجیر شامل $|E| + ۱$ یال جهت‌دار است که هزینه روی آن‌ها صفر و ظرفیت آن‌ها بی نهایت می‌باشد. مجموعه یالی که متناظر با یال (x, y) است را با سه تایی (x, x_y, y) نمایش می‌دهیم، همچنین زنجیرهایی که x را به x_y وصل می‌کند، **زنجیر نوع ۱** می‌خوانیم. علاوه بر این دو راس s' و t' را نیز به G' اضافه می‌کنیم و با F زنجیر s را به s' و t را به t' وصل می‌کنیم. این نوع از زنجیرها را **زنجیر نوع ۲** می‌خوانیم. در پایان به ازای هر سه تایی (x, x_y, y)، یک زنجیر از s' به x_y و یک زنجیر از y به t' اضافه می‌کنیم. به این نوع از زنجیرها، **زنجیر نوع ۳** می‌گوییم (شکل ۲-۸ را ببنید). فرض کنید $\mathcal{P}'$ مسئله پیداکردن $k = |E| + F$ تا



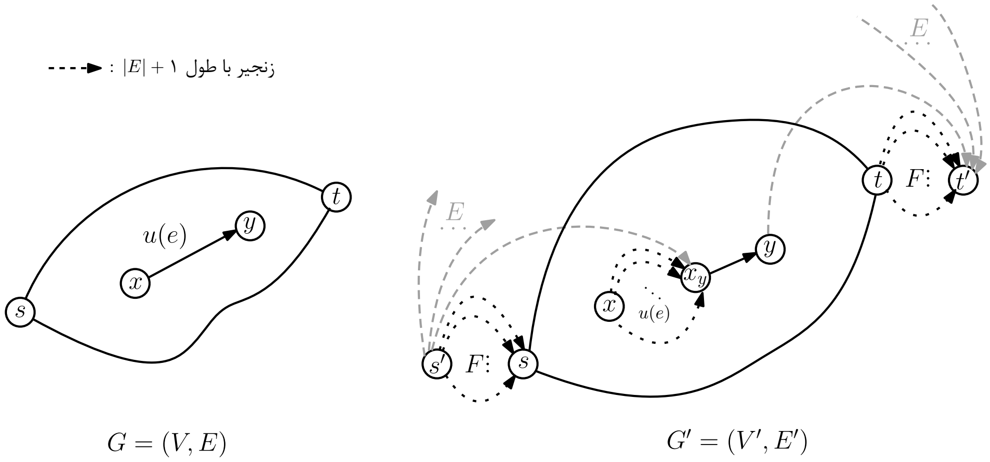
شکل ۲-۷: تبدیل یک یال در $MECF$ به مجموعه‌ای از یال‌ها در MSE. خط‌چین‌ها، زنجیره‌هایی به طول $|E| + ۱$ هستند.

$s - t$-مسیر در $G' = (V', E')$ با حداقل یال‌اشتراکی $S \leq |E|$ باشد. تعداد یال‌های اشتراکی را S می‌نامیم و نشان می‌دهیم که جواب‌ها برای $\mathcal{P}$ و $\mathcal{P}'$ یک‌به‌یک با هم متناظر هستند. برای این کار ابتدا اثبات می‌کنیم که هر جواب برای $\mathcal{P}'$ یک جواب برای $\mathcal{P}$ است و سپس معکوس آن را ثابت می‌کنیم.

یک جواب برای $\mathcal{P}'$ یک مجموعه P شامل k تا $s' - t'$-مسیر در G' است به طوری‌که تعداد یال‌های اشتراکی $S \leq |E|$ باشد. دقت کنید‌که هرکدام از زنجیرها در G' حداکثر یک بار می‌توانند در یک مسیر ظاهر شوند، درغیراین صورت اگر یکی از زنجیرها در دو مسیر ظاهر شود آنگاه تمام یال‌های آن زنجیر یال‌اشتراکی می‌شوند. از طرفی تعداد یال‌های هر زنجیر برابر با $|E| + ۱$ است در نتیجه تعداد یال‌های اشتراکی بیشتر از

$|E|$ می‌شود. از آنجایی که درجه خروجی s' در G' برابر $k = |E| + F$ است، هر زنجیر که از s' خارج می‌شود دقیقا در یکی از مسیرهای P ظاهر می‌شود و همچنین هر زنجیر که به t' وارد می‌شود می‌تواند دقیقا در یک مسیر P ظاهر شود. علاوه بر این هر یال (x, y) از مجموعه یال (x, x_y, y) حداقل در یک مسیر P ظاهر شده، بنابراین (x_y, y) تنها یال‌هایی هستند که می‌توانند در مسیرهای مجموعه P بیشتر از یک‌بار ظاهر شوند یا به عبارت دیگر یال‌اشتراکی باشند.

حال متناظر با P، یک جریان f' با اندازه $|E| + F$ از s' به t' در نظر بگیرید. با تغییر و حذف بعضی از یال‌ها در G'، جریان f' را به یک $s - t$–جریان f با اندازه F در گراف G تبدیل می‌کنیم و نشان می‌دهیم تعداد یال‌های اشتراکی جریان f' در G' برابر با هزینه $s - t$–جریان در G است، بنابراین $S = C$ (C هزینه جریان f در G و S تعداد یال‌های اشتراکی در G' است). ابتدا به‌ازای‌هر مسیر p به



شکل ۲–۸: کاهش مسئله $MEFC$ به مسئله MSE با هزینه واحد برای هر یال.

شکل $s' \dashrightarrow x \dashrightarrow x_y \rightarrow y \dashrightarrow t'$ که $\dashrightarrow$ نمایش یک زنجیر است، جریان با مقدار یک واحد که روی p قرار دارد را از f' حذف می‌کنیم. با این کار جریان روی هر زنجیر نوع ۳ برابر صفر می‌شود و در نتیجه می‌توانیم زنجیرهای نوع ۳ را از گراف حذف کنیم. به‌ازای‌هر مجموعه یال (x, x_y, y) متناظر با یال (x, y) در G جریان روی (x', y) حداکثر $u(e)$ است. از آنجایی که جریان روی x_y از x می‌آید و به y ادامه پیدا می‌کند، در نتیجه می‌توان زنجیرهای نوع ۱ را متراکم کرد و مجموعه یال (x, x_y, y) را با یال (x, y) که دارای ظرفیت $u(e)$ و هزینه $c(e) = ۱$ است جایگزین کرد. به طور مشابه زنجیرهای نوع ۲ را متراکم و s را با s' و t را با

t' ادغام می‌کنیم. در نتیجه گراف حاصل و گراف G هم‌ریخت[1] هستند. جریان جدید f' روی G' متناظر با $s-t$-جریان f روی گراف G با مقدار F است.

مشاهده می‌کنیم که در f' یال‌هایی که جریان روی آن‌ها مثبت است دقیقا همان یال‌هایی هستند که جریان روی آن‌ها در f' قبل از کاهش بزرگ‌تر از یک است یا به‌عبارت‌دیگر این یال‌ها متناظر با یال‌های اشتراکی در مسئله $\mathcal{P}'$ در گراف G' هستند. هزینه هر یال در G برابر ۱ است و مجموع هزینه‌های جریان f' برابر تعداد یال‌های اشتراکی است، یعنی $C = S$. در صورتی که روند بالا را به صورت معکوس طی کنیم، می‌توانیم نشان دهیم که هر جواب در $\mathcal{P}$، یک جواب برای $\mathcal{P}'$ نیز هست. بنابراین جواب‌های دو مسئله یک‌به‌یک با هم متناظر هستند.

اندازه گراف G' از $O(|V| + |E|(F + \sum_{e\in E} u(e)))$ است و توجه کنید که $u(e) \in \{۱, poly(n)\}$ می‌باشد، در نتیجه کاهش مسئله MSE به مسئله $MECF$ در زمان چندجمله‌ای قابل انجام است. هرگاه $F \leq |V^۲|$ باشد می‌توان از قضیه ۱-۶-۱ استفاده کرد، بنابراین کران‌پایین تقریب‌پذیری مسئله $MECF$ برای مسئله MSE نیز برقرار است[۲۱]. $\square$

کران‌پایین تقریب‌پذیری که در قضیه فوق برای مسئله MSE به‌دست‌آورده شده است، بستگی به اندازه گراف ورودی دارد. از طرفی کرانی برای مقدار k نداریم و طبق تعریف مسئله، k می‌تواند هر مقداری بگیرد، و لذا با این شرایط نمی‌توان به طور مستقیم از قضیه فوق استفاده کرد. براساس قضیه‌ای که در ادامه بیان می‌شود، یک کران برای k به‌دست‌می‌آید.

قضیه ۲-۵-۲. *اگر $k > |E|$ باشد، آنگاه حداقل تعداد یال‌اشتراکی در گراف برابر اندازه کوتاه‌ترین $s-t$-مسیر است.*

اثبات. به ازای هر مجموعه از k مسیر برای $k > |E|$، باید یک مسیر از s به t پیدا کنیم به طوری که همه یال‌های مسیر، یال‌اشتراکی باشد. در غیراین صورت هر مسیر نیاز به حداقل یک یال متفاوت از دیگر مسیرها دارد، که نیاز است بیش از $|E|$ تا یال داشته باشیم، که این امکان‌پذیر نیست.

[1] Isomorphic

به‌ازای‌هر مجموعه از k مسیر که $k > |E|$، کوتاه‌ترین مسیر از s به t را انتخاب و k بار آن را گزارش می‌کنیم، بنابراین همه یال‌های روی مسیر، یال‌اشتراکی می‌شوند. $\square$

از قضیه ۲-۵-۲ می‌توان نتیجه گرفت که هرگاه $k > |E|$ باشد، مسئله MSE در زمان چندجمله‌ای قابل حل است. فرض می‌کنیم که $k \leq |E| = O(|V|^۲)$ باشد، بنابراین شرایط قضیه ۲-۵-۱ برقرار است، در نتیجه کران پایین تقریب مسئله MSE به‌ازای‌هر $\epsilon > ۰$ برابر با $۲^{\log^{۱-\epsilon} n}$ است.

۲-۶ روش‌های ابتکاری

در این بخش چند روش ابتکاری برای مسئله MSE بیان می‌شود و این روش‌ها به همراه الگوریتم $(k-1)$-تقریبی پیاده‌سازی و نتایج با هم مقایسه می‌شوند[۲۱].

۲-۶-۱ روش به‌هنگام‌سازی پی در پی

الگوریتم تقریبی که در بخش ۲-۴-۲ برای مسئله MSE توضیح داده شده است، براساس الگوریتم جریان با هزینه کمینه (MCF) طراحی شده است و یک جواب $(k-1)$-تقریبی برای مسئله $MECF$ به‌دست‌می‌آورد، که با استفاده تبدیل قضیه ۲-۴-۱ می‌توان گفت که الگوریتم یک جواب $(k-1)$-تقریبی برای مسئله MSE به‌دست‌آورده است. گراف ورودی که توسط الگوریتم MCF دریافت می‌شود، یک گراف انتقال یافته از روی گراف اولیه می‌باشد، با این ویژگی که هزینه هر یال در مجموعه $\{۰, ۱/(k-۱)\}$ قرار دارد. هرگاه الگوریتم یک یال e که هزینه $۱/(k-۱)$ دارد را انتخاب می‌کند، دو حالت ممکن است رخ دهد: حالت اول، بر روی یال e هیچ جریانی نیست و در حالت دوم، یک جریان مثبت بر روی یال e قرار دارد. از نظر الگوریتم این دو حالت با هم برابر است، زیرا در مسئله MCF می‌خواهیم $\sum_{e \in E} f(e)c(e)$ کمینه شود و تعداد یال‌ها مهم نیستند. از طرفی انتخاب هر یالی که هزینه $۱/(k-۱)$ دارد، متناظر با در نظر گرفتن یک یال اشتراکی در مسئله MSE است. در نتیجه برای الگوریتم تفاوتی نمی‌کند که از یال‌های اشتراکی قبلی استفاده کند یا این که یک یال اشتراکی جدید به راه‌حل اضافه کند.

اولین کار ابتکاری که انجام می‌دهیم بدین صورت است که الگوریتم تقریبی را مجبور می‌کنیم تا یال‌هایی که قبلا در راه‌حل استفاده شده‌اند را به کارگیرد. این روش ابتکاری را **به‌هنگام‌سازی پی در پی** [۱] می‌نامیم، این روش در الگوریتم ۲٫۱ آمده است. برای این که الگوریتم MCF را به استفاده دوباره از یال‌های قبلی ترغیب کنیم، در هر تکرار یالی که بیشترین جریان از آن عبور می‌کند و هزینه ناصفر دارد را انتخاب و هزینه آن یال را صفر می‌کنیم و سپس دوباره الگوریتم MCF را با هزینه‌های جدید اجرا می‌کنیم. به‌هنگام‌سازی هزینه‌ها هیچ تاثیری بر عامل تقریب الگوریتم ندارد، چون اولین مرتبه‌ای که جریان با حداقل هزینه را به‌دست‌می‌آوریم یک جواب $(k-1)$-تقریبی به‌دست‌آورده‌ایم و در تکرارهای بعدی سعی می‌کنیم جواب تقریبی را بهبود دهیم.

الگوریتم ۲٫۱: الگوریتم به‌هنگام‌سازی پی در پی

ورودی: گراف جهت‌دار G و عدد صحیح k
خروجی: تعداد یال‌های اشتراکی مورد نیاز برای یافتن k مسیر از s به t

با استفاده از قضیه ۲-۴-۱، گراف G' را از روی گراف G بساز
گراف G_0 را از روی گراف G'، با تغییر هزینه هر یال e به $c(e)/u(e)$ بساز
حداقل $s-t$-جریان f با اندازه k را در G_0 محاسبه کن
$i = 0$
تا زمانی که $f \neq 0$
  یال e در G_0 را به گونه‌ای پیدا کن که بیشترین جریان از آن عبور می‌کند
  گراف G_{i+1} را از روی گراف G_i، با تغییر هزینه یال e به صفر بساز
  حداقل $s-t$-جریان f با اندازه k را در G_{i+1} محاسبه کن
  $i = i + 1$
i **را برگردان**

۲-۶-۲ روش کوتاه‌ترین مسیر

بنا بر قضیه ۲-۵-۲ تعداد یال‌های کوتاه‌ترین $s-t$-مسیر یک کران بالا برای تعداد یال‌های اشتراکی در مسئله MSE است. فرض کنید ℓ اندازه کوتاه‌ترین $s-t$-مسیر باشد. اگر یک جواب‌شدنی مسئله MSE برای k تا $s-t$-مسیر بیش از ℓ یال‌اشتراکی داشته باشد، ما می‌توانیم k مسیر را با کوتاه‌ترین مسیر جایگزین کنیم و تعداد یال‌های اشتراکی را به ℓ کاهش دهیم. در الگوریتم ۲٫۲ از این روش ابتکاری استفاده کرده‌ایم.

[۱] Successive update

الگوریتم ۲,۲: الگوریتم کوتاه‌ترین مسیر

ورودی: گراف جهت‌دار G و عدد صحیح k

خروجی: تعداد یال‌های اشتراکی مورد نیاز برای یافتن k مسیر از s به t

فرض کنید p خروجی الگوریتم ۲,۱ با اجرا روی G باشد

فرض کنید ℓ طول کوتاه‌ترین $s-t$-مسیر در G باشد

$\min\{\ell, p\}$ **را برگردان**

۲-۶-۳ روش‌های تصادفی

در خط ۶، از الگوریتم ۲,۱ به صورت حریصانه[1] عمل می‌کنیم و هر بار هزینه یالی که بیش‌ترین جریان از آن عبور کرده است را صفر می‌کنیم. الگوریتم حریصانه با انجام یک سری انتخاب، که هر یک در لحظه‌ای خاص، بهترین به نظر می‌رسد عمل می‌کند، یعنی انتخاب در جای خود بهینه است. امید این است که یک حل بهینه سراسری یافت شود، ولی همواره چنین نیست. در ادامه با چند روش تصادفی یک یال را انتخاب و هزینه آن را صفر می‌کنیم. به سه روش یال‌ها را تصادفی انتخاب می‌کنیم:

تصادفی ساده: در این روش، خط ۶ الگوریتم ۲,۱ را بدین صورت تغییر می‌دهیم: هر بار یال e به صورت تصادفی از بین یال‌های G_i که جریان مثبت دارند انتخاب می‌شود و هزینه آن را صفر می‌کنیم.

تصادفی وزن‌دار: در این روش به هر یال e، یک وزن $w(e)$ اختصاص می‌دهیم که $w(e)$ برابر است با تعداد مرتبه‌هایی که یال e در $s-t$-مسیرها ظاهر شده است (به‌عبارت‌دیگر $w(e)$ برابر است با اندازه $f(e)$ در G_i) و فرض کنید W مجموع وزن همه یال‌ها باشد. بر خلاف روش قبل، در هر تکرار الگوریتم ما یال e را به صورت تصادفی با احتمال $w(e)/W$ انتخاب می‌کند، در نتیجه احتمال انتخاب یال‌هایی که بیشتر در $s-t$-مسیرها ظاهر شده‌اند افزایش می‌یابد.

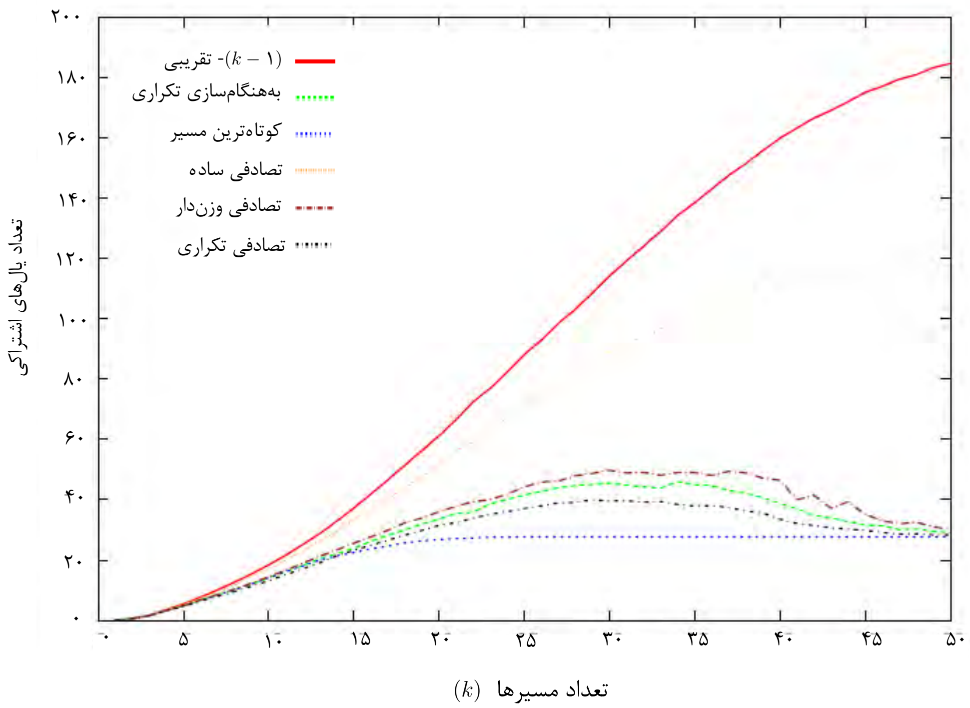
تصادفی پی در پی : در این روش، روش تصادفی وزن‌دار را چندین بار اجرا می‌کنیم، چون تعداد یال‌های اشتراکی در هر اجرا با هم یکسان نیست، جواب اجرایی را گزارش می‌کنیم که تعداد یال‌های اشتراکی آن از همه کمتر باشد.

[1] Greedy

۲۰۲۰۰

۲-۷ پیاده‌سازی الگوریتم‌ها

الگوریتم $(k-۱)$-تقریبی و ۵ الگوریتم ابتکاری بیان شده، در [۲۱] پیاده‌سازی شده و عملکرد الگوریتم‌ها بر



شکل ۲-۹: نتایج تجربی از اجرای الگوریتم‌های ابتکاری بر روی گراف شهر رم.

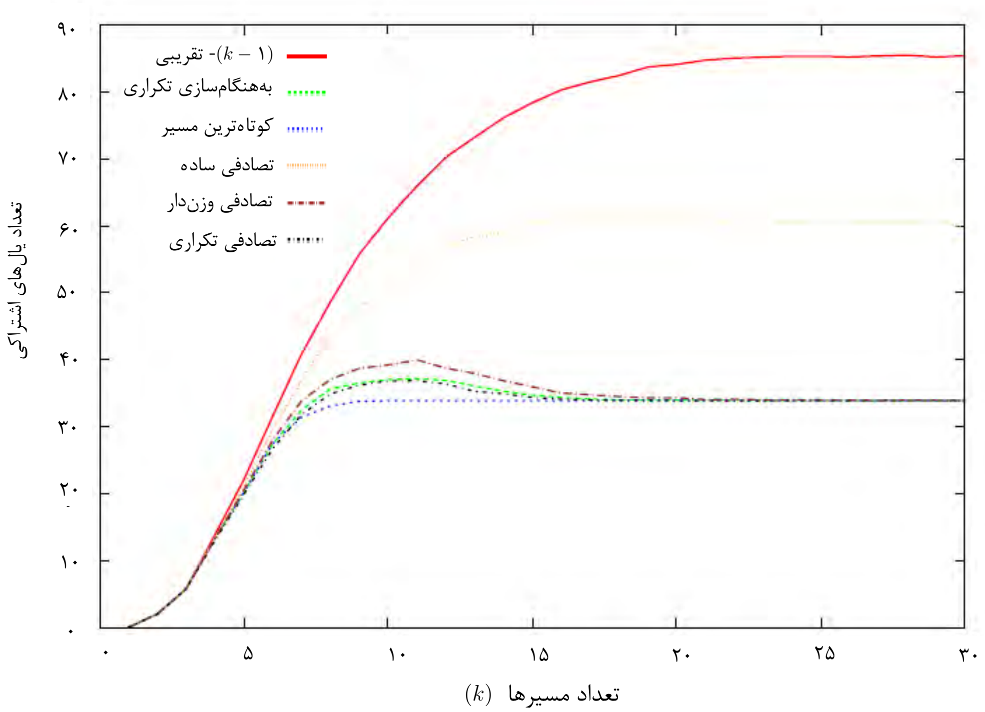
روی دو گراف مشابه بررسی شده است. شکل ۲-۹ و شکل ۲-۱۰، خلاصه‌ای از نتایج اجرای برنامه بر روی دو گراف است. شکل ۲-۹ مربوط به شبکه راه‌های شهر رم[۱] و شکل ۲-۱۰ مربوط به یک گراف تصادفی تولید شده توسط نظریه گراف‌های آزمایشی[۲] DARPA HPCS SSCA#2 است. گراف SSCA#2 نماینده محاسبات در زمینه امنیت ملی، محاسبات علمی و محاسبات زیست‌شناسی است. هر دو گراف دارای ۳۳۵۰ راس و ۸۸۷۰ یال هستند. الگوریتم‌ها برای k از ۱ تا ۵۰ اجرا شده است. برای هر k، ۱۰۰ جفت راس به صورت تصادفی انتخاب شده‌اند و میانگین تعداد یال‌های اشتراکی گزارش شده است. برای اطمینان از این که مبدا و مقصد تصادفی به اندازه کافی از هم دور هستند، جفت‌هایی انتخاب شده‌اند که اندازه کوتاه‌ترین مسیر

[۱] Rome
[۲] Graph theory benchmark

بین مبدا و مقصد بیشتر از $\sqrt{n}/۴$ بوده است که n تعداد راس‌هاست. برای اجرا از کلاستر $Boewulf$ از آزمایشگاه مجازی محاسبات با کارآیی بالا استفاده شده است، که دارای ۶۴ کلاستر، هر کلاستر دارای سرعت ۲,۲*۴ گیگا هرتز و ۸ گیگا بایت حافظه اصلی می‌باشد.

همان‌طور که در شکل ۲-۹ و شکل ۲-۱۰ مشاهده می‌کنید، عملکرد روش‌های ابتکاری به طور قابل توجهی بهتر از الگوریتم $(k-1)$-تقریبی اولیه است (چون که مقایسه عملکردها واضح است، داده‌ها برای k های کوچک ارائه شده است). وقتی مقدار k به اندازه کافی بزرگ باشد، از ٪۵۰ تا ٪۸۵ تعداد یال‌های اشتراکی در گراف با الگوریتم ابتکاری بهبود می‌یابد. وقتی k در یک بازه خاصی باشد الگوریتم کوتاه‌ترین مسیر بهتر از الگوریتم به‌هنگام‌سازی پی در پی عمل می‌کند. هر چند وقتی k به اندازه کافی بزرگ شود در نهایت جواب‌ها به یکدیگر همگرا می‌شوند. همگرایی به این دلیل است که وقتی تعداد مسیرها (k) بزرگ شود، احتمال این که یال‌های کوتاه‌ترین مسیر به عنوان یال‌اشتراکی در مسیرها انتخاب شوند، افزایش می‌یابد.

در این فصل مسئله حداقل یال‌اشتراکی معرفی و بررسی شد و نشان داده شد که این مسئله، یک مسئله NP-سخت می‌باشد و دو الگوریتم تقریبی با عامل‌های k و $k-۱$ برای مسئله ارائه شد. در ادامه فصل چند روش ابتکاری را با استفاده از الگوریتم $(k-1)$-تقریبی به کار بردیم و نتایج به‌دست‌آمده از پیاده‌سازی روش‌ها با هم مقایسه شد. در فصل بعد چند الگوریتم تقریبی با عامل بهتر برای مسئله ارائه می‌شود.



شکل ۲-۱۰: نتایج تجربی از اجرای الگوریتم‌های ابتکاری بر روی گراف آزمایشی $SSCA$.

فصل ۳

مسئله حداقل آسیب‌پذیری

در این فصل مسئله حداقل یال‌اشتراکی به مسئله حداقل آسیب‌پذیری تعمیم داده می‌شود و در ادامه فصل یک الگوریتم تقریبی برای مسئله حداقل آسیب‌پذیری و یک الگوریتم تقریبی با عامل زیرخطی برای مسئله حداقل یال‌اشتراکی ارائه می‌شود. علاوه‌براین نشان داده می‌شود که اگر k ثابت باشد، برای مسئله حداقل یال‌اشتراکی راه‌حل چندجمله‌ای وجود دارد.

۳-۱ تعریف مسئله حداقل آسیب‌پذیری

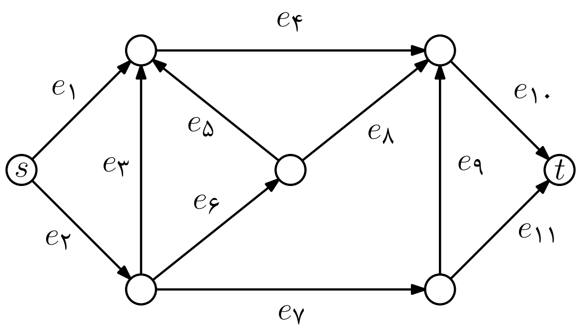
در این بخش مسئله حداقل r-آسیب‌پذیری را معرفی و بررسی می‌کنیم، که تعمیمی از مسئله حداقل یال اشتراکی (MSE) است. به طور کلی مسئله MSE را در سه بخش تعمیم داده‌اند که این منجر به تعریف مسئله جدید r-آسیب‌پذیری می‌شود و آن را به کاربردهای عملی نزدیک‌تر می‌کند. در مسئله MSE هزینه یال‌های اشتراکی با هم برابر است، در حالی که در عمل هزینه هر یال متفاوت است. بنابراین در وهله اول به هر یال هزینه c_e اختصاص می‌دهیم که نشان دهنده هزینه یال اشتراکی e می‌باشد. از طرف دیگر در مسئله MSE هر یال می‌تواند حداکثر در k تا مسیر ظاهر شود، اما در بخش دوم تعمیم برای هر یال e، یک کران‌بالای U_e در نظر می‌گیریم که نشان می‌دهد، یال e حداکثر می‌تواند در U_e مسیر استفاده شود. در بخش سوم تعمیم، به مسئله MSE یک پارامتر دیگر به نام $0 \leq r < k$ اضافه می‌کنیم، به این صورت که اگر یال e بیش از r بار در مسیرها مورد استفاده قرار گرفت، آنگاه باید برای یال e هزینه c_e پرداخت شود. به عبارت دیگر برای یال‌هایی که کمتر از r بار در مسیرها ظاهر شده‌اند هزینه‌ای پرداخت نمی‌شود. با توجه به مطالب فوق و تعمیم مسئله MSE، مسئله r-آسیب‌پذیری را به صورت زیر تعریف می‌کنیم. یک یال را **r-آسیب‌پذیر** می‌خوانیم هرگاه بیش از rبار در k مسیر استفاده شده باشد.

مسئله ۳-۱-۱. *مسئله حداقل آسیب‌پذیری[1]: گراف جهت‌دار $G = (V, E)$ را در نظر بگیرید که به هر یال $e \in E$ یک هزینه نامنفی c_e و ظرفیت U_e اختصاص داده شده است. همچنین دو راس مشخص $s, t \in V$ و دو عدد صحیح r و k به طوری‌که $0 \leq r < k$ باشد را در نظر بگیرید. k مسیر از s به t طوری پیدا کنید که مجموع هزینه یال‌های r-آسیب‌پذیر حداقل شود [۲۰].*

[1] The miminum vulerability problem

در صورتی که به‌ازای همه یال‌ها $c_e = 1$ و $U_e = k$ باشد، آنگاه مسئله ۱-آسیب‌پذیری به مسئله MSE تبدیل می‌شود. علاوه‌براین، مسئله ۰-آسیب‌پذیری برابر با مسئله جریان با یال-هزینه کمینه ($MECF$) است. یک کاهش دیگر که برای مسئله حداقل آسیب‌پذیری می‌توانیم در نظر بگیریم به این صورت است که k تا مسیر به گونه‌ای پیدا کنید که هیچ یال r-آسیب‌پذیری ($r > 0$) در جواب وجود نداشته باشد، که این برابر است با مسئله شناخته شده مسیرهای مجزا و می‌توانیم آن را با استفاده از الگوریتم جریان بیشینه در زمان چندجمله‌ای حل کنیم [۸].

به عنوان مثال گراف شکل ۳-۱ را در نظر بگیرید. مسئله حداقل آسیب‌پذیری با $r = 1$ و $k = 3$، سه مسیر از s به t به‌دست‌می‌آورد که تعداد یال‌های ۱-آسیب‌پذیری آن برابر ۲ است و مسیرها شامل π_1, π_2 و π_3 می‌باشند. در این مجموعه فقط یال‌های e_2 و e_{10} بیش از یک‌بار در مسیرها ظاهر شده‌اند یا به عبارت دیگر یال‌های e_2 و e_{10}، ۱-آسیب‌پذیر هستند. علاوه‌براین برای $k = 3$ و $r = 2$ می‌توان سه $s - t$-مسیر به‌دست‌آورد که هیچ یال ۲-آسیب‌پذیری ندارد، در حالی که اگر $k = 5$ و $r = 2$ باشد، می‌توان پنج مسیر $\pi_1, \pi_1, \pi_2, \pi_2$ و π_3 از s به t به‌دست‌آورد که دو یال ۲-آسیب‌پذیر e_2 و e_7 در مجموعه مسیرها وجود دارد.



$$\pi_1 = \langle e_1, e_4, e_{10} \rangle$$
$$\pi_2 = \langle e_2, e_7, e_{11} \rangle$$
$$\pi_3 = \langle e_2, e_6, e_8, e_{10} \rangle$$
$$\pi_4 = \langle e_2, e_3, e_4, e_{10} \rangle$$
$$\pi_5 = \langle e_2, e_7, e_9, e_{10} \rangle$$
$$\pi_6 = \langle e_2, e_6, e_5, e_4, e_{10} \rangle$$

شکل ۳-۱: شش $s - t$-مسیر در گراف G، که با π_1 تا π_6 مشخص شده‌اند.

۳-۲ الگوریتم اولیه-دوگان مسئله آسیب‌پذیری

در مسئله r-آسیب‌پذیری، هدف پیدا کردن k تا مسیر از راس s به راس t در گراف جهت‌دار G است، به طوری که مجموع هزینه یال‌هایی که بیش از r بار در مسیرها استفاده شده‌اند حداقل شود. هزینه هر یال r-آسیب‌پذیر

در گراف c_e می‌باشد و از یال e نباید بیش از U_e مسیر عبور کند. در این بخش یک الگوریتم تقریبی را برای مسئله r-آسیب‌پذیری بیان می‌کنیم که دارای عامل تقریب $\lfloor \frac{k}{r+1} \rfloor$ می‌باشد. به طور خلاصه الگوریتم بدین صورت عمل می‌کند، ابتدا ظرفیت همه یال‌ها را r قرار می‌دهد و سپس با افزایش ظرفیت بعضی از یال‌ها از r به U_e، تمام $s-t$-برش‌های با اندازه کمتر از k در گراف را از بین می‌برد تا این که اندازه برش‌کمینه به حداقل k برسد. سپس یک جریان f با اندازه k از s به t از روی گراف به دست می‌آورد و در پایان مسیرها را از روی جریان f می‌سازد.

فرض کنید مجموعه S تمام $s-t$-برش‌هایی در گراف G است که اندازه آن‌ها از $\lceil \frac{k}{r} \rceil$ کمتر است. در یک حالت خاص اگر $r = ۰$ بود، مجموعه تمام $s-t$-برش‌ها در G را به عنوان مجموعه S تعریف می‌کنیم. همان‌طور که در تعریف ۱-۲-۳ آمده، یک $s-t$-برش برابر است با افراز مجموعه راس‌های گراف G به دو مجموعه V' و $V-V'$ به طوری که برای راس‌های s و t داشته باشیم $s \in V'$ و $t \in V - V'$، جمع هزینه یال‌های عبوری از V' به $V-V'$ را هزینه $s-t$-برش می‌گویند.

با در نظر گرفتن تعریف مجموعه S، مشاهده می‌شود که به ازای هر جواب‌شدنی در مسئله r-آسیب‌پذیر، هر برش $C \in S$ باید حداقل یک یال r-آسیب‌پذیر در مجموعه یال‌هایش وجود داشته باشد. منظور از جواب‌شدنی پیدا کردن k مسیر از s به t در G می‌باشد. در صورتی که فرض خلف را در نظر بگیریم، یعنی به ازای حداقل یک برش $C \in S$، هیچ یال r-آسیب‌پذیری نداشته باشیم، آنگاه حداکثر $k > r \times (\lceil \frac{k}{r} \rceil - ۱)$ مسیر می‌تواند از C عبور کند زیرا اندازه برش‌کمینه حداکثر برابر با این مقدار می‌شود، این نتیجه می‌دهد که این جواب، جواب‌شدنی نیست، در حالی که یک جواب‌شدنی برای مسئله در نظر گرفته بودیم یا به‌عبارت‌دیگر فرض کرده بودیم در G، k تا مسیر از s به t وجود دارد.

برای بیان الگوریتم تقریبی ابتدا یک مدل برنامه‌ریزی خطی متناظر با مسئله را به‌دست‌می‌آوریم. در گراف G به ازای هر یال e یک متغیر بولی x_e را در نظر می‌گیریم، در صورتی که یال e بیش از r بار در k مسیر ظاهر شده باشد (به‌عبارت‌دیگر یال e آسیب‌پذیر باشد)، $x_e = ۱$، و درغیراین‌صورت $x_e = ۰$ در نظر می‌گیریم. هزینه و ظرفیت هر یال را با c_e و U_e نمایش می‌دهیم. در صورتی که فرض کنیم یال‌ها بدون‌کران هستند (به ازای هر یال $U_e = \infty$ باشد)، می‌توان آن را به صورت یک برنامه‌ریزی خطی عدد صحیح زیر بیان

نمود:

$$\begin{aligned} \min \quad & \sum_{e\in E} c_e x_e \\ s.t. \quad & \sum_{e\in C} x_e \geq ۱ \quad \forall C \in S \\ & x_e \in \{۰, ۱\} \quad \forall e \in E \end{aligned}$$

در برنامه‌ریزی عدد صحیح فوق هدف کاهش تعداد یال‌های آسیب‌پذیر است و هر قید متناظر با یک برش $C \in S$ می‌باشد. عبارت $\sum_{e\in C} x_e \geq ۱$ بدین معنی است که در برش C حداقل یک یال آسیب‌پذیر باید وجود داشته باشد. با تعویض محدودیت‌های $x_e \in \{۰,۱\}$ با $x_e \geq ۰$ می‌توان برنامه‌ریزی عدد صحیح را به برنامه‌ریزی خطی تبدیل نمود. مدل برنامه‌ریزی خطی مسئله در زیر نمایش داده شده است:

$$\begin{aligned} \min \quad & \sum_{e\in E} c_e x_e \\ s.t. \quad & \sum_{e\in C} x_e \geq ۱ \quad \forall C \in S \\ & x_e \geq ۰ \qquad \forall e \in E \end{aligned}$$

برای طراحی الگوریتم لازم است دوگان مدل برنامه‌ریزی خطی فوق را به‌دست‌آوریم. فرض کنید y_Cها متغیرهای تصمیم در برنامه‌ریزی خطی دوگان باشند، برای محاسبه $\sum_{e\in C} y_C$ بدین صورت عمل می‌کنیم که برای یال مشخص e، نگاه می‌کنیم که e در چه برش‌هایی استفاده شده است، سپس y_C متناظر با برش‌ها را در سری جمع می‌زنیم. فرض کنید یال e در برش‌های $C_۱, C_۲, \ldots, C_t$ ظاهر شده باشد، آنگاه $\sum_{e\in C} y_C = \sum_{i=۱}^{t} y_{C_i}$.

$$\begin{aligned} \max \quad & \sum_{C\in S} y_C \\ s.t. \quad & \sum_{e\in C} y_C \leq c_e \quad \forall e \in E \\ & y_C \geq ۰ \qquad \forall C \in S \end{aligned}$$

الگوریتم تقریبی که با استفاده از دوگان برنامه‌ریزی خطی به‌دست‌آمده، در الگوریتم ۳٫۱ نمایش داده شده است. الگوریتم بدین صورت عمل می‌کند که با یک جواب‌شدنی دوگان $y = ۰$ و یک جواب‌نشدنی مسئله اولیه $R = \emptyset$ شروع می‌کند. سپس در تکرارهای متوالی جواب‌شدنی را بهبود می‌دهد. مجموعه یال‌هایی که در طول الگوریتم به عنوان یال آسیب‌پذیر شناخته می‌شوند در مجموعه R نگهداری می‌شوند. در ابتدا ظرفیت برای همه یال‌ها $u_e = r$ قرار می‌دهیم، به‌عبارت‌دیگر مشخص می‌کنیم حداکثر r مسیر می‌تواند از یال e استفاده کند. الگوریتم با استفاده از یک حلقه تکرار هر بار جواب‌شدنی را بهبود می‌دهد، بدین صورت که در هر تکرار یک $s-t$-برش C با اندازه کمتر از k را به‌دست‌می‌آورد و مقدار متغیرهای متناظر با محدودیت y_C را آنقدر افزایش می‌دهد تا زمانی که $\sum_{e\in C} y_C \leq c_e$، برای بعضی از یال‌ها مثل e به بالاترین حد خود برسد. سپس

یال e را به مجموعه یال‌های آسیب‌پذیر R اضافه می‌کنیم و ظرفیت یال e را برابر با U_e (ظرفیت اولیه e) قرار می‌دهیم.

الگوریتم ۳٫۱: الگوریتم اولیه–دوگان

ورودی: گراف جهت‌دار G و عدد صحیح k

خروجی: جریان f با اندازه k از راس s به راس t

۱ $y = ۰$

۲ $R = \emptyset$

۳ **برای هر** $e \in E$

۴ $\quad u_e = r$

۵ **تا زمانی که** یک $s-t$–برش C *با اندازه کمتر از* k *در* G *وجود دارد*

۶ $\quad$ مقدار y_C را افزایش بده تا برای یال e، $\sum_{e\in C} y_C = c_e$ شود

۷ $\quad R = R \cup \{e\}$

۸ $\quad u_e = U_e$

۹ یک $s-t$–جریان f با اندازه k در گراف G پیدا کن

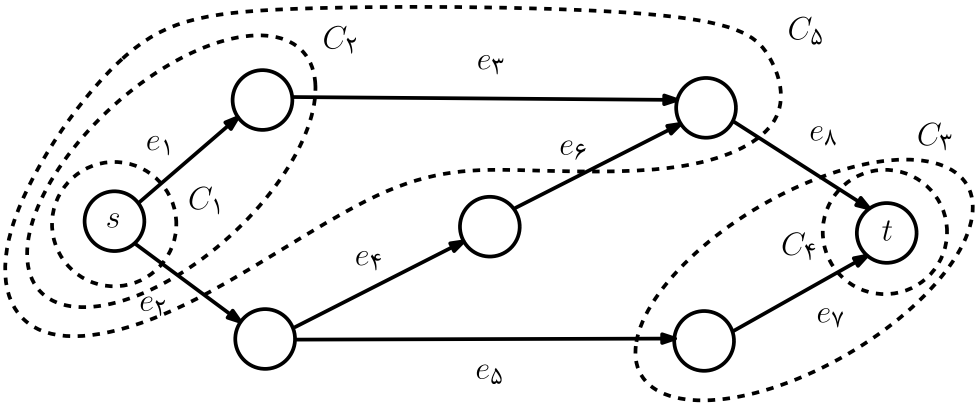
۱۰ f **را برگردان**

زمانی که حلقه تکرار خاتمه پیدا می‌کند، همه $s-t$–برش‌های با ظرفیت کوچک‌تر از k از گراف حذف شده‌اند. بنابراین اندازه برش‌کمینه در G حداقل k می‌شود و سپس در خط ۹ الگوریتم ۳٫۱، یک جریان f با اندازه k از روی گراف G به‌دست‌می‌آورد. طبق قضیه جریان بیشینه–برش‌کمینه (قضیه ۱–۲–۵)، چون اندازه برش‌کمینه در G حداقل k است، بنابراین اندازه جریان بیشینه در G نیز حداقل k می‌باشد. در نتیجه می‌توان جریان f با مقدار k را در گراف G به‌دست‌آورد. در پایان براساس قضیه تجزیه جریان به مسیر می‌توان از روی جریان f، k مسیر از s به t استخراج کرد.

به عنوان مثال فرض کنید می‌خواهیم مسئله r–آسیب‌پذیری را برای گراف شکل ۳–۲ با $r = ۱$ و $k = ۳$ حل کنیم. هزینه و ظرفیت هر یال گراف را به ترتیب $c_e = ۱$ و $U_e = \infty$ در نظر بگیرید. برای حل مسئله ابتدا باید تمام $s-t$–برش‌هایی که کوچک‌تر از $۳ = \lceil \frac{k}{r} \rceil$ است را به‌دست‌آوریم. گراف G شامل ۵، $s-t$–برش با اندازه کمتر از ۳ است که در شکل ۳–۲ نمایش داده شده است. با استفاده از این برش‌ها مجموعه S را تشکیل می‌دهیم، بنابراین مجموعه S به صورت زیر به‌دست می‌آید:

$$C_{۱} = \{e_{۱}, e_{۲}\}, C_{۲} = \{e_{۲}, e_{۳}\}, C_{۳} = \{e_{۸}, e_{۵}\}, C_{۴} = \{e_{۸}, e_{۷}\}, C_{۵} = \{e_{۸}, e_{۲}\}$$
$$S = \{C_{۱}, C_{۲}, C_{۳}, C_{۴}, C_{۵}\}$$

به‌ازای‌هر یال در گراف متغیر بولی x_e را تعریف می‌کنیم. در صورتی که یال e آسیب‌پذیر باشد $x_e = ۱$ و در صورتی که یال e آسیب‌پذیر نباشد $x_e = ۰$ است. مسئله برنامه‌ریزی خطی متناسب با مسئله آسیب‌پذیری



شکل ۳-۲: در گراف G برش‌های با اندازه کمتر از $k = ۳$ مشخص شده.

فوق در گراف G را به صورت زیر تشکیل می‌دهیم.

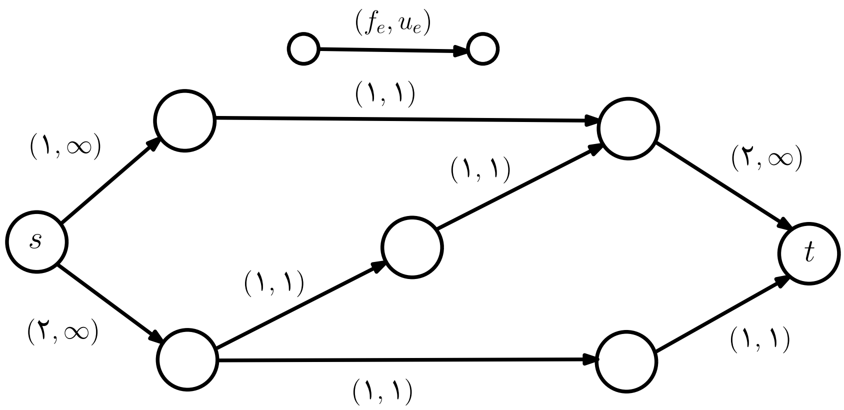
$$
\begin{aligned}
Z = \quad & \min \quad \sum_{i=۱}^{۸} c_i x_i \\
& s.t. \quad x_۱ + x_۲ \geq ۱ \\
& \qquad\quad x_۲ + x_۳ \geq ۱ \\
& \qquad\quad x_۲ + x_۸ \geq ۱ \\
& \qquad\quad x_۷ + x_۸ \geq ۱ \\
& \qquad\quad x_۸ + x_۵ \geq ۱ \\
& \qquad\quad x_e \geq ۰ \qquad \forall e \in E
\end{aligned}
$$

با حل کردن برنامه‌ریزی خطی فوق، برای تابع هدف مقدار $Z = ۲$ به‌دست می‌آید. این نتایج بدین معنی است که جواب بهینه مسئله آسیب‌پذیری در گراف G برای $k = ۳$ و $r = ۱$ برابر ۲ است، یا به‌عبارت‌دیگر برای پیدا کردن $k = ۳$ مسیر از s به t در گراف G، حداقل به ۲ یال آسیب‌پذیر نیاز است. در مسئله آسیب‌پذیری ممکن است چندین جواب بهینه وجود داشته باشد و با حل برنامه‌ریزی خطی حداقل یکی از جواب‌ها را به‌دست‌می‌آوریم. در گراف G، برای $k = ۳$ و $r = ۱$ هر جواب بهینه به دو یال آسیب‌پذیر نیاز دارد. در ادامه دوگان برنامه‌ریزی خطی فوق را به‌دست‌می‌آوریم و سپس الگوریتم ۳,۱ را بر روی گراف G اجرا و جواب آن

را با جواب بهینه مقایسه می‌کنیم. دوگان برنامه‌ریزی خطی، مسئله شکل ۳-۲ به‌صورت‌زیر می‌باشد.

$$\begin{array}{lll} \max & \sum_{i=1}^{5} y_{C_i} & \\ s.t. & y_{C_1}+y_{C_2}+y_{C_5}\leq 1 & \\ & y_{C_3}+y_{C_4}+y_{C_5}\leq 1 & \\ & y_{C_1}\leq 1 & \\ & y_{C_2}\leq 1 & \\ & y_{C_3}\leq 1 & \\ & y_{C_4}\leq 1 & \\ & y_{C_i}\geq 0 & i=1,\dots,5 \end{array}$$

با اجرای الگوریتم ۳٫۱ بر روی گراف G با $k=3$ و $r=1$ در مرحله اول $y=0$ و $R=\emptyset$ قرار داده می‌شود ظرفیت هر یال e را برابر با r قرار می‌دهیم. سپس در حلقه تکرار هر بار یک برش‌کمینه که اندازه آن کوچک‌تر از $k=3$ است را پیدا می‌کنیم. گراف G، برش با اندازه ۱ ندارد و ۵ برش با اندازه ۲ دارد. فرض کنید در تکرار اول برش C_1 انتخاب شود، با اجرای خط ۶ الگوریتم برای یال e_1، معادله $\sum_{e_1\in C_1} y_{C_1}=c_{e_1}$ به‌دست‌می‌آید. بنابراین ظرفیت یال e_1 را $u_{e_1}=U_{e_1}$ قرار می‌دهیم و آن را به مجموعه R اضافه می‌کنیم. در ادامه فرض کنید برش‌های C_3 و C_5 انتخاب شوند، در نتیجه یال‌های e_2 و e_8 به مجموعه R اضافه می‌شوند و ظرفیت آن‌ها را به ترتیب U_2 و U_8 قرار می‌دهیم. با پیدا کردن برش‌های C_1، C_2 و C_5، اندازه برش‌کمینه درگراف به اندازه ∞ می‌رسد و این مقدار بیش‌تر از $k=3$ می‌باشد، بنابراین شرط حلقه تکرار دیگر برقرار نیست و حلقه پایان می‌یابد. درگام بعدی لازم است یک جریان با اندازه $k=3$ درگراف پیدا کنیم. برای این کار می‌توان جریان f را به صورت شکل ۳-۳ به‌دست‌آورد و سپس k مسیر را از روی جریان f پیدا کرد.



شکل ۳-۳: جریان f با اندازه ۳ درگراف.

الگوریتم ۳٫۱ در پایان یال‌هایی که در مجموعه $R = \{e_۱, e_۲, e_۸\}$ قرار دارند را به عنوان یال آسیب‌پذیر گزارش می‌کند. این یک جواب‌تقریبی از جواب بهینه می‌باشد. در جواب الگوریتم ۳٫۱، ۳ یال به عنوان یال آسیب‌پذیر انتخاب شده‌اند در حالی که تعداد یال‌های آسیب‌پذیر در جواب بهینه برابر ۲ می‌باشد. در جواب بهینه یال‌های $e_۲$ و $e_۸$ به عنوان یال آسیب‌پذیر انتخاب می‌شوند.

در ادامه قصد داریم که عامل تقریب الگوریتم را محاسبه کنیم، فرض کنید OPT هزینه جواب بهینه مسئله حداقل آسیب‌پذیری باشد و فرض کنید Z_{LP} جواب بهینه تابع هدف برنامه‌یزی خطی باشد. در مسئله بهینه‌سازی برای یال‌ها یک ظرفیت U_e در نظر گرفتیم، ولی در برنامه‌ریزی خطی این محدودیت در نظر گرفته نشده است، به‌عبارت‌دیگر $U_e = \infty$. بنابراین هر جواب‌شدنی برای مسئله بهینه‌سازی یک جواب‌شدنی برای برنامه‌ریزی خطی نیز می‌باشد، بنابراین $Z_{LP} \leq OPT$.

قضیه ۳-۲-۱. *[۲۰] اگر APX جواب به‌دست‌آمده توسط الگوریتم ۳٫۱ باشد، آنگاه $APX \leq \lfloor \frac{k}{r+۱} \rfloor OPT$.*

اثبات. فرض کنید APX جواب به‌دست‌آمده توسط الگوریتم ۳٫۱ باشد و T مجموعه یال‌هایی باشد که جریان با اندازه بیشتر از r از آن‌ها عبور می‌کند. همچنین همان‌طور که قبلا تعریف شده، R مجموعه یال‌هایی است که توسط الگوریتم به عنوان یال آسیب‌پذیر شناخته شده‌اند. واضح است که $T \subseteq R$ می‌باشد زیرا ممکن است بعضی از یال‌هایی که در R قرار دارند، جریان کمتر مساوی r از آن‌ها عبور کند و به عنوان یال آسیب‌پذیر در نظر گرفته نشوند.

$$\begin{aligned} APX &= \sum_{e \in T} c_e \\ &= \sum_{e \in T} \sum_{e \in C} y_C \\ &= \sum_{C \in S} (y_C \times |\{e \in T \cap C\}|) \\ &\leq \lfloor \tfrac{k}{r+۱} \rfloor \sum_{C \in S} y_C \\ &\leq \lfloor \tfrac{k}{r+۱} \rfloor Z_{LP} \end{aligned} \tag{۳-۱}$$

در اولین معادله ۳-۱، c_e را براساس خط ۶ الگوریتم ۳٫۱ با $\sum_{e \in C} y_C$ تعویض می‌کنیم، در دومین معادله ۳-۱ ترتیب شمارش را تغییر می‌دهیم و به جای محاسبه y_C برای یال‌هایی که در T قرار دارند؛ به ازای هر برش $C \in S$، y_C را برای یال‌هایی که در آن برش آسیب‌پذیر هستند ($e \in T \cap C$) محاسبه می‌کنیم. در مثال

قبل مجموعه $T = \{e_۲, e_۸\}$ به‌دست‌آمد و هر یال $e \in T$ به صورت زیر در برش‌ها ظاهر شده است:

$$\begin{array}{lllllll} e_۲ & : & C_۱ & , & C_۲ & , & C_۵ \\ e_۸ & : & C_۳ & , & C_۴ & , & C_۵ \end{array}$$

با توجه به اولین معادله ۳-۱، برای مثال مقدار APX به صورت زیر به‌دست می‌آید:

$$APX = y_{C_۱} + y_{C_۲} + y_{C_۳} + y_{C_۴} + ۲y_{C_۵}.$$

از طرفی اگر برای هر برش $C \in S$، مجموعه $\{e \in T \cap C\}$ به صورت زیر است:

$$\begin{array}{lll} C_۱ & : & \{e_۲\} \\ C_۲ & : & \{e_۲\} \\ C_۳ & : & \{e_۸\} \\ C_۴ & : & \{e_۸\} \\ C_۵ & : & \{e_۸, e_۲\} \end{array}$$

با توجه به دومین معادله ۳-۱، برای مثال مقدار APX به صورت زیر به‌دست‌می‌آید:

$$APX = y_{C_۱} + y_{C_۲} + y_{C_۳} + y_{C_۴} + ۲y_{C_۵}.$$

در سومین نامعادله ۳-۱، از این واقعیت استفاده می‌کنیم که هر برش C می‌تواند حداکثر $\lfloor \frac{k}{r+۱} \rfloor$ یال آسیب‌پذیر (یالی که جریان با اندازه بیشتر از r از آن عبور می‌کند) داشته باشد. الگوریتم در ابتدا از یک جواب‌شدنی $y = ۰$ شروع می‌کند و سپس آن را بهینه می‌کند، طبق قضیه دوگانگی ضعیف (قضیه ۱-۴-۱)، در برنامه‌ریزی خطی اگر یکی از مسائل اصلی یا دوگان دارای جواب بهینه باشند، آنگاه دیگری نیز دارای جواب بهینه است و جواب‌ها با هم برابرند. از طرف دیگر، قبلا نشان دادیم که $Z_{LP} \leq OPT$، بنابراین:

$$APX \leq \lfloor \frac{k}{r+۱} \rfloor OPT.$$

به‌عبارت‌دیگر الگوریتم ۳,۱ جواب‌تقریبی با فاکتور $\lfloor \frac{k}{r+۱} \rfloor$ برای مسئله آسیب‌پذیری به دست می‌آورد. $\square$

قضیه ۳-۲-۲. *زمان اجرای الگوریتم ۳,۱ ، بر روی گراف G از مرتبه $O(nm^۲ \log(n^۲m))$ است که n تعداد راس‌ها و m تعداد یال‌های G می‌باشد.*

اثبات. حلقه اصلی الگوریتم حداکثر به اندازه m بار تکرار می‌شود چون ممکن است به ازای هر یال یک $s-t$-برش داشته باشیم در هر تکرار لازم است یک $s-t$-برش‌کمینه برای G به‌دست‌آوریم. می‌توان برش‌کمینه

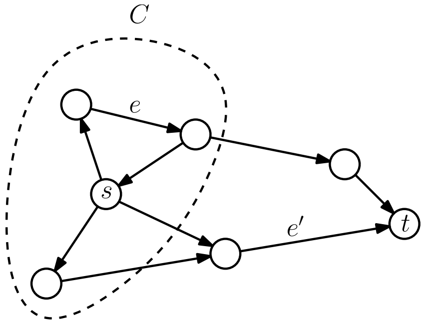
را در زمان $O(nm\log(n^۲m))$ محاسبه کرد [۹]. در خط ۶ الگوریتم ۳٫۱ حداکثر برای k یال در هر تکرار مقدار y_C باید محاسبه شود که این در زمان $O(k) = O(n)$ انجام می‌شود، بنابراین الگوریتم ۳٫۱ در زمان $O(m(nm\log(n^۲m))) = O(nm^۲\log(n^۲m))$ اجرا می‌شود. $\square$

۳-۳ الگوریتم دقیق برای مقادیر ثابت k

در این بخش نشان داده می‌شود در صورتی که مقدار k ثابت باشد، آنگاه مسئله آسیب‌پذیری برای هر $r > ۰$ در زمان چندجمله‌ای قابل حل می‌باشد و یک الگوریتم دقیق برای آن وجود دارد. با این وجود، در بخش ۳-۵ نشان داده می‌شود که مسئله آسیب‌پذیری NP-سخت می‌باشد و همچنین مسئله r-آسیب‌پذیری به‌ازای‌هر $\epsilon > ۰$ و به‌ازای‌هر $r > ۰$ تقریب $۲^{\log^{۱-\epsilon} n}$ را نمی‌پذیرد، مگر این‌که $NP \subseteq DTIME(n^{Polylog\, n})$. فرض کنید G یک گراف جهت‌دار و C یک $s-t$-برش در G باشد. بدون این‌که کلیت مسئله آسیب ببیند، فرض می‌کنیم که برای هر راس v، یک مسیر از s به v و یک مسیر از v به t وجود دارد چون درغیراین‌صورت می‌توانیم راس v را حذف کنیم زیرا در هیچ $s-t$-مسیری ظاهر نمی‌شود. $s-t$-برش C و یال $e = (u, v) \in C$ را در نظر بگیرید، اگر مسیری از s به u در گراف وجود داشته باشد به طوری که از هیچ کدام از یال‌های C استفاده نکند، می‌گوییم یال e قبل از C قرار دارد و اگر مسیر از v به t وجود داشته باشد به طوری که از هیچ یک از یال‌های C استفاده نکند، می‌گوییم یال e بعد از C قرار دارد (شکل ۳-۴ را ببینید). توجه کنید که یک یال نمی‌تواند هم زمان قبل و بعد از C قرار داشته باشد، زیرا C یک $s-t$-برش است و مجموعه راس‌ها را به دو دسته تقسیم می‌کند که یکی شامل s و دیگری شامل t می‌باشد. اگر یال e بخواهد هم زمان قبل و بعد از برش C باشد، این بدین معنی است که s و t در یک دسته قرار دارند، و این امکان‌پذیر نیست. دو $s-t$-برش $C_۱$ و $C_۲$ را در نظر بگیرید، در صورتی که هر یال $C_۱$، قبل از $C_۲$ باشد ویا جزء یال‌های $C_۲$ باشد، آنگاه می‌نویسیم $C_۱ \preccurlyeq C_۲$. برای نمایش رابطه برش‌ها از عملگر $\preccurlyeq$ استفاده می‌کنیم. در ادامه بعضی از خواص این رابطه بیان می‌شود.

قضیه ۳-۳-۱. *اگر برای دو $s-t$-برش $C_۱$ و $C_۲$، داشته باشیم $C_۱ \preccurlyeq C_۲$ و $C_۲ \preccurlyeq C_۱$ آنگاه $C_۱ = C_۲$.*

اثبات. فرض کنید یال $e \in C_۱$ در $C_۲$ قرار نداشته باشد، آنگاه یال e نسبت به برش $C_۲$ به دو صورت می‌تواند



شکل ۳-۴: یال e قبل از برش C و یال e' بعد از برش C قرار دارد.

باشد. یال e یا قبل از $C_۲$ است ویا بعد از آن قرار دارد. در صورتی که e قبل از $C_۲$ باشد آنگاه با فرض $C_۲ \preccurlyeq C_۱$ در تناقض است. همچنین اگر e بعد از $C_۲$ باشد با فرض $C_۱ \preccurlyeq C_۲$ در تناقض است، بنابراین $e \in C_۲$ می‌باشد. برای اثبات اینکه هر یال $C_۲$ باید در $C_۱$ باشد می‌توان از روش مشابه استفاده کرد، بنابراین دو برش $C_۱$ و $C_۲$ با هم برابر هستند. □

قضیه ۳-۳-۲. *اگر برای سه* $s-t$*-برش داشته باشیم* $C_۱ \preccurlyeq C_۲$ *و* $C_۲ \preccurlyeq C_۳$ *آنگاه* $C_۱ \preccurlyeq C_۳$.

اثبات. بر اساس فرض $C_۱ \preccurlyeq C_۲$ هر یال $e \in C_۱$ قبل از $C_۲$ قرار دارد و یا جزء یال‌های $C_۲$ است. در حالت اول اگر e قبل از $C_۲$ باشد طبق فرض داریم $C_۲ \preccurlyeq C_۳$ بنابراین e قبل از $C_۳$ نیز قرار دارد. در حالت دوم چون $C_۲ \preccurlyeq C_۳$ است، بنابراین e یا در $C_۳$ قرار دارد و یا اینکه قبل از $C_۳$ می‌باشد. در نتیجه همه یال‌های $C_۱$ قبل از $C_۳$ قرار دارند و $C_۱ \preccurlyeq C_۳$. □

یک مسئله حداقل r-آسیب‌پذیری را در نظر بگیرید. تابع ظرفیت $\mathsf{u} : E \Rightarrow \mathbb{Z}$ را مطلوب می‌خوانیم، اگر یک $s-t$-جریان f با مقدار k وجود داشته باشد به طوری که به ازای همه یال‌های $e \in E$، $\mathsf{u}(e) \leq U_e$ و $f(e) \leq \mathsf{u}(e)$ باشد. یک تابع ظرفیت مطلوب را کمینه می‌نامیم در صورتی که اگر ظرفیت هر کدام از یال‌هایی که $\mathsf{u}(e) > r$ است را کاهش دهیم، تابع u نامطلوب می‌شود یا به‌عبارت‌دیگر نمی‌توان جریان f را به‌دست‌آورد. دقت کنید که فرض ما بر این است که مسئله اولیه با ظرفیت یال‌های U_e دارای جریان با مقدار حداقل k می‌باشد، درغیراین‌صورت مسئله جواب‌شدنی ندارد. در ادامه نشان می‌دهیم به‌ازای‌هر تابع ظرفیت کمینه u، دنباله‌ای از برش‌ها با ظرفیت k وجود دارند به طوری که هر دو برش C_i و C_j که $i < j$ است، از

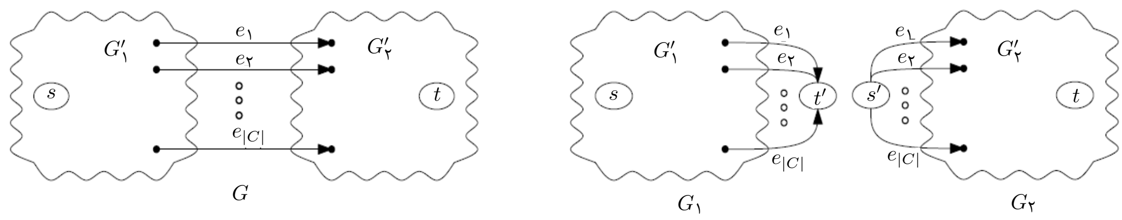
این دنباله انتخاب شوند $C_i \preccurlyeq C_j$ است و هر یال r-آسیب‌پذیر در حداقل یکی از این برش‌ها وجود دارد.

قضیه ۳-۳-۳. *[۲۰] اگر تابع ظرفیت کمینه* u *را در نظر بگیرید، آنگاه می‌توان یک دنباله* $C_1 \preccurlyeq \cdots \preccurlyeq C_\gamma$ *از* $s-t$*-برش‌ها را پیدا کرد، به طوری که به ازای* $1 \leq i \leq \gamma$*،* $\sum_{e \in C_i} \mathsf{u}(e) = k$ *و هر یال* $e \in E$ *با* $\mathsf{u}(e) > r$ *در حداقل یکی از برش‌ها قرار دارد.*

اثبات. طبق فرض به‌ازای‌هر یال $e \in E$ با $\mathsf{u}(e) > r$، باید یک $s-t$-برش C وجود داشته باشد به طوری که $e \in C$ و $\sum_{e' \in C} \mathsf{u}(e') = k$، چون u تابع ظرفیت مطلوب است. فرض کنید برش C_1 که از مجموعه یال‌هایی که از s خارج می‌شوند و برش C_2 از مجموعه یال‌هایی که به t وارد می‌شوند، به وجود آمده‌اند. اگر به ازای هر یال e که $\mathsf{u}(e) > r$ است، داشته باشیم $e \in C_1 \cup C_2$، آنگاه دنباله $C_1 \preccurlyeq C_2$ شرایط قضیه را دارا می‌باشد و توانسته‌ایم دنباله مورد نظر را پیدا کنیم.

حالتی را در نظر بگیرد که یال e با $\mathsf{u}(e) > r$ در مجموعه $C_1 \cup C_2$ قرار نداشته باشد. فرض کنید C یک برش از گراف G باشد که شامل e است و با توجه به تعریف تابع کمینه C دارای ظرفیت k می‌باشد. گراف G_1 را از روی گراف G، با حذف یال‌هایی که بعد از C قرار دارند می‌سازیم و سپس یک راس جدید t' به G_1 اضافه می‌کنیم و سر همه یال‌های برش C را به t' وصل می‌کنیم. به طور مشابه G_2 را از روی گراف G با حذف یال‌هایی که قبل از C قرار دارند می‌سازیم و سپس یک راس جدید s' به G_2 اضافه می‌کنیم و دم همه یال‌های برش C را به s' وصل می‌کنیم (شکل ۳-۵ را ببنید).

با این کار هر $s-t$-مسیر در G به دو مسیر شکسته می‌شود، یک مسیر از s به t' در G_1 و یک مسیر



شکل ۳-۵: گراف G با یک $s-t$-برش C به دو گراف G_1 و G_2 تقسیم می‌شود.

از s' به t در G_2. بنابراین مسئله به دو زیرمسئله در G_1 و G_2 تقسیم می‌شود. از آنجایی که C با برش‌های

C_1 و C_2 برابر نیست و از طرفی $|V(G_1)|$ و $|V(G_2)|$ اکیدا کوچک‌تر از $|V(G)|$ است، بنابراین این روند بازگشتی پایان‌پذیر می‌باشد. چون یال‌های برش C در هر دو گراف G_1 و G_2 استفاده شده است، بنابراین C در هر دو دنباله زیرمسئله‌های G_1 و G_2 قرار دارد. فرض کنید دنباله‌ی برش‌ها در G_1 و G_2 به ترتیب برابر با $C \preccurlyeq \cdots \preccurlyeq C_1$ و $C_\gamma \preccurlyeq \cdots \preccurlyeq C$ باشد، C در G_1 آخرین برش و در G_2 اولین برش دنباله است. با ترکیب دنباله‌ها می‌توان دنباله‌ای از برش‌ها به صورت $C_\gamma \preccurlyeq \cdots \preccurlyeq C \preccurlyeq \cdots \preccurlyeq C_1$ برای G به‌دست‌آورد. $\square$

قضیه ۳-۳-۴. *[۲۰] در صورتی که مقدار k ثابت باشد، می‌توان مسئله حداقل r-آسیب‌پذیری را در زمان چندجمله‌ای برای هر $r > 0$ حل نمود.*

اثبات. $s-t$-برش C را در نظر بگیرد، جفت (C,θ) را یک حالت[1] تعریف می‌کنیم که C یک $s-t$-برش با حداکثر k یال باشد و θ یک متغیر $|C|$-تایی به صورت زیر باشد:

$$\theta = (\theta_{e_1}, \theta_{e_2}, \ldots, \theta_{e_{|C|}})$$

به طوری که به‌ازای‌هر $e \in E$؛ $\theta_e \leq U_e$ و $\sum_{e \in E} \theta_e = k$ باشد.

مجموعه k مسیر از s به t را با $s-t$-جریان f نشان می‌دهیم. می‌گوییم جریان f برای C مطلوب است اگر برای همه یال‌های $e \in C$، $f(e) \leq \theta_e$ و به ازای همه یال‌های $e \in E \backslash C$، $f(e) \leq U_e$ باشد. به‌عبارت‌دیگر ما ظرفیت یال‌های برشی C را به مقادیر جدیدی که توسط θ مشخص می‌شود، کاهش داده‌ایم. قبلا فرض کرده بودیم که $\sum_{e \in C} \theta_e = k$، بنابراین برای همه یال‌های $e \in C$ مقدار θ_e با $f(e)$ برابر است، درغیراین صورت باید حداقل یک یال $e \in C$ با $f(e) \leq \theta_e$ وجود داشته باشد. از آنجایی که اندازه جریان f برابر با k و $\sum_{e \in C} \theta_e = k$ است، در نتیجه باید حداقل یال دیگری مثل $e' \in C$ با $f(e') > \theta_{e'}$ وجود داشته باشد و این امکان پذیر نیست. فرض کنید C و C' دو $s-t$-برش مجزا باشند به طوری که $C \preccurlyeq C'$ و متغیرهای چندتایی θ و θ' به ترتیب به برش‌های C و C' اختصاص داده شده است، به طوری که برای یال‌های $e \in C \cap C'$، $\theta_e = \theta'_e$ باشد. حالت (C', θ') را اکیدا بعد از (C,θ) می‌خوانیم اگر مجموعه k تا $s-t$-مسیر برای (C,θ) و (C',θ') وجود داشته باشد که برای هر دو آن‌ها مناسب باشد و هیچ یال r-آسیب‌پذیری بین C و C' (بعد از C و قبل از C') وجود نداشته باشد.

[1] State

برای چک کردن اینکه (C',θ') اکیدا بعد از (C,θ) قرار دارد، تابع ظرفیت $w(e)$ را به صورت زیر تعریف می‌کنیم:

$$w(e) = \begin{cases} \theta_e & e \in C \text{ اگر} \\ \theta'_e & e \in C' \text{ اگر} \\ \min(U_e, r) & \text{اگر } e \text{ بین } C' \text{ و } C \text{ قرار دارد} \\ U_e & \text{درغیراین‌صورت} \end{cases}$$

جریان بیشینه در گراف G با تابع ظرفیت w دقیقا k است، اگر و فقط اگر (C',θ') اکیدا بعد از (C,θ) قرار داشته باشد. حالت (C,θ) را در نظر بگیرید، مجموع هزینه یال‌های آسیب‌پذیر که در C یا بعد از C قرار دارند را به عنوان یک جواب برای حالت (C,θ) تعریف می‌کنیم. از آنجایی که ممکن است چندین جواب برای (C,θ) وجود داشته باشد، جوابی که مجموع هزینه یال‌های آسیب‌پذیر آن کمینه است، را به عنوان جواب (C,θ) در نظر می‌گیریم. زمانی که هیچ یال r-آسیب‌پذیری بعد از برش C قرار نداشته باشد، در این حالت جواب برای (C,θ) برابر با مجموع هزینه یال‌های r-آسیب‌پذیر برش C است، این حالت را یک حالت پایانی[1] می‌خوانیم.

الگوریتم ۳,۲: الگوریتم پیدا کردن هزینه (C,θ)

ورودی: حالت (C,θ)، که C یک $s-t$-برش ، θ یک متغیر $|\theta|$-تایی

خروجی: جواب دقیق برای حالت (C,θ)

$cost_C = \sum_{e\in C, \theta>r} c_e$

اگر (C,θ) یک *حالت پایانی است* **آنگاه**

  $cost_C$ **را برگردان**

$ans = \infty$

برای هر *حالت* (C',θ') *که اکیدا بعد از* (C,θ) *قرار دارد*

  با استفاده از الگوریتم ۳,۲ جواب (C',θ') را محاسبه کن و در $cost_{C'}$ قرار بده

  $ans = \min\{ans, cost_{C'} + cost_{C\setminus C'}\}$

ans **را برگردان**

الگوریتم ۳,۲، جواب حالت (C,θ) را محاسبه می‌کند و این کار به صورت بازگشتی و بر اساس حالت‌هایی که اکیدا بعد از (C,θ) قرار دارند انجام می‌شود. الگوریتم ابتدا همه حالت‌های (C',θ') که اکیدا بعد از (C,θ) قرار دارند را پیدا می‌کند و سپس مسئله را به صورت بازگشتی برای (C',θ') حل می‌کند و هزینه اضافه $cost_{C\setminus C'}$ را به آن اضافه می‌کند و سپس کم‌ترین مقدار را به عنوان جواب (C,θ) انتخاب می‌کند. $cost_{C\setminus C'}$ برابر با مجموع هزینه یال‌های آسیب‌پذیر C است که در C' قرار ندارند $(e \in C\setminus C')$، هزینه $cost_{C\setminus C'}$ برای هر برش C' متفاوت است و هر چه تعداد یال‌های مشابه در C و C' کمتر باشد، اندازه $cost_{C\setminus C'}$ افزایش

[1] Final state

می‌یابد.

وقتی (C, θ) یک حالت پایانی است جواب را می‌توان به راحتی همان‌طور که قبلا اشاره کردیم، محاسبه کرد. برای محاسبه جواب بهینه باید یک حالت اولیه (C, θ) تعریف کنیم، تا الگوریتم ۳٫۲ را ابتدا با این حالت فراخوانی کنیم. (C, θ) را یک حالت اولیه می‌خوانیم اگر یک جریان مطلوب برای (C, θ) وجود داشته باشد به طوری که هیچ یال r–آسیب‌پذیری در این مجموعه قبل از C قرار نداشته باشد. برای محاسبه جواب بهینه مسئله r–آسیب‌پذیری به‌ازای‌هر حالت اولیه (C, θ)، جواب متناظر با آن را محاسبه می‌کنیم و سپس از بین تمام جواب‌ها، جوابی که کم‌ترین مقدار را دارد به عنوان جواب بهینه گزارش می‌کنیم. الگوریتم ۳٫۳، این روند را نمایش می‌دهد.

الگوریتم ۳٫۳: الگوریتم پیدا کردن هزینه بهینه

ورودی: گراف جهت‌دار G و عدد ثابت k

خروجی: یک جواب دقیق برای مسئله حداقل آسیب‌پذیری

۱ $ans = \infty$

۲ **برای هر** *حالت اولیه* (C, θ)

۳ | با استفاده از الگوریتم ۳٫۲ مقدار (C, θ) را محاسبه کن و در $cost_C$ قرار بده

۴ | $ans = \min\{ans, cost_C\}$

۵ ans **را برگردان**

برای اثبات درستی الگوریتم، باید نشان دهیم که جواب بهینه در میان راه‌حل‌های به دست آمده توسط الگوریتم وجود دارد. برای یک جواب بهینه OPT، جریان f با اندازه k متناظر با جواب بهینه را در نظر بگیرید، تابع ظرفیت u را به شکل زیر تعریف می‌کنیم:

$$\mathsf{u}(e) = \max\{۱, f(e)\} \quad \forall e \in E.$$

واضح است که u یک تابع ظرفیت کمینه است، زیرا درغیراین‌صورت می‌توان ظرفیت یکی از یال‌هایی که $\mathsf{u}(e) > r$ است را کاهش داد و یک $s-t$–جریان f' با اندازه k و با تعداد کمتری یال آسیب‌پذیر به‌دست‌آورد، که این با بهینه بودن OPT در تناقض است.

با توجه به قضیه ۳-۳-۳ یک دنباله $C_۱ \preccurlyeq \cdots \preccurlyeq C_\gamma$ از $s-t$–برش‌ها وجود دارد به طوری که به‌ازای‌هر $۱ \leq i \leq \gamma$، $\sum_{e \in C_i} \mathsf{u}(e) = k$ و هر یال $e \in E$ با $\mathsf{u}(e) > r$ در حداقل یکی از γ برش قرار دارد. فرض کنید به‌ازای $۱ \leq i \leq \gamma$، $C_i \leq C_{i+۱}$ و $\theta = u$ باشد. از آنجایی که هر یال r–آسیب‌پذیر در

حداقل یکی از برش‌ها قرار دارد، بین C_i و $C_{i+۱}$ هیچ یال r–آسیب‌پذیری وجود ندارد و از طرفی θ برای C_i و $C_{i+۱}$ مطلوب است. بنابراین برای هر $۱ \leq i < \gamma$، حالت (C_i, θ) اکیدا بعد از حالت $(C_{i+۱}, \theta)$ قرار دارد و برای هر یال $e \in E$، $\theta_e = f(e)$ است. از آنجایی که هیچ یال r–آسیب‌پذیری قبل از $C_۱$ قرار ندارد، در نتیجه $(C_۱, \theta)$ یک حالت اولیه است و از طرفی هیچ یال r–آسیب‌پذیری بعد از C_γ قرار ندارد، در نتیجه (C_γ, θ) یک حالت پایانی است. تمام دنباله‌ی برش‌ها با تابع ظرفیت θ و حالت اولیه $(C_۱, \theta)$ توسط الگوریتم ۳٫۳ بررسی شده‌اند. بنابراین جواب برگردانده شده توسط الگوریتم ۳٫۳ بهینه است.

در ادامه زمان اجرای الگوریتم ۳٫۳ را به‌دست‌می‌آوریم؛ یک برش با اندازه حداکثر k یال، متناظر با انتخاب یک زیرمجموعه با اندازه حداکثر k، از مجموعه یال‌های E است. بنابراین تعداد برش‌هایی که اندازه آنها حداکثر k باشد برابر با $O(m^k)$ است که $m = |E|$. علاوه‌براین برای حالت (C, θ) کران‌بالای تعداد انتخاب‌های θ برای برش C، به طوری که (C, θ) یک حالت باشد، برابر است با تعداد مواردی که می‌توان به متغیرهای $\theta = (\theta_۱, \ldots, \theta_{|C|})$ یک عدد صحیح و نامنفی نسبت داد به طوری که $\sum_{e \in C} \theta_e = k$ باشد و تعداد انتخاب‌های θ برای C حداکثر از $O((k+۱)^k) = O(۱)$ می‌باشد. بنابراین تعداد کل حالت‌های (C, θ) از $O(m^k)$ است.

یک زیرمجموعه C شامل یال‌ها را در نظر بگیرید، ما می‌توانیم در زمان $O(n^۳)$ بررسی کنیم که C یک $s-t$–برش است یا خیر. همچنین می‌توانیم در زمان $O(k) = O(۱)$ بررسی کنیم که آیا θ برای C مطلوب است یا خیر، علاوه‌براین بررسی این که یک حالت، حالت اولیه (یا پایانی) است و بررسی این که یک حالت اکیدا بعد از حالت دیگر قرار دارد در زمان $O(n^۳)$ قابل انجام است. بنابراین برای هر حالت $O(n^۳)$ زمان لازم است.

برای پیاده‌سازی الگوریتم از برنامه‌ریزی پویا[۱] استفاده می‌شود، بدین صورت که جواب هر حالت را یک‌بار به‌دست‌می‌آید و درجایی ذخیره می‌شود تا نیاز به محاسبه چندباره آن نباشد. برای محاسبه جواب هر حالت اولیه (C, θ) با استفاده از الگوریتم ۳٫۳ به زمان $O(m^k n^۳)$ نیاز دارد. چون تعداد کل حالت‌ها از $O(m^k)$ است بنابراین زمان کل الگوریتم ۳٫۳ از $O(m^{۲k} n^۳)$ می‌باشد. در صورتی که فقط برش‌های با اندازه کمتر از k را انتخاب کنیم، زمان الگوریتم به $O(m^{۲(k-۱)} n^۳)$ کاهش می‌یابد. $\square$

[۱] Dynamic programming

۳-۴ الگوریتم تقریبی با عامل زیرخطی برای مسئله MSE

در بخش ۳-۲ یک الگوریتم $\lfloor \frac{k}{r+1} \rfloor$-تقریبی برای مسئله r-آسیب‌پذیری مطرح کردیم. از آنجایی که اگر $r = 1$ باشد، مسئله r-آسیب‌پذیری تبدیل به مسئله MSE می‌شود، بنابراین یک الگوریتم $\lfloor \frac{k}{2} \rfloor$-تقریبی نیز برای مسئله MSE به‌دست‌آمده است.

در مسئله MSE هیچ قیدی روی مسیرهای انتخابی نداریم، در نتیجه ممکن است یک مسیر دو یا چندبار در مجموعه مسیرها ظاهر شود. هر چه مقدار k بیشتر باشد، احتمال اینکه مسیرها شبیه به هم شوند افزایش می‌یابد، تا جایی که در قضیه ۲-۵-۲ ثابت شد اگر $k > |E|$ باشد، k تا $s-t$-مسیر راه‌حل، برابر با کوتاه‌ترین $s-t$-مسیر می‌شود. در الگوریتم ۳٫۴ ما از این ایده استفاده کردیم: ابتدا با استفاده از الگوریتم ۳٫۱ برای مسئله MSE یک جواب $\lfloor \frac{k}{2} \rfloor$-تقریبی به‌دست‌می‌آوریم و در صورتی که تعداد یال‌های اشتراکی خروجی الگوریتم بیشتر از تعداد یال‌های کوتاه‌ترین $s-t$-مسیر باشد، همه مسیرها را با کوتاه‌ترین $s-t$-مسیر جابجا می‌کنیم و نشان می‌دهیم که این ایده باعث می‌شود زمان اجرا الگوریتم ۳٫۴ زیرخطی شود.

الگوریتم ۳٫۴: الگوریتم تقریبی زیرخطی

ورودی: گراف جهت‌دار G و عدد صحیح k

خروجی: یک جواب تقریبی برای مسئله حداقل یال‌اشتراکی

فرض کنید P_1 خروجی الگوریتم ۳٫۱ با w یال‌اشتراکی

فرض کنید P_2 کوتاه‌ترین $s-t$-مسیر با ℓ یال باشد.

اگر $w < \ell$ **آنگاه**

| P_1 **را برگردان**

درغیراینصورت

| P_2 **را برگردان**

فرض کنید P^* یک مجموعه مسیر بهینه مسئله MSE باشد به طوری که از کم‌ترین تعداد یال (یال‌هایی که جریان مثبت دارند) استفاده کرده باشد. فرض کنید D گرافی باشد که از P^* به وجود آمده و m^* تعداد یال‌های D باشد. تعداد یال‌های جواب بهینه را با OPT نمایش می‌دهیم. بدون اینکه کلیت مسئله تغییر کند فرض کنید $OPT \neq 0$ باشد.

لم ۳-۴-۱. *گراف D، یک گراف جهت‌دار بدون‌دور است.*

اثبات. فرض کنید که D دور داشته باشد. ظرفیت هر یالی که در دور قرار دارد، را به مقدار کم‌ترین جریانی که روی یال‌های دور قرار دارد کاهش می‌دهیم. با این کار جریان روی حداقل یکی از این یال‌ها صفر می‌شود یا به‌عبارت‌دیگر یکی از تعداد یال‌های D کم می‌شود بدون اینکه تعداد یال‌های اشتراکی افزایش پیدا کند و این با کمینه بودن یال‌های D در تناقض است. □

لم ۳-۴-۲. *[۲۰] اگر ℓ طول کوتاه‌ترین $s-t$-مسیر باشد، آنگاه* $OPT \geq \frac{k\ell - m^*}{k}$.

اثبات. فرض کنید f یک $s-t$-جریان با مقدار k در D باشد. دراین‌صورت داریم:

$$\sum_{\substack{e\in E\\ f(e)>1}} 1 \leq \sum_{e\in E} \max\{0, f(e)-1\}.$$

سمت چپ تعداد یال‌های اشتراکی را نمایش می‌دهد سمت راست کران بالایی برای تعداد یال‌های اشتراکی است. از طرف دیگر، هر یال اشتراکی حداکثر می‌تواند در k مسیر ظاهر شود، بنابراین داریم:

$$\sum_{e\in E} \min\{0, f(e)-1\} \leq k\, OPT.$$

چون D کمینه است، در نتیجه یالی با جریان صفر ندارد. ما نحوه شمارش سمت چپ نامعادله فوق را تغییر می‌دهیم. یعنی به جای اینکه از هر یال ابتدا یک واحد جریان کم کنیم و بعد آن را در سری جمع بزنیم، ابتدا جریان روی همه یال‌ها را با هم جمع می‌کنیم و سپس به تعداد یال‌های D از مجموع کم می‌کنیم.

$$\sum_{e\in E} \max\{0, f(e)-1\} \geq \sum_{e\in E} f(e) - m^*.$$

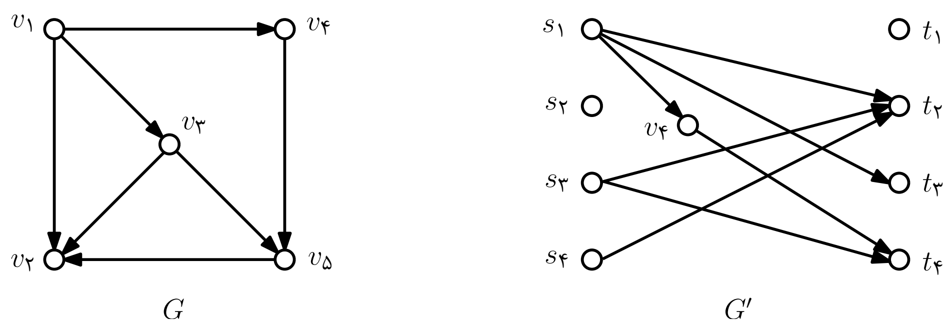
طبق تعریف قبل، m^* تعداد یال‌های D می‌باشد. طول هر $s-t$-مسیر حداقل ℓ است. بنابراین تعداد یال‌هایی که در k مسیر استفاده می‌شوند، حداقل $k\ell$ می‌باشد. با ترکیب نامعادله‌های فوق داریم:

$$k\ell - m^* \leq \sum_{e\in E} f(e) - m^* \leq k\, OPT.$$

□

لم ۳-۴-۳. *[۲۰] فرض کنید $G=(V,E)$ یک گراف جهت‌دار بدون‌دور با n راس باشد به طوری که برای همه راس‌ها، بجز k راس، درجه ورودی و درجه خروجی آن‌ها با هم برابر است. آنگاه G دارای $O(kn\sqrt{n}+k^2)$ یال می‌باشد.*

اثبات. برای هر راس v_i که $\mathrm{outdeg}(v_i) \neq \mathrm{indeg}(v_i)$ باشد، راس v_i را به دو راس s_i و t_i تقسیم می‌کنیم به این صورت که $\mathrm{outdeg}(s_i) = \mathrm{outdeg}(v_i)$، $\mathrm{indege}(t_i) = \mathrm{indeg}(v_i)$ و $\mathrm{indeg}(s_i) = \mathrm{outdeg}(t_i) = ۰$. فرض کنید S و T به ترتیب مجموعه‌هایی شامل همه s_iها و t_iها باشند. شکل ۳-۶ یک مثال از این تبدیل را نمایش می‌دهد. از آنجایی که گراف G جهت‌دار و بدون‌وزن است، بنابراین یک ترتیب توپولوژیکی برای آن وجود دارد. راس‌های $V \backslash (S,T)$ را به صورت توپولوژیکی مرتب می‌کنیم. سپس راس‌ها را به همان ترتیب که مرتب کرده‌ایم به صورت دسته‌های $\sqrt{n}$-تایی تقسیم می‌کنیم (شکل ۳-۷ را ببنید). با این کار، یال‌های گراف G به چهار دسته تبدیل می‌شوند و هر یال جزء یکی از این دسته‌ها است: ۱) یال‌هایی از S به T، ۲) یال‌هایی از S به دسته‌های میانی، ۳) یال‌هایی که دو راس انتهایی آن‌ها در دسته‌های میانی قرار دارد و ۴) یال‌هایی که در داخل هر دسته میانی قرار دارد. در هر دسته میانی حداکثر n یال وجود دارد و در نتیجه در دسته‌های میان حداکثر $n\sqrt{n}$ یال وجود دارد.



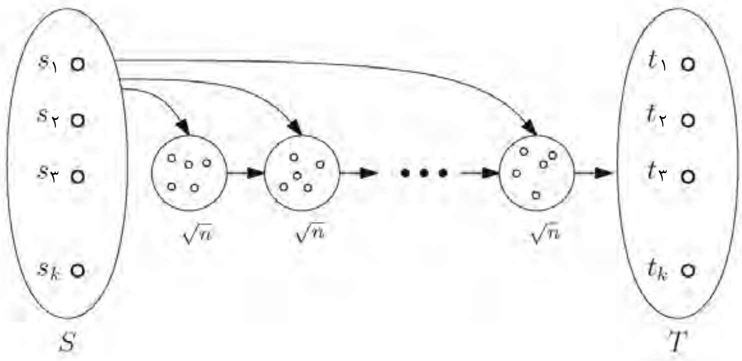
شکل ۳-۶: یک مثال از تبدیل بیان شده در لم ۳-۴-۳ برای گراف جهت‌دار و بدون‌دور G.

در ادامه کران بالایی برای تعداد یال‌هایی که بین دسته‌ها قرار دارند، به‌دست‌می‌آوریم. گراف G' را از روی گراف G بدین صورت می‌سازیم که به جای دسته i، یک راس v'_i و به جای دسته S و T، دو راس S' و T' در G' اضافه می‌کنیم. در صورتی که بین دسته i و راس $s_i(t_i)$ یک یال در G وجود داشت متناظر با آن یک یال بین راس${v'}_i({v'}_j)$ و راس $S'(T')$ در G' اضافه می‌کنیم، با این کار ممکن است یال‌های چندتایی نیز به وجود آید. سپس هر یال $e' = (v'_i, v'_j)$ که $j > i + ۱$ را به دو یال $(v'_i, v'_{i+۱})$ و $(v'_{i+۱}, v'_j)$ تبدیل می‌کنیم، و این روند را آنقدر ادامه می‌دهیم تا تمام یال‌ها به فرمت $(v'_i, v'_{i+۱})$ درآیند.

در هر مرحله برای یال $e' = (v'_i, v'_j)$ که $j > i + ۱$ است، فاصله بین i و j یک واحد کم می‌شود،

بنابراین این روند پایان پذیر است و منجر به افزایش تعداد یال‌های G' می‌شود. با این کار در G' درجه هر راس v'_i برابر است با درجه خروجی راس $v'_{i-۱}$ به علاوه تعداد یال‌هایی که از راس S' به راس v'_i وجود دارند، که این تعداد حداکثر $|S|\sqrt{n}$ می‌باشد. می‌توان نشان داد که درجه خروجی $v'_{\sqrt{n}}$ حداکثر $n|S|$ می‌شود. بنابراین می‌توانیم تعداد یال‌هایی که از S به T وجود دارد را محاسبه کنیم، حداکثر تعداد آن‌ها برابر می‌شود با:

$$O(|S|n\sqrt{n} + |S| \times |T|) = O(kn\sqrt{n} + k^۲).$$



شکل ۳-۷: اثبات قضیه ۳-۴-۳، یال‌های گراف G را چهار دسته تقسیم کردیم.

□

قضیه ۳-۴-۴. *[۲۰] الگوریتم ۳٫۴ یک الگوریتم با عامل تقریب $O(\min(n^{\frac{۲}{۳}}, m^{\frac{۱}{۲}}))$، برای مسئله حداقل یال‌اشتراکی (MSE) است.*

اثبات. فرض کنید $\alpha = \frac{\ell}{OPT}$، عامل تقریب به‌دست‌آمده از طریق کوتاه‌ترین مسیر باشد. بنابراین الگوریتم ۳٫۴، دارای عامل تقریب $\min\{k, \alpha\} \le \min\{\lfloor\frac{k}{۲}\rfloor, \alpha\}$ است. ما دو کران‌بالا برای عامل تقریب الگوریتم به صورت $m^{\frac{۱}{۲}}$، $n^{\frac{۲}{۳}}$ به دست می‌آوریم و از بین آن‌ها کم‌ترین را به عنوان عامل تقریب الگوریتم انتخاب می‌کنیم. برای محاسبه عامل $O(m^{\frac{۱}{۲}})$ دو حالت در نظر می‌گیریم:

حالت اول: فرض کنید $kl \ge ۲m^*$، بر اساس لم ۳-۴-۲ داریم:

$$\alpha \le \frac{k\ell}{k\ell - m^*} \le \frac{۲m^*}{۲m^* - m^*} = ۲$$

در نتیجه $\min\{k,\alpha\} \leq ۲$ می‌باشد.

حالت دوم: فرض کنید $k\ell < ۲m^*$ باشد،

$$k\alpha \leq k\alpha OPT = k\ell < ۲m^* \leq ۲m$$

بنابراین در این حالت $\min\{k,\alpha\} < \sqrt{۲m}$ می‌باشد به عبارت دیگر $\max(\min\{k,\alpha\}) < \sqrt{۲m}$.

در نتیجه $O(\sqrt{m})$ را برای کران‌بالای اول عامل تقریب به‌دست‌آوردیم. برای به‌دست‌آوردن کران‌بالای دوم $O(n^{\frac{۳}{۴}})$، فرض کنید D، گراف جهت‌دار و بدون‌دوری باشد که قبلا نحوه ساخت آن را در بخش ۳-۴ توضیح داده‌ایم. همه راس‌های D، بجز راس‌هایی که راس انتهایی، یال‌های اشتراکی هستند، و همچنین راس‌های s و t دارای درجه ورودی و خروجی یکسان هستند. بر طبق لم ۳-۴-۳ حداکثر تعداد یال‌های D برابر با $O(n\sqrt{n}OPT + OPT^۲)$ است.

اگر $OPT \geq \sqrt{n}$ باشد، دراین‌صورت کوتاه‌ترین مسیر یک جواب $\sqrt{n}$-تقریبی برای مسئله MSE است، زیرا $\ell \leq n$. اگر $OPT < \sqrt{n}$ باشد آنگاه بر اساس نتایج زیر m^* با $O(n\sqrt{n}OPT)$ کران‌دار می‌شود.

$$\begin{aligned} k\ell < ۲m^* < cn\sqrt{n}OPT &\Rightarrow k\alpha OPT < cn\sqrt{n}OPT \\ &\Rightarrow k\alpha < cn^{۳/۲} \\ &\Rightarrow \min\{k,\alpha\} < \sqrt{cn^{۳/۲}}. \end{aligned}$$

با ترکیب نتایج به دست آمده در حالت‌های قبل داریم $\min\{k,\alpha\} = O(\min\{n^{\frac{۳}{۴}}, m^{\frac{۱}{۲}}\})$. بنابراین الگوریتم ۳,۴ جواب‌هایی با عامل تقریب $O(\min\{n^{\frac{۳}{۴}}, m^{\frac{۱}{۲}}\})$ برای مسئله MSE به‌دست‌می‌آورد. □

نتیجه ۳-۴-۵. *درگراف‌هایی‌که $m = O(n)$ است، از قبیل گراف‌های مسطح و گراف‌هایی‌که درجه راس‌های گراف محدود است، عامل تقریب الگوریتم ۳,۴ می‌تواند به $O(\sqrt{n})$ بهبود یابد.*

۳-۵ عدم تقریب پذیری مسئله حداقل آسیب‌پذیری

ایون و همکاران در [۷] ثابت کردند که در مسئله $MECF$ وقتی هزینه‌ها یکسان باشد نمی‌توان الگوریتم تقریبی با عامل $۲^{\log^{۱-\epsilon} n}$ برای هر $\epsilon > ۰$، پیدا کرد مگر اینکه $NP \subseteq DTIME(n^{Polylog\, n})$. در بخش ۲-۵ از مسئله $MECF$ استفاده کردیم و نتایج مشابهی را برای MSE به‌دست‌آمد و نشان داده شد که برای

MSE نمی‌توان یک الگوریتم تقریبی با عامل $2^{\log^{1-\epsilon} n}$ برای هر $\epsilon > 0$، پیدا کرد. مسئله MSE یک نمونه از مسئله r–آسیب‌پذیری است. با استفاده از یک استدلال مشابه عدم تقریب‌پذیری برای مسئله r–آسیب‌پذیر به ازای هر $r > 0$، را نشان می‌دهیم.

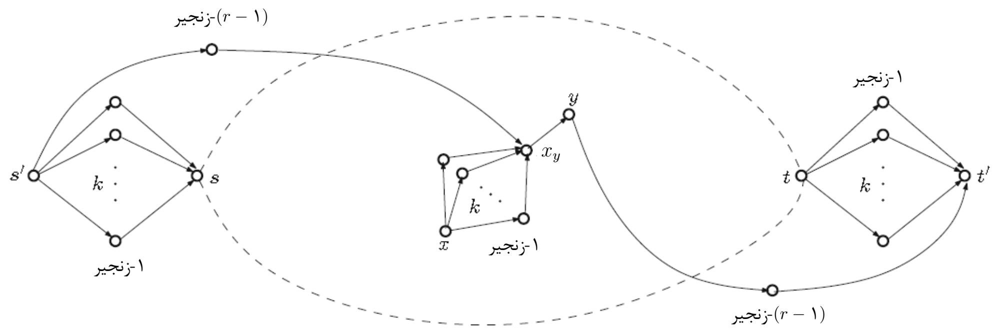
قضیه ۳-۵-۱. *[۲۰] مسئله r–آسیب‌پذیری به‌ازای‌هر $\epsilon > 0$ و به‌ازای‌هر $r > 0$ تقریب $2^{\log^{1-\epsilon} n}$ را نمی‌پذیرد، مگر این‌که $NP \subseteq DTIME(n^{Polylog\, n})$.*

اثبات. برای اثبات قضیه از مسئله MSE استفاده می‌کنیم و مسئله MSE را به مسئله r–آسیب‌پذیر کاهش می‌دهیم. فرض کنید (G,k) یک نمونه از مسئله MSE باشد که $k \leq |E|$. برای یک مقدار مشخص $r > 1$، ما (G,k) را به یک نمونه (G',k') از مسئله حداقل r–آسیب‌پذیری کاهش می‌دهیم. قرار دهید $k' = k+(r-1)\times|E|$. مسیرهایی به طول ۲ به طوری که هر یال ظرفیت u دارد را u**–زنجیر**[1] می‌نامیم. در ادامه G' را بدین صورت می‌سازیم: ابتدا راس‌های s' و t' را اضافه می‌کنیم، سپس به‌ازای‌هر یال $(x,y) \in E$، یک راس جدید x_y را به G' اضافه می‌کنیم. به راس x_y راس میانی[2] نیز می‌گوییم. سپس یک یال از x_y به y با ظرفیت $+\infty$ و k تا 1–زنجیر از x به x_y اضافه می‌کنیم. علاوه بر آن به‌ازای‌هر یال $(x,y) \in E$، یک $(r-1)$–زنجیر از s' به x_y و یک $(r-1)$–زنجیر از y به t' اضافه می‌کنیم (شکل ۳-۸). در پایان k تا ۱–زنجیر از s به s' و k تا ۱–زنجیر از t به t' اضافه می‌کنیم.

در ادامه ثابت می‌کنیم که هر جواب بهینه برای MSE در (G,k) یک جواب بهینه برای مسئله حداقل r–آسیب‌پذیری در (G',k') است و بالعکس.

فرض کنید P مجموعه‌ای از k تا $s-t$–مسیر با μ یال اشتراکی در G باشد. ما یک مجموعه معادل Q شامل k' تا مسیر از s' به t' در G' با r تا یال آسیب‌پذیر پیدا می‌کنیم. برای هر یال (x,y) در G، $r-1$ مسیر در G' به دست می‌آوریم. مسیرها شامل سه بخش است: بخش اول زنجیر $r-1$–تایی از s' به x_y، بخش دوم شامل یال (x_y,y) و بخش سوم شامل زنجیر $r-1$–تایی از y به t' می‌باشند. سپس k تا مسیر از s' به t' در نظر می‌گیریم. این مسیرها نیز شامل سه بخش می‌باشند. بخش اول k تا مسیر شامل زنجیره‌های ۱–تایی از s به s'، بخش دوم شامل k مسیر از s به t است که مشابه با مسیرهایی که در P قرار دارند انتخاب

[1] u-Chain
[2] Mid vertex



شکل ۳-۸: کاهش مسئله MSE به مسئله حداقل r-آسیب‌پذیری با هزینه یال‌های یکسان. هر یال (x,y) در G با kتا ۱-زنجیر از x به x_y و یال (x_y,y) جابجا می‌شود.

می‌شوند. به‌عبارت‌دیگر در صورتی که یال (x,y) در مسیرهای P ظاهر شده باشد، آنگاه یک مسیر از x به y شامل یک زنجیر از x به x_y و یال (x_y,y) در نظر می‌گیریم و بخش سوم مسیر شامل k مسیر از t به t' است. با این کار ما $k' = (r+۱) \times |E| + k$ مسیر از s' به t' در G' پیدا می‌کنیم. در مورد یال‌های r-آسیب‌پذیر در G'، مشاهده می‌کنیم که فقط یال‌هایی که از راس‌های میانی خارج می‌شوند می‌توانند r-آسیب‌پذیر باشند، زیرا ظرفیت دیگر یال‌ها کمتر از r می‌باشد و فقط این نوع از یال‌ها هستند که ظرفیت $+\infty$ دارند و بیش از r مسیر می‌تواند از آن‌ها عبور کند. $r-۱$ تا از مسیرها از s' به x_y و حداقل دو مسیر دیگر از x به x_y می‌آید و باعث r-آسیب‌پذیر شدن یال (x_y,y) می‌شود. بنابراین تعداد یال‌های اشتراکی در P برابر با تعداد یال‌های r-آسیب‌پذیر Q است.

در ادامه عکس رابطه بالا را ثابت می‌کنیم. فرض کنید Q شامل مجموعه k' تا مسیر از s' به t' در G با λ یال r-آسیب‌پذیر باشد. این مسیرها را می‌توان با یک جریان با اندازه k' از s' به t' نمایش داد، که از λ تا آن‌ها جریان بیش از r عبور می‌کند. دقت کنید که مجموع یال‌های خروجی از s' برابر k' است، و این برای یال‌های ورودی t' نیز برقرار است و از ظرفیت این یال‌ها به طور کامل استفاده شده است. حال برای اینکه مجموعه k مسیر در G را از G' به‌دست‌آوریم، جریان روی همه یال‌هایی که از راس میانی x_y به y می‌باشد را به اندازه $r-۱$ واحد کاهش می‌دهیم. سپس $r-۱$ جریان را از مسیرهای s' به x_y و از y به t' حذف می‌کنیم. بعد از این که روند فوق را برای همه راس‌های میانی انجام دادیم به اندازه $(r-۱) \times |E|$ واحد جریان از s' به t' کم می‌شود و دقیقا k واحد جریان از s' به t' باقی می‌ماند. این جریان متناظر است با یک جریان با

اندازه k از s به t در G است. اگر از یک یال G جریان بیش از یک واحد عبور کند، بدین معنی است که در G' یال متناظر با آن جریان بیش از r را از خود عبور می‌دهد. بنابراین تعداد یال‌های r–آسیب‌پذیر در Q با تعداد یال‌های اشتراکی متناظر آن در G برابر است.

تعداد یال‌ها در G' حداکثر $O(k|E|)$ برابر تعداد یال‌های G است. از آنجایی که در MSE وقتی که $k > |E|$ است مسئله در زمان چندجمله‌ای قابل حل است (قضیه ۲–۵–۲)، ما می‌توانیم فرض کنیم $k = O(E)$، در نتیجه روند کاهش در زمان چندجمله‌ای قابل انجام است. بنابراین اگر بتوان برای مسئله حداقل r–آسیب‌پذیری یک الگوریتم تقریبی با عامل کمتر از $۲^{\log^{۱-\epsilon} n}$ به ازای هر $\epsilon > ۰$ به‌دست‌آورد، این الگوریتم تقریبی برای MSE نیز کار می‌کند و این با عدم تقریب پذیری MSE که در قضیه ۲–۵ آمده در تناقض است. □

در قضیه فوق عدم تقریب‌پذیری مسئله r–آسیب‌پذیری به‌ازای‌هر $r > ۰$ نشان داده شد. با این وجود در این فصل یک الگوریتم تقریبی با عامل $\lfloor \frac{k}{r+۱} \rfloor$ برای مسئله ارائه شد و همچنین بیان شد که اگر k ثابت باشد، برای مسئله آسیب‌پذیری راه‌حل چندجمله‌ای وجود دارد.

در این فصل مسئله حداقل آسیب‌پذیری بررسی شد، این مسئله تعمیم یافته مسئله حداقل یال‌اشتراکی می‌باشد. نشان داده شد که برای مسئله حداقل آسیب‌پذیری یک الگوریتم تقریبی وجود دارد و در صورتی که پارامترهای مسئله ثابت باشند، می‌توان مسئله را در زمان چندجمله‌ای حل کرد. علاوه‌براین در این فصل مسئله حداقل یال‌اشتراکی نیز بررسی شد و نشان داده شد که یک الگوریتم زیرخطی برای آن وجود دارد و اگر گراف ورودی تنک باشد، عامل تقریب برابر با $O(\sqrt{n})$ می‌شود.

نتیجه‌گیری

یکی از مسائل کاربردی در گراف، مسئله پیدا کردن k تا مسیر یال‌مجزا در گراف $G = (V, E)$ از راس s به راس t می‌باشد. k تا مسیر را یال‌مجزا گویند، اگر هر یال $e \in E$ حداکثر یک‌بار در مسیرها استفاده شده باشد. تعداد مسیرهای یال‌مجزا در گراف محدود است، فرض کنید تعداد مسیرهای یال‌مجزا در G، k' است. در صورتی که $k > k'$ باشد، بدیهی است که پیدا کردن k تا مسیر یال‌مجزا در G ناممکن است، یک راه‌حل جایگزین پیدا کردن k تا مسیر با حداقل یال‌اشتراکی می‌باشد. به یالی که بیش از یک‌بار در مسیرها ظاهر شود، یال‌اشتراکی می‌گویند. در این پایان‌نامه مسئله حداقل یال‌اشتراکی (MSE) را بررسی کردیم. در مسئله حداقل یال‌اشتراکی هدف پیدا کردن k تا مسیر در گراف G از راس s به راس t با کم‌ترین یال‌اشتراکی می‌باشد، این مسئله در طراحی بعضی از شبکه‌های ارتباطی و حمل‌ونقل کاربرد دارد. زمانی که $k < k'$ است، هر راه‌حل مسئله مسیرهای یال‌مجزا، یک راه‌حل برای مسئله MSE نیز می‌باشد و این یک جواب بهینه است و هیچ یال‌اشتراکی در آن وجود ندارد، و زمانی که $k > m$ باشد، m تعداد یال‌های گراف G است، آنگاه حداقل تعداد یال‌های اشتراکی برای پیدا کردن k تا مسیر از s به t برابر با اندازه کوتاه‌ترین مسیر از s به t می‌باشد و همه k مسیر راه‌حل برابر با کوتاه‌ترین مسیر از s به t هستند.

در فصل ۲، ثابت شد که مسئله MSE، NP-سخت است و نشان داده شد که هیچ الگوریتم تقریبی با عامل $2^{\log^{1-\epsilon} n}$، به‌ازای‌هر $\epsilon > 0$، برای مسئله MSE وجود ندارد، مگر این که $NP \subset DTIME(n^{polygon\, n})$ باشد. با این حال دو الگوریتم تقریبی با عامل‌های k و $k - 1$ برای مسئله ارائه شد. علاوه‌براین یک الگوریتم با عامل تقریب زیرخطی $O(n^{\frac{3}{4}})$، برای مسئله معرفی شد که در گراف‌های تنک از قبیل گراف‌های مسطح که $m = O(n)$ است، عامل تقریب می‌تواند به $O(\sqrt{n})$ نیز بهبود یابد. در صورتی که گراف ورودی سری-موازی باشد، آنگاه یک الگوریتم $(1 + \epsilon)$-تقریبی با زمان $O(m^3(1 + 1/\epsilon) \log k)$ برای مسئله وجود دارد، همچنین برای مقادیر ثابت و کوچک k، الگوریتم چندجمله‌ای

با زمان $O(m^{2k}n^3)$ برای مسئله MSE ارائه شد.

در فصل ۳، مسئله حداقل آسیب‌پذیری بررسی شد، هدف از مسئله حداقل آسیب‌پذیری پیدا کردن k مسیر از راس s به راس t در G، با کم‌ترین یال آسیب‌پذیر می‌باشد، به یالی که بیش از r بار در مسیرها ظاهر شود، یال آسیب‌پذیر می‌گویند. r یک عدد ثابت است که جزء ورودی‌های مسئله حداقل آسیب‌پذیری می‌باشد و هرگاه $r = 1$ باشد، مسئله حداقل آسیب‌پذیری به مسئله حداقل یال‌اشتراکی تبدیل می‌شود. پیدا کردن k مسیر از s به t به گونه‌ای که هیچ یال ۱-آسیب‌پذیری نداشته باشد، برابر با مسئله مسیرهای یال‌مجزا است. برای مسئله حداقل آسیب‌پذیری یک الگوریتم $\lfloor \frac{k}{r+1} \rfloor$- تقریبی ارائه شد که این منجر به معرفی یک الگوریتم $\lfloor \frac{k}{2} \rfloor$- تقریبی برای مسئله حداقل یال‌اشتراکی می‌شود. الگوریتم $\lfloor \frac{k}{2} \rfloor$- تقریبی بهترین الگوریتم تقریبی است که برای مسئله حداقل اشتراکی ارائه شده است، اکنون این سوال مطرح است که آیا می‌توان الگوریتمی با عامل تقریب بهتر برای مسئله ارائه کرد؟

نسخه تصمیم‌گیری مسئله MSE به این صورت تعریف می‌شود: سه‌تایی $\langle G, k, h\rangle$، که G یک گراف بدون‌جهت با دو راس s و t و $k, h \in N$ دو عدد می‌باشند، را در نظر بگیرید. آیا مجموعه P شامل k مسیر از s به t وجود دارد به طوری که تعداد یال‌های اشتراکی مسیرهایی که در P قرار دارند، حداکثر h باشد؟ تیل فبروری[1] و همکاران در سال ۲۰۱۶، نسخه تصمیم‌گیری مسئله MSE را در بررسی کرده‌اند و نشان داده‌اند که اگر گراف ورودی بدون‌جهت باشد، مسئله تصمیم‌گیری MSE، یک مسئله NP-سخت می‌باشد و اگر k و h ثابت باشند، یک الگوریتم با زمان $O((k-1)^h(n+m)^2)$ برای آن وجود دارد. همچنین برای بعضی از گراف‌های خاص مثل گراف‌های سری-موازی، الگوریتم چندجمله‌ای برای مسئله ارائه شده است. علاوه‌براین آنها مسئله را در گراف‌های مسطح نیز بررسی کرده‌اند و ثابت کرده‌اند که مسئله تصمیم‌گیری برای گراف‌های بدون‌جهت و مسطح که درجه رئوس آن حداکثر ۴ باشد و برای گراف‌های جهت‌دار و مسطح که درجه ورودی و خروجی رئوس آن حداکثر ۳ باشد، یک مسئله NP-سخت است. اکنون این سوال مطرح است که آیا مسئله تصمیم‌گیری MSE در گراف‌های مسطح که حداکثر درجه ۳ دارند نیز NP-سخت است؟ همچنین اگر مقدار k و h ثابت باشند، آیا می‌توان الگوریتم‌های بهتری برای گراف‌های مسطح ارائه کرد؟

[1] Till February

منابع و مآخذ

[1] Ahuja, R., Goldberg, A., Orlin, J., and Tarjan, R. Finding minimum-cost flows by double scaling. *Mathematical Programming 53*, 1-3 (1992), 243–266.

[2] Carr, R. D., Fleischer, L. K., Leung, V. J., and Phillips, C. A. Strengthening integrality gaps for capacitated network design and covering problems. in *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms* (Philadelphia, PA, USA, 2000), SODA '00, Society for Industrial and Applied Mathematics, pp. 106–115.

[3] Charikar, M., Hajiaghayi, M., and Karloff, H. *Improved Approximation Algorithms for Label Cover Problems*. Springer Berlin Heidelberg, 2009.

[4] Colin De Verdière, E., and Schrijver, A. Shortest Vertex-Disjoint Two-Face Paths in Planar Graphs. in *STACS 2008* (Bordeaux, France, Feb 2008), P. W. Susanne Albers, ed. , IBFI Schloss Dagstuhl, pp. 181–192.

[5] Cormen, T. H., Stein, C., Rivest, R. L., and Leiserson, C. E. *Introduction to Algorithms*, 3rd ed. . The MIT Press, 2009.

[6] Dinneen, M. J., Kim, Y., and Nicolescu, R. Edge- and node-disjoint paths in P systems. in *Proceedings Fourth Workshop on Membrane Computing and Biologically Inspired Process Calculi, MeCBIC 2010, Jena, Germany, 23 August 2010.* (Feb 2010), pp. 121–141.

[7] Even, G., Kortsarz, G., and Slany, W. On network design problems: Fixed cost flows and the covering steiner problem. *ACM Trans. Algorithms 1*, 1 (July 2005), 74–101.

[8] Goldberg, A. V., and Rao, S. Beyond the flow decomposition barrier. *J. ACM 45*, 5 (1998), 783–797.

[9] Goldberg, A. V., and Tarjan, R. E. A new approach to the maximum-flow problem. *J. ACM 35*, 4 (Oct. 1988), 921–940.

[10] Itai, A., Perl, Y., and Shiloach, Y. The complexity of finding maximum disjoint paths with length constraints. *Networks 12*, 3 (1982), 277–286.

[11] Karp, R. M. Reducibility among combinatorial problems. in *Complexity of Computer Computations* (1972), pp. 85–103.

[12] Karp, R. M. On the computational complexity of combinatorial problems. *Networks* (June 1975), 45–65.

[13] Kobayashi, Y., and Sommer, C. On shortest disjoint paths in planar graphs. *Discrete Optimization 7*, 4 (2010), 234–245.

[14] Krumke, S., Noltemeier, H., Schwarz, S., Wirth, H.-C., and Ravi, R. Flow improvement and network flows with fixed costs. in *Operations Research Proceedings 1998* (1999), P. Kall and H.-J. Lüthi, eds. , volume 1998 of *Operations Research Proceedings 1998*, Springer Berlin Heidelberg, pp. 158–167.

[15] Lapaugh, A. S., and Rivest, R. L. The subgraph homeomorphism problem. *Journal of Computer and System Sciences 20*, 2 (1980), 133–149.

[16] Lester Randolph Ford, J., and Fulkerson, D. R. Maximal flow through a network. *Canadian Journal of Mathematics* (1956), 399–404.

[17] Li, C.-L., McCormick, S., and Simchi-Levi, D. The complexity of finding two disjoint paths with min-max objective function. *Discrete Applied Mathematics 26*, 1 (1990), 105–115.

[18] Li, C.-L., Thomas McCormick, S., and Simchi-Levi, D. Finding disjoint paths with different path-costs: Complexity and algorithms. *Networks 22*, 7 (1992), 653–667.

[19] Linz, P. *An Introduction to Formal Languages and Automata, Fifth Edition*, 5th ed. . Jones and Bartlett Publishers, Inc., USA, 2011.

[20] M. T. Omran, J.-R. S., and Zarrabi-Zadeh, H. Finding paths with minimum shared edges. *Computing and Combinatorics* (February 2011), 567–578.

[21] Omran, M. T., Sack, J.-R., and Zarrabi-Zadeh, H. Finding paths with minimum shared edges. *J. Comb. Optim. 26*, 4 (Nov. 2013), 709–722.

[22] robertson, N., and Seymour, P. the disjoint path problem. *Combinatorial Theory* (1995), 65–110.

[23] Schrijver, A. Finding k disjoint paths in a directed planar graph. *SIAM J. Comput. 23*, 4 (1994), 780–788.

[24] Seymour, P. D. Disjoint paths in graphs. *Discrete Mathematics* (1980), 293–309.

[25] Thomassen, C. 2-linked graphs. *European Journal of Combinatorics 1*, 4 (1980), 371–378.

[26] Vazirani, V. V. *Approximation Algorithms*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.

[27] Xu, D., Chen, Y., Xiong, Y., Qiao, C., and He, X. On the complexity of and algorithms for finding the shortest path with a disjoint counterpart. *IEEE/ACM Trans. Netw. 14*, 1 (Feb. 2006), 147–158.

[28] Yang, B., Yang, M., Wang, J., and Zheng, S. Minimum cost paths subject to minimum vulnerability for reliable communications. in *Parallel Architectures,Algorithms and Networks, 2005. ISPAN 2005. Proceedings. 8th International Symposium on* (Dec 2005), pp. 25–31.

Abstract

Finding k edge disjoint paths between two given vertices s and t in a graph $G = (V, E)$ is a known problem in graph theory. The paths is called edge-disjoint if each edge $e \in E$ appears at most once in the paths. Each graph can only have a limited number of edge disjoint paths. Suppose that the maximum number of edge disjoint paths in G is k'. When $k > k'$, finding k edge disjoint paths in G is obviously impossible. An alternative problem is to find k paths with minimum number of shared edges; any edge that appears in more than one of the k paths is called a shared edge. In this thesis, we study the minimum shared edge problem, whose goal is to find k paths from s to t with minimum number of shared edges. It has been proven that minimum shared edge problem is an NP-hard problem, and there exists no polynomial time approximation algorithm with factor $2^{\log^{1-\epsilon} n}$ (where $\epsilon > 0$) that can solve it. The best approximation algorithm developed for this problem has an approximation factor of $\lfloor \frac{k}{2} \rfloor$, and only when k is a constant this problem can be solved by a polynomial time algorithm.

Another problem that will be discussed in this thesis is the problem of minimum vulnerability. The objective of minimum vulnerability problem is to find k paths from s to t in G, using minimum number of vulnerable edges. We call an edge vulnerable when it appears more than r times in the paths; r is a constant defined as an input of minimum vulnerability problem. Obviously for $r = 1$, the minimum vulnerability problem turns into the minimum shared edge problem.

Yazd University
Department of Mathematics

Thesis submitted
for the degree of M.Sc.

Title:

On the problem of k-paths with minimum shared edges

Supervisor: Dr. Mohammad Farshi

Advisor: Dr. Mahdieh Hasheminezhad

By: Alireza Jalayegh

March 2016